ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení:	**Ondráčková**	Jméno: **Lada**	Osobní číslo: **406362**
Fakulta/ústav:	**Fakulta elektrotechnická**		
Zadávající katedra/ústav:	**Katedra počítačů**		
Studijní program:	**Otevřená informatika**		
Studijní obor:	**Umělá inteligence**		

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Detekce on-line reklamních podvodů monitorováním síťových dat

Název diplomové práce anglicky:

Online Advertising Fraud Detection Via Network Traffic Monitoring

Pokyny pro vypracování:

1.Survey the state of the art approaches to online advertising frauds and methods of their detection.
2.Study the network communication processes of the advertisement delivery using the information available in the HTTP access logs.
3.Propose network behavioural features that can be used for advertising fraud detection.
4.Develop a model for behavioural detection of advertising frauds.
5.Evaluate proposed model using real network data captured from a number of various networks.

Seznam doporučené literatury:

[1] Aggarwal, Charu C. "Outlier analysis." Springer International Publishing, 2013.
[2] Stringhini, Gianluca, Christopher Kruegel, and Giovanni Vigna. "Shady paths: Leveraging surfing crowds to detect malicious web pages." Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security. ACM, 2013.
[3] Li, Zhou, et al. "Knowing your enemy: understanding and detecting malicious web advertising." Proceedings of the 2012 ACM conference on Computer and communications security. ACM, 2012.
[4] Hua, Jingyu, An Tang, and Sheng Zhong. "Advertiser and publisher-centric privacy aware online behavioral advertising." Distributed Computing Systems (ICDCS), 2015 IEEE 35th International Conference on. IEEE, 2015.

Jméno a pracoviště vedoucí(ho) diplomové práce:

Ing. Martin Grill, Ph.D., katedra počítačů FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **21.12.2017** Termín odevzdání diplomové práce: ____________

Platnost zadání diplomové práce: **30.09.2019**

[illegible]
podpis vedoucí(ho) práce

[illegible]
podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomantka bere na vědomí, že je povinna vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

Datum převzetí zadání Podpis studentky

master's thesis

Online Advertising Fraud Detection Via Network Traffic Monitoring

Bc. Lada Ondráčková

May 2018

supervisor: Ing. Martin Grill, Ph.D.

Czech Technical University in Prague
Faculty of Electrical Engineering, Department of Computer Science

Acknowledgement

I would like to thank my supervisor, Ing. Martin Grill, PhD., for his patient guidance, willingness and assistance during the writing of my thesis. I would like to thank my family, for the support in my studies.

Declaration

I declare that I worked out the presented thesis independently and I quoted all used sources of information in accord with Methodical instructions about ethical principles for writing academic thesis.

Abstract

Stále se zvyšující zájem o Internet ovlivňuje i nárůst reklam. Skoro všechny webové stránky obsahují nějakou formu reklamy, která umožňuje uživatelům přístup k bezplatnému obsahu a vlastníci stránky si tím vydělají na její provoz. Obecně je reklamní byznys místo, kde se soustředí velké množství financí.

Rostoucí zisky z internetové reklamy přitahují čím dál více útočníků, kteří aktivně hledají nové zranitelnosti systémů, aby mohli jejich uživatele infikovat malwarem.

Zisky útočníků jsou zejména spojeny s počtem falešných kliknutí na reklamy, což je motivuje vyvíjet nebezpečný software. Ten generuje velké množství požadavků simulující tato kliknutí. Požadavky zatěžují internetové spojení, zpomalují zařízení oběti, negativně ovlivnňují její zážitky strávené na internetu a mohou narušit její soukromí.

V této práci jsme navrhli detektor anomálií, jehož důležitou částí je rozpoznání reklamy v síťovém provozu. Ukázali jsme, že běžně používané listy pro blokovani reklamy, jako Yoyo ad-block list nebo EasyList ad-block list, jsou velice limitované a těžko se rozšiřují o nové reklamní sítě. Proto jsme navrhli algorithmus pro detekci reklamního provozu, který je schopný odhalit desetkrát více reklam než dříve zmíněné blacklistovací listy. Navíc nevyžaduje žádný lidský zásah ani ruční doplňování nové znalosti, protože může být opakovaně spouštěn a objevit tak nové reklamní sítě.

Nakonec jsme navržený detektor anomálií otestovali na HTTP a HTTPS z reálného síťového provozu shromážděného z firemních sítí.

Klíčová slova

Detekce anomálií, Detekce reklamy, Adware, Click fraud

Abstract

The explosive growth in the size and use of the Internet is followed by the growth of ads. Almost all popular web pages contain advertisement, which plays a vital role in supporting free websites by generating revenue for placing the advertisement. Therefore, the online advertisement can be a lucrative business for a website with high visibility.

The increasing revenue of the online advertisement attracts more and more attackers who are interested to explore new vulnerabilities to find new ways to infect the broad audience with adware.

The attacker's revenue is mainly related to the amount of fake impression, which motivates them to create malware that generates a large number of advertisement requests. This wastes the bandwidth, it can slow down user's device, and negatively impacts the user experience while browsing the Internet and disturbs the user's privacy.

We propose anomaly detector for advertisement fraud detection on the network level from the network behaviour. The important part of the proposed anomaly detector is advertisement traffic detection, where we show that current blacklisting methods are limited as they can not effectively cover the increasing number of ad-servers and ad-networks. We propose an algorithm to detect ad-related HTTP/HTTPS requests captured by web proxies. This algorithm can detect ten times more ads than the conventional pattern matching approaches like Yoyo and EasyList ad-block lists, and will not need any human interventions nor manual updates. It can be periodically run to detect new ad-servers and ad-networks.

Finally, we evaluated the proposed anomaly detector on HTTP and HTTPS traffic from real network data collected in a number of corporate networks.

Keywords

Anomaly detection, Advertisement detection, Adware, Click fraud

Contents

1 Introduction

The explosive growth in the size and use of the Internet gives the advertisers opportunity to reach significantly more consumers through online advertisement compared to traditional advertising. The amount of online advertisement is significantly growing every year as the advertisement is an essential part of the Internet because it drives it's economy. Almost all popular web pages contain advertisement, which plays a vital role in supporting free content websites. Even the advertiser's community grows every year as it is easy for anybody to create an account with major providers, such as DoubleClick and the advertisement can be immediately promoted.

The advertisers are motivated to use online advertisement because the process of advertisement delivery to the audience is becoming more effective and more sophisticated, as it is based on user behaviour on the Internet. This helps advertisers promote effectively and target important part of potential customers. This leads online advertising to grow into multi billion-dollar business [1].

The increasing revenue of the online advertisement attracts more and more attackers who are interested to explore new vulnerabilities to find new ways to infect the broad audience with the adware. Malware monetisation via fraudulent advertisement traffic is lucrative, relatively easy, and growing structure of the Internet. Online ad-networks, which are complex and generally not well described, but well funded, opens new opportunities for fraudulent activities.

Once the attackers find the way how to infect a user with adware, there is an increased chance that the adware escalates to higher severity threats.

However, the attacker's revenue is mainly related to the amount of fake impression, which motivates them to create malware that generates a large number of advertisement requests, which wastes the bandwidth. Subsequently, the high number of advertisement requests can slow down user's device and negatively impact user experience while browsing the internet and disturb the user's privacy.

Whole advertisement fraud is driven by the conflict between different parties in advertisement delivery process, where each part wants to maximize it's profit while other parts are harmed by such behaviour.

For the unaware user, it is relatively easy to get infected because the infection can be hidden even on popular web pages. Popular examples include the Yontoo browser plugin which modified 4.5 million user's private Facebook sessions to include an advertisement that earned Yontoo $8 million [2]; and the New Your Time's malvertising incident, in which fake virus scanner was found on its home page [3].

Hence, it is important to detect these threats to be able to recognize them before they cause vast damages.

In the first section of this chapter, we discuss the importance of online advertisement by presenting its scale in numbers and online advertisement fraud in numbers. In the second part of this chapter, we will summarize the reasons to detect online advertisement, set up the problem and our goal in this area.

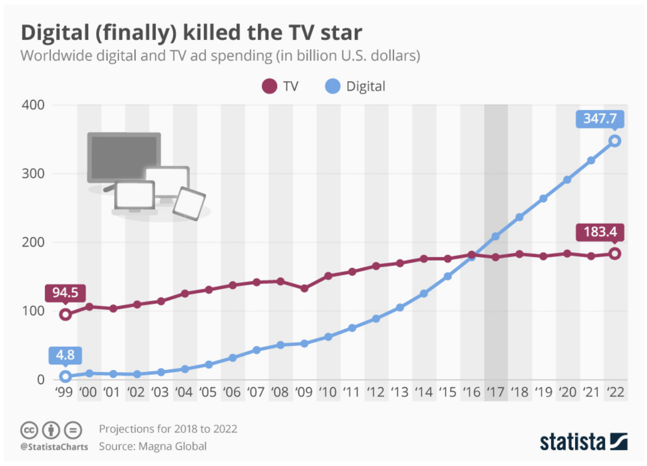


Figure 1 Digital ad surpassed TV ad in 2017 [4].

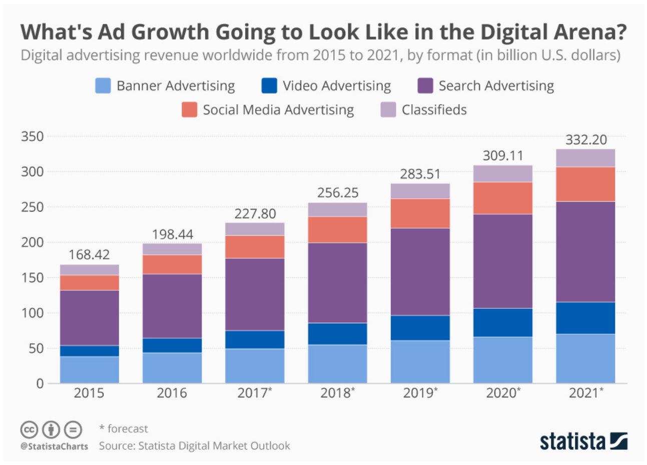


Figure 2 Different types of online advertisement distribution [5].

1.1 Advertisement in Numbers

The increasing importance of the online advertisement supports the fact that the online ad spends reached \$209 billion worldwide in 2017. It was an important milestone because it finally surpassed TV advertisement which spent \$178 billion, according to the research agency Magna Global. It is expected that this trend in the online advertisement will continue in following years and possibly reach \$348 billion in 2022, while TV will stall as it is shown in Figure 1.

The advantage of online advertisement is that it is targeted to a specific user and their needs and thus, it has better payback rate. The search advertisement, placed on web pages that show result from search engine queries, takes the biggest chunk of digital advertisement revenue, making up 40 percent share of the total revenue. The social media advertisement, placed on social media web pages, and banner advertisement, placed on the side, top, and bottom sections across different websites, follow with significant distance as it is shown in Figure 2.

1.2 Advertisement Frauds in Numbers

Fraud is an inseparable part of the online advertisement. The amount of global advertising revenue wasted on fraudulent traffic, or clicks automatically generated by bots, could reach $16.4 billion in 2017, according to a new study commissioned by WPP and cited by Business Insider [6]. Some unofficial resources estimate even more. Nevertheless, it is more than double the $ 7.2 billion loss in ad fraud in 2016 according to Association of National Advertisers estimation [7]. Whatsoever, the World Federation of Advertisers predicted in 2016 that the advertisement spend would be $ 50 billion by 2025 [6].

1.3 Problem Setting

The problem of detecting adversarial advertisements is complicated by scale. With millions of advertisers and billions of advertiser landing pages, the automated detection methods are needed.

Our goal is to detect advertisement fraud on the network level where we have an access to HTTP proxy logs later described in Chapter 5.1. We want to be able detect users who are infected with malware, which generates a large number of advertisement requests.

1.4 Goals of the Thesis

This main goals of this thesis are:

Survey state of the art approaches to online advertising frauds and methods of their detection.

The related work in analysis and detection techniques of advertisement frauds is reviewed in Chapter 2. However, the amount of the research in this field is very limited and mainly focused on the fraud detection from the ad-provider point of view.

Study the network communication processes of the advertisement delivery using the information available in the HTTP access logs.

The online advertisement and delivery mechanism is depicted in Chapter 3. We describe the mechanism of ad-exchange as it is currently the most popular approach of advertisement delivery. In Chapter 4 we reverse the advertisement loading process to get a hands-on overview about advertisement loading and redirection process.

Propose network behavioural features that can be used for advertising fraud detection.

We propose method for advertisement detection in Chapter 6.4 which is later use for ratio of number of advertisement flows in the user's traffic and total number of users flow. These features are used for modelling for behavioural detection of advertising frauds,

Develop a model for behavioural detection of advertising frauds.

The model for behavioural detection of advertising frauds is described in Chapter 9. We model the behaviour as the percentage of the ad-related traffic for each user in the monitored network. The ratio of the individual user is compared to the ratios of all other users in the network to asses the anomaly score of the user.

Evaluate proposed model using real network data captured from a number of various networks.

Finally, we evaluated the proposed solution in Chapter 7.

2 State of the Art

In this chapter, we review the related work in analysis and detection techniques of advertisement frauds. The amount of research in this field is very limited and mainly focused on the fraud detection from the ad-provider's point of view. For the sake of completeness we are going to cover those also even though they are not directly relevant to this thesis.

2.1 Knowing Your Enemy: Understanding and Detecting Malicious Web Advertising

Zhou Li *at al.* [8] analyse ad-related Web traces crawled over a three-month period. They discover that top ranking Web sites fall victims of advertising fraud and leading ad networks such as DoubleClick are infiltrated. They analyse the advertisement fraud and show examples of advertisement fraud campaigns.

They study URL redirection chains in their analysis of advertisement fraud where each URL is a node of redirection chain. They examine properties of advertisement fraud paths searching for any node on the advertisement delivery path that performs malicious activities. They identify prominent features from malicious advertising nodes and their related content delivery paths. They annotate the set of features of one delivery path as fraudulent if it was at least on of the URL from the redirection path recognised as malicious by Google Safe-Browsing API or Microsoft Forefront 2010. They divide available data set of features for obtained delivery paths into training and testing data set and use it to for testing and training of proposed detection system MadTracer, which adopt a statistical learning framework based on decision trees to automatically generate a set of detection rules.

This approach cannot be applied to our problem because their data are captured from the browser and they use the information from the content of Web pages. They crawler over selected Web pages and they can derive what is the advertisement in the Web page from the content and later they use this to build redirection paths only from redirection which are caused by the advertisement. However, they discuss the advertisement redirection property, which is useful knowledge for our research.

2.2 Detecting Adversarial Advertisements in the Wild

D. Sculley *at al.* [9] present a large-scale data mining mechanism that detects and blocks adversarial advertisements. Their detection system combines tiered strategy, automated and semi-automated methods to ensure reliable classification. Their solution was deployed at Google for detecting and blocking adversarial advertisements in 2011.

This study stress following key challenges:

- The cost of both FP's and FN's is high. To address that they combine automated and semi-automated ML principles to drive both rates towards zero.
- The vast majority of ads are from good-faith advertisers, thus detecting adversarial advertisements presents a difficult class imbalances problem.

- They have to rely on human experts for ground truth.

Their detection system learns from seven different types of features. They are based on the text of advertisement keywords, properties and structure of landing page, features extracted from the results of the HTTP fetch of the landing page, information about links and redirects, and advertiser account information.

The proposed machine learning solution is designed as inherently multi-class. It is based on sets of binary-class linear classifiers deployed as per-class classifiers. This solution is mainly design to increase the effective impact of human experts and this solution can reduce up to 50% of human experts time.

Despite, this solution cannot be applied on our problem, they discuss the problems which appear with advertisement fraud detection such as high cost of both FP's and FN's, class imbalance problem and necessity of human experts for ground truth.

2.3 Shady Paths: Leveraging Surfing Crowds to Detect Malicious Web Pages

Gianluca Stringhini *at al.* [10] develop a system, called SPIDERWEB, which detects Web pages that deliver malware. They use the browsers of a collection of web users to record their interactions with websites, as well as the redirections they go through to reach their final destinations. Thus, they do not develop features of a malicious web page; they instead analyse how a large and diverse set of Web browsers reach these pages and finally aggregate the different redirection chains that lead to a specific web page and investigate the characteristics of the resulting redirection graph.

They build redirection graphs from HTTP redirection chains. The redirection graph contains two types of entities, users and Web pages. A user entity includes information about user IP, operating system, browser type and country. A Web page is represented by following features: URL to the web page, the domain of the URL, length of the domain name, IP of Web page and country. Then, they define redirection chain as a tuple of the user visiting the redirection chain and graph of visited web pages and their redirections, referrer (URL user is coming from), landing page (the Web page on which the first redirection happens) and final page (the last Web page in the redirection chain). They build redirection graph bases on this component and develop a set of 28 features that describe the characteristics of a redirection graph. Finally, the SPIDERWEB classifies each redirection graph and decides whether it is malicious.

