PRESERVING PRIVACY THROUGH CRYPTOGRAPHY IN ONLINE BEHAVIOURAL ADVERTISING

LEON J. HELSLOOT

to obtain the degree of Master of Science in Computer Science
Data Science & Technology Track
with a specialisation in Cyber Security
to be defended publicly on August 30, 2017

DELFT UNIVERSITY OF TECHNOLOGY
Faculty of Electrical Engineering, Mathematics & Computer Science
Department of Intelligent Systems
Cyber Security Group

Leon J. Helsloot: *Preserving Privacy through Cryptography in Online Behavioural Advertising,* © August 30, 2017

STUDENT NUMBER:
4235991

THESIS COMMITTEE:
Prof.dr.ir. R.L. Lagendijk
Prof.dr. A. Hanjalic
Prof.dr. C. Witteveen
Assist.Prof.Dr. Z. Erkin
G. Tillem, MSc

SUPERVISORS:
Assist.Prof.Dr. Z. Erkin
G. Tillem, MSc

ABSTRACT

Online advertising is a multi-billion dollar industry, forming one of the primary sources of income for many publishers that offer free web content. Online Behavioural Advertising (OBA), the practice of showing advertisements based on individual users' web browsing behaviours, greatly improves the expected effectiveness of advertisements, and is believed to be beneficial to advertisers and users alike. The privacy of users, however, is threatened by the widespread collection and exchange of users' browsing behaviour by dozens of companies for the purpose of behavioural advertising.

The aim of this thesis is to alleviate these privacy concerns by investigating how an online advertising system can serve advertisements tailored to users' interests within the current online advertising ecosystem, such that no party other than the user themselves can gain any knowledge of the user's interests. To protect user privacy in the online advertising ecosystem, two main challenges need to be overcome. Not only do advertising companies need to adapt their machine learning models to a setting in which they have no knowledge of user interests, they also need to integrate their privacy-preserving targeting systems into a complex advertising landscape where advertisement impressions are traded within a fraction of a second.

We present two complete privacy-preserving protocols for online behavioural advertising that combine machine learning methods commonly encountered in existing advertising systems with secure multi-party computation techniques. The first protocol uses a threshold variant of an additively homomorphic cryptosystem to distribute trust between parties while allowing computations on encrypted data, such that advertisements can be served based on detailed user profiles. The second protocol distributes trust between advertising companies using an additively homomorphic threshold secret sharing scheme, allowing collaborative computations on user profiles while preventing a coalition of colluding parties smaller than a predefined threshold from obtaining any sensitive information. Both protocols achieve performance multi-linear in the size of user profiles and the number of advertising campaigns, and show promising initial results in terms of privacy and performance. To the best of our knowledge, our two protocols are the first protocols that preserve user privacy in behavioural advertising while allowing the use of detailed user profiles and machine learning methods.

PREFACE

Just two years ago, when I started the Computer Science MSc programme, I couldn't have imagined writing a thesis on applied cryptography, or anything related to machine learning, let alone a combination of the two. After all, cryptography is hard, as is machine learning, and I am really not that smart. Yet the formidable teaching skills of some of the professors here at the Delft University of Technology have managed to steer my interests in directions I could not have foreseen. Not only did they do a magnificent job of explaining the intuition behind topics that I did not even know existed, they also somehow managed to make those topics interesting and, at times, even fun. But most importantly, they gave me the confidence that I could do things I could not even imagine before.

Special thanks go out to Zeki Erkin and Gamze Tillem, my thesis supervisors, for guiding me through the overwhelming number of possible directions in which this topic could have taken me. Without your guidance and quick feedback I would certainly have lost sight of the big picture. But more importantly, thank you Zeki for seeing beyond hard results and acknowledging the human side of writing a thesis. The stress-free environment you create and the level of involvement between you, your PhD candidates and your students provided me with more support than any number of technical discussions could have. Speaking of which, I would like to thank Gamze, Chibuike and Oğuzhan for helping me whenever I had questions – and even when I did not.

Finally, I would like to thank the friends who helped me through those moments in which nothing made sense (you know who you are!). Thank you for not expecting me to be smart all the time, for listening whenever I had something to say and for showing me the flaws I could not see myself. I would be nowhere without you.

Leon Helsloot
Delft, August 2017

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF PROTOCOLS

ACRONYMS

OBA	Online Behavioural Advertising
RTB	Real-Time Bidding
AdX	Ad Exchange
DSP	Demand-Side Platform
SSP	Supply-Side Platform
DMP	Data Management Platform
CTR	Click-Through Rate
CVR	Conversion Rate
ODP	Open Directory Project
LDA	Latent Dirichlet Allocation
SGD	Stochastic Gradient Descent
FM	Factorization Machine
FFM	Field-aware Factorization Machine
DNN	Deep Neural Network
DNT	Do Not Track
PII	Personally Identifiable Information
SC	Secure Coprocessor
CDN	Content Delivery Network
PIR	Private Information Retrieval
ORAM	Oblivious RAM
TTP	Trusted Third Party
PSP	Privacy Service Provider
FTRL	Follow The (Proximally) Regularized Leader

1 INTRODUCTION

Online advertising is a multi-billion dollar industry, forming one of the driving economic factors behind free web services [35, 85]. With a worldwide spend of $178 billion in 2016 [62], advertising has become a pervasive phenomenon on the internet as we know it today, allowing publishers to offer content to users free of charge. In the past few years, however, an increasing number of people have chosen to prevent advertisements from being shown on web pages they visit. Blocking advertisements has become as easy as a few mouse clicks with the availability of a class of web browser extensions named ad blockers. Installation of the *uBlock Origin* ad blocker for Mozilla Firefox[1], for instance, required only three mouse clicks. An estimated 615 million devices use such ad blockers, which amounts to 11% of the global internet population, and the number is expected to grow further [76].

The consequence of the increased use of ad blockers is that publishers suffer from a significant decrease in revenue from the advertising space they offer. The worldwide loss of advertising revenue due to ad blocking was estimated to be $41.4 billion in 2016, which is 23% of the total ad spend [77]. In an effort to address the economic impact of ad blocking, some publishers now request users of ad blockers to allow ads on their websites, pay for content that was previously offered free of charge, or deny access to users of ad blockers altogether [9]. Such practices have sparked an arms race of technologies to block advertisements, and to circumvent blockage. These developments are a threat to the business models of many websites with freely available content. For a sustainable ad-supported internet economy, it is necessary to address the objections users have to advertisements.

In a recent survey among users of a popular ad blocker, 32% of respondents indicated that privacy concerns were among the reasons for using an ad blocker [9]. Similarly, in a 2011 survey about privacy and advertising among United States residents, 94% of respondents considered online privacy an important issue, and 70% of respondents believed that online advertising networks and online advertisers should be responsible for safeguarding an individual's online privacy [97]. The practice of selecting advertisements based on users' browsing behaviour, and the data collection required for such targeting, is one of the factors leading to privacy concerns [92, 98]. Although advertisements tailored to users' interests are recognized as being useful for both users and publishers, users are reluctant to

1 https://www.ublock.org/

accept behaviourally targeted advertisements due to a mistrust of advertising companies. Moreover, users experience a lack of control over what information is collected about them [98]. This chapter gives an overview of the current trend in online advertising, then describes the practice of online behavioural targeting, and finally explains the privacy issues of these practices.

1.1 REAL-TIME BIDDING

Since the appearance of the first online advertisement in 1994, the online advertising industry has experienced massive growth, and with it rapid innovation [99]. The first Internet advertisements were shown in a 'scattershot' approach, much like traditional advertising media such as television, radio, newspapers and billboards [104]. Unlike these traditional media, however, the Internet has a much more interactive nature. Rather than consuming whatever happens to be sent their way by publishers, users explicitly request information on the Internet. Advertisers therefore have a unique view into the behaviour of users, allowing a precise targeting of advertisements at the relevant audience. Such new opportunities, along with advances in computer technology, have led to the emergence of new concepts within the advertising industry.

The current trend in online advertising is the Real-Time Bidding (RTB) mechanism of buying and selling advertisements [99]. RTB facilitates a real-time auction of advertising space, which allows buyers to decide how much to bid per impression. This gives advertisers fine-grained control over how they spend their advertising budget, as well as on which pages and to whom their advertisements are shown. These auctions take place at an ad exchange, which operates much like a stock exchange for ad impressions. The real-time nature of such auctions, in which bids must be placed in a fraction of a second, requires the whole auction process to be carried out programmatically.

To manage the added complexity of RTB, other types of platforms emerged along with ad exchanges. Such platforms include Demand-Side Platforms (DSPs), which provide advertisers with technology to bid on individual impressions from multiple inventories, and Supply-Side Platforms (SSPs), which support publishers in optimizing advertising yield. Moreover, the opportunity to target ads at individual users has led to the emergence of Data Management Platforms (DMPs), which provide user data to DSPs and SSPs.

The interplay between the various parties in the online advertising ecosystem is illustrated in Fig. 1.1. Prior to ad impressions, advertisers set up their campaign, including targeting criteria and campaign characteristics, with the DSP. When a single ad impression is triggered by a user visiting a web page that includes an ad slot, the user's web browser is instructed to contact the SSP to request an advertisement. The SSP forwards the ad request to the ad exchange, along with user data collected by the SSP. The ad exchange then sends a bid request to all connected bidders (DSPs), each of which may contact DMPs to retrieve additional information about the user. Each DSP decides how much to bid on behalf of its advertisers and responds to the ad exchange with the bid value and the associated advertisement. After collecting responses from bidders, the ad exchange chooses the winning bid and sends a win notice back to the winner. The ad exchange then sends the advertisement of the winning bidder to the SSP, which forwards it to the user. Finally, the advertisement is displayed in the user's browser. If the ad is clicked, the user's browser contacts the advertiser, and the advertiser records the ad click.

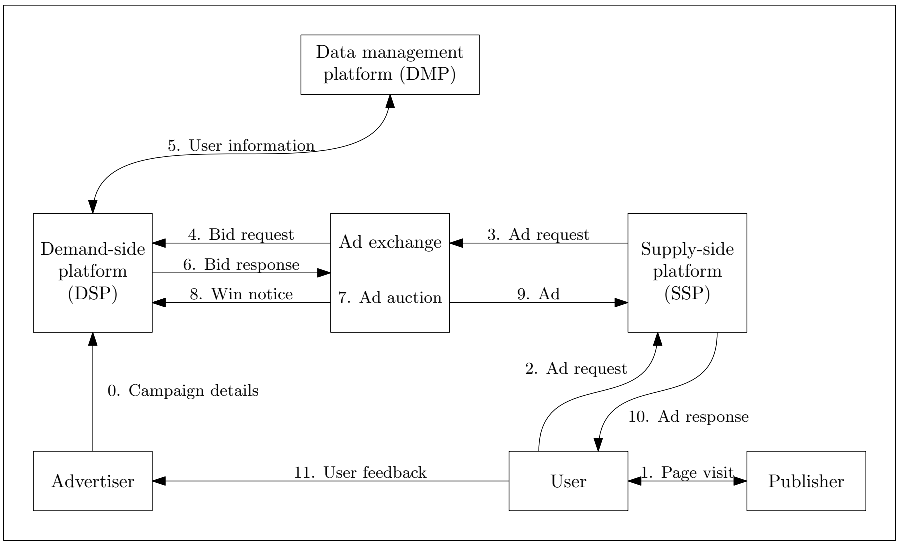

Figure 1.1: Overview of the online advertising ecosystem.

Each type of platform attempts to maximise the value of ad impressions. Supply-Side Platforms help publishers to manage their advertising inventory. They maximize advertising yield by offering inventory to multiple ad exchanges and direct sales contracts, and predicting which channel will maximise revenue for an impression. SSPs can further increase the value of a publisher's inventory by selling impressions together with user data. Since SSPs can track user behaviour across multiple publishers, they can augment impressions with behavioural data useful for targeting, driving the price of the ad impression up [70].

Demand-Side Platforms, on the other hand, help advertisers to target their audience more effectively. They use algorithms to predict user response to advertisements and optimise buying strategies that maximise the expected revenue for advertisers. DSPs also connect to multiple inventory sources, such as ad exchanges, to reach a larger audience. Finally, DSPs integrate with data management platforms and may also collect their own data, enabling precise targeting [70].

1.2 ONLINE BEHAVIOURAL ADVERTISING

The RTB ecosystem allows advertisers to target advertisements at individual users based on previously exhibited behaviour, such as visited webpages and clicked advertisements. This practice of showing the most relevant advertisement for a viewer based on their browsing behaviour is known as Online Behavioural Advertising (OBA). OBA has been shown to greatly improve click-through rates of advertisements, allowing for highly effective targeting of advertisements at specific consumer groups or even individual consumers [103]. Users see advertisements that are more relevant to their interests, and may help them discover a product or service they need. Advertisers, on the other hand, benefit from accurately targeted advertisements because they are more likely to reach the desired audience, and thus increase the expected return on investment of the advertising budget. Finally, publishers benefit from an increased value of the advertising space they offer, allowing for the continuation of the 'free content' business models that internet users have become accustomed to.

For OBA to work, information about both the webpage and the user associated with the auctioned ad impression must be passed along a number of different parties. Typically, the webpage served by the publisher includes a piece of code that causes the user's web browser to contact an SSP. The SSP then identifies the user, and sends information about the user to the ad exchange. The ad exchange, in turn, sends this user information, along with information about the webpage and publisher, to all DSPs participating in the auction. Each of these DSPs may have its own information about the user, and may contact one or more DMPs to obtain additional user information. The gathered information is then used to predict the response of the user to specific advertisements, which in turn is used to place a bid on the ad impression.

1.3 PRIVACY ISSUES

The nature of the data that is used for OBA, as well as the exchange of such data between a large number of parties, raises privacy concerns. Dozens of third party advertising companies track users across the web to keep track of the web pages visited by users, which products they purchase, users' location, which devices and software they use, and more [40]. Such information about user activity is widely shared between companies, often without users' knowledge or consent [14, 74]. While many advertising companies allow users to opt out of behavioural advertising, opt-out programs only limit a particular use of data, rather than limiting the collection of user data or other uses of tracking data [64].

Behavioural targeting primarily relies on a user's web browsing history to estimate the probability of a user clicking on an advertisement. Such browsing histories include visited web pages, but also search queries and previously clicked advertisements. The browsing history is rarely the only input for the estimation, however. Apart from characteristics of the current page, additional information about the user is gathered to improve the accuracy of the estimation. Such information commonly includes the user's location, (partial) IP address, and user agent, which describes the specific browser and operating system used. When possible, this may be further augmented with demographics, such as age and gender, and a history of online purchases.

There are a number of issues with widespread collection and sharing of user data. Firstly, the nature of the data that is collected is highly sensitive. Pages visited by a user can indicate location, interests, purchases, employment status, sexual orientation, financial challenges, medical conditions, and more [64]. Moreover, web browsing histories are analysed at large scale to find patterns of activity, which allows even more inferences about users' interests. While these inferences are made to serve advertisements tailored to users' interests, knowledge of people's behaviour and interests, possibly including political, financial, or medical information, leaves ample opportunity for abuse.

The possibility of abuse is further increased by the shape of the advertising landscape. User information is not securely stored at a single party, but is widely shared amongst parties. Advertising companies normally have access only to partial web browsing histories, as each company relies on the cooperation of publishers to allow tracking on their web pages. Using a technique called *cookie matching*, however, parts of these histories are leaked to other companies [74]. Cookie matching is a widespread practice that allows ad exchanges, DSPs and other parties to synchronize unique user identifiers and share data [14, 42, 74], and each of the parties involved in cookie matching is a potential attacker.

Users are given little insight and control over which data is collected about them and which parties have access to this data. Due to information leakage via cookie matching, and the exchange of personal data via DMPs, user data is spread to a large number of third parties. Many users are unfamiliar with some of these third parties, and would not voluntarily allow data collection by these parties [98]. While opt-out mechanisms are offered by the advertising industry, these mechanisms are difficult to use and mislead users into thinking that browsing data is not collected about them [61]. Furthermore, tracking companies adopt new techniques to track users who actively attempt to avoid online tracking, for instance by regularly clearing browser cookies. A variety of such persistent tracking mechanisms was found to be used in practice, indicating that even users who choose to take countermeasures against tracking are not necessarily protected from unwanted data collection [1].

Browsing histories not only contain sensitive personal information, they are also tightly linked to users' identities. Olejnik et al. [73] found that a vast majority of internet users had unique browsing histories, and that these unique histories could be used as fingerprints of users. With a small number of observed web page visits, users could be identified based on their browsing fingerprint. Moreover, even indirect observations about user history, such as interest categories inferred by some advertisers, were sufficient to uniquely identify a large number of users [73]. It is therefore possible to link distinct browsing sessions to the same user, implying that some countermeasures against tracking, such as private browsing modes offered by most web browsers, may not be effective at protecting users.

Possible attackers are not limited to companies within the advertising ecosystem; adversaries acting as advertisers or monitoring users can also abuse behavioural advertising to learn sensitive information about users. Castelluccia et al. [22] find that interest categories can be reconstructed from the advertisements that are shown to a user. An adversary who can observe which advertisements are shown during a short timespan could therefore infer a user's interests, even if those interests were not apparent during the time of observation. For instance, an employer could infer that an employee was looking for a new job outside office hours, based on the advertisements shown to the employee during office hours. Korolova [59] shows that precise targeting mechanisms and reporting options offered by Facebook can be abused by an advertiser to infer private attributes of user profiles. Their attack shows that information that should not be visible to advertisers can be leaked using precise targeting, undermining the trust people may have in Facebook to hide private information from third parties. Similarly, Conti et al. [30] demonstrate an attack that allows inference of user interests by combining data from advertiser and publisher systems offered by Google, showing a significant privacy issue in targeted advertising.

1.4 RESEARCH GOAL

The ad blocking arms race calls for a profound change to the use of personal information by the online advertising industry. In order to sustain the profitability of OBA and regain users' acceptance, the online advertising industry should address privacy issues. The goal of this research is therefore to alleviate the privacy concerns caused by the use of sensitive personal information for advertisement selection, while allowing behavioural targeting of advertisements. We aim to demonstrate the feasibility of behavioural targeting of advertisements while protecting user privacy to the extent that no party within the advertising ecosystem can gain any information about users' interests. The underlying research question of this thesis is as follows:

> *How can an online advertising system serve advertisements tailored to users' interests in the Real-Time Bidding model, such that no party other than the user themselves can gain any knowledge of the users' interests, without degrading the system's responsiveness perceived by users?*

This research question raises the following sub-questions:

1. How can the value of an advertisement for a user be estimated, based on the user's past behaviour, without any knowledge of that past behaviour or the user's interests?
2. How can information about a user's behaviour be made available for the purpose of advertisement value estimation, without disclosing the meaning of that information?
3. How can a model, used to estimate the value of an advertisement for a user, be trained with individual user responses without disclosing the user's response to advertisements?
4. How can a privacy-preserving value estimation model be integrated into the RTB model, such that the auction and billing processes reveals no information about users' interests?
5. How can the four sub-questions described above be accomplished without degrading the responsiveness of the system perceived by users?

1.5 OUR CONTRIBUTIONS

In this thesis, we present two complete protocols that preserve user privacy in OBA, are compatible with modern online advertising mechanisms, and support the use of highly detailed user profiles. The two protocols satisfy the same set of requirements, but one protocol uses additively homomorphic encryption to achieve user privacy, whereas the other uses secret sharing techniques. We believe the protocols presented in this thesis to be the first to make use of distributed trust in machine learning for preserving privacy in OBA tasks, as well as the first to allow learning users' preference patterns without relying on a trusted party. Both protocols are based upon machine learning techniques commonly encountered in existing OBA systems, and allow multiple data processors to operate on the same user data, while not revealing any information about users' behaviour or interests.

Based on proof-of-concept implementations of the presented protocols, we show the decrease in computational cost that can be obtained from cooperative computation by competing parties on secret-shared data, compared to an approach in which each party individually performs computations on encrypted data. Moreover, our implementations suggest that a cryptographic approach to preserving privacy in OBA can achieve the efficiency required for real-time predictions, as demanded by the RTB model of buying and selling advertisements.

1.6 THESIS OUTLINE

The structure of this thesis is as follows. The first part of this thesis provides a background on OBA and previous works on the topic of privacy in online advertising. Chapter 2 gives an overview of common OBA mechanisms that are used in the RTB setting, describing bidding strategies, user profiling methods, and response prediction models. In Chapter 3, various previous approaches to preserving privacy in online advertising are presented, along with a discussion on why these approaches are inadequate or incomplete. The second part of this thesis describes novel solutions to preserve privacy in OBA, starting in Chapter 4 with a description of the application setting, research challenges, and methodology. Subsequently, the two protocols are described. The first protocol, AHEad, is presented in Chapter 5, followed by the second protocol, BAdASS, in Chapter 6. Finally, we discuss the obtained results and provide an outlook for future research opportunities in Chapter 7

2 ONLINE BEHAVIOURAL ADVERTISING MECHANISMS

The Online Behavioural Advertising (OBA) landscape is formed by a complex combination of mechanisms that enable behavioural targeting of advertisements. In the RTB model, the process of targeting is largely left to bidding parties, such as DSPs. Bidders generally employ a number of different techniques to optimize their bids. To improve the understanding of users, tracking and profiling techniques are used to obtain a user profile. When an advertisement request arrives, triggered by a web page visit, the request is combined with an existing user profile to predict the response of the specific user to an advertisement. The predicted response is subsequently used in a bidding strategy to calculate the final bid price. If the bid wins the auction, the advertisement is served to the user, and the user's actual response (clicking on the advertisement or not) is reported back to the bidder, who uses the feedback to improve the response prediction process. The sequence of these techniques is illustrated in Fig. 2.1. In this chapter, each of the general mechanisms used in behavioural targeting is described, and some of the specific techniques that are commonly used in existing systems are outlined.

2.1 BIDDING STRATEGIES

Ad exchanges are the central components of the real-time bidding ecosystem, providing open marketplaces to buy and sell advertising space. They set the price for each ad impression in a real-time auction by offering the impression up for bids. Each buyer connected to the ad exchange places a bid based on their valuation of the impression, after which the ad exchange sells to the highest bidder. While some large advertisers connect to ad exchanges directly, most advertisers connect to a DSP, which bids on impressions on behalf of the advertisers.

As noted in [108], the optimal bid price is determined by the expected utility of the ad impression, and the expected cost. In practice, the utility of an impression is generally based on an estimation of a user's response to an advertisement, such as the Click-Through Rate (CTR) or Conversion Rate (CVR). The cost, on the other hand, is a prediction of how much an advertiser needs to pay to win the auction. Since ad exchanges generally hold a second-price auction, where the winner pays the second-highest bid, participants can benefit from adjusting their bid based on expected competing bids. Some recently proposed

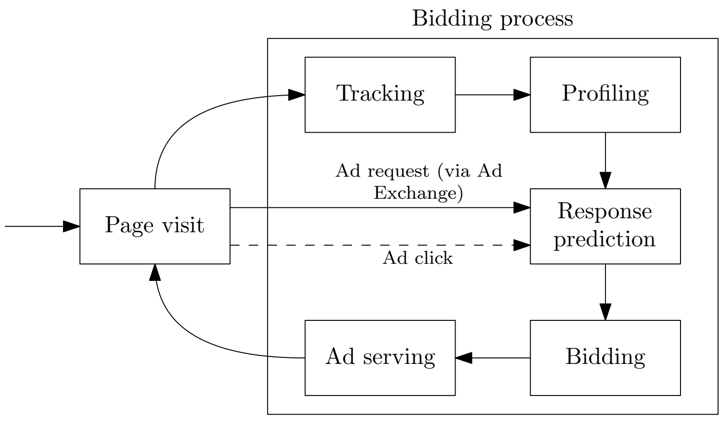

Figure 2.1: The components commonly used by DSPs to behaviourally target advertisements. Arrows indicate information flows.

methods to estimate the distribution of competing bids make use of the full bid request (e.g. [100, 102]). Many other methods to predict the bid landscape, however, rely only on historical observations of winning prices (e.g. [24, 108]).

To calculate a bid value, response predictions are turned into bids through a bidding strategy. A bidding strategy can be defined as a function $b(\theta(\boldsymbol{x}), \boldsymbol{x})$, where $\boldsymbol{x} \in \mathbb{R}^d$ is a d-dimensional bid request and $\theta(\boldsymbol{x})$ is the predicted user response. Note that throughout this document, bold symbols such as $\boldsymbol{x}$ are used to describe vectors. Bidding strategies may incorporate additional information to optimize the advertising campaign performance, such as the available budget and the value of a click to the advertiser. A simple, but very widely used bidding strategy bids the true value of an impression. Such truthful bidding maximises the expected reward for an advertiser [99].

In the presence of budget constraints and a dynamic market, however, bidders may choose to adopt non-truthful bidding strategies. Taking into account competing bids allows optimization of the bidding strategy based on the probability of winning the ad auction, such that the expected revenue is maximized given budget and campaign duration constraints. The probability of winning can be defined as a function $w(b(\theta(\boldsymbol{x}), \boldsymbol{x}), \boldsymbol{x})$. As described in [108], assuming a sequential dependency $\boldsymbol{x} \to \theta \to b$ greatly simplifies optimization of bidding. The bidding function b can then be assumed to only directly depend on $\theta(\boldsymbol{x})$, that is $b(\theta(\boldsymbol{x}), \boldsymbol{x}) \equiv b(\theta(\boldsymbol{x}))$, while still gaining the dependency of $\boldsymbol{x}$ through $\theta(\boldsymbol{x})$. Likewise, the winning function can be assumed to depend only on the bid value through $\boldsymbol{x} \to \theta \to b \to w$, that is: $w(b, \boldsymbol{x}) \equiv w(b)$. Analysis of data on winning bids in [108] suggests this is a sensible assumption, as the dependency on the bid request is far smaller than that on the bid price. With these assumptions, the bidding function can be optimized with respect to the response prediction θ, with no further dependency on the bid request $\boldsymbol{x}$.

2.2 USER PROFILING

In order to calculate a suitable bid price for an advertisement impression, a DSP must be in the possession of a profile of the user associated with the impression. Various techniques are combined in the creation of user profiles. This section describes methods used to track users' browsing behaviour across the web, followed by techniques to link user identifiers between different companies. Finally, techniques to model observed browsing events into usable feature vectors for machine learning tasks are described.

