International Journal of Advanced Research in

Computer Science Engineering and Information Technology

Volume: 5 Issue: 1 25-Mar-2015,ISSN_NO: 2321-3337

RELATED PACKET PADDING FOR ANONYMOUS WEB BROWSING IN MOBILE DEVICES AGAINST TRAFFIC ANALYSIS ATTACK

[1]B.PAVITHRA, [2]K.KANYA

[1]M.TECH Student, [2]Assistant Professor

[1,2]Dr.M.G.R EDUCATIONAL AND RESEARCH INSTITUTE UNIVERSITY
Chennai, TamilNadu, India.

ABSTRACT *– Anonymous web browsing is becoming more popular to meet web privacy protection. To meet anonymity, we propose related packet padding strategy in which web page related to user request is selected as cover page based on the popularity for anonymous web browsing systems. Earlier predicted packet strategy was used for anonymous web browsing systems in which web page based on popularity is selected as cover page. In related packet padding strategy the probability of reusing cover page is relatively increased.*

Usage of mobile devices for web browsing is currently being increased. One of the major disadvantages of web browsing using mobile device is the limited availability of computational power and memory space. The increase in the data due to the cover page is offset by the improvement in the caching technique. We propose LFU-LS (Least frequently used –Local statistics) algorithm to use the small size cache memory devices effectively for storing web pages by calculating Local statistics for a file. If a new file is downloaded into cache initial read hits is calculated by using statistics from the server. When a user reads a cached file, the local read-hits-counter is increased. To prevent ageing of a file in cache, a threshold value is assigned. Whenever read-hits-counter attains a threshold value, that value is halved to prevent the aging file to reside in the cache memory. By using LFU-LS algorithm we can efficiently utilize the memory space.

I. INTRODUCTION

Concerns about privacy and security have received greater attention with the rapid growth and public acceptance of the Internet, Which is being mostly used through mobile devices. Although encryption can hide the web pages from the eavesdropper, encryption does not hide all the information. Eavesdropping on an encrypted communication can make a person come to know who is communicating, how many data is being transferred and which web site is browsed by hacking the information relevant to the attacked web pages such as packet size, HTTP object count and their sizes. To provide more privacy for the web pages from attackers anonymous web browsing is becoming a necessary and legitimate topic.

Anonymous communication systems can be classified into two types. High-latency anonymity systems also called as message based anonymity applications have the ability of providing strong anonymity, but are only applicable for non-interacting applications [6]. Low-latency anonymity systems can provide better performance and applicable for real time applications.

The process of monitoring and understanding the nature and behaviour of traffic rather than its contents called as traffic analysis. Attacker can identify the web pages by referring their fingerprints [1]. Anonymous

web browsing means that providing the security against traffic analysis. Anonymity refers to disguising the identity of the web page. Anonymity of the web page is calculated in terms of CCA [19].

There are a few implementation mechanisms for current flow based anonymous systems such as Onion routing [11], Tor system [14], and Crowds system [12]. Research on flow-based anonymity applications has recently received great attention in order to preserve anonymity in low-latency applications, including Web browsing and peer-to-peer file sharing [17]. Most of them use the dummy packet padding strategy. But dummy packet padding strategy causes delay and bandwidth waste. To mitigate delay and bandwidth waste predicted packet padding strategy [19] had been developed.

In predicted packet padding, the packet which has the high popularity in server side is predicted and then padded with the intended web page as response [19].This method disguises the fingerprints of web pages at the server side by injecting predicted web pages, which users are going to download, as cover traffic. This cover may be used by the users in future. But the probability of using this cover traffic increases only when the number of cover pages increases. It leads to more memory usage. But currently more users use the mobile devices for web browsing which has low memory in size. Thus in mobile devices predicted packet padding becomes dummy packet padding. The basic idea is to cache recent mostly requested pages at the server so that they do not have to be fetched again.

In this paper, related packet padding is used in order to reduce the disadvantages of predicted packet padding in the concept mobile phone browsing. Huberman et al. [2] found that the average browsing length for a browsing session was 15 pages at the end of the 1990s. Xie and Yu's recent investigation demonstrated that the average browsing length for a session is around 30 [22].