They believe that SPIDERWEB complements previous work on detecting malicious web pages since it leverages features that are harder to evade (graph-specific features related to advertisement redirection chains). As they collect data from different users, they use anti virus labels of user traffic as their ground truth. Thus, they have labels for each set of extracted features and they use 10-fold cross validation to validate their results. Their achieve F-measure score is equal to 0.881.

We cannot apply this approach because their study uses different data than are ours, but they examine the advertisement redirection chains, which we study in this thesis.

2.4 Click-fraud Monetizing Malware: A Survey and Case Study

Tommy Blizard and Nikola Livic [11] propose detail study of click fraud malware containing study of click fraud motivation and different types of behaviour. They analyse mechanics of how specific malware executed click fraud.

They discover that each bot (infected user) has unique id created from observed information about the victims machine. This id is used for communication with C&C. The studied malware hook internet connections so when a victim searches via Bing or Google it sends an encrypted blob containing the bot id, the search portal, the query string of the search and an affiliate id. The affiliate id is used to track fraudulent clicks generated by this instance of the malware to get credited to a a particular affiliate. Then, the C&C sends a response back that includes a referrer, a URL and an agent. These are used when the user clicks on an advertisement link from the search results and one of the following situation happen. In 30% the user go to actual ad landing page, 20% go to sub-syndicated search portal results, 30% go to non-germane site, and 15% to a different site of similar content. 5% returned error.

The goal of this study is the understanding of malware that monetises via click fraud to help to build prevention systems for malware-generated click fraud in the future.

Nevertheless, they do not propose ML solution for detecting advertisement fraud, however, they present a detailed study of click fraud behaviour, which is crucial knowledge to understand the processes in advertisement fraud.

3 Online-advertisement

The money in online advertisement increase every year, and it makes the online advertisement a lucrative business. Not surprisingly, successful business is followed by fraud and malware monetisation via fraudulent advertisement traffic is lucrative and relatively easy. Unfortunately, fraudulent advertisement traffic decreases the computation power of the infected machine and what more, it can lead to higher severity threats. These impacts on online advertisement are a good reason to study online advertisement fraud.

To correctly understand the various types of online advertisement fraud we need to first understand the legitimate advertisement. The legitimate online advertisement overview is in Section 3.1. The categorisation of different kinds of threats is in Section 3.2.

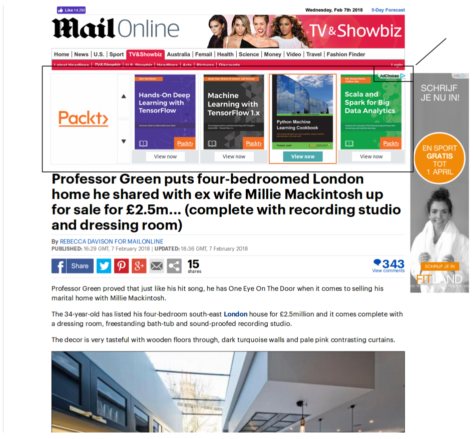
3.1 Legitimate advertisement

The advertisement accompanies a significant amount of Web pages on the Internet. It can appear as a banner advertisement on the side, top, and bottom sections of Web pages. Another forms, which are for many users harder to recognise as an advertisement, are a search advertisement, which is part of the search result of a search engine, and social media paid content.

The online advertisement is so popular because it is advantageous for both sites. It helps to generate revenue for the owners of the Web pages and to promote advertiser's advertisement. Appealing advertising messages, logos, animations, photos and graphics are used to manipulate the audience to make them more receptive to advertised goods.

To increase the advertising effect, advertisers target audiences with particular traits to improve the success rate of the advertising campaign to return their investment. The users need to be profiled according to their interests to be able to serve targeted ads. A common technique for user identification are cookies. Cookies are data stored on user's computers in a browser. Incomplete shopping information, unique user id generated by a Website, data entered into the Web page (e.g. passwords, card numbers). This information can be stored and used for session management, targeted advertisement, personalisation or tracking. For example, a user can be re-targeted with unfinished shopping list retrieved from the cookies while visiting some other pages via ads [12].

To make advertising campaigns more precise, advertisers collect data about user's online activity across multiple external Web pages. One Website can have none up to dozens of trackers which can track the number of visitors, enable specific functionality, or see where the visitor is coming from. The owner can use this information to insight into his Web traffic, but the significant portion of trackers belong to companies whose primary goal is to build up a profile of the user. Third party tracker tries to collect data such as age, location, and interests. The personal data collected by the third party tracker are packed and sold to advertisers or other companies. The advertisers can collect the data and connect particular record by unique identifier for the user such as a fingerprint of user's browser [13]. Based on collected data, advertisers can use


Like 14.2M
Wednesday, Feb 7th 2018
5-Day Forecast
Mail Online
TV&Showbiz
Home | News | U.S. | Sport | TV&Showbiz | Australia | Femail | Health | Science | Money | Video | Travel | Fashion Finder
Latest Headlines | TV&Showbiz | U.S. Showbiz | Headlines | Arts | Pictures | Discounts
Login
AdChoices
Packt>
Hands-On Deep Learning with TensorFlow
Machine Learning with TensorFlow 1.x
Python Machine Learning Cookbook
Scala and Spark for Big Data Analytics
View now
Info
SCHRIJF JE NU IN!
EN SPORT GRATIS TOT 1 APRIL
SCHRIJF JE IN
FITLAND
Share
15 shares
343 View comments

Professor Green puts four-bedroomed London home he shared with ex wife Millie Mackintosh up for sale for £2.5m... (complete with recording studio and dressing room)

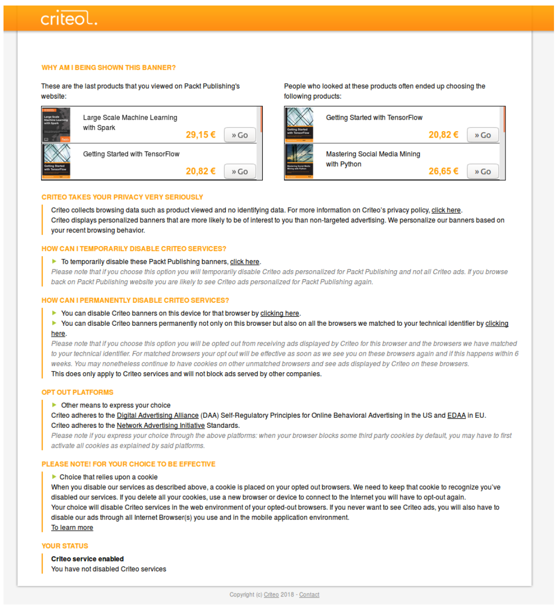
By REBECCA DAVISON FOR MAILONLINE
PUBLISHED: 16:29 GMT, 7 February 2018 | UPDATED: 18:36 GMT, 7 February 2018

Professor Green proved that just like his hit song, he has One Eye On The Door when it comes to selling his marital home with Millie Mackintosh.

The 34-year-old has listed his four-bedroom south-east London house for £2.5million and it comes complete with a dressing room, freestanding bath-tub and sound-proofed recording studio.

The decor is very tasteful with wooden floors through, dark turquoise walls and pale pink contrasting curtains.

a) The initial page of `www.dailymail.co.uk` with banner advertisement.



WHY AM I BEING SHOWN THIS BANNER?

These are the last products that you viewed on Packt Publishing's website:

Large Scale Machine Learning with Spark	29,15 €
Getting Started with TensorFlow	20,82 €

People who looked at these products often ended up choosing the following products:

Getting Started with TensorFlow	20,82 €
Mastering Social Media Mining with Python	26,65 €

CRITEO TAKES YOUR PRIVACY VERY SERIOUSLY

Criteo collects browsing data such as product viewed and no identifying data. For more information on Criteo's privacy policy, click here.
Criteo displays personalized banners that are more likely to be of interest to you than non-targeted advertising. We personalize our banners based on your recent browsing behavior.

HOW CAN I TEMPORARILY DISABLE CRITEO SERVICES?

- To temporarily disable these Packt Publishing banners, click here.

Please note that if you choose this option you will temporarily disable Criteo ads personalized for Packt Publishing and not all Criteo ads. If you browse back on Packt Publishing website you are likely to see Criteo ads personalized for Packt Publishing again.

HOW CAN I PERMANENTLY DISABLE CRITEO SERVICES?

- You can disable Criteo banners on this device for that browser by clicking here.
- You can disable Criteo banners permanently not only on this browser but also on all the browsers we matched to your technical identifier by clicking here.

Please note that if you choose this option you will be opted out from receiving ads displayed by Criteo for this browser and the browsers we have matched to your technical identifier. For matched browsers your opt out will be effective as soon as we see you on these browsers again and if this happens within 6 weeks. You may nonetheless continue to have cookies on other unmatched browsers and see ads displayed by Criteo on these browsers.
This does only apply to Criteo services and will not block ads served by other companies.

OPT OUT PLATFORMS

- Other means to express your choice

Criteo adheres to the Digital Advertising Alliance (DAA) Self-Regulatory Principles for Online Behavioral Advertising in the US and EDAA in EU.
Criteo adheres to the Network Advertising Initiative Standards.
Please note if you express your choice through the above platforms: when your browser blocks some third party cookies by default, you may have to first activate all cookies as explained by said platforms.

PLEASE NOTE! FOR YOUR CHOICE TO BE EFFECTIVE

- Choice that relies upon a cookie

When you disable our services as described above, a cookie is placed on your opted out browsers. We need to keep that cookie to recognize you've disabled our services. If you delete all your cookies, use a new browser or device to connect to the Internet you will have to opt-out again.
Your choice will disable Criteo services in the web environment of your opted-out browsers. If you never want to see Criteo ads, you will also have to disable our ads through all Internet Browser(s) you use and in the mobile application environment.
To learn more

YOUR STATUS

Criteo service enabled
You have not disabled Criteo services

Copyright (c) Criteo 2018 - Contact

b) Explanation why certain advertisement was shown to the user after clicking to advertisement options in right top corner of the banner.

Figure 3 Example of personalised advertisement from `http://www.dailymail.co.uk`.

observed behaviour of the user to choose targeted advertisement [14]. The example of targeted personalised advertisement is in Figure 3.

Another option is to serve advertisement related to the context of the publisher's site [15] or to use geotargeting based on IP Address [16].

3.1.1 Advertisement delivery mechanism

Firstly, we describe the major parties of advertisement delivery mechanism. Secondly, we describe three types of the advertisement delivery mechanism.

The advertisement delivery mechanism involves three major parties:

- *Publishers* are owners of some Web page where they display advertisement for which they are paid. The usual business models for a publisher to calculate revenue are pay-per-impression and pay-per-click model. The pay-per-impression is the measure of cost that the publisher gets paid for advertising to a user. While the pay-per-click is the measure of cost that the publisher gets paid for the ad-click of a user.
- *Advertisers* are creators of the marketing messages (advertisement) which they want to show to a large population through publishers Web pages. They are the revenue sources of online advertisement because they pay for displaying their advertisement on publishers Web pages.
- *Audience* are Internet users who visit publisher's Web page and receive the advertisement.

The advertisement exchange between publisher and advertisers can come in three different scenarios:

- First, publishers can have in-house advertisement departments and own ad-servers.

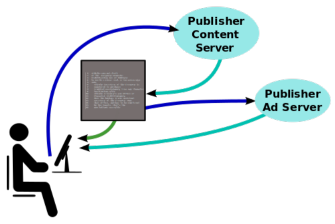


Figure 4 The advertisement exchange between publisher and advertiser when publishers have in-house advertisement department and own ad-servers [18].

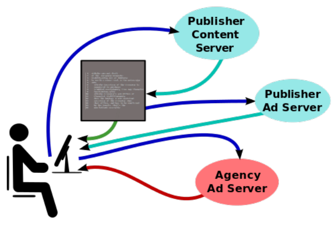


Figure 5 The advertisement exchange between publisher and advertiser when publishers outsource an advertisement department to an advertising agency [18].

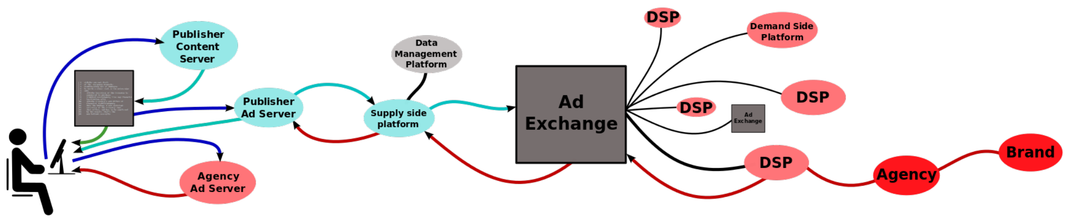


Figure 6 The advertisement exchange between publisher and advertiser when publishers offer their space to the ad-networks [18].

The publishers have their agreement with advertisers and directly use their advertisement [17]. This scenario is in Figure 4.

- Second, publishers can outsource an advertisement department to an advertising agency and publish the advertisement from the agency's ad-servers [18]. This scenario is in Figure 5.
- Finally, publishers can offer their ad space to the ad-networks. The ad-exchange is a real-time bidding market that facilitates the buying and selling of media advertising inventory from multiple ad-networks. This is the most popular approach which is discussed in detail in the following section. This scenario is in Figure 6.

3.1.2 Mechanism of ad-networks

The description of the mechanism of ad-networks in this section follows the visualisation in the Figure 6. The advertisement loading process starts when a user's browser requests content of a Web page through publishers content server that answers with a Web page content containing links to the publisher's ad-server. User's browser sends the request for the advertisement together with the information about the user from cookies to the publisher's ad-server which will resent this information to a *supply-side platform server* (SSP).

SSP acts as a technical interface between the publisher, advertiser, and ad exchange. It can bring unsold inventory to many ad exchange markets at once and simplify the complexity of ad-network relationships for the publisher [19].

SSP servers enrich the data based on user identification with data from the *data management platform* DMP.

DMP is data warehouse which collects data about users, sorts them and splits it in a way that is useful for serving advertisement such as demographic information, previous purchases, and other data related to the customer behaviour [20]. This package of information about the user, his behaviour, and publishing site is forwarded to the ad-exchange.

The ad-exchange offers this opportunity to *demand-site platforms* (DSP).

DSP acts on behalf of ad agencies, who sell ads of different brands. Thus, DSPs searches for the right advertisement for the right user at the right time. DSP receives all the information about the user and has to decide in 10 milliseconds if the user is suitable for their advertisement and how much they can bid to buy the ad space.

Then the ad-exchange selects DSP with the highest bid and passes the link through the redirection chain back to the user's browser, which requests the ad from the agency's ad server [17].

The top online ad-exchanges by ranker.com are OpenX, AdExpert, and DoubleClick.

3.2 Advertisement frauds

The goal of this thesis is to detect noisy advertisement fraud, which is caused by adware. Adware is malicious software bringing the unwanted advertisement to user's browser and displaying advertisements on user's computer through malicious tool-bars, ad-injection or invisible clicks on tons of advertisement and others. The advertisement fraud categories caused by adware, we are interested in, are click fraud, impression advertisement fraud, click hijacking, ad-injection and online pop-up advertisement scams. All these types of advertisement frauds generate large number of advertisement traffic in order to gain a considerable amount of money. These frauds are described in detail in following sections.

3.2.1 Click fraud

The click fraud is connected to pay-per-click revenue model. It can be caused by person, automated script or computer program that imitates a legitimate user of a Web browser clicking on an advertisement unintendedly. We speak about fraud because the advertiser is paying for a clicks without receiving any true value.

We can distinguish two types of click frauds according to their intentions:

- We speak about *inflationary click fraud* if the advertisement appear at third-party Web page with pay-per-click revenue model, then these third parties may be interested in clicking the advertisement to inflate their revenues.
- We speak about *competitive click fraud* if advertisers are interested in clicking rivals advertisement with the purpose of driving up their costs or exhausting their budget. Once the advertiser budget is exhausted, the advertiser exits the advertisement auction.

3.2.2 Impression advertisement fraud

The impression advertisement fraud is connected to pay-per-impression revenue model. It is one of the most commonly seen advertisement fraud, which can appear in several forms.

- One form involves *fabrication HTTP requests* to either the publisher's page or the ad-server directly to artificially inflate the actual revenue.
- Another form of impression advertisement fraud is advertisement stacking. The advertisement stacking is when advertisements are stacked on top of each other in the same space but only the top advertisement is visible. However, the lower layer with advertisements are reported as viewed advertisements to the advertiser, who pay for unseen promotion.

3.2.3 Click hijacking

The click hijacking happen, when a user of an infected computer clicks on a search result link displayed through a search engine query or clicks on something different from what the user perceives they are clicking on, then the malware causes that the users is brought to a different Website, usually designed by the attacker, instead of the Website to which the user wants to go by clicking on a search result.

3.2.4 Ad-injection

Ad-injection is a technique, which secretly insert advertisement into Web pages without the Web page owner permission and knowing. Thus, advertisement is displayed on the owner Web page but he is not payed for it.

