Side Channel Attacks in the Browser

Case for Support

Background

Side-Channel Attacks (SCA) are an important class of cryptanalytic techniques which target unintentional information leakage in order to disclose private data. In general, Side-Channel attacks monitor a device performing an interesting operation and try to find a correlation between the side-channel information and the internal state of the processing device, which is related to the secret parameters involved in the computation. Side-channel information can be obtained by monitoring variations in the execution time (*Timing Attacks*), the power consumption (*Power Analysis*), the electro-magnetic emission (*Electromagnetic Analysis - EMA*), or the state of the cache memory (*Cache Attacks*) [1]. The goal of an attacker in side-channel attacks is to either recover the secret parameters involved in the computation by targeting faults in the implementation of some cryptographic algorithm, or disclose private information about a user, such as their location or browsing history. Side-channel attacks exploit vulnerabilities in a specific implementation rather than targeting an abstract algorithm, making them less generic but more powerful [1].

The expansion of the Internet determined browser vendors to add more and more features to their software in order to offer richer, dynamic and more interactive web pages. The evolution of browser software not only improved the overall user experience but also opened the way for a new category of side-channel attacks known as browser attacks. These types of attacks target vulnerabilities in the browser software. Browser attack fall into two categories: traditional timing attacks which measure the time it takes the server to respond, and cache timing attacks which measure how long it takes to retrieve a resource from the browser's cache memory.

Browser attacks are a very powerful category of side-channel attacks, which require minimal set up and affect a wide range of users. In general, these types of attacks require a user to navigate to an untrusted website, which contains attacker-controlled content. The web page has a background script which collects information about the victim which can be used by the attacker to infer a user's location, browsing history, or even current browsing activity. The victims are usually unaware that the website is collecting private data about them. One might think that such attacks can be avoided by simply not visiting untrusted websites; however, researchers discovered that even high-profile websites are using similar techniques to collect information about their visitors[2]. For example, a health insurance provider might want to know if the user has previously visited any health related web pages, this can easily be achieved using a side-channel attack.

Vulnerabilities in browser software emerge every day, but they also disappear within weeks. This gives attackers a limited time frame in which they can exploit potential vulnerabilities in web browsers. Although, most browser vulnerabilities get patched immediately, there are side-channel attacks in the browser discovered over fifteen years ago which are still possible.

In 2000, Felten and Schneider[3] presented the first browser side-channel attack, where they were able to determine if the user has previously visited a web page. A few years later, Bortz et al.[4] used a similar technique to determine the size of a hidden file, which can disclose private information about the user.

Their work was extended by Van Goethem et al.[5], who used newer web development technologies in order to determine the size of a remote origin resource. Browser attacks are not only used to determine how much information a user has access to, in 2015 Jia et al.[6] demonstrated that it is possible to determine a user's location. Additionally, Oren et al.[7] showed that it is possible to disclose a user's current browsing activity by analysing the state of their browser's cache memory.

Motivation

The aforementioned papers present various techniques for disclosing private information about users, by either measuring the amount of data the users have access to, or simply determining what pages they previously accessed by checking the state of their browser's cache memory. Additionally, most of the papers present real world scenarios where the attacks can be used to de-anonymize a user; however, the attacks are presented from the perspective of a malicious user who is trying to gain access to private data.

The world of browser attacks is rapidly changing with new vulnerabilities being discovered and patched every week. In general, adversaries have a limited time frame in which they can exploit the newly found vulnerabilities. In general, vulnerabilities found in the browser software are fixed within weeks; however, sometimes the measures proposed for stopping the attacks will bring severe performance penalties and browser vendors refuse to include them in their software, or a countermeasure is yet to be found. This is the case of the attacks discovered by Felten et al.[3] and Bortz et al.[4], which have been around for years.

This motivated us to investigate new applications for this particular kind of browser attacks. Our research will explore how side-channel information can be used to obtain a user's list of connections on social networking websites. In their paper, Van Goethem et al.[5], briefly mention a real world application of the browser attack in which they can determine if two people are friends on Facebook, or connected on LinkedIn. We are going to use their idea and try to efficiently determine a user's full list of contacts on any social media platform.

Project Objectives

The aim of this research is to provide theory, tools and techniques for disclosing a user's list of connections on social media platforms. More specifically, we will focus on the following objectives:

1. To analyse all known browser attacks which can be used to determine if two users are friends and determine which technique performs the best.
2. To build a basic model which determines a user's connections on social media websites.
3. To test the model on data provided by security companies. This will allow us to improve the algorithms.
4. To test the model in the real world. We will to determine a user's list of connections on real social medial website.

Programme and Methodology

Throughout this section we will refer to the list of a user's friends or connections on social media platforms as the **friend graph**.

Workpackage 1 (WP1) : A preliminary analysis of published attack vectors

Research leader: Dr. Ana Dumitras, University of Bristol

Principle research objective:

> To review the existing literature on side-channel attacks in the browser and determine the efficiency of the attack vectors.

Principle deliverables:

1. Provide implementations for different browser attacks, optimised to work on the most recent versions of browser software.

The goal of this package is to review existing attack vectors and analyse their efficiency in state-of-the-art browser software. Browser attacks are a particular kind of side-channel attacks, where most vulnerabilities appear and disappear within weeks. This gives adversaries a limited time frame in which they can exploit potential vulnerabilities. There are very few attacks which have been around for years, like [3] and [4], because there is no known countermeasure which will not significantly affect the performance of the browser software.

