XRay: Enhancing the Web's Transparency with Differential Correlation

Mathias Lecuyer, Guillaume Ducoffe, Francis Lan, Andrei Papancea, Theofilos Petsios, Riley Spahn, Augustin Chaintreau, Roxana Geambasu

XRay: Enhancing the Web's Transparency with Differential Correlation

Mathias Lécuyer, Guillaume Ducoffe, Francis Lan, Andrei Papancea, Theofilos Petsios,
Riley Spahn, Augustin Chaintreau, *and* Roxana Geambasu
Columbia University

Abstract

Today's Web services – such as Google, Amazon, and Facebook – leverage user data for varied purposes, including personalizing recommendations, targeting advertisements, and adjusting prices. At present, users have little insight into how their data is being used. Hence, they cannot make informed choices about the services they choose.

To increase transparency, we developed *XRay*, the first fine-grained, robust, and scalable personal data tracking system for the Web. XRay predicts which data in an arbitrary Web account (such as emails, searches, or viewed products) is being used to target which outputs (such as ads, recommended products, or prices). XRay's core functions are service agnostic and easy to instantiate for new services, and they can track data within and across services. To make predictions independent of the audited service, XRay relies on the following insight: by comparing outputs from different accounts with similar, but not identical, subsets of data, one can pinpoint targeting through correlation. We show both theoretically, and through experiments on Gmail, Amazon, and YouTube, that XRay achieves high precision and recall by correlating data from a surprisingly small number of extra accounts.

1 Introduction

We live in a "big data" world. Staggering amounts of personal data – our as locations, search histories, emails, posts, and photos – are constantly collected and analyzed by Google, Amazon, Facebook, and a myriad of other Web services. This presents rich opportunities for marshaling big data to improve daily life and social well-being. For example, personal data improves the usability of applications by letting them predict and seamlessly adapt to future user needs and preferences. It improves business revenues by enabling effective product placement and targeted advertisements. Twitter data has been successfully applied to public health problems [38], crime prevention [43], and emergency response [22]. These beneficial uses have generated a big data frenzy, with Web services aggressively pursuing new ways to acquire and commercialize it.

Despite its innovative potential, the personal data frenzy has transformed the Web into an opaque and privacy-insensitive environment. Web services accumulate data, exploit it for varied and undisclosed purposes, retain it for extended periods of time, and possibly share it with others – *all without the data owner's knowledge or consent*. Who has what data, and for what purposes is it used? Are the uses in the data owners' best interests? Does the service adhere to its own privacy policy? How long is data used after its owner deletes it? Who shares data with whom?

At present, users lack answers to these questions, and investigators (such as FTC agents, journalists, or researchers) lack robust tools to track data in the ever-changing Web to provide the answers. Left unchecked, the exciting potential of big data threatens to become a breeding ground for data abuses, privacy vulnerabilities, and unfair or deceptive business practices. Examples of such practices have begun to surface. In a recent incident, Google was found to have used institutional emails from ad-free Google Apps for Education to target ads in users' personal accounts [23, 32]. MySpace was found to have violated its privacy policy by leaking personally identifiable information to advertisers [28]. Several consumer sites, such as Orbitz and Staples, were found to have adjusted their product pricing based on user location [24, 25]. And Facebook's 2010 ad targeting was shown to be vulnerable to micro-targeted ads specially crafted to reveal a user's private profile data [26].

To increase transparency and provide checks and balances on data abuse, we argue that new, robust, and versatile tools are needed to effectively track the use of personal data on the Web. Tracking data in a *controlled environment*, such as a modified operating system, language, or runtime, is an old problem with a well-known solution: taint tracking systems [12, 16, 7, 47]. However, is it possible to track data in an *uncontrolled environment*, such as the Web? Can robust, generic mechanisms assist in doing so? What kinds of data uses are trackable and what are not? How would the mechanisms scale with the amount of data being tracked?

As a first step toward answering these questions, we built *XRay*, a personal data tracking system for the Web.

XRay correlates designated data *inputs* (be they emails, searches, or visited products) with data *outputs* results (such as ads, recommended products, or prices). Its correlation mechanism is service agnostic and easy to instantiate, and it can track data use within and across services. For example, it lets a data owners track how their emails, Google+, and YouTube activities are used to target ads in Gmail.

At its core, XRay relies on a *differential correlation* mechanism that pinpoints targeting by comparing outputs in different accounts with similar, but not identical, subsets of data inputs. To do so, it associates with every personal account a number of *shadow accounts*, each of which contains different data subsets. The correlation mechanism uses a simple Bayesian model to compute and rank scores for every data input that may have triggered a specific output. Intuitively, if an ad were seen in many accounts that share a certain email, and never in accounts that lack that email, then the email is likely to be responsible for a characteristic that triggers the ad. The email's score for that ad would therefore be high. Conversely, if the ad were seen rarely in accounts with or lacking that email, that email's score for this ad would be low.

Constructing a practical auditing system around differential correlation raises significant challenges. Chief among them is scalability with the number of data items. Theoretically, XRay requires a shadow account for each combination of data inputs to accurately pinpoint correlation. That would suggest an *exponential number of accounts*! Upon closer examination, however, we find that a few realistic assumptions and novel mechanisms let XRay reach high precision and recall with only a *logarithmic number of accounts in number of data inputs*. We deem this a major new result for the science of tracking data-targeting on the Web.

We built an XRay prototype and used it to correlate Gmail ads, Amazon product recommendations, and YouTube video suggestions to user emails, wish lists, and previously watched videos, respectively. While Amazon and YouTube provide detailed explanations of their targeting, Gmail does not, so we manually validated associations. For all cases, XRay achieved 80-90% precision and recall. Moreover, we integrated our Gmail and YouTube prototypes so we could track cross-service ad targeting. Although several prior measurement studies [10, 46, 21, 20, 33] used methodologies akin to differential correlation, we believe we are the first to build a generic, service agnostic, and scalable tool based on it. Overall, we make the following contributions:

1. The first general, versatile, and open system to track arbitrary personal Web data use by uncontrolled services. The code is available from our Web page `https://xray.cs.columbia.edu/`.
2. The first in-depth exploration into the scalability challenges of tracking personal data on the Web.
3. The design and implementation of robust mechanisms to address scaling, including data matching.
4. System instantiation to track data on three services (Gmail, Amazon, YouTube) and across services (YouTube to Gmail).
5. An evaluation of our system's precision and recall on Gmail, Amazon, and YouTube. We show that XRay is accurate and scalable. Further, it reveals intriguing practices now in use by Web services and advertisers.

2 Motivation

This paper lays the algorithmic foundations for a new generation of scalable, robust, and versatile tools to lift the curtain on how personal data is being targeted. We underscore the need for such tools by describing potential usage scenarios inspired by real-life examples (§2.1). We do this not to point fingers at specific service providers; rather, we aim to show the many situations where transparency tools would be valuable for end-users and auditors alike. We conclude this section by briefly analyzing how current approaches fail to address these usage scenarios (§2.2).

2.1 Usage Scenarios

Scenario 1: Why This Ad? Ann often uses her Gmail ads to discover new retail offerings. Recently, she discussed her ad-clicking practices with her friend Tom, a computer security expert. Tom warned her about potential privacy implications of clicking on ads without knowing what data they target. For example, if she clicks on an ad targeting the keyword "gay" and then authenticates to purchase something from that vendor, she is unwittingly volunteering potentially sensitive information to the vendor. Tom tells Ann about two options to protect her privacy. She can either disable the ads altogether (using a system like AdBlock [1]), or install the XRay Gmail plugin to uncover targeting against her data. Unwilling to give up the convenience of ads, Ann chooses the latter. XRay clearly annotates the ads in the Gmail UI with their target email or combination, if any. Ann now inspects this targeting before clicking on an ad and avoids clicking if highly sensitive emails are being targeted.

Scenario 2: They're Targeting *What*? Bob, an FTC investigator, uses the XRay Gmail plugin for a different purpose: to study sensitive-data targeting practices by advertisers. He suspects a potentially unfair practice whereby companies use Google's ad network to collect sensitive information about their customers. Therefore, Bob creates a number of emails containing keywords such as "cancer," "AIDS," "bankruptcy," and "unemployment." He refreshes the Gmail page many times, each time recording the targeted ads and XRay's explanations

for them. The experiment reveals an interesting result: an online insurance company, TrustInUs.com, has targeted multiple ads against his illness-related emails. Bob hypothesizes that the company might use the data to set higher premiums for users reaching their site through a disease-targeted ad. He uses XRay results as initial evidence to open an investigation of TrustInUs.com.

Scenario 3: What's With The New Policy?[1] Carla, an investigative journalist, has set up a watcher on privacy policies for major Web services. When a change occurs, the watcher notifies her of the difference. Recently, an important sentence in Google's privacy policy has been scrapped:

> *If you are using Google Apps (free edition), email is scanned so we can display conceptually relevant advertising in some circumstances. ~~Note that there is no ad-related scanning or processing in Google Apps for Education or Business with ads disabled.~~*

To investigate scientifically whether this omission represents a shift in implemented policy, she obtains institutional accounts, connects them to personal accounts, and uses XRay to detect the correlation between emails in institutional accounts and ads in corresponding personal accounts. Finding a strong correlation, Carla writes an article to expose the policy change and its implications.

Scenario 4: Does Delete Mean Delete? Dan, a CS researcher, has seen the latest news that Snapchat, an ephemeral-image sharing Website, does not destroy users' images after the requested timeout but instead just unlinks them [19]. He wonders whether the reasons for this are purely technical as the company has declared (e.g., flash wearing levels, undelete support, spam filtering) [40, 39] or whether these photos, or metadata drawn from them, are mined to target ads or other products on the Website. The answer will influence his decision about whether to continue using the service. Dan instantiates XRay to track the correlation between his expired Snapchat photos and ads.

2.2 Alternative Approaches

The preceding scenarios illustrate the importance of transparency in protecting privacy across a range of use cases. *We need robust, generic auditing tools to track the use of personal data at fine granularity (e.g., individual emails, photos) within and across arbitrary Web services.* At present, no such tools exist, and the science of tracking the use of personal Web data at a fine grain is largely non-existent.

[1]In Feb. 2014, it was revealed based on court documents that Google could have used institutional emails to target ads in personal accounts [23]. In May 2014, Google committed to disable that feature [**?**]. Scenario 3 presents an XRay-based approach to investigate the original allegation.

Existing approaches can be broadly classified in two categories: *protection tools*, which prevent Web services' acquisition or use of personal data, and (2) *auditing tools*, which uncover Web services' acquisition or use of personal data. We discuss these approaches next; further related work is in §9.

Protection Tools. A variety of protection tools exist [11, 37, 1, 48]. For example, Ann could disable ads using an ad blocker [1]. Alternatively, she could encrypt her emails, particularly the sensitive ones, to prevent Google from using them to target ads. Dan could use a self-destructing data system, such as Vanish [14], to ensure the ephemerality of his Snapchat photos.

While we encourage the use of protection tools, they impose difficult tradeoffs that make them inapplicable in many cases. If Ann blocks all her ads, she cannot benefit from those she might find useful; if she encrypts all of her emails, she cannot search them; if she encrypts only her sensitive emails, she cannot protect any sensitive emails she neglected to encrypt in advance. Similarly, if Dan encrypts his Snapchat photos, sharing them becomes more difficult. While more sophisticated protection systems address certain limitations (e.g., searchable [5], homomorphic [15, 35], and attribute-based encryption [18], or privacy-preserving advertising [42, 13]), they are generally heavyweight [15], difficult to use [44], or require major service-side changes [15, 42, 13].

Auditing Tools. Given the limitations of protection tools, transparency is gaining increased attention [46, 12, 21]. If protecting data proves too cumbersome, limiting, or unsupportive of business needs, then users should at least be able to know: (1) *who is handling their data?*, and (2) *what is it being used for?*

Several tools developed in recent years partially address the first question by revealing where personal data flows from a local device [36, 12, 8]. TaintDroid [12] uses taint tracking to detect leakage of personal data from a mobile application to a service or third-party backend. ShareMeNot [36] and Mozilla's Lightbeam Firefox add-on [30] identify third parties that are observing user activities across the Web. These systems track personal data – such as location, sensor data, Web searches, or visited sites – *until it leaves the user's device*. Once the data is uploaded to Web services, it can be used or sold without a trace. In contrast, XRay's tracking just begins: we aim to tell users how services use their data *once they have it*.