The Overall goals of this paper are listed as follows:

1) We introduce the related page padding method to improve the browsing speed. This method reduces the required storage area for cover traffic because more relevant and popularly used web pages only selected as cover pages.
2) We propose the LFU-LS algorithm for web caching. This algorithm is used to improve the cache efficiency.
3) We use the page validation technique to use the updated static web pages without any conflict.

The rest of the paper constructed as follows: The related work is discussed in Section II. We explain about the problem in the existing paper and the necessary of implementing the solution in section III. Modelling of the proposed system is being talked in section IV. In section V, effectiveness of the system is analysed. The implementation details of the related page padding are concluded in section VI. Implementation and performance analysis is discussed in section VII. Further discussion and future work are stated in section VIII and IX.

2, RELATED WORK

i. Anonymous Web browsing

We can understand by studying the HTTP protocol that when the client sends an http request to the server, the server will send the HTML text as response. The HTML text may contain the references of related web objects such as images, audio, video. This information composes fingerprints of web pages. But

an attacker can easily find out the packet heads, in which traffic information is available.

Anonymity is measured in terms of entropy. Shannon proposed some conditions for perfect secrecy [4], and stated that a system is in the best anonymity, when its entropy is in maximum. But to overcome some problems in Shannon's statement Sarjentov et al. [2] investigated an information theoretical measure of anonymity which is based on the users sending and receiving messages.

Already we know encryption is not the effective method against traffic analysis. Sin et al. assessed encrypted network traffic using the HTTP object number and size, and their tests indicated that these two features are sufficient to identify a significant fraction of the world wide websites. Coull *et al.* [17] evaluated the strength of the anonymization methodology in terms of preventing the assembly of behavioural profiles and concluded that anonymization offers less privacy to web browsing traffic than what people expected. Liberatore and Levine's [13] experiments showed that the above discussed method can identify the source with an accuracy of up to 90%. Some solutions have been suggested for anonymous web browsing against traffic analysis.

Hintz [1] found a method of injecting cover traffic into the intended traffic to disguise the actual fingerprint of the intended web page. So we can create a fake connection against fingerprint attacks. This methods consisted packet padding and packet splitting. Many suitable methods were employed to select the cover traffic. This method is only focused on the dummy packet padding.

Time consumption and bandwidth were the major issues at that time. Wang *et al.* [22] proposed the minimizing sending rate for a given set of flows by controlling the generation rate of the covering traffic; therefore, the traffic pattern of the communication channel is the same all the time. As a result, full anonymity is obtained according to Shannon's perfect secrecy theory.

But the delay is caused while adding dummy packets into communication channels and proposed transmission schedules on relay nodes to maximize network throughput to obtain a desired level of anonymity. Zuyuan et al. [25] recently introduced game theory into the optimization of communication anonymization in wireless networks. To overcome this problem Shui Yu et al. [20] proposed predicted packet padding in which the packets which have more popularity were added to the intended traffic. The popularity if the web pages are calculated based on some constraints. For example, if the requested pages have a lot of links to some ―important‖ page, that page has a higher probability of being the next one requested. So this page has the high popularity. This method requires more cache memory size. In small cache memory devices this strategy become as dummy packet padding.

ii. Web Perfecting Strategies

Web caching and prefetching is the better solution in case of bandwidth waste. Web prefetching is the process of fetching web pages from the server before requesting a web page. Teng *et al.* [23] argued that there must be elaborate coordination between client side caching and prefetching and formulated a normalized profit function to evaluate the profit from caching an object. The proposed function integrated a number of factors, such as object size, fetching cost, reference rate, an invalidation cost, and invalidation frequency. Their event-driven simulations demonstrated that the proposed method performed well.

The web pages are grouped together by referring the page ranking approach. First the related web pages are collected as a cluster. This approach uses the link structure of a requested page to determine the most important‖ linked pages and to identify the page(s) to be prefetched.Breslau *et al.* [11] analysed web accessing behaviour and found that it followed a Zipf-like distribution. After receiving the web request from

the client the intended cluster is found. Suppose there are *n* web pages in total in the cluster, and they are sorted by the rank as *w1, w2 . . . , wn*. This rank value is high for the web page used mostly and low for the web pages used less. Let *p(wr)* be the access probability of the *r*th web page, then

$$P(x_r) = \Omega / i^{\alpha z} \qquad (1)$$

Where α_Z is the Zipf index, which varies from trace to trace. When $\alpha z = 1$, (1) becomes Zipf distribution.