The ad-injection can have several forms:

- First, advertisement can be inserted on top of Web page content or advertisements that are already on the Web page legitimately by making the original ads impossible to see.
- Second, injected advertisement can replace other legitimately placed advertisement, or appear on pages that were not supposed to include advertisement.

On top of this, ad-injectors frequently monitor all of a user's browser activities, including page interactions and search queries, which reports to third parties for tracking and advertisement selection.

4 Analysis of advertisement loading process

In this Chapter, we explain the methodology of reversing the advertisement loading process to get a hands-on experience with advertisement loading process and it's redirection chains. The understanding of advertisement loading process gives us a crucial knowledge which is later used while designing the algorithm for advertisement detection, which are explained in Chapter 6.

4.1 Methodology of capturing website loading process

The analysis of advertisement loading process is based on reversing network traffic of Websites with different suppliers of advertisement. The goal is to explore advertisement loading process and identify general patterns of advertisement requests.

The first step consists of finding websites for our analysis. For this purpose, we use Wappalyzer [21]. The wappalyzer is a cross-platform utility that uncovers the technologies used on websites in general. In our case, we use it for identifying different ad-networks used on Websites. We restrict our analysis to top advertising networks based on the statistics on wappalyzer.com. The statistics of top advertising networks is in Figure [21].

The second step was to capture and analyse the network traffic generated by loading process of selected Websites. The browser's developer tool was used to capture the traffic.

4.2 Analysis of Website loading process

Firstly, we collect set of URLs, which download advertisement pictures into examined Websites for different suppliers of advertisement. The traffic was analysed and captured through the browser as it shown in Figure 8. Examples of these URLs are in Table 1. These URLs are the unique indicators that the advertisement is loaded. Unfortunately, these URLs were impossible to generalise or use for a general advertisement detection

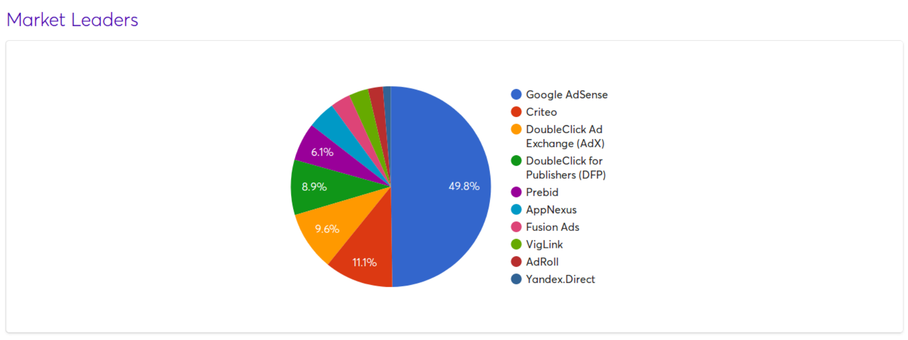


Figure 7 Top advertisement networks according to [21].

`https://tpc.googlesyndication.com/simgad/5502983634562170`
`https://s.adroll.com/a/73H/K7E/73HK7ERMARBMLBBITHHTHU.gif`
`files.cointraffic.io/pub/4670/11918.gif`
`https://vcdn.adnxs.com/p/creative-image/de/dc/6b/7d/*.jpg`
`http://pix.eu.criteo.net/img/img*`

Table 1 Examples of advertisement pictures loading URLs.

algorithm. Therefore, we use these URLs later in our research as true positive labels for our advertisement detection algorithm.

Secondly, all components of advertisement on the web page are encapsulated in the iframe tag. The iframe tag specifies an inline frame tag, which is used to embed another document within the current HTML document. The advertisement context is dynamically loads the advertisement content to the website. Thus, we collect URLs of sources of advertisement iframes. Interestingly, only URLs of a few suppliers appear in examine Websites, which repeat and load the final advertisement pictures from different ad-servers. Examples of the iframes supplier source's URLs are in Table 2.

`http://tpc.googlesyndication.com/safeframe/*`
`https://s0.2mdn.net*`
`https://a3275.casalemedia.com/ifnotify?*`

Table 2 Examples of iframe loading URLs.

Finally, we prove our theory, that the advertisement content is referred by the iframe source, by analysing the redirection chain of loading process of web pages. The redirection chain of loading process consists of edges that are represented as tuples (referrer, host) or (host, location). We analyse the redirection chains in a two ways.

For analysis of redirection chain with full URLs, we develop the tree structure displaying tool, which shows the relationship between loaded components into a website. In Figure 10, we can see the subset of URLs needed to load arestechnica Website, where a bunch of URLs, related to the advertisement, are referred by the source of the iframe, "*.googlesyndication.com/safeframe/*".

For the analysis of hostnames, we use Gephi visualiser. In Figure 9, we can see the advertisement providers, as casalemedia.com, amoazon-adsystem.com or tcp.googlesyndication.com, refer another advertisement parties, as gstatic.com, casalemecia.com, and many others. The size of each node is based on the number of output edges, and the darkness of the nodes is based on the number of incoming edges.

In Figures 10 and 9, we can see that the advertisement provider is always between the website and an advertisement server. The advertisement provider can refer the advertisement from different ad-servers.

Hence, we can understand the URL of iframe source as a advertisment provider between the Web page and different ad-servers. Different Ad-networks are connected to the Ad-exchange and based on their bid the advertisement is loaded with iframe source URL as a referrer to the website. The principle of Ad-exchange was in depth explained in Section 3. This crucial knowledge is used while designing advertisement detection algorithm in Chapter 9.

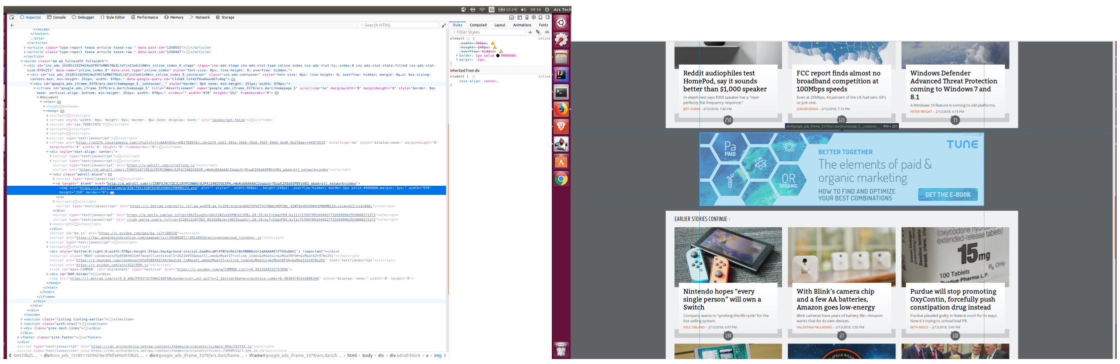
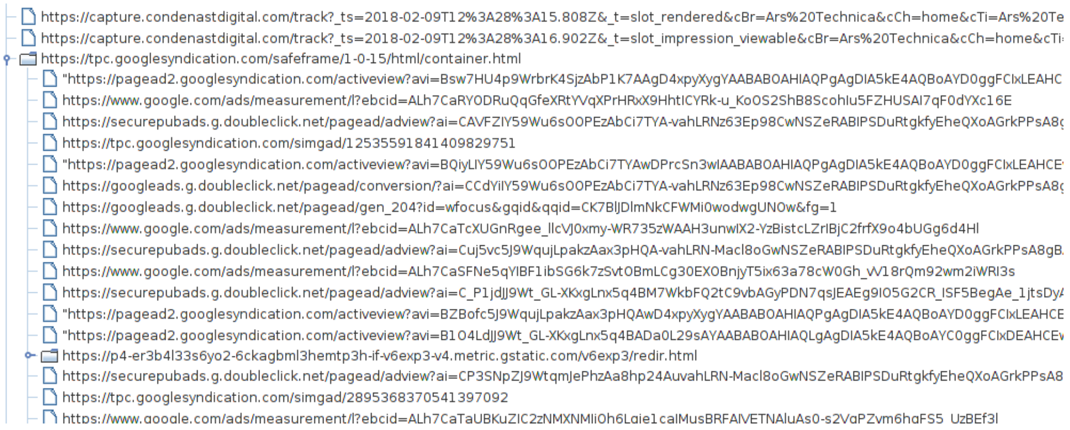


Figure 8 ArsTechnica analysis of advertisement element.



```
https://capture.condenastdigital.com/track?_ts=2018-02-09T12%3A28%3A15.808Z&_t=slot_rendered&cBr=Ars%20Technica&cCh=home&cTi=Ars%20Te
https://capture.condenastdigital.com/track?_ts=2018-02-09T12%3A28%3A16.902Z&_t=slot_impression_viewable&cBr=Ars%20Technica&cCh=home&cTi
https://tpc.googlesyndication.com/safeframe/1-0-15/html/container.html
  "https://pagead2.googlesyndication.com/activeview?avi=Bsw7HU4p9WrbrK4SjzAbP1K7AAgD4xpyXygYAABABOAHIAQPgAgDIA5kE4AQBoAYD0ggFClxLEAHC
  https://www.google.com/ads/measurement/l?ebcid=ALh7CaRYODRuQqGfeXRtYVqXPrHRxX9HhtICYRk-u_KoOS2ShB8Scohlu5FZHUSAl7qF0dYXc16E
  https://securepubads.g.doubleclick.net/pagead/adview?ai=CAVFZIY59Wu6sOOPEzAbCi7TYA-vahLRNz63Ep98CwNSZeRABIPSDuRtgkfyEheQXoAGrkPPsA8
  https://tpc.googlesyndication.com/simgad/12535591841409829751
  "https://pagead2.googlesyndication.com/activeview?avi=BQiyLIY59Wu6sOOPEzAbCi7TYAwDPrcSn3wIAABABOAHIAQPgAgDIA5kE4AQBoAYD0ggFClxLEAHCE
  https://googleads.g.doubleclick.net/pagead/conversion/?ai=CCdYiIY59Wu6sOOPEzAbCi7TYA-vahLRNz63Ep98CwNSZeRABIPSDuRtgkfyEheQXoAGrkPPsA8
  https://googleads.g.doubleclick.net/pagead/gen_204?id=wfocus&gqid&qqid=CK7BlJDlmNkCFWMi0wodwgUN0w&fg=1
  https://www.google.com/ads/measurement/l?ebcid=ALh7CaTcXUGnRgee_llcVJ0xmy-WR735zWAAH3unwIX2-YzBistcLZrIBjC2frfX9o4bUGg6d4Hl
  https://securepubads.g.doubleclick.net/pagead/adview?ai=Cuj5vc5J9WqujLpakzAax3pHQA-vahLRN-Macl8oGwNSZeRABIPSDuRtgkfyEheQXoAGrkPPsA8gB
  https://www.google.com/ads/measurement/l?ebcid=ALh7CaSFNe5qYIBF1ibSG6k7zSvt0BmLCg30EXOBnjyT5ix63a78cW0Gh_W18rQm92wm2iWRI3s
  https://securepubads.g.doubleclick.net/pagead/adview?ai=C_P1jdJJ9Wt_GL-XKxgLnx5q4BM7WkbFQ2tC9vbAGyPDN7qsJEAEg9I05G2CR_ISF5BegAe_1jtsDy
  "https://pagead2.googlesyndication.com/activeview?avi=BZBofc5J9WqujLpakzAax3pHQAwD4xpyXygYAABABOAHIAQPgAgDIA5kE4AQBoAYD0ggFClxLEAHCE
  "https://pagead2.googlesyndication.com/activeview?avi=B1O4LdJJ9Wt_GL-XKxgLnx5q4BADa0L29sAYAABABOAHIAQLgAgDIA5kE4AQBoAYC0ggFClxDEAHCEw
  https://p4-er3b4l33s6yo2-6ckagbml3hemtp3h-if-v6exp3-v4.metric.gstatic.com/v6exp3/redir.html
  https://securepubads.g.doubleclick.net/pagead/adview?ai=CP3SNpZJ9WtqmJePhzAa8hp24AuvahLRN-Macl8oGwNSZeRABIPSDuRtgkfyEheQXoAGrkPPsA8
  https://tpc.googlesyndication.com/simgad/2895368370541397092
  https://www.google.com/ads/measurement/l?ebcid=ALh7CaTaUBKuZIC2zNMXNMIi0h6Lgie1calMusBRFAIVETNAluAs0-s2VgPZvm6hgFS5_UzBEf3l
```

Figure 9 Tree structure visualisation tool: Part of ArsTechnica.com loading traffic URLs of advertisement iframe and referred URLs of advertisement elements.

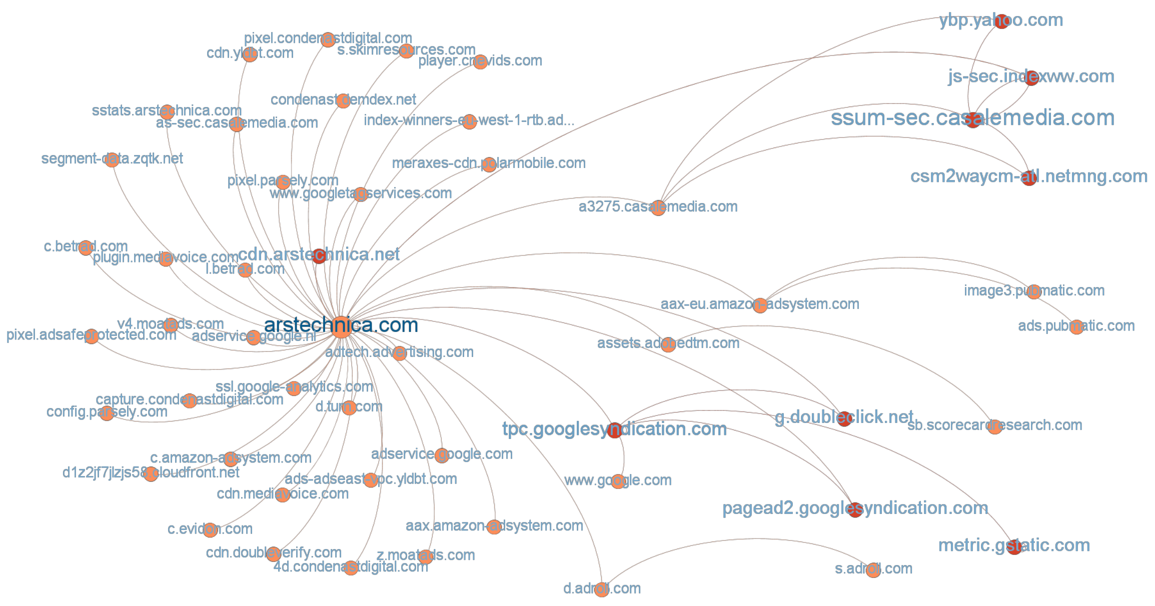


Figure 10 The redirection graph of hostnames from arstechnica.com loading process. The advertisement providers, googlesyndication.com, casallemedia.com and others, refer the advertisement from different ad-servers.

5 Data Explanation

This section provides a detailed description of the dataset we use for the evaluation of proposed algorithm. The data set consists of one week of network traces of 2 740 475 users from 636 companies. Particularly, we use one week from January. In spite of, our data set contains only one week of the data, we already speak about this data as about Big Data. The data contain features extracted from HTTP, HTTPS and inspected HTTPS Web Access logs which were captured on the network perimeter. We describe those in detail in the following sections.

5.1 Web Access logs

The access logs are records from CISCO Web Security Appliances [22]. It is a Web proxy that inspects and then either forwards or drops web traffic based on reputation filters or the outcome of inline file scanning.

The individual records contain information extracted from the HTTPS/HTTPS headers of request and response pairs. In the following text we will denote the record as *flow*.

5.2 Description of a single flow

Each flow contains features extracted from HTTP header, so the content of the requests/responses is not inspected. Following fields are extracted from the header: username, source IP address, destination IP address, source port, destination port, protocol, number of bytes transferred from client to server and from server to client, flow duration, timestamp, user agent, URL, host, referrer, referrer host, location, location host, MIME-Type, HTTP status, tag signifying whether the connection was blocked or not. The number of extracted features is higher for non-encrypted traffic (HTTP) than for encrypted traffic (HTTPS). In HTTPS, we do not have features as user agent, URL host, referrer, referrer host, location, location host, MIME-Type and HTTP status. As the amount of HTTPS traffic is increasing, we have to deal with the lower number of known fields.

Another disadvantage of HTTPS flows is that it generates significantly fewer flows than the HTTP traffic because for the HTTPS we can see only the initial connection to set up the tunnel and the rest of the flows we do not see as it is shown in Figure 11. In HTTP, we can see a bunch of flows while only one page is loading, because firstly the HTML index of the web page is loaded and then the other parts referred in the index are loaded. Hence, loading process of one web page can be 100-150 HTTP requests.