Several new tools and personalization measurement studies partially address the second question: what data is being used for [10, 46, 21, 20, 33]. In general, all existing tools are highly specialized, focusing on specific input types, outputs, or services. No general, principled foundation for data use auditing exists, that can be applied effectively to many services, a primary

motivation for this our work. For example, Bobble [46] reveals search result personalization based on user location (e.g., IP) and search history. Moreover, existing tools aim to discover only *whether* certain types of user inputs – such as search history, browsing history, IP, etc. – influence the output. None pinpoints at fine grain *which* specific input – which search query, which visited site, or which viewed product – or combination of inputs explain which output. XRay, whose goals we describe next, aims to do just that.

3 Goals and Models

Our overarching goal is to develop the core abstractions and mechanisms for tracking data within and across arbitrary Web sites. After describing specific goals (§3.1), we narrow our scope with a set of simplifying assumptions regarding the data uses that XRay is designed to audit (§3.2) and the threats it addresses (§3.3).

3.1 Goals

Three specific goals have guided XRay's design:

Goal 1: *Fine-Grained and Accurate Data Tracking.* Detect which specific data inputs (e.g., emails) have likely triggered a particular output (e.g., an ad). While coarse-grained data use information (such as Gmail's typical statement, "This ad is based on emails from your mailbox.") may suffice at times, knowing the specifics can be revelatory, particularly when the input is highly sensitive and aggressively targeted.

Goal 2: *Scalability.* Make it practical to track significant amounts of data (e.g., past month's emails). We aim to support the tracking of hundreds of inputs with reasonable costs in terms of shadow accounts. These accounts are generally scarce resource since their creation is being constrained by Web services. While we assume that users and auditors can obtain *some* accounts on the Web services they audit (e.g., a couple dozen), we strive to minimize the number required for accurate and fine-grained data tracking.

Goal 3: *Extensibility, Generality, and Self-Tuning.* Make XRay generic and easy to instantiate for many services and input/output types. Instantiating XRay to track data on new Web sites should be simple, although it may require some service-specific implementation of input/output monitoring. However, XRay's correlation machinery – the conceptually challenging part of a scalable auditing tool – should be turn key and require no manual tuning.

3.2 Web Service Model

These goals may appear unsurmountable. An extremely heterogeneous environment, the Web has perhaps as many data uses as services. Moreover, data mining algorithms can be complex and proprietary. How can we abstract away this diversity and complexity to design

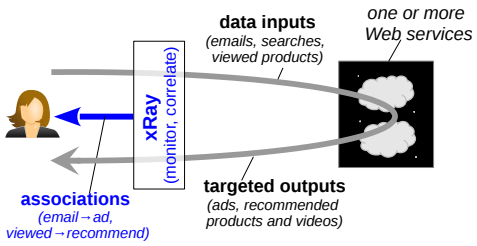


Figure 1: **XRay Conceptual View.** XRay views Web services as black boxes, monitors user *inputs* and *outputs* to/from them, and detects data use through correlation. It returns to the user or auditor *associations* of specific inputs and outputs.

robust and generic building blocks for scalable data tracking? Fortunately, we find that certain popular classes of Web data uses lend themselves to principled abstractions that facilitate scalable tracking.

Figure 1 shows XRay's simplified view of Web services. Services, and networks of services that exchange user data, are *black boxes* that receive personal data *inputs* from users – such as emails, pictures, search queries, locations, or purchases – and use them for varied purposes. Some uses materialize into *outputs* visible to users, such as ads, product or video recommendations, or prices. Others invisible to the users. XRay correlates some visible data inputs with some visible outputs by monitoring them, correlating them, and reporting strong *associations* to users. An example association is which email(s) contributed to the selection of a particular ad.

XRay relates only *strongly correlated* inputs with outputs. If an output is strongly correlated to an input (i.e., the input's presence or absence changes the output), then XRay will likely be able to detect its use. If not (i.e., the monitored input plays but a small role in the output), then it may go undetected. XRay also relates small combinations of inputs with strongly correlated outputs.

Although simple, this model efficiently addresses several types of personal data functions, including product recommendations, price discriminations, and various personalization functions (e.g., search, news). We refer to such functions generically as *targeting functions* and focus XRay's design on them.

Three popular forms of targeting are:

1. *Profile Targeting*, which leverages static or slowly evolving explicit information – such as age, gender, race, or location – that the user often supplies by filling a form. This type of targeting has been studied profusely [10, 46, 21, 20, 33]; we thus ignore it here.
2. *Contextual Targeting*, which leverages the content currently being displayed. In Gmail, this is the currently open email next to which the ad is shown. In Amazon or Youtube, the target is the product or video next to which the recommendation is shown.
3. *Behavioral Targeting*, which leverages a user's past actions. An email sent or received today can trig-

ger an ad tomorrow; a video watched now can trigger a recommendation later. Use of histories makes it harder for users to track which data is being used, a key motivation for our development of XRay.

Theoretically, our differential correlation algorithms could be applied to all three forms of targeting. From a systems perspective, XRay's design is geared towards *contextual targeting* and *a specific form of behavioral targeting*. The latter requires further attention. We observe that this broad targeting class subsumes multiple types of targeting that operate at different granularities. For example, a service could use as inputs a user's most recent few emails to decide targeting. This would be similar to an extended context. Alternatively, a service could use historical input to learn a user's coarse interests or characteristics and base its targeting on that.

XRay currently aims to disclose any targeting applied at the level of individual user data, or small combinations thereof. Our differential correlation algorithms could be applied to detect targeting that operates on a coarser granularity. However, the XRay system itself would require significant changes. Unless otherwise noted, we use *behavioral targeting* to denote the restricted form of behavioral targeting that XRay is designed to address. We formalize these restrictions in §4.2.

3.3 Threat Model

To further narrow our problem's scope, even further, we introduce threat assumptions. We assume that data owners (users and auditors) are trusted and do not attempt to leverage XRay to harm Web services or the Web ecosystem. While they trust Web services with their data, they wish to better understand how that data is being used. Data owners are thus assumed to upload the data in cleartext to the Web services.

The threat models relevant for Web services depend on the use case. For example, Scenarios 1 and 2 in §2.1 assume Google is trusted, but its users wish to understand more about how advertisers target them through its ad platform. In contrast, in Scenarios 3 and 4, investigators may have reason to believe that Web services might intentionally frustrate auditing.

This paper assumes an *honest-but-curious* model for Web services: they try to use private data for financial or functional gains, but they do not try to frustrate our auditing mechanism, e.g., by identifying and disabling shadow accounts. The service might attempt to defend itself against more general types of attacks, such as spammers or DDoS attacks. For example, many Web services constrain the creation of accounts so as to limit spamming and false clicks. Similarly, Web services may rate limit or block the IPs of aggressive data collectors. XRay must be robust to such inherent defenses. We discuss challenges and potential approaches for stronger adversarial models in §7.

4 The XRay Architecture

XRay's design addresses the preceding goals and assumptions. For concreteness, we draw examples from our three XRay instantiations: tracking email-to-ad targeting association within Gmail, attributing recommended videos to those already seen on YouTube, and identifying products in a wish list that generate a recommendation on Amazon.

4.1 Architectural Overview

XRay's high-level architecture (Figure 2) consists of three components: (1) a *Browser Plugin*, which intercepts tracked inputs and outputs to/from an audited Web service and gives users visual feedback about any input/output associations, (2) a *Shadow Account Manager*, which populates shadow accounts with inputs from the plugin and collects outputs (e.g., ads) for each shadow account, and (3) the *Correlation Engine*, XRay's core, which infers associations and provides them to the plugin for visualization. While the Browser Plugin and Shadow Account Manager are *service specific*, the Correlation Engine, which encapsulates the science of Web-data tracking, is *service agnostic*. After we describe each component, we focus on the design of the Correlation Engine.

Browser Plugin. The Browser Plugin intercepts designated inputs and outputs (i.e., *tracked inputs/outputs*) by recognizing specific DOM elements in an audited service's Web pages. Other inputs and outputs may not be tracked by XRay (i.e., *untracked inputs/outputs*). The decision of what to track belongs to an investigator or developer who instantiates XRay to work on a specific service. For example, we configure the XRay Gmail Plugin to monitor a user's emails as inputs and ads as outputs. When the Plugin gets a new tracked input (e.g., a new email), it forwards it both to the service and to the Shadow Account Manager. When the Plugin gets a new tracked output (e.g., an ad), it queries the Correlation Engine for associations with the user's tracked inputs (message `get_assoc`).

Shadow Account Manager. This component: (1) populates the shadow accounts with subsets of a user account's tracked inputs (denoted D_i), and (2) periodically retrieves outputs (denoted O_k) from the audited service for each shadow account. Both functions are service specific. For Gmail, they send emails with SMTP and call the ad API. For YouTube, they stream a video and scrape recommendations, and for Amazon, they place products in wish lists and scrape recommendations. The complexity of these tasks depends on the availability of APIs or the stability of a service's page formats. Outputs col-

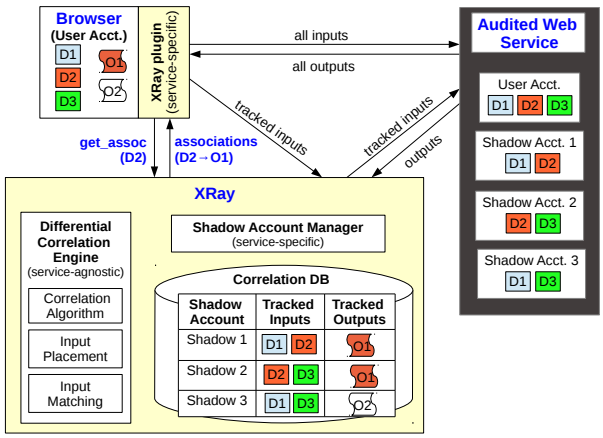


Figure 2: **The XRay Architecture.**

lected from the Web service are placed into a *Correlation Database* (DB), which maps shadow accounts to their input sets and output observations. Figure 2 shows a particular assignment of tracked inputs across three shadow accounts. For example, Shadow 1 has inputs D_1 and D_2. The figure also shows the outputs collected for each shadow account. Output O_1 appears in Shadows 1 and 2 but not in 3; output O_2 appears in Shadow 3 only.

Differential Correlation Engine. This engine, XRay's service-agnostic "brain," leverages the data collected in the Correlation DB to infer input/output associations. When new outputs from shadow accounts are added into the Correlation DB, the engine attempts to diagnose them using a *Correlation Algorithm*. We developed several such algorithms and describe them in §4.3. This process, potentially time-consuming process, is done as a background job, asynchronously from any user request. In Figure 2, differential correlation might conclude that D_2 triggers O_1 because O_1 appears consistently in accounts with that D_2. It might also conclude that O_2 is *untargeted* given inconsistent observations. The engine saves these associations in the Correlation DB.

When the plugin makes a `get_assoc` request, the Correlation Engine looks up the specified output in its DB and returns any pre-computed association. If no output is found, then the engine replies *unknown* (e.g., if an ad never appeared in any shadow account or there is insufficient information). Periodic data collection, coupled with an online update of correlation model parameters, minimizes the number of unknown associations. Our experience shows that collecting shadow account outputs in Gmail every ten hours or so yielded few unknown ads.

While the preceding example is simple, XRay can handle complex challenges occurring in practice. First, outputs are never consistently seen across all shadow accounts containing the input they target. We call this the *limited-coverage* problem; XRay handles it by placing each data input in more shadow accounts. Second, an output may have been triggered by one of several targeted inputs (e.g., multiple emails on the same topic may cause related ads to appear), a problem we refer to as *overlapping-inputs*. This exacerbates the number of accounts needed, since it diminishes the differential signal we receive from them. XRay uses robust, service-agnostic mechanisms and algorithms to match overlapping inputs, place them in the same accounts, and detects their use as a group.

