Acta Informatica Pragensia, 2015, 4(3): 206–225
DOI: 10.18267/j.aip.70
Peer-reviewed paper

It Leaks More Than You Think: Fingerprinting Users from Web Traffic Analysis

Xujing Huang*

Abstract

We show how, in real-world web applications, confidential information about user identities can be leaked through “non-intuitive communications”, in particular web traffic which appear to be not related to the user information. In fact, our experiments on Google users demonstrate that even Google accounts are vulnerable on traffic attacks against user identities, using packet sizes and directions. And this work shows this kind of non-intuitive communication can leak even more information about user identities than the traffic explicitly using confidential information. Our work highlights possible side-channel leakage through cookies and more generally discovers fingerprints in web traffic which can improve the probability of correctly guessing a user identity. Our analysis is motivated by Hidden Markov Model, distance metric and guessing probability to analyse and evaluate these side-channel vulnerabilities.

Keywords: Side-channel leakages, User identities, Web applications, Google accounts.

1 Introduction

It is not a secret that communication of a web application can reveal a user's web activities through side channels, even when the web traffic is encrypted. Previous work (Chen et al., 2010; Cheng, Avnur, 1998) has shown that an eavesdropper can infer a user's web browsing activities from the lower-lever traffic features, such as packet sizes, numbers, timings, etc.

Traffic-analysis attacks exploit web traffic from communications involving direct transmission of confidential information. Large number of studies of side-channel leakages in web applications have already analysed communications explicitly interacting with sensitive information. However, we suppose that a communication, though not explicitly transiting sensitive information, may still leak or even leak more about user secrets.

For example, Fig. 1 outlines a scenario showing how user search inputs in an online shopping center can be revealed. In step 1, the communication explicitly transmits a search keyword typed in a search bar to the server. The web traffic generated is indistinguishable from user inputs, so an eavesdropper gains no knowledge of user search keyword from traffic analysis.

* School of Electronic Engineering and Computer Science, Queen Mary University of London, Mile End Road, E1 4NS, London, United Kingdom
✉ x.huang@qmul.ac.uk

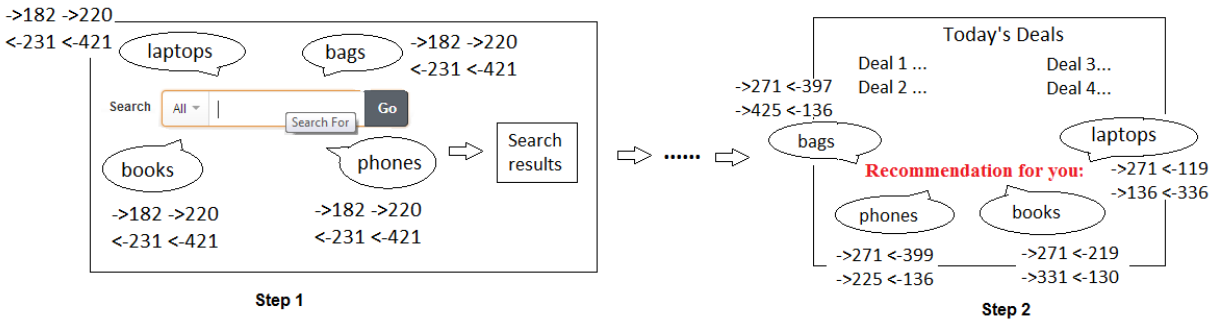

Fig. 1. *Communications explicitly and implicitly transmitting sensitive information. Source: Author*

Now consider the communication illustrated in step 2 in Fig. 1. A transition, some-distance away from the communication in step 1, is triggered when a user visits a web page regarding "Today's deals" on the shopping site. This appears to transmit no information about the user's search keyword. However, as indicated in step 2, web traffic in this transition varies depending on search keywords, reflected in the content of “Recommendation", which can actually reveal user search inputs.

Therefore, this work pays attention to communications which implicitly interact with sensitive information. Moreover, we extend communications including those with third-party websites outside the server of which the confidential information is located.

In this work we propose a new traffic-analysis attack using packet sizes and directions to reveal user identities from 50 Google user accounts to websites included in the Alexa (Alexa, 2015) Top 150 websites. There are 25 out of the 150 websites which are likely to leak a large amount of user identities. They are examined from five locations, to ensure the leakages are consistent among different locations. Moreover, we develop four testing scenarios for investigating the leaking sources, more specifically, whether cookies and logged user accounts are the factors causing web traffic to leak user identities. Although cookies is a well-known source of tracking users in web applications, there is little work studying on the effect of cookies in traffic analysis in web applications.

A novel approach based on the Hidden Markov Model (Baum & Petrie, 1966; Baum & Eagon, 1967) and the Damerau-Levenshtein distance (Navarro, 2001) to analyse the collected web traffic. Then we use guessing probability (Malacaria, 2015) to evaluate the probability of guessing user identities correctly in one try.

In summary, our main contributions are as

- We propose a novel side-channel attack scenario against user identities and demonstrate that even Google users are vulnerable under this kind of attack in real-world applications, through communications with third-parity websites.
- We analyse the web traffic based on the hidden Markov model and an optimised Damerau-Levenshtein distance, to construct traffic patterns of transitions. And we present a new fingerprinting model with a “one to many” mapping between transitions and their observations.
- We analyse the effect of cookies and logged user accounts in side-channel leakages of user identities.

2 Fingerprinting Model and Threat Scenario

2.1 Fingerprinting Model

To better explain our work, we define a transition system, as the basis of our model. This model is similar to the one in (Backes, Doychev, & Kopf, 2013).

Formally the transition system is remodelled as *TS = (G, O, A, f, g, MLS, S)*.

A web application is modelled as a directed graph *G* = (*N*, *E*), where *N* is the set of nodes, each node $n \in N$ representing a web page of the web application.

An edge from one node to the next is a transition between web pages $e = (n_1, a, n_2) \in E$, where $n_1, n_2 \in N$.

A user action $a \in A$ is defined as a sequence of pairs (*element*, *value*), where *element* is a component the user performs on, and *value* is the input on this component. For example, the action of logging into a user account consists of two pairs: one for the user name, and the other for the password.

We partition user actions as $A = A_h \cup A_l$, where A_h are the user actions containing sensitive data and A_l those with no sensitive data.

Transitions of interest in this chapter are partitioned into direct and indirect transitions. They are defined according to the type of user actions which trigger them. A **direct transition**, expressed as $(n_1, a \in A_h, n_2) \in E$, is triggered by a user action containing sensitive data.

An example of a direct transition is the one originated from an authentication on a login page.

An **indirect transition** is triggered by a user action with no sensitive data input, which can be represented by: $(n_1, a \in A_l, n_2) \in E$. An indirect transition, for example, is performed by clicking on a (non sensitive) link on a page. An execution path containing both direct and indirect transitions is regarded as a **combined path**.