5.3 Inspection of HTTPS packets

The information in encrypted (HTTPS) flows is crucial. Hence, some companies inspect the HTTPS connections. The inspection is computationally expensive, and therefore

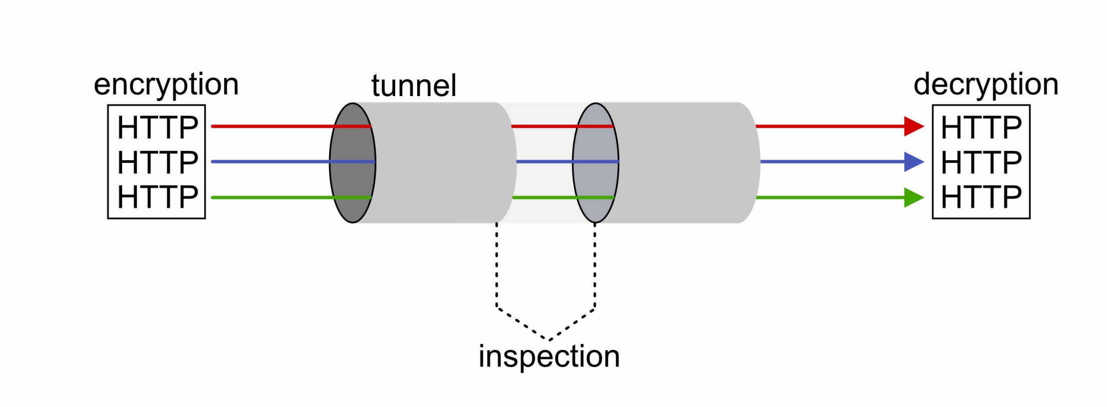


Figure 11 The HTTPS traffic is encrypted and we can see only the initial connection (lebeled as tunnel in the Figure). To be able to see the flows from the encrypted connection some proxies inspect the traffic.

only some HTTPS connections are inspected. The strategy, which packets will be inspected, depends on company's capturing strategy.

The inspection of HTTPS is executed by the men in the middle (MITM) proxy. Proxied HTTPS requests are terminated by the proxy and resent to the remote webserver. The server certificates presented to the client are dynamically signed by the proxy and contain most of the same fields as the original webserver certificate. However, the issuer distinguish name is now set to the name of the proxy's self-signed certificate and the public/private keys of the proxy are used in creating the forged certificate.

6 Advertisement traffic detection

The goal of this thesis is to propose network behaviour features and develop a model for advertisement fraud detection. The crucial milestone in advertising fraud detection is the differentiation between advertisement flows and non-advertisement flows in the HTTP and the HTTPS access logs. This chapter describes methods of advertising traffic detection which are the essential part of our advertising fraud detection solution.

First, we use available intelligence from the Internet, but the recall of these methods is low. Therefore we develop own advertisement detection algorithm.

6.1 Pattern matching - Ad-block lists

The public intelligence about the advertisement is accumulated in ad-block lists typically used in Web browsers. The ad-block lists are sets of rules originally designed for ad-block filters. These rules were manually co-created by the members of advertisement crowdsourcing community.

The ad-block filters filter the content and block the advertisement on web pages. It automatically removes unwanted content from the Web page, including annoying advertisement, bothersome banners and troublesome tracking. The visualisation of Ad-block behaviour is in Figure 12. The content of the Website is filtered with ad-block filters, and then only the content of the Web page without the advertisement is shown to the user. We use two ad-block lists, which are described in this section, Yoyo and EasyList.

6.1.1 Yoyo ad-block list

The Yoyo ad-block list [24] contains about 2 600 rules for hostnames of ad-servers. The example of such rule is "ad.doubleclick.com". The owner of the list is collecting these hostnames to get rid of annoying advertisement by redirecting advertisement related traffic to the local host.

But this detection technique has a low recall because with only 2 600 rules for hostnames it is possible to detect only a very small portion of advertisement flows.

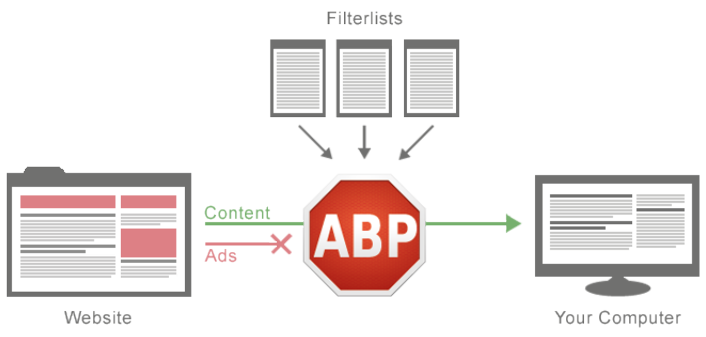


Figure 12 Behaviour of ad-block filter [23]. The content of the website is filtered with ad-block filters, then only the content of the website without the advertisement is shown to the user.

6.1.2 EasyList

The EasyList ad-block list is the primary filter list that removes the advertisement from international web pages, including unwanted frames, images and objects. It is the most popular list used by many ad block filters [25]. The EasyList ad-block list contains set of rules, which determine the addresses that are related to the advertisement. These rules are more sophisticated than the ones used in Yoyo block list. The rules can detect advertisement by following rules:

- **Address part rule** The regular expression searches for a certain pattern in the path of the URL. For example the pattern *//ads.example.com^* will detect this URL *http://example.com/banner/foo/img* but will not detect this URL *http://example.com/banner/img* as an advertisement.
- **Domain name rule** The regular expression searches for an exact match in the domain name. For example the pattern *//ads.example.com^* will detect this URL http://server1.ads.example.com/foo.gif but will not detect this URL *http://ads.example.com.ua/foo.gif* as an advertisement.
- **Exact Address rule** The regular expression searches for an exact match in the URL. For example the pattern *|http://example.com/|* will detect only *http://example.com/*. Everything else will not be labeled as an advertisement.

The biggest problem with EasyList is that it labels as an advertisement even URLs which belong to different trackers and other non-advertisement servers.

6.1.3 Problems of the static patterns

The advantage of both lists is that they are maintained and continuously updated by the community. But if the community behind them would fall apart, the rules would outdate and would be no longer useful. Nevertheless, both have low recall on our data, which is partially caused by small number of rules in each list. The yoyo ad-block list has only 2600 rules for hostnames. Compared to yoyo ad-block list, the EasyList has more rules, specifically 31930, but these rules have to be whitelisted with 5502 rules because the rules can match even non-advertisement URLs as advertisement without whitelisting.

The advantage of Yoyo rules is that it operates only on hostnames. Thus this list works well even for HTTPS traffic, where we do not have the query and path parts of the URL. Compare to that, EasyList ad-block list has just part of the rules oriented to the hostnames, which works on HTTPS. Since the main portion of communication is HTTPS in these days, part of this list can be used only restrictively.

The main problem of both ad-block lists is small coverage of advertisement flows. When we analyse the data manually, we can see that the evident advertisement domains are not labelled as the advertisement on a big scale (Section 7). The other problem is that the lists have to be frequently updated because the number of advertisement paths will increase with time and some paths and domains can change.

6.2 Advertisement Propagation Algorithm

The motivation for advertisement propagation algorithm is that we want to recognise as much advertisement traffic as possible if we have only limited initial knowledge. If the limited initial knowledge is extended by the algorithm, then even updates can be done automatically, and no human maintenance is needed.

We get the initial knowledge by reversing and analysing advertising loading process as described in Chapter 4. We discovered that the advertisement was most frequently loaded into the iframe element in the Web page which has very often the same source with a clear pattern in the URL "http://tpc.googlesyndication.com/safeframe/*". This URL loads the iframe code for the advertisement and the URL of the advertisement content. The connection to the URL of advertisement content follows and it is referred from the URL of the iframe. Thus, we labelled flows as the advertisement if it is referred from this googlesyndication URL. Even though, this iframe source is very popular in our research of advertisement loading process, it does not help us to label significantly more advertisement in our data than with ad-block lists.

The most significant disadvantage of this approach is the fact that it can work only on HTTP flows and inspected HTTPS flows which have the information about the referrer.

The number of flows we are able to label as an advertisement with propagation of advertisement from googlesyndication as a referrer was not sufficient enough to use it as a label for advertisement flows, but it was sufficient enough to gain the advertisement hostnames.

With our previous knowledge of ad-exchanges and advertisement loading process from Chapter 3 and Chapter 4, we know that a publisher site has an agreement with the advertisement provider or with more providers. The advertisement provider is a mediator between the page owner and advertisement network. The advertisement provider provides the infrastructure for automatic advertisement loading and the advertisement itself and finally, he pays to the owner of the website for advertisement placement.

Thus the important consequence of this knowledge is that the advertisement can be loaded from various ad-servers over the internet. Even if the demand site platform cooperates directly with only limited amount of ad-servers, the demand side platform can resell the advertisement of another demand site platform. Thus it makes the advertisement network behind ad-exchanges even more complicated. The consequence is that with such knowledge and only one advertisement provider, we can design an algorithm to mine the advertisement networks.

The algorithm to mine the advertisement networks is composed of several partial algorithms, which are described in following sections.

6.2.1 The Advertisement Propagation Algorithm

The advertisement propagation algorithm uses all available flows with information about the referrer, the host and the knowledge of URLs of advertisement providers. Thus, all HTTP and inspected HTTPS flows (section 5.3). In these flows, the algorithm searches for flows, whose referrer match one of the advertisement providers restriction, and collects the hostnames from these flows. The advertisement provider restriction can be domain name of the advertisement provider or pattern of URL defined as a regular expression, or restriction over another fields e.g. content type has to be image. Then the occurrences of each hostname are counted, and only hostnames with the occurrences higher than 100 are returned as the result of the algorithm. The illustration of the behaviour of the algorithm is in the Figure 13.

We define the advertisement propagation algorithm in the following way. We have initial set of web flows $F = (f_1, ..., f_n)$. Each web flow contains a set of information about URL, host name and content type. It can optionally contain the information about the referrer URL and referrer host name if it is HTTP or inspected HTTPS flow. Then we have non-empty set of advertisement providers restrictions $A = (a_1, ..., a_n)$.

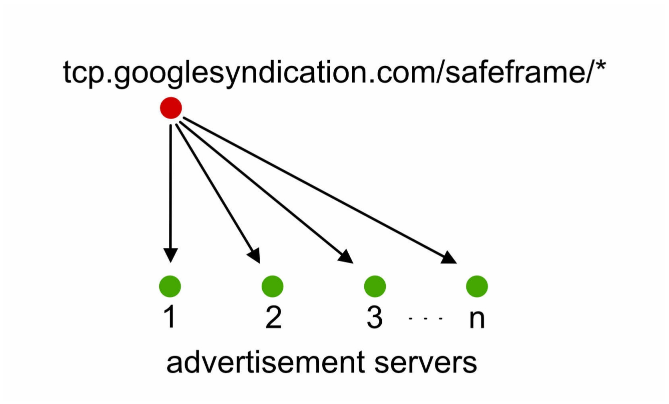


Figure 13 The illustration of the advertisement propagation algorithm with only one advertisement provider. The red dot is restriction on advertisement provider and the green dots are propagated hostnames of ad-servers. The arrow $x \longrightarrow y$ means the relation, x refers y.

The advertisement provider restrictionis defined as a restriction over content type, URL or hostname. We store all propagated ad-servers hostnames in list P and in set Q, we store propagated hostnemes with their occurrences. The result of the Algorithm (1) is a set of ad-server's host names S.

Algorithm 1 The advertisement propagation algorithm

```
1: function AdvertisementPropagation($F$, $A$)
2:     $P = \{\}$
3:     $Q = \{\}$
4:     for all $f_i \in F$ do
5:         if $f_i$ has information about the referrer then
6:             if $f_i$ satisfies one of the restrictions $A$ then
7:                 $P = P \cup \{f_i\}$
8:                 $e^{domainName} \leftarrow$ExtractDomainNameFromHostName($f_i^{host}$)
9:                 if $\exists q_k \in Q : (q_k^{domainName}$ equals $e^{domainName})$ then
10:                    $q_k \leftarrow (q_k^{domainName}, q_k^{count} + 1)$
11:                else
12:                    $Q = Q \cup \{(e^{domainName}, 1)\}$
13:                end if
14:            end if
15:        end if
16:    end for
17:    $S = \{\}$
18:    for all $p_i \in P$ do
19:        $e^{domainName} \leftarrow$ExtractDomainNameFromHostName($p_i^{host}$)
20:        if $\exists q_j \in Q : (q_j^{domainName}$ equals $e^{domainName})$ & $(q_j^{count} > 100)$ then
21:            $S = S \cup \{p_i^{host}\}$
22:        end if
23:    end for
        return $S$
24: end function
```

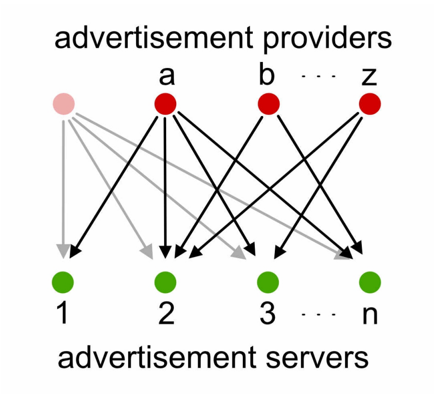


Figure 14 The illustration of the advertisement back-propagation algorithm from ad-servers hostnames from the advertisement propagation algorithm. The green dots are ad-servers and sharp red dots are new advertisement providers. The arrow $x \longrightarrow y$ means the relation, x refers y.

6.3 Advertisement Back-propagation Algorithm

Initially, we ran the advertisement propagation algorithm described in Section 6.2.1 with only one advertisement provider using the restriction on a URL and a content type. But as we explained in Section 6.2, the advertisement supplied to the web page is not connected with only one advertisement provider. Thus instead of creating more advertisement provider restrictions manually, where we would need human maintenance, we propose an algorithm for advertisement back-propagation, which generates the restrictions automatically.

We use the knowledge that the ad-network behind the ad-exchange is very complicated, and it leads to the fact that one ad-server can sell his advertisement through different advertisement providers. Therefore, once we have a set of hostnames of ad-servers from the advertisement propagation algorithm, we can expect that at least part of them will be referred by another advertisement provider in our data. Thus we can extend input for our advertisement propagation algorithm.

The advertisement back-propagation algorithm uses all available flows with information about the referrer and the ad-server hostnames generated by the advertisement propagation algorithm. The algorithm checks hostname of each flow with the referrer if the hostname matches with some of the ad-server hostnames from the input set. If it matches, we extract the domain name of the referrer and store it with the number of occurrences. Then, we take top fifteen domain names of advertisement providers. These domain names can be used as an advertisement providers restriction in the following advertisement propagation algorithm. We use domain names instead of hostnames because the low-level sub-domain of the advertisement provider hostname can vary for big advertisement providers for each advertisement delivery. The illustration of advertisement back-propagation algorithm is in Figure 14.

We define the advertisement back-propagation algorithm in the following way. We have initial set of web flows $F = (f_1, ..., f_n)$. Each web flow contains a set of information about URL, hostname and content type of targeted destination and it can optionally contain the information about the referrer URL and referrer hostname. Then we have non-empty set $S = (s_1, ..., s_n)$ of ad-servers hostnames as identified using the propagation algorithm described in section 6.2.1. The result of the Algorithm (2) is a set D of advertisement provider's domain names with number of occurrences

$d_i = (d_i^{domainName}, d_i^{count})$.

Algorithm 2 The advertisement back-propagation algorithm

```
1: function advertisementBackPropagation($F$, $S$)
2:     $D = \{\}$
3:     for all $f_i \in F$ do
4:         if $f_i$ has information about the referrer then
5:             for all $s_j \in S$ do
6:                 if $s_j^{host}$ equals $f_i^{host}$ then
7:                     $e^{domainName} \leftarrow$ extractDomainNameFromHostName($f_i^{referrer}$)
8:                     if $\exists d_i \in D : (d_k^{domainName}$ equals $e^{domainName})$ then
9:                         $d_k \leftarrow (d_k^{domainName}, d_k^{count} + 1)$
10:                    else
11:                        $D = D \cup \{(e^{domainName}, 1)\}$
12:                    end if
13:                end if
14:            end for
15:        end if
16:    end for
        return $D$
17: end function
```

6.4 Advertisement Network Mining

In the previous sections, we describe the advertisement propagation algorithm and the advertisement back-propagation algorithm. These algorithms are principal components of the advertisement network mining algorithm that is described in detail in this section.

The network mining algorithm combines all our previous knowledge about the ad-networks and the advertisement loading process. This algorithm shows how we are able to gain 384 863 number hostnames of ad-servers with access to Big Data web flows and the knowledge about only one advertisement provider. Because even non-advertisement servers can be referred by the advertisement providers we propose the automatic white-listing algorithm to clean the result of the network mining algorithm.

Firstly in this section, we describe the automatic white-listing algorithm, and then we present the advertisement network mining algorithm with white-listing.