Organization. The remainder of this section describes the Differential Correlation Engine. After constructing it for Gmail, we applied it as-is for Amazon and YouTube, where it achieved equally high accuracy and scalability despite observable differences in how targeting works on these three services. After establishing notations and formalizing our assumptions (§4.2), we describe multiple correlation algorithms, which build up to our self-tuning correlation algorithm that made this adaptation convenient (§4.3). §4.4 describes our input matching.

4.2 Notation and Assumptions

We use f to denote the black-box function that represents the service (e.g., Gmail) associating inputs D_is (e.g., the emails received and sent) to targeted outputs O_ks (e.g., ads). Other inputs are either ignored by XRay, known only to the targeting system, or under no known control. We assume they are independent or fixed, captured in the randomness of f.

We assume that f decides targeting using: (1) a single input (e.g., show O_k if D_4 is in the account), (2) a conjunctive combination of inputs (e.g., show O_k if D_5 *and* D_8 are in the account), or (3) a disjunctive combination of the previous (e.g., show O_k if (D_5 *and* D_8) are in the account *or* if D_4 is in the account). We refer to conjunctive and disjunctive combinations as AND and OR combinations, respectively, and assume that their is bounded by a maximum *input size*, r. This corresponds to the preceding definition of behavioral targeting from §3.2. Contextual targeting will always be a single-input (size-one) combination.

Our goal is to decide whether f produced each output O_k as a reaction to a bounded-size combination of the D_is. We define as *untargeted* any ad that is not targeted against any combination of D_is, though in reality the ad could be targeted against untracked inputs. We denote untargeting as $D_\emptyset$, meaning that the ad is targeted against the "void" email. Our algorithms compute the most likely combination from the N inputs that explains a particular set of observations, $\vec{x}$, obtained by XRay.

We define three probabilities upon which our algorithms and analyses depend. First, the *coverage*, p_{in}, is the probability that an account j containing the input D_i targeted by a particular ad, will see that ad at least once.

Second, an account j' lacking input D_i will see the ad with a smaller probability, p_{out}. Third, if the ad is not behaviorally targeted, it will appear in each account with the same probability, $p_\emptyset$. We assume that $p_{\text{in}}, p_\emptyset, p_{\text{out}}$ are constant across all emails, ads, and time, and that p_{out} is strictly smaller than p_{in} (bounded noise hypothesis).

Finally, we consider all outputs to be independent of each other across time. §8 discusses the implications.

4.3 Correlation Algorithms

A core contribution of this paper is our service-agnostic, self-tuning differential correlation algorithm, which requires only a logarithmic number of shadow accounts to achieve high accuracy. We wished not only to validate this result experimentally, but also to prove it theoretically in the context of our assumptions. This section constructs the algorithm in steps, starting with a naïve polynomial algorithm that illustrates the scaling challenges. We then define a base algorithm using set intersections and prove that it has the desired logarithmic scaling properties; it has parameters which, if not carefully chosen, can lead to poor results. We therefore extend this base algorithm into a self-tuning Bayesian model that automatically adjusts its parameters to maximize correctness.

4.3.1 Naïve Non-Logarithmic Algorithm

An intuitive approach to differential correlation is to create accounts for every combination of inputs, gathering maximum information about their behaviors. With a sufficient number of observations, one could expect to detect which accounts, and hence which subsets of inputs, target a particular ad. Unfortunately, this method requires a number of accounts that grows *exponentially* as the number of items N to track grows. When restricting the size of combinations to r, as we do in XRay, the number of accounts needed is *polynomial* (in $O(N^r)$), or *linear* if we study unique inputs only. Even a linear number of accounts in the number N of inputs remains impractical to scale to large input sizes (e.g., a mailbox).

4.3.2 Threshold Set Intersection

We now show that it is possible to infer behavioral targeting using no more than a *logarithmic* number of accounts as a function of the number of inputs. Specifically, we prove the following theorem:

Theorem 1 *Under §4.2 assumptions, for any $\varepsilon > 0$ there exists an algorithm that requires $C \times \ln(N)$ accounts to correctly identify the inputs of a targeted ad with probability $(1-\varepsilon)$. The constant C depends on ε and the maximum size of combinations r ($O(r2^r \log(\frac{1}{\varepsilon}))$).*

To demonstrate the theorem, we define the *Set Intersection Algorithm* and prove that it has the correctness and scaling properties specified in the theorem. Given

```
// Set Intersection Algo:
// Runs with each collected ad.
In: Output O_k (e.g. an ad).
Params: MIN_ACTIVE_ACCTS, THRESHOLD.
Out: Targeted input combination.
// Step 1: Compute active accounts.
A_k = the accounts that see ad O_k.
if |A_k| < MIN_ACTIVE_ACCTS
    return ∅
end
// Step 2: Create input combination hypothesis.
targeted_set = ∅
foreach input D_i do
    if (number of A_k containing D_i) / |A_k| > THRESHOLD
        targeted_set += D_i
    end
end
// Step 3: Verify it is a real combination.
if (number of A_k containing entire targeted_set) / |A_k| < THRESHOLD
    return ∅
end
// targeted_set triggered the output.
return targeted_set
```

Figure 3: **The Set Intersection Algorithm.** Can be proven to predict targeting correctly under certain assumptions with a logarithmic number of accounts.

that outputs will appear more often in accounts containing the targeting inputs, the core of the algorithm is to determine the set of inputs appearing in the highest number of accounts that also see a given ad. This paper describes a basic version of the algorithm that makes some simplifying assumptions and provides a brief proof sketch. The detailed proof and complete algorithm are described in Appendix.

Algorithm. The algorithm relies on a randomized placement of inputs into shadow accounts, with some redundancy to cope with imperfect coverage. We thus pick a probability, $0 < \alpha < 1$, create $C\ln(N)$ shadow accounts, and place each input D_i randomly into each account with probability α. Figure 3 shows the Set Intersection algorithm for a set of observations, $\vec{x}$. Given an output O_k collected from the user account, we compute the set of *active accounts*, A_k, as those shadow accounts that have seen the output (Step 1). We then compute the set of inputs that appear in at least a threshold fraction of active accounts; this set is our candidate for the combination being targeted by the ad (Step 2). Finally, we check that the entire combination is in a threshold fraction of the active accounts (Step 3). Theoretically, we prove that there exists a threshold for which the algorithm is arbitrarily correct with the available $C\ln(N)$ accounts. Practically, this threshold must be tuned experimentally to achieve

// Bayesian Prediction Alg:
// Runs with each collected ad.
In: Output O_k (e.g. an ad).
Out: Targeted input.
// Compute probabilities.
foreach input D_i **do**
  $\mathbb{P}[D_i|\vec{x}]$ = bayes($\mathbb{P}[\vec{x}|D_i]$)
end
// Compute untargeted prob.
$\mathbb{P}[D_\emptyset|\vec{x}]$ = bayes($\mathbb{P}[\vec{x}|D_\emptyset]$)
// Return event with max prob.
return D_i with max $\mathbb{P}[D_i|\vec{x}]$

// Parameter Learning Alg:
// Runs periodically.
// Initialize params (arbitrary).
$p_{in} = .7, p_{out} = .01, p_\emptyset = .1$
do
  foreach output O_k **do**
    Run Bayesian Prediction.
  end
  Update p_{in}, p_{out}, $p_\emptyset$ from predictions.
until p_{in}, p_{out}, $p_\emptyset$ converge
end

Figure 4: **Bayesian Correlation.** Left: Bayesian prediction algorithm for behavioral targeting. Right: typical iterative inference process to learn parameters.

good accuracy on every service – a key reason for our Bayesian enhancement in §4.3.3.

Correctness Proof Sketch. The proof shows that if there were targeting, every non-targeting input would have a vanishingly small probability to be in a significant fraction of the active accounts. Let us call S the set of inputs contained in a significant fraction of the active accounts. Without targeting, these inputs would be present in the accounts by mere chance. Since inputs are independently distributed into the accounts, we show that the probability of S not being empty decreases exponentially with the number of active accounts (through Chernoff bounds). With targeting, we show that with high probability no other input than the explaining combination is in S, because of the bounded noise hypothesis. Appendix A.2 provides further proof details.

The proofs and algorithm included in this paper work only for conjunctive combinations (e.g., D_1 and D_2, see §4.2). The theory, however, can be extended to disjunctive combinations (e.g., (D_1 and D_2) or D_5), but the algorithm for detecting such combinations is more complex and relies on a recursive argument: if we find one combination from the disjunction, then the active accounts that include this combination define a context where the combination appears non-targeting because it is everywhere. If we recursively apply our algorithm in this context, we can detect the second combination in the disjunction, then the third, etc (see Appendix).

4.3.3 Self-Tuning Bayesian Algorithm

The Set Intersection algorithm provides a good theoretical foundation; however, it requires parameters be tuned and applies only to behavioral targeting, not contextual targeting. Thus, we include in XRay a more robust, self-tuning version that leverages a Bayesian algorithm to adjust parameters automatically through iterated inference. Our algorithm relies on three models: one that predicts behavioral targeting, one that predicts contextual targeting, and one that combines the two.

Behavioral Targeting. The Bayesian behavioral targeting model uses the same random assignment as the Set Intersection algorithm, and it leverages the same information from the shadow account observations, $\vec{x}$. It counts the observations x_j of ad O_k in an account j as a binary signal: if the ad has appeared at least once in account j, we count it once; otherwise we do not count it. Briefly, the Bayesian model is a simple generative model that simulates the audited service given some targeting associations (e.g., D_i triggers O_k). It computes the probability for this model to generate the outputs we do observe for every targeting association. The most likely association will be the one XRay returns.

In more detail if the ad were targeted towards D_i, then an account j containing D_i would see this ad at least once with a *coverage* probability p_{in}; otherwise, it would miss it with probability $(1-p_{\text{in}})$. An account j' without input D_i would see the ad with a smaller probability, p_{out}, missing it with probability $(1-p_{\text{out}})$. If the ad were not behaviorally targeted, it would appear in each account with the same probability, $p_\emptyset$. If we define A_k as the set of active accounts that have seen the ad, and A_i as the set of accounts that contain email D_i, then we have the following definitions for the probabilities:

$$\begin{aligned}\mathbb{P}[\vec{x}|D_i] = &\ (p_{\text{in}})^{|A_i\cap A_k|}(1-p_{\text{in}})^{|A_i\cap\bar{A_k}|}\\ &\times(p_{\text{out}})^{|\bar{A_i}\cap A_k|}(1-p_{\text{out}})^{|\bar{A_i}\cap\bar{A_k}|},\\ \mathbb{P}[\vec{x}|D_\emptyset] = &\ (p_\emptyset)^{|A_k|}(1-p_\emptyset)^{|\bar{A_k}|},\end{aligned}$$

where $D_\emptyset$ designates the untargeted prediction.

The preceding formula has an interesting interpretation that is visible if placed in the equivalent form:

$$\begin{aligned}\mathbb{P}[\vec{x}|D_i] = &\ (p_{\text{in}})^{|A_k|}(1-p_{\text{out}})^{|\bar{A_k}|}\\ &\times\left(\frac{1-p_{\text{in}}}{1-p_{\text{out}}}\right)^{|A_i\cap\bar{A_k}|}\left(\frac{p_{\text{out}}}{p_{\text{in}}}\right)^{|\bar{A_i}\cap A_k|}\end{aligned}$$

From the point of view of the event D_i, an account found in $A_i\cap\bar{A_k}$ is a false positive (an ad was expected but was not shown). This should lower the probability, especially when the *coverage* p_{in} is close to 1. Inversely, an account found in $\bar{A_i}\cap A_k$ acts as a false negative (we observed an ad where we did not expect it), which should decrease the probability, especially when p_{out} is close to 0.

These formulas let us infer the likelihood of event D_i according to Bayes' rule: $\mathbb{P}[A|B] = \frac{\mathbb{P}[B|A]\times\mathbb{P}[A]}{\mathbb{P}[B]}$. Figure 4 shows two algorithms. First, the prediction algorithm (left) predicts the targeting of O_k by computing the probabilities defined above, applying Bayes' rule, and returning the input with the maximum probability. Second, the parameter learning algorithm (right) computes the variables that those probabilities depend upon

(p_{in}, p_{out}, and $p_{\emptyset}$) using an iterative process. It repeatedly runs the prediction algorithm for all outputs and re-computes p_{in}, p_{out}, and $p_{\emptyset}$ based on the predictions. It stops when the variables converge (i.e., their variation from one iteration to another is small).