A more general form of the distribution is called the Zipf-Mandelbrot distribution [29], which is defined as follows:

$$P(x_i) = \Omega / (i+q)^{\alpha z} \qquad (2)$$

Where *q* is called as plateau factor, which makes the probability of the highest ranked objects flat.

$$\text{Since } \Sigma_{in=1} p\{x_i) = 1, \quad \Omega = (\Sigma_{in=1} (1)/((i+q)_{\alpha z}))_{-1}.$$

The zipf –Mandelbrot distribution becomes the Zipf distribution when q=0. If there have been *r* web pages in the cache, then the hit ratio *h(r)* for the next request of for web page *i* is as follows:

$$h(r) = \Sigma_{in=1} p(x_i)[1-(1-p(x_i))_r] \quad (3)$$

The distribution of the size of web object is composed of two distributions: a log normal distribution for the body up to a transition point, and a Pareto distribution for the tail. This is referred as double Pareto distribution [25].However, the difference between the double Pareto distribution and the Pareto distribution is at the tail part, with the majority of the two distributions similar to each other.

3, PROBLEM STATEMENT

We discuss the problem as follows: An attacker knows that the monitored user accesses a web site, but wants to know which page is accessed. In order to provide user's privacy the following actions will be taken. First, the packet padding is used to disguise the fingerprint of the intended web page. Finally, the packets are encrypted to provide more security. As a user, attacker also has the full knowledge of the web site in the form of encrypted or plaintext. The attacker expects to use traffic analysis methods to identify the user browsed web page.

For a given web site, there are *n* number of web pages, denoted as $\{x_1, x_2, \ldots, x_n\}$, x_i is the *i*th ranked web page. For web page x_i of the web site, we consider that it contains *m* (m>0) web objects, $\{x_i^1, x_i^2, \ldots, x_i^m\}$. x_i^k is the *k*th web object of the *i*th web page. We denote the size of web object x_i^k as $|x_i^k|$.

Fingerprint of a web page is denoted as a set, $\{t_i^1, t_i^2, \ldots, t_i^m\}$. Each element of the set, t_i^k ($1 \le i \le n$, $1 \le k \le m$), is defined as follows:

$$T_i^k = |x_i^k| / \sum_j^m{}_{=1} |x_i^j| \qquad (4)$$

Obviously we know that $\Sigma_{in=1} P(x_i) = 1$. And the attacker also knows this along with the full information of the fingerprint of every web page of the web site. Attacker is able to monitor user's local network and can observe his on-going browsing in terms of the number of packets for each web object

$$\tau = \{\tau 1, \tau 2, \tau 3, \ldots\}$$

Where τ_i is the number of packets of the encrypted *i*th web object that user has downloaded.From [14] and [29] we can clearly know that encryption will not hide the fingerprint information. So packet padding Method has been introduced. The intended page is padded with the requested web page. So, all the packets will be viewed as same size for the attacker's view. The traditional method is to achieve this is dully packet padding, which causes delay and bandwidth waste. To overcome these drawbacks predicted packet padding method was used, in which the packets with high popularity is padded with the intended web page to use in future and to reducing delay. But by using predicted padding strategy in finite cache size memory devices for web browsing such as smartphones, tablets it becomes as dummy packet padding.

4,RELATED PAGE PADDING MODELLING

From the above discussion, we can clear that encryption is not suit against traffic analysis and also predicted padding does not fit for small size cache devices. We extend the predicted packet padding method to predict the web page related to the requested web page and also combine this with the encryption method to provide more security.

The proposed packet padding system model is shown in Fig.1. At the server side, the related pages are predicted and then the cover page Y is selected for packet padding with the intended page X. The output of padding mechanism Z will be encrypted and forwarded to the client through the internet or related anonymous networks. At the client side the encrypted packet Z will be decrypted and the intended page X is displayed by the web browser and the cover page Y is sent to the cache directly.