Initially, this work package will focus on selecting a series of side-channel attacks in the web browser to be reviewed. We will use Semantics Scholar [8] and Google Scholar[9] to identify the set of papers to be reviewed. In order to determine the friend graph in later WP3 and WP4 we need to select a specific set of side-channel attacks in web browsers. The criteria for selecting the papers is that they can be applied to real world scenarios where we are trying to determine the size of a remote file. Such attacks have been around for years and there are lots of resources available. The role of the researcher working on this work package is to implement the attacks and alter them to work with current versions of the browser software.

This work package is to be completed by a PhD student, under the supervision of the Principal Investigator. The attack vectors which will be reviewed are to be determined by the PhD student together with the supervisor.

Workpackage 2 (WP2) : Build a model for answering yes-no questions about the user.

Research leader: Dr. Brian May, University of Bristol

Principle research objective:

> To build a web applications which is able to answer yes or no questions about the victim's profile on social network websites.

Principle deliverables:

1. Provide a web application which can be used to determine information about the victim.
2. A collection of tests for analysing the performance of the web application.

This work package builds on the work from WP1, where we determine a set of suitable methods for estimating the size of a remote file. The goal of this work package is to build a web application which is able to infer certain information about a social media website user, such as group membership or if they have access to specific resources on the social media platform. More specifically, given a link to a resource on the social media website, our web application should be able to tell if the user has access to

that resource or not by using techniques from WP1. This is similar to asking the application a series of questions whose answer's can only be yes or no.

The work is to be carried by a PhD student, under the supervision of the co-Investigator, who has experience in web applications and social media platforms. The criteria for success will be how our web application guesses is certain facts about the user are true or false. Additionally, the application should be able to check if two users are connected, this would later be used in WP3.

For ethical reasons we are not going to use data from actual users, but we are going to set up several accounts on the most popular social networks, like Facebook, Twitter or LinkedIn, and test our application on those accounts. These accounts will also be used for testing since we already know what answers to expect from the application.

Workpackage 3 (WP3) : Build model for determining the friend graph

Research leader: Dr. Brian May, University of Bristol

Principle research objective:

> To extend the application developed in WP2 such that it works on a small size social network website.

Principle deliverables:

1. Provide a web application which discloses the friend graph on a small size social network.

This work package focuses on disclosing the friend graph in a small size social network platform. As mentioned in WP2, for ethical reasons we are should not test out web application on real world social media platforms, like Facebook or LinkedIn, so the Defence Science and Technology Laboratory (DSTL) has agreed to provide us access to a internal social media platform where we are going to test our web application.

The aim of this work package is to have a web application which automatically displays the friend graph when given a link to a user's profile on DSTL's social network platform. The work is to be completed by a PhD student, supervised by the PI. The success of this work package is measured in how accurately we can determine the friend graph on the given social network.

Workpackage 4 (WP4) : Test the model on real world social networks

Research leader: Dr. Brian May, University of Bristol

Principle research objective:

> To build a web application which discloses the friend graph of a user on a real world social media website.

Principle deliverables:

1. Provide a web application for obtaining the friend graph of a user on a real world social media website.

References

[1] François-Xavier Standaert. Introduction to side-channel attacks. In *Secure Integrated Circuits and Systems*, pages 27–42. Springer, 2010.

[2] Dongseok Jang, Ranjit Jhala, Sorin Lerner, and Hovav Shacham. An empirical study of privacy-violating information flows in javascript web applications. In *Proceedings of the 17th ACM conference on Computer and communications security*, pages 270–283. ACM, 2010.

[3] Edward W Felten and Michael A Schneider. Timing attacks on web privacy. In *Proceedings of the 7th ACM conference on Computer and communications security*, pages 25–32. ACM, 2000.

[4] Andrew Bortz and Dan Boneh. Exposing private information by timing web applications. In *Proceedings of the 16th international conference on World Wide Web*, pages 621–628. ACM, 2007.

[5] Tom Van Goethem, Wouter Joosen, and Nick Nikiforakis. The clock is still ticking: Timing attacks in the modern web. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1382–1393. ACM, 2015.

[6] Yaoqi Jia, Xinshu Dong, Zhenkai Liang, and Prateek Saxena. I know where you've been: Geo-inference attacks via the browser cache. *Internet Computing, IEEE*, 19(1):44–53, 2015.

[7] Yossef Oren, Vasileios P Kemerlis, Simha Sethumadhavan, and Angelos D Keromytis. The spy in the sandbox–practical cache attacks in javascript. *arXiv preprint arXiv:1502.07373*, 2015.

[8] Semantics scholar. `https://www.semanticscholar.org/`. [Accessed: 2016-05-18].

[9] Google scholar. `https://scholar.google.co.uk/`. [Accessed: 2016-05-18].

Budget

a one-page (maximum) Budget: a table/spreadsheet of expenditure

Item	Cost for 3.5 years (£)
PI	350,000
CI	87,500
PDRA	350,000
PhD student	100,000
Equipment	3,000
Cloud computing resources	27,452
International travel costs	28,000
National travel costs	10,500
Workshops	1,800
Total cost	958,252

Justification For Resources

a one-page (maximum) Justification For Resources: essentially a written narrative on why you need the expenditure on each line-item in your budget

Impact Statement

a one-page (maximum) Impact Statement: a description of how you intend to ensure that your work makes a difference in the world, rather than sitting on a shelf gathering dust)

Workplan

and also a one-page (maximum) Workplan:typically a GANTT chart or similar diagram indicating the order in which workpackages are carried out)