Contextual Targeting. Contextual targeting is more straightforward since it uses content shown next to the ad. XRay also uses Bayesian inference and defines the observations as how many times ad O_k is seen next to email D_i. Our causal model assumes imperfect coverage: if this ad were contextually targeted towards D_i, it would occur next to that email with probability $p_{\text{in}} < 1$ and next to any other email with probability p_{out}. Alternatively, if the ad were untargeted, our model predicts it would be shown next to any email with probability $p_{\emptyset}$. Hence, $\mathbb{P}[\vec{x}|D_i] = (p_{\text{in}})^{x_i}(p_{\text{out}})^{\sum_{i' \neq i} x'_i}, \mathbb{P}[\vec{x}|D_{\emptyset}] = (p_{\emptyset})^{\sum_i x_i}$. For this model, parameters are also automatically computed by iterated inference.

Composite Model (XRay). The contextual and behavioral mechanisms were designed to detect different types of targeting. To detect both types, XRay must combine the two scores. We experimented with multiple combination functions, including a decision tree and the arithmetic average, and concluded that the arithmetic average yields sufficiently good results. XRay thus defines the *composite model* that averages scores from individual models, and we demonstrate in §6.3 that doing so yields higher recall for no loss in precision.

4.4 Input Matching and Placement

Our design of differential correlation, along with our logarithmic results for random input placement, relies on the fundamental assumption that the probability of getting an ad O_1 targeted at an input D_1 in a shadow account that lacks D_1 is vanishingly small. However, when inputs attract the same ads (a.k.a., overlapping inputs), a naive input placement can contradict this assumption. Imagine a Gmail account with multiple emails related to a Caribbean trip. If placement includes Caribbean emails in every available shadow account, related ads will appear in groups of accounts with no email object in common. XRay will thus classify them as untargeted. Our Amazon experiments showed XRay's recall dropping from 97% to 30% with overlapping inputs (§6.5).

To address this problem, XRay's Input Matching module identifies similar inputs and directs the Placement Module to co-locate them in the same shadow accounts. The key challenge is to identify similar inputs. One method is to use content analysis (e.g., keywords matching), but this has limitations. First, it is not service agnostic; one needs to reverse engineer complex and ever-changing matching schemes. Second, it is hard to apply to non-textual media, such as YouTube videos.

In XRay, we opt for a more robust, systems technique rooted in the key insight that we can deduce similar inputs from contextual targeting. Intuitively, inputs that trigger similar targeting from the Web service should attract similar outputs in their context. The Input Matching module builds and compare inputs' *contextual signatures*. Contextual signature similarity is the distance between inputs (e.g., email) in a Euclidean space, where each output (e.g., ad) is a dimension. The coordinate of an email in this dimension is the number of times the ad was seen in the context of the email. XRay then forwards close inputs to the same shadow accounts. Once the placement is done, behavioral targeting against that email's group can be inferred effectively.

This input matching mechanism differs fundamentally from any content analysis technique, such as keyword matching, because it groups inputs *the same way the Web service does*.[2] It is robust and very general: we used it on both Gmail and Amazon without changing a single line of code to change.

5 XRay-based Tools

To evaluate XRay's extensibility, we instantiated it on Gmail, YouTube, and Amazon. The engine, about 3,000 lines of Ruby, was first developed for Gmail. We then extended it to YouTube and Amazon, without any changes to its correlation algorithms. We did need to do minor code re-structuring, but the experience felt turn key when integrating a new service into the correlation machinery.

Building the full toolset required non-trivial coding effort, however. Instantiating XRay for a specific Web service is a three-step process. First, the developer instantiates appropriate data models (less than 20 code lines for our prototypes). Second, she implements a service-specific shadow account manager and plugin; care must be taken not be too aggressive to avoid adversarial service reactions. While these implementations are conceptually simple, they require some coding; our Amazon and YouTube account managers were built by two graduate students new to the project, and have around 500 lines of code. Third, the developer creates a few shadow accounts for the audited service and runs a small exploratory experiment to determine the service's coverage. XRay uses the coverage to estimate the number of shadow accounts needed for a given input size. All other parameters are self-tuned at runtime.

6 Evaluation

We evaluated XRay with experiments on Gmail, Amazon, and YouTube. While Amazon and YouTube provide ground truth for their targeting, Gmail does not. We therefore manually labeled ads on Gmail and measured

[2] We call this method "monkey see, monkey do" because we watch how the service groups inputs and group them similarly.

Service	Sample Category	Sample Input
Gmail	Electronics	Email: "We need to buy that TV."
	Accounting	Email: "Know a good tax accountant?"
Amazon	Toys & Games	Product: Crayola 64 Crayons
	Health & Personal	Product: Waterpik Ultra Flosser
YouTube	Skin Health	Video: Organic Acne Treatment
	Books & Literature	Video: Rowling's Love of Hufflepuff

Figure 5: **Sample Inputs and Categories.** In total, we developed inputs in 64 categories for Amazon and YouTube and in 51 categories for Gmail.

Ad Keyword	Targeted Email	Detected by XRay?	XRay Scores	# Accounts & Displays
Chaldean Poetry	Like Chaldean Poetry?	Yes	0.99, 1.0	13/13, 1588/1622
Steampunk	Fan of Steampunk?	Yes	0.99, 1.0	13/13, 888/912
Cosplay	Discover Cosplay.	Yes	0.99, 1.0	13/13, 440/442
Falconry	Learn about Falconry.	Yes	0.99, 1.0	13/13, 1569/1608

Figure 6: **Self-Targeted Ads.** Fourth column shows XRay's correlation scores X, Y, the (Bayesian) Behvioral and Contextual scores, respectively. Fifth column shows raw behavioral and contextual data for better interpretation: X/Y, Z/T means that the ad was seen in X active accounts that contain the targeted email out of a total of Y active accounts; the ad was shown Z times in the context of the targeted email out of a total of T times.

XRay's accuracy, as described in §6.1 and validated in §6.2. We sought answers to four questions:

Q1 How accurate are XRay's inference models? (§6.3)
Q2 How does XRay scale with input size? (§6.4)
Q3 Does input matching reduce overlap? (§6.5)
Q4 How useful is XRay in practice? (§6.6)

6.1 Methodology

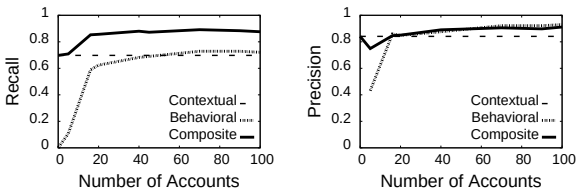
We evaluated XRay with experiments on Gmail, Amazon, and YouTube. For inputs, we created a workload for each service by selecting topics from well-defined categories relevant for that service. For Gmail and YouTube, we crafted emails and selected videos based on AdSense categories [17]; for Amazon, we selected products from its own product categories [2]. Figure 5 shows several sample categories and sample inputs in each. We used these categories for most of our experiments (§6.3–§6.5). We used these categories to create two types of workloads: (1) a non-overlapping workload, in which each data item belonged to a distinct category, and (2) an overlapping workload, with multiple data items per category (described in §6.5).

To assess XRay's accuracy, we needed the ground truth for associations. Amazon and YouTube provide it for their recommendations. For instance, Amazon provides a link "Why recommended?" for each recommendation; when clicked, it shows an explanation of the form "The [Coloring Book] is recommended because your wish list includes [Crayola Crayons Set]." For Gmail, we manually labeled ads based on our personal assessment. The ads for different experiments were labeled by different people, generally project members. A non-computer scientist labeled the largest experiment (51 emails).

We evaluate two metrics: (1) *recall*, the fraction of positive associations labeled as such, and (2) *precision*, the fraction of correct associations. We define *high accuracy* as having both high recall and high precision.

6.2 Sanity-Check Experiment

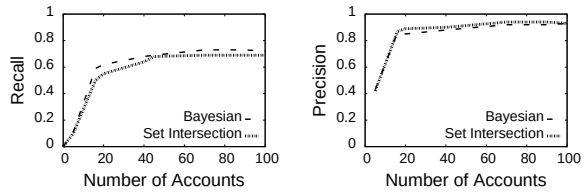
To build intuition into XRay's functioning, we ran n simple sanity-check experiment on Gmail. Recall that, unlike Amazon and YouTube, Gmail does not provide any ground truth, requiring us to manually label associations, a process that can be itself faulty. Before measuring XRay's accuracy against labeled associations,



(a) **Recall** (b) **Precision**

Figure 7: **Bayesian Model Accuracy.** Recall and precision for each of the three Bayesian models vs. shadow account number, using the Bayesian algorithm. XRay needed 16 accounts to reach the "knee" with high recall and precision.

we checked that XRay can detect associations for our own ads, whose targeting we control. For this, we strayed away from the aforementioned methodology to create a highly controlled experiment. We posted four Google AdWords campaigns targeted on very specific keywords (Chaldean Poetry, Steampunk, Cosplay, and Falconry), crafted an inbox that included one email per keyword, and used XRay to recover the associations between our ads and those emails. In total, we saw our ads 1622, 912, 442, and 1608 times, respectively, across all accounts (shadows and master). Figure 6 shows our results. After one round of ad collection (which involved 50 refreshes per email), XRay correctly associated all four ads with the targeted email. It did so with very high confidence: composite model scores were 0.99 in all cases, with very high scores for both contextual and behavioral models. The figure also shows some of the raw contextual/behavioral data, which provides intuition into XRay's perfect precision and recall in this controlled experiment. We next turn to evaluating XRay in less controlled environments, for which we use the workloads and labeling methodology described in §6.1.



(a) **Recall** (b) **Precision**

Figure 8: **Bayesian vs. Set Intersection Comparison.** Recall and precision for detecting *behavioral* targeting with each algo.

6.3 Accuracy of XRay's Inference Models (Q1)

To assess the accuracy of XRay's key correlation mechanisms (Bayesian behavioral, contextual, and composite), we measured their recall and precision under non-overlapping workloads. Figures 7(a) and 7(b) show how these two metrics varied with the number of shadow accounts for a 20-email experiment on Gmail. The results indicate two effects. First, both contextual and behavioral models were required for high recall. Of the 193 distinct ads seen in the user account, 121 (62%) were targeted, and XRay found 109 (90%) of them, a recall we deem high. Of the associations XRay found, 37% were found by only one of the models: 15 by the contextual model only, and 24 by the behavioral model only. Thus, both models were necessary, and composing them yielded high recall. Our Amazon and YouTube experiments (which provide ground truth) yielded very similar results: on a 20-input experiment, we reached over 90% recall and precision with only 8 and 12 accounts, respectively.

Second, the composite model's recall exhibited a knee-shaped curve for increasing shadow account numbers, with a rapid improvement at the beginning and slow growth thereafter. With 16 accounts, XRay exceeded 85% recall; increasing the number of accounts to 100 yielded a 1.9% improvement. Precision also remained high (over 84%) past 16 accounts. We define the *knee* as the minimum number of accounts needed to reap most of the achievable recall and precision.

We also wished to compare the accuracy of the Bayesian algorithm, which conveniently self-tunes its parameters, to the parameterized Set Intersection algorithm. We manually tuned the latter as best as we could. Figures 8(a) and 8(b) show the recall and precision for detecting behavioral targeting with the two methods for a non-overlapping workload. The two algorithms performed similarly, with the Bayesian staying within 5% of the manually tuned algorithm. We also tested the algorithms on an Amazon dataset, and using a version of the Set Intersection algorithm with empirical optimizations. The conclusion holds: the Bayesian algorithm, with self-tuned parameters, performs as well as the Set Intersection technique with manually tuned parameters. We focus the remainder of this evaluation on the Bayesian algorithm.