2.2.1 *Tracking techniques*

Web pages visited by users are one of the primary sources of information for behavioural targeting [84]. To track the web browsing behaviour of users across the web, companies within the advertising ecosystem use a variety of techniques. Most of these techniques require collaboration between the tracker and the publisher, typically in the form of the publisher including some content to be loaded from the tracker's servers on their web pages. By collaborating with many publishers, trackers can gain an increasingly complete view of web page visits across the web.

The most well-known tracking method uses HTTP cookies, which are used to persist small pieces of data across requests in a user's browser. Cookies are associated with a domain, and are sent along with every request to that domain. While cookies are useful to persist user preferences or authentication data, cookies are also used to record web browsing behaviour. A web page can contain content, such as images, that are retrieved from other domains. During retrieval of such third-party content, the external domains can also set cookies. These third-party cookies are sent back with every subsequent request to the external domain, along with an HTTP referrer header containing the URL of the first-party web page on which the content was included. Using a unique identifier as the cookie content, this mechanism allows trackers to link web page visits to the same user, and thus to track user behaviour. Typically, trackers instruct publishers to include on their pages a small, 1×1 pixel transparent image that is loaded from the tracker's domain, allowing the tracker to log web page visits on those pages [21].

Browser cookies can be easily removed by users, however. Therefore, some trackers have adopted more resilient tracking techniques to actively circumvent users' tracking preferences, such as *evercookies* or fingerprinting. Evercookies refer to combinations of different browser storage vectors such as Flash cookies, browser cache, and HTML local browser storage, which are used to recreate deleted HTTP cookies [1]. Browser fingerprinting refers to tracking techniques

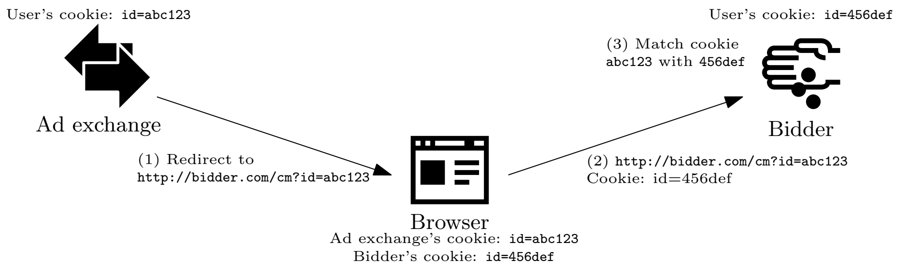

Figure 2.2: Cookie matching sequence, adapted from [74].

which do not rely on local browser storage to persist an identifier, but identify users based on unique combinations of browser and device characteristics [37]. These techniques typically rely on inclusion of JavaScript from the tracker, which reports back to the tracker's servers with a user identifier and the visited web page. Both evercookies and fingerprinting are used in practice, and combinations of tracking techniques are used to increase the resilience against countermeasures [1].

2.2.2 *Cookie matching*

Since each tracking party uses its own unique identifiers to track users, companies can significantly improve their knowledge of users by linking user identifiers and sharing data [74]. The practice of sharing user identifiers stored in cookies is known as *cookie matching* or *cookie syncing*. Cookie matching is widely used by ad exchanges and bidders to translate user identifiers [14, 42, 74]. The general cookie matching sequence is illustrated in Fig. 2.2. Typically, the ad exchange sends a script or redirection, which instructs the user's browser to send a request to the bidder. This request contains the cookie or user identifier used by the ad exchange, and since the request is made to the bidder's domain it also includes the bidder's cookie. The bidder thus receives both its own cookie and the cookie of the ad exchange, which allows linking the two identifiers to the same user [74].

2.2.3 *User modelling*

The collection of raw browsing events, such as web page visits and search queries, supplies advertising companies with detailed user information. Using this data directly in a machine learning task for response prediction is challenging, however, since the data has a very high dimensionality. Moreover, positive events in the form of ad clicks or conversions are very rare. It is therefore preferable to reduce the raw data to a lower dimensional user model. Behavioural modelling techniques can be divided into roughly three categories: interest-based segmentation, web page clustering, and using raw data.

The assumption underlying many of the proposed user modelling methods is that similar users will have a similar response to a given advertisement. A common practice in behavioural advertising is to assign users to predefined interest categories using a hierarchical categorization scheme [29, 96]. An example of such an interest category, as used by Google Ads, is *Finance → Credit & Lending → Loans*, where the arrows indicate an hierarchical relation [48]. User segmentation based on interest categories allows for simple audience selection by only showing advertisements to users within interest segments related to the advertisement's topic.

Interest categories can be extracted from web pages visited by a user in a number of ways. Raeder et al. [87] suggest the use of existing web page categorizations obtained from third party data providers. An example of a category data provider is the Open Directory Project (ODP), a community-maintained directory organizing URLs into a hierarchical ontology. Using such a directory, the high-dimensional feature space of URLs becomes a low-dimensional space of categories. RePriv [43], an attempt to implement client-side software to locally perform user modelling for content personalization, also uses interest categories from the ODP. Rather than relying directly on the manual classification of web pages in the ODP, RePriv trains a Naïve Bayes classifier on documents obtained from the ODP to locally classify documents. For each category in the taxonomy, 3000 documents are retrieved from the ODP. From these documents, words that occur in at least 15% of the documents are selected as attribute words. These attribute words are then used to train a Naïve Bayes classifier, which classifies new documents based on the words they contain.

Toubiana et al. [96] describe a similar category extraction method, which uses a social bookmarking site to map keywords extracted from documents into interest categories. From the social bookmarking site, the URL and title of bookmarked pages are collected, as well as tags assigned by users. A precomputed matrix of similarities between the user-assigned tags and interest category names is then used to obtain the categories for a web page, by computing the similarities between keywords obtained from the page and the tags associated with categories.

Using a fixed interest taxonomy for response prediction, however, may be too coarse due to the limited number of categories [80]. Moreover, assigning users to disjoint categories does not capture relations between interests, and it fails to describe complex combinations of multiple interests. Therefore, it may be desirable to assign mixtures of interests to users in an unsupervised fashion, instead of mapping users to hierarchical categories. Ahmed et al. [6] use Latent Dirichlet Allocation (LDA) to characterize users by a mixture of topics. LDA is a topic-based semantic modelling algorithm that describes the available data without requiring editorial data. The benefit of such unsupervised modelling is that it is able to cover topics that an editor is not aware of, and is thus not language and culture specific. LDA is a hierarchical Bayesian model, introduced in [18] for the purpose of modelling discrete data such as text. It models each item of a collection, such as a word in a search query or page title, as a mixture over an underlying set of topics. Each of these topics is, in turn, modelled as a mixture over an underlying set of topic probabilities. While the application of LDA to behavioural targeting still describes users with a subset of a fixed number of topics, it has several advantages over clustering. Firstly, modelling users as a mixture of topics enables capturing time-varying user interests. Secondly, topics themselves, being extracted from a dynamic corpus of search queries and web pages, can change over time. Since LDA is unsupervised, the model can adapt to changing topics without editorial intervention. Finally, all user actions are represented using a single vocabulary of words, providing a consistent representation of user interests.

Given sufficient data, however, behavioural targeting benefits from fine-grained user models [80]. Aly et al. [7, 8] therefore represent a user as a stream of historical behaviour. The user history includes both active events, such as clicking ads and issuing search queries, and passive events, such as viewing ads and visiting web pages. Visited web pages are clustered into several smaller subdirectories to reduce the dimensionality. Moreover, categories are extracted from web pages using an existing hierarchical page categorizer. These features are weighted by both frequency and recency: events that occur often are assumed to be more relevant, as are events that occurred recently.

Raeder et al. [87] also use clustering of visited web pages to reduce the profile dimensionality. However, they do not cluster web pages on the similarity of their content or co-visitation patterns of site visitors. Instead, URLs are clustered based on the likelihood of conversions in past advertising campaigns. The underlying assumption is that future campaigns that are similar to past campaigns will attract comparable user responses within the same URL clusters. To obtain a clustering flexible to newly observed URLs, the 15 000 domain names with the highest visitation rates were clustered using a hierarchical clustering algorithm. Using a subset of frequently observed URLs reduces the danger of variance due to a lack of data for parameter estimation. When a user visits a web page, instead of adding the URL of the visited page to the user model, an indicator that the user visited one or more of the domains in the associated cluster is added. Such a binary representation keeps the user model simple, while not resulting in decreased predictive performance.

Driven by the abundance of browsing data compared to a lack of sufficient user conversion data, Raeder et al. [86] model users as a collection of URLs. These URLs are hashed into a high-dimensional space, using the resulting value as the index in a binary feature vector. This practice, known as feature hashing or the hashing trick and originally introduced by Weinberger et al. [101], allows capturing arbitrary profile information $\boldsymbol{p}$ into a d-dimensional feature vector $\boldsymbol{x}$ through the use of hashing functions:

$$x_i = \phi_i^{(h,\xi)}(\boldsymbol{p}) = \sum_{j:h(j)=i} \xi(i) p_i, \tag{2.1}$$

where $h: \mathbb{N} \to \{1, \ldots, d\}$ and $\xi: \mathbb{N} \to \{\pm 1\}$ are hash functions, and ϕ is a hashed feature map. Such feature hashing enables the inclusion of any sort of information in the user profile, while limiting the size of the feature vector to a fixed number of dimensions. In practice, the hashing trick is sometimes simplified to result in a binary feature vector, where any nonzero element is set to 1.

Feature hashing, sometimes combined with binary feature vectors, is widely used in user response estimation [99], as described in work conducted at Yahoo [3], Criteo [25], Alibaba [27], Dstillery [86], and the winning contributions to two recent click-through rate prediction challenges [57]. Google, however, did not benefit from feature hashing [65]. Adding semantic information about web pages to the user model did not result in significantly improved predictive performance. These findings suggest that, given sufficient browsing history data, user models need not incorporate additional semantic and demographic information.

2.3 RESPONSE PREDICTION

When a DSP receives a bid request, the value of an impression must be estimated for each of the advertising campaigns run by the DSP. The value of an impression is based on a prediction of the user response, such as CTR or CVR. The predicted user response is subsequently used in the bidding strategy to determine the final bid price [99]. Since user response differs per campaign, each campaign is treated as a separate targeting task, such that a new set of one or more models is trained for each new campaign [5, 7, 31, 80].

The goal of user response prediction is, given a bid request and a user profile, to predict the probability of a user clicking the advertisement (or, in the case of CVR prediction, the probability of a conversion within a timespan after viewing the advertisement). Response prediction is a typical supervised machine learning problem. Data instances can be represented as a pair $(\boldsymbol{x}, y)$, where $\boldsymbol{x} \in \mathbb{R}^d$ is a d-dimensional input feature vector describing the bid request and user information, and y is the user response label. The label is usually binary, such as whether an ad click occurred or not. The input features are often encoded as a binary vector, either through feature hashing or by encoding each categorical field as a high-dimensional space, with every category of the field a separate dimension. As a result, the binary feature vector is typically extremely sparse. From a collection of historical observations, DSPs then estimate the probability of a click: $\hat{y} = P(y = 1 \mid \boldsymbol{x})$ [99].

2.3.1 *Logistic regression*

One straightforward solution to predict the CTR is to use logistic regression, which is used to estimate the probability of the binary outcome y based on predictor variables $\boldsymbol{x}$ [99]. The logistic regression model obtains from the binary output a continuous variable by taking the log odds ratio, or logit, of an observed outcome y given a predictor vector $\boldsymbol{x}$. The logit is fitted to $\boldsymbol{x}$ using a weight vector $\boldsymbol{w} \in \mathbb{R}^d$, such that the logistic regression model is a linear model of the log odds ratio [25]:

$$\log \frac{P(y = 1 \mid \boldsymbol{x}, \boldsymbol{w})}{P(y = 0 \mid \boldsymbol{x}, \boldsymbol{w})} = \boldsymbol{w}^{\top} \boldsymbol{x}. \tag{2.2}$$

To predict the outcome of a new bid request $\boldsymbol{x}$, the inverse of the logit, which is the sigmoid function, is used to estimate the probability of a click $\hat{y}$:

$$\hat{y} = \sigma(\boldsymbol{w}^{\top} \boldsymbol{x}) = \frac{1}{1 + e^{-\boldsymbol{w}^{\top} \boldsymbol{x}}}. \tag{2.3}$$

Important factors in the predictive performance of a response prediction system are the recency of the data used to train the model and the frequency with which the model is trained [54]. Therefore, advertising companies regularly train new models, use online learning algorithms, or combine online and batch learning [3, 25, 65]. McMahan et al. [65] use an online learning scheme based on a Stochastic Gradient Descent (SGD) algorithm. For every new instance $\boldsymbol{x}$, the click prediction $\hat{y}$ is calculated using Eq. (2.3). After some time, the actual label y is observed, either by the user clicking on the ad ($y = 1$), or by the absence of an ad click within a predefined timespan ($y = 0$). From this observed label, the logistic loss or negative log-likelihood is calculated as

$$l(\boldsymbol{w}) = -y \log \hat{y} - (1 - y) \log(1 - \hat{y}). \tag{2.4}$$

To train the logistic regression model, weight parameters $\boldsymbol{w}$ are searched that minimize the logistic loss on a set of training samples. In an online SGD algorithm, the gradient of the loss function, $\nabla l(\boldsymbol{w}) = (\hat{y} - y)\boldsymbol{x}$, is used to update the logistic regression model incrementally for each individual sample. The update rule is then defined as

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \eta \nabla l(\boldsymbol{w}), \tag{2.5}$$

where η is the learning rate. The learning rate can be a constant value, but may also be set to decay per iteration. He et al. [54] explore several schemes for updating the learning rate, and find that a per-coordinate learning rate achieves the best prediction accuracy. Such a per-coordinate learning rate scheme, proposed in [65], updates the learning rate for each feature separately, such that the learning rate for feature i at iteration t is set to

$$\eta_{t,i} = \frac{\alpha}{\beta + \sqrt{\sum_{j=1}^{t} g_{j,i}^2}}, \tag{2.6}$$

where $g_{j,i}$ is the i^{th} coordinate of the gradient $\boldsymbol{g}_j = \nabla l_j(\boldsymbol{w}_j)$ at iteration j, and α and β are two tunable parameters. The advantage of a per-coordinate learning rate over a global learning rate is that the learning rate decreases faster for features that occur often, whereas features for which little data is available decrease slowly [65]. Note that while the feature vector $\boldsymbol{x}$ might have billions of dimensions, it is generally very sparse, containing only a small number of non-zero entries [65]. Both the calculation of $\hat{y}$ and the update of $\boldsymbol{w}$ involve only the non-zero entries and are therefore very fast [99].

The sparse, high-dimensional input vectors encountered in OBA can also lead to overfitting of the model. Therefore, a regularization term on the weight vector is often added to the loss function [99]. Typically, L_2 regularization is used to prevent overfitting, resulting in the optimization problem

$$\boldsymbol{w} = \arg\min_{\boldsymbol{w}} \sum_{t=1}^{n} \left(l_t(\boldsymbol{w}) + \frac{\lambda_2}{2} \|\boldsymbol{w}\|_2^2 \right), \tag{2.7}$$

where n is the number of training samples, $l_t(\boldsymbol{w})$ is the loss for sample t, such as the logistic loss from Eq. (2.4), and λ_2 is the regularization parameter. Additionally, L_1 regularization may be used to reduce the number of weight parameters in the model, resulting in a sparse model and thus reduced memory use [25]. For a more thorough description of L_1 and L_2 regularization for logistic regression and their properties, see e. g. [69].

Logistic regression is the basis for many response prediction systems used in practice, e. g. at Criteo [25], Facebook [54], Google [65], and Dstillery [84]. These systems use logistic regression because it is easy to implement, scales well to large numbers of features as well as large numbers of samples, and can be trained in an incremental fashion [25, 65].

Although linear models such as logistic regression are easy to implement with high efficiency, only shallow patterns for individual features can be learned. In OBA, however, interactions between different features might have a significant predictive quality. Chapelle et al. [25] therefore introduce conjunction features, which are the

Cartesian products of individual features. These conjunctions capture interactions between any two features with a logistic regression model.

2.3.2 *Factorization machines*

Another approach to capturing feature interactions, which does not involve a quadratic growth in the number of features, is to use a non-linear model. The Factorization Machine (FM) model, proposed in [89], considers feature interactions by mapping them into a low-dimensional space. FMs learn a latent vector for each feature. Each latent vector consists of k factors, where k is a user-specified parameter. The effect of feature interactions is then modelled by the inner product of two latent vectors:

$$\hat{y} = \sigma(w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \langle \boldsymbol{v}_i, \boldsymbol{v}_j \rangle x_i x_j), \tag{2.8}$$

where σ is an activation function, such as the sigmoid function, w_0 is a global bias term, w_i is a bias factor for feature i, $\boldsymbol{v}_i$ is the latent vector for feature i, and $\langle \cdot, \cdot \rangle$ is the inner product of two vectors.

If the available data is very sparse, FMs may estimate interactions better than feature conjunctions in a linear model [89]. Because interaction parameters are factorized, the data for one interaction helps estimate the parameters for related interactions. Therefore, meaningful predictions are possible even for feature pairs not observed during training. Moreover, the computational complexity of prediction for FMs is $O(\bar{n}k)$, where $\bar{n}$ is the average number of non-zero features per instance, whereas that of logistic regression with feature conjunctions is $O(\bar{n}^2)$ [57].

FMs and variants thereof have recently received attention in CTR prediction problems. Oentaryo et al. [72] extended FMs with importance weights and hierarchical learning to improve predictive performance in cold-start scenarios. In [94], an online learning method for FMs is introduced by combining FMs with adaptive learning rate and regularization techniques from McMahan et al. [65]. The resulting method showed an improved performance over standard FM trained with SGD. Pan et al. [79] modify FMs to address the problem of highly sparse data for CTR prediction.

In [57], Juan et al. introduce a variant of FMs called Field-aware Factorization Machines (FFMs) as an effective approach for CTR prediction. Since input features for CTR prediction are typically encoded as binary vectors, different values for the same categorical variable are assigned distinct features. The FFM model groups these distinct features into 'fields', such that different features describing the same variable, such as which publisher will display an advertisement, are grouped together.

In the regular FM model, every feature has one latent vector, which is used to learn the interactions with all other features. However, a feature may have a different effect on features from different fields. Therefore, the FFM model keeps several latent vectors for each feature, such that the interaction of a feature with every field can be captured. Feature interactions are then modelled by the inner product of the latent vectors of a feature that are learned for the field of another feature:

$$\hat{y} = \sigma(w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \langle \boldsymbol{v}_{i,f_j}, \boldsymbol{v}_{j,f_i} \rangle x_i x_j), \tag{2.9}$$

where f_i and f_j are the fields of features i and j, respectively. The complexity of prediction for FFMs is $O(\bar{n}^2 k)$.

FFMs have been used to win two recent CTR prediction challenges, and experiments in a production system have shown that response prediction using FFM can significantly improve business metrics compared to logistic regression [56].

2.3.3 *Transfer learning*

One of the major challenges in CTR prediction, and particularly in CVR prediction, is that positive labels, i. e. clicks on ads or conversions, are extremely rare [84]. Web browsing data, on the other hand, is relatively abundant. Transfer learning deals with such problems by transferring knowledge from a *source* task with ample training data to a *target* task where learning data is expensive to get [106].

Dalessandro et al. [31] transfer knowledge from a source model trained on browsing events to a CVR prediction target task. The source model, a logistic regression model trained on browsing behaviour on an advertiser's website, is expected to identify users who are similar to the clients of an advertiser. The resulting model parameters $\boldsymbol{\mu}$ are then used as a prior to the regularization term of the target task. With an L_2 regularization term as in Eq. (2.7), the optimization problem becomes

$$\hat{\boldsymbol{w}} = \arg\min_{\boldsymbol{w}} \sum_{t=1}^{n} \left(l_t(\boldsymbol{w}) + \frac{\lambda_2}{2} \|\boldsymbol{w} - \boldsymbol{\mu}\|_2^2 \right). \tag{2.10}$$

Perlich et al. [84] evaluate transfer learning in a production CVR prediction system, where multiple models from different source tasks are combined into a new low-dimensional feature space. However, the reported production results only include a comparison with a randomly targeted control group.

Zhang et al. [106] use transfer learning to implicitly and jointly learn user profiles on both web browsing behaviour and ad response behaviour. The two-stage mechanism of user segmentation, as described in Section 2.2.3, followed by response prediction, is combined into a single model. The model is the combination of tasks: web browsing prediction, described as a collaborative filtering task, and response prediction, described as a CTR task. Both tasks use FMs as prediction models. Knowledge from the collaborative filtering task is then transferred to the CTR task by using the weights and latent vectors learned in the collaborative filtering task as priors for their counterparts in the CTR task.

2.3.4 *Deep learning*

Recently, some attempts have been made to utilize Deep Neural Networks (DNNs) in response prediction for display advertising. Chen et al. [27] use a DNN model to predict the CTR of image advertisements, based on the contents of the images. Zhang et al. [107] adopt DNNs to automatically detect interactions between categorical features. A supervised-learning embedding layer using FMs is used to reduce the high-dimensional, sparsely populated input feature space to a low-dimensional, dense feature space for the neural network. While the industry claims the use of DNNs for CTR prediction [109], very little details on models or implementation are available.

3 PRIOR ART

The privacy risks posed by tracking and profiling practices in online advertising have led to the development of a variety of tools to protect the privacy of users [40]. The advertising industry itself offers some tools that allow users to opt out of behavioural advertising. However, these opt-out programs merely limit the use of data, and do not limit the collection of user data nor other uses of tracking data. Users of opt-out tools frequently believe that they are opting out of tracking, making these tools somewhat misleading [61]. The Do Not Track (DNT) header is another industry attempt to limit tracking, using an HTTP header sent by a user's web browser to indicate that the user wishes not to be tracked. Similar to opt-out mechanisms, the DNT header requires cooperation from the parties performing online tracking. DNT headers have repeatedly been found to be ineffective at limiting behavioural targeting of advertisements [2, 13].

Other tools designed to protect user privacy in online advertising adopt various approaches. Some tools aim to be compatible with the current economic model of ad-supported websites, whereas others offer methods to block online advertisements altogether. This chapter gives an overview of existing tools aimed at privacy protection in online advertising, describes the underlying techniques, and evaluates the potential utility the tools offer advertising companies in the current OBA landscape.

3.1 TRACKER BLOCKING

One widely used approach to protect privacy in online advertising is to use client-side tools that block requests to tracking or advertising parties [67]. Such blocking tools operate on the user's machine, without requiring any cooperation from tracking parties or website operators. Some blocking tools aim to prevent advertisements from being shown altogether, whereas others focus primarily on blocking trackers.

Blocking tools employ a variety of techniques to block requests to trackers. Many popular ad-blocking tools are implemented as extensions to web browsers, whereas others operate at the network level. Ad-blocking extensions, such as the popular *Adblock Plus*[1], have fine-grained access to network requests. A number of different methods

1 Adblock Plus (https://adblockplus.org) claims to be the world's most popular browsing extension, used on 100 million devices. A list of other common ad blocking tools is given in [67].

can be used to determine which network requests are to be blocked. Some extensions use community-maintained blacklists of Uniform Resource Identifier (URI) patterns to block, whereas others use a centralized approach, where the blacklist is maintained by the developer of the browser extension [67]. Blacklists, however, are difficult to maintain and difficult to use. Therefore, machine learning seems a promising direction for tracking countermeasures [15]. One browser extension employing heuristics to algorithmically detect trackers is *Privacy Badger*[2], supported by the Electronic Frontier Foundation. Privacy Badger detects tracking parties by observing requests to third-party websites, and searching for the same high-entropy strings in multiple requests [67]. Similarly, Acar et al. [2] use heuristics to detect web browser fingerprinting by intercepting accesses to browser properties that could be used for fingerprinting.

Network-based blocking techniques, on the other hand, work independent of the underlying application or web browser [67]. These techniques block entire domain names using DNS filtering, or use interception proxies to filter on URI patterns. While network-based blocking tools have seen an increased popularity for blocking on mobile devices (e.g. [66]), these tools lack the fine-grained access to network requests that browser extensions have [67].

While tracker blocking gives users some control over which trackers are contacted by their browser and which advertisements they see, tracker blockers generally fail to completely block tracking. Moreover, blocking network requests breaks some websites, particularly when using heuristic detection techniques [67]. Finally, ad blockers and tracker blockers severely limit the advertising revenue for publishers, thus endangering the business model of many websites [40].

3.2 OBFUSCATION AND ANONYMIZATION

Since many potential tracking parties provide useful functionality to websites, such as social media widgets or Content Delivery Networks (CDNs) hosting website content, blocking all trackers often reduces the functionality of a website [34]. An approach to hiding user interests from trackers without blocking content is to add noise to the collected data, a technique known as obfuscation [20, 34, 60]. Obfuscation techniques deliberately produce misleading, false, or ambiguous data [20]. Obfuscation has been used to make the process of tracking a user more difficult, but also to make identifying a user based on collected data more difficult. Two different but related obfuscation techniques are used in the works described in this section. The first approach replaces original data with pseudo-random data, whereas the second approach inserts pseudo-random traffic into a stream of original traffic.

2 https://www.eff.org/privacybadger

Nikiforakis et al. [71] obfuscate web browser attributes to protect against tracking based on web browser fingerprints. A key insight in their work is that not uniqueness of a web browser fingerprint, but linkability across multiple visits enables fingerprint-based tracking. Therefore, browser fingerprints are made non-deterministic, which makes them hard to link across browser sessions. A similar approach is taken by Torres et al. [95], who spoof attributes such that their combination does not make the user stand out due to inconsistencies between the attributes.

Degeling and Herrmann [34] analyse user interest profiles generated by Google, and use this information to obfuscate profiles with dummy traffic. After extracting the most likely user profile from a browsing history, an *anti-profile* is created based on the interests that are least likely to be part of the real user profile. The anti-profile is used as a basis for the generation of dummy traffic, such that the generated traffic does not reinforce interests already present in the real user profile. Although the insertion of dummy traffic resulted in measurable changes to user interests as reported by Google, the effect of obfuscation was overshadowed by a high variance of user profiles over time. Moreover, the creation of artificial user profiles did not resemble human browsing patterns, and may thus have been detected as obfuscation attempts.