Set *S* is a set of sensitive information.

O is an observation set about the web traffic observed in a transition. Each observation is a sequence of directional sizes describing the packets generated in the communication. For example, an observation $\rightarrow 125, \leftarrow 237$ indicates a request packet of length 125 bytes sent from the browser and a response packet of length 237 bytes sent back from the server.

In this work, multiple observations are collected for a transition. Hence we define function $(n_1, a \in A_h, n_2) \in E$, which considers that a transition $e \in E$ is associated with a set of observations $O_e \in P(O)$.

Assume each transition associates to a hidden traffic pattern which is followed by most observations for this transition. Given a set of observations O_e for a transition *e*, we define a **most likely sequence** mls_e, which is the traffic pattern best matching the observations in set O_e.

Then a function $g: P(O) \rightarrow MLS$ maps each observation set $O_e \in P(O)$ to a most likely sequence $mls_e \in MLS$. The terms *most likely sequence* and *traffic pattern* are used interchangeably throughout this thesis.

2.2 Threat Model

We propose a threat scenario of revealing user identities from Google user accounts.

A user first logs into a Google account, where the transition is regarded as a direct transition. Then the user accesses to a new website which is either a Google or non-Google website by typing a URL address of the website into a URL bar. This communication is regarded as an indirect transition.

Assume an attacker can be a Google service provider or a database administrator who is able to profile traffic patterns of web traffic for each user account, given a space of Google users. Consider that the attacker has prior knowledge concerning which website a victim is visiting. Details of how an eavesdropper fingerprints websites can be seen in many literature, see e.g. (Cai et al., 2012).

The attacker aims to identify a user's identity, i.e. which user out of the space of Google users carries out the communication, by eavesdropping a web traffic and matching it to a traffic pattern for a user.

Consider a non-Google website as a **third-party website** or an **external website**, relative to a Google website which is regarded as an **internal website**.

Experiments in this threat scenario are implemented in a close-world environment from a perspective of a developer. Instead of obtaining an accurate leakage of user identities, this work aims to demonstrate a possible threat of leaking user identities through communications with third-party websites.

3 Testing Specification

In this section, we show how we configured the experimental settings, to build the test cases and perform the experiments.

3.1 Testing Process

First, the secret we consider is the user identity represented by a user account, instead of the user name and the password of the account. We examine user identities from 50 Google accounts on Alexa Top 150 websites (Alexa, 2015), excluding Google websites. A round-robin test is performed on each user account for all the 150 websites. Each test case is run 10 times. Caches and cookies are cleared every time before a new test starts. The aim is to discover if the user identity can be uncovered through communications with a third-party websites.

Selenium (Selenium, 2015), a web driver, is used to perform the tests by simulating the user actions on websites automatically; and Jpcap (2015) – a network packet capture library, records the web traffic generated during the experiments.

A sentence “the website fingerprints user" indicates that the indirect transitions in the execution paths in terms of this website reveal user identities from the web traffic. We use “a website fingerprints users" and “a website leaks user identities" interchangeably.

Since there are 150 websites and 50 user accounts, 7500 test cases are required to be executed in a testing round. Moreover, if each test case is executed ten times, there are 75000 tests in total when testing on all the websites one round. Hence a large amount of time will be spent when executing one round for all the websites.

Moreover, with a preliminary analysis of some collected web traffic, we discover that not every website leaks a large amount of user identities.

Therefore, to save time for performing in-depth analysis on websites which likely leak large amounts of user identities, a coarse *cleansing operation* was first conducted to roughly refine the websites by eliminating those which may leak few user identities.

3.1.1 Cleansing Operation

To make results more convincing, in the beginning all 7500 test cases were tested using two laptops located in a same network. A test case was executed where a user account was first authenticated and then an external website was accessed to. It took around two months to complete the cleansing operation.

Analysing and quantifying the leaks using the methodologies described in sections 4, if the leakages for a website at both laptops are similar and the guessing probabilities are relatively high, the website probably leaks user identities stably at a network. Thus the website will be selected to be tested in depth in next stage. As a result, 25 websites were obtained, each of which is likely to leak large amount of the secret.

The two laptops were configured, one of Windows XP system with Intel Core i3-2310M CPU @ 2.10GHz and 3.41 GB of RAM, and the other of Windows 7 system with Intel Core i5-3230M CPU @ 2.60GHz and 3.88 GB of RAM. Next, we show how to perform the in-depth analysis.

3.1.2 In-depth Testing Process

In addition to the 25 external websites, two internal websites **www.google.com.br** and **www.google.it** were connected to in the indirect transitions after authenticating. They were performed in order to compare with those from internal to external websites.

The 50 user accounts on the 27 websites were examined in the *PlanetLab* testbed (PlanetLab, 2015), from five virtual machines located at the Technical University of Berlin (Berlin, Germany), the University of Cambridge (Cambridge, UK), Imperial College (London, UK), the University of Neuchatel (Neuchatel, Switzerland) and the University of Stuttgart (Stuttgart, Germany) respectively.

We tested them at multiple locations to (1) guarantee that the leakages related to a website from different networks are consistent, in order to mitigate the error from a single testing site as far as possible; and to (2) investigate whether the leakage depends on locations, which may reveal user locations.

The Firefox web browser was installed in the five machines with versions ranging from 3.6.24 to 3.6.28 under the Fedora 14 operating system. It is the browser used by Selenium for executing the testing.

3.2 Testing Scenario

In the real world, a user often visits an external website with her Google account logged in. It is possible that the external web server requests information related to the logged user account. The data may be insignificant to user sensitive information, however, it may contain some unique data which leads to distinguishable web traffic between different users.

Moreover, cookies which store user login information may be sent to the network during communications with an external website. Concerning that logged user accounts and cookies

may bring information of user identities into communications, we design four testing scenarios to analyse the effects of these factors on the leakage of user identities.

Scenario 1. General

This is a general scenario same as the one used in the cleansing process. Communications start from a direct transition of authenticating a user account, and then immediately access to a different website, either an internal or external website.

Scenario 2. Delete cookies

In this scenario, the cookie is cleared every time after authenticating, just before triggering a following indirect transition. Since logging into a Google account simply sets a cookie, an action of deleting cookie makes completely no information related to the user account exist.

Comparing with the result in scenario 1, this scenario can be used to analyse that whether the leakage is affected greatly by removing cookies. If the guessing probabilities are relatively high in scenario 1 but are relatively low in scenario 2, it is probable that the web traffic is affected largely by the cookies. However, simply from the statistics in scenarios 1 and 2, we are still unable to get a clear picture of which factor, out of cookies and logged user accounts, decides the web traffic. Hence, scenario 3 is developed.

Scenario 3. Log out user accounts

In scenario 3, the user account is logged out before communicating with a new website, but the cookie is retained. This is a way attempting to clean user trails as much as possible, but only keeping cookies as a source of user identities. It aims to see if there is an influence on web traffic when a user account is logged out.