6.4 Scalability of XRay with Input Size (Q2)

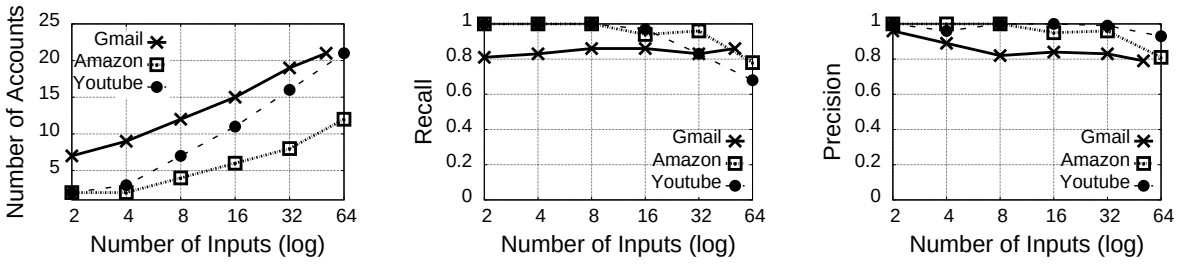
A main contribution of this paper is the realization that, under certain assumptions, the number of accounts needed to achieve high accuracy for XRay scales logarithmically with the number of tracked inputs. We have proven that under certain assumptions, the Set Intersection algorithm scales logarithmically. This theoretical result is hard to extend to the Bayesian algorithm, so we evaluated it experimentally by studying three metrics with growing input size: the number of accounts required to reach the recall knee and the value of recall/precision at this knee. Figures 9(a), 9(b) and 9(c) show the corresponding results for Gmail, YouTube and Amazon. For Gmail, the number of accounts necessary to reach the knee increased less than 3-fold (from 8 to 21) as input size increased more than 25-fold (from 2 to 51). For Amazon and YouTube, the increases in accounts were 6- and 8-fold respectively, for a 32-fold increase in input size. In general, the roughly linear shapes of the log-x-scale graphs in Figure 9(a) confirm the logarithmic increase in the number of accounts required to handle different inputs. Figure 9(b) and 9(c) confirm that the "knee number" of accounts achieved high recall and precision (over 80%).

What accounts for the large gap between the number of accounts needed for high accuracy in Gmail versus Amazon? For example, tracking a mere two emails in Gmail required 8 accounts, while tracking two viewed products in Amazon needed 2 accounts. The distinction corresponds to the difference in coverage exhibited by the two services. In Gmail, a targeted ad was typically seen in a smaller fraction of the relevant accounts compared to a recommended product in Amazon. XRay adapted its parameters to lower coverage automatically, but it needed more accounts to do so.

Overall, these results confirm that our theoretical scalability results hold for real-world systems given carefully crafted, non-overlapping input workloads. We next investigate how more realistic overlapping input workloads challenge the accuracy of our theoretical models and how input matching – a purely systems technique – helps address this challenge.

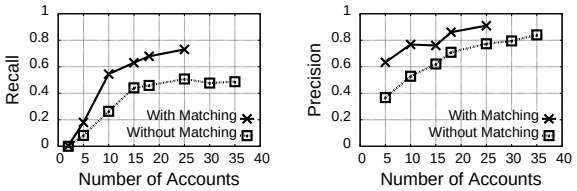
6.5 Input Matching Effectiveness (Q3)

To evaluate XRay's accuracy with overlapping inputs, we infused our workloads with multiple items from the same category (e.g., multiple emails targeting the same AdSense categories on Gmail and multiple products in the same category in Amazon). For the Gmail experiments, we (as users) could not tell when Gmail



(a) **Scalability with Input Size** (b) **Recall with Input Size** (c) **Precision with Input Size**

Figure 9: **Scalability.** (a) Number of accounts required to achieve the knee accuracy for varied numbers of inputs. (b), (c) Recall/precision achievable with the number of accounts in (a). Behavioral uses the Bayesian algorithm.



(a) **Recall** (b) **Precision**

Figure 10: **Input Matching effectiveness.** Behavioral (Bayesian) recall and precision in Gmail with overlapping inputs, with and without Matching.

targeted a specific email from a group of similar emails. We therefore ran two different types of experiments: (1) a controlled, albeit unrealistic, one for Gmail, and (2) a more realistic one for Amazon.

For Gmail, our controlled experiment replicated various emails *identically* in a user's inbox: 1 email was replicated 4 times, 2 emails 3 times, 4 emails 2 times, and 12 were single, for a total of 30 emails. This end-of-a-spectrum workload demonstrates how matching works ideally. XRay matched *all* redundant emails correctly. More importantly, Figures 10(a) and 10(b) show XRay's precision/recall with and without matching-aware placement for XRay's behavioral model, the only model improved by matching. Without input matching, XRay struggled to find differential signals: even with 35 shadow accounts for a 30-email experiment, recall was only 48%. With input matching, XRay's correlation model drew a stronger signal from each account and attained close to 70% recall for 16 accounts.

For Amazon, we created an overlapping workload by selecting three *distinct* products in each of six product categories (e.g., from the Outdoor & Cycling category, we selected a helmet, pedals, and shoes). With a total workload of 18 products, XRay's input matching matched all but one item (shoes) into its correct group. With the new grouping, XRay's recall improved by a factor of 3 (from 30% to 93%) compared to the no-matching case for 18 products with 10 accounts; precision was 2.6 times higher (from 34% to 88%).

These results demonstrate that XRay's matching scheme is both portable across Web services and essential for high accuracy with overlapping workloads.

6.6 Anecdotal Use Experience (Q4)

To gain intuition into XRay's value in practice, we ran a small-scale, anecdotal experiment that looked for ads attracted by a few specific topics in Gmail. We created emails focused on a few topics, including cancer, Alzheimer, depression, HIV, race, homosexuality, pregnancy, divorce, debt, and others. Each email consisted of a number of keywords closely related to one topic (e.g., the depression-related email included *depression*, *depressed*, and *sad*; the homosexuality email included *gay*, *homosexual*, and *lesbian*). We then launched XRay/Gmail's ad collection several times at intervals of two days, and examined its targeting associations. Figure 11 shows example XRay associations for each of the topics we considered, along with its confidence scores and some of the raw data behind its scores (see §**??**). We conservatively show only a select few of the ads we gathered, for which XRay's confidence – particularly in behavioral score – was particularly high. While our experiment is too small to draw definitive and detailed conclusions about ad targeting in Gmail, we make three high-level observations from our experience.

First, our small-scale experiment confirms that it is possible to target sensitive topics in users' inboxes. For example, all disease-related emails, except for the HIV-related one, correlated very strongly with various ads. For example, Pregnancy, homosexuality, race, divorce, and debt also attracted ads. Interestingly, our experience suggests that disease- and Overall, we have been surprised . For instance, ads 7/8, 15/16, and 17 target race, sexual orientation and cancer, respectively. The ads we observed were mostly benign or even positive. However, if no keyword in the ad suggested relation

Topic	Targeted Ads	XRay Scores	# Accounts & Displays
Alzheimer	Black Mold Allergy Symptoms? Expert to remove Black Mold.	0.99, 0.05	9/9, 61/198
	Adult Assisted Living. Affordable Assisted Living.	0.99, 0.99	8/8, 12/14
Cancer	Ford Warriors in Pink. Join The Fight.	0.96, 0.98	9/9, 1022/1106
	Rosen Method Bodywork for physical or emotional pain.	0.98, 0.05	7/7, 24/598
Depression	Shamanic healing over the phone.	0.99, 0.99	16/16, 117/117
	Text Coach - Get the girl you want and Desire.	0.93, 0.04	7/7, 31/276
African American	Racial Harassment? Learn your rights now.	0.99, 0.2	10/10, 851/5808
	Racial Harassment, Hearing racial slurs?	0.99, 0.2	10/10, 627/7172
Homosexuality	SF Gay Pride Hotel. Luxury Waterfront.	0.99, 0.1	9/9, 50/99
	Cedars Hotel Loughborough, 36 Bedrooms, Restaurant, Bar.	0.96, 1.0	8/8, 36/43
Pregnancy	Ralph Lauren Apparel. Official Online Store.	0.99, 0.6	10/10, 85/181
	Clothing Label-USA. Best Custom Woven Labels.	0.99, 1.0	9/9, 14/14
	Find Baby Shower Invitations. Get Up To (60% Off) Here!	0.99, 1.0	9/9, 22/22
Divorce	Law Attorneys specializing in special needs kids education.	0.99, 0.99	9/9, 635/666
	Cerbone Law Firm, Helping Good People Thru Bad Times	0.99, 1.0	10/10, 94/94
Enough with this marriage	Maui Beach Weddings Serving. Affordable ceremonies.	0.99, 0.0	7/7, 2/728
	Romantic Wedding. Ceremony Planning.	0.99, 0.04	8/8, 4/31
Debt	Take a New Toyota Test Drive, Get a $50 Gift Card On The Spot.	0.99, 0.9	7/7, 58/65
	Great Credit Cards Search. Apply for VISA, MasterCard...	0.99, 0.0	9/9, 151/2358
	Stop Creditor Harassment, End the Harassing Calls.	0.99, 0.96	8/8, 256/373

Figure 11: **Example Targeted Ads Uncovered by XRay.** Columns three and four show the same data as columns four and five in Figure 6.

with sensitive topics (e.g., ad 17), a users clicking on the ad may not realize that they could be disclosing private information to advertisers. This case inspired Scenario 2 in §2.1. Suppose an insurance company wanted to gain insight into pre-existing conditions of its customers before signing them up. It could create two ad campaigns – one that targets cancer and another youth – and assign different URLs to each campaign. It could then offer higher premium quotes to visitors who come through the cancer-related ads to discourage them from signing up while offering lower premium quotes to those who come through youth-related ads.

Second, our experiments suggest that some advertisers use targeting capabilities to focus their campaigns on vulnerable subgroups. In one case, a shamanic phone healing service heavily targeted keywords in our depression email (ad 1). In another case, our "broke" email attracted many personal loan offers (ad 10) and deals with high scam potential (ad 11). Whether these practices are fair is beyond the scope of this work, but we believe that informed users are empowered users.

Third, many cases, targeting did not have a good semantic understanding of the emails. For instance, an email about divorce, that also contained the word marriage received many ads about wedding ceremonies, like ad 13. The TV Show email also contained the word "watch" and hence got targeted heavily by watch brands (ads 5 and 6). Context does not seem to be used to disambiguate specific keywords. We could not tell if the targeting algorithm were incapable of such semantic analysis, or if the feature were not exposed or used by advertisers.

These results show probable correlations, although we cannot be sure that they denote targeting. However, we selected only those cases with strong evidence of correlation between email and ad.

6.7 Summary

Our evaluation results show that XRay supports fine-grained, accurate data tracking in popular Web services, scales well with the size of data being tracked, is general and flexible enough to work efficiently for three Web services, and robustly uses systems techniques to discover associations when ad contents provide no indication of them. We next discuss how XRay meets its last goal: robustness against honest-but-curious attackers.

7 Security Analysis

As stated in §3.3, two threat models are relevant for XRay and applicable to different use cases. First, an *honest-but-curious* Web service does not attempt to frustrate XRay, but it could incorporate defenses against typical Web attacks, such as DDoS or spam, that might interfere with XRay's functioning. Second, a *malicious service* takes an adversarial stand toward XRay, seeking to prevent or otherwise disrupt its correlations. Our current XRay prototype is robust against the former threat and can be extended to be so against the latter. In either case, third-party advertisers are untrusted and can attempt to frustrate XRay's auditing. We discuss each threat in turn.

Non-Malicious Web Services. Many services incorporate protections against specific automated behaviors. For example, Google makes it hard to create new accounts, although doing so remains within reach. Moreover, many services actively try to identify spammers and click fraud. Gmail includes sophisticated spam filtering mechanisms, while YouTube rate limits video viewing to prevent spam video promotion. Finally, many services rate limit access from the same IP address.

XRay-based tools must be aware of these mechanisms and scale back their activities to avoid raising red flags. For example, our XRay-based tools for Gmail, YouTube, and Amazon rate limit their output collection in the shadow accounts. More importantly, XRay's very design is sensitive to these challenges: by requiring as few accounts as possible, we minimize: (1) the load on the

service imposed by auditing, and (2) the amount of input replication across shadow accounts. Moreover, XRay's workloads are often atypical of spam workloads. Our XRay Gmail plugin sends emails from one to a few other accounts, while spam is sent from one account to many other accounts.