In general, When the user submits an encrypted HTTP request for web age w_i to the web server, the server will return the intended page $x_i = \{x_i^1, x_{i2}, \ldots, x_i^m\}$.The size of the web object x_i^k ($1 \le k \le m$) *is* $|x_i^k|$ in terms of packets.In order to disguise the fingerprint of page x_i, we need to provide fake fingerprint for that web page for the attacker by injecting cover traffic $y_i=\{y_i^1, y_i^2,\ldots,y_i^{m'}\}$, where y_i^k ($1 \le k \le m', m \le m'$) is the cover traffic for x_i^k into the intended traffic w_i to obtain a covered output $z_i = \{z_i^1, z_i^2,\ldots, z_i^{m'}\}$.
As a result, the observation, $\tau = \{\tau 1, \tau 2, \tau 3, \ldots\}$,which attacker obtains is the encrypted version of $\{z_i^1, z_i^2, \ldots, z_i^{m'}\}$.

$$Z = X \oplus Y. \qquad (5)$$

The anonymization operation at the server side is as follows: We can further discuss this for a small object as follows:

$$|z_i^k| = |x_i^k| + |y_i^k|, \; k = 1, 2, 3 \ldots m' \qquad (6)$$

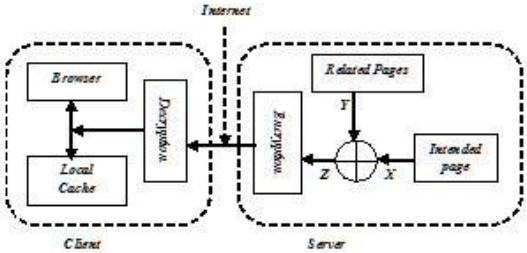

Fig.1. Anonymous web browsing model with Related Packet Padding.

We have to make every browsing session the same from attacker's point of view. This means that every browsing session has the same number of objects and every web object has the same size, namely, the following two equations hold:

$|z_i| = |z_j|$ i, j =1, 2, 3, ... $i \neq j$

$|z_i^{k1}| = |z_i{}^{k2}|$ *k1, k2 =1, 2, 3,…* *k1≠k2*

Instead of predicted packet padding we use related web pages that users are going to use in near future with high probability as the cover traffic. At user side, after the decryption, the intended traffic goes to the web browser, and the cover traffic goes to the cache.

Every user request will be checked with the cache, if it is found in the cache, then no need to download it from the server. Thus the request will not be sent to the server. As a result, anonymity will be reached.

Obviously we can realize that the probability of using predicted pages by the user cannot be 100%. So, some predicted pages which were stored in the cache may never be used. It leads to waste of memory and also becomes like dummy packet padding. In order to measure the anonymity the following metric is used:

Cost Coefficient of Anonymization (CCA):

Let function *C(X)* represents the costs for network traffic *X* in terms of number of packets. The *CCA* can be defined as

$$\beta = (C(Y|W) + C(W)) / C(W) \qquad (7)$$

Where *C(Y|W)* denotes the cost of traffic *Y*, which is used to cover an intended traffic *W*.

Suppose a user needs to browse *k* web pages, then the *CCA* will be defined as follows:

$$\beta = \sum_{i=1}^{k} |z_i| / \sum_{i=1}^{k} |x_i| \qquad (8)$$

Where $|z_i| = |y_i| + |x_i|$.

From the concept itself we can easily understand that the value of *CCA* in related packet padding is higher than the value of *CCA* in predicted padding.

In the proposed method we inject a standard page and use this standard page as the benchmark for packet padding. We assume that the standard page has N objects and the size of each object is S where N is the statistical average of the number of web objects on a web page. N is defined as

$$N = \Sigma_{in=1}(p(x_i) * \Sigma_{jm=1} I_{sj}) \qquad (9)$$

Where *n* is the total number of web pages, *m* is the total number of objects in the given web page, and I_{sj} is 1 when s_j is the j*th* web object of a web page.Otherwise I_{sj} is 0.S can be defined as follows:

$$S = \Sigma_{in=1}(p(x_i) * \Sigma_{jm=1}|s_j|) \qquad (10)$$

In an ideal situation, the intended page possesses N objects and each object's size is S, and the same for the cover traffic. Therefore, each session of downloading possesses 2N web objects and each web object's size is S packages. As a result, the standard size of one data downloading session is

$$|Z| = |X| + |Y| = 2N * S.$$

5,PERFORMANCE ANALYSIS OF RELATED PACKET PADDING METHOD

In this section, we first analyse the effectiveness of the system. Practically people use more devices to surf the web. Some devices have sufficient cache size, usually at the Gigabyte level. The cache space demanded for one browsing session is usually limited as the browsing length is limited. So the cache size is ―infinite‖ in this case.