While randomization of web browser characteristics or user interests is likely to have a somewhat higher utility to publishers than completely blocking advertisements, such obfuscation techniques do not allow any behavioural targeting. Papaodyssefs et al. [81] therefore use k-anonymity to ensure that browsing histories are not uniquely identifiable, yet contain useful behavioural information. Proposed by Sweeney [93], the k-anonymity model ensures that a user is indistinguishable from at least $k-1$ other individuals. Such k-anonymity is achieved by a combination of suppression of data points that make a user too dissimilar from other users, and insertion of pseudo-random data points that increase the similarity between users.

To perform the anonymization, parts of browsing histories are suppressed by a proxy that maps public cookies, as set by trackers, to private cookies exposed to a user's web browser. Browsing events generated by a user are only passed through to the tracker if they do not make the browsing signature of the user distinguishable from browsing signatures of at least $k-1$ other users. If a browsing event threatens to make a user's browsing history too dissimilar from other browsing histories, one of two intervention policies is applied: either the tracking cookie is dropped, or the user's browsing history is obfuscated with web pages that increase the similarity with other browsing histories.

By ensuring that at least k users expose similar browsing histories to trackers, an attempt is made to prevent re-identification of users based on their browsing behaviour. The proposed architecture, however, relies on a proxy that has access to the browsing histories of all users. The privacy problem is thus shifted from one tracking party to another. Moreover, the used similarity metric compares domains rather than individual web pages, which may make the proposed approach ineffective against trackers who extract information from visited web pages.

The obfuscation-based approaches described here have been shown to have some effect on user profiles in practice, but neither the effect on user privacy nor the impact on advertising utility is thoroughly analysed. Obfuscation attempts in similar domains have been shown to be unsuccessful at protecting user privacy, e. g. in web search [82], location-based services [83], or DNS queries [55]. Furthermore, feedback from users with obfuscated profiles, in the form of e. g. clicks on advertisers, may lead to a decrease in targeting performance for all other users if user profiles contain false interests.

3.3 LOCAL PROFILING

Many of the proposed solutions to enhance the privacy of OBA store user profiles locally on the user's own machine. Local profiling techniques construct a user profile, typically consisting of a set of interest keywords, from a user's observed web browsing behaviour. These profiles are then made available to advertising companies in a privacy-preserving way. Browsing behaviour is thus generalized to a set of interest keywords before being shared with third parties. Generalization reduces the granularity of the represented data to a range of possible values, and is commonly used in privacy-preserving techniques such as *k*-anonymity [4]. Advertising companies thus no longer have access to detailed web browsing histories, which reduces the amount of information leaking to third parties.

Bilenko and Richardson [17] propose a method for storing user profiles within the user's web browser, while profile updates are performed on a server. The profile contents are based on search keywords and clicks on advertisements. While storing profile content on the user's machine gives the user insight into the data used for advertising, as well as some level of control over that data, the proposed method does not prevent the server operator from building a user profile since all profile updates are performed by the server.

Fredrikson and Livshits [43] present RePriv, a framework for locally discovering user interests and sharing them with third parties. Interest topics are extracted from visited web pages into a taxonomy maintained by the ODP, and are assigned a level of interest. To ensure that the user population is not distributed too sparsely among interest topics, the taxonomy is limited in depth. Given explicit user permission, the categories that comprise most of the browser history, as well as the interest levels associated with these categories, are shared with third parties via an HTTP header.

Locally constructed user profiles are used in a variety of ways in work on privacy-preserving online advertising. In Adnostic [96], an architecture for privacy-preserving targeted advertising, both profiling and targeting are performed within the user's web browser. When a user visits a web page containing an ad slot, an advertising network is contacted by the user's web browser. The advertising network sends back a list of multiple advertisements based on the page content. The Adnostic web browser extension then selects the most relevant advertisement from the downloaded list by comparing topic tags of the advertisement with the locally constructed user profile. A major drawback of Adnostic is that it does not hide the user's web browsing behaviour from the advertising network. Furthermore, only a small set of advertisements is downloaded for the web browser extension to make a selection from, and this set is selected without any behavioural targeting.

Privad [51, 52] combines locally computed user profiles with an honest-but-curious third-party anonymizing proxy, called the dealer, between the user and the advertising network, called the broker. The client software of Privad requests advertisements from the broker by anonymously subscribing to broad interest keywords and broad non-sensitive demographics. The broker sends a set of advertisements matching the interest keywords and demographics, which are locally filtered and cached by the client software. If a user has multiple interest categories, these are separately reported to the broker in a manner that prevents linking interests to the same user. Communication between the client and the broker is anonymized by the dealer, whereas the contents of the communication are hidden from the dealer by means of encryption with either the broker's public key or an agreed upon session key. Since advertisements are selected from the local cache, and thus no third parties are contacted to retrieve advertisements on individual web page visits, advertising companies cannot track web browsing behaviour. However, realising a third party proxy that does not collude with the broker, yet satisfies the broker's and advertiser's demands, may be a difficult problem to solve [12]. Moreover, such a party may introduce a single point of failure in the advertising system.

Proposed by Backes et al. [12], ObliviAd uses a Secure Coprocessor (SC) to perform behavioural advertising while ensuring that user data remains private. ObliviAd assumes the presence of a locally computed user profile, consisting of a set of interest keywords. Advertisements are selected by matching advertisement keywords to profile keywords on an SC. The contents of user profiles and selected advertisements are hidden from the broker by the client directly communicating with the SC over a secure channel. A more detailed description of the advertisement selection process in ObliviAd is given in Section 3.4. Contrary to Privad, ObliviAd allows the broker to learn the identity of the user, but hides any information about user interests. An anonymizing proxy or network is thus avoided.

Androulaki and Bellovin [10] propose an advertising system where a user manually selects product categories and specific products that they are interested in to create partial user profiles. A user is thus not associated with a single profile describing all their interests, but with a number of profiles each describing a subset thereof. When an advertisement is requested, one of the partial profiles is shared with the advertising network via an anonymizing network and a proxy responsible for fraud detection and billing.

While local user profiling can be used to alleviate many of the privacy concerns in OBA, its adoption in existing solutions is subject to some pitfalls. All solutions described here assume an online advertising model where a central party, called broker or ad network, is responsible for selecting which advertisement to show to a user. This selection is based on a combination of interest keywords and targeting keywords submitted by advertisers. In the current RTB model, however, advertisement selection has shifted towards bidders such as DSPs, each of whom uses profile contents to optimize their bids. Response prediction methods used by bidders are far more complex than trivial keyword matching as used in the described solutions. Not only do many response prediction models rely on information far more detailed than generalized interest categories, they also combine the data of millions of impressions to learn patterns in user response. Generalization through local user profiling is thus expected to result in worse personalization performance than current advertisement selection algorithms [40].

Finally, even generic interest keywords are sensitive information. Generalized interest segments alone may disclose sensitive information, such as getting pregnant, repairing bad credit, and debt relief [64]. Moreover, Olejnik et al. [73] show that categories of websites can be used to uniquely identify a large amount of web users. Given a unique interest fingerprint, a single leakage of Personally Identifiable Information (PII) is sufficient to link previous and future observations to a user identity.

3.4 CRYPTOGRAPHIC APPROACHES

One of the earliest works on privacy-preserving targeted advertising, by Juels [58], proposed the use of cryptographic constructions to protect user privacy while allowing behaviourally targeted advertising. Green et al. [49] classify cryptographically anonymized advertising systems into three categories: systems based on mix networks, voting systems, and hardware-based systems. Mix networks, introduced by

Chaum [26], are networks of servers through which multiple messages are anonymized by randomly permuting and re-encrypting them. Hardware-based approaches use specialized Secure Coprocessors (SCs) to hide user data from advertising networks. Voting systems are used to privately report advertisement impressions or clicks to the advertiser. They are typically based on an additively homomorphic encryption scheme, such as ElGamal [38, 96] or Paillier [49, 78]. Such schemes allow anyone with access to a public key pk and ciphertexts $\mathcal{E}_{pk}(m_1)$ and $\mathcal{E}_{pk}(m_2)$ to create the ciphertext $\mathcal{E}_{pk}(m_1 + m_2)$, without having knowledge of m_1 and m_2 or the corresponding secret key. A central party can thus sum individual encrypted impression reports to obtain a single aggregate ciphertext [49].

Juels [58] uses a Private Information Retrieval (PIR) scheme, which enables a user to request an advertisement from a server without the server learning any information about the user's request. The scheme is built upon the assumption that a set of advertisement servers is present. To ensure privacy and output correctness, the number of honest servers must be above a threshold. When retrieving an advertisement, each user locally computes which advertisement to request, based on an advertiser-supplied selection function and local profile information. The identifier of the selected advertisement is encrypted and sent to the servers along with a user identifier, where the request is anonymized by a mix network. Simultaneously, the encrypted identifier is replaced by an encrypted advertisement, which is re-encrypted with the user's public key using proxy re-encryption, and sent back to the user.

The threshold PIR scheme used by Juels, however, is not suitable for real-time ad serving due to the time needed for the mixing process [12, 58]. Therefore, Backes et al. [12] propose ObliviAd, a more efficient PIR approach based on secure hardware. The advertising model upon which ObliviAd is based is that of a single untrusted broker acting as a matchmaker between multiple advertisers and publishers. To request an advertisement using ObliviAd, a user sends a locally generated profile, consisting of interest keywords, to the broker over a secure channel. The broker then retrieves advertisements associated with profile keywords from an advertisement storage, modelled as a dictionary of keyword and advertisement pairs.

To secure the advertisement selection process, ObliviAd uses a Secure Coprocessor (SC), combined with an Oblivious RAM (ORAM) scheme. The ORAM model protects the access pattern of software on memory [46], and is used in ObliviAd to hide the access pattern of the SC on the storage containing advertisements. The SC ensures that the broker cannot learn user information from the internal processor state, and additionally allows anyone to remotely verify that the broker is running the correct program. The ORAM, on the other hand, ensures that the broker cannot learn which advertisement is retrieved for a user. To privately report ad impressions or clicks, ObliviAd uses electronic tokens that are sent to the user along with an advertisement. These tokens contain encrypted information about the shown advertisement, as well as the time at which they were generated, and are digitally signed by the SC. Upon viewing or clicking the advertisement, the user sends the token back to the SC, where it is decrypted and mixed with tokens from other users before being published. By mixing tokens on the SC, the broker cannot determine which user viewed which advertisement.

Although the use of secure hardware to protect privacy seems like a promising solution, ObliviAd comes with some limitations. As noted by Green et al., hardware-based solutions seem infeasible without increasing the cost beyond the point of profitability [49]. Moreover, the advertising model upon which ObliviAd is based does not correspond well to the complexity of the current RTB model. Not only are many different parties involved in advertisement selection, the models used to select appropriate advertisements are also more complex than straightforward keyword matching. Finally, advertising companies may not be willing to disclose which algorithms they use for ad selection due to competition among advertising platforms [41].

Adnostic [96], of which the local profiling component is described in Section 3.3, uses a voting system based on homomorphic encryption to privately report which advertisement was viewed by a user. Adnostic creates, for each downloaded advertisement, an encryption of either 0 or 1, to indicate if the advertisement was viewed. A vector of the encrypted values is sent to the server, along with zero-knowledge proofs that each value is an encryption of either 0 or 1, and that the sum of the values is exactly 1. Upon receiving a vector of ciphertexts, the server multiplies each encrypted value with the cost of an impression, and adds the encrypted values to encrypted counters for each advertisement. Finally, the encrypted counters are decrypted by a Trusted Third Party (TTP).

Green et al. [49] propose an alternative to the voting scheme used in Adnostic. The proposed protocol uses zero-knowledge proofs that are more efficient, in terms of bandwidth, than the proofs in Adnostic. The improved efficiency is achieved by using proofs of set membership, proposed by Groth and Kohlweiss [50], as well as reducing the sizes of encryptions. Moreover, a computational PIR scheme is used to retrieve advertisements. Similar to Adnostic, however, the authors suggest downloading advertisements prior to impressions, making the scheme unsuitable for real-time targeting.

3.5 PRIVACY OF SELECTED ADVERTISEMENTS

While much of the research on privacy in online advertising focuses on sensitive user information being exposed to companies in the advertising ecosystem, behaviourally targeted advertisements can reveal user interests to other parties as well. Castelluccia et al. [22] show that an adversary can infer users' interests by observing only a small number of targeted advertisements. Their attack uses Google Ads Preferences, a tool that gives users insight into their interest categories as inferred by Google, to categorize individual advertisements. Countermeasures proposed by the authors include ad-blocking software, locally selecting advertisements as is done in e. g. Adnostic [96], or sending advertisement requests and responses over a secure channel to prevent eavesdropping.

A different attack on privacy in targeted advertising is demonstrated by Korolova [59], who exploits highly detailed targeting capabilities offered by Facebook to infer private information about individuals. Their attack is performed by creating advertisements on Facebook, with targeting criteria set such that only a single user matches the criteria. The targeting criterion for a private attribute that the attacker wants to learn, such as a person's sexual orientation, is set to a different value for each of the created advertisements. When a view is reported for one of the advertisements, the attacker learns the value of the private attribute.

Lindell and Omri [63] propose countermeasures to attacks abusing highly specific targeting, such as the attack demonstrated by Korolova. Their proposed solution uses a differentially private mechanism for releasing advertising campaign statistics. Differential privacy, suggested by Dwork et al. [36], requires that a change of a single database entry changes the distribution of responses given by the database only slightly [63]. This is achieved by adding randomized noise to the output of queries for campaign statistics, such as impressions, clicks, unique impressions, and unique clicks. The added noise ensures that an attacker cannot determine with high probability if an advertisement was viewed or clicked by any specific user, and thus prevents the attack described by Korolova.

Korolova [59] notes, however, that advertisers might be hesitant to accept being charged based on noisy campaign statistics. Indeed, differential privacy requires a trade-off between privacy and utility. Lindell and Omri [63] deem an error of over a hundred clicks to be reasonable, which may decrease the cost-effectiveness of small, highly targeted campaigns in particular. Moreover, Korolova argues that, since symmetrically distributed random noise is typically used to achieve differential privacy, an attacker can make accurate inferences by averaging a sufficiently large number of campaigns [59]. Finally, advertisers who use click feedback to optimize their campaigns, as is common in CTR prediction tasks as described in Section 2.3, may suffer from a decreased predictive performance due to noisy campaign statistics.

4 RESEARCH CHALLENGES AND METHODOLOGY

The ad blocking arms race makes it evident that, in order to maintain the ad-supported web as we know it today, the privacy concerns of users must be addressed while maintaining the profitability of OBA. While several approaches to enhance the privacy of OBA have been suggested, none of the existing solutions can achieve user privacy in the current RTB model without harming the economic benefits of OBA. Previous work limits the available targeting information, underestimates the complexity of the advertising landscape and ad selection models, or precludes learning preference patterns from individual responses.

The goal of this thesis is to alleviate some of the privacy concerns associated with OBA and the RTB model. In this chapter, the concrete setting within which this thesis is placed is described, as are assumptions about the behaviour of the different parties involved in online advertising. Afterwards, challenges that need to be overcome to ensure user privacy are outlined. Finally, the methodology used to achieve the objective of this research is discussed.

4.1 APPLICATION SETTING

The online advertising landscape is a complex, heterogeneous ecosystem consisting of a wide variety of parties, some of which offer highly specialized niche services whereas others attempt to operate throughout the whole advertising process[1]. To obtain a well-defined application setting, we simplify the complex advertising landscape to the set of roles described in Section 1.1, as is common in academic discourse related to RTB. A schematic overview of the setting is given in Fig. 4.1.

We assume that no party has a monopoly position, i. e. all roles are fulfilled by at least two different parties. As a simplification, advertisers always use exactly one DSP to manage their campaigns, and thus do not connect to ad exchanges directly. Moreover, we assume that DSPs are the only parties that perform operations on user data, such that personalization of advertisements is performed entirely by DSPs, whereas ad exchanges only request bids from DSPs and select the highest bid. These assumptions allow us to disregard SSPs and DMPs as data processors, and therefore not consider either as active

1 The *LUMAscape*, which is the industry-standard categorization of companies operating in the online advertising landscape, attempts to give a clear overview of the sector. See http://www.lumapartners.com/lumascapes/display-ad-tech-lumascape/

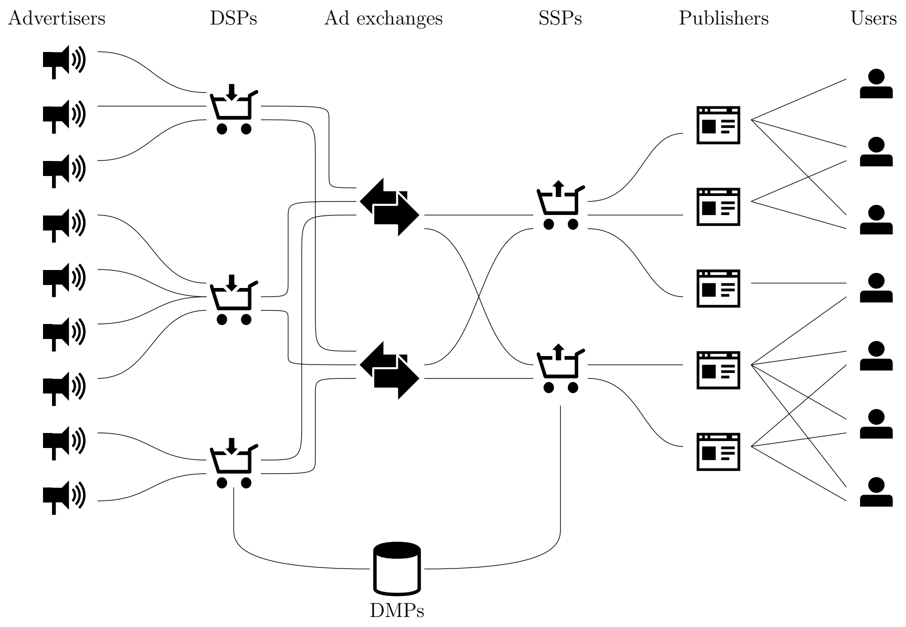

Figure 4.1: Schematic overview of the application setting. The depicted number of parties fulfilling any role and the connections between parties in different roles are for illustrative purposes only; the application setting is not limited to this exact configuration.

players in our application. Instead, we assume that SSPs and DMPs act as a mere conduit for encrypted data with no access to keying material, and ignore either party in the description of our protocol design for simplification purposes. However, an SSP may still be present between the user and ad exchanges for the purpose of offering advertising space to multiple ad exchanges.

We base our protocols on a semi-honest security model, where parties may attempt to gather sensitive information from the protocol execution, but do not deviate from the protocol. Since some existing companies act as both ad exchange and DSP, we must assume that ad exchanges and DSPs may collude to obtain information.

We assume that all DSPs use a logistic regression model to perform response prediction, given the widespread acceptance of logistic regression as a strong baseline model in industry and academia alike, as well as its computational simplicity. The logistic regression models are trained using unregularized online SGD as described in Section 2.3.1 and used by McMahan et al. [65]. We further assume that DSPs use linear bidding functions of the form $B_k(\hat{y}) = c_{k,1}\hat{y} + c_{k,2}$ for campaign-specific constants $c_{k,1}$ and $c_{k,2}$ for a campaign k. Although a linear bidding function may not be capable of fully capturing complex bidding strategies used in practice, it is used for illustrative purposes, and could be replaced by a more complex bidding function. Additionally, we assume that the learning rate is a campaign-specific constant shared across all coordinates, such that a single learning rate is used for all model parameters of a campaign.

4.2 RESEARCH CHALLENGES

To alleviate privacy concerns within the current RTB model, three main challenges need to be overcome. Firstly, machine learning models employed by DSPs in the advertisement selection process must be adapted to a setting in which they do not have access to raw user data. Secondly, the advertisement selection models must be integrated into the RTB model without disclosing user interests to any party other than the user. Finally, the applied privacy protections must have adequate efficiency for use in real-time applications. These three main challenges can be divided into specific components, each of which is described in this section.

4.2.1 *Advertisement selection*

The data-driven approach to response prediction that allows DSPs to estimate the value of an ad impression currently entails the collection and use of user data by DSPs. Preventing DSPs from gaining any knowledge of user interests, while allowing prediction of click-through rates and learning from observed user behaviour, poses the first of the main research challenges. Several aspects of this challenge need to be considered.

USER PROFILING The collection and encoding of user data for use in CTR prediction is currently performed in a manner that discloses individual page views to DSPs, as described in Chapter 2. Both the contents of user profiles and the method of data collection need to be altered to preserve user privacy. The high dimensionality of user profiles, often encoded as sparse binary vectors before being used in the response prediction model, is a particularly challenging problem. While such sparse binary vectors can be efficiently compressed and used in the clear, processing high-dimensional user profiles without disclosing the contents of these profiles foregoes efficient compression and hinders performance optimizations based on the profile sparsity.

RESPONSE PREDICTION AND BIDDING Naturally, the lack of raw user data requires changes to the response prediction and bidding algorithms. These algorithms must make estimations by combining the user profile, private to the user, with the model parameters, which we consider to be private to the DSP due to fierce competition in the advertising market. These estimations must be made without disclosing information about user interests, which could be inferred from e. g. the predicted response or the bid value, in the process.

USER FEEDBACK Bidders typically require user feedback, such as clicks on ads, to improve their targeting models. In previous work, such as the works described in Section 3.4, such feedback is generally aggregated or mixed to obtain global click-through rates, which are considered sufficient for campaign optimization in those works. Modern ad selection models as described in Section 2.3, however, use individual responses as training data. It is thus necessary to report such individual responses in a privacy-preserving manner, and process them quickly, privately, and in the absence of the user.

4.2.2 *Integration into the application setting*

Although preserving privacy in individual bidders' advertisement selection tasks is challenging in and of itself and is vital to our goal of preserving privacy in OBA, it cannot guarantee the user's privacy throughout the whole RTB process. Revealing which advertisement was shown to a user, for instance, may harm the user's privacy as argued in e. g. [22]. We therefore consider the additional challenge of integrating the advertisement selection process into the RTB setting. Again, several factors contribute to this challenge.

COMPLEX APPLICATION SETTING The application setting we consider, as described in Section 4.1, is based on a complex advertising landscape. Some companies within this online advertising ecosystem aim to own the entire value chain, i. e. act as DSP, SSP and ad exchange simultaneously. One can therefore not reasonably assume that parties fulfilling different roles will not collude. We must thus assume that parties offering different services collude.

PRIVACY OF SELECTED ADVERTISEMENTS Revealing the advertisements that were shown to a user may harm the privacy of that user, as described in Section 3.5. Moreover, DSPs with a small set of similar campaigns may expose users to homogeneity attacks. If a bidder with a portfolio consisting entirely of campaigns for loans wins an auction, they can infer that the user may have an interest in loans. Such inferences can be made even if the DSP is not aware of the exact advertisement that is shown to the user. Ideally, a DSP should not know that one of their bids won the auction to prevent such attacks. However, this conflicts with the need for individual user responses, and may not be desirable due to business considerations.

REQUEST FILTERING To limit the amount of computation and communication required by both DSPs and ad exchanges, some ad exchanges allow DSPs to filter ad requests before they are sent to the DSP. Such filtering conditions allow the exclusion of ad requests based on various characteristics. Filtering of ad requests may disclose user information to DSPs in a manner similar to the microtargeting attack described by Korolova [59] (see Section 3.5). Sufficiently precise filtering conditions allow DSPs to make inferences about user groups or even individual users. Not performing filtering, however, may result in a significant increase in computation and communication costs, as every ad request must be processed by every DSP.

ADVERTISEMENT SERVING Many online advertisements contain large media elements, such as images or videos. To reduce the bandwidth necessary for bidding, these media elements are typically not included in the bid response. Instead, the bid response that wins the auction instructs the user's web browser to retrieve the media elements from an advertisement server. However, this practice may give the server insight into which advertisement was shown to which user.

4.2.3 *Efficiency*

Naïvely applying privacy protections to a task that makes extensive use of sensitive data, such as OBA, can have disastrous performance consequences. Although privacy-preserving protocols exist for many of the primitive operations required for OBA, simply replacing each of these operations with the corresponding protocols is likely to lead to extremely bad performance. We therefore consider as a final research challenge minimization of the performance impact, such that sufficient efficiency is gained for a real-time application.

4.3 METHODOLOGY

To achieve the main research goal of alleviating the privacy concerns caused by the use of sensitive personal information for advertisement selection while allowing behavioural targeting of advertisements, we design and implement two complete privacy-preserving OBA protocols. The two protocols are based on two different secure multi-party computation techniques, but adhere to the same application setting as defined in Section 4.1 and fulfil the same set of design requirements. To answer research questions 1 to 4 defined in Section 1.4, we divide the protocols into four subprotocols. To answer research question 5, we analyse the efficiency of the protocols in terms of computational and communicational complexity, and measure the runtime of proof-of-concept implementations of the protocols.

4.3.1 *Design requirements*

In designing a privacy-preserving OBA system, we focus on a subset of the identified challenges and techniques. As the main contribution of this thesis is privacy-preserving advertisement selection in the

RTB model, we limit the challenges that we consider to exclude homogeneity attacks, request filtering, and efficient advertisement serving. Based on the research goals described in Section 1.4 and the identified challenges, we specify the following requirements for the design of a privacy-preserving OBA system:

PREDICTIVE PERFORMANCE The system must be able to learn from observed user responses. The logistic regression models that are used by DSPs to predict responses to ad impressions must be updated periodically with the goal of improving predictive performance of the model. We assume that the use of online or batched updates of the form of Eq. (2.5), using a static global learning rate parameter, converge to provide sufficient predictive performance and do not require a minimum level of prediction accuracy.

INTEGRATION The system must be applicable to the RTB model and integrated into the advertising landscape as described in Section 4.1. The following properties must be satisfied:

- A user receives the advertisement with the highest estimated value after sending a single request to a fixed set of known parties (e. g. one or more ad exchanges). The user cannot be assumed to know the topology of the OBA landscape, and can thus not send individual ad requests to every DSP.
- An ad exchange requests bids from all DSPs upon receiving an ad request;
- DSPs submit bids on ad impressions on behalf of advertisers.