Malicious Third-Party Advertisers. Third-party advertisers have many ways to obfuscate their targeting from XRay, particularly if it may arouse a public outcry. First, an advertiser could purposefully weaken its targeting by, for example, targeting the same ad 50% on one topic and 50% on another topic. This weakens input/output correlation and may cause XRay to infer untargeting. However, it also makes the advertisers' targeting less effective and potentially more ambiguous if their goal is to learn specific sensitive information about users. Second, an advertiser might target complex combinations of inputs that XRay's basic design cannot discover. We show in Appendix an example of how advertisers might use it, and that our theoretical results extend to those combinations. It also extends our theoretical models so they can detect targeting on linear combinations with only a constant factor increase in the number of accounts. We plan to incorporate and evaluate these extensions in a future prototype.

Malicious Web Services. A malicious service could identify and disable shadow accounts. Identification could be based on abnormal traffic (successive reloads of email pages), data distribution within accounts (one account with lots of data, several others with subsets of it), and perhaps more. XRay could be extended to add randomness and deception (e.g., fake emails in shadow accounts, vary email copies). More importantly, a collaborative approach to auditing, in which users contribute their ads and input topics in an privacy-preserving way is a promising direction for strengthening robustness against attacks. Web services cannot, after all, disable legitimate user accounts to frustrate auditing. We plan to pursue this direction in future work.

8 Discussion

XRay takes a significant step toward providing data management transparency in Web services. As an initial effort, it has a number of limitations. First, both the Set Intersection and Bayesian algorithms assume independent targeting across accounts and over time. In reality, ad targeting is not always independent across either. For example, advertisers set daily ad budgets. When the budget runs out, an ad can stop appearing in accounts mid-experiment even though it has the targeted attributes. The system might incorrectly assume that no targeting is taking place, when it could resume the next day. XRay takes reduced coverage into account, but differences between ads can let some targeting pass unnoticed. XRay does not currently account for these dependencies, but estimating their impact is an important goal for future work.

Second, we assume that targeting noise is bounded and smaller than the targeting signal. While this condition seems to hold on the evaluated services, other services making more local decisions may be harder to audit. For instance, a social network (e.g., Facebook) could target ads based on friends' information. The noise created by the environment could potentially be as high as the targeting signal. A future solution might be to create shadow accounts with the same friends or shadows of friends.

Third, XRay uses Web services atypically. To the best of our knowledge, it does not violate any terms of service. It does, however, collect ads paid for by advertisers to detect correlation. Ad payment is per impression and pay per click. The former is vastly less expensive than the latter [34]. XRay creates false impressions only but never clicks on ads. A back-of-the-envelope calculation using impression pricing from [34] of $0.6/thousand impressions reveals that XRay's cost should be minimal: at most 50 cents per ad for our largest experiments.

Despite these limitations, XRay has proven itself useful for many needs, particularly in an auditing context. An auditor can craft inputs that avoid many of these limitations. For example, emails can be written to avoid as much overlap as possible and keep the size of inputs used for targeting within reasonable bounds. We hope that XRay's solid correlation components will streamline much-needed investigations – by researchers, journalists, or the FTC – into how personal data is being used.

9 Related Work

While §2.2 covered Web data protection and auditing related works, we next cover other related topics. Our work relates to recent efforts to measure various forms of personalization, such as search [21, 46], pricing [33], and ad discrimination [41]. These efforts start from the assumption that personalization has a dark side (e.g., censorship [46], filter bubble [21]). They generally employ a methodology similar in spirit to differential correlation, but their goals differ from ours. They aim to quantify *how much* output is personalized and what *type* of information is used overall (be it a user's geography, demographic attributes, or past behavior). In contrast, XRay seeks to provide fine-grained diagnosis of which *input data* generates which personalized results. Through its scaling mechanisms – unique in the personalization and data tracking literature – XRay scales well even when the relevant inputs are many and unknown in advance.

Our work also relates to a growing body of research measuring advertising networks. These networks, notably complex and difficult to crawl [3], are rendered opaque by the need to combat click fraud [9], and have

been shown to be susceptible to leakage [27] and profile reconstruction attacks [6]. As for other personalization, prior studies have focused mostly on macroscopic trends (e.g., What fraction of ads are targeted overall?) [3] or qualitative trends (e.g., Which ads are targeted toward gay males?) [20]. Various studies showed traces – but not a prevalence – of potential abuse through concealed targeting [20] and data exchange between services [45]. These works primarily focus on display advertising, and each distinguishes contextual advertising using a specific classifier with semantic categories obtained from Google's Ad Preferences Managers or another public API [31].

XRay departs significantly from these works. First, since it entirely ignores the content and even the domain of targeting, it is readily applied as-is to ads in Gmail, product recommendations, and videos. Second, while previous methods label ads as "behavioral" in bulk once other explanations fail [31], XRay remains grounded on positive evidence of targeting, and it determines to *which* inputs an output should be attributed. Third, XRay's mechanisms to avoid exponential input placement and deal with overlapping inputs are unprecedented in the Web-data-tracking context. While they resemble *black box* software testing [4], the specific targeting assumption we leverage have, to our knowledge, no prior equivalent.

10 Conclusions

The tracking of personal data usage poses unique challenges. XRay shows for the first time that accurate, *fine-grained* tracking need not compromise portability and scalability. For users who care about *which* piece of their data has been targeted, it offers a unique level of precision and protection. Our work calls for and promotes the best practice of voluntary transparency, while at the same time empowering investigators and watchdogs with a significant new tool for increased vigilance.

11 Acknowledgements

We extend special thanks to our shepherd, Dan Boneh, for his valuable guidance. We also thank the anonymous reviewers and numerous colleagues who have given us feedback, including: Jonathan Bell, Sandra Kaplan, Michael Keller, Yoshi Kohno, Hank Levy, Yang Tang, Nicolas Viennot, and Junfeng Yang. This work was supported by funds from DARPA Contract FA8650-11-C-7190, NSF CNS-1351089, Google, and Microsoft.

References

[1] Adblock plus - surf the web without annoying ads! `https://adblockplus.org`.

[2] I. Amazon. Amazon taxonomy. `http://services.amazon.com/services/soa-approval-category.htm#openCategories`.

[3] P. Barford, I. Canadi, D. Krushevskaja, Q. Ma, and S. Muthukrishnan. Adscape: Harvesting and Analyzing Online Display Ads. *WWW '14: Proceedings of the 23nd international conference on World Wide Web*.

[4] B. Beizer. *Black-Box Testing*. Techniques for Functional Testing of Software and Systems. John Wiley & Sons, May 1995.

[5] D. Boneh, G. Crescenzo, R. Ostrovsky, and G. Persiano. Public Key Encryption with Keyword Search. In *Proc. of the ACM European Conference on Computer Systems (EuroSys)*, pages 506–522. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[6] C. Castelluccia, M. A. Kaafar, and M. Tran. Betrayed by your ads! *PETS'12: Proceedings of the 12th international conference on Privacy enhancing technologies*, 2012.

[7] W. Cheng, Q. Zhao, B. Yu, and S. Hiroshige. Tainttrace: Efficient flow tracing with dynamic binary rewriting. In *Proceedings of the 11th IEEE Symposium on Computers and Communications*. IEEE Computer Society, 2006.

[8] Chrome web store - collusion for chrome. `https://chrome.google.com/webstore/detail/collusion-for-chrome/ganlifbpkcplnldliibcbegplfmcfigp`.

[9] V. Dave, S. Guha, and Y. Zhang. Measuring and fingerprinting click-spam in ad networks. In *SIGCOMM '12: Proceedings of the ACM SIGCOMM 2012 conference on Applications, technologies, architectures, and protocols for computer communication*. ACM Request Permissions, Aug. 2012.

[10] N. Diakopoulos. Algorithmic accountability reporting: On the investigation of black boxes. Tow Center for Digital Journalism, Columbia University. February, 2014.

[11] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. *Technical Report*, 2004.

[12] W. Enck, P. Gilbert, B. gon Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth. TaintDroid: An information-flow tracking system for realtime privacy monitoring on smartphones. In *Proc. of the USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2010.

[13] M. Fredrikson and B. Livshits. RePriv: Re-imagining Content Personalization and In-browser Privacy. *2011 IEEE Symposium on Security and Privacy*, pages 131–146, 2011.

[14] R. Geambasu, T. Kohno, A. Levy, and H. M. Levy. Vanish: Increasing data privacy with self-destructing data. In *Proc. of USENIX Security*, 2009.

[15] C. Gentry. Fully homomorphic encryption using ideal lattices. In *Proc. of the ACM Symposium on Theory of Computing (STOC)*, 2009.

[16] D. B. Giffin, A. Levy, D. Stefan, and D. Terei. Hails: Protecting data privacy in untrusted web applications. *10th Symposium on ...*, 2012.

[17] I. Google. Adsense categories. `https://support.google.com/adsense/answer/3016459`.

[18] V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proc. of the ACM Conference on Computer and Communications Security (CCS)*, 2006.

[19] T. Guardian. Snapchat's expired snaps are not deleted, just hidden — media network - partner zone infosecurity — guardian professional. `http://www.theguardian.com/media-network/partner-zone-infosecurity/snapchat-photos-not-deleted-hidden`.

[20] S. Guha, B. Cheng, and P. Francis. Challenges in measuring online advertising systems. In *IMC '10: Proceedings of the 10th annual conference on Internet measurement*. ACM Request Permissions, Nov. 2010.

[21] A. Hannak, P. Sapiezynski, A. M. Kakhki, B. Krishnamurthy, D. Lazer, A. Mislove, and C. Wilson. Measuring personalization of web search. In *WWW '13: Proceedings of the 22nd international conference on World Wide Web*. International World Wide Web Conferences Steering Committee, May 2013.

[22] A. L. Hughes and L. Palen. Twitter adoption and use in mass convergence and emergency events. *International Journal of Emergency Management*, 2009.

[23] S. Jeff Gould. Safegov.org - google admits data mining student emails in its free education apps. http://safegov.org/2014/1/31/google-admits-data-mining-student-emails-in-its-free-education-apps.

[24] T. W. S. Journal. On orbitz, mac users steered to pricier hotels - wsj.com. http://online.wsj.com/news/articles/SB10001424052702304458604577488822667325882.

[25] T. W. S. Journal. Websites vary prices, deals based on users' information - wsj.com. http://online.wsj.com/news/articles/SB10001424127887323777204578189391813881534.

[26] A. Korolova. Privacy Violations Using Microtargeted Ads: A Case Study. In *ICDM Workshops*, 2010.

[27] A. Korolova. Privacy violations using microtargeted ads: A case study. *Data Mining Workshops (ICDMW), 2010 IEEE International Conference on*, pages 474–482, 2010.

[28] B. Krishnamurthy and C. E. Wills. On the leakage of personally identifiable information via online social networks. In *Proceedings of the 2Nd ACM Workshop on Online Social Networks*, WOSN '09, pages 7–12, New York, NY, USA, 2009. ACM.

[29] J. Lanier. Who owns the future? p112-114. In *Who Owns the Future?* Simon and Schuster, 2013.

[30] Lightbeam for firefox mozilla. http://www.mozilla.org/en-US/lightbeam/.

[31] B. Liu, A. Sheth, U. Weinsberg, J. Chandrashekar, and R. Govindan. AdReveal: improving transparency into online targeted advertising. In *HotNets-XII: Proceedings of the Twelfth ACM Workshop on Hot Topics in Networks*. ACM Request Permissions, Nov. 2013.

[32] C. LLP. Declaration of kyle c. wong in support of google inc.'s opposition to plaintiffs' motion for class certification. http://safegov.org/media/60266/google_gmail_litigation_-_declaration_of_kyle_c.wong.pdf.

[33] J. Mikians, L. Gyarmati, V. Erramilli, and N. Laoutaris. Detecting price and search discrimination on the internet. In *Proceedings of the 11th ACM Workshop on Hot Topics in Networks*, HotNets-XI, pages 79–84, New York, NY, USA, 2012. ACM.