By comparing the results between scenarios 1, 2 and 3, a clearer picture regarding the influential factors on the leakages can be suggested, in terms of which factor, logged user accounts or cookies, accounts for a larger effect on the leakage.

Scenario 4. Log out user accounts + Delete cookies

In addition to logged user accounts and cookies, other external unknown-sourced factors can lead to a variation of web traffic. It can be mistakenly regarded that the variation is originated from cookies or logged user accounts.

Therefore, how influential these unknown factors are in the variation of web traffic is estimated in this scenario, to mitigate the error caused by external unknown factors. This scenario attempts to clear all the user trails generated during communications. Instead of logging out user accounts and cleansing all trails, there is an alternative in a more straight way, i.e. not logging into the user accounts, which stays away from the user identities.

This scenario should give rise to a same effect as cleansing the cookies in scenario 2, because there is not any information related to users retained. Therefore, the aim of this scenario is to provide a reference for the result in scenario 2, which confirms the accuracy of results in these scenarios. Moreover, it can also be used to check if exceptions exist when the results from scenarios 2 and 4 are different.

In scenario 4, each test case contains a single transition, starting from Google's authentication page and forwards to an examined website straightly. A test case in terms of a website was performed 500 times and every ten traces of web traffic were aggregated into one group, to construct a space with as same the number as the user accounts analysed in other scenarios. Analysing the observations between each group, the data obtained represents the variation

generated by the unknown factors. Test cases in this scenario are considered as indirect transitions.

The unknown factors evaluated in scenario 4 are regarded as *external factors* which have no relation to user identities. Relatively, logged user accounts and cookies are deemed as *internal factors* as they are related to user identities.

For a website, each testing round executed the test cases on 50 user accounts in four scenarios. The testing on 27 websites, consisting of 25 external and 2 internal websites, were performed at five locations simultaneously. One round of testing on all the 27 websites lasted over a two-week period. At least two testing rounds for the 27 websites were performed.

4 Data Analysis

In this section, we explain how we identify users through side-channel analysis on the collected web traffic. We ran each test case in terms of a user account 10 times. A traffic sequence is considered as an observation. Given a set of observations, request packets in each observation can be extracted according to the packet directions, to form a sequence of request packets to be analysed. Currently we do not analyse response packets. Hence each observation means a sequence of request packets.

Given a set of observations for a transition, we attempt to determine the **most likely sequence**, which is a hidden traffic pattern best matching the observations of a transition. If a transition holds a most likely sequence, we consider the most likely sequence fingerprinting the user whom the transition corresponds to. Now we explain how to normalise a most likely sequence. First define the packet characteristics.

4.1 Packet Specification

Given a set of observations O_e, we define the following notations:

O_e: a set of observations for a transition *e*;

o_{e_i}: the i[th] observation where $o_{e_i} \in O_e$; In a given set O_e, o_{e_i} can also be abbreviated as $o_i \in O_e$;

o_{i_k}: the packet at position *k* in o_{e_i}. It is presented by a pair (*value*, *k*), where *value* is the packet size and *k* is the position the packet is located in this observation. The use of packet location is to deal with a packet-disorder issue introduced later.

Definition 1 [Indistinguishable] Packets from two traffic sequences are considered to be *indistinguishable*, if they are located at the same position in the sequences and their sizes *a*, *b* satisfy:

$$|a - b| \leq tp \cdot max(a, b),$$

where $tp \in [0,0.5]$ is the threshold of indistinguishable packets. Moreover, two traffic sequences are determined to be *indistinguishable*, if every two packets, which are located at the same position in these two sequences, are indistinguishable. By default, we set $tp = 0.1$

Ideally the observations of a transition are identical in a limit time period. But external factors, e.g. performing time, may result in the sizes of packets which are located at the same position of observations slightly fluctuating. The use of indistinguishable packets aims to reduce the probably impacts from external factors on packet sizes among different observations.

In the real world, an issue of network packet disorder often occurs. When a reliable in-order delivery of packets is required, for instance when using Transmission Control Protocol (TCP), a transmitter resends the packets which are not received properly. This may cause packets disordered, e.g. a retransmitted packet is observed after a packet where it should be observed before the packet in a proper order.

This research analyses TCP packets, where sequence numbers of packets can be extracted to recover a correct order. However, to provide an extensive mechanism which can analyse traffic precisely even when packet sequence numbers are not available, this work proposes a stronger analyser using a *shift operation*, which, to a great extent, mitigates possible errors caused by disordered packets.

Definition 2. [Displacement Range] A displacement range is an integer $d > 0$.

Definition 3. [Shift with Displacement Range *d*] Given a packet located at position *k* in an observation $\vec{q}$, it can be shifted up to *d*-position forward or backward away from the current position *k*. This aims to recover the original position where the packet is actually located in a proper order. After shifting the sequence becomes sequence $\overrightarrow{q'}$.

Every packet after *n* times shifting in sequence $\overrightarrow{q'}$should be within the displacement range from their originial positions in observation $\vec{q}$. Displacement range is used to simulate the degree of packet disorder caused by a packet transmission error. If there is a packet retransmitted and observed in an incorrect order, it can be moved back to its correct position if the disorder happens within the displacement range.

So the proper setting of the displacement range is important for the precision of a most likely sequence. If the value is too small, it will miss the disordered packets which are outside the displacement range. But if it is too large, there may be incorrect judgements for the packets which are already in the proper order. In this work we set $d = 2$.

Next we describe how to construct the most likely sequence mls_e for transition *e*, i.e. $g(O_e) = mls_e$.

4.2 Modelling Packet Data

In a hidden Markov model (HMM) (Baum & Petrie, 1966; Baum & Eagon, 1967), states are not observable while outputs depending on the states are observable. We use a HMM to model the packet data in an observation set. We consider observations in a set are the outputs, and the traffic pattern generated from the observations can be deemed as a sequence of hidden states.

Given an observation set O_e of a transition *e*, a *position-dependent* model can be denoted as follows:

K = number of positions, i.e. the maximum length among the numbers of packets in observations

N_k= number of states at position *k*

$ST_k = \{st_{k_1}, \dots, st_{k_{N_k}}\} \in ST$= a set of possible hidden states at position *k*

Each state $st_{k_i} \in ST_k$ is unique and originated from the size of a packet observed at position *k* in an observation. This set may contain states related to the potential disordered packets, which are not observed but within displacement range from position *k*. The final confirmed state at position k is denoted by st_k.

M_k = number of outputs at position k

$OP_k = \{op_{k_1}, \dots, op_{k_{M_k}}\} \in OP$= a set of output at position k

Set OP_k collects packets that can be observed at position k in each observation $o_{e_i} \in O_e$, expressed by packet sizes. Similar to the states in a state set, outputs in this set may also include those probably disordered but within displacement range from position k.