PROFILE PRIVACY Information from which the interests of a user can be inferred must not be revealed to any party other than the user. The following information is considered to be private to a user:

- The contents of user profiles;
- The advertisements that are shown to a user, the campaigns to which these advertisements belong, and the DSPs that are responsible for these campaigns;
- The predicted response of a user to an advertisement;
- The bid value associated with an advertisement for a user;
- The actual response of a user to an advertisement.

Note that user anonymity is not assumed nor required for privacy; we aim to hide the *behaviour* of a user rather than the *identity* of a user. Therefore, user identifiers are not considered to be private to a user. A consequence of revealing user identifiers is that temporal behaviour of users can be tracked in the sense that advertising companies can determine when a user is online. Hiding such temporal behaviour, however, requires unlinkability of advertisement requests, which may impose significant additional computational costs.

MODEL PRIVACY Due to fierce competition in the advertisement industry, model parameters used by a DSP must not be revealed to any party other than the DSP. The following parameters used by DSPs in estimating the value of an ad impression are considered to be private to the DSPs:

- The model parameters $\boldsymbol{w}$;
- The parameters c_1 and c_2 used in a campaign's bidding function;
- The learning rate η;
- The predicted response of a user to an advertisement;
- The bid value associated with an advertisement for a user.

Note that the predicted user response and the bid value are private to both the user and the DSP. These values should therefore not be directly revealed to any party, and can only be used in encrypted or aggregated form.

4.3.2 *Evaluation*

Our research is aimed at preserving privacy in OBA rather than optimizing the predictive accuracy of OBA techniques. We therefore evaluate the two protocols based on whether they preserve user privacy and the computational cost of the provided protections, rather than the predictive performance of the implementations. Since we design the protocols to meet the privacy requirements outlined in Section 4.3.1, under the assumption that these requirements provide the desired privacy protection, the focus of our evaluation will be on the computational performance of the protocols.

Due to the lack of existing work that aims to satisfy the same goals as our research, a comparison between the protocols we present and previous work would be inaccurate. Although the ObliviAd protocol described in Section 3.4 aims to hide user interests from advertisers in a manner similar to this research, it cannot estimate the value of an ad impression and is therefore incompatible with the RTB setting, nor with the data-driven prediction models encountered in modern OBA solutions. We therefore compare our two protocols with each other rather than with previous work.

We measure the computational performance of a proof-of-concept implementation of our protocols by simulating the OBA process for varying application settings. We measure the runtime of our protocols w.r.t. two variables. Firstly, we increase the number of DSPs present in the simulation, while keeping all other variables fixed, to analyse the impact of the number of DSPs on the runtime of the protocol. Secondly, we increase the size of user profiles while keeping all other variables fixed. For an honest comparison, the implementations of both protocols simulate all parties in the advertising landscape in a single process thread, such that all computations are performed sequentially. Moreover, we assume the existence of local user profiles.

To simulate user profiles, we use a public real-world dataset containing 10 days of click-through data from the *Avazu* advertising platform. The *Avazu* dataset, originally released for a click-through rate prediction challenge[2], contains chronologically ordered impression data containing both clicks and non-clicks, as well as 20 variables describing the ad request. Contrary to most other publicly available click-through datasets, the *Avazu* datasets contains information that can be used to uniquely identify individual users, which we use to simulate the profile update phase for different users. Since we use feature hashing, the meaning of the other variables encountered in the dataset, some of which are anonymized, is irrelevant.

2 Avazu Click-Through Rate Prediction challenge on Kaggle, https://www.kaggle.com/c/avazu-ctr-prediction

5 AHEAD: ADDITIVELY HOMOMORPHIC ENCRYPTION IN ADVERTISING

The first of two protocols we present is AHEad, a privacy-preserving protocol for OBA that ensures user privacy by processing sensitive user information in encrypted form. AHEad combines a threshold variant of an additively homomorphic encryption scheme with a logistic regression model for response prediction. Advertisements can therefore be served based on detailed user profiles, while performance linear in the size of user profiles can be achieved. To the best of our knowledge, AHEad is the first protocol to achieve our goals of preserving user privacy in OBA, that is compatible with the RTB mechanism of buying ads, and allows the use of detailed user profiles and machine learning methods. Our use of threshold homomorphic encryption distributes trust between parties, such that no single party can decrypt sensitive information, while enabling multiple DSPs to operate on the same encrypted user data.

In this chapter, we describe and evaluate the design of AHEad. We start by outlining cryptographic preliminaries in Section 5.1, after which we describe the protocol design in Section 5.2. An analysis of the efficiency of the protocol is provided in Section 5.3. Finally, we evaluate the AHEad protocol in Section 5.4.

5.1 CRYPTOGRAPHIC PRELIMINARIES

The AHEad protocol makes extensive use of a threshold variant of an additively homomorphic encryption scheme, such as Paillier [78] as used in our implementation of AHEad. The homomorphic properties of Paillier's encryption scheme allow the computation of an encryption $\mathcal{E}_{pk}(m_1 + m_2) = \mathcal{E}_{pk}(m_1) \cdot \mathcal{E}_{pk}(m_2)$, where $\mathcal{E}_{pk}(x)$ is the encryption of message x under public key pk, by any party with access to the public key pk. Using this property, one can also compute $\mathcal{E}_{pk}(c \cdot m) = \mathcal{E}_{pk}(m)^c$ for a public constant c.

AHEad uses a two-party threshold variant of the Paillier encryption scheme by Hazay et al. [53] to distribute trust between parties. The scheme described by Hazay et al. is a simplification of the t-out-of-n threshold decryption scheme presented by Damgård and Jurik [32] to form a 2-out-of-2 scheme. The two-party threshold Paillier scheme uses additive shares of a private decryption exponent as private key

This chapter is an extended version of a paper submitted to the 9th IEEE International Workshop on Information Forensics and Security (WIFS) 2017.

shares, allowing both parties to compute a decryption share. The plaintext is then immediately computable from the combination of both decryption shares.

In the remainder of this chapter, we use $[m]$ to denote an encryption of a message m using the two-party threshold Paillier scheme, and have omitted individual share decryption and combination steps from our protocol descriptions for the sake of brevity. The notations $[\cdot]_u$ and $[\cdot]_{DSP}$ represent encryptions using any asymmetric encryption scheme under the public key of the user or the DSP, respectively. Finally, we use the notation $[\boldsymbol{x}]$ to denote the element-wise encryption of a vector $\boldsymbol{x}$.

5.2 PROTOCOL DESIGN

As described in Section 4.1, ad exchanges and DSPs are assumed to collude. To preserve privacy in the presence of colluding parties, we slightly modify our application setting by introducing an additional entity called Privacy Service Provider (PSP). The PSP is assumed not to collude with any other party and is trusted to follow the protocol specification. However, the PSP is not trusted with private data in unencrypted form. To distribute trust between the PSP and other parties, we use a threshold homomorphic cryptosystem, where one share of the private key is held by the PSP, and each of the DSPs and ad exchanges holds a copy of the other key share. This setup requires collaboration between the PSP and an advertising company in order to decrypt an encrypted value.

AHEad is divided into 5 different phases: initial setup, user profiling, bidding, auction, and model update. Each of these phases is described in the following subsections. An explanation of symbols used in the description of the protocol design is provided in Table 5.1.

5.2.1 *Initial setup*

Prior to execution of the protocol, a key pair for the two-party Paillier scheme is generated. Key generation can be performed either by a TTP or using a distributed protocol as outlined e. g. by Hazay et al. [53]. The private key is secret shared between the PSP and advertising companies, such that the PSP holds one part of the private key, and each DSP and ad exchange holds a copy of the other part of the private key. Note that we assume the existence of a secure communication channel between each party, such that a partial decryption sent from the PSP to a DSP cannot be intercepted by any other party. Moreover, each DSP generates a key pair for use in the profile update protocol, using any asymmetric cryptosystem and publishes their private key. Similarly, each user generates a key pair using any asymmetric cryp-

SYMBOL	EXPLANATION
u	Unique user identifier.
k	Unique campaign identifier.
γ	Unique DSP identifier.
K_γ	Set of campaigns run by a DSP γ.
$\boldsymbol{x}$	Input feature vector obtained by feature hashing.
$\boldsymbol{w}_k$	Model parameter vector for a campaign k, where $w_{k,i}$ is the i^{th} coordinate of $\boldsymbol{w}_k$.
η_k	Learning rate parameter for campaign k.
$B_k(\cdot)$	Bidding function for campaign k.
b_k	Bid value for campaign k.
a_k	Advertisement associated with campaign k.
$\pi(\boldsymbol{x})$	Random permutation function, which re-orders the elements of vector $\boldsymbol{x}$.
$\pi^{-1}(\cdot)$	Inverse permutation of $\pi(\cdot)$, such that $\pi^{-1}(\pi(\boldsymbol{x})) = \boldsymbol{x}$.

Table 5.1: Explanation of symbols used in AHEad

tosystem and publishes their public key. Finally, advertisers set up their campaigns such that DSPs can bid on their behalf.

5.2.2 *User profiling phase*

In the user profiling phase, information about a user's behaviour must be shared with a DSP without disclosing any information about the user's interests, as per the *profile privacy* requirement. The common practice of sharing individual browsing events via e. g. tracking cookies, as described in Section 2.2.1, evidently does not fulfil this requirement. Although browsing events could be shared in encrypted form to ensure profile privacy, encoding this information under encryption into a finite feature vector usable in the response prediction task poses a significant challenge. Therefore, browsing behaviour is recorded within the user's web browser to form a locally stored user profile. Local profiling can be performed using existing techniques, such as RePriv [43] as described in Section 3.3. The resulting profile information is captured into a d-dimensional feature vector $\boldsymbol{x}$ using feature hashing (see Section 2.2.3).

Since sending the full d-dimensional feature vector during each advertisement request would incur prohibitively high communication costs, feature vectors are cached at DSPs. Caching significantly reduces the amount of communication required during the time-sensitive advertisement selection phase, and allows feature vectors to be updated in the background to minimize delays experienced by the

Protocol 5.1 Profile update protocol of AHEad, executed jointly between a user, DSP, and PSP, and initiated periodically by every user.

```
1:  procedure USER:SEND-PROFILE-UPDATE($x, u$)
2:      Pick $r \in_R \mathbb{Z}_d$
3:      $P \leftarrow \{(i + r \pmod d), [x_i]) \mid x_i \neq 0\}$
4:      invoke PSP:EXPAND($P, \left[(u, r)\right]_{DSP}$) at PSP
5:  end procedure

6:  procedure PSP:EXPAND($P, \left[(u, r)\right]_{DSP}$)
7:      for $j \leftarrow 1, d$ do
8:          $\left[\tilde{x}_j\right] \leftarrow \begin{cases} [v] & \text{if } (j, [v]) \in P \\ [0] & \text{otherwise} \end{cases}$
9:      end for
10:     invoke DSP:UPDATE-PROFILE($[\tilde{x}], \left[(u, r)\right]_{DSP}$) at DSP
11: end procedure

12: procedure DSP:UPDATE-PROFILE($[\tilde{x}], \left[(u, r)\right]_{DSP}$)
13:     Decrypt $\left[(u, r)\right]_{DSP}$
14:     Select profile $[x]$ for user $u$
15:     for $i \leftarrow 1, d$ do
16:         $[x_i] \leftarrow \left[\tilde{x}_{i-r \pmod d}\right]$
17:     end for
18: end procedure
```

user. To further decrease communication costs, profile updates are periodically sent by the user. Profile updates can be performed in either an incremental fashion or by completely replacing the user profile, depending on the expected size of user profiles. Incremental updates may be less costly to encrypt and transmit for users, but require computationally expensive operations at DSPs as each coordinate of the cached user profile must be updated with the incremental update. AHEad takes the latter approach of replacing the user profile. However, the protocol can easily be modified to process incremental updates by using the additively homomorphic properties of the used encryption scheme.

Protocol 5.1 outlines the steps that are performed during a periodic profile update. When a user u initiates the profile update protocol, they generate a d dimensional feature vector x from their local profile information using the hashing trick. Due to this hashing trick, x is expected to be a very sparse, but high-dimensional vector. To reduce the computation and communication costs for the user, x is encoded as a set of pairs of indices and values for only the non-zero elements in x:

$$P \leftarrow \left\{\left(i + r \pmod d), [x_i]\right) \mid x_i \neq 0,\ r \in_R \mathbb{Z}_d\right\}, \tag{5.1}$$

where r is a randomization term used to rotate the indices of non-zero elements. The compression of $\boldsymbol{x}$ greatly reduces the amount of encryptions performed by the user, as well as the bandwidth required to transmit the encrypted profile. However, profile compression comes at the cost of additional computational cost and bandwidth at the PSP and DSP, as the compressed profile must still be expanded to a dense form that is usable by the logistic regression model without revealing information about the profile contents. Since the indices of non-zero elements could be used to infer profile contents, profile expansion must be performed in such a way that these indices are not revealed. Therefore, profile expansion is performed jointly by the PSP and DSP, where the PSP knows that certain elements are non-zero, but not what the locations of those elements are in $\boldsymbol{x}$, and the DSP knows the locations of elements, but not what their values are.

To perform the expansion, the compressed representation P of the user's profile is sent to the PSP, along with the ciphertext $\left[(u,r)\right]_{DSP}$. The rotation of indices prevents the PSP from learning the true indices of the non-zero elements of $\boldsymbol{x}$. The PSP subsequently expands P into an element-wise encryption of the original feature vector $\boldsymbol{x}$ with its elements rotated r times due to the randomization of indices performed by the user, setting any element not present in P to an encryption of zero. This expansion is sent to the relevant DSP, along with the ciphertext $\left[(u,r)\right]_{DSP}$. The DSP decrypts the user identifier and random number, and rotates the received profile expansion back to its original indices using r. Finally, the DSP stores the encrypted profile information of the user.

5.2.3 *Bidding phase*

During the bidding phase, every DSP calculates a bidding price for each of their campaigns, based on a cached user profile. The bidding phase is initiated by the user contacting an ad exchange with a request for an advertisement. The ad exchanges sends a bid request to every DSP, each of which executes the bidding protocol. For each advertising campaign k, the user response $\hat{y}$ is assumed to be estimated using a logistic regression model as described in Section 2.3.1. The bidding price is subsequently derived from $\hat{y}$ using a linear bidding function B_k as per the assumptions outlined in Section 4.1.

The sigmoid function σ that is used to make predictions in logistic regression models (see Eq. (2.3)) is non-trivial to compute under additively homomorphic encryptions due to the division and exponentiation used in the function. In existing literature considering the sigmoid function in privacy-preserving protocols, two different approaches are used to compute the result of the sigmoid function. The first approach is to use an approximation of the function, either using a Taylor series [105] or a piecewise linear approximation [28]. The

Protocol 5.2 Bidding protocol of AHEad, executed jointly between a DSP γ and the PSP, and invoked by the ad exchange at every DSP.

```
1: procedure DSP:SUBMIT-BID($\{(\boldsymbol{w}_k, B_k, a_k) \mid k \in K_\gamma\}, [\boldsymbol{x}]$)
2:     for all $k \in K_\gamma$ do
3:         $[s_k] \leftarrow \prod_{i=1}^{d} [x_i]^{w_{k,i}}$
4:     end for
5:     Pick random permutation function $\pi(\cdot)$
6:     $[\boldsymbol{s}'] \leftarrow \pi([\boldsymbol{s}])$
7:     invoke $[\hat{\boldsymbol{y}}'] \leftarrow$ PSP:CALCULATE-SIGMA($[\boldsymbol{s}']$) at PSP
8:     $[\hat{\boldsymbol{y}}] \leftarrow \pi^{-1}([\hat{\boldsymbol{y}}'])$
9:     for all $k \in K_\gamma$ do
10:        Re-randomize $[\hat{y}_k]$
11:        $[b_k] \leftarrow B_k([\hat{y}_k])$
12:        send $[a_k]_u, [b_k], [\hat{y}_k], [k]$ to PSP
13:    end for
14: end procedure

15: procedure PSP:CALCULATE-SIGMA($[\boldsymbol{s}]$)
16:     Decrypt $[\boldsymbol{s}]$
17:     for all $s_i \in \boldsymbol{s}$ do
18:         $\hat{y}_i \leftarrow \sigma(s_i)$
19:     end for
20:     return $[\hat{\boldsymbol{y}}]$
21: end procedure
```

second approach is revealing the input of the sigmoid function to the holder of the private key, such that the result can be computed in the clear [11, 75]. We argue that in our setting, revealing the input $\boldsymbol{w}^\top \boldsymbol{x}$ of the sigmoid function to the PSP is acceptable. Since $\boldsymbol{w}$ is not known to the PSP, revealing $\boldsymbol{w}^\top \boldsymbol{x}$ does not leak information about $\boldsymbol{x}$. Note that, if $\boldsymbol{w}$ were known to the PSP, it could be possible for the PSP to extract information about $\boldsymbol{x}$. Similarly, if $\boldsymbol{x}$ were known to the PSP, it could be possible to extract information about $\boldsymbol{w}$. The result $\hat{y}$ of the sigmoid function, which is the estimated click probability, may leak information about the degree to which the user is interested in a particular topic if a DSP has a small set of similar campaigns. However, no information about the identity of the user is passed to the PSP during the bidding phase, and thus the PSP cannot infer any information other than that a user exists who is interested in a particular topic. Since each DSP submits to the PSP multiple inner products $\boldsymbol{s}$, computed for all campaigns the DSP is responsible for and permuted before submission, the PSP cannot link estimated click probabilities to individual campaigns, provided the DSP is responsible for multiple campaigns. Alternatively, a DSP can submit additional encryptions of randomly generated values to mask the real inner product. Finally, the result of the sigmoid function does not reveal information about bid values to the PSP, as the bidding functions are unknown to the PSP. We therefore take the second approach of computing the sigmoid function in the clear at the PSP, and returning an encryption of the click probability to the DSP.

Protocol 5.2 outlines the bidding protocol, as invoked at every DSP γ by the ad exchange when requesting bids. The model parameters $\boldsymbol{w}_k$ of a campaign k are known to the DSP, allowing computation of $\boldsymbol{w}_k^\intercal \boldsymbol{x}$ in a single round using the homomorphic properties of the used cryptosystem (see line 3 of Protocol 5.2). The resulting vector of inner products $\boldsymbol{s}$ is randomly permuted before being sent to the PSP to obtain encryptions of click probabilities. Each encrypted click probability $[\hat{y}_k]$ is passed through the bidding function B_k, which can also be computed using the homomorphic properties of the used cryptosystem as both the scaling factor c_1 as the offset term c_2 are known to the DSP. Finally, an encryption of the advertisement a_k under the user's key, along with encryptions of the bid value b_k, the click probability $\hat{y}_k$, and the campaign identifier k under the shared threshold key are sent to the PSP for use in the auction protocol.

5.2.4 *Auction phase*

During the auction phase, the PSP and the ad exchange engage in a secure comparison protocol, such as described in e.g. [68, 90], to select the highest bid and associated advertisement. The use of a secure comparison protocol prevents either party learning both the winning advertisement and the corresponding bid value, such that user interests are not revealed as per the profile privacy requirement and bid values are not revealed as per the model privacy requirement. At the start of the auction phase, the PSP holds encryptions of all bids submitted by the DSPs through the bidding protocol. These bids consist of encryptions of the bid value, the predicted response, and the campaign identifier, each under the shared threshold key, as well as an encryption of the advertisement under the user's key. The bids are randomly permuted by the PSP before being sent to the ad exchange. After repeated execution of the secure comparison protocol to find the highest bid, the ad exchange holds an encryption of the index of the highest bid. The index is decrypted by the ad exchange with cooperation of the PSP, allowing the ad exchange to forward the encrypted bid information to the user, who decrypts and displays the advertisement and stores the other ciphertexts for use in the model update protocol. Since the bids were permuted before being sent to the ad exchange, revealing the index of the highest bid does not reveal any information about which campaign won the auction.

5.2.5 *Model update phase*

After an advertisement is shown to a user, the response prediction model associated with the shown advertisement can be updated to learn from the observed user action y, which is a binary label indicating either a click (1) or no click (0), as described in Eq. (2.5). In the current, non-privacy-preserving setting, only clicks are reported to the DSP, and non-clicks are inferred from the absence of click reports. Since we want to hide whether a user clicked on an advertisement, however, we must always report something, including non-clicks. Note that we cannot reveal the value $\hat{y} - y$ used in the model update, since its value leaks information about both $\hat{y}$ and y. Moreover, we cannot reveal $\hat{y}$ to the PSP as the PSP could link that to values observed earlier during the bidding phase. Therefore, we must rely on users to calculate the full gradient of the loss function $\boldsymbol{g} = (\hat{y} - y)\boldsymbol{x}$ used in model updates. Ensuring profile privacy through the use of user-computed gradients, however, comes at the cost of significant bandwidth use by the user, as they must upload the full d-dimensional gradient vector to the PSP for every advertisement shown.

Since DSPs must know the values of model parameters $\boldsymbol{w}$ to perform efficient predictions during the bidding phase, model updates must be revealed to the DSP. A single update gradient, however, contains information about the contents of the user profile used to calculate the gradient. Therefore, we aggregate update gradients of different users on a per-campaign basis at the PSP before revealing the aggregated model update to the DSP. From the aggregated model update, the DSP cannot determine which values originated from which individual gradient, thus preserving profile privacy.

Protocol 5.3 outlines the model update protocol of AHEad. The user calculates the gradient of the loss function in the encrypted domain based on the encryption of $\hat{y}$ received together with the advertisement. The encrypted gradient is sent to the PSP along with encryptions of the bid value and the campaign identifier. Communication is performed via the ad exchange, which acts as a proxy to hide the identity of the user from the PSP to ensure that the PSP cannot link the user identity to the shown advertisement. After decrypting the campaign identifier in cooperation with the ad exchange, the PSP adds the encrypted gradient and the encrypted bid value to the campaign-specific aggregates. After aggregating sufficient values for a campaign from a small batch of users, the aggregate gradient is sent to the DSP along with decrypted shares of the gradient elements, such that the DSP can decrypt the gradient and update the campaign's model parameters. Additionally, the PSP reveals the aggregated bid values such that advertisers can be billed without revealing individual bid values.

Protocol 5.3 Model update protocol, executed jointly by a user, PSP, and DSP, and initiated by a user after viewing an advertisement

```
1: procedure USER:SEND-MODEL-UPDATE($x, y, [\hat{y}], [b], [k]$)
2:     $[\delta] \leftarrow [\hat{y}] \cdot [-y]$
3:     for $i \leftarrow 1, d$ do
4:         $[g_i] \leftarrow \begin{cases} [\delta]^{x_i} & \text{if } x_i \neq 0 \\ [0] & \text{otherwise} \end{cases}$
5:     end for
6:     Re-randomize $[b], [k]$
7:     invoke PSP:AGGREGATE($[g], [k], [b]$) at PSP via AdX
8: end procedure

9: procedure PSP:AGGREGATE($[g], [k], [b]$)
10:    Decrypt $[k]$
11:    for $i \leftarrow 1, d$ do
12:        $[\hat{g}_{k,i}] \leftarrow [\hat{g}_{k,i}] \cdot [g_i]$
13:    end for
14:    $[\hat{b}_k] \leftarrow [\hat{b}_k] \cdot [b]$
15:    if sufficient values are aggregated for campaign $k$ then
16:        invoke DSP:UPDATE($[\hat{g}_k], k$) at DSP
17:    end if
18: end procedure

19: procedure DSP:UPDATE-MODEL($[g], k$)
20:    Decrypt $[g]$
21:    for $i \leftarrow 1, d$ do
22:        $w_{k,i} \leftarrow w_{k,i} - \eta_k g_i$
23:    end for
24: end procedure
```

5.3 PERFORMANCE ANALYSIS

We evaluate the performance of AHEad using both a theoretical analysis in terms of the number of cryptographic operations performed by parties participating in each of the subprotocols, and a set of measurements obtained from a proof-of-concept implementation. Moreover, we describe the communication complexity of AHEad.

5.3.1 *Computational complexity*

To analyse the computational complexity and scalability of AHEad, we list the number of cryptographic operations performed by each party per invocation of every subprotocol. The amount of performed operations depends on a number of variables, listed in Table 5.2. The

SYMBOL	DESCRIPTION
ν	Number of non-zero elements in the user's profile.
d	Dimensionality of user profiles.
d'	Number of ciphertexts required to describe a full d-dimensional user profile.
κ	Number of campaigns of a DSP.
K	Total number of campaigns.
ζ	Number of model updates aggregated per campaign.
ϵ	Size in bits of a single ciphertext.
ϵ'	Size in bits of a partially decrypted ciphertext.
δ	Size in bits of an integer.
ρ	Number of rounds required for the secure comparison of K values.
γ_a	Number of bits transmitted by the data holder in $K-1$ invocations of the secure comparison protocol.
γ_b	Number of bits transmitted by the private key holder in $K-1$ invocations of the secure comparison protocol.

Table 5.2: Symbols used in the computational analysis of AHEad.

number of non-zero elements and the dimensionality of user profiles are dependent on the profiling method used, and can be chosen by the advertising industry to form a trade-off between expressiveness of user profiles and computational costs. The number of model updates aggregated per campaign forms a trade-off between learning speed (and thus prediction accuracy) and privacy. Finally, the number of campaigns of a DSP and the total number of campaigns are determined by the shape of the advertising landscape. The latter variable is based on the existence of a single ad exchange and a single PSP, and can be reduced by dividing DSPs among multiple ad exchanges and PSPs. Note that ν, κ, K, and ζ are expected to be several orders of magnitude smaller than d.