[34] L. Olejnik, T. Minh-Dung, C. Castelluccia, et al. Selling off privacy at auction. In *In Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2013.

[35] R. A. Popa, C. M. S. Redfield, N. Zeldovich, and H. Balakrishnan. Cryptdb: Protecting confidentiality with encrypted query processing. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, SOSP '11, pages 85–100, New York, NY, USA, 2011. ACM.

[36] F. Roesner. Sharemenot. https://sharemenot.cs.washington.edu/.

[37] F. Roesner, T. Kohno, and D. Wetherall. Detecting and defending against third-party tracking on the web. In *NSDI'12: Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation*. USENIX Association, Apr. 2012.

[38] A. Sadilek and H. Kautz. Modeling the impact of lifestyle on health at scale. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*. ACM Request Permissions, Feb. 2013.

[39] Snapchat. http://blog.snapchat.com/.

[40] Snapchat - how snaps are stored and deleted. http://blog.snapchat.com/post/50060403002/how-snaps-are-stored-and-deleted.

[41] L. Sweeney. Discrimination in online ad delivery. *Communications of the ACM*, 56(5), May 2013.

[42] V. Toubiana, A. Narayanan, and D. Boneh. Adnostic: Privacy preserving targeted advertising. *Proc. NDSS*, 2010.

[43] X. Wang, M. Gerber, and D. Brown. Automatic crime prediction using events extracted from twitter posts. In S. Yang, A. Greenberg, and M. Endsley, editors, *Social Computing, Behavioral - Cultural Modeling and Prediction*, volume 7227 of *Lecture Notes in Computer Science*, pages 231–238. Springer Berlin Heidelberg, 2012.

[44] A. Whitten and J. D. Tygar. Why Johnny can't encrypt: A usability evaluation of PGP 5.0. In *Proc. of USENIX Security*, 1999.

[45] C. E. Wills and C. Tatar. Understanding What They Do with What They Know. *WPES '12: Proceedings of the 12th annual ACM workshop on Privacy in the electronic society*.

[46] X. Xing, W. Meng, D. Doozan, N. Feamster, W. Lee, and A. C. Snoeren. Exposing Inconsistent Web Search Results with Bobble. *Passive and Active Measurements Conference*, 2014.

[47] Y. Zhu, J. Jung, D. Song, T. Kohno, and D. Wetherall. Privacy scope: A precise information flow tracking system for finding application leaks. Technical Report UCB/EECS-2009-145, EECS Department, University of California, Berkeley, Oct 2009.

[48] P. R. Zimmermann. The official PGP user's guide. 1995.

A Proof of Theorem 1

A.1 Targeting functions, Axioms and Core Family

To formalize our main result we need to carefully define how targeting works and the simple qualitative axioms that it obeys. We show in this section that, provided those axioms are satisfied, targeting can always be associated with a small number of input combinations that we call its core.

A.1.1 Definitions and main result

Given a fixed universe of N inputs, a *combination* $\mathscr{C}$ of order r, also called r_combination, is a subset of r elements among the N inputs.

Each given ad is associated with a *targeting function* defined as a mapping f from any subset $\mathscr{C}$ of the N inputs into $\{0,1\}$, where $f(\mathscr{C})=1$ denotes that an account containing $\mathscr{C}$ as inputs should be targeted. By convention, untargeted ads are associated with the null function $f(.)=0$. Any targeting function f satisfies two axioms:

- **monotonicity**: $\mathscr{C} \subseteq \mathscr{C}' \implies f(\mathscr{C}) \leq f(\mathscr{C}')$.
- **input-sensitivity**: $\exists \mathscr{C}, \mathscr{C}'$ s.t. $f(\mathscr{C}) \neq f(\mathscr{C}')$.

Monotonicity simply reflects that an account with strictly more interest or hobbies should in theory be relevant to more ads, and never to less. Input sensitivity prevents the degenerate case where a targeting function is constant.

A *family* S of *size* l is any collection of l distinct combinations. The *order* of this family is defined as the largest order of a combination it contains. For any family S, one can define a targeting function that takes value $f(\mathscr{C})=1$ whenever the subset $\mathscr{C}$ contains at least one combination in S. We now show the converse also holds.

Lemma 1 *For each monotone, input-sensitive targeting function there exists a unique family S satisfying:*

(i) S has size l and order r and it explains *f, which means $f(\mathscr{C})=1$ holds if and only if $\exists \mathscr{C}' \in S, \mathscr{C}' \subseteq \mathscr{C}$.*

(ii) No family of size $l' < l$ explains f.

(iii) No family of order $r' < r$ explains f.

Hence, associated with each ad and therefore each targeting function is a unique family of input combinations that is targeted. We call this the ad's *core family*.

Before proving the result above, we discuss its meaning and consequences. Let us first introduce a definition, the following order relation will play an important role: We say that a family S *explains* another S' if for any combination $\mathscr{C}'$ in S' there exists a combination $\mathscr{C} \in S$ such that $\mathscr{C} \subseteq \mathscr{C}'$. Note that according to the definitions above, S explains a function f if and only if it explains $S_f = \{\ \mathscr{C}\ \mid f(\mathscr{C}) = 1\ \} = f^{-1}(\{1\})$ and $S \subseteq S_f$.

For example, with $n = 4$ inputs, $S = \{\ \{1,3\}, \{4\}\ \}$ and $S' = \{\ \{1,2,3\}, \{4\}, \{2,4\}, \{1,3\}\ \}$ we see that S explains S'. Intuitively, if S explains S', then if we were to observe that all combinations in S' receive an ad, this could in theory be explained by the hypothesis that the ad is targeted at accounts which contain any of the combinations of inputs in S. Alternatively, if S does not explain S', then it shows that S is not sufficient on its own to interpret this observation. Similarly, a family S explains f if all its combinations are relevant to the ad, and for any subset of inputs S' that leads f to take value 1, at least one combination in S is included in S'.

Note that, by definition S_f explains f, but it does not explain f *succinctly*. In particular S_f is a big family that contains a lot of combinations, and since by monotonicity we have $f(\{1,\ldots,N\}) = 1$ then S_f contains the combination of all inputs (which has order N)). What Lemma 1 and the definition of a core family indicate is that it is possible to find a *small* family, as small as possible both in terms of number and length of combinations involved, that also explains f. Note that this result is a consequence of the monotonicity axiom and does not hold for non-monotonic function.

Take the following example: if $f(S) = 1$ if and only if S contains a particular input D_i. S_f contains all supersets of $\{D_i\}$, a family containing 2^{N-1} combinations, but the family $S = \{\{D_i\}\}$ explains f as well, it is of size 1 and order 1.

Proof: Let $\overrightarrow{D}_f$ be the digraph with vertex-set S_f and with arc-set $\{\ (\mathscr{C},\mathscr{C}')\ \mid \mathscr{C} \subsetneq \mathscr{C}'\ \}$. We have that $\overrightarrow{D}_f$ is a DAG because the subset-containment relation defines a partial order. So, let S be the non-empty set of combinations with null in-degree in $\overrightarrow{D}_f$. By construction, S explains S_f and $S \subseteq S_f$, hence S explains f.

Furthermore, we claim that S is contained in *any* family S' explaining f: indeed, since S' is required to contain a subset of any combination $\mathscr{C} \in S$, and no combination of S_f is strictly contained in $\mathscr{C}$, then it must contain $\mathscr{C}$. This shows that S satisfies all conditions of Lemma 1. Finally, since another family explaining f needs to include S, then it will necessarily have a higher size l, hence S is the unique with both minimum size and order. □

Hence, associated with each ad and therefore each targeting function is a unique family of input combinations that are targeted, called the ad's *core family*, and we now sketch why it is correctly identified by our algorithm.

A.2 Algorithm and correctness

We first describe the gist of the proof of Theorem 1 as following from two main claims. These claims are established by using properties of random subsets of elements, which we analyze before providing a formal complete proof.

A.2.1 Definitions and proof overview

A subset of inputs $\mathscr{C}$ is an *x_intersecting subset* of a family S (for $0 \leq x \leq 1$) if at least a fraction x of the subsets in S intersect $\mathscr{C}$ (i.e., each contains an input chosen in $\mathscr{C}$):

$$\{\ \mathscr{S} \in S\ \mid \exists i \in \mathscr{C}, i \in \mathscr{S}\ \} \geq x \cdot |S|\,.$$

Similarly, we say that S' is an *x_intersecting family* of a family S if at least a fraction x of the subsets contained in S contain a combination chosen in S':

$$\left\{\ \mathscr{S} \in S\ \middle|\ \exists \mathscr{C} \in S', \mathscr{C} \subseteq \mathscr{S}\ \right\} \geq x \cdot |S|\,.$$

One can immediately deduce the following lemma

Lemma 2 *Let S' be an x_intersecting family of $|S|$, $\exists \mathscr{C}$ an x_intersecting subset of S such that $|\mathscr{C}| \leq |S'|$.*

Indeed, one can build $\mathscr{C}$ by including for each combination of S' any single input it contains.

Overview of the proof: The gist of the argument for Theorem 1 is an original connection between small intersecting subsets and the effect of a core family. Given an ad, let us denote by $S^{(\text{ad})}$ the family of all inputs combinations that are receiving the ad. The proof relies on the following claim: There exists a value of $0 < x < 1$ such that with high probability an x_intersecting subset of order $\leq l$ exists for $S^{(\text{ad})}$ if and only if the ad is targeted with a core family of size $\leq l$. Hence, finding such subset is a sound and complete test for detecting that targeting occurs.

The proof unfolds with two complementary claims:

- **Completeness:** Let $S^{(\text{core})}$ be the ad's core family, then an x_intersecting subset of $S^{(\text{ad})}$ with size $|S^{(\text{core})}|$ exists.

This claim holds trivially when targeting is strict and the ad is never shown outside the target (i.e., $p_{\text{out}} = 0$). Indeed, all combinations of inputs $\mathscr{S}$ seeing the ad (i.e., in $S^{(\text{ad})}$) necessarily need to be within the target (i.e., $f(\mathscr{S}) = 1$) and hence they have to include a combination chosen in $S^{(\text{core})}$. We then deduce that $S^{(\text{core})}$ is a 1_intersecting family of $S^{(\text{ad})}$ with size l, hence by Lemma 2 a 1_intersecting subset exists with at most the same size.

When targeting is not strict (i.e., $p_{\text{out}} > 0$) it is more complicated, we can however prove that a similar claim holds for a smaller value of x with high probability by exploiting properties of random subsets as shown below.

- **Soundness:** If targeting does not occur, then $S^{(\text{ad})}$ does not admit an x_intersecting subset of size l.

This claim follows naturally from the properties of random subsets and is not qualitatively surprising. However, it is important that we prove that the same value of x that is used for completeness also allows to obtain that property, which is why a careful analysis is required. Note also that it is critical that both properties hold using a small number m of accounts to test (i.e., m should be of the order $\ln(N)$ where N is the number of inputs to monitor).

Finally, while the above argument explains an algorithm can *detect* that targeting takes place, it does not explain how the core family can be *exactly computed*. Again, this can be done by leveraging stronger results of random subsets, and we present different algorithms that determine the core family with varying time-complexity tradeoff.

A.2.2 Random subsets and probabilistic inequalities

Let us start with some definitions:

- A random *Bernoulli subset*, denoted by $B(n,p)$, is a subset such that any of n elements is contained with probability p independently of all others.
- A random *Bernoulli family* of size m is a collection of m independent Bernouilli subsets.

Since Bernouilli subsets and families derive from many independent decisions to include or not a single element, we will use inequalities on the distribution of sum of binary variables, especially this one due to Chernoff:

Lemma 3 *If Y is a sum of independent binary variables, let $\mu = \mathbb{E}[Y]$, we have for any $0 < \delta \leq 1$:*

$$\mathbb{P}[Y \geq (1+\delta)\mu] \leq \exp\left(-\frac{\delta^2\mu}{3}\right) \text{ , and}$$
$$\mathbb{P}[Y \leq (1-\delta)\mu] \leq \exp\left(-\frac{\delta^2\mu}{2}\right)$$

Thus, for any polynomial P, integer N and value $\varepsilon > 0$,

$$\mu \geq \frac{3}{\delta^2}\ln\left(\frac{2P(N)}{\varepsilon}\right) \Longrightarrow \mathbb{P}[|Y-\mu| \leq \delta\mu] \geq 1 - \frac{\varepsilon}{P(N)}.$$

In other words, such variable Y remains close to its expectation (i.e., up to a constant multiplicative factor) except on an event of polynomially small probability. This holds as soon as its expectation is at least logarithmic.