6.4.1 The Automatic White-listing Algorithm for network mining algorithm

As was previously mentioned, some hostnames, which are not ad-servers, can be referred by the advertisement providers. This can happen due to the fact that the URL is related to the advertisement delivery mechanism, but the indication of advertisement delivery is in the path of the URL. Thus, if we take just hostname, we lose some information, and the hostname itself can be mainly related to non-advertisement flows. Thus, such label would generate false positive, and it is a problem if such hostname is for example "google.com". Therefore we design the algorithm, which recognises such servers and filters them out.

For the white-listing we take an advantage of the fact that user typically does not reach the ad-server directly. Contraty, the non-ad-servers are accessed by the user directly without any redirection. Thus access to non-ad-server should be captured in the flow data as a flow without referrer, and the URL should not contain any path. On the other hand, access to an ad-server has these fields typically filled. Proposed algorithm counts the number of occurrences the hostname appears in our data without referrer and path in the URL because the non-ad-server hostnames appear in such form significantly more often than the ad-server hostname. The ad-servers are always referred by another ad-server or by the web page where the advertisement is located. For this algorithm, we can use only HTTP flows and inspected HTTPS flows as the HTTPS flows do not contain the information about the referrer and path in the URL.

The automatic white-listing algorithm uses all available flows with the information about the referrer and the ad-server hostnames generated by the advertisement propagation algorithm, which should be white-listed. Firstly, the algorithm checks if the flow is inspected HTTPS or HTTP. Then the algorithm checks if the flow does not have referrer and then if the URL does not contain the path. To conclude, the flow that has the information about referrer, but its referrer field is empty, and the URL of the flow does not contain the path, represents direct visit of the destination hostname. The hostname is white-listed, if it was directly visited more than ten times.

We define the automatic white-listing algorithm in the following way. We have initial set of web flows $F = (f_1, ..., f_n)$. Each web flow contains a set of information about URL, hostname and content type and it can optionally contain the information about the referrer URL and referrer hostname. Then we have set of results from the advertisement propagation algorithm $S = s_1, ..., s_n$. Each result $s_i = (s_i^{host})$ is potential ad-server hostname. The result of the Algorithm (3) is a set K of ad-server hostnames $k_i = (k_i^{host})$ without non-ad-server hostnames.

6.4.2 The Advertisement Network Mining Algorithm

The proposed advertisement network mining algorithm that combines all previously described algorithms. The algorithm takes as an input set of flows and information about one advertisement provider. Then the algorithm exploits given knowledge and generates almost four hundred thousand ad-servers hostnames, which are able label both, HTTP and HTTPS traffic.

The advertisement network mining algorithm uses all available flows and one advertisement provider restriction. Then the advertisement propagation algorithm is applied to the input data. The white-listing algorithm is applied to the result of the advertisement propagation algorithm. The advertisement back propagation extends the knowledge of advertisement providers, and top fifteen is taken as an extended input for the second run of advertisement propagation algorithm. Finally, the result is white-listed and generated domains can be used as advertisement indicators. The illustration of the advertisement network mining algorithm is in Figure 15.

The algorithm consists of several steps and combines previously described algorithms in this Chapter. As was previously mentioned, the input for our advertisement network mining algorithm is set of web flows $F = (f_1, ..., f_n)$ and the restriction of one advertisement provider $A = (a_{googlesyndication})$. Each web flow contains a set of information about URL, hostname and content type and optionally the information about the referrer URL and referrer hostname. The output is a set of ad-server hostnames $M = \{m_1, ..., m_n\}$. The advertisement network mining algorithm is described in pseudocode in the Algorithm (4).

Algorithm 3 White-listing algorithm

```
1: function WHITELISTING($F$, $S$)
2:     $L = \{\}$
3:     for all $f_i \in F$ do
4:         if $\exists s_j^{host} \in S : f_i^{host}$ equals $s_j^{host}$ then
5:             if $f_i$ is HTTP or $f_i$ is inspected HTTPS then
6:                 if $f_i^{referrer}$ is empty then
7:                     if $f_i^{URL}$ not contains path then
8:                         if $\exists l_k \in L : (l_k^{host}$ equals $f_i^{host})$ then
9:                             $l_k \leftarrow (l_k^{host}, l_k^{count} + 1)$
10:                        else
11:                            $L = L \cup \{(f_i^{host}, 1)\}$
12:                        end if
13:                    end if
14:                end if
15:            end if
16:        end if
17:    end for
18:    $K = \{\}$
19:    for all $l_i \in L$ do
20:        if $l_i^{count} > 10$ then
21:            $K = K \cup \{f_i^{host}\}$
22:        end if
23:    end for
        return $K$
24: end function
```

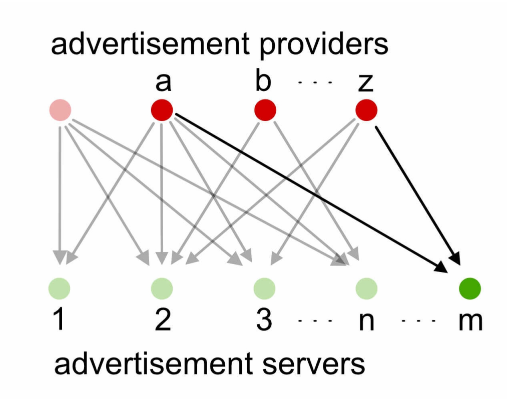


Figure 15 The illustration of the advertisement network mining algorithm. The light red dot is the initial advertisement provider, the light green dots are ad-server hostnames generated by the first run of the advertisement propagation algorithm, and the sharp green dots represent new ad-servers hostnames generated by the second run of the advertisement propagation algorithm. The arrow $x \longrightarrow y$ means the relation, x refers y.

Algorithm 4 The advertisement network mining algorithm

```
1: $A = \{a_{safefarame}\}$
2: function advertisementNetworkMining($F$, $A$)
3:     $S$ = advertisementPropagation($F$, $A$)
4:     $S$ = whiteListing($F$, $S$)
5:     $D$ = advertisementBackPropagation($F$, $S$)
6:     $D$ = takeTop15HostWithTheHighestNumberOfOccurences($D$)
7:     $M$ = advertisementPropagation($F$, $D$)
8:     $M$ = whiteListing($F$, $M$)
       return $M$
9: end function
```

6.5 Conclusion of advertisement traffic detection

To conclude, we exploit several techniques for advertisement traffic detection.

First, we start with pattern matching with available knowledge at the Internet, the first knowledge was Yoyo ad-block list and then EasyList. Nevertheless, these methods have the low recall, and they have to be manually updated.

Second, we explore the ad-networks and propose the advertisement network mining algorithm, which extends previous approaches and generates the list of ad-servers host-names. The advantage of the proposed method is that it is automatic, does not need human maintenance and thus it can be easily automatically updated. The generated labels can for both types of traffic, HTTP and HTTPS.

7 Evaluation of the advertisement traffic detection

In this chapter, we evaluate methods for advertisement detection proposed in Chapter 6. The first part of this chapter describes the evaluation of the results of the proposed algorithm for advertisement detection. The second part evaluates and compares different approaches for advertisement detection.

7.1 Evaluation of the results of proposed Advertisement Network Mining Algorithm

In this section, we evaluate each step and result of the advertisement network mining algorithm. The advertisement network mining algorithm, described in Section 6.4.2, is initialised with the set of flows and the initial set of restrictions of advertisement providers. The initial set contains restriction for one particular structure of the existing advertisement provider. The restriction consist of regular expression for the URL in the form of "http(s)://tcp\.googlesyndication\.com/safeframe/.*" and content type limitation to image. As the advertisement network mining algorithm consists of several sub-algorithms, we evaluate each sub-algorithm separately to show the algorithm working process.

The first step is the advertisement propagation algorithm which takes all input parameters of the advertisement mining algorithm and generates advertisement provider's hostnames. These hostnames are then white-listed with the white-listing algorithm, described in Section 6.4.1. The white-listing algorithm filters the set of advertisement providers hostnames generated from the advertisement propagation algorithm. This step leads to 7 046 generated hostnames of ad-servers.

The result of the first step was manually verified to confirm that it includes only hostnames of ad-servers. The other evaluation of this step is the fact that if this step would contains a significant amount of false positives, it would massively negatively impact the following stages of the advertisement network mining algorithm and its results.

The second step is the advertisement back-propagation algorithm, described in Section 6.2.1, which takes as an input set of flows and the result from the previous step (ad-server's hostnames). It generates set of advertisement providers domain names. Out of the result, only top fifteen domain names were taken for the next step of the advertisement network mining algorithm. The top domain names of advertisement providers are in Table 3. We took only top fifteen domain names in order to be able to be sure that they are related only to the advertisement delivery process. Thus, they are not introducing false positives in the next step or the propagation.

We manually evaluat all top fifteen domain names, and all of them were true advertisement providers.

The last step is the second run of the advertisement propagation algorithm with the set of flows and set of restrictions on hostnames generated from domain names of top advertisement providers, and generates set of advertisement providers hostnames. These hostnames are then white-listed with the white-listing algorithm.

Finally, the last step generates 384 863 hostnames of different ad-servers.

We manually evaluated the result of the last step. Our strategy, to manually analyse as many hostnames as possible, was to check the most popular hostnames in our dataset because the most popular hostnames would generate a significant amount of false positives. The other strategy was to group the hostnames by their second-level domain name and evaluate each domain name if it belongs to some advertisement provider.

Our evaluation proves that generated hostnames are exclusively related to advertisement servers, and no popular non-advertisement server occurs. Manual evaluation is the only possible evaluation method for us to find out if the generated set of ad-servers contains any FPs. However, very rarely some non-popular non-ad-servers can appear. It can happen due to loss of part of the information while extracting hostnames from the URLs because some small servers can be used for serving advertisement and at the same time for other non-advertisement related purposes, and the discriminate information is hidden in the path of the URL.

As the online world of advertisement is complex and not generally well structured, it is hard or even impossible to measure the recall (the coverage) of our algorithm. The most relevant way, how to measure the coverage, we have an access to, is to compare our result with Cisco Cloudlock list **??**. The Cisco Cloudlock list contains 793 manually evaluated hostnames of top ad-servers, which appear in our dataset that generate 4 304 542 901 flows in our dataset. Proposed network mining algorithm covers 603 out of 793 Cloudlock ad-servers domain. Hence, the proposed algorithm has coverage 76% on hostnames.

However, proposed network mining algorithm covers 3 849 660 860 flows out of 4 304 542 901 flows detected by the Cisco Cloudlock list as an advertisement in our data. Hence, the proposed algorithm has coverage 89% on flows.

Although, we do not discover 190 hostnames from the Cisco Cloudlock list, which is 24% of total number of Cloudlock list hostnames, we do not cover only 454 882 041 flows covered by the Cloudlock list, which is 11% of total number of flows detected by the Cloudlock list. Hence, we cover all top ad-servers, which generates majority of advertisement flows according to Cloudlock list, however some of the less noisy are missing. As they are less noisy they do not propagate to our advertisement network mining. However, the result is sufficient enough to cover majority of the advertisement traffic and we prove that at least half of the detected traffic by the advertisement network mining algorithm are true positives.

7.2 Comparison of different advertisement detection techniques

In this section, we compare different advertisement detection techniques, described in Chapter 6 with the Cloudlock list used as ground truth for the advertisement network mining algorithm in the previous section.

First, we compare the power of each approach regarding the number of ad-server's hostnames covered by each method. In Figure 16, we compare Yoyo ad-block list (Yoyo), EasyList ad-block list (EasyList), the result of advertisement mining algorithm (Mining) and Cloudlock list (Cloudlock).

We can see that the advertisement network mining algorithm covers significantly more ad-servers hostnames than the rest of the methods. Precisely, the advertisement

Domain names of	Hostname variations	Occurrence
casalemedia.com	3280	103194
rfihub.com	3895	89492
doubleclick.net	8809	71598
adnxs.com	23	11352
googlesyndication.com	4	8449
openx.net	45	6426
adnxs.com	23	11352
amazon-adsystem.com	20	6514
credio.com	37	3337
adroll.com	8	2260
appnexus.com	1	3380
viglink.com	4	2956
pubmatic.com	11	1347
viglink.com	4	2953

Table 3 Table with top advertisement providers from the advertisement back-propagation algorithm. The first number is number of variation of the hostname for given domain name; the second number is number of flows related to ad-servers with the domain name as referrer.

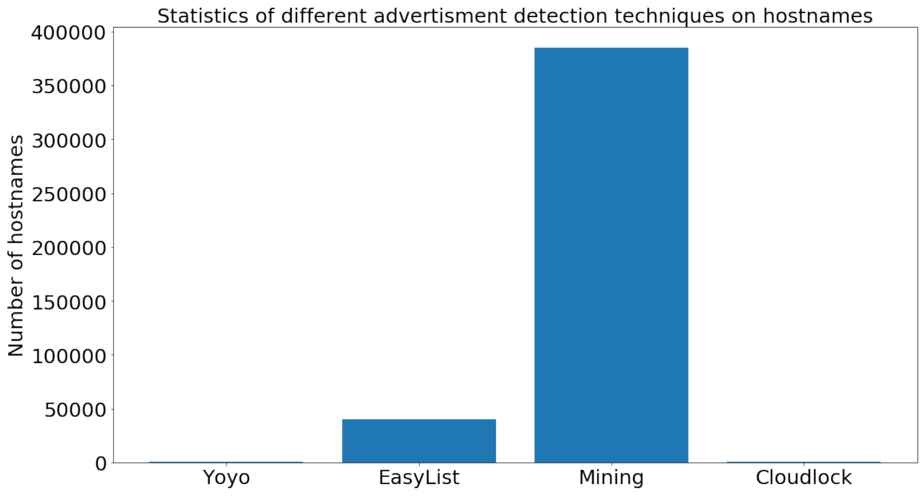


Figure 16 Comparison of different advertisement detection techniques on hostnames.

mining algorithm identifies 384 863 hostnames in our dataset as ad-servers out of total 31 313 560 hostnames. Thus, according to the Network Mining Algorithm, the ad-servers hostnames take 1.4% of the total number of hostnames.

Second, we compare the power of each advertisement detection approach regarding the number of advertisement flows they are able to detect. In Figure 17, we can see the comparison of Yoyo ad-block list (Yoyo), EasyList ad-block list (EasyList), the result of advertisement mining algorithm (Mining), Cloudlock list (Cloudlock) and the total number of flows in our dataset (Total).

We can see that the Advertisement Network Mining Algorithm detects the most advertisement flows. Particularly, the Advertisement Network Mining Algorithm identifies 8 574 591 399 flows out of total 41 403 764 161 flows in our dataset as an advertisement. Nevertheless, the Cloudlock list also detects a significant portion of an advertisement, particularly 4 304 542 901.

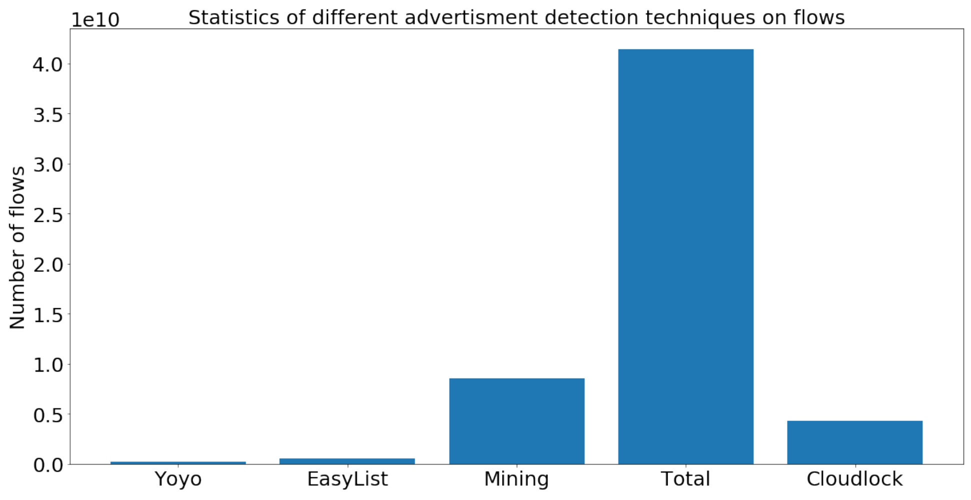


Figure 17 Comparison of different advertisement detection techniques on flows.

To conclude, the ad-block lists (Yoyo and EasyList) do not have a good results in advertisement detection. Although, the Cloudlock list, used as the ground truth for the Advertisement Mining Algorithm, contains a low number of hostnames in comparison with the rest of the methods, however, it can detect a significant number of advertisement flows. Nevertheless, proposed Network Mining Algorithm can detect significantly more advertisement flows than any other examined technique. Particularly, we discover that $20,7\%$ of the traffic is related to the advertisement.

8 Network intrusion detection techniques

The main part of technical work presented in this thesis is based on the Anomaly Detection. This chapter introduces theoretical background used for solving and modelling the advertisement frauds.