Table 5.3 lists the amortized number of operations performed by each party for each subprotocol. All subprotocols have a complexity at most linear in the number of campaigns and the size of user profiles. The bidding protocol requires the largest number of operations due to the computation of $\left[\boldsymbol{w}^{\top}\boldsymbol{x}\right]$ for every campaign of a DSP. However, the computation of these values is trivially parallelizable and can thus be sped up significantly using multiple processors. Moreover, the required number of operations can be reduced by employing a sparsity-inducing update model, such as the Follow The (Proximally) Regularized Leader (FTRL) model described by He et al. [54], which performs efficient L_1 and L_2 regularization to reduce

PROTOCOL	OPERATION	USER	ADX	DSP	PSP
Profile update	Encryption	ν			$d-\nu$
	Decryption			1	
Bidding	Encryption			2κ	K
	Share decryption			κ	K
	Multiplication			$\kappa(d-1)$	
	Exponentiation			κd	
	Randomization			κ	
	Bidding function			κ	
Auction	Comparison		$K-1$		$K-1$
	Decryption		1		
Model update	Exponentiation	ν			
	Encryption	$d'-\nu$			
	Multiplication	1			$1+d'$
	Share decryption			$1+\frac{d'}{\zeta}$	$1+\frac{d'}{\zeta}$

Table 5.3: Number of cryptographic operations performed per invocation of each subprotocol of AHEad.

the number of non-zero values in w. A sparse model parameter vector reduces the amount of computation necessary since only non-zero parameters need to be considered in computing the inner product. Since the final stage of model parameter updates is performed in the clear, a model such as FTRL can be implemented without significant changes to the protocol and at little to no cost to the runtime of the model update protocol. Further speed improvements can be achieved by packing multiple values into a single ciphertext in the model update phase, such that $d' < d$, reducing the number of encryptions performed by the user, the number of multiplications performed by the DSP, and the number of decryptions performed jointly by the DSP and PSP. Moreover, packing reduces the amount of communication bandwidth required between the parties.

5.3.2 *Communication complexity*

We analyse the communication complexity of AHEad by listing the number of bits transmitted by each party in every round of a single invocation of every subprotocol. The required communication bandwidth depends on a number of variables, which are described in Table 5.2. The size of a partially decrypted ciphertext ϵ' includes both the original ciphertext and the partially decrypted ciphertext, as

PROTOCOL	ROUND	USER	ADX	DSP	PSP
Profiling	1	$\nu(\epsilon+\delta)+\epsilon$			
	2				$d\epsilon+\epsilon'$
Bidding	1			$\kappa\epsilon'$	
	2				$K\epsilon$
	3			$4\kappa\epsilon$	
Auction	1				$4K\epsilon$
	$2,\dots,\rho+1$		γ_a		γ_b
	$\rho+2$		4ϵ		
Update	1	$(d'+2)\epsilon$			
	2		$(d'+1)\epsilon+\epsilon'$		
	3				$\frac{d'\epsilon'+\delta}{\zeta}$

Table 5.4: Communication bandwidth in bits per invocation of each subprotocol of AHEad.

both are necessary to fully decrypt the ciphertext, such that $\epsilon' = 2\epsilon$. The number of rounds required for the auction protocol ρ depends on the round complexity of the used secure comparison protocol and the number of campaigns K, as does the total communication cost of the auction protocol. Note that the maximum of K values can be found in $\lceil \log_2 K \rceil$ invocations of the secure comparison protocol, such that the round complexity ρ is logarithmic with respect to K, as are the communication complexities γ_a and γ_b.

Table 5.4 lists the amortized number of bits transmitted by each party for each subprotocol, as well as the number of rounds of communication. The communication complexity of all protocols is at most linear in either the size of user profiles d or the number of campaigns K, provided that the communication complexity of a single invocation of the secure comparison protocol is constant. Note that if the user profile is distributed to multiple DSPs in the profile update phase, the communication complexity of the profile update phase becomes multilinear in the profile size and the number of DSPs. Only the auction protocol has a non-constant round complexity, which is logarithmic in the number of campaigns K. Since d is expected to be much larger than K, the communication complexity is dominated by the profile update and model update protocols due to their linearity with respect to d. Although the amount of communication bandwidth required between the parties during the model update protocol can be reduced by packing multiple values into a single ciphertext, as described in Section 5.3.1, the large dimensionality of user profiles and thus update gradients leads to significant bandwidth requirements.

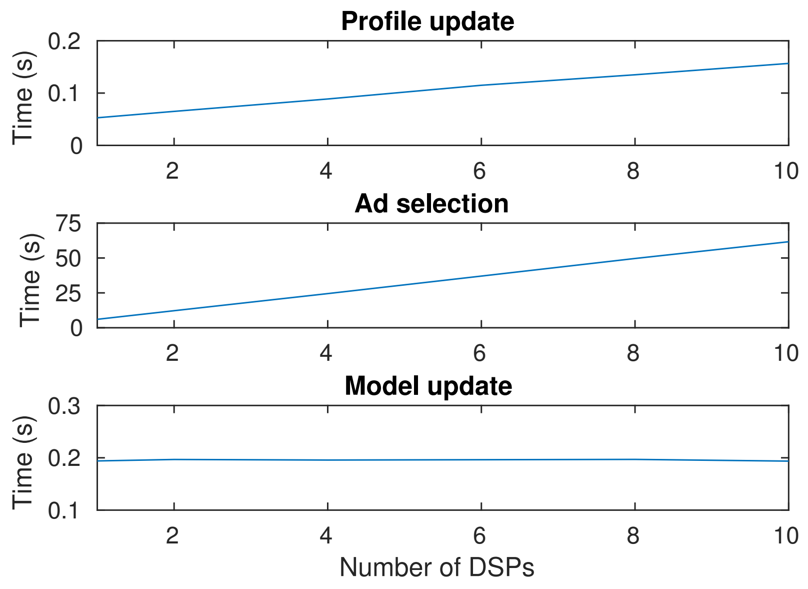

Figure 5.1: Total computation time required by a single run of each subprotocol of AHEad for an increasing number of DSPs, with a fixed profile size $d = 2^{12}$ and number of campaigns per DSP $\kappa = 10$. Model update time is averaged using $\zeta = 10$.

5.3.3 *Implementation*

To measure the runtime of AHEad, we made a proof-of-concept implementation of the protocol in C++ using the *SeComLib* library[1]. The implementation simulates all parties within a single process thread, thus performing all operations sequentially. All cryptographic operations use a key length of 2048 bits to achieve a sufficiently high security level[2]. Real values, such as model weights, are represented as 16-bit fixed-point numbers. For the auction phase, an implementation of the secure comparison protocol by Erkin et al. [39] as provided by *SeComLib* is used. Furthermore, data packing is used to speed up model updates.

The runtime tests were executed on a mobile workstation running Arch Linux on an Intel® Core™ i7-3610QM 2.3 GHz quad-core processor with 8 GB RAM. Figure 5.1 shows the impact of the number of DSPs, and thus the total number of campaigns, on the total computation time. The time spent in the profile update protocol increases linearly with the number of DSPs since a profile update must be processed by each DSP. In a real-world setting, DSPs would operate in parallel, rather than our sequential simulation, resulting in constant-time performance of the profile update protocol. The bidding and auction protocols cannot be fully parallelized across DSPs due to the

1 Secure Computation Library, http://cys.ewi.tudelft.nl/content/secomlib
2 See e. g. https://www.keylength.com for key lengths as recommended by various organizations. The NIST considers a key length of 2048 sufficiently secure until 2030.

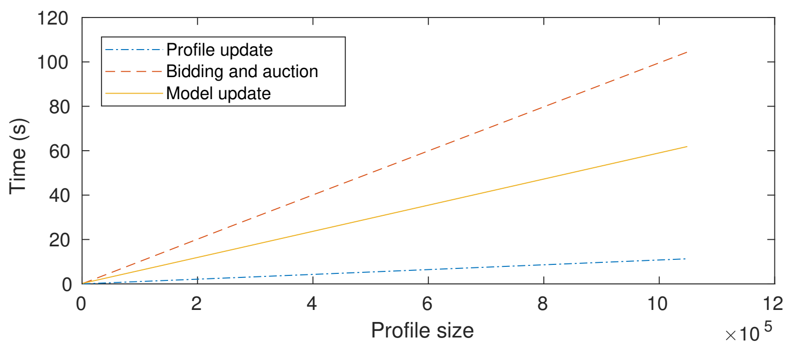

Figure 5.2: Total computation time required by a single run of each subprotocol of AHEad for an increasing profile size d, with a single DSP running a single campaign. Model update time is averaged using $\zeta = 10$.

operations performed by the PSP and ad exchange, and thus scale linearly in the number of DSP. Since the model update protocol is only performed once for every advertisement shown, and the number of advertisements shown is independent of the number of DSPs, its runtime is independent of the number of DSPs.

Figure 5.2 shows the impact of profile dimensionality on the total computation time. It is evident that, although all subprotocols scale linearly with profile size, high-dimensional profiles as used in practice, such as $d = 2^{24}$ as used by Chapelle et al. [25], result in dozens of seconds of computation time for every shown advertisement. The runtime of the bidding protocol increases particularly fast with an increase of the profile dimensionality due to the exponentiations required in the computation of the inner product $\boldsymbol{w}^\top \boldsymbol{x}$. Although the runtime of the bidding protocol can be decreased through the use of parallelization and sparse models, calculating the inner product of large vectors under additively homomorphic encryption is an inherently expensive operation, and will lead to user-noticeable delays.

As described in Section 5.3.2, the communication costs are dominated by the profile update and model update protocol. We show this in Table 5.5 by listing the communication bandwidth in kibibytes required by the AHEad protocol for specific parameters used in our implementation. Given a key length of 2048 bits, a single Paillier ciphertext requires $\epsilon = 4096$ bits of communication. Using a 16-bit fixed-point representation of real values and a single overflow bit between values, we can pack $\lfloor 2048/17 \rfloor = 120$ values into a single ciphertext. With a profile dimensionality of $d = 2^{20}$, we thus need $d' = \lceil d/120 \rceil = 8739$ ciphertexts to represent a full update gradient. Assuming the presence of $\nu = 100$ non-zero values in a user profile and a 32-bit integer representation, and taking $\kappa = 10$, $K = 100$, and $\zeta = 10$, we obtain the bandwidth use listed in Table 5.5. From the table, it is evident that the bandwidth used during the profile update phase

PROTOCOL	ROUND	USER	ADX	DSP	PSP
Profile update	1	51			
	2				524 289
Bidding	1			10	
	2				100
	3			20	
Auction	1				200
	$2, \ldots, \rho + 1$		γ_a		γ_b
	$\rho + 2$		2		
Model update	1	4371			
	2		4371		
	3				874

Table 5.5: Bandwidth usage of AHEad in KiB per invocation of each subprotocol, based on realistic parameters used in our implementation of AHEad.

far exceeds any other phase, with the PSP transmitting 512 MiB per profile update. The bidding phase, however, is very efficient, with each DSP transmitting a total of only 3 KiB per campaign. Moreover, at 4.3 MiB per viewed advertisement, the bandwidth use of the user is acceptable for modern unmetered connections.

5.4 DISCUSSION

AHEad is, as far as we know, the first protocol using machine learning over encrypted data to preserve privacy in OBA. DSPs must cooperate with a semi-honest, non-colluding PSP to estimate a user's response and the corresponding bid price within the encrypted domain, after which the PSP and ad exchange engage in a privacy-preserving auction to select the winning bid. Encrypted reports of individual clicks and views of advertisements are reported to the PSP, where they are aggregated for billing and model update purposes.

Profile privacy as well as model privacy are ensured in the semi-honest setting by distributing trust between parties using a threshold additively homomorphic encryption scheme. At no point in the protocol are the contents of the user profile or the shown advertisement revealed to any party other than the user themselves, nor are campaign-specific model parameters maintained by the DSP revealed to any party other than the DSP. Individual bid prices are revealed to no party at all but are aggregated for billing purposes. Finally, estimated click probabilities, although revealed to the PSP, cannot be linked to users or individual campaigns, provided a DSP is responsible for multiple campaigns or masks their campaign with phony submissions.

The computation required by AHEad quickly increases with profile dimensionality, resulting in over 100 seconds of computation per bid for a profile dimensionality $d = 2^{20}$ on a single processor core of our modest hardware. Moreover, an encrypted user profile of size 2^{20} requires 500 MiB of storage space and communication bandwidth between DSPs and the PSP, as well as 4.3 MiB of bandwidth between the user, the ad exchange and the PSP for every viewed advertisement if data packing is used. Nevertheless, AHEad shows promising initial results in terms of privacy and achieves performance linear in the profile size and the number of DSPs, and can benefit from significant decreases in runtime if operations are parallelized.

6 BADASS: BEHAVIOURAL ADVERTISING WITH APPLIED SECRET SHARING

The second of the two protocols we present is BAdASS, a privacy-preserving protocol for OBA that protects user privacy by distributing trust between DSPs. Where AHEad uses expensive homomorphic encryption to operate on encrypted user data, BAdASS uses additively homomorphic secret sharing techniques to securely split sensitive information between parties. Parties collaboratively compute response predictions using a logistic regression model, and use a hierarchical secure auction protocol to select the winning advertisement. Our use of threshold secret sharing distributes trust between parties, such that collusion between any number of parties smaller than a predefined threshold does not reveal any sensitive information, while enabling DSPs to collaboratively estimate the values of ad impressions without having insight into private data.

In this chapter, we describe and evaluate the design of BAdASS. We first describe cryptographic preliminaries in Section 6.1. Subsequently, we provide a detailed description of the design of the protocol in Section 6.2, after which we analyze the computational costs of the protocol in Section 6.3. Finally, we briefly discuss the BAdASS protocol in Section 6.4.

6.1 CRYPTOGRAPHIC PRELIMINARIES

To cryptographically protect user privacy, BAdASS makes use of two forms of secret sharing, as well as an asymmetric cryptosystem that allows universal re-encryption. Secret sharing schemes distribute a secret value s between a group of parties, such that each party obtains a share of the secret. Reconstruction of s from the shares is only possible if a sufficient number of parties combine their shares, whereas any subset of parties that is not qualified to know the secret value cannot reconstruct the secret. Both secret sharing schemes used in BAdASS prevent an unqualified set of parties from learning *any* information about s, even when they combine their shares.

ADDITIVE SECRET SHARING The first form of secret sharing used in BAdASS is a simple 2-out-of-2 additive secret sharing scheme, in which a secret is shared between two parties, and both shares are re-

This chapter is an extended version of a paper to be submitted to the 21st International Conference on Financial Cryptography and Data Security 2017 quired to reconstruct the secret. In an additive secret sharing scheme, as described by e. g. Bogdanov [19], a secret value $s \in \mathbb{Z}_n$ for some number n is split into shares $s_1 \in_r \mathbb{Z}_n$ and $s_2 = s - s_1 \pmod{n}$, using as s_1 a value chosen uniformly at random from $\mathbb{Z}_n$, and as s_2 the subtraction of the random value from the secret value. Reconstruction of the secret value is as simple as addition of the two shares, modulo n. We use the notation $\langle\!\langle s \rangle\!\rangle$ to denote an additive sharing of a value s, where $\langle\!\langle s \rangle\!\rangle_i$ is the i^{th} share, and the sharing $\langle\!\langle \boldsymbol{v} \rangle\!\rangle$ of a vector $\boldsymbol{v}$ indicates an element-wise additive secret sharing of $\boldsymbol{v}$.

SHAMIR SECRET SHARING The second scheme used in BAdASS is Shamir's secret sharing scheme [91], a t-out-of-n threshold scheme in which a secret is shared among n parties, from which any subset of size at least t can reconstruct the secret. Shamir's secret sharing scheme is based on polynomial evaluations in a finite field of prime order p. To share a secret value $s \in \mathbb{Z}_p$, the holder of the secret generates a polynomial f of degree $t - 1$ with coefficients randomly chosen from $\mathbb{Z}_p$, with $f(0) = s$. Each party i for $i = 1, \ldots, n$ is given a share $s_i = (i, f(i))$. Any set of parties of size at least t can reconstruct the secret value using Lagrange polynomials:

$$f(0) = \sum_{j=1}^{t} f(x_j) \prod_{\substack{m=1 \\ m \neq j}}^{t} \frac{x_m}{x_m - x_j}, \tag{6.1}$$

where x_j is the first element of the share of a party j. We use the notation $\langle s \rangle$ to indicate a (t, n) Shamir secret sharing of a value s, for some predefined t and n, and $\langle \boldsymbol{v} \rangle$ denotes an element-wise Shamir sharing of the vector $\boldsymbol{v}$.

Shamir's secret sharing scheme is additively homomorphic, such that $\langle m_1 \rangle + \langle m_2 \rangle = \langle m_1 + m_2 \rangle$. Parties holding shares of two secret values can thus compute shares of the sum of the two values, without interaction with other parties. Furthermore, a public value c can be added to a shared secret m without interaction by adding c to each of the shares, i. e. $\langle m \rangle + c = \langle m + c \rangle$. Likewise, a shared secret can be multiplied with a public value c by multiplying each of the shares with c, i. e. $\langle m \rangle \cdot c = \langle cm \rangle$.

Multiplication of the shares of two secret values, described by polynomials f and g of degree $t - 1$, results in points on the polynomial $h = f \cdot g$ with $h(0) = f(0) \cdot g(0)$, such that the resulting shares describe the multiplication of the two secret values. However, h is of degree $2(t - 1)$, thus requiring $2t - 1$ shares to reconstruct the secret. Ben-Or et al. [16] therefore describe a multiplication protocol in which the resulting polynomial is reduced to degree $t - 1$ and randomized. Given a sharing of a value s, where $\langle s_i \rangle$ is held by party i, the degree reduction step is performed by each party splitting their share $\langle s \rangle_i$ into a new (t, n) sharing $\langle s \rangle_{i,1}, \ldots, \langle s \rangle_{i,n}$. Each party i distributes their subshares among all parties, such that party j is given the subshare $\langle s \rangle_{i,j}$.

Each party j then combines the received subshares $\langle s \rangle_{1,j}, \ldots, \langle s \rangle_{n,j}$ into a new share $\langle s \rangle'_j$. The resulting sharing $\langle s \rangle'$ is a new (t, n) sharing of the value s. Gennaro et al. [45] simplify the degree reduction and randomization steps into a single step, requiring a single round of communication per multiplication. Note that n needs to be at least $2t - 1$ for degree reduction to work.

UNIVERSAL RE-ENCRYPTION Universal re-encryption, presented as a technique for mix networks by Golle et al. [47], allows re-randomization of a ciphertext without access to the public key that was used during encryption. In BAdASS, we use the universal re-encryption technique presented in [47], based on a multiplicatively homomorphic cryptosystem such as ElGamal [38]. In this chapter, we use the notation $[\![x]\!]_u$ to denote the encryption of a value x under the public key of user u using a universal re-encryption scheme.

6.2 PROTOCOL DESIGN

In the BAdASS protocol, the contents of user profiles are secret-shared between DSPs to provide profile privacy. To perform computations on the user profile, as is necessary in computing bids and model updates, collaboration between the DSPs holding a share of the user profile is thus required. As described in the design requirements, the protocol needs to satisfy both profile privacy and model privacy. For collaborative computation of bids and model updates, however, knowledge of profile information as well as model parameters is necessary. Therefore, not only the user profile $\boldsymbol{x}$ is secret-shared between DSPs, but also campaign-specific parameters $\boldsymbol{w}$, η, c_1, and c_2.

Collaborative computation on secret-shared values introduces computational overhead, as operations are duplicated across each of the parties holding a share of the secret. We therefore introduce the notion of a *DSP group*, where we define a DSP group Γ_i to be a set of DSPs of size at least $m = 2t - 1$. Computations on behalf of a DSP $\gamma_{i,j} \in \Gamma_i$ are performed entirely within Γ_i, such that the number of DSPs performing computations for $\gamma_{i,j}$ is minimized. In our protocol descriptions, any operations performed by DSPs on secret-shared values are assumed to be performed by all DSPs in a DSP group Γ_i. Plaintext values and encrypted values are generated by the DSP responsible for the campaign on which computations are performed and, where necessary, published within Γ_i.

Since each DSP needs the cooperation of the other DSPs within their group, the DSPs have an incentive to perform computations on each other's campaigns. A potential imbalance between a DSP's computation on their own campaigns and that on campaigns of other DSPs may arise due to differing numbers of campaigns or a different rate of shown advertisements, as the model update protocol is invoked only

SYMBOL	EXPLANATION
u	Unique user identifier.
v	Unique bid request identifier.
k	Unique campaign identifier.
Γ_i	DSP group i.
$\gamma_{i,j}$	Unique DSP identifier, where $\gamma_{i,j}$ is the j^{th} DSP of DSP group Γ_i
K_{Γ_i}	Set of campaigns run by DSP group Γ_i.
$\boldsymbol{x}$	Input feature vector obtained by feature hashing.
$\boldsymbol{w}_k$	Model parameter vector for a campaign k, where $w_{k,i}$ is the i^{th} coordinate of $\boldsymbol{w}_k$.
$\boldsymbol{c}_k$	Bidding function parameters for a campaign k.
η_k	Learning rate parameter for campaign k.
b_k	Bid value for campaign k.
a_k	Advertisement associated with campaign k.
$\pi(\boldsymbol{x})$	Random permutation function, which re-orders the elements of vector $\boldsymbol{x}$.
$\pi^{-1}(\cdot)$	Inverse permutation of $\pi(\cdot)$, such that $\pi^{-1}(\pi(\boldsymbol{x})) = \boldsymbol{x}$.
M	List of vectors containing information associated with bid values for use in the auction protocol.

Table 6.1: Explanation of symbols used in BAdASS

for shown advertisements. To prevent consistent imbalances, a new grouping of DSPs can periodically be made, at the cost of a re-sharing of campaign parameters among the new DSP groups.

The choice of the recombination threshold t directly influences the size of DSP groups, and thus the amount of duplicated operations and the computational overhead experienced by DSPs. However, t also determines the trust a user can place in the system; picking a small t increases the probability of sufficient DSPs colluding to reveal secret values. A suitable trade-off must thus be made between privacy and performance.

Similar to AHEad, we introduce a PSP into the application setting to assist in privacy-preserving computations. Assumptions regarding the behaviour of the PSP remain unchanged. The PSP is thus not trusted with private data in unencrypted form, but is assumed to follow the protocol specification without colluding with any other party.

BAdASS is divided into 5 different phases: initial setup, user profiling, bidding, auction, and model update. Each of these phases is described in the following subsections. An explanation of symbols used in the description of the protocol design is provided in Table 6.1.

6.2.1 *Initial setup*

During the initial setup phase, advertisers set up their campaigns such that DSPs can bid on their behalf, and the PSP divides the DSPs into groups, such that each group contains at least m parties. Moreover, each DSP shares campaign-specific parameters among the DSPs in their group. Finally, each user generates a key pair using any multiplicatively homomorphic asymmetric cryptosystem and publishes their public key.

6.2.2 *User profiling phase*

To preserve profile privacy during the user profiling phase, browsing behaviour is recorded locally within the user's web browser. Analogous to Section 5.2.2, we assume the existence of local profile information, which is captured into a d-dimensional feature vector $\boldsymbol{x}$ using feature hashing.

To satisfy the *integration* requirement, which dictates that a user cannot be assumed to know the topology of the OBA landscape, the user profiling phase must ensure that the locally generated user profile is securely shared among DSPs without the user contacting each DSP individually. Therefore, the user splits their profile into two additive shares in $\mathbb{Z}_p$, one of which is given to the ad exchange, the other to the PSP. Both the ad exchange and the PSP, in turn, create Shamir shares from their additive shares, which are distributed among the DSP groups for which the profile update is intended. Every DSP within the group thus receives two Shamir shares, which are combined into a single Shamir sharing using the sum of the two shares. The user profiling phase is illustrated in Protocol 6.1.

Depending on the feature hashing method used, the feature vector $\boldsymbol{x}$ is either binary or contains only small values. Assuming the use of a binary feature vector, it would be ideal from the user's perspective to create additive shares of the feature vector in $\mathbb{Z}_2$, rather than $\mathbb{Z}_p$, as smaller shares reduce the required communication bandwidth. Subsequent computations on the user profile, however, must be performed in $\mathbb{Z}_p$ to represent real values with sufficient precision. Securely converting shares in $\mathbb{Z}_2$ to shares in $\mathbb{Z}_p$, however, is a problem of its own requiring at least one round of communication (see e. g. Damgård and Thorbek [33] or Bogdanov [19]). In our setting with two additive shares, it would suffice to compute the exclusive disjunction of the two shared bits in $\mathbb{Z}_p$, which, if calculated as $a \oplus b = a + b - 2ab$, requires a single invocation of the multiplication protocol per conversion. Given the large dimensionality of user profiles, however, sharing the user profile in $\mathbb{Z}_2$ would come at the cost of d invocations of the multiplication protocol in one round. One must thus make a trade-off between communication cost for the user, and computation

Protocol 6.1 Profile update protocol of BAdASS, executed jointly between a user, ad exchange, PSP and DSP group, and initiated periodically by every user.

```
1: procedure USER:SEND-PROFILE-SHARE($x, u$)
2:     Pick $\boldsymbol{r} \in_R \mathbb{Z}_p^d$
3:     $\langle\!\langle \boldsymbol{x} \rangle\!\rangle_1 \leftarrow \boldsymbol{x} - \boldsymbol{r}$
4:     $\langle\!\langle \boldsymbol{x} \rangle\!\rangle_2 \leftarrow \boldsymbol{r}$
5:     invoke SHARE-PROFILE($u, \langle\!\langle \boldsymbol{x} \rangle\!\rangle_1$) at AdX
6:     invoke SHARE-PROFILE($u, \langle\!\langle \boldsymbol{x} \rangle\!\rangle_2$) at PSP
7: end procedure

8: procedure SHARE-PROFILE($u, \langle\!\langle \boldsymbol{x} \rangle\!\rangle_m$)
9:     $\langle \langle\!\langle \boldsymbol{x} \rangle\!\rangle_m \rangle \leftarrow$ SHAMIR-SHARE($\langle\!\langle \boldsymbol{x} \rangle\!\rangle_m$)
10:    for all $\gamma_{i,j} \in \Gamma_i$ do
11:        invoke DSP:COMBINE-PROFILE($u, \langle \langle\!\langle \boldsymbol{x} \rangle\!\rangle_m \rangle_{\gamma_{i,j}}$) at DSP $\gamma_{i,j}$
12:    end for
13: end procedure

14: procedure DSP:COMBINE-PROFILE($u, \langle \langle\!\langle \boldsymbol{x} \rangle\!\rangle_m \rangle$)
15:    store $\langle \langle\!\langle \boldsymbol{x} \rangle\!\rangle_m \rangle$ for user $u$
16:    if $\langle \langle\!\langle \boldsymbol{x} \rangle\!\rangle_1 \rangle$ and $\langle \langle\!\langle \boldsymbol{x} \rangle\!\rangle_2 \rangle$ are both stored then
17:        $\langle \boldsymbol{x}_u \rangle \leftarrow \langle \langle\!\langle \boldsymbol{x} \rangle\!\rangle_1 \rangle + \langle \langle\!\langle \boldsymbol{x} \rangle\!\rangle_2 \rangle$
18:    end if
19: end procedure
```

and communication costs for the DSPs. Note that if sharing the user profile in $\mathbb{Z}_p$, the number of bits sent by the user is $2d\lceil \log_2 p \rceil$. Given a realistic choice of profile size $d = 2^{20}$ and a 32-bit field order p, this amounts to only 8 MiB per profile update. Therefore, the option of sharing the user profile in $\mathbb{Z}_p$ has been favoured in Protocol 6.1.