But most of the web users use wireless mobile devices to surf the web which has finite cache memory. We will discuss these two cases in this section.

i. Effectiveness of proposed method

According to our padding strategy, when the user has browsed *m* web pages, the attacker may observe *k (k = m/4)* downloading session where *k=m/2* in predicted packet padding.

We now obtain the anonymization effectiveness of the proposed method as

$$AE_{related}(m) = 1 - \sum_{}^{3m/4} p(x_j) \qquad (11)$$

But the anonymization effectiveness of the predicted packet padding is

$$AE_{predict}(m) = 1 - \sum_{j=1}^{m/2} p(x_j) \quad (12)$$

While for dummy packet padding

$$AE_{Dummy}(m) = 1 - \sum_{j=1}^{m} p(x_j) \quad (13)$$

Where w_j is the jth popular web page.Comparing (6), and (7), we define anonymization effectiveness of related packet padding as

$$AE_{related}(m) = AE_{related}(m) + \sum_{j=1+(3m/4)}^{m} p(x_j) \quad (14)$$

From this equation (14) we clearly come to know related packet padding strategy is better than predicted packet padding.

ii. Infinite and finite cache size

Yu et al. (23) found that when the number of pages for a browsed for a web site is increased, then the anonymization effectiveness has increased. It is illustrated in the fig.1.

Suppose a person has browsed q web pages of a web site and the volume of the traffic is $|x_{i1}|, |x_{i2}|, \ldots, |x_{iq}|$, which is denoted by $|X_q|$ as follows:

$$|X_q| = \sum_{j=1}^{q} |x_{ij}|.$$

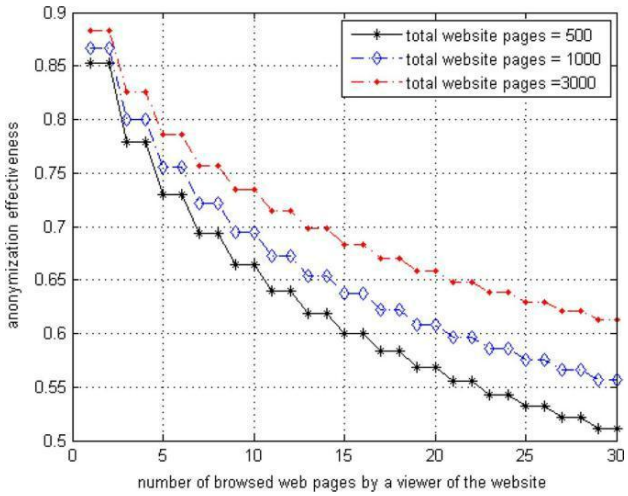

Fig.2. Anonymization effectiveness of Predicted Packet Padding against number of browsed pages under different total number of pages as 1000, 5000 and 10000.

when q is sufficiently large, we know that the probability of the q intended pages are the q most popular web pages is $\sum^{q}_{i=1} p(x_i)$, defined as

$$\sum_{i=1}^{q} p(x_i) = \sum_{i=1}^{q} |x_i| / |X_q| \qquad (15)$$

The volume of the cover traffic is defined as

$$\begin{aligned} |Y_q| &= \sum_{j=1}^{q} |y_{ij}| \\ &= (1- h(q))\ |X_q| \\ &\leq (1 - \textstyle\sum^{q}_{i=1} p(x_i))^* \ |X_q| \end{aligned} \qquad (16)$$

By combining (15) and (16) with the definition of *CCA*, we can find out the *CCA* when user has browsed Q web pages

Suppose a user has browsed q web pages, the size of the covered traffic is

$$|Z_q| = q^* 2 * N * S$$

This is less than the value of *CCA* in dummy packet padding. But it is not fitted for browsing devices with small cache memory. A simple and effective way for this case is to cache the related and most popular pages within the limited cache size. The hit ratio of a device with a cache size of c is calculated as

$$h(c) = \sum_{i=1}^{c} p(x_i)$$

6, RELATED PACKET PADDING ALGORITHM

We propose the algorithms for the proposed anonymization strategy in this section. First, we present a simple and effective algorithm to decide which pages will be selected for padding. Secondly, we use the predicted packet padding algorithm proposed by Yu et al. (23). Finally we introduce the algorithm namely LFU-SS to fetch the web pages from the local cache in the client side.

i. Related Page Prediction Algorithm

Related Page Prediction algorithm is used to find out the web pages related to the given user request and also to find the cover page from that list which has the most popularity.