8.1 Intrusion detection system

In an information system, intrusion detection is the process used to identify intrusions. The intrusions are the activities that violate the security policy of the system.

The intrusion detection is used for many host-based or networked computer systems, where a collection of different kinds of data contains operating system calls, network traffic, or other activity in the system.

8.1.1 Usage of intrusion detection systems in network security

Network intrusion detection systems (NIDS) are usually deployed as a second line of defence that protects information systems, after firewalls, proxies, intrusion prevention systems and other perimeter devices. The primary purpose of NIDS is to detect the network attacks that successfully breached the first line of defence.

Intrusion can happen due to inadequate information security software engineering practice and computer systems applications design flaws or bugs. The other problem is that attacks are becoming more and more sophisticated and with increasing number of encrypted traffic harder to detect. All these vulnerabilities can be used by an attacker to attack the system or application without any notice of first-line defence (e.g. firewalls). To be able to reduce the potential damage of intruded malware, the second line of defence is needed and should be deployed along with other preventive security mechanisms as a part of a comprehensive defence system. The NIDS represents such a second line defence.

When the intrusion is detected in the system, an alarm is raised, and network administrator has to analyse and determine the existence and scope of the damage, repair it and improve the defence infrastructure against future attacks.

8.1.2 Categorization of intrusion detection systems

IDS can be categorised according to Guide to Intrusion Detection and Prevention Systems [26] to the following categories by different criteria.

- First, we can categorise IDS by their location in the network. Based on it, IDS is capable of detecting different types of intrusions. The IDS can be categorised as a wired network-base, a wireless network-base or a host-based.
  - The *wired network-based* IDS monitors network traffic for particular network segments or divides and analyses the network and application protocol activity to identify suspicious activity.

  - The *wireless network-base* IDS monitors wireless network traffic and analyses its wireless networking protocols to identify suspicious activity involving the protocols themselves.
  - The *host-based* IDS monitors the characteristics of a single host and the events occurring within that host for suspicious activity.

- Second, we can categorise IDS by their detection methodology. It is the most relevant categorisation for this thesis. Hence, the description is more detailed. Most IDS combine multiple detection methodologies, but the main detection methodologies are as follows:

  - The *signature-based* methods compare known threat signatures to observed events to identify the intrusion. The signatures are created based on the knowledge from previously identified and delineate attacks. However, creating signatures is time-consuming and specific expert knowledge is needed. Thus, this method can be advantageous at detecting already known attacks, but these attacks can use smart evasion techniques rendering this method non-applicable. Other limitations of this method are the increasing number of encrypted traffic, use of self-modifying malware and other evasion techniques.
  - The *stateful protocol analysis* relies on vendor-developed universal profiles that specify how particular protocols should and should not be used. This analysis of "normal" and generally "expected" behaviour of protocols is used for each protocol state against observed events to identify deviations. The weakness of stateful protocol analysis is difficulty or impossibility to develop entirely accurate models of protocols, and restriction to the attacks that violate the characteristics of generally acceptable protocol behaviour.
  - The *anomaly-based* intrusion detection approach is used in this thesis. The fundamental idea behind anomaly-based intrusion detection systems is that an attacker behaviour will be noticeably different from that of a legitimate user and these deviations will be detectable. The anomaly-based intrusion detection system relies on building a statistical model of a normal behaviour and detecting all significant deviations from it. The strength of this approach is that it tries to decrease the human work, mainly manual creation of signatures, and it can be very effective at detecting previously unknown threats. The weakness is that it is hard to develop the statistical model of "normal" behaviour that would not include malicious activity and that would be sufficiently complex to reflect the real-world problem. These issues often lead to comparatively higher false alarm rate, and that is why the pattern based IDS are still widely used even when they are not able to detect new types of attack.

8.2 Anomaly detection

Anomaly detection is a technique of Outlier Analysis, which is design to detect/classify outlier data points.

An outlier data point is significantly different from the remaining data. The formal Hawkins's definition of an outlier is following:

"An outlier is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism."

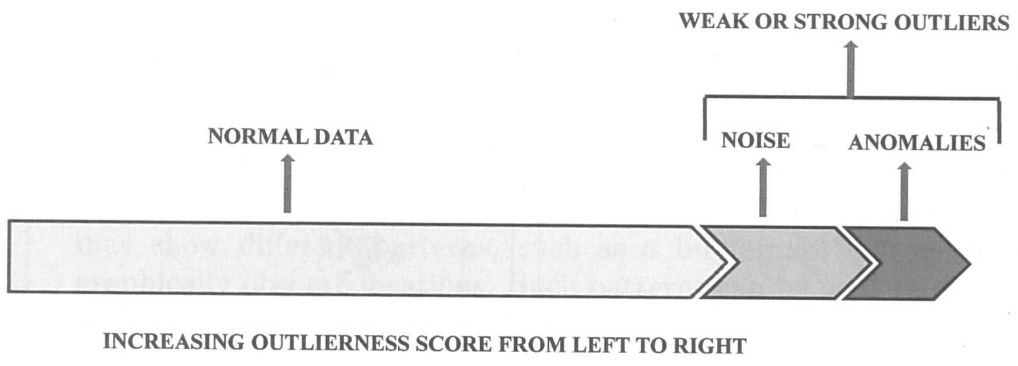


Figure 18 The spectrum from normal data to outliers [27].

Outlier point is also referred as an anomalous point. In most cases, anomalous points are the result of generating process which behaves in an unusual way instead of 0ingbe generated by the normal process, which reflects normal activity in the system. Thus, the data points have a "*normal*" model, and anomalous points are deviations from this normal model. The worth of such anomalous data points is that they can contain useful information about abnormal characteristics of the systems, which impact the data generation process. Such recognition of anomalous data points provides valuable application-specific insights. In our case, we want to discover abnormal usage of online advertisement.

In many real-world applications, it is often a subjective judgement what is a sufficient deviation for a point to be considered as an outlier. The dataset can contain a significant amount of noise which we do not mark as an anomalous data point. Only the significantly interesting deviations are usually anomalous points [27].

8.2.1 Unsupervised Anomaly detection

When previously seen examples of anomalous data points are not available, we speak about the unsupervised scenario. Then the noise represents the semantic boundary between normal data and true anomalies because it does not always meet the robust criteria necessary for a data point to be considered anomalous enough. Therefore, the noise is often modelled as a weak form of outliers, and most of the outlier detection algorithms use some quantified measure of the anomalousness of a data point. Thus, we can represent this property in following way. Every data point lies on a continuous spectrum from normal data to noise, and finally to anomalies as shown in Figure 18. The separation of the individual regions of this spectrum is often not precisely defined, and the delimiter is chosen on an ad-hoc basis according to application-specific criteria. Furthermore, the separations are not pure, and many noisy data points may be deviant enough to be interpreted as anomalies by the outlier score. Thus, the separation criteria are rather up to the expert in the application area to design it [27].

8.2.2 Supervised Anomaly detection

The critical property to distinguish anomalous point from the noise is that anomalies need to be unusual in an interesting way. Thus supervision process re-defines what is interesting. It means that supervised methods need to be designed for application-specific anomaly detection. We distinguish several levels of supervision in practical scenarios.

The first level is a fully supervised scenario when examples of both normal and abnormal data are available and can be readily distinguished.

The second level is semi-supervised scenarios where we do not have samples for both classes. In one semi-supervised scenario, examples of outliers are available, but the examples of "normal" data may also contain outliers in some unknown proportion. In another semi-supervised scenario, only examples of normal data or only examples of anomalous data are available. The variation of the problem depends on the application, and each variation requires a related but dedicated set of techniques [27].

8.2.3 The data model for anomaly detection

Data model is a crucial part of the design of anomaly detection algorithm. The data model describes normal patterns in the data, and then an outlier score of a given data point is computed by the deviation from these patterns by evaluating the quality of the fit between data point and model.

An incorrect choice of the data model may lead to poor results. Therefore, the selection of the model and correct formulation of the initial hypothesis has to be determined by the specialist's understanding of the kind of deviation relevant to the application [27].

The basic oulier models according to [27] are following:

- Extreme Value Analysis, in the basic form, is on 1-dimensional data. The key is to determine the statistical tails of the underlying distribution. The values which are either too large or too small are outlisers.
- In Probabilistic and statistical model, the key assumption is about the choice of the data distribution with which the modelling is performed and parameters of this model are learned.
- Liner model models the data into lower dimensional embedded subspaces with the use of linear correlations.
- Proximity-based model models outliers as points which are isolated from the remaining data. These can be done by cluster analysis, density-based analysis or nearest neighbour analysis.
- Information theoretic model uses summarization methods to provide a small summary of the data, the deviations from which are flagged as outliers.

8.2.4 Categorization

According to [27], anomaly detectors can be categorised into following categories. Firstly, we can categorise it based on the fact that they use some normal user model continuously learned from the data into two groups: static and adaptive.

- The **static** detectors exploit domain knowledge and use a small subset of features and thresholding to detect new anomalous points.
- The **adaptive** detectors maintain models of normal user behaviour or normal network stat and label the deviations from the models as anomalous.

Secondly, we can divide anomaly detectors into two groups based on the type of the output they are producing. The output can be label or score.

- The **score** is a soft decision representing the level of "outlierness" for each observed data point. The advantage of the scoring output is that data points are comparable by their outlier tendency and it retains all the information provided by a particular algorithm. However, it does not provide a concise division between anomalous and normal data points.

- The **label** is a hard decision where each data point has a normal or anomalous label. Although some algorithms directly return binary labels, most of the anomaly detection produce outlier scores. The standard practice is to use thresholding to transform the continuous values to binary labels. Unfortunately, binary labels contain less information, but it provides the final decision which is needed for decision making in practical applications.

8.3 Limitation of anomaly-based intrusion detection system

Although anomaly detection can discover unknown patterns of attacks, it suffers from several problems that need to be discussed before anomaly detection is deployed. These problems include Big Data, the anomaly detector has to process, potential adversarial attack, costly false positive alarms, class imbalance and lack of training and validation data.

8.3.1 Big Data

Network security, where we are applying anomaly detection, is a Big Data problem. Every user connected to the Internet generates network traffic and employees of companies, can together generate easily terabytes of data per day. Thus, we speak about Big Data because the amount of data increase rapidly every day.

Hence, we need to design solution which has low computational space and complexity.

8.3.2 Adversarial attack to anomaly detection

When the attackers know about such detection technique as anomaly detection of the particular type of fraud, then they can adapt their malicious behaviour to fit under the limit of anomalous events. Thus, they make the task of defining normal behaviour even more difficult [28].

8.3.3 Costly false positive alarms

The current anomaly detection approaches usually suffer from a high false-alarm rate and each false positive alarm is very expensive [29]. The network administrator has to react to such false-alarm event, do the deep analysis of the incident to disproof the maliciousness of the event. Such analysis can take up to several working days of the network administrator. Therefore, the research goal is to develop precise anomaly detectors or to find a trade-off between false positives and true positives rate.

8.3.4 Class imbalance

From the definition of anomalies, that anomalies are rare instances in the data, it is expected that class of normal data will be more significant than the class of anomalous data points. Thus the design of supervised techniques used in the anomaly detection should take into consideration this fact.

8.3.5 Labelled data for training and validation

Availability of labelled data for training and validation of statistical models used by anomaly detection is usually an issue. The biggest problem is that anomalies are domain specific and number and type of anomalies vary for each environment. If we would have labelled data for a particular domain and the anomaly we are interested in detecting, it will be pointless to create an anomaly detector. Thus, with such natural property of anomaly, it is almost impossible to have precise data for real word problem. It would be difficult, time-consuming and expensive to develop such data set.

8.3.6 Feature selection

The crucial moment in building anomaly detection models is to pick the features to be used as the input of the statistical model since strong prior expert knowledge is needed. Incorrect feature selection can lead to poor results of anomaly detector.

9 Advertisement fraud detection

For detecting the advertisement fraud in the network traffic, we use anomaly detection approach. In this chapter, we describe the CTA system, advertisement fraud anomaly detector, and incorporation of our advertisement fraud anomaly detector into the CTA system.

9.1 CTA

Proposed advertisement fraud anomaly detector will be part of the Cognitive Threat Analytics (CTA) anomaly detection system, which we describe in detail in this section to be able to explain the integration of our anomaly detector into the CTA system.

The CTA system is developed by Cisco systems. It analyses HTTP proxy logs, produced by Web proxy servers located on a network perimeter, to detect infected computers within the network. The CTA anomaly detection system consists of four main layers. The structure of the system is depicted in Figure 19.

The first layer analyses all network traffic data by *set of 70 anomaly detectors*. Each detector works with a set of selected features that allows identifying specific malicious behaviour by applying various low complexity algorithms enabling the near real-time analysis of the enormous amount of network data. The anomaly detectors are of two different categories.

In the first category, the anomaly detectors are based on a direct application of

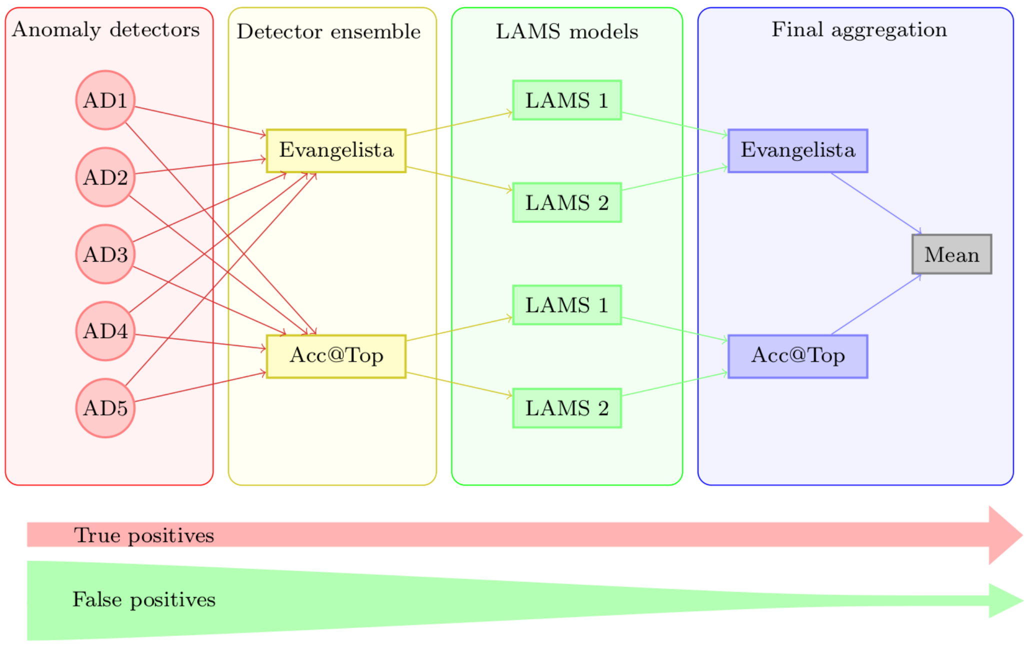


Figure 19 The Cognitive Threat Analystics anomaly detection architecture containing four layers [30].

statistical methods on a selected subset of traffic features. The promising approach to statistical anomaly detection is identified and tested on a large number of traffic feature projections and select one or more combinations.

In the second category, the anomaly detectors are knowledge-based. The development starts by noticing exploitable characteristics of network traffic and analyses whether the traits are relevant for the detection of a network attack. Then the characteristics features are designed and the most straightforward detection algorithm is applied to the features.

Both categories of detectors complement each other and help to achieve better efficacy.

Even though the high sensitivity of the individual detectors is a crucial element of any intrusion detection system, no individual detector can achieve a high enough precision to be useful on its own. Therefore, the ensemble approach is applied.

The second layer aggregates the results from the previous layer by two parallel ensemble systems [31]. The first ensemble system is Evangelista aggregation which use mean of mean and maximum scores of the scores assigned by the individual detectors within the ensemble, as it should be more robust to the presence of poor detectors without knowing which ones are poor. The second ensemble system is parallel aggregation which use labelled data of currently known malicious samples to improve the recall of the whole system on the known network attacks or malware.

The third layer contains several Local Adaptive Multivariate Smoothing (LAMS) models [32], which are designed to reduce the number of false positives of the whole detection system by smoothing the outputs of the ensembles over time and feature space to have the more robust estimate of the true anomaly.

The last layer aggregates the results of individual LAMS models using Evangelista and parallel aggregation ensemble systems. The results of these two ensemble systems are aggregated into the final anomaly score using the mean of the individual anomaly scores of each network flow.

9.2 Advertisement fraud anomaly detector

Online advertisement is an inseparable part of the majority of web pages. We expect that if the user is infected with some adware then the portion of his traffic related to advertisement increases.

The normal behaviour of the user while browsing the Internet consist of loading Web pages, where the non-advertisement flows come first and then they are optionally followed by the flows related to the advertisement.

The deviant behaviour of the infected user is when the significant portion of advertisement flows appear after non-advertisement flows or if non-advertisement flows are missing between the advertisement flows.