6.2.3 *Bidding phase*

During the bidding phase, every DSP group cooperatively calculates bidding prices for each of the campaigns the group is responsible for. When a user contacts an ad exchange with a request for an advertisement, the ad exchange sends a bid request to every DSP group, each of which executes the bidding protocol. As described in Section 4.1, user responses are estimated using a logistic regression model, and bidding values are derived from the response estimations using a linear bidding function. Moreover, the sigmoid function that is used to make predictions is evaluated in the clear by the PSP. The rationale behind letting the PSP evaluate the sigmoid function is given in Section 5.2.3.

Protocol 6.2 Bidding protocol of BAdASS, executed jointly by a DSP group Γ_i and the PSP, and invoked by the ad exchange at every DSP group.

```
procedure DSP:CALCULATE-BID($\{\langle \boldsymbol{w}_k \rangle, \langle \boldsymbol{c}_k \rangle, \langle \eta_k \rangle \mid k \in K_{\Gamma_i}\}, u, v$)
    for all $k \in K_{\Gamma_i}$ do
        $\langle s_k \rangle \leftarrow \sum_{i=1}^{d} \langle x_{u,i} \rangle \cdot \langle w_{k,i} \rangle$
        Pick a unique random value $r_k$
        Store mapping $r_k \rightarrow ([\![a_k]\!]_u, \Gamma_i)$ at PSP via AdX
    end for
    Pick random permutation function $\pi(\cdot)$
    $\langle \boldsymbol{s}' \rangle \leftarrow \pi(\langle \boldsymbol{s} \rangle)$
    invoke $\langle \hat{\boldsymbol{y}}' \rangle \leftarrow$ PSP:CALCULATE-SIGMA($\langle \boldsymbol{s}' \rangle$) at PSP
    $\langle \hat{\boldsymbol{y}} \rangle \leftarrow \pi^{-1}(\langle \hat{\boldsymbol{y}}' \rangle)$
    for all $k \in K_{\Gamma_i}$ do
        $\langle b_k \rangle \leftarrow \langle c_{k,1} \rangle \cdot \langle \hat{y}_k \rangle + \langle c_{k,2} \rangle$
    end for
end procedure

procedure PSP:CALCULATE-SIGMA($\langle \boldsymbol{s} \rangle$)
    $\boldsymbol{s} \leftarrow$ combine $\langle \boldsymbol{s} \rangle$
    for all $s_i \in \boldsymbol{s}$ do
        $\hat{y}_i \leftarrow \sigma(s_i)$
    end for
    return $\langle \hat{\boldsymbol{y}} \rangle$
end procedure
```

Protocol 6.2 outlines the bidding protocol, as invoked at every DSP group Γ_i. The model parameters $\boldsymbol{w}_k$ of a campaign k and the user profile $\boldsymbol{x}$ are shared among the DSPs in Γ_i. The multiplications that are required for the calculation of the inner product of $\boldsymbol{w}_k$ and $\boldsymbol{x}$ in line 3 are performed locally, without degree reduction. Since the results of these local multiplications are not used in further multiplications, the sum of all multiplied values is a single sharing $\langle s_k \rangle$ of degree $2t-2$. As the PSP subsequently collects and combines all shares of s_k, no degree reduction step is required in the calculation of the inner product.

As stated by the model privacy requirement, campaign parameters $c_{k,1}$, $c_{k,2}$ and η_k are private to the DSP responsible for campaign k, and must not be revealed to other DSPs. Therefore, each of these parameters is shared among the parties in the DSP group. Calculation of the bid price $b = c_1\hat{y} + c_2$ therefore requires a single invocation of the multiplication protocol for every campaign, which can be parallelized such that all bid values are calculated in a single round of communication.

To adhere to the profile privacy requirement, the shown advertisement a_k and the responsible DSP $\gamma_{i,j}$ should be revealed only to the user. During the bidding protocol, each advertisement a_k is encrypted using the user's public key. The encrypted advertisement is submitted to the PSP, via the ad exchange such that the PSP cannot link the submission to a specific DSP, along with a random number r_k and the group descriptor Γ_i. Finally, the PSP stores the mapping $r_k \to (\llbracket a_k \rrbracket_u, \Gamma_i)$, which will be used in the auction phase to retrieve the winning advertisement.

6.2.4 *Auction phase*

To ensure profile and model privacy, the predicted response $\hat{y}$ and the bid value b must not be revealed to any party, and the campaign identifier k of the advertisement shown to the user may be revealed only to the user. The values $\hat{y}$, b, η, and k are all required during the model update protocol, however. A DSP group must thus be able to retrieve the correct shares of $\hat{y}$, b, η, and k during the model update protocol, while preventing linkability between a user who reports a response and the advertisement that was shown to that user. In order to store the correct shares for later use, the auction protocol uses a hierarchical auction in which each DSP group individually engages in a secure comparison protocol to select the highest of the bids within the DSP group, after which each of the highest bids is submitted to a global auction to select the final winner.

Note that, due to the use of secret sharing, the global auction cannot be performed by the ad exchange alone. Although the PSP could engage in a secure comparison protocol with the ad exchange, secure multiplication of Shamir shares requires at least 3 parties to participate, and to maintain in the auction protocol the same level of trust as in the bidding protocol, at least m parties are required, where m is the minimum number of parties in a DSP group. In BAdASS, the global auction is therefore not performed by the ad exchange, but by a DSP group that is randomly selected by the ad exchange, denoted Γ^*. Alternatively, the ad exchange and PSP could participate in the global auction by selecting $m-2$ individual DSPs to reach a sufficiently large auction group.

The auction protocol is shown in Protocol 6.3. The protocol relies on a secure comparison protocol that takes as input shares of two values a and b, and gives as output shares of 1 if $a \geq b$, and shares of 0 otherwise. Such a protocol is described by e. g. Reistad and Toft [88]. During the procedure to find the maximum bid, shares of the highest bid, as well as additional information associated with the highest bid, are obtained via multiplication with the result of the comparison. During the multiplication, performed as $\langle \hat{b} \rangle \leftarrow \langle \rho \rangle \cdot \langle b_i \rangle + (1 - \langle \rho \rangle) \cdot \langle \hat{b} \rangle$, where $\hat{b}$ is the current highest bid, and b_i is compared against the

Protocol 6.3 Auction protocol of BAdASS, executed jointly by every DSP group Γ_i, an auction group Γ^*, and the PSP.

```
1: procedure DSP:PREPARE-AUCTION($v, \langle \boldsymbol{b} \rangle, \langle \boldsymbol{r} \rangle, \langle \boldsymbol{\hat{y}} \rangle, \langle \boldsymbol{\eta} \rangle, \langle \boldsymbol{k} \rangle$)
2:     for all $\Gamma_i$ do
3:         $\langle M_i \rangle \leftarrow \big(\langle \boldsymbol{r_i} \rangle, \langle \boldsymbol{\hat{y}_i} \rangle, \langle \boldsymbol{\eta_i} \rangle, \langle \boldsymbol{k_i} \rangle\big)$
4:         $\big(\langle b_i^{max} \rangle, \langle r_i^{max} \rangle, \langle \hat{y}_i^{max} \rangle, \langle \eta_i^{max} \rangle, \langle k_i^{max} \rangle\big) \leftarrow$ MAX-BID($\langle \boldsymbol{b_i} \rangle, \langle M_i \rangle$)
5:         Store mapping $v \rightarrow (\langle b_i^{max} \rangle, \langle \hat{y}_i^{max} \rangle, \langle \eta_i^{max} \rangle, \langle k_i^{max} \rangle)$
6:     end for
7:     invoke PERFORM-AUCTION($v, \langle \boldsymbol{b^{max}} \rangle, \langle \boldsymbol{r^{max}} \rangle$) at $\Gamma^*$
8: end procedure

9: procedure PERFORM-AUCTION($v, \langle \boldsymbol{b} \rangle, \langle \boldsymbol{r} \rangle$)
10:    $\big(\perp, \langle r^{max} \rangle\big) \leftarrow$ MAX-BID($\langle \boldsymbol{b} \rangle, \langle \boldsymbol{r} \rangle$)
11:    invoke PSP:SEND-AD($\langle r^{max} \rangle$) at PSP
12: end procedure

13: procedure MAX-BID($\langle \boldsymbol{b} \rangle, \langle M \rangle$)
14:    $\langle \hat{b} \rangle \leftarrow \langle b_1 \rangle$
15:    for $j \leftarrow 1, \big|\langle M \rangle\big|$ do
16:        $\langle \hat{M}_j \rangle \leftarrow \langle M_{j,1} \rangle$
17:    end for
18:    for $i \leftarrow 2, \big|\langle \boldsymbol{b} \rangle\big|$ do
19:        $\langle \rho \rangle \leftarrow \langle b_i \rangle \geq \langle \hat{b} \rangle$
20:        $\langle \hat{b} \rangle \leftarrow \langle \rho \rangle \cdot \langle b_i \rangle + (1 - \langle \rho \rangle) \cdot \langle \hat{b} \rangle$
21:        for $j \leftarrow 1, \big|\langle M \rangle\big|$ do
22:            $\langle \hat{M}_j \rangle \leftarrow \langle \rho \rangle \cdot \langle M_{j,i} \rangle + (1 - \langle \rho \rangle) \cdot \langle \hat{M}_j \rangle$
23:        end for
24:    end for
25:    return $\langle \hat{b} \rangle, \langle \hat{M} \rangle$
26: end procedure

27: procedure PSP:SEND-AD($\langle r \rangle$)
28:    $r \leftarrow$ combine $\langle r \rangle$
29:    $\big([\![a]\!]_u, \Gamma_i\big) \leftarrow$ lookup $r$
30:    Re-randomize $[\![a]\!]_u$
31:    Send $\big([\![a]\!]_u, \Gamma_i\big)$ to user via AdX
32: end procedure
```

highest bid, a single invocation of the degree reduction step after the addition suffices to obtain the desired value.

In the first stage of the auction protocol, each individual DSP group performs its own auction, resulting in shares of the values b, $\hat{y}$, η, and k that are associated with the highest bid. These values are stored along with the unique bid request identifier v for later use. Note that in the last comparison of this first stage, the degree of the shares of b_i^{max}, r_i^{max}, and k_i^{max} should not be reduced after the multiplications in lines 20 and 22. The shares of b and k associated with the winning bid will be sent to the PSP in a randomized order in the model update phase, and a high degree of the shares greatly reduces the probability that the PSP successfully recombines the shares. After the local auction, shares of the values b and r associated with the highest bid are shared with the auction group Γ^*, which performs a global auction of the highest bids of all DSP groups. Due to the re-sharing, the degree of the shares of b and r is reduced such that Γ^* can perform calculations on the values. Shares of the random identifier r associated with the highest bid in the global auction are sent to the PSP, where the shares are combined to retrieve the encrypted advertisement and campaign identifier associated with the highest bid.

Although it is tempting to store the mapping between r and the advertisement at the ad exchange to eliminate a round of communication during ad retrieval, doing so would violate profile privacy due to the assumption that DSPs and ad exchanges may collude. The mapping must therefore be maintained by the non-colluding PSP. To ensure unlinkability between the encrypted advertisement retrieved from the PSP after the auction and the values submitted prior to the auction, the PSP performs re-randomization of the encrypted advertisement on line 30. However, the PSP should get no information about the user's identity in the bidding or auction phase, as such information could be used in combination with response predictions from the bidding phase to infer the degree to which the user is interested in a particular topic, as described in Section 5.2.3. The PSP should therefore perform the randomization of the advertisement without access to the user's public key, which can be achieved using universal re-encryption [47]. Finally, the encrypted advertisement and the group descriptor, as well as the bid request identifier v, are sent via the ad exchange to the user, who decrypts and displays the advertisement.

Since the mappings $r_k \to \left([\![a_k]\!]_u, \Gamma_i\right)$ are sent via the ad exchange during the bidding phase, and we assume the PSP not to collude with any party, the PSP cannot match values r_k to individual DSPs, nor to campaigns. Revealing the value r^{max} to the PSP therefore gives the PSP no more information than the DSP group of the winner of the auction. We consider the winning DSP group not to be private information, as a DSP group is assumed to be responsible for a large, heterogeneous set of campaigns, from which no user interests can be inferred. The winning campaign is thus not revealed to any party.

6.2.5 *Model update phase*

During the model update phase, the response prediction model associated with the shown advertisement is updated using the update rule from Eq. (2.5). The update rule combines the observed user response y, the predicted response $\hat{y}$, the learning rate η and the user profile $\boldsymbol{x}$ to calculate an update gradient. The profile privacy and model privacy requirements dictate that each of these values is private to either the user, the responsible DSP, or both. Moreover, the campaign identifier k must not be linkable to the user identifier u as linking these values would reveal which advertisement was shown to which user, yet k must be revealed in order to select the correct model parameter vector to be updated, and u must be revealed in order to select the correct user profile.

In the model update phase of AHEad, the user is responsible for computing update gradients under encryption. These gradients are sent to the PSP, via the ad exchange acting as an anonymizing proxy, and aggregated per campaign before being revealed to the DSP. User-computed gradients, however, place a heavy computational and communicational burden on the user. Furthermore, the approach of the user being the sole party computing update gradients cannot easily be adapted to the computationally less expensive secret sharing setting without revealing the value $\hat{y}$ to the user, yet $\hat{y}$ is considered private to the DSP as per the model privacy requirement. Therefore, the model update phase of BAdASS shifts the computational burden of computing update gradients from the user to the DSP group responsible for the shown advertisement.

In order to ensure unlinkability between users and campaigns, the model update protocol is split into three stages. During the first stage, the user identifier is revealed to the DSP group responsible for the shown advertisement, which calculates shares of the update gradient $\boldsymbol{g} = \eta(\hat{y} - y)\boldsymbol{x}$. In the second stage, each DSP submits a set of multiple gradient shares to the PSP, which mixes the received shares via random rotation. The PSP then re-shares the set of gradient shares among the DSP group. In the final stage, the campaign identifiers of the set of gradients are revealed to the DSP group, allowing the DSP group to apply the gradients calculated in the first stage to the correct parameter vector. Since the gradient shares have been mixed, the DSP group cannot link values revealed in the third phase to values revealed in the first phase.

The model update protocol is outlined in Protocol 6.4. The protocol is initiated by a user u, who reports shares of their response y directly to the responsible DSP group based on the group descriptor Γ_i received along with the advertisement. Moreover, the user sends the bid request identifier v and the user identifier u to Γ_i. In the first phase, the update preparation phase, each DSP retrieves their shares of b, $\hat{y}$, η, and k, stored during the auction phase, based on the bid request identifier v. Using these shares, the DSP group calculates shares of $\delta = \eta(\hat{y} - y)$. Since this calculation involves multiplication of real values, encoded as fixed-point numbers, the result must be scaled back to the appropriate fixed-point representation, using e. g. the truncation protocol outlined by Catrina and Saxena [23]. The res-

Protocol 6.4 Model update protocol of BAdASS, invoked by the user at the DSP group responsible for the displayed advertisement.

```
1: procedure DSP:PREPARE-MODEL-UPDATE($v, u, \Gamma_i, \langle y \rangle$)
2:     $(\langle b \rangle, \langle \hat{y} \rangle, \langle \eta \rangle, \langle k \rangle) \leftarrow$ lookup $v$
3:     $\langle \delta \rangle \leftarrow$ TRUNCATE($\langle \eta \rangle \cdot (\langle \hat{y} \rangle - \langle y \rangle)$)
4:     for $i \leftarrow 1, d$ do
5:         $\langle g_i \rangle \leftarrow \langle x_{u,i} \rangle \cdot \langle \delta \rangle$
6:     end for
7:     $\langle G \rangle \leftarrow \langle G \rangle \cup \langle \boldsymbol{g} \rangle$
8:     $\langle B \rangle \leftarrow \langle B \rangle \cup \langle b \rangle$
9:     $\langle K \rangle \leftarrow \langle K \rangle \cup \langle k \rangle$
10:    if sufficient values are accumulated then
11:        for all $\gamma_{i,j} \in \Gamma_i$ do
12:            Pick random $r_{i,j}$
13:            Rotate $(\langle G \rangle_{i,j}, \langle B \rangle_{i,j}, \langle K \rangle_{i,j})$ $r_{i,j}$ times
14:        end for
15:        invoke PSP:MIX-SHARES($\Gamma_i, \langle G \rangle, \langle B \rangle, \langle K \rangle$) at PSP
16:    end if
17: end procedure

18: procedure PSP:MIX-SHARES($\Gamma_i, \langle G \rangle, \langle B \rangle, \langle K \rangle$)
19:     if shares from all $\gamma_{i,j} \in \Gamma_i$ have been received then
20:         Pick random $r$
21:         Rotate $(\langle G \rangle, \langle B \rangle, \langle K \rangle)$ $r$ times
22:         $\langle G' \rangle \leftarrow$ re-share $\langle G \rangle$
23:         $\langle B' \rangle \leftarrow$ re-share $\langle B \rangle$
24:         $\langle K' \rangle \leftarrow$ re-share $\langle K \rangle$
25:         invoke UPDATE-MODEL($\langle G' \rangle, \langle B' \rangle, \langle K' \rangle$) at $\Gamma_i$
26:     end if
27: end procedure

28: procedure DSP:UPDATE-MODEL($\langle G' \rangle, \langle B' \rangle, \langle K' \rangle$)
29:     for all $\gamma_{i,j} \in \Gamma_i$ do
30:         Rotate $(\langle G' \rangle_{i,j}, \langle B' \rangle_{i,j}, \langle K' \rangle_{i,j})$ back $r_{i,j}$ times
31:     end for
32:     $K \leftarrow$ combine $\langle K' \rangle$
33:     for all $\langle \boldsymbol{g}' \rangle \in \langle G' \rangle$, $\langle b' \rangle \in \langle B' \rangle$, $k \in K$ do
34:         for $i \leftarrow 1, d$ do
35:             $\langle w_{k,i} \rangle \leftarrow \langle w_{k,i} \rangle - \langle g'_i \rangle$
36:         end for
37:         $\langle \tilde{b}_k \rangle \leftarrow \langle \tilde{b}_k \rangle + \langle b' \rangle$
38:     end for
39: end procedure
```

ulting value δ is multiplied with each element of the user profile to form the update gradient. Since new sharings of the update gradient will be created by the DSP in the mixing step, the d multiplications that are required to calculate the update phase can be performed locally, without reducing the degree of the result. The update gradient shares, as well as the bid value shares and the campaign identifier shares, are locally stored as lists $\langle G \rangle$, $\langle B \rangle$, and $\langle K \rangle$ until sufficient values are aggregated for mixing.

When sufficient values are accumulated by a DSP group Γ_i, the shares of all DSPs in Γ_i are sent to the PSP for mixing. To prevent recombination of shares by the PSP, each DSP $\gamma_{i,j}$ in the group Γ_i picks a random number $r_{i,j}$, and rotates their lists of shares $\langle G \rangle_{i,j}$, $\langle B \rangle_{i,j}$, and $\langle K \rangle_{i,j}$ $r_{i,j}$ times. Given a sufficiently large aggregation threshold, the average number of attempts needed for the PSP to successfully combine the received shares becomes prohibitively large. The PSP subsequently picks a random number r_{PSP}, and rotates each of the lists of shares r_{PSP} times, such that the positions of output values cannot be linked to the positions of input values.

The share values themselves need also be randomized before being sent back to the DSPs to prevent linking input values to output values. Randomization of a secret-shared value $\langle s \rangle$ is typically performed by generating a new random sharing $\langle 0 \rangle$, and adding the generated shares to $\langle s \rangle$. Since the PSP does not know which received shares belong to the same value, however, such randomization cannot be performed by the PSP. Instead, the PSP splits each received share $\langle s \rangle_i$ into a new sharing $\langle s \rangle_{i,1}, \ldots, \langle s \rangle_{i,n}$. These subshares are distributed among the DSPs in Γ_i in a manner analogous to the degree reduction step described in Section 5.1, such that each party j receives subshares $\langle s \rangle_{1,j}, \ldots, \langle s \rangle_{n,j}$. The DSPs then recombine the received subshares to obtain new shares of the original values $\langle s \rangle_i$.

The PSP cannot perform recombination of shares by itself due to the random rotation of input lists, such that the PSP does not know which of the shares belong to the same value. Therefore, each DSP is given lists of the subshares created by the PSP, where a subshare originating from a share submitted by DSP $\gamma_{i,j}$ is rotated $r_{i,j} + r_{PSP}$ times. To match subshares originating from the same value, each DSP rotates the subshares originating from DSP $\gamma_{i,j}$ back $r_{i,j}$ times, such that subshares of the same value are in the same position in the lists. After recombining the rotated subshares, each DSP has lists $\langle G' \rangle$, $\langle B' \rangle$, and $\langle K' \rangle$, where each of the lists contains shares of the same values as were submitted to the PSP, rotated r_{PSP} times.

The DSPs proceed to combine $\langle K' \rangle$ to reveal the list of campaign identifiers to which the gradient and bid shares belong. Due to the mixing of the lists, DSPs cannot link the campaign identifiers revealed in the third phase to user identifiers revealed in the first phase, provided a sufficiently large mixing threshold is chosen. Finally, each

DSP locally updates their shares of the model parameter vectors with the list of gradient shares, and adds the received bid shares to the corresponding bid value aggregates.

6.3 PERFORMANCE ANALYSIS

To evaluate the performance of BAdASS, we provide both a theoretical analysis of the computational and communication complexities of the subprotocols, and a set of measurements obtained from a proof-of-concept implementation. The theoretical analysis provides an overview of the performance impact incurred by the use of secret sharing for each of the parties, whereas measurements from the implementation show the actual performance of the protocol, taking into account all required computations.

6.3.1 *Computational complexity*

To analyse the computational complexity and scalability of BAdASS, we list the number of costly operations performed by each party per invocation of every subprotocol. We consider operations that require communication between parties, such as multiplications, reconstruction of a secret value (if the reconstruction requires an additional round of communication), and invocations of the comparison and truncation protocols, to be costly operations. Moreover, we list the re-sharing of shares by the PSP in the model update phase as a costly operation, as it is essentially the same as the degree reduction step of the multiplication protocol. Finally, we count the number of cryptographic operations performed by the protocols. Computations that are performed locally, such as homomorphic additions and local multiplications without degree reduction, are not considered in the complexity analysis.

The computational complexity of BAdASS depends on a number of variables, which are described in Table 6.2. Note that the dimensionality of user profiles d is expected to be much larger than the number of campaigns K and the number of DSPs n. Table 6.3 lists the amortized number of operations performed by each party for each subprotocol. The profile update protocol does not make use of any expensive subprotocols, as it consists only of distributing shares of the user profile, and local additions of shares. Note, however, that the user creates d additive shares, and the ad exchange and PSP both create d shamir shares. In the bidding protocol, each DSP performs a multiplication for every campaign within its DSP group to calculate the bid value, and an encryption of the advertisement for all its own campaigns. Calculation of shares of the inner product $\langle \boldsymbol{w}^\top \boldsymbol{x} \rangle$ is performed locally, as explained in Section 6.2.3, and since the resulting value is reconstructed by the PSP, no degree reduction step is neces-

SYMBOL	DESCRIPTION
d	Dimensionality of user profiles.
n	Number of DSPs.
g	Number of DSP groups.
m	Size of a DSP group.
κ	Number of campaigns of a DSP.
K	Total number of campaigns.
K_i	Number of campaigns within a DSP group Γ_i.
ζ	Number of model updates accumulated per DSP group.
σ	Size in bits of a secret share.
ξ	Size in bits of an advertisement tuple, consisting of an encrypted advertisement, a group descriptor, and a random identifier.
λ	Total number of comparisons required to find the maximum bid. Equal to $\lceil \log_2 (K_i - 1) \rceil + \lceil \log_2 (g - 1) \rceil$.
ρ	Number of rounds required by a single run of the comparison protocol.
γ	Number of bits transmitted in a single run of the comparison protocol.
T	Number of rounds required by a single run of the truncation protocol.
τ	Number of bits transmitted in a single run of the truncation protocol.

Table 6.2: Symbols used in the computational analysis of BAdASS.

sary. The multiplications involved in calculating the inner product are thus 'free'.

In the first step of the auction protocol, each DSP group Γ_i loops through the MAX-BID procedure $K_i - 1$ times, where each loop invokes the comparison protocol once, and the multiplication protocol 5 times to obtain shares of the maximum values (as described in Section 6.2.4, each sum of two multiplication requires a single invocation of the degree reduction step). The second step of the auction protocol performs the same loop g times, where g is the number of DSP groups, but with 2 multiplications instead of 5. The second step, however, is performed by any single DSP group only once for every g auctions, as the ad exchange picks a random DSP group to perform the global auction for every ad request. The amortized cost of the global auction for a DSP group is thus 1 comparison and 2 multiplications per invocation of the auction protocol.

PROTOCOL	OPERATION	USER	ADX	DSP	PSP
Profiling	Sharing	d	d		d
Bidding	Multiplication			K_i	
	Encryption			κ	
	Reconstruction				K
Auction	Comparison			K_i	
	Multiplication			$5K_i + 2$	
	Reconstruction				1
	Re-encryption				1
	Decryption	1			
Update	Multiplication			1	
	Truncation			1	
	Re-sharing				$(d+2)m$
	Reconstruction			1	

Table 6.3: Number of invocations of operations performed per invocation of each subprotocol of BAdASS.