The cost of the current prediction algorithms is expensive. In order to improve their browsing accuracy, a system should maintain the server statistics and browsing history of all the clients. The server always maintains this detail. In our algorithm, based on the server statistics, the web page with highest global hit is selected as cover traffic. This cover traffic will be padded with the intended page. So, we can ignore the relationship between the last requested web page and the next requested web page.

$$\beta(q) = [\ \sum_{i=1}^{q} p(x_i)\ ]^{-1} \qquad (17)$$

The sequence of our algorithm is discussed here. First we select the web pages related to the request. Then related *m* web pages are stored in a queue called related queue after sorting them based on the popularity. If the requested page is not in the related queue then the first page in the related queue is taken as cover traffic. Otherwise the requested page is first fetched from the queue and other (*m*-1) pages are again sorted. Now the above procedure is followed. After taking a web page from the related queue as cover traffic that page will be deleted from the queue. For each browsing session, related queue is created once. This algorithm is shown in Algorithm 1.

ii. Predicted Packet Padding Algorithm

In this algorithm, we discuss detail of the packet padding strategy. The main goal of this algorithm is to make the intended page all the same using the cover traffic. We can use the same predicted packet padding strategy for both finite and infinite cache size cases.

The process of the algorithm is preceded below. The page selected as cover traffic is padded with the intended page to make all the pages same in attacker's point of view.

iii. LFU-LS Algorithm

In this section we propose the detail of the process of storing the web pages in the cache. When the user request a web page the browser checks the cache for that page. If it is not found in the cache then set the read_hit value by referring the server statistics.

Algorithm 1: Related Page Prediction

// define the related queue

1. Declare a related queue Q_R with length of *l*.
2. Web page X_q is requested.
3. Load Q_R with the most popular *l* pages, sorted by relevancy and global hit and then set head=1.

While *true* **do**

// Check X_q is in the queue.

4. **If** X_q E Q_R **then**
   $Q_R = Q_R - \{X_q\}$;
   End

// the related page Y_q is predicted.

5. $Y_q = Q_R$(head);
6. head+ =1;

End

Algorithm 2: Predicted Packet Padding Algorithm

```
//calculating the standard download size;
1. |Z_s| = 2N̄ · S̄;
while true do
    for A new user i do
        while user i request page j(1 ≤ j ≤ n) do
            2. |Y_j| = |Z_s| − |w_j|;
            //identifying padding pages;
            3. call algorithm 1 until |Y_j| is met;
            //padding with predicted pages ;
            4. using the packets of |Y_j| to meet
            requirement of equation (10);
        end
    end
end
```

Algorithm 3: LFU-LS

```
1. Web page X_q is requested.
If X_q not in cache then While cache full then
        2. Delete the file which has least read_hit ratio.
        3. Rearrange the heap in order to be a min_heap.
    End
    4. Compute read_hit value for X_q by referring server statistics.
    5. Download X_q and store it in the cache.
    6. Insert metadata details into the heap.
    7. Rearrange the heap in order to be a minheap.
Else
    8. increase read_hit value of X_q by 1.
    If read_hit ≥ threshold then For each file in cache do
            9. Read_hit = threshold/2;
        End
    End
    10. Rearrange the heap if necessary.
End
```

If the cache memory is full, the existing pages will be replaced based on the read_hit ratio. If the requested web page is present in the cache then the read_hit is increased by 1 and the browser fetches that page from the cache. In this case no need to send the request to the server. So the hacker cannot hack the web page by referring continuous requests.

7, IMPLEMENTATION AND PERFORMANCE COMPARISON

In this section we discuss about the real world implementation of Related Packet Padding to investigate the effectiveness and performance of the anonymous web browsing compared with the existing methods.

i. Experimental Setup

We use the smart phone for client side. In the client side we implement a new algorithm namely Least

Frequently Used-Local Statistics (LFU-LS). This algorithm is used to replace cached web pages by new one based on the local statistics when the cache memory is full. Local statistics means the read_hit calculated for a web page after stored it in the local cache of the client.