$IP = \{ip_1, \dots, ip_{N_1}\} \in IP$= initial state distribution at position 1

ip_i denotes the initial probability of state st_{1_i}.

$C_k = \{c_k(st_{k_i}, st_{k+1_j})\} \in C$= transition probabilities between states at position k

It is a $N_k \times N_{k+1}$ matrix, and $c_k(st_{k_i}, st_{k+1_j})$ denotes the probability of transferring from state $st_{k_i} \in ST_k$ at position k to state $st_{k+1_j} \in ST_{k+1}$ at position $k+1$.

$B_k = \{b_k(op_{k_t} \vee st_{k_i})\} \in B$= output probabilities at position k

It is a $N_k \times M_k$ matrix, and $b_k(op_{k_t} \vee st_{k_i})$ denotes the probability of observing output $op_{k_t} \in OP_k$ in state $st_{k_i} \in ST_k$.

This model can be also characterised by a set of parameters $\mu = \{IP, C, B\}$. Then the Baum-Welch algorithm (Baum et al., 1970) is performed until the resulting probabilities converge satisfactorily, which maximises $\mu = \{IP, C, B\}$.Then the Viterbi algorithm (Forney, 1973) is performed to get a sequence of most likely states which has the maximum probability. The most likely sequence generated from$O_e$is regarded as a sequence of most likely states, where

$$mls_e = (st_1, st_2, \dots)$$

Next, we show how to quantify leakages. Fig. 2 gives an overview of the process of quantification. Given the most likely sequences generated from the observation sets, each for a transition, we define *confidence*, i.e. $conf(O_e, mls_e)$, for each most likely sequencemls_e.Then the probabilities of the most likely sequences can be obtained and the probability distribution *pd* on the set of most likely sequences *MLS* is built to quantify the leakage. The details are described in the following.

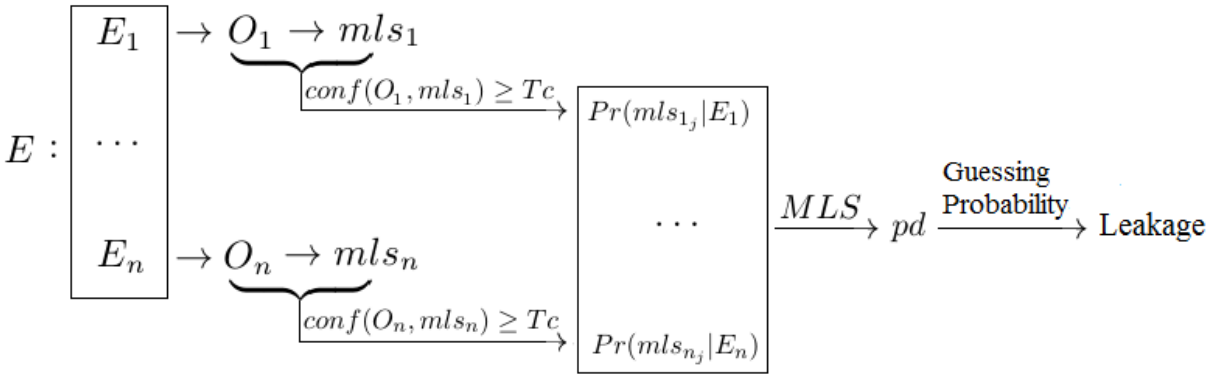

Fig. 2. *Overview of quantifying the leakage. Source: Author*

4.3 Confidence of Most Likely Sequence

First we define **confidence of most likely sequence**, a measure to assess the similarity between sequence$mls_e$and the observations in O_e.It can be abbreviated as **confidence.**

Definition 4. [Confidence] Given a set of observations O_e,and a most likely sequence $mls_e$for transition *e*, the confidence of most likely sequence is defined as

$$conf(O_e, mls_e) = 1 - min\left(\frac{ad + dm}{|mls_e|}, 1\right) \quad \textbf{(1)}$$

where *ad* and *dm* are the *average distance* and the *dissimilarity* between$mls_e$and observations in O_e respectively, which are defined later. $|mls_e|$ is the number of packets in sequence mls_e.In this equation,$mls_e$is regarded as the *reference sequence*. $conf(O_e, mls_e)$ can be abbreviated as$conf_e$.

Confidence is used to assess the overall similarity between a most likely sequence and the observations.

Definition 5. [Distance between two sequences] A traffic sequence$\vec{a}$can be obtained from traffic sequence$\vec{b}$through at least *n* operations of shift, insertion, deletion, substitution of packets, and transposing two packets within displacement range. The distance between sequences $\vec{a}$ and $\vec{b}$ is defined as

$$dis(\vec{a}, \vec{b}) = min(n, |\vec{a}|) \quad \textbf{(2)}$$

where $|\vec{a}|$ is the number of packets in sequence $\vec{a}$.

Definition 6. [Average Distance] The average distance between a traffic pattern mls_e and a set of observations$O_e$is defined as

$$ad = \frac{\sum_{o_i \in O_e} dis\,(o_i, mls_e)}{|O_e|} \quad \textbf{(3)}$$

where$dis(o_i, mls_e)$is the distance between mls_e and observationo_i,defined in Definition 5, and $|O_e|$ is the cardinality of set O_e.

After calculating the distances, each observation $o_i \in O_e$ is indistinguishable from traffic patternmls_e.

Nevertheless, there may still be dissimilarity between indistinguishable packets between observation$o_i$and sequence mls_e,when their packet sizes are not identical. Hence *dissimilarity* is taken into consideration, in terms of the standard deviation (Bland, Altman,1996) of indistinguishable packets.

Definition 7. [Dissimilarity] Given a set of observations O_e after Damerau-Levenshtein process and a traffic pattern mls_e, dissimilarity is defined by:

$$dm = \sum_{mls_{e_k} \in mls_e} \frac{sd_k}{mls_{e_k}} \quad \textbf{(4)}$$

where

$$sd_k = \sqrt{\frac{\sum_{o_i \in O_e} \left(o_{i_k} - mls_{e_k}\right)^2}{|O_e|}} \quad \textbf{(5)}$$

is the standard deviation of indistinguishable packets at position k, and mls_{e_k} and o_{i_k} represent the packet sizes at position k in sequences mls_e and o_i respectively.

Next we define a threshold of confidence Tc to assess determine if sequence$mls_e$is actually the traffic pattern of observations for transition e.

4.3.1 Determination of the most likely sequence

Given a transition e, a set of observations O_e and the most likely sequence mls_e, if

$conf(O_e, mls_e) \geq Tc$, $mls_e$is confirmed to be the most likely sequence for transition e.

Otherwise, transition e is considered generating random web traffic not conforming to any traffic patterns. We instead propose a **null** traffic pattern, so that in this case

$$mls_e = null \text{and } conf(O_e, mls_e) = 1$$

Accordingly, it can be concluded that $Tc \leq conf(O_e, mls_e) \leq 1$.