In the model update protocol, a DSP group performs 1 multiplication and 1 truncation to obtain shares of δ, and subsequently performs d local multiplications to compute the update gradient. As explained in Section 6.2.5, the d multiplications need no degree reduction step, as the PSP creates new shares of the values in the mixing step, and are thus 'free'. The mixing step is performed once every ζ invocations of the model update protocol, processing ζ updates at once. The amortized cost of the mixing step is thus equal to the cost of processing a single update. To process a single update, the PSP re-shares all m shares of the d-dimensional update gradient, as well as m shares of the bid value and m shares of the campaign identifier. Note that each DSP needs to perform $d+2$ local recombinations of the created subshares, which are not listed in the table. Moreover, the DSPs need to perform a single reconstruction to obtain the campaign identifier.

The computational complexity of the profile update protocol is $O(d)$, i. e. linear in the profile size d. Note that the number of shares created by the ad exchange and DSP depends on the number of DSPs that receive the profile udpate. The computational complexity of the bidding protocol is $O(K)$, i. e. linear in the total number of campaigns due to its reliance on the PSP to calculate the user response estimation for every campaign. Note, however, that the number of computations performed locally by the DSPs is dependent on the profile size d. The amortized complexity of the auction protocol is $O(K_i)$, i. e. linear in

PROTOCOL	ROUNDS	USER	ADX	DSP	PSP
Profiling	2	$2d\sigma$	$dm\sigma$		$dm\sigma$
Bidding	2		$K\xi$	$(m+1)\,K_i\sigma + \kappa\xi$	$Km\sigma$
Auction	$\lambda(\rho+1)+3$		ξ	$(5K_i-3)\,m\sigma + K_i\gamma + 2\frac{1}{g}\sigma$	ξ
Update	$T+1\frac{3}{\zeta}$	$m\sigma$		$(d+1)m\sigma + \tau + (d+3)\sigma$	$(d+2)m^2\sigma$

Table 6.4: Communication bandwidth in bits and number of rounds of communication per invocation of each subprotocol of BAdASS.

the number of campaigns run within a DSP group, and even the number of local computations is independent of the profile size d. Finally, the amortized complexity of the model update phase is $O(dm)$ due to the re-sharing of every share received by the PSP. The group size m, however, is bounded by $4t-3$, since any group of size $4t-2$ can be split into two groups of size $2t-1$. Therefore, m can be considered a small constant, resulting in a computational complexity linear in the profile size.

6.3.2 *Communication complexity*

We analyse the communication complexity of BAdASS by listing the number of bits transmitted by each party in a single invocation of every subprotocol. Moreover, we list the number of rounds of communication required by every subprotocol. The variables used in the description of the communication complexity are listed in Table 6.2. The number of rounds and number of transmitted bits for the auction and model update protocols depend on implementations of the secure comparison and truncation protocols, and are thus expressed using variables describing the complexities of these protocols.

Table 6.4 lists the amortized number of bits transmitted by each party for each subprotocol, as well as the number of rounds of communication required by each subprotocol. The round complexities of the profile update and bidding protocols are constant. Since the group size m is bounded by the constant $4t-3$, and the share size σ is constant, the communication complexity of the profile update phase can be considered linear with respect to the profile size d. Note that if the user profile is distributed among multiple DSP groups, the complexity of the user profiling phase becomes multilinear in the profile size and the number of DSPs. During the bidding phase, the ad exchange acts as a proxy to transmit a total of K advertisement tuples, and the PSP transmits K sharings of the estimated user response. Each DSP performs K_i multiplications, each requiring m shares to be transmitted, sends K_i shares of inner products to the PSP, and sends an advertisement tuple for each of its κ own campaigns to the PSP via the ad exchange. The total communication complexity of the bidding phase is thus $O(K)$, or linear in the number of campaigns.

The auction protocol consists of a number of comparisons logarithmic with respect to the number of campaigns within a group and the number of groups. Note that the total number of campaigns K is bounded by $K^{max}g$, where K_i^{max} is the maximum of the number of campaigns of all groups. The round complexity of the auction phase is thus $O(\log_2 K)$, or logarithmic in the number of campaigns, provided the round complexity of the comparison protocol is constant. Since each DSP performs an average of K_i comparisons per invocation of the auction protocol and transmit a fixed number of shares for each multiplication, the communication complexity of the auction protocol is linear in the number of campaigns, provided the communication complexity of the comparison protocol is constant.

The model update protocol contains a single invocation of the truncation protocol, of which the number of rounds is considered constant, as well as one round of multiplication. Every ζ invocations, three more rounds for mixing and combining are performed. The amortized round complexity of the model update protocol is therefore constant. Although the amount of bits transmitted by the PSP contains a factor m^2, we can assume this to be a small constant due to the small upper bound on m. The average communication complexity of the is $O(d)$, provided the communication complexity of the truncation protocol is constant and the group size m is bounded by a constant.

6.3.3 *Implementation*

To measure the runtime of BAdASS, we made a proof-of-concept implementation of the protocol in C++. We use a custom implementation of secret sharing protocols, and an implementation of ElGamal and universal re-encryption based on the *SeComLib* library. The secure comparison protocol is based on a simplified version of the protocol by Reistad and Toft [88] as implemented in VIFF[1] [44]. Real values, such as model weights, are represented as 16-bit fixed-point numbers. All operations using Shamir shares are performed in a prime field of order $p = 2^{31} - 1$, such that share values can be represented as 32-bit integers, and two fixed-point values can be multiplied without causing an overflow. The reconstruction threshold t is set to 3, resulting in a DSP group size m of 5. The key length for the ElGamal

1 Virtual Ideal Functionality Framework. Available online at https://github.com/kljensen/viff/blob/master/viff/comparison.py

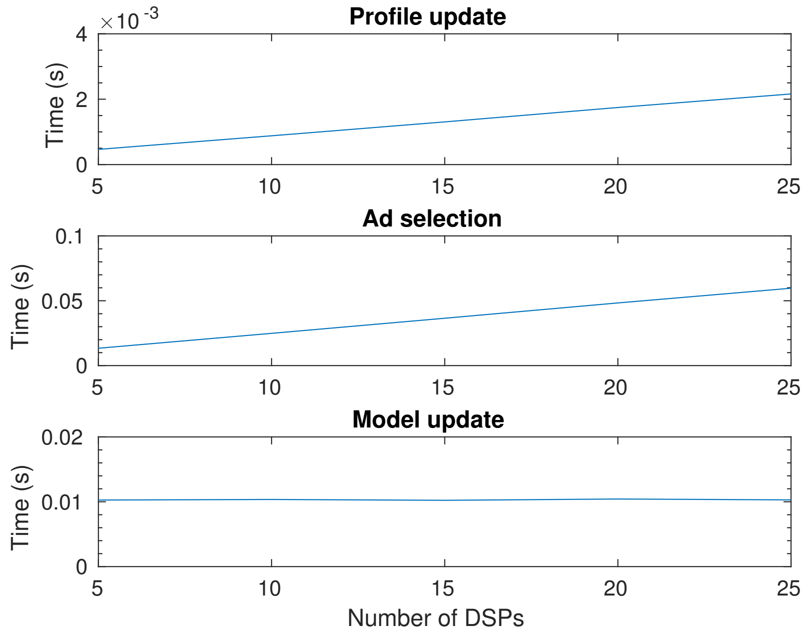

Figure 6.1: Total computation time required by a single run of each subprotocol of BAdASS for an increasing number of DSPs, with a fixed profile size $d = 2^{12}$ and number of campaigns per DSP $\kappa = 10$. Model update time is averaged using $\zeta = 10$.

cryptosystem is set to 2048 bits to achieve a sufficiently high security level[2].

The implementation simulates all parties within a single process thread, thus performing all operations sequentially. The measurements of computation time therefore do not take communication overhead into account. However, they do show the computational cost of all calculations, including invocations of subprotocols such as the multiplication protocol and the comparison protocol, as well as all local computations.

The runtime tests were executed on a mobile workstation running Arch Linux on an Intel® Core™ i7-3610QM 2.3 GHz quad-core processor with 8 GB RAM. Fig. 6.1 shows the impact of the number of DSPs, and thus the total number of campaigns, on the total computation time, using a fixed profile size and number of campaigns per DSP. The time spent in the profile update protocol grows linearly with the number of DSPs due to the profile update being processed by an increasing number of DSPs. The computation time per DSP, however, is independent of the number of DSPs, as are the computation times for the user. Only the ad exchange and the PSP need to perform more computations when the number of DSPs increases, as more secret shares need to be created. Although the profile update

2 See e. g. `https://www.keylength.com` for key lengths as recommended by various organizations. The NIST considers a key length of 2048 sufficiently secure until 2030.

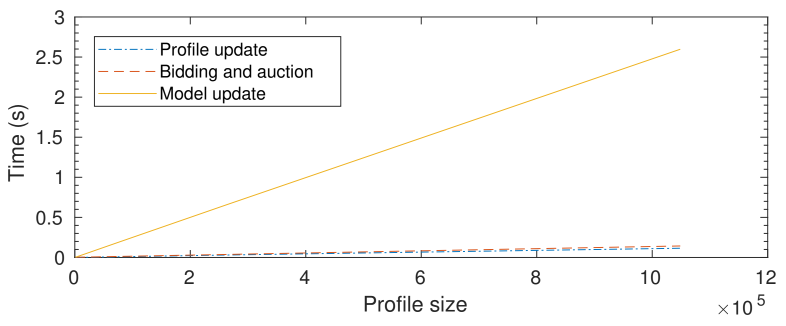

Figure 6.2: Total computation time required by a single run of each subprotocol of BAdASS for an increasing profile size d, with a single DSP group running a total of 5 campaigns. Model update time is averaged using $\zeta = 10$.

phase cannot be fully parallelized due to the central role of the ad exchange and PSP, a large part of the computation will be performed in parallel by independent DSP groups in a realistic setting.

The bidding and auction protocols, combined in the middle graph in Fig. 6.1, scale linearly with the number of DSPs. As in the profile update protocol, most of the computation can be parallelized across DSP groups, such that computing the inner product, the bid value, and the maximum bid within a group are all independent of the number of DSP groups. However, the computation of the sigmoid function for each of the campaigns is performed by a single party. Therefore, the bidding protocol scales linearly in the number of campaigns and, given a fixed number of campaigns per DSP, linearly in the number of DSPs. The number of values compared in the global auction depends on the number of DSP groups, and therefore scales linearly in the number of DSPs. Since the model update protocol is performed once for every shown advertisement, and the computations are performed jointly by a single DSP group and the PSP, its runtime is independent of the number of DSPs.

Fig. 6.2 shows the impact of profile dimensionality on the total computation time, given a fixed number of DSP groups and a fixed number of campaigns per DSP. The shown times are the combined computation times of all parties, with a single DSP group consisting of five DSPs. The computation time required by the model update protocol far exceeds that of the profile update, bidding, and auction protocols, due to the large number of subshare recombinations performed by the DSPs as well as the large number of sharings created by the PSP. Based on our measurements, we estimate that if the computations performed by the DSPs are parallelized, the average time spent on the model update protocol for a profile dimensionality of $d = 2^{20}$ drops from 2.6 seconds, as shown in Fig. 6.2, to about 750 ms per invocation. Note that the computation time is averaged, as the expensive mixing and update steps are only performed once every

PROTOCOL	ROUNDS	USER	ADX	DSP	PSP
Profiling	2	8192	20 480		20 480
Bidding	2		51	6.25	2
Auction	$\lambda(\rho+1)+3$		0.5	$5+K_i\gamma$	0.5
Update	$T+1\frac{3}{\zeta}$	0.02		$24\,576+\tau$	102 400

Table 6.5: Bandwidth usage of BAdASS in KiB per invocation of each subprotocol, based on realistic parameters used in our implementation of BAdASS.

$\zeta = 10$ invocations of the model update protocol. The computation times of the bidding and auction phase and the profile update phase are both below 150 ms for large profile sizes, even when DSPs are simulated sequentially, and thus seem very well suited for use in a real-time setting as required by the RTB advertising model. The relatively large amount of computation performed in the model update phase is less time sensitive, and can thus be periodically performed as a background task without harming the user experience.

In Section 6.3.2 we listed the average communication complexity of each of the subprotocols of BAdASS, and showed that the bandwidth consumption of the model update and profile update protocols far exceeds that of the bidding and auction protocols. In Table 6.5, we list the average communication bandwidth in kibibytes required by the BAdASS protocols for specific parameters used in our implementation. We use a share size $\sigma = 32$ bits, and a group size of $m = 5$ DSPs. Each DSP is responsible for $\kappa = 10$ campaigns, and with two groups the total number of campaigns is $K = 100$. The profile dimensionality is $d = 2^{20}$, and the size of an advertisement descriptor is assumed to be 4160 bits, of which 4096 bits are the encrypted advertisement, 32 bits the random identifier, and 32 bits the group descriptor. From the table, it is evident that the profile update and model update require a significant amount of communication, with up to 100 MiB per invocation, on average, of the model update protocol. The time-sensitive bidding and auction protocols, however, require very little bandwidth. Moreover, very little bandwidth is used by the user, with only the periodically executed profile update protocol requiring more than a few dozen bytes at 8 MiB per invocation, making the bandwidth use of the user very acceptable for modern unmetered connections.

6.4 DISCUSSION

With BAdASS, we have shown that privacy can be preserved in OBA at the cost of minimal user-noticeable delays. Trust is distributed among

DSPs using threshold secret sharing, such that any group of colluding DSPs smaller than the reconstruction threshold cannot obtain any sensitive information. DSPs collaboratively compute bid prices within the secret-shared domain and use a hierarchical secure auction to determine the highest bid, without any knowledge of a user's interests. Reports of individual clicks and views of advertisements are secret-shared among DSPs, where they are used to privately update model parameters via a mixing step at the PSP.

BAdASS ensures profile privacy and model privacy by distributing sensitive information, and thus trust, among parties. The contents of the user profile, the shown advertisement, and the actual user response are not revealed to any party other than the user themselves, and campaign-specific model parameters are not revealed to any party other than the DSP responsible for the campaign. Individual bid prices are not revealed to any party, but are aggregated for billing purposes. The predicted user response, although revealed to the PSP, cannot be linked to individual users or campaigns. The mixing step in the model update protocol allows the response prediction models used by the DSPs to be trained using observed user responses, without disclosing the value of the response or the shown advertisement, such that the predictive performance of the models can be improved over time. Finally, the protocols are integrated into the RTB setting by forming DSP groups from existing DSPs, with the addition of a single new party.

Although the presented model update protocol requires a total of 200 MiB of communication per shown advertisement, as well as an estimated 750 ms of computation time, on average, on our modest hardware, the computational performance of BAdASS as a whole is promising. Simulated sequentially on commodity hardware, the profile update, bidding, and auction protocols complete in a fraction of a second even for a large profile dimensionality $d = 2^{20}$. Most computations can be parallelized across DSPs or DSP groups, and operations that cannot be parallelized are at most linear in the number of campaigns or the profile size. Moreover, the communication complexities of the bidding and auction protocols are very modest, and although the profile update protocol has a communication complexity linear in d, resulting in the transmisssion of 48 MiB of data for $d = 2^{20}$, it can be invoked infrequently, such that the bandwidth use remains acceptable.

7 DISCUSSION AND FUTURE WORK

The multi-billion dollar online advertising industry has helped shape the ad-supported web as we know it today. Privacy concerns raised over modern behavioural advertising practices, however, drive users to adopt advertisement blocking tools, posing a threat to the business models of publishers and advertising companies. The resulting ad blocking arms race shows that, in order to maintain the ad-supported web, privacy concerns of users must be addressed while maintaining the profitability of OBA.

Several approaches to enhance user privacy in OBA have been suggested in previous work, but none of the previously proposed solutions succeed in achieving user privacy in the current online advertising landscape without harming the economic benefits of OBA. Previous work does not take into account the complexity of the online advertising landscape, precludes learning preference patterns from individual responses, or severely limits the available targeting information. In this research, we demonstrate the feasibility of behavioural targeting of advertisements without disclosing users' interests by focussing on the following research question:

> *How can an online advertising system serve advertisements tailored to users' interests in the Real-Time Bidding model, such that no party other than the user themselves can gain any knowledge of the users' interests, without degrading the system's responsiveness perceived by users?*

In this chapter, we return to the research question and discuss how the two presented protocols achieve the research goal. Moreover, we provide future research directions by identifying remaining open problems and improvements.

7.1 DISCUSSION

In this thesis, two complete privacy-preserving protocols for OBA are presented. Both protocols combine a machine learning method commonly encountered in existing advertising systems with secure multi-party computation techniques, allowing the system to learn from observed user responses while maintaining both profile privacy and model privacy. In the first protocol, AHEad, a threshold variant of the additively homomorphic Paillier cryptosystem is used to distribute trust between DSPs, the ad exchange, and the PSP. Each DSP performs computations on its own advertising campaigns in cooperation with the PSP, using encrypted user profiles to ensure profile privacy. In the second protocol, BAdASS, the additively homomorphic Shamir secret sharing scheme is used to distribute trust between DSPs. DSPs are clustered into small groups, within which they collaboratively perform computations on each other's campaigns, together with the PSP, using secret-shared user profiles and campaign parameters to ensure profile privacy and model privacy.

Contrary to some of the prior art described in Chapter 3, the protocols presented in this thesis do not rely on absolute anonymity and unlinkability of users. Anonymity and unlinkability requirements would require users to distribute a fresh user profile for every requested advertisement, leading to a great increase in both computational and communication costs. Instead, the protocols hide the *behaviour* of users, such that the *identity* of a user is not considered private information. As a consequence, data exchange practices such as cookie matching, described in Section 2.2.2, or the collection and trade of user data by DMPs, can be performed without infringing on user privacy. Because of the format of encrypted or secret-shared user profiles, obtained through feature hashing, data from multiple sources such as DMPs can easily be combined by using the additively homomorphic properties of the underlying encryption or secret sharing scheme, without any party gaining insight into the meaning of the data.

In Section 1.4, we identify five sub-questions of the main research question. The first sub-question focuses on estimating the value of an advertisement for a user, based on the user's past behaviour, without any knowledge of that past behaviour of the user's interests. The presented protocols both solve this problem in the bidding phase, using a logistic regression model to predict the user's response based on encrypted or secret-shared user profiles, followed by a linear bidding function with which the truthful value of an estimated response is calculated. Although the sigmoid function that is used in predicting user responses is computed in the clear by the PSP, we argue that no information about user interests is disclosed to any party. Moreover, no information about campaign parameters is disclosed to any party other than the responsible DSP.

The second sub-question describes the problem of making information about a user's behaviour available for the purpose of advertisement value estimation, without disclosing the meaning of that information. The two protocols both rely on locally generated user profiles, as described in previous work, which are distributed to DSPs in a privacy-preserving manner during the user profiling phase. Because user profiles are prepared for direct use by the privacy-preserving machine learning model employed for user response estimation within the user's web browser, the DSPs need no knowledge of the meaning of the encrypted or secret-shared information.

In the third sub-question, we inquire how the privacy-preserving response estimation model can be trained with observed user responses, without disclosing the value of the response. In AHEad, the model update phase requires users to compute encrypted update gradients, which are sent to the PSP via the ad exchange, which acts as an anonymizing proxy. The PSP aggregates updates of multiple users on a per-campaign basis before revealing the aggregated gradients to the DSP, which updates its model parameters in the clear. In BAdASS, the user secret-shares the value of their response among the DSP group, which computes the update gradient using secret-shared model parameters. To ensure profile privacy, the link between user and campaign is broken by mixing complete update gradients, the bid value and the campaign identifier at the PSP. In both protocols, neither the user response nor the link between user and campaign are disclosed to any party other than the user.

The fourth sub-question focuses on integration of the privacy-preserving value estimation model into the RTB setting, such that advertisement auctions reveal no information about users' interests. Both protocols serve advertisements in encrypted form, such that the contents of advertisements are not revealed during the auction process. The highest bid value is determined using a secure comparison protocol, and the campaigns associated with encrypted advertisements are hidden by randomizing identifiers. Therefore, no party learns which campaign is the winner of the auction, nor which advertisement is shown to which user.

The fifth and final sub-question investigates the efficiency of the proposed protocols. Both protocols achieve performance multi-linear in the size of user profiles and the number of advertising campaigns. However, the two protocols differ significantly in terms of computational and communicational complexity. AHEad, based on homomorphic encryption, requires more than 100 seconds of computation time to calculate a single bid value using a profile dimensionality of 2^{20} on our modest hardware. BAdASS, on the other hand, simulates the calculation of five bid values by five DSPs in less than 150 milliseconds on a single processor core. Although model updates are expensive in both protocols, the model update phase of BAdASS requires more than 20 times less computation time than that of AHEad. In terms of computational complexity, BAdASS is not only much more efficient than AHEad, it is even efficient enough to serve advertisements in real time as required by the RTB model, provided the communication between DSPs incurs minimal latency.

In terms of communication, both protocols have considerable bandwidth requirements due to the transmission of complete user profiles and update gradients. In AHEad, the PSP must transmit 500 MiB per DSP for every invocation of the profile update protocol with a profile dimensionality of $d = 2^{20}$. In BAdASS, on the other hand, the model update protocol requires the transmission of 100 MiB by the PSP for every shown advertisement. Although the bidding and auction phase require significantly less bandwidth, the prohibitive communication complexity of both protocols leaves opportunity for further improvements. Notwithstanding, the results obtained with BAdASS show that by collaboratively computing predictions using secret sharing techniques, a level of performance can be achieved in the time-sensitive bidding and auction phases that does not degrade the responsiveness perceived by the user.

7.2 FUTURE WORK

The presented protocols are, to the best of our knowledge, the first protocols that combine machine learning models with secure multi-party computation techniques to preserve privacy in OBA. The two protocols, and BAdASS in particular, show promising results in terms of privacy and performance. There is, however, ample opportunity for improvement of the protocols.

PERFORMANCE OF MODEL UPDATES Although the performance achieved by BAdASS is significantly better than that of AHEad, the model update protocol, invoked for every shown advertisement, requires the transmission of multiple secret shares, as well as the generation of new sharings and recombinations of secrets, for every dimension of the user profile. These computations are performed to break the link between the user identifier, needed to select the correct user profile, and the campaign identifier, needed to select the model parameters to update. The resulting performance of the model update protocol, however, is unsatisfactory for use in a real-time application. Therefore, further performance improvements of the model update protocol are necessary.

A possible direction for improvement is an ORAM-based storage for model parameters, allowing DSPs to update model parameters without knowing the campaign identifier that is currently needed to select the correct model parameters. Similar to the current mixing-based approach, however, ORAM-based parameter storage requires the transmission of complete parameter vectors between the storage and the processor, as well as re-randomization of shares to avoid linkability between updates.

MALICIOUS BEHAVIOUR A crucial assumption made in the design of the two protocols is that all parties are semi-honest, i.e. they are assumed to follow the protocol description, but attempt to learn as much sensitive information as possible. In a real-world scenario, in particular in the case of BAdASS, parties engaging in the protocols may have incentives to act maliciously and deviate from the protocol description. DSPs, for instance, may attempt to modify the bid values submitted on behalf of their competitors in order to gain an unfair advantage. Likewise, DSPs may attempt to corrupt model parameters of their competitors in the model update phase, or may report invalid bid price aggregates. Users, too, may act maliciously by submitting invalid profiles or responses in order to corrupt model parameters. Although malicious behaviour can be discouraged via regulation established by ad exchanges, detection of cheating parties is difficult due privacy requirements preventing disclosure of most values used in the protocol.

In future work, privacy-preserving OBA protocols should be adapted to a malicious setting by including auxiliary proofs and integrity checks with which the integrity of computations can be ascertained and cheating parties can be detected. Such adaptation, however, is faced with the same efficiency challenges as the current research due to high profile dimensionalities. The performance impact of proofs should thus be minimized, calling for a specialized solution rather than naïve applications of existing verifiable secret sharing schemes.

BIDDING FUNCTIONS AND AUCTIONS Some bidders may employ non-truthful bidding strategies based on historical observations of winning bid prices, as described in Section 2.1. In the proposed protocols, we considered only linear bidding functions with predefined parameters, whereas DSPs may employ more complex bidding functions in practice. Adoption of non-linear or data-driven bidding strategies in a privacy-preserving bidding protocol, however, is a challenge on its own due to privacy requirements. Individual bid prices should not be revealed to any party, nor should the winning DSP learn when it wins an advertisement auction. A future research direction is therefore to investigate the integration of non-linear, data-driven bidding strategies into a privacy-preserving bidding protocol.

PARAMETER TUNING AND OPTIMIZATION The response prediction models used in AHEad and BAdASS are generally considered to be capable of achieving competitive prediction performance, provided suitable parameters are chosen. Tuning the parameters to achieve competitive performance, however, requires insight into the predictive performance of the model, as well as access to sufficient training data. To gain insight into the predictive performance of the model, however, one needs access to both predicted responses and observed responses, such that a performance metric can be calculated based on the differences between predictions and observations. Due to privacy requirements, AHEad and BAdASS do not disclose click predictions or observed responses.

An additional challenge to parameter tuning is that in BAdASS, individual parties cannot obtain complete training datasets due to the use of secret sharing. DSPs could choose to tune model parameters in the live system, but may be reluctant to experiment on ongoing campaigns. Alternatively, DSPs within a DSP group could collaboratively tune model parameters using stored historic observations as training data, but given the computational cost of parameter tuning, DSPs may have little incentive to perform computations only to improve the prediction models of their competitors. In future work, the problem of parameter tuning in privacy-preserving machine learning models should be investigated in order to alleviate objections that data processors might have to adopting data encryption.

7.3 CONCLUDING REMARKS

The objective of this research has been to alleviate privacy concerns caused by the use of sensitive personal information for advertisement selection, while allowing behavioural targeting of advertisements. Although several approaches to enhance privacy in OBA have been suggested, previous work is unable to achieve user privacy in the current online advertising landscape due to the use of overly generalized user profiles, underestimations of the complexity of the advertising landscape and ad selection models, or an inability to learn preference patterns from individual responses.