The advertisement fraud anomaly detector should give the high score for the users who are infected with some adware. Hence, their traffic contains enormous portion of advertisement flows to gain as much money as possible to generate profit for the attacker.

9.2.1 Feature design

The proposed advertisement fraud anomaly detection algorithm is based on the fact that the normal users consume the amount of advertisement flows proportional to the amount of non-advertisement flows. The advertisement flows are detected by our network mining algorithm (see Section 6.4.2). We reflect the relationship between advertisement flows and non-advertisement flows in the ratio in the following way.

We design the feature for each user u as a ratio $\delta(u)$ of number of user's advertisement flows $\alpha(u)$ and total number of user's flows $\tau(u)$,

$$\delta(u) = \frac{\alpha(u)}{\tau(u)}. \tag{1}$$

This relation reflects the general relationship between advertisement flows and non-advertisement flows, that advertisement flows should not appear without the non-advertisement flows. The ratio gives us information, what portion of user's traffic is the advertisement traffic. We expect that advertisement flows will not take more than half of the traffic of a normal user, because otherwise it would mean that the main content which the user consumes at the Internet is an advertisement. Hence, we expect that majority of normal users will have the ratio $\delta(u)$ smaller than 0.5 and infected users will have the ratio close to 1.

However, the disadvantage of this feature is the precondition that we have to have proper labels of advertisement and non-advertisement flows. The advertisement detection is discussed in detail in Chapter 6 where we proposed the new algorithm for advertisement network mining. The other problem is that part of the user's traffic is non-encrypted (HTTP) and part is encrypted (HTTPS). The non-encrypted traffic generates significantly more flows than the encrypted traffic because for the HTTPS we can see only initial connection to set up the tunnel and the rest of the flows we do not see as it is described in Section 5.3. But we expect that this will not be a significant problem because the ratio of the number of advertisement flows and non-advertisement flows should stay the same. Even in HTTPS, we see the initial connection to load the content of web page and the initial connection to load the advertisement.

We compute the ratio for each user in our one-week data who was connected to the Internet for at least four days. It means that the difference between their first and last connection to the Internet was at least four days and they produce at least 10K flows during the week. We need this restriction because in another way the ratio would be skewed. For example, the user who connects only for a few minutes and has already loaded the page with advertisement would have only the advertisement flows because the web page will load only new advertisement but no non-advertisement traffic.

The distribution of proposed feature is in Figure 20. The y-axis is in the logarithmic scale because the number of users with the value of the ratio close to the zero is the majority. The ratio was computed for 57 312 users. Out of the total number of users, 12 660 (22%) users have the ratio lower than 0.1, and 41 442 (72%) users have the ratio lower than 0.5. Only 1850 (3%) users have the ratio higher than 0.8.

9.2.2 Anomaly score: Z-value test

The anomaly score for our algorithm is calculated using a statistical model for outlier analysis known as *Z-value test* [27]. The score is defined in following way:

Consider a set of 1-dimensional quantitative data observations, denoted by $X_1...X_N$, with mean μ and standard deviation σ. The Z-value for the data point X_i is denoted by Z_i , and is defined as follows:

$$Z_i = \frac{X_i - \mu}{\sigma}. \tag{2}$$

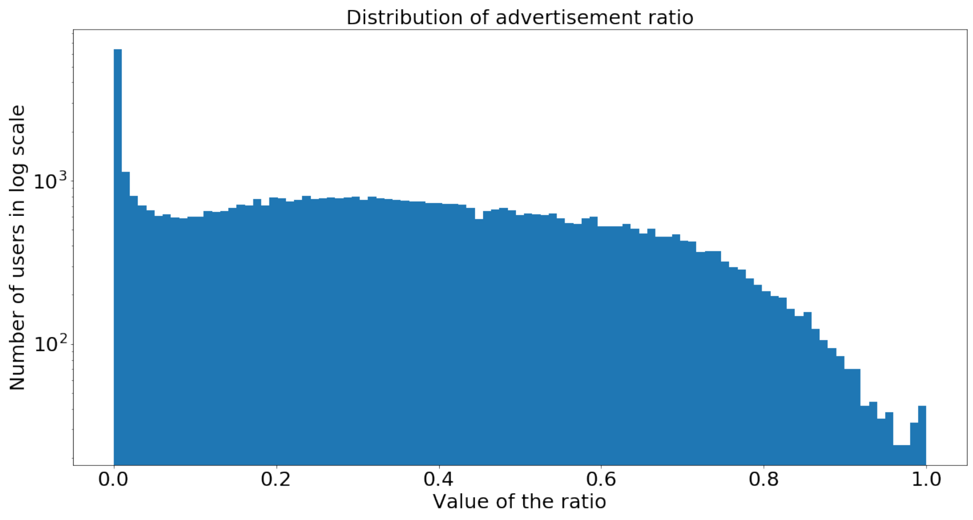


Figure 20 Distribution of advertisement ratio.

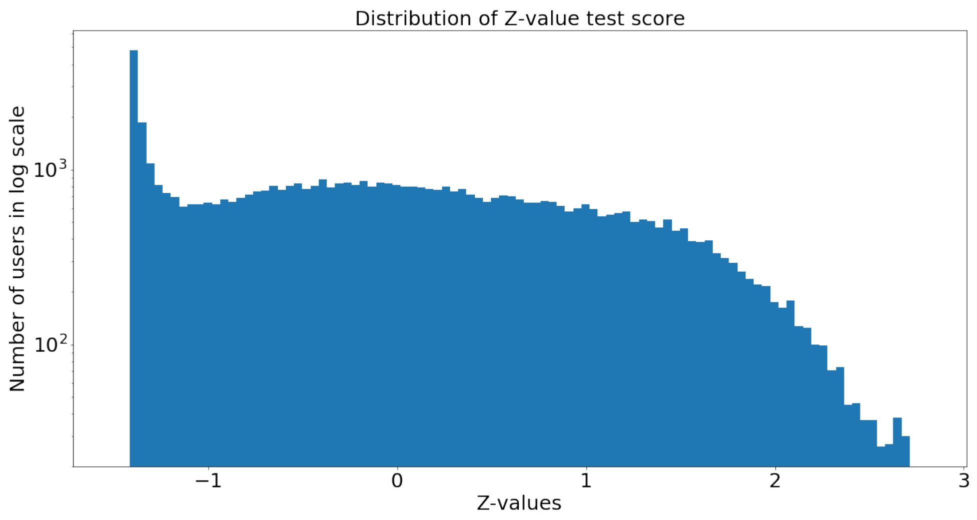


Figure 21 Z-score distribution.

The Z-value test computes the number of standard deviations by which the data varies from the mean. This definition of anomaly has an implicit assumption that the data is drawn from a normal distribution. When this is not the case, the corresponding Z-values needs to be interpreted carefully [27].

The anomaly score for ratio feature is in Figure 21, where the y-axis is in the logarithmic scale. We can see that the anomaly score distribution has a heavy tail on the right with blob close to the 2.5σ. These users close to 2.5σ can be suspect from being infected with adware.

9.2.3 Normalisation / Unifying outlier scores

The advertisement fraud anomaly detector will be part of CTA network intrusion detection systems, and for this reason, we need our outlier score to be comparable with other anomaly detectors in the system.

Since different outlier scores may differ in their meaning, range, and the contrast between different outlier models, they are not easily comparable or interpretable. The other problem, with varying scores of the outlier, is that they often lack the contrast between outliers and inliers.

Hence, if we want to compare or combine output score of several anomaly detectors easily, we need to unify the outlier scores and facilitate types of functions to transform an outlier score to a comparable, normalised value in the range $[0, 1]$ or even to a probability value.

Any framework for transforming outlier scores [33] has to grantee that the resulting score fulfils the term of regularity and normality and the transformation is ranking-stable.

- An outlier score S is called **regular** if $S(o) \geq 0$ for any object o, $S(o) \approx 0$ if o is an inlier, and $S(o) \gg 0$ if o is an outlier.
- An outlier score S is called **normal** if S is regular and the values are restricted by $S(o) \in [0, 1]$.
- The transformation of an outlier score T_S is **ranking-stable**, when it does not change the ordering obtained by the original score. Formally the ranking-stable transformation fulfil the requirement for any o_1, o_2 :

$$S(o_1) \leq S(o_1) \Rightarrow T_S(o_1) \leq T_S(o_2) \tag{3}$$

Formally, we can apply two steps in the process of unification scores, where either step can be optional. First, we can apply regularisation Reg that transforms a score S onto the interval $[0, \infty)$ such that $Reg_S(o) \approx 0$ for inliers and $Reg_S(o) \gg 0$ for outliers. Second, normalisation transforms a score onto the interval $[0, 1]$. Both steps can be used to enhance the contrast between inliers and outliers. Thus even an already normal scores can be unified to stretch interesting regions and shrink irrelevant intervals.

The simplest normalisation method is a linear transformation. It brings an outlier score onto a normalised scale by applying a linear transformation. The minimum score is mapped to 0, and the maximum score is mapped to 1. Then simple linear normalisation is obtained by:

$$Norm_S^{linear}(0) := \frac{S(o) - S_{min}}{S_{max} - S_{min}} \tag{4}$$

This simple method maps values to the normalised interval, but it does not add any contrast to the distribution of scores. Thus, we search for other methods to enhance the difference between inliers and outliers based on some exemplary statistical scaling methods.

For the integration our anomaly detector into CTA, we use more complicated **Gaussian Scaling method** proposed by Kriegel at al. [33], which enhance the contrast between inliers and outliers based on exemplary statistical scaling methods. The method is based on the precondition that according to the central limit theorem, the most general distribution for values derived from a large number of similarly distribute values is the normal distribution. The method is not susceptible to over-fit because it has just two degrees of freedom, the mean μ and the standard deviation σ.

The normalisation $Norm_S^{gauss}(o)$ of the anomaly score $S(o)$ is assigned to each user $o \in O$ by the anomaly detector S. Then we have the mean μ_S and the standard deviation σ_S of the set of derived values using the outlier score S. These can be estimated from the data using $\widehat{\mu_S} = E(S)$ and $\widehat{\sigma_S} = \sqrt{E(S^2) - E(S)^2}$ in a single run or adaptively adjusted when running in an on-line scenario. Finally, we can use these values to compute cumulative distribution function and the Gaussian error function $erf()$. The Gaussian Error Function $erf()$ satisfy an important property of transformations

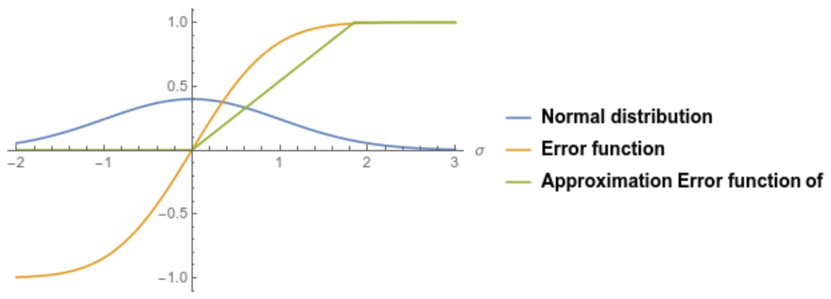


Figure 22 The approximation of Error function. The y-axis has values of PDF for Normal distribution, Anomaly Error values of Error function and Anomaly Approximation values fo rApproximation of Error function.

that they should not change the ordering obtained by the original score, because it is monotone and thus ranking-stable. The equation for computing Gaussian Scaling transformation is following:

$$Norm_S^{gauss}(o) := \max\left\{0, erf\left(\frac{S(o)-\mu_S}{\sigma_S \cdot \sqrt{2}}\right)\right\} \tag{5}$$

This transformation transforms the anomaly scores of an anomaly detector into probability estimates, where the probability of zero represents the normal user, aligned with the model, whereas one indicates highly anomalous user.

The Gaussian Scaling method is based on Z-value score which is divided by the constant $\sqrt{2}$ and transformed by the Gaussian error function.

Unfortunately, the Gaussian Error Function is computationally expensive, hence, it is not convenient to use it in a solution for Big Data. Therefore, we approximate the Gaussian Error Function as shown in the Figure 22. The approximation is zero for negative values of σ, and then the values from 0σ to 1.85σ are linearly scaled onto the values from the interval $[0, 1]$. The values higher than 1.85σ are equal to one. This approximation function approximates the error function well enough for our application, and it is easy to compute. Thus, we use proposed approximation function in the formula for Gaussian Scaling transformation $Norm_S^{gauss}(o)$ instead of the error function.

The score transform in a such a way can be directly compared with another anomaly score transformed in the same way, and they can be aggregated using many combination techniques.

The normalised anomaly score is in the Figure 23, where the y-axis is in the logarithmic scale. We can see that the majority of users have the normalise anomaly score equal to zero, and then the number of users decreases with increasing value of the normalised anomaly score up to one. The users with the normalised anomaly score equal to one take 0.28% of the total number of users. We can expect that these users are infected with some adware and the probability that the user is infected with adware decrease with decreasing value of the score.

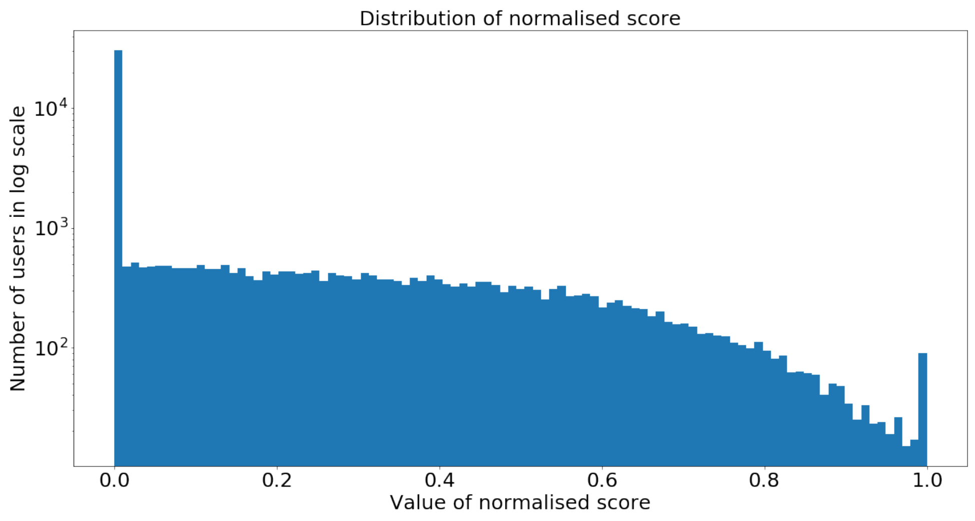


Figure 23 Distribution of normalised Z-score values.

10 Evaluation of advertisement fraud anomaly detector

In this chapter, we evaluate the advertisement fraud anomaly detector proposed in Chapter 9. The available labels for anomaly detection are an issue. Hence, the evaluation of proposed solution is limited as we usually want to discover new types of attacks for which we do not have an intelligence. Nevertheless, we have an access to two different types of labels, but unfortunately, these labels do not perfectly fit our problem as discussed in Section 10.1.2. Thus, we have to analyse the result of proposed solution manually to get an appropriate information about the success rate in Section 10.2.2.

10.1 Evaluation with available labels

10.1.1 Long term adware captures

We have an access to long-term sandbox samples with different types of infection captured by Stratosphere IPS [34]. The advantage of these samples is that they are captured for a long period (weeks or months). On the other hand, the disadvantage of these samples for our research is that they do not contain user interaction and they are not categorised by the severity or the type of threat. Thus, we have to decide whether the sample is related to advertisement fraud from the name of the sample, which is not precise in all cases, or the behaviour in the sample. Hence, specific expert knowledge is needed during the manipulation of the samples.

Unfortunately, the number of advertisement fraud oriented captures in the Stratosphere IPS dataset is limited. Particularly, we find only eight different types of captures related to advertisement fraud, but they do not leave a significant ad-related footprint in the captured network traces that could be detected by the proposed anomaly detector. Hence, the anomaly score is very low for these samples. The list of analysed samples with their probable names and number of identified advertisement requests according to proposed advertisement network mining algorithm is in Table 4.

We manually evaluate the advertisement request labelling generated by proposed advertisement detection algorithm and all hostnames related to the advertisement are labelled correctly as the advertisement except "ad.bench.utorrent.com" and "cdn.bitmedianetwork.com", which we do not detect as an advertisement. However, these hostnames are not popular advertisement servers hostnames.

The low amount of network traffic related to advertisement delivery in analysed samples may be caused by the lack of user interaction during the sandboxed run of the malware sample or the revenue, which is generated by the huge amount of advertisement requests, is not the primary objective of the analysed malware samples.