The confidence assesses a most likely sequencemls_e in terms of the similarity betweenmls_e and observations. A most likely sequence is regarded as a fingerprint of a user in a communication. After determining the most likely sequence for a transition, we build a relation of the “one to many" mapping between a transition and multiple traffic patterns.

4.4 One Many Mapping for a Transition

Previous work consider a "one to one" mapping where each transition associates to either one observation or one traffic pattern. Instead, this work builds a “one to many" mapping, where a transition can associate to more than one traffic pattern.

4.4.1 Similarity

First we define **similarity** between an observation and its traffic pattern to measure the similarity between them.

Definition 8. [Similarity] Given an observation $o_i \in O_e$and a most likely sequence mls_e, the similarity between$o_i$and $mls_e$is defined by

$$simi(o_i, mls_e) = conf(\{o_i\}, mls_e) \quad \textbf{(6)}$$

The similarity examines the degree of an observation$o_i$following the most likely sequence mls_e.

With similarity $simi(o_i, mls_e)$, we can determine if observation o_i follows most likely sequence mls_e by defining **a threshold of similarity**-Tm ($0.5 < Tm \leq 1$).

Given the threshold of similarity Tm, observation$o_i$is considered conforming to most likely sequencemls_eif and only if $simi(o_i, mls_e) \geq Tm$.

4.4.2 “One to Many" Mapping

1. When $conf(O_e, mls_e) < 1$

The probability of observations from transition e conforming to most likely sequence mls_e is denoted by:

$$Pr(mls_e \vee e) = \frac{M}{|O_e|} \quad \textbf{(7)}$$

where M is the number of observations in set O_e which follow traffic pattern mls_e, i.e. an observation o_i satisfies $simi(o_i, mls_e) \geq Tm$.

This probability $Pr(mls_e \vee e)$ represents the probability of observations from transition e following traffic pattern mls_e.

However, random observations not following mls_e can also be generated. They are considered following no traffic patterns. Hence we consider that they follow the **null** pattern, and the probability of observations following pattern **null** is defined as:

$$Pr(null \vee e) = 1 - Pr(mls_e \vee e).$$

As a whole, when $conf(O_e, mls_e) \neq 1$,an observation for transition e follows one of two most likely sequences, i.e. either mls_eor pattern **null**.

2. When $conf(O_e, mls_e) = 1$

Most likely sequencemls_ecan be either pattern null or a non-null pattern. In this case, all the observations follow most likely sequence mls_e. Hence the probability of observations for transition e following pattern$mls_e$is:$Pr(mls_e \vee e) = 1$.

By bringing in a null pattern, random web traffic for a transition is also associated to a traffic pattern. It is still an advance towards using multiple traffic patterns in traffic analysis. This model can improve the accuracy when evaluating leakage of user privacy, as a larger space of observations including random web traffic generated during communications are considered.

With a set of most likely sequences and their probabilities, this section describes how to quantify the leakage of the related secret.

4.5 Probability Distribution in terms of Traffic Patterns

First we build a probability distribution on the set of most likely sequences, in terms of the probabilities of observations for a set of transitions.

Definition 9. [Probability Distribution for a set of Transitions] Given a set of transitions $\boldsymbol{E}$, a set of observation sets$P(O)$where$f: E \to P(O)$, a set of most likely sequences *MLS* where$g: P(O) \to E$, and the probability distribution$pd: y \to [0,1]$on set *MLS* is defined by:

$$pd(mls_i) = \mu(e) \times Pr(mls_i \vee e) \quad \textbf{(7)}$$

whereμdenotes the probability of a transition e and mls_i is a most likely sequence for transition *e*.

In this chapter, we suppose a uniform distribution on set *E*, such that $\mu(e) = 1/|E|$.

4.6 Quantifying Leakages

With the most likely sequences and their probabilities, now we build an equivalence relation to measure the leakage of user identities.

Given a set of transitions *E*, a set of most likely sequences *MLS* for set *E*, and their probabilities $Pr(mls_e \vee e)$, where $mls_e \in MLS$ is a most likely sequence for transition $e \in E$, an equivalence relation *X* on set *MLS* can be built as:

$\forall mls_i, mls_j \in MLS, mls_i \sim mls_j$ if and only if the similarity between them satisfies:

$$simi\big(mls_i, mls_j\big) \geq max\Big(Pr(mls_i \vee e_k), Pr\big(mls_j \vee e_t\big)\Big)$$

where mls_i is a traffic pattern for transition e_k and mls_j for transition e_t, and $Pr(mls_i \vee e_k)$ is the probability of an observation for transition e_k following mls_i.

The sequence with a shorter length is regarded as the reference sequence, mentioned in Equation 1, i.e. $|mls_i| \leq |(mls_j)|$. This makes the similarity smaller, which provides a "stricter" matching between two most likely sequences.

In an equivalence relation, there may be an equivalence class classifying null patterns, as even the irregular web traffic can leak user privacy. Given an observation not following any traffic patterns, e.g., it is more likely that this observation is originated from a transition which holds the highest probability of generating irregular web traffic among all the transitions.

Guessing probability (Malacaria, 2015) is used to evaluate the probability of successfully guessing user identity in one try, based on the Smith's *vulnerability* (Smith, 2009) with an equivalence relation on most likely sequences.

Definition 9. [Guessing probability] Given an equivalence relation *X* on a set of most likely sequences *MLS* in terms of a secret *sec* and a probability distribution $pd: MLS \rightarrow [0,1]$, the guessing probability for secret *sec* is defined by

$$Gue(sec \vee X) \sum_{X_i \in X} gue\,(X_i)$$

where $gue(X_i) = max_{mls_k \in X_i} pd(mls_k)$ is the vulnerability of equivalence class X_i, i.e. the worst-case probability the secret can be guessed in one try.

5 Experimental Results

This section presents the experimental results of the in-depth analysis related to the 27 websites.

Given 50 Google accounts, in original the priori chance of guessing a user before observing web traffic is 0.02. The analysis used the following configurations: threshold of indistinguishable packets *Tp=0.1*, displacement range *d=2*, and the thresholds of confidence and of confidence *Tc =Tm= 0.7*.

5.1 Direct Transition

Between testing scenarios 1, 2 or 3, the direct transitions in test cases are only dependent on user accounts, regardless of the websites the indirect transitions communicate with.

By analysing the web traffic of direct transitions in different scenarios and in different testing rounds, in general, the guessing probabilities of user identities from direct transitions are all less than or around 0.1 at each of the five locations.

In the following sections, therefore, our focus is turned to the analysis of indirect transitions. Without particular explanation, the guessing probabilities mentioned are for indirect transitions.

In the following, we analyse the results of guessabilities for indirect transitions.