The two protocols presented in this thesis achieve the research objective by distributing trust, previously placed in a single party, among a set of parties. The first presented protocol, based on a threshold cryptosystem, suffers from considerable computational costs, particularly during the highly time-sensitive bidding phase. However, the second protocol, based on a secret-sharing scheme, achieves a level of efficiency that allows computing predictions and bids for individual users within a fraction of a second, as demanded by the RTB mechanism of trading advertisement impressions. These findings suggest that the online advertising ecosystem could harmonize demands for privacy with the economic benefits of behavioural advertising through the adoption of cryptographic tools to protect user information, without introducing significant delays in the advertisement serving process.

The privacy and performance achieved by the BAdASS protocol provide a first step towards adoption of privacy-preserving methods by the online advertising ecosystem. The heavily fragmented shape of the online advertising landscape lends itself particularly well to the use of efficient secret-sharing techniques, giving advertising companies the opportunity to cooperatively move towards acceptable forms of behavioural advertising. Doing so can provide a competitive advantage, as ad-blocking tools may allow advertisements served by companies that respect users' privacy, and publishers that care for the privacy of their visitors may prefer selling advertisement space via ad exchanges that engage in privacy-preserving auctions. The results obtained with BAdASS show that it is possible to serve advertisements tailored to users' interests without disclosing those interests to any party, all within a fraction of a second. Given these results, the online advertising ecosystem as a whole no longer has an excuse not to invest in users' privacy – not only to safeguard their own business, but to save the web as we know it today.

BIBLIOGRAPHY

[1] G. Acar, C. Eubank, S. Englehardt, M. Juarez, A. Narayanan and C. Diaz. 'The Web Never Forgets: Persistent Tracking Mechanisms in the Wild'. In: *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. CCS '14. Nov. 2014, pp. 674–689.

[2] G. Acar, M. Juarez, N. Nikiforakis, C. Diaz, S. Gürses, F. Piessens and B. Preneel. 'FPDetective: Dusting the Web for Fingerprinters'. In: *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*. CCS '13. Nov. 2013, pp. 1129–1140.

[3] A. Agarwal, O. Chapelle, M. Dudík and J. Langford. 'A Reliable Effective Terascale Linear Learning System'. In: *J Mach Learn Res* 15.1 (Jan. 2014), pp. 1111–1133.

[4] C. C. Aggarwal and P. S. Yu. 'A General Survey of Privacy-Preserving Data Mining Models and Algorithms'. In: *Privacy-Preserving Data Mining*. Advances in Database Systems 34. June 2008, pp. 11–52.

[5] A. Ahmed, A. Das and A. J. Smola. 'Scalable Hierarchical Multitask Learning Algorithms for Conversion Optimization in Display Advertising'. In: *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*. WSDM '14. Feb. 2014, pp. 153–162.

[6] A. Ahmed, Y. Low, M. Aly, V. Josifovski and A. J. Smola. 'Scalable Distributed Inference of Dynamic User Interests for Behavioral Targeting'. In: *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '11. Aug. 2011, pp. 114–122.

[7] M. Aly, A. Hatch, V. Josifovski and V. K. Narayanan. 'Web-Scale User Modeling for Targeting'. In: *Proceedings of the 21st International Conference on World Wide Web*. WWW '12 Companion. Apr. 2012, pp. 3–12.

[8] M. Aly, S. Pandey, V. Josifovski and K. Punera. 'Towards a Robust Modeling of Temporal Interest Change Patterns for Behavioral Targeting'. In: *Proceedings of the 22nd International Conference on World Wide Web*. WWW '13. May 2013, pp. 71–82.

[9] M. An. *Why People Block Ads (And What It Means for Marketers and Advertisers)*. HubSpot Research, July 2016. URL: https://research.hubspot.com/reports/why-people-block-ads-and-what-it-means-for-marketers-and-advertisers (visited on 08/03/2017).

[10] E. Androulaki and S. M. Bellovin. 'A Secure and Privacy-Preserving Targeted Ad-System'. In: *Proceedings of the 14th International Conference on Financial Cryptograpy and Data Security*. FC'10. Jan. 2010, pp. 123–135.

[11] Y. Aono, T. Hayashi, L. Trieu Phong and L. Wang. 'Scalable and Secure Logistic Regression via Homomorphic Encryption'. In: *Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy*. CODASPY '16. Mar. 2016, pp. 142–144.

[12] M. Backes, A. Kate, M. Maffei and K. Pecina. 'ObliviAd: Provably Secure and Practical Online Behavioral Advertising'. In: *2012 IEEE Symposium on Security and Privacy*. May 2012, pp. 257–271.

[13] R. Balebako, P. Leon, R. Shay, B. Ur, Y. Wang and L. Cranor. 'Measuring the Effectiveness of Privacy Tools for Limiting Behavioral Advertising'. In: *Web 2.0 Security and Privacy Workshop*. May 2012.

[14] M. A. Bashir, S. Arshad, W. Robertson and C. Wilson. 'Tracing Information Flows Between Ad Exchanges Using Retargeted Ads'. In: *Proceedings of the 25th USENIX Security Symposium*. Aug. 2016, pp. 481–496.

[15] J. Bau, J. Mayer, H. Paskov and J. C. Mitchell. 'A Promising Direction for Web Tracking Countermeasures'. In: *Web 2.0 Security and Privacy Workshop*. May 2013.

[16] M. Ben-Or, S. Goldwasser and A. Wigderson. 'Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation'. In: *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*. 1988, pp. 1–10.

[17] M. Bilenko and M. Richardson. 'Predictive Client-Side Profiles for Personalized Advertising'. In: *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '11. Aug. 2011, pp. 413–421.

[18] D. M. Blei, A. Y. Ng and M. I. Jordan. 'Latent Dirichlet Allocation'. In: *J. Mach. Learn. Res.* 3 (Mar. 2003), pp. 993–1022.

[19] D. Bogdanov. 'How to Securely Perform Computations on Secret-Shared Data'. Thesis. University of Tartu, May 2007.

[20] F. Brunton and H. Nissenbaum. 'Vernacular Resistance to Data Collection and Analysis: A Political Theory of Obfuscation'. In: *First Monday* 16.5 (Apr. 2011).

[21] C. Castelluccia. 'Behavioural Tracking on the Internet: A Technical Perspective'. In: *European Data Protection: In Good Health?* Feb. 2012, pp. 21–33.

[22] C. Castelluccia, M.-A. Kaafar and M.-D. Tran. 'Betrayed by Your Ads!' In: *Privacy Enhancing Technologies*. July 2012, pp. 1–17.

[23] O. Catrina and A. Saxena. 'Secure Computation with Fixed-Point Numbers'. In: *Financial Cryptography and Data Security*. Jan. 2010, pp. 35–50.

[24] O. Chapelle. 'Offline Evaluation of Response Prediction in Online Advertising Auctions'. In: *Proceedings of the 24th International Conference on World Wide Web*. WWW '15 Companion. May 2015, pp. 919–922.

[25] O. Chapelle, E. Manavoglu and R. Rosales. 'Simple and Scalable Response Prediction for Display Advertising'. In: *ACM Trans Intell Syst Technol* 5.4 (Dec. 2014), 61:1–61:34.

[26] D. L. Chaum. 'Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms'. In: *Commun ACM* 24.2 (Feb. 1981), pp. 84–90.

[27] J. Chen, B. Sun, H. Li, H. Lu and X.-S. Hua. 'Deep CTR Prediction in Display Advertising'. In: *Proceedings of the 2016 ACM on Multimedia Conference*. MM '16. 2016, pp. 811–820.

[28] T. Chen and S. Zhong. 'Privacy-Preserving Backpropagation Neural Network Learning'. In: *IEEE Trans. Neural Netw.* 20.10 (Oct. 2009), pp. 1554–1564.

[29] Y. Chen, D. Pavlov and J. F. Canny. 'Large-Scale Behavioral Targeting'. In: *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '09. 2009, pp. 209–218.

[30] M. Conti, V. Cozza, M. Petrocchi and A. Spognardi. 'TRAP: Using Targeted Ads to Unveil Google Personal Profiles'. In: *2015 IEEE International Workshop on Information Forensics and Security (WIFS)*. Nov. 2015, pp. 1–6.

[31] B. Dalessandro, D. Chen, T. Raeder, C. Perlich, M. Han Williams and F. Provost. 'Scalable Hands-Free Transfer Learning for Online Advertising'. In: *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '14. 2014, pp. 1573–1582.

[32] I. Damgård and M. Jurik. 'A Generalisation, a Simplification and Some Applications of Paillier's Probabilistic Public-Key System'. In: *Public Key Cryptography*. Vol. 1992. 2001, pp. 119–136.

[33] I. Damgård and R. Thorbek. 'Efficient Conversion of Secret-Shared Values Between Different Fields.' In: *IACR Cryptol. EPrint Arch.* 2008 (2008), p. 221.

[34] M. Degeling and T. Herrmann. ‘Your Interests According to Google - A Profile-Centered Analysis for Obfuscation of Online Tracking Profiles’. In: *ArXiv e-prints* (Jan. 2016).

[35] J. Deighton and H. M. Brierley. *Economic Value of the Advertising-Supported Internet Ecosystem*. Interactive Advertising Bureau, Sept. 2012. URL: https://www.iab.com/wp-content/uploads/2015/07/iab_Report_September-24-2012_4clr_v1.pdf (visited on 16/01/2017).

[36] C. Dwork, F. McSherry, K. Nissim and A. Smith. ‘Calibrating Noise to Sensitivity in Private Data Analysis’. In: *Theory of Cryptography*. Mar. 2006, pp. 265–284.

[37] P. Eckersley. ‘How Unique Is Your Web Browser?’ In: *Privacy Enhancing Technologies*. July 2010, pp. 1–18.

[38] T. ElGamal. ‘A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms’. In: *IEEE Trans. Inf. Theory* 31.4 (July 1985), pp. 469–472.

[39] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk and T. Toft. ‘Privacy-Preserving Face Recognition’. In: *Privacy Enhancing Technologies*. Vol. 5672. Lecture Notes in Computer Science. 2009, pp. 235–253.

[40] J. Estrada-Jiménez, J. Parra-Arnau, A. Rodríguez-Hoyos and J. Forné. ‘Online Advertising: Analysis of Privacy Threats and Protection Approaches’. In: *Computer Communications* 100 (Mar. 2017), pp. 32–51.

[41] D. S. Evans. ‘The Online Advertising Industry: Economics, Evolution, and Privacy’. In: *J. Econ. Perspect.* 23.3 (Sept. 2009), pp. 37–60.

[42] M. Falahrastegar, H. Haddadi, S. Uhlig and R. Mortier. ‘Tracking Personal Identifiers Across the Web’. In: *Passive and Active Measurement*. Mar. 2016, pp. 30–41.

[43] M. Fredrikson and B. Livshits. ‘RePriv: Re-Imagining Content Personalization and In-Browser Privacy’. In: *2011 IEEE Symposium on Security and Privacy*. May 2011, pp. 131–146.

[44] M. J. B. Geisler. ‘Cryptographic Protocols: Theory and Implementation’. PhD thesis. Department of Computer Science, Aarhus University, 2010.

[45] R. Gennaro, M. O. Rabin and T. Rabin. ‘Simplified VSS and Fast-Track Multiparty Computations with Applications to Threshold Cryptography’. In: *Proceedings of the Seventeenth Annual ACM Symposium on Principles of Distributed Computing*. PODC ’98. 1998, pp. 101–111.

[46] O. Goldreich. 'Towards a Theory of Software Protection and Simulation by Oblivious RAMs'. In: *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*. STOC '87. 1987, pp. 182–194.

[47] P. Golle, M. Jakobsson, A. Juels and P. Syverson. 'Universal Re-Encryption for Mixnets'. In: *Topics in Cryptology CT-RSA 2004*. Feb. 2004, pp. 163–178.

[48] Google. *Topics Used for Personalized Ads*. 2017. URL: https://support.google.com/ads/answer/2842480?hl=en (visited on 28/03/2017).

[49] M. Green, W. Ladd and I. Miers. 'A Protocol for Privately Reporting Ad Impressions at Scale'. In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. CCS '16. Oct. 2016, pp. 1591–1601.

[50] J. Groth and M. Kohlweiss. 'One-Out-of-Many Proofs: Or How to Leak a Secret and Spend a Coin'. In: *Advances in Cryptology - EUROCRYPT 2015*. Apr. 2015, pp. 253–280.

[51] S. Guha, B. Cheng and P. Francis. 'Privad: Practical Privacy in Online Advertising'. In: *Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation*. NSDI'11. Mar. 2011, pp. 169–182.

[52] S. Guha, A. Reznichenko, K. Tang, H. Haddadi and P. Francis. 'Serving Ads from Localhost for Performance, Privacy, and Profit.' In: *Proceedings of the 8th ACM SIGCOMM Workshop on Hot Topics in Networks*. HotNets-VIII. 2009.

[53] C. Hazay, G. L. Mikkelsen, T. Rabin and T. Toft. 'Efficient RSA Key Generation and Threshold Paillier in the Two-Party Setting'. In: *Topics in Cryptology CT-RSA 2012*. Feb. 2012, pp. 313–331.

[54] X. He et al. 'Practical Lessons from Predicting Clicks on Ads at Facebook'. In: *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising*. ADKDD'14. 2014, 5:1–5:9.

[55] D. Herrmann, M. MaaSS and H. Federrath. 'Evaluating the Security of a DNS Query Obfuscation Scheme for Private Web Surfing'. In: *ICT Systems Security and Privacy Protection*. June 2014, pp. 205–219.

[56] Y. Juan, D. Lefortier and O. Chapelle. 'Field-Aware Factorization Machines in a Real-World Online Advertising System'. In: *Proceedings of the 26th International Conference on World Wide Web Companion*. 2017, pp. 680–688.

[57] Y. Juan, Y. Zhuang, W.-S. Chin and C.-J. Lin. 'Field-Aware Factorization Machines for CTR Prediction'. In: *Proceedings of the 10th ACM Conference on Recommender Systems*. RecSys '16. 2016, pp. 43–50.

[58] A. Juels. 'Targeted Advertising ... and Privacy Too'. In: *Topics in Cryptology CT-RSA 2001*. Apr. 2001, pp. 408–424.

[59] A. Korolova. 'Privacy Violations Using Microtargeted Ads: A Case Study'. In: *2010 IEEE International Conference on Data Mining Workshops*. Dec. 2010, pp. 474–482.

[60] J. Krumm. 'A Survey of Computational Location Privacy'. In: *Pers Ubiquit Comput* 13.6 (Aug. 2009), pp. 391–399.

[61] P. Leon, B. Ur, R. Shay, Y. Wang, R. Balebako and L. Cranor. 'Why Johnny Can't Opt out: A Usability Evaluation of Tools to Limit Online Behavioral Advertising'. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '12. May 2012, pp. 589–598.

[62] V. Letang and L. Stillman. *Global Advertising Forecast*. MAGNA, Dec. 2016. URL: http://magnaglobal.com/wp-content/uploads/2016/12/MAGNA-December-Global-Forecast-Update-Press-Release.pdf (visited on 03/03/2017).

[63] Y. Lindell and E. Omri. 'A Practical Application of Differential Privacy to Personalized Online Advertising.' In: *IACR Cryptol. EPrint Arch.* (Mar. 2011).

[64] J. R. Mayer and J. C. Mitchell. 'Third-Party Web Tracking: Policy and Technology'. In: *2012 IEEE Symposium on Security and Privacy*. May 2012, pp. 413–427.

[65] H. B. McMahan et al. 'Ad Click Prediction: A View from the Trenches'. In: *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '13. 2013, pp. 1222–1230.

[66] A. Mense, S. Steger, D. Jukic-Sunaric, A. Mészáros and M. Sulek. 'Open Source Based Privacy-Proxy to Restrain Connectivity of Mobile Apps'. In: *Proceedings of the 14th International Conference on Advances in Mobile Computing and Multi Media*. MoMM '16. 2016, pp. 284–287.

[67] G. Merzdovnik, M. Huber, D. Buhov, N. Nikiforakis, S. Neuner, M. Schmiedecker and E. Weippl. 'Block Me If You Can: A Large-Scale Study of Tracker-Blocking Tools'. In: *2nd IEEE European Symposium on Security and Privacy*. Apr. 2017.

[68] M. Nateghizad, Z. Erkin and R. L. Lagendijk. 'An Efficient Privacy-Preserving Comparison Protocol in Smart Metering Systems'. In: *EURASIP J. on Info. Security* 2016.1 (Dec. 2016), p. 11.

[69] A. Y. Ng. 'Feature Selection, L1 vs. L2 Regularization, and Rotational Invariance'. In: *Proceedings of the Twenty-First International Conference on Machine Learning*. ICML '04. 2004, pp. 78–.

[70] S. Nicholls, A. Malins and M. Horner. *Real-Time Bidding in Online Advertising*. GP Bullhound, 2013. URL: `http://www.gpbullhound.com/wp-content/uploads/2014/09/Real-Time-Bidding-in-Online-Advertising.pdf` (visited on 08/12/2016).

[71] N. Nikiforakis, W. Joosen and B. Livshits. 'PriVaricator: Deceiving Fingerprinters with Little White Lies'. In: *Proceedings of the 24th International Conference on World Wide Web*. WWW '15. 2015, pp. 820–830.

[72] R. J. Oentaryo, E.-P. Lim, J.-W. Low, D. Lo and M. Finegold. 'Predicting Response in Mobile Advertising with Hierarchical Importance-Aware Factorization Machine'. In: *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*. WSDM '14. 2014, pp. 123–132.

[73] L. Olejnik, C. Castelluccia and A. Janc. 'Why Johnny Can't Browse in Peace: On the Uniqueness of Web Browsing History Patterns'. In: *5th Workshop on Hot Topics in Privacy Enhancing Technologies*. HotPETs 2012. July 2012.

[74] L. Olejnik, T. Minh-Dung and C. Castelluccia. 'Selling Off Privacy at Auction'. In: *Network and Distributed System Security Symposium*. NDSS '14. Feb. 2014.

[75] C. Orlandi, A. Piva and M. Barni. 'Oblivious Neural Network Computing via Homomorphic Encryption'. In: *EURASIP J Inf Secur* 2007 (Jan. 2007), 18:1–18:10.

[76] PageFair. *The State of the Blocked Web*. Feb. 2017. URL: `https://pagefair.com/downloads/2017/01/PageFair-2017-Adblock-Report.pdf` (visited on 07/03/2017).

[77] PageFair and Adobe. *The Cost of Ad Blocking*. Aug. 2015. URL: `http://downloads.pagefair.com/reports/2015_report-the_cost_of_ad_blocking.pdf` (visited on 19/01/2017).

[78] P. Paillier. 'Public-Key Cryptosystems Based on Composite Degree Residuosity Classes'. In: *Advances in Cryptology EUROCRYPT 99*. May 1999, pp. 223–238.

[79] Z. Pan, E. Chen, Q. Liu, T. Xu, H. Ma and H. Lin. 'Sparse Factorization Machines for Click-through Rate Prediction'. In: *2016 IEEE 16th International Conference on Data Mining (ICDM)*. Dec. 2016, pp. 400–409.

[80] S. Pandey, M. Aly, A. Bagherjeiran, A. Hatch, P. Ciccolo, A. Ratnaparkhi and M. Zinkevich. 'Learning to Target: What Works for Behavioral Targeting'. In: *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*. CIKM '11. 2011, pp. 1805–1814.

[81] F. Papaodyssefs, C. Iordanou, J. Blackburn, N. Laoutaris and K. Papagiannaki. 'Web Identity Translator: Behavioral Advertising and Identity Privacy with WIT'. In: *Proceedings of the 14th ACM Workshop on Hot Topics in Networks*. HotNets-XIV. Nov. 2015, 3:1–3:7.

[82] S. T. Peddinti and N. Saxena. 'On the Privacy of Web Search Based on Query Obfuscation: A Case Study of TrackMeNot'. In: *Privacy Enhancing Technologies*. July 2010, pp. 19–37.

[83] S. T. Peddinti and N. Saxena. 'On the Limitations of Query Obfuscation Techniques for Location Privacy'. In: *Proceedings of the 13th International Conference on Ubiquitous Computing*. UbiComp '11. Sept. 2011, pp. 187–196.

[84] C. Perlich, B. Dalessandro, T. Raeder, O. Stitelman and F. Provost. 'Machine Learning for Targeted Display Advertising: Transfer Learning in Action'. In: *Mach. Learn.* 95.1 (Apr. 2014), pp. 103–127.

[85] PwC. *IAB Internet Advertising Revenue Report: 2015 Full Year Results*. Interactive Advertising Bureau, Apr. 2016. URL: http://www.iab.com/wp-content/uploads/2016/04/IAB_Internet_Advertising_Revenue_Report_FY_2015-final.pdf (visited on 16/01/2017).

[86] T. Raeder, B. Dalessandro and C. Perlich. *Considering Privacy in Predictive Modeling Applications*. 2014. URL: https://dstillery.com/wp-content/uploads/2016/07/Considering-Privacy.pdf (visited on 29/03/2017).

[87] T. Raeder, C. Perlich, B. Dalessandro, O. Stitelman and F. Provost. 'Scalable Supervised Dimensionality Reduction Using Clustering'. In: *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '13. 2013, pp. 1213–1221.

[88] T. I. Reistad and T. Toft. 'Secret Sharing Comparison by Transformation and Rotation'. In: *Information Theoretic Security*. Lecture Notes in Computer Science. May 2007, pp. 169–180.

[89] S. Rendle. 'Factorization Machines'. In: *2010 IEEE International Conference on Data Mining*. Dec. 2010, pp. 995–1000.

[90] A.-R. Sadeghi, T. Schneider and I. Wehrenberg. 'Efficient Privacy-Preserving Face Recognition'. In: *Information, Security and Cryptology ICISC 2009*. Dec. 2009, pp. 229–244.

[91] A. Shamir. 'How to Share a Secret'. In: *Commun ACM* 22.11 (Nov. 1979), pp. 612–613.

[92] E. G. Smit, G. Van Noort and H. A. M. Voorveld. 'Understanding Online Behavioural Advertising: User Knowledge, Privacy Concerns and Online Coping Behaviour in Europe'. In: *Computers in Human Behavior* 32 (Mar. 2014), pp. 15–22.

[93] L. Sweeney. 'K-Anonymity: A Model for Protecting Privacy'. In: *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.* 10 (05 2002), pp. 557–570.

[94] A.-P. Ta. 'Factorization Machines with Follow-the-Regularized-Leader for CTR Prediction in Display Advertising'. In: Oct. 2015, pp. 2889–2891.

[95] C. F. Torres, H. Jonker and S. Mauw. 'FP-Block: Usable Web Privacy by Controlling Browser Fingerprinting'. In: *Computer Security – ESORICS 2015*. Sept. 2015, pp. 3–19.

[96] V. Toubiana, A. Narayanan, D. Boneh, H. Nissenbaum and S. Barocas. 'Adnostic: Privacy Preserving Targeted Advertising.' In: *Network and Distributed System Symposium*. NDSS '10. Mar. 2010.

[97] TRUSTe. *2011 Consumer Research Results: Privacy and Online Behavioural Advertising*. July 2011. URL: https://www.eff.org/files/truste-2011-consumer-behavioral-advertising-survey-results.pdf (visited on 14/03/2017).

[98] B. Ur, P. G. Leon, L. F. Cranor, R. Shay and Y. Wang. 'Smart, Useful, Scary, Creepy: Perceptions of Online Behavioral Advertising'. In: *Proceedings of the Eighth Symposium on Usable Privacy and Security*. SOUPS '12. July 2012, 4:1–4:15.

[99] J. Wang, W. Zhang and S. Yuan. 'Display Advertising with Real-Time Bidding (RTB) and Behavioural Targeting'. In: *ArXiv e-prints* (Oct. 2016).

[100] Y. Wang, K. Ren, W. Zhang, J. Wang and Y. Yu. 'Functional Bid Landscape Forecasting for Display Advertising'. In: *Machine Learning and Knowledge Discovery in Databases*. Sept. 2016, pp. 115–131.

[101] K. Weinberger, A. Dasgupta, J. Langford, A. Smola and J. Attenberg. 'Feature Hashing for Large Scale Multitask Learning'. In: *Proceedings of the 26th Annual International Conference on Machine Learning*. ICML '09. June 2009, pp. 1113–1120.

[102] W. C.-H. Wu, M.-Y. Yeh and M.-S. Chen. 'Predicting Winning Price in Real Time Bidding with Censored Data'. In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '15. 2015, pp. 1305–1314.

[103] J. Yan, N. Liu, G. Wang, W. Zhang, Y. Jiang and Z. Chen. 'How Much Can Behavioral Targeting Help Online Advertising?' In: *Proceedings of the 18th International Conference on World Wide Web*. WWW '09. Apr. 2009, pp. 261–270.

[104] S. Yuan, J. Wang and X. Zhao. 'Real-Time Bidding for Online Advertising: Measurement and Analysis'. In: *Proceedings of the Seventh International Workshop on Data Mining for Online Advertising*. ADKDD '13. 2013, 3:1–3:8.

[105] Q. Zhang, L. T. Yang and Z. Chen. 'Privacy Preserving Deep Computation Model on Cloud for Big Data Feature Learning'. In: *IEEE Trans. Comput.* 65.5 (May 2016), pp. 1351–1362.

[106] W. Zhang, L. Chen and J. Wang. 'Implicit Look-Alike Modelling in Display Ads'. In: *Advances in Information Retrieval*. Mar. 2016, pp. 589–601.

[107] W. Zhang, T. Du and J. Wang. 'Deep Learning over Multi-Field Categorical Data'. In: *Advances in Information Retrieval*. Lecture Notes in Computer Science 9626. Mar. 2016, pp. 45–57.

[108] W. Zhang, S. Yuan and J. Wang. 'Optimal Real-Time Bidding for Display Advertising'. In: *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '14. 2014, pp. 1077–1086.

[109] Y. Zou, X. Jin, Y. Li, Z. Guo, E. Wang and B. Xiao. 'Mariana: Tencent Deep Learning Platform and Its Applications'. In: *Proc VLDB Endow* 7.13 (Aug. 2014), pp. 1772–1777.