In the server side, *m* number of related web pages are selected which were listed in the descending order based on the page rank. To store those web pages, a queue namely Related Queue (Q_R) is used. From this queue the web page in the rear will be selected for padding with the requested web page, if that is not similar to the requested page. Otherwise, the next web page will be selected for padding.

ii. Performance Analysis

The performance of the proposed system is nearly increased than the existing predicted packet padding. In predicted packet padding the cover page is selected based on popularity only. Because the cover page is selected based on the popularity as well as relevancy, the usage of cover page in future is obviously increased.

8, FURTHER DISCUSSION

In this paper we propose the related packet padding strategy to provide the better anonymous browsing in mobile phones such as smart phone, i-pad. But an attacker can easily break the anonymity of the system by using any vulnerable component. We discuss some drawbacks of the proposed system and its future works.

1) Client Side Protection. In this paper we discuss the attacks in the server side only. But attacker can easily break the anonymity based on the client requests.
2) Dynamic Web Pages. We can effectively use this system for static web pages only. Dynamic web pages are not browsed effectively by using this strategy.
3) Link Padding. In this paper we investigate the method in case of page padding and neglected link padding.
4) Cache-Oblivious algorithm. In this paper propose a new algorithm for cache handling. But we can try to use cache oblivious algorithms for this purpose.

9, SUMMARY AND FUTURE WORK

The proposed method is basically based on reducing the delay and bandwidth waste of anonymous web browsing effectively in mobile devices in order to make anonymous web browsing applicable for web viewers of mobile phones. A thorough analysis and comparison between proposed related packet padding and existing predicted packet padding has been established. The way of using the small size cache memory effectively for web browsing has been identified.

For our future work, there are a few directions to discuss further. First, our future goal is to extend our research to dynamic web site case. In the current paper we discuss about static web sites Only. Secondly , we can use the cache oblivious algorithm which is more effective than the LFU-LS we proposed in this paper. Thirdly, we would like to use the link padding strategy instead of packet padding. Finally, we focus

on the performance of the anonymous web browsing. The browsing speed in mobile devices is less than the speed in other devices such as personal computers.

IX. REFERENCES

[1] A. Hintz, ―Fingerprinting websites using traffic analysis,‖ in *Proc. Workshop Privacy Enhancing Technologies*, 2002.

[2] A. Serjantov and G. Danezis, R. Dingledine and P. Syverson, Eds., ―Towards an information theoretic metric for anonymity,‖ in *Proc. Privacy Enhancing Technologies Workshop (PET 2002), LNCS 2482*, Apr. 2002.

[3] B. A. Huberman, P. L. T. Pirolli, J. E. Pitkow, and R. M. Lukose, ―Strong regularities in world wide web surfing,‖ *Science*, vol. 280, no. 3, 1998.

[4] C. Diaz, S. Seys, J. Claessens, and B. Preneel, R. Dingledine and P. Syverson, Eds., ―Towards measuring anonymity,‖ in *Proc. Privacy Enhancing Technologies Workshop (PET 2002)*, Apr. 2002, Springer-Verlag, LNCS 2482.

[5] C. E. Shannon, ―Communication theory of secrecy systems,‖ *J. Bell Syst. Technol.*, vol. 28, pp. 656–715, 1949.

[6] D. Chaum, ―Untraceable electronic mail, return addresses, and digital pseudonyms,‖ *Commun. ACM*, vol. 24, no. 2, pp. 84–88, 1981.

[7] http://docs.oracle.com/cd/E26180_01/platform.94/ATGProgGuide/html/s0702requestprocessinginanucleusbased01.html

[8] http://shika.aistnara.ac.jp/products/wcol/tec h/p_concept.html dated 7/9/2011.

[9] http://www.w3.org/protocols/rfc2616/rfc261 6.html dated 01/02/2011.

[10] L. Breslau, P. Cao, L. Fan, G. Phillips, and Shenker, ―Web caching and zipf-like distributions: Evidence and implications,‖ in *Proc. INFOCOM*, 1999, pp. 126–134.