5.2 Indirect Transition

We assume that the web traffic for an indirect transition examined in a testing scenario can fingerprint the user when it is consistent regardless of the testing locations, testing machines,

testing time, etc. In other words, similar guessing probabilities will be obtained among different testing rounds and different locations.

In general, we consider the transitions of a website leak user identities if

1. the guessing probabilities generated at different locations are consistent. The five variation trends of the guessing probabilities in four scenarios, each generated from a location, generally follow a consistent variation pattern; and

2. the highest gap between two guessing probabilities among any two scenarios should be large enough.

In condition 2, currently we only examine the leakage when the largest gap between guessing probabilities is originated from scenarios 1 and 4, i.e. the highest is from scenario 1 and the lowest from scenario 4. Leakages in other cases are left for future investigation.

To ensure the variation trends of guessing probabilities from different locations are consistent, we execute test cases for each website at least two rounds at each location, except the testing sites in Berlin and Neuchatel. This is because that the two testing sites are unavailable after a round of testing for all the websites.

Fig. 3 plots six websites which leak user identities among the examined 27 websites, in terms of guessabilities from five locations. A line plotted in a sub figure manifests the variation trend of guessabilities at one testing site. As can be seen, the guessing probabilities are generally highest in scenario 1 whereas lowest in scenario 4.

Although test cases were only tested one round in Berlin and Neutchatel, the two locations have consistent variation trends for most of the six websites. This can be deemed as another sign which indirectly manifests the high consistency of guessing probabilities between locations, as the variation trends at these two locations are consistent to the variation pattern in one try.

There may be a few vibrations between consistent variation trends. For example, in Fig. 3(4) in terms of website **dailymail.co.uk**, the guessing probabilities in London and Neuchatel are essentially identical under scenarios 2, 3 and 4, while in Berlin and Stuttgart, the statistics in scenario 3 are lower than those in scenarios 2 and 4. Nevertheless, relative to the gaps with guessing probabilities in scenario 1, the differences between the guessing probabilities in scenarios 2, 3 and 4 are fairly small. Thus it can still be considered that the variation trends are consistent among the five locations.

From the variation trends displayed in Fig. 3, we produce three templates mainly followed by the variation trends for the six websites. As the websites are considered revealing user identities, a template is also deemed as a **leaking pattern of user identities**.

For a website whose variation trends of guessing probabilities from indirect transitions follow a leaking pattern, the indirect transitions are considered as the vulnerabilities of leaking user identities.

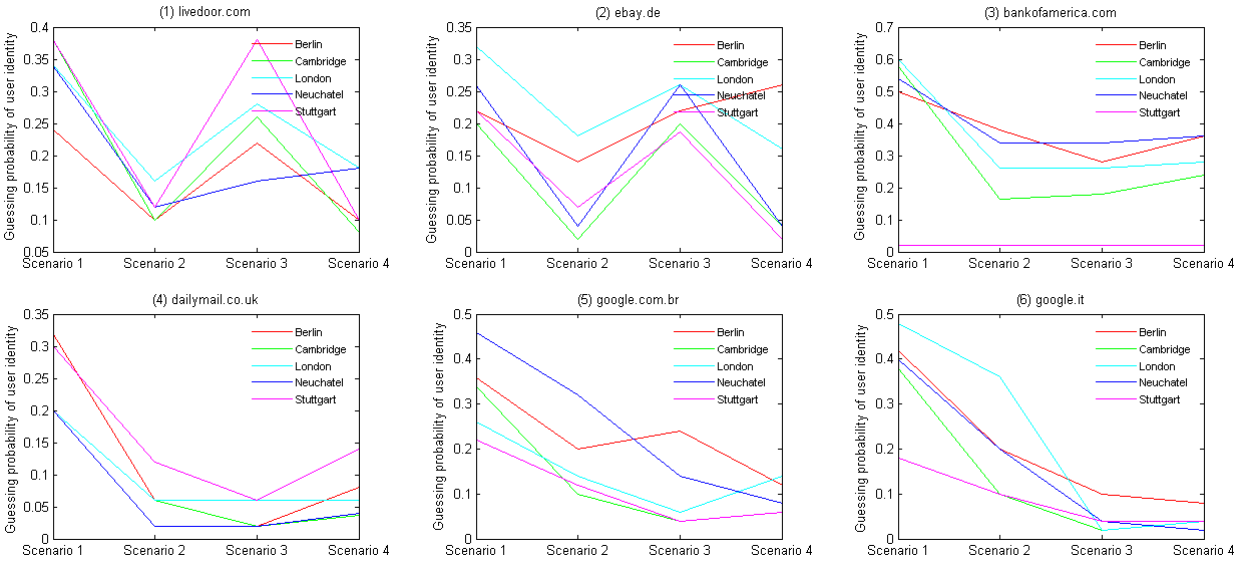

Fig. 3. *Websites with consistent variation trends of guessabilities between locations. Source: Author*

Next we analyse the three leaking patterns.

1. $G(1) \simeq G(3) > G(2) \simeq G(4)$

In this leaking pattern, guessing probabilities in scenarios 1 and 3 are close to each other, and so are those in scenarios 2 and 4. However, the figures in scenarios 1 and 3 are much larger than those in scenarios 2 and 4.

Symbol $\simeq$ is used to express that two guessing probabilities are with similar values. Compared with scenario 1, guessing probabilities in scenario 2 fall down sharply when the cookies were cleared, to a same level as the values in scenario 4.

This verifies what we expected as indirect transitions in both scenarios 2 and 4 should involve no information related to user accounts. On the contrary, the guessing probabilities stay at a high level as long as the cookies were retained, even when the user accounts were logged out in scenario 3.

This suggests, therefore, that cookies cause the web traffic for indirect transitions depending on users, for a website whose the variation trends of guessing probabilities follow this leaking pattern.

For websites **livedoor.com** and **ebay.de** displayed in Fig. 3(1) and 3(2), their variation trends generally follow this leaking pattern, excluding that in Neuchatel for **livedoor.com** and that in Berlin for **ebay.de**.

2. $G(1) > G(2) \simeq G(3) \simeq G(4)$

For websites in Fig. 3(3) and 3(4), their variation trends generally follow this leaking pattern.

In this leaking pattern, guessing probabilities in scenario 1 are much higher than those in other scenarios, which are similar from each other.

Unlike in leaking pattern 1, where guessing probabilities maintain on a high level as long as the cookies are retained, in this leaking pattern the high-level guessing probabilities only come from scenario 1 when both cookies and logged user accounts were kept. This indicates that without logged user accounts, user identities are not revealed even when the cookies exist. It suggests that some information related to the logged user accounts may be propagated to the indirect transitions, but not via cookies.

Now consider the variation trends depicted in Fig. 3(3) for **bankofamerica.com**.

The one in Stuttgart is exceptional, where the guessing probabilities in all scenarios are 0.02, identical to the prior chance.