10.1.2 CTA threat intelligence

The CTA has threat intelligence, which contains manually created set of indicators of compromise for different categories of threats. Several categories are related to advertisement fraud as click fraud, ad-injection and malicious advertising. We analyse the

Capture name	Probable infection name	Ad	Total	Ratio
CTU-Malware-Capture-Botnet-169-2	Miuref	435	18 583	0.02
CTU-Malware-Capture-Botnet-177-1	Adware.Win32.Amonetize.heur?	0	89 539	0.00
CTU-Malware-Capture-Botnet-337-1	Unknown.PUA	0	20	0.00
CTU-Malware-Capture-Botnet-237-1	PUA.Taobao	0	29 804	0.00
CTU-Malware-Capture-Botnet-222-1	PUP.Plumbytes	0	1 464	0.00
CTU-Malware-Capture-Botnet-193-1	Win32/Bundled.Toolbar.Google.D	14	1 064	0.00
CTU-Malware-Capture-Botnet-194-1	OpenCandy	3997	27 086	0.15
CTU-Malware-Capture-Botnet-199-1	MediaGet	142	4 051	0.04

Table 4 Overview of eight advertisement fraud related samples captured by Stratosphere IPS.

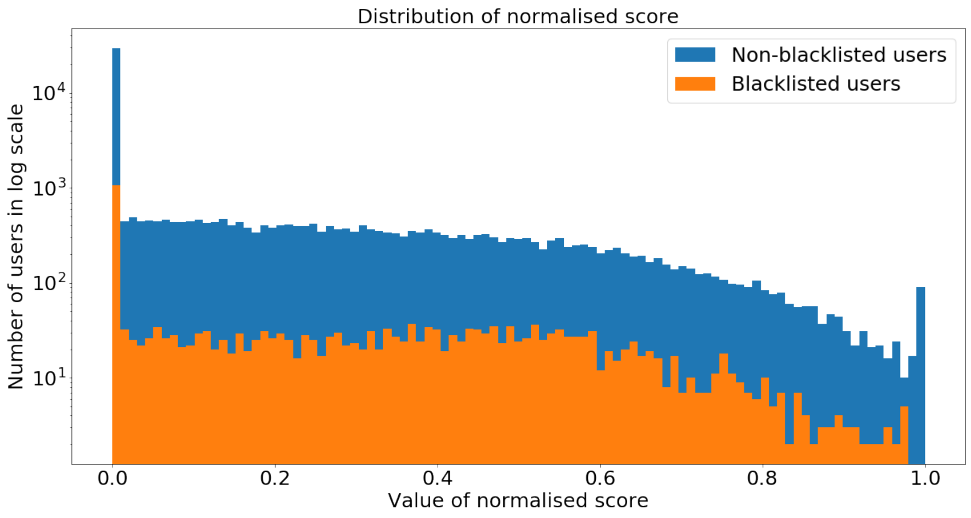


Figure 24 Distribution of normalised score of blacklisted users by CTA threat intelligence and non-blacklisted users.

behaviour of users, which match the advertisement fraud categories, and only minority of them has the malicious behaviour we are searching for. The majority of infected users were not infected with advertisement fraud, which generates a huge volume of advertisement requests to generate revenue.

The traffic of blacklisted users, which match the CTA advertisement fraud intelligence, do not contain massive communication to popular ad-servers, only communication with the C&C. Hence, they cannot be taken as ground truth for proposed anomaly detector. However, we can use it to evaluate the distribution of different advertisement frauds against proposed anomaly detector score.

The distribution of blacklisted users is in Figure 24. We can see that the amount of blacklisted users is meagre. Thus we have to use the log scale on the y-axis to be able to evaluate it. Blacklisted users have variate behaviour regarding the anomaly score. This indicates, that the advertisement fraud is a broad area, which can appear in different forms.

10.2 Manual evaluation

In this section, we present our manual analysis of top 0.1% of most anomalous users according to anomaly score, because available labels do not sufficiently cover the threats we want to detect. First, we describe the methodology, we use to analyse the user traffic.

Second, we present results of our analysis.

10.2.1 Methodology of manual analysis

Crucial part of every manual evaluation is analytic method, because it can significantly reduce time of the analysis. We develop analytic method, which help us to recognise advertisement fraud and minimise the time we have to spend with each user.

The methodology consists of frequency analysis of the occurrence of each hostname the user visits and the knowledge if the hostname belongs to ad-server.

In each user traffic we search for:

- if the hostnames labelled as an ad-servers are the true ad-servers,
- if the distribution of hostnames contains normal user behaviour which could generate such amount of advertisement traffic,
- for C&C hostname, which would prove the infection.

All the most anomalous users, we analyse, have the portion of advertisement flows higher than 95%. This is already strong indication of malicious behaviour related to advertisement fraud. However, the users are not infected with advertisement fraud if one of the following situation occurs:

- Some hostnames in a user traffic are labelled as an advertisement hostnames while they are not. Hence, the ratio is negatively impacted by the inaccurate advertisement detection algorithm.
- The massive advertisement traffic is generated by a legitimate behaviour in the user traffic.

The manual analysis divide the users into the three following groups:

- The first group contains confirmed TP (true positives). The traffic of users from this group does not contain user interaction at all or interaction which would be responsible for the anomalous portion of advertisement. In addition, we find hostname of C&C which was confirmed by some public threat intelligence, e.g. ThreatGrid.
- The second group contains TP (true positives). The traffic of users from this group does not contain user interaction at all or interaction which would be responsible for the anomalous portion of advertisement.
- The last group contains FP (flase positives). Some hostnames in a user traffic are wrongly labelled as ad-servers, while they are not the true ad-servers and it negatively affect the ratio. The user traffic contains legitimate behaviour which caused the enormous advertisement traffic.

10.2.2 Manual evaluation of proposed advertisement fraud anomaly detector

We analyse 0.1% most anomalous users according to anomaly score and divide them into three categories confirmed TP, TP and FP, described in Subsection 10.2.1. There is 73 most anomalous users with normalised score equal to one. All these users have more than 96% of their traffic related to advertisement labelled by proposed advertisement mining algorithm.

Out of 73 most anomalous users, we label 48 users as confirmed TP, 22 as TP and three as FP.

Example of confirmed TP is one user, which massively communicates to popular ad-server's hostnames and to non-ad-hostname "Cj.dotomi.com". This hostname is according to available intelligence (Virustotal, Malwaretips, Fixyourbrowser, VirusResearch and Malwarefixes) related to pop-up ads. Thus, we can confirm from amount of advertisement traffic and references for "Cj.dotomi.com" that the user is infected with adware.

The users labelled as TP, contain in general massive communication to popular ad-servers where one ad-server hostname is visited significantly more often than the other ad-server's hostnames.

All three FPs share the same behaviour, which does not contain user interaction. They contain one hostname "query.yahooapis.com" with high occurrence, which is mislabelled as an advertisement even though it is a hostname that can be used for many non-advertisement related purposes. This mislabelled advertisement hostname causes increase of the anomaly score of the user.

To conclude, we show that our anomaly detector can detect new advertisement fraud, which was not covered by the CTA threat intelligence before.

11 Future Work

In this thesis, we open the chapter of advertisement fraud anomaly detection. Future work concerns detection of other types of advertisement fraud and anomalous user behaviour in the network.

The network mining algorithm for advertisement traffic detection described in Section 6.4 creates many opportunities to deeply explore advertisement networks from the user site or to detect other types of advertisement fraud as we are already able to distinguish advertisement requests, which was the biggest issue of our research. The following ideas could be tested:

- Threat Mining using several runs of ad-propagation and back-propagation can give wider coverage of the advertisement providers and ad-servers they are serving.
- Modelling typical ad-server for popular sites to be able to detect ad-injections or less noisy ad-frauds.

12 Conclusion

The amount of online advertisement is growing significantly every year, and the process of advertisement delivery to the audience is becoming more and more sophisticated. It uses knowledge about previous user behaviour on the Internet to precisely target the advertisement to the right user. The increasing revenue of online advertisement attracts more and more attackers who are interested to explore new vulnerabilities to find new ways to infect the broad audience with adware. Once the attackers find the way to infect a user with adware, there is a high chance that adware escalates to higher severity threats. As the attacker's revenue is mainly related to the amount of fake impressions, the attackers are motivated to create malware that generates a large number of advertisement requests, which waste the resources. Subsequently, the high number of advertisement requests can slow down user's device and negatively impact user experience while browsing the internet and also disturb the user's privacy.

In Chapter 2, we review the related work in analysis and detection techniques of advertisement frauds. However, the amount of research in this field is very limited and mainly focused on the fraud detection from the ad-provider's point of view. Therefore, the existing state of the art in advertisement fraud detection cannot directly be applied to the web proxy access log data.

In this thesis, we have first reviewed the process of advertisement delivery in Chapter 3 with more detailed analysis of the advertisement delivery mechanism as seen by the web proxies with a focus on the ad-exchange networks in Section 3.1.1. Additionally, in Section 3.2, we review the existing advertisement fraud approaches with a description of the approaches that generate a large number of malicious traffic. These are the types that we focus on while designing our advertisement fraud anomaly detector.

In Chapter 6, we propose an algorithm to detect ad-related HTTP/HTTPS requests captured by web proxies. The algorithm is able to extend a limited input information about one particular structure of the existing advertisement provider to cover 384 863 ad-server's hostnames. In Chapter 7, we show that the proposed algorithm can detect ten times more ads than the conventional pattern matching approaches like Yoyo and EasyList ad-block lists. Accordingly, we discovered that advertisement takes $20,7\%$ of all traffic. Additionally, the proposed detection mechanism does not need any human interventions nor manual updates, but can periodically run to detect new ad-servers and ad-networks.

In Chapter 9, we propose a network anomaly detection algorithm that uses the above detected ad-related traffic to model the percentage of the ad-related traffic for each user in the monitored network. The ratio of the individual user is compared to the ratios of all other users in the network to access the anomaly score of the user. In Section 9.2.3, we propose a normalisation method for the anomaly score to be able to incorporate the proposed anomaly detector into the existing Cognitive Threat Analytic anomaly-based intrusion detection system, developed by Cisco Systems.

Finally, the proposed anomaly detector was evaluated on HTTP and HTTPS traffic of 57 312 users giving us 73 most anomalous users. Out of 73 most anomalous users, we manually label 48 users as confirmed TP with found C&C to prove the infection, 22 as TP with containing malicious behaviour but without found C&C to prove the infection, and three as FP.

We show that the proposed solution may be used for the detecting advertisement fraud.

Appendix A
CD content

- 2018-ondrackova-thesis.pdf
- LaTeXsource code of the Thesis

Bibliography

[1] CB. *ZenithOptimedia predicts global ad expenditure to return to pre-recession peak level this year.* [online]. [cit. 11. 5. 2018]. URL: `http://www.campaignbriefasia.com/2011/07/zenithoptimedia-predicts-globa.html`.

[2] Jacqui Cheng. *Ad-injecting trojan targets Mac users on Safari, Firefox, and Chrome.* [online]. [cit. 11. 5. 2018]. URL: `https://arstechnica.com/gadgets/2013/03/ad-injecting-trojan-targets-mac-users-on-safari-firefox-and-chrome/`.

[3] Ashlee Vance. *ZenithOptimedia predicts global ad expenditure to return to pre-recession peak level this year.* [online]. [cit. 11. 5. 2018]. URL: `https://www.nytimes.com/2009/09/15/technology/internet/15adco.html`.

[4] Dyfed Loesche. *Digital (Finally) Killed the TV Star.* [online]. [cit. 18. 01. 2018]. URL: `https://www.statista.com/chart/12136/worldwide-digital-and-tv-ad-spending/`.

[5] Dyfed Loesche. *Where's Digital Advertising Headed?* [online]. [cit. 31. 01. 2018]. URL: `https://www.statista.com/chart/9043/digital-advertising-revenue-worldwide/`.

[6] Lara O'Reilly. *The ad fraud issue could be more than twice as big as first thought — advertisers stand to lose $16.4 billion to it this year.* [online]. [cit. 11. 4. 2018]. URL: `http://www.businessinsider.com/thepartnership-msix-and-adloox-ad-fraud-2017-2017-3?utm_source=feedburner&utm_medium=feed&utm_campaign=Feed:%20typepad/alleyinsider/silicon_alley_insider%20%28Silicon%20Alley%20Insider%29&utm_term=BII%20List%20DMedia%20ALL`.

[7] Margaret Boland. *Cyber criminals are stealing billions from the ad industry each year.* [online]. [cit. 11. 4. 2018]. URL: `http://www.businessinsider.com/the-ad-fraud-report-bot-traffic-2016-3`.

[8] Zhou Li et al. "Knowing your enemy: understanding and detecting malicious web advertising". In: *Proceedings of the 2012 ACM conference on Computer and communications security.* ACM. 2012, pp. 674–686.

[9] D Sculley et al. "Detecting adversarial advertisements in the wild". In: *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM. 2011, pp. 274–282.

[10] Gianluca Stringhini, Christopher Kruegel, and Giovanni Vigna. "Shady paths: Leveraging surfing crowds to detect malicious web pages". In: *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security.* ACM. 2013, pp. 133–144.

[11] Tommy Blizard and Nikola Livic. "Click-fraud monetizing malware: A survey and case study". In: *Malicious and Unwanted Software (MALWARE), 2012 7th International Conference on.* IEEE. 2012, pp. 67–72.

[12] Miguel Helft and Tanzina Vega. "Retargeting ads follow surfers to other sites". In: *The New York Times* (2010), pp. 8–11.

[13] *How does browser tracking work?* [online]. [cit. 18. 01. 2018]. URL: `https://myshadow.org/browser-tracking`.

[14] Lauren Drell. "4 ways behavioral targeting is changing the web". In: *Mashable Digital Marketing Series. http://mashable. com/2011/04/26/behavioraltargeting* (2011).

[15] Koen W De Bock. *Advanced database marketing: innovative méthodologies and applications for managing Customer relationships.* Routledge, 2016.

[16] Barry Adams. *Geotargeting Based On IP Address.* [online]. [cit. 10. 01. 2018]. URL: `http://www.stateofdigital.com/geotargeting-based-on-ip-address-is-broken/`.

[17] IAB Interactive Advertising Bureau. *How an Ad is Served with Real Time Bidding (RTB) - IAB Digital Simplified.* [online]. [cit. 10. 01. 2018]. URL: `https://www.youtube.com/watch?v=-Glgi9RRuJs`.

[18] *Online advertising?* [online]. [cit. 10. 01. 2018]. URL: `https://en.wikipedia.org/wiki/Online_advertising#cite_note-17`.

[19] Anne Do. *Supply Side Platforms : Where do they fit in the digital ecosystem?* [online]. [cit. 10. 01. 2018]. URL: `http://blog.adtrue.com/2017/06/26/how-supply-side-platforms-digital-ecosystem/`.

[20] Jack Marshall. *WTF is a data management platform?* [online]. [cit. 10. 01. 2018]. URL: `https://digiday.com/media/what-is-a-dmp-data-management-platform/`.

[21] *Wappalyzer.* [online]. [cit. 31. 01. 2018]. URL: `https://www.wappalyzer.com/categories/advertising-networks`.

[22] Ashlee Vance. *CISCO Web Security Appliance.* [online]. [cit. 20. 5. 2018]. URL: `https://www.cisco.com/c/en/us/products/security/web-security-appliance/index.html`.

[23] *Adblock Plus filter.* [online]. [cit. 3. 5. 2018]. URL: `https://adblockplus.org`.

[24] Peter Lowe. *Ad server list to block ads.* [online]. [cit. 31. 01. 2018]. URL: `https://pgl.yoyo.org/as/serverlist.php?showintro=0;hostformat=hosts`.

[25] *EasyList.* [online]. [cit. 31. 01. 2018]. URL: `https://easylist.to/`.

[26] Karen Scarfone and Peter Mell. "Guide to intrusion detection and prevention systems (idps)". In: *NIST special publication* 800.2007 (2007), p. 94.

[27] Charu C Aggarwal. "Outlier analysis". In: *Data mining.* Springer. 2015, pp. 237–263.

[28] Varun Chandola, Arindam Banerjee, and Vipin Kumar. "Anomaly detection: A survey". In: *ACM computing surveys (CSUR)* 41.3 (2009), p. 15.

[29] Peng Ning and Sushil Jajodia. "Intrusion detection techniques". In: *The Internet Encyclopedia* (2003).

[30] Martin Grill. "Combining Network Anomaly Detectors". PhD dissertation. Czech Technical University in Prague, 2016.

[31] Martin Grill and Tomáš Pevný. "Learning combination of anomaly detectors for security domain". In: *Computer Networks* 107 (2016), pp. 55–63.

[32] Martin Grill, Tomáš Pevný, and Martin Rehak. “Reducing false positives of network anomaly detection by local adaptive multivariate smoothing”. In: *Journal of Computer and System Sciences* 83.1 (2017), pp. 43–57.

[33] Hans-Peter Kriegel et al. “Interpreting and unifying outlier scores”. In: *Proceedings of the 2011 SIAM International Conference on Data Mining*. SIAM. 2011, pp. 13–24.

[34] Stratosphere IPS. *Malware Capture Facility Project*. [online]. [cit. 4. 5. 2018]. URL: `https://www.stratosphereips.org/datasets-malware/`.