[11] M. G. Reed, P. F. Syverson, and D. M. Goldschlag, ―Anonymous connections and onion routing,‖ *IEEE J. Select. Areas Commun*, vol. 16, no. 2, pp. 482–494, Feb. 1998.

pp. M. K. Reiter and A. D. Rubin, ―Crowds: Anonymity for web transactions,‖ *ACM Trans. Inform. Syst. Security*, vol. 1, no. 1, 66–92, 1998.

qq. M. Liberatore and B. N. Levine, ―Inferring the source of encrypted http connections,‖ in *Proc. 13th ACM Conf. Computer Communications Security*, New York, 2006, 255–263, ACM.

[12] Q. Sun, D. R. Simon, Y.-M. Wang, W. Russell, V. N. Padmanabhan, and L. Qiu, ―Statistical identification of encrypted web browsing traffic,‖ in *Proc. IEEE Symp. Security and Privacy*, 2002.

[13] R. Dingledine, N. Mathewson, and P. F. Syverson, ―Tor: The second generation onion router,‖ in *Proc. USENIX Security Symp.*, 2004, pp. 303–320.

[14] R. Pries, W. Yu, X. Fu, and W. Zhao, ―A new replay attack against anonymous communication networks,‖ in *Proc. ICC*, 2008, pp. 1578–1582.

[15] Rezaul Alam Chowdhury, ―Cache-efficient Algorithms and Data Structures: Theory and

Experimental Evaluation‖, PhD Thesis, Department of Computer Sciences, The University of Texas at Austin, 2007.

[16] S. Brin and L. Page, ―The anatomy of a large-scale hyper textual web search engine‖, In Proceedings of the Seventh
World Wide Web Conference, Apr. 1998.

[17] S. Brin and L. Page. The PageRank Citation Ranking: Bringing Order to the Web. January 29, 1998.

[18] S. E. Coull, M. P. Collins, C. V. Wright, F. Monrose, and M. K. Reiter, ―On web browsing privacy in anonymized netflows,‖ in *Proc. 16th USENIX Security Symp.*, Berkeley, CA, 2007, pp. 1–14.

[19] S. Yu, T. Thapngam, S. Wei, and W. Zhou, ―Efficient web browsing with perfect anonymity using page prefetching,‖ in *Proc. 10th Int. Conf. Algorithms and Architectures for Parallel Processing (ICA3PP 2010)*, 2010, pp. 1–12.

[20] Shui Yu, *Member, IEEE*, Guofeng Zhao, Wanchun Dou, and Simon James, ―Predicted Packet Padding for Anonymous Web Browsing Against Traffic Analysis Attacks‖,

[21] IEEE transactions on information forensics and security, Vol. 7, No. 4, August 2012.

[22] Victor Safronov and Manish Parashar,

―Optimizing Web Servers Using Page Rank Prefetching for Clustered Accesses‖.

[23] W. J. Reed and M. Jorgensen, ―The double pareto-lognormal distribution— A new parametric model for size distributions,‖

Commun. In Statistics—Theory and Methods, vol. 33, no. 8, pp. 1733–1753, 2003.

[24] W. Wang, M. Motani, and V. Srinivasan,

―Dependent link padding algorithms for low latency anonymity systems,‖ in *Proc. ACM Conf. Computer and Communications Security*, 2008, pp. 323–332.

[25] W.G. Teng and C.-Y. Chang, ―Integrating web caching and web prefetching in client-side proxies,‖ *IEEE Trans. Parallel Distributed Syst.*, vol. 16, no. 5, pp. 444– 455, May 2005.

[26] Y. Xie and S.-Z. Yu, ―A large-scale hidden semi-Markov model for anomaly detection on user browsing behaviors,‖ *IEEE/ACM Trans. Networking*, vol. 17, no. 1, pp. 54– 65, Jan. 2009.

[27] Z. Ling, J. Luo, W. Yu, X. Fu, D. Xuan, and W. Jia, ―A new cell counter based attack against Tor,‖ in *Proc. ACM Conf. Computer Communications Security*, 2009, pp. 578– 589

[28] Zuyuan Fang and Brahim Bensaou, ―Fair Bandwidth Sharing Algorithms based on Game theory Frameworks for Wireless Ad-hoc Networks‖, in IEEE Infocom, 2004.