To further investigate, communications with website **bankofamerica.com** were repeatedly performed several more rounds in Stuttgart during a large time period. However, identical guessing probabilities were always obtained in every testing round. Hence the user identity is not leaked through the indirect transitions when testing in Stuttgart.

3. $G(1) > G(2) > G(3) \simeq G(4)$

This leaking pattern is summarised from the variation trends for internal websites **google.com.br** and **google.it** in Fig. 3(5) and 3(6) respectively.

The guessing probabilities gradually decline over the first three scenarios and then level off in scenario 4, close to the value in scenario 3.

Compared with the guessing probabilities for the external websites shown in Figure 3(1-4), in this case when communicating with internal websites, guessing probabilities in scenario 2 are higher than those in scenario 4, which do not conform to the expectation where the guessing probabilities in scenarios 2 and 4 are similar.

Therefore, a suspicion is proposed when communications happened between Google internal websites. Google wanted to get the information about user accounts in communications. Assume that during the direct transition when a user logged into her user account, a temporary session, containing the information related to the user account and the IP address of the user, was stored in the server.

When an indirect transition was performed in scenario 3, Google detected that the cookie is available. Hence it did not request any information related to the user account as the cookie has existed and it would be used in future communications. As regards in scenario 4, no information related to the user account can be requested, therefore the guessing probabilities in scenarios 3 and 4 are close.

However, when the cookie was deleted in scenario 2, Google attempted to get the information about the user account from a different channel.

It can request for the temporary session from the direct transition saved in the server, through the match between the user's IP address in the indirect transition and that stored in the temporary session. This process can lead to the web traffic for indirect transitions varying depending on users.

It is supposed, therefore, that even when the cookies are deleted, communications between Google internal websites can still implicitly transmit the information related to user accounts. This may cause a side-channel leakage of user identities.

As can be seen that the guessing probabilities for **google.it** are more stable than for **google.com.br**, we suppose that the communications with **google.it** happened within the Europe, while the communications with **google.com.br** were the global transmission to Brazil, which may cause the web traffic with a higher unsteady.

Therefore, it can be supposed that user identities are leaked through communications with internal websites, even when the cookies are deleted.

From the variation trends of guessabilities shown in Figure 3, we make a conclude that it is not a coincidence when a website follows a same variation pattern. The results indicate that

the control variables, i.e. the cookies cleared in scenario 2 and the user accounts logged out in scenario 3, can be the essentials of providing web traffic the capability of fingerprinting users.

As shown in Figure 3, the average increment of guessing probabilities, i.e. the maximum gap between scenarios 1 and 4, is around 0.25 when communicating with an external website. And it is around 0.3 when connecting to an internal website.

Moreover, we also analysed the vulnerabilities from communications without consistent variation trends.

5.3 Leaking User Locations

When analysing the web traffic which leaks no user identities, it is by accident discovered that there is a website whose observations completely rely on the locations. More specifically, the observations from one location always terminate with the same packet sequence, which is unique from those in other locations.

In the communications with website **craigslist.org,** one of the 25 external websites, the web traffic for indirect transitions is constant and indistinguishable from users, so that no user identities are leaked. However, the observations generated from Berlin end with a packet with size 412 in any scenarios, and the observations in Cambridge, London, Neuchatel, and Stuttgart always terminate with packet sizes 421, 415, 408 and 418 respectively.

To confirm, several more rounds of testing on this website were conducted during a large period of time. As a result, the observations always end with the same packet sizes, which are unique from locations.

This discovery is exciting, but at the same time questions like “will the observations still leak user locations when the number of testing locations increase largely", “is the location really the factor leading to the uniqueness of ending packet sizes in observations", etc. can be asked and wait for further investigation.

Inspired by this finding, we examined the web traffic for the six websites in Fig. 3, in terms of the leakage of user identities. However, the observations are independent on the five testing locations.

6 Discussion

This work starts an in-depth analysis of the side-channel leakage of user identities. It demonstrates a potential leaking threat which reveals user identities from Google accounts to external websites in the real world. Moreover, user location is also identifiable by traffic analysis in our experiments.

We also analyse the effects of cookies and logged Google accounts on the web traffic varying depending on user identities. It is not a surprise to see that cookies leak user privacy, as it has been a well-known mean of web tracking. However, to our best knowledge, this research is the first study on cookies in side-channel leakages of user identities via traffic analysis.

On the other hand, one may come up with an argument like “is the increment of guessing probability of around 0.3 large enough to indicate that the user identity is leaked". Moreover, questions like “is the sample space too small compared with millions of Google accounts", “does the guessing probability reduce rapidly when the sample space expands" etc. are still open.

Opposite to previous work which quantifies leakages by simulating real attacks, we simply use the trained data to evaluate leakages, without mounting attacks. Furthermore, when

assessing whether a maximum gap between the guessing probabilities is large enough, we do not establish a threshold of the existence of leakage. Instead we subjectively determine whether the gap is large enough to indicate the leakage of user identities.

Even though these limitations exist, the purpose of this research is achieved. It is not to examine how serious the leakages in real-world web applications. Instead a primary aim is to illustrate a new traffic-analysis threat in terms of user identities, based upon user accounts on a website like Google, where the privacy may be leaked through communications with external websites.

In theory, this work gives a **null** pattern for observations which do not follow the non-null patterns. The null pattern completes the construction of a “one to many" mapping between each transition and multiple traffic patterns. Although the null pattern is weak to present the traffic features, it is a beginning in the multiple mapping between transitions and observations, which is closer to the cases in real-world communications. Hence it is an advance towards a more precise evaluation of traffic-analysis vulnerabilities in real-world web applications.

On the whole, this research demonstrates a new security threat in side-channel attacks in web applications. Instead of closing a case, this research, as believed, opens a new case towards the in-depth investigation of side-channel vulnerabilities in communications with external websites.

7 Related Work

7.1 Side-Channel Attacks in Web Applications

There is a long standing interest in side-channel attacks against encrypted web traffic. In 1996, Wagner and Schneier (Wagner & Schneier, 1996) first proposed the idea of information leakage through traffic analysis from a personal communication. Cheng a Avnur (1998) present the first actual experiment which discloses that user browsing histories of web page can be revealed by observing the highly unique HTML file sizes from the encrypted traffic.

From then on, a large amount of work has studied the attacks, via traffic features, against user privacy, such as (Cai et al., 2012; Danezis, 2010). Moreover, timing attacks have also been examined, e.g. (Felten & Schneider, 2000). Felten a Schneider (2000) propose a cache-based timing attack against the response time of accessing a web page, to determine if the web page has been visited.

In 2010, Chen (2010) turned focus of side-channel attacks from inferring web pages/sites visited to obtaining the user sensitive data on a website. They investigate real-world websites and find out that user health conditions, financial statuses, search inputs, etc. can be largely leaked through traffic analysis, based on the packet sizes and directions.

The first automated tool – Sidebuster, for detecting and quantifying side-channel leakages of user sensitive information, is proposed by Zhang et al. (Zhang, 2010), based on the source code of Struts-based web applications. Then Chapman and Evans (Chapman & Evan, 2011) propose a black-box analysis for detecting side-channel vulnerabilities in real-world web applications. In 2013, Huang and Malacaria (2013) advanced towards the automated test case generation in the detection of traffic-analysis vulnerabilities in web applications.

7.2 Fingerprinting Users

In anonymity networks which particularly hiding identities of endpoints, practical traffic analysis have been proposed for compromising user identities, e.g. (Chakravarty, Stavrou, & Keromytis, 2010). In this work, an attacker exploits traffic fluctuations from characterising bandwidth variations to compromise the anonymity of Tor users.

Device fingerprinting has also been concerned in terms of breaking user anonymity. For example, (Eckersley, 2010) shows that trackers can recognise users without the needs of cookies, through fingerprinting the properties of browsers and their plug-ins.

There has been an increasing public concern about the disclosure of online user identities with the widespread usage of social media, e.g. (Wondracek et al., 2010). Wondracek et al. (2010) introduce a de-anonymising attack triggered by an evil website a user visits, which examines the group memberships of a user on social networking sites, i.e. the groups in a social network to which the user belongs. The information of group memberships is obtained by mounting a web browser history stealing attack. Then the attacker uses the group memberships to identify users.

In this work, however, a novel traffic-analysis scenario of fingerprinting users is proposed. It is, as believed, a more common scenario of inferring user identities. User identities, based upon user accounts on a website like Google, can be recognised via traffic analysis in communications with third-party websites.

8 Conclusion and Future Work

This work demonstrates Google users can be identified via traffic-analysis attacks. User identities are leaked through transitions which appear to be irrelevant to users, via the communications with third-party websites. We also discover that cookies and logged user accounts can be the sources of fingerprinting users.

An immediate work will be to investigate leakages of user locations in depth. Future work can determine if the user location can be leaked through traffic analysis, and check what sources lead to varying web traffic.

In terms of communications with third-party websites, this research analysed leakages when the maximum distance of guessing probabilities coming from scenarios 1 and 4. However, user privacy is possibly revealed from communications when the maximum gap between guessing probabilities is derived from other two scenarios. Hence future work can be extended to analyse leakages in communications with maximum distances of guessing probabilities from any two scenarios. Furthermore, we will simulate real-world attacks on websites with state of the art browsers.

References

Alexa. (2015). *Alexa Top websites.* Retrieved from http://www.alexa.com/topsites

Baum, L. E., & Petrie, T. (1966). Statistical inference for probabilistic functions of finite state Markov chains. *The annals of mathematical statistics*, 37(6), 1554-1563.

Baum, L. E., & Eagon, J. A. (1967). An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology. *Bulletin of the American Mathematical Society*, *73*(3), 360-363. doi: 10.1090/S0002-9904-1967-11751-8

Baum, L. E., Petrie, T., Soules, G., & Weiss, N. (1970). A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *The annals of mathematical statistics*, 41(1), 164-171.

Bland, J. M., & Altman, D. G. (1996). Statistics notes: measurement error. *British Medical Journal*, *313*(7059), 744.

Cai, X., Zhang, X. C., Joshi, B., & Johnson, R. (2012). Touching from a distance: Website fingerprinting attacks and defenses. In *Proceedings of the ACM conference on Computer and communications security* (pp. 605-616). New York: ACM.

Chakravarty, S., Stavrou, A., & Keromytis, A. D. (2010). Traffic analysis against low-latency anonymity networks using available bandwidth estimation. In *Proceedings of the 15th European conference on Research in computer security* (pp. 249-267). Berlin: Springer.

Chapman, P., & Evans, D. (2011). Automated black-box detection of side-channel vulnerabilities in web applications. In *Proceedings of the 18th ACM conference on Computer and communications security* (pp. 263-274). New York: ACM.

Cheng, H., & Avnur, R. (1998). *Traffic analysis of ssl encrypted web browsing*. Retrieved from https://www.cs.berkeley.edu/~daw/teaching/cs261-f98/projects/final-reports/ronathan-heyning.ps

Danezis, G. (2010). Traffic Analysis of the HTTP Protocol over TLS. Retrieved from http://research.microsoft.com/en-us/um/people/gdane/papers/TLSanon.pdf

Eckersley, P. (2010). How unique is your web browser?. In *Proceedings of the* 10th international conference on Privacy enhancing technologies (pp. 1-18). Berlin: Springer.

Felten, E. W., & Schneider, M. A. (2000). Timing attacks on web privacy. In *Proceedings of the 7th ACM conference on Computer and communications security* (pp. 25-32). New York: ACM.

Forney Jr, G. D. (1973). The viterbi algorithm. *Proceedings of the IEEE*, *61*(3), 268-278. doi: 10.1109/PROC.1973.9030

Huang, X., & Malacaria, P. (2013). SideAuto: quantitative information flow for side-channel leakage in web applications. In *Proceedings of the 12th ACM workshop on Workshop on privacy in the electronic society* (pp. 285-290). New York: ACM.

Backes, M., Doychev, G., & Kopf, B. (2013). Preventing side-channel leaks in web traffic: A formal approach. Retrieved from http://www.internetsociety.org/sites/default/files/04_2_0.pdf

Jpcap. (2015). *Jpcap*. Retrieved from http://jpcap.sourceforge.net/

Malacaria, P. (2015). Algebraic foundations for quantitative information flow. *Mathematical Structures in Computer Science*, 25(2), 404-428. doi: 10.1017/S0960129513000649

Navarro, G. (2001). A guided tour to approximate string matching. *ACM computing surveys, 33*(1), 31-88. doi: 10.1145/375360.375365

PlanetLab. (2015). *PlanetLab Euorpe.* Retrived from https://www.planet-lab.eu/

Selenium. (2015). *Selenium Webdriver.* Retrieved from http://docs.seleniumhq.org/projects/webdriver/

Smith, G. (2009). On the foundations of quantitative information flow. In *Proceedings of the 12th International Conference on Foundations of Software Science and Computational Structures* (pp. 288-302). Berlin: Springer. doi: 10.1007/978-3-642-00596-1_21

Wagner, D., & Schneier, B. (1996). Analysis of the SSL 3.0 protocol. In Proceedings of the 2nd conference USENIX Workshop on Electronic Commerce (pp. 29-40).

Wondracek, G., Holz, T., Kirda, E., & Kruegel, C. (2010). A practical attack to de-anonymize social network users. In *IEEE Symposium on Security and Privacy* (pp. 223-238). New York: IEEE.

Zhang, K., Li, Z., Wang, R., Wang, X., & Chen, S. (2010). Sidebuster: automated detection and quantification of side-channel leaks in web application development. In *Proceedings of the 17th ACM conference on computer and communications security* (pp. 595-606). New York: ACM.