The lemma below allows us to prove soundness:

Lemma 4 *Let $1 > x > 0$, $l \in \mathbb{N}$, $p < 1-(1-x)^{\frac{1}{l}}$, and a Bernouilli family $B_1(n,p), B_2(n,p), \ldots, B_m(n,p)$. There exists $C > 0$ such that for any $\varepsilon > 0$ and polynomial P, if $m \geq C \cdot (l\ln(n) + \ln P(n) + \ln(1/\varepsilon))$, then with probability $(1-\varepsilon/P(n))$ no x_intersecting subset exists of size l for this Bernouilli family.*

Proof: Let us consider an arbitrary subset $\mathscr{C}$ of size l. The probability that it intersects an arbitrary Bernouilli subset is $1-(1-p)^l$. If we introduce Y the variable counting how many Bernouilli subset $\mathscr{C}$ intersects, we observe that it is a sum of binary independent variables, with expectation $\mu = \left(1-(1-p)^l\right)m$. We also note that $\mathscr{C}$ is an x_intersecting subset exactly if $Y \geq xm$. Assuming $p < 1-(1-x)^{\frac{1}{l}}$ as we do, μ is multiplicatively smaller than xm. Hence we can apply Chernoff Bound to conclude that $\mathbb{P}[Y \geq xm] \leq \frac{\varepsilon}{P(n)n^l}$ when

$$m \geq C \cdot \left(\ln\left(n^l P(n)/\varepsilon\right)\right), \quad \text{with } C = 3\frac{1-(1-p)^l}{\left(x-\left(1-(1-p)^l\right)\right)^2}.$$

Since there are $\binom{n}{l} \leq n^l$ choices of $\mathscr{C}$, by the union bound the probability that at least one of them is an x_intersecting subset is at most $\frac{\varepsilon}{P(n)}$. □

A.2.3 Detailed proof

First, let us consider soundness. Assuming no targeting takes place, subsets of inputs in $S^{(\text{ad})}$ are chosen independently of the inputs that they contain. Hence it is a Bernouilly family of average size $p_\emptyset m$ with parameter N (the number of inputs) and $p = \alpha$. By choosing $x > 1-(1-\alpha)^l$ and m sufficiently large, with very high probability no x_intersecting subset of size l exists. In this case, our test correctly concludes that no targeting is taking place.

Now, let us consider completeness. We already explained why this test will be correct when the ad is received only by accounts within the target (i.e., $p_{\text{out}} = 0$) but it remains to be shown in the general case. We start from the following observation: The family $S^{(\text{ad})}$ is composed of two families. The first, $S^{(\text{ad,in})}$, contains subsets of inputs that are in the target, and hence contain a combination of $S^{(\text{core})}$. The second, $S^{(\text{ad,out})}$, includes subsets that are not in the target but received the ads due to $p_{\text{out}} > 0$. It can be observed that the size of both families depends on the values of p_{out}, p_{in}, α. We already know that a 1_intersecting subset of size l exists for $S^{(\text{ad,in})}$, that we can construct using $S^{(\text{core})}$. Note that it is also an x_intersecting subset for $S^{(\text{ad})}$, where $x = |S^{(\text{ad,in})}|/(|S^{(\text{ad,in})}| + |S^{(\text{ad,out})}|)$.

Lemma 5 *We assume that targeting takes place, where the targeting function admits a core family of size l and order r, and uses targeting probability p_{in} and p_{out}. Let $x > 0$, $\alpha > 0$, and assume $p_{out}/p_{in} < \frac{1-x}{x}\frac{\alpha^r}{1-\alpha^r}$. Finally, let $\mathscr{C}$ be any combination.*

There exists $C > 0$ such that for any $\varepsilon > 0$ and polynomial P, whenever we have

$$m \geq C \cdot (\ln P(N) + \ln(1/\varepsilon)) \,.$$

then with probability $(1-\varepsilon/P(N))$ the following holds: among accounts containing $\mathscr{C}$ and receiving the ad, at least a fraction x of them is within the targeting scope,

$$i.e., \; \frac{\left|\left\{ \mathscr{S} \in S^{(ad,in)} \mid \mathscr{C} \subseteq \mathscr{S} \right\}\right|}{\left|\left\{ \mathscr{S} \in S^{(ad)} \mid \mathscr{C} \subseteq \mathscr{S} \right\}\right|} \geq x \,.$$

Proof: For each of the accounts $A_1, \ldots, A_m$ we introduce Y_j which takes he following value:

$$\begin{cases} 1 & \text{if } A_j \text{ is in target, sees the ad, and } \mathscr{C} \subseteq A_j, \\ -\frac{x}{1-x} & \text{if } A_j \text{ is not in target, sees the ad, and } \mathscr{C} \subseteq A_j, \\ 0 & \text{otherwise} \end{cases}$$

We then introduce $Y = \sum_{j=1}^{m} Y_j$, which is a sum of binary independent variables. We also note that the property of the theorem holds exactly if $Y \geq 0$, it is then sufficient to prove that this occurs with high probability using a Chernoff bound argument.

First by the linearity of expectation we have that:

$$\begin{aligned} \mathbb{E}[Y] &= \sum_{j=1}^{m} \left(\alpha^{|\mathscr{C}|} q_{\mathscr{C}} \, p_{in} - \frac{x}{1-x} \alpha^{|\mathscr{C}|} (1 - q_{\mathscr{C}}) \, p_{out} \right) \\ &= \left(q_{\mathscr{C}} \, p_{in} - \frac{x}{1-x} (1 - q_{\mathscr{C}}) \, p_{out} \right) \alpha^{|\mathscr{C}|} m, \end{aligned}$$

where $q_{\mathscr{C}}$ denotes the probability for an account to be within scope knowing that it contains $\mathscr{C}$. This expectation is positive as long as it holds that $p_{\text{out}}/p_{\text{in}} < \frac{1-x}{x} \frac{q_{\mathscr{C}}}{1-q_{\mathscr{C}}}$. Moreover, the above upper-bound is monotonically increasing with $q_{\mathscr{C}}$, which is at least α^r because it suffices to complete $\mathscr{C}$ with any combination of the core to be within scope. As a result, it always holds that $\mathbb{E}[Y] > 0$ (with respect to our assumption about the ratio $p_{\text{out}}/p_{\text{in}}$ for the lemma).

Accordingly we deduce whenever $m \geq C \cdot \ln(P(N)/\varepsilon)$

$$\begin{aligned} \text{with } C = \; & \frac{2}{\alpha^{|\mathscr{C}|} q_{\mathscr{C}} \, p_{\text{in}}} \left(1 - \frac{x}{1-x} \frac{1 - q_{\mathscr{C}}}{q_{\mathscr{C}}} \frac{p_{\text{out}}}{p_{\text{in}}} \right)^{-1} \\ \leq \; & \frac{2}{\alpha^{|\mathscr{C}|+r} p_{\text{in}}} \left(1 - \frac{x}{1-x} \frac{1 - \alpha^r}{\alpha^r} \frac{p_{\text{out}}}{p_{\text{in}}} \right)^{-1}, \end{aligned}$$

that $\mathbb{P}[Y \geq 0] \geq 1 - \frac{\varepsilon}{P(N)}$ holds $\square$

Final argument. According to Lemma 5 (applied with $\mathscr{C} = \emptyset$), the existence of an x_intersecting set of size l is guaranteed with high probability, if we can satisfy the condition on x.

In particular, since we could a priori fix α to satisfy $x > 1 - (1-\alpha)^l$ we have that both proof apply simultaneously whenever there exists $0 < x < 1$ verifying:

$$\frac{p_{\text{out}}}{p_{\text{in}}} < \frac{1-x}{x} \frac{(1 - (1-x)^{\frac{1}{l}})^r}{1 - (1 - (1-x)^{\frac{1}{l}})^r} = \varphi_{l,r}(x) \,. \qquad (1)$$

Whenever this condition is verified (i.e., whenever the gap between p_{out} and p_{in} is sufficiently large), one can choose a value of x, α and subsequently C such that if $m \geq C \ln(n/\varepsilon)$ the detection test is correct with probability $1 - \varepsilon/n$.

Note that while finding an x_intersecting subset is a sufficient evidence that targeting takes place, it does not allow us to directly compute the core family. In particular this subset is neither a combination of the core family, it is a union of elements that all appear in at least one combination of the core family, but it is not unique.

However, using this detection brick, various algorithms can be used to exhaustively search for a core family. We will also show that a polynomial-time algorithm can refine this analysis to compute the core family at the expense of a more complex recursion.

A.2.4 When is the condition verified?

The condition of Eq.(1) is important because it denotes the maximum ratio $p_{\text{out}}/p_{\text{in}}$ that can be detected by our algorithm. Intuitively, if this ratio is 1 and $p_{\text{out}} = p_{\text{in}}$ targeting has no effect and hence its presence and its core family remains impossible to determine. Since the choice of the percentage x is a parameter of the algorithm that can be tuned (along with the value of α) it would be interesting to know under which condition we can detect targeting with the largest $p_{\text{out}}/p_{\text{in}}$ ratio. The following lemma answers that question precisely:

Lemma 6 *Let $M_{l,r} = \max_{x \in]0;1[} \varphi_{l,r}(x)$, we have*

$$\begin{cases} \text{if } l = 1, & M_{1,r} = 1/r, \\ \text{if } r = 1, & M_{l,1} = 1/l, \\ \text{if } r = l = n > 1, & M_{n,n} = 1/(2^n - 1)^2, \\ \text{if } r > 1, l > 1, & M_{l,r} = \frac{z^l}{1-z^l} \frac{(1-z)^r}{1-(1-z)^r}, \end{cases}$$

where z is the only solution in $]0;1[$ of

$$r z^{l+1} - l(1-z)^{r+1} - (r+l)z + l = 0 \,,$$

and this maximum is attained for $x = 1 - z^l$.

Proof: When $l = 1$ one can easily see that $\varphi_{1,r}$ is strictly increasing on this interval and computes its limit as x approaches 1. A similar argument holds for $r = 1$.

Whenever $r > 1$ and $l > 1$, introducing the new variable $z = (1-x)^{1/l}$ we first observe:

$$\varphi_{l,r}(z) = f_l(z) \cdot f_r(1-z) \,, \text{ where } f_n(z) = \frac{z^n}{1-z^n}.$$

We observe $\varphi'_{l,r}(z) = f'_l(z) \cdot f_r(1-z) - f_l(z) \cdot f'_r(1-z)$, and note that this derivative becomes null whenever $f'_l(z)/f_l(z) = f'_r(1-z)/f_r(1-z)$. Moreover, we have

$$f'_n(z) = -\frac{l z^{n-1}}{(1-z^n)^2} \;\text{ hence }\; f'_n(z)/f_n(z) = \frac{n}{z(1-z^n)}$$

so that the condition is $\frac{l}{z(1-z^l)} = \frac{r}{(1-z)(1-(1-z)^r)}$ which yields the value of z reaching the maximum.

To conclude, we just need to observe that there is a unique solution in $]0;1[$. We can immediately observe, when $r > 1$ and $l > 1$ that the product $f_l(z) \cdot f_r(1-z)$ has null limits on both side, and a derivative that is positive near 0^+ and 1^-. Since its third derivative is strictly positive, its second derivative increases and can only be null once. We deduce that the derivative cannot cancel twice between 0 and 1 since it would create two inflexion points.

Finally, when $r = l = n$, since the product is symmetric in z and it has a unique maximum on $]0;1[$ it has to be in $z = \frac{1}{2}$ which yields the result. □

According to this lemma, when a single input is used for targeting *i.e.*, $l = 1, r = 1$, the condition is always verified as soon as $p_{\text{out}} < p_{\text{in}}$ and hence any targeting is detected. When the targeting uses a single combination of $r > 1$ inputs (i.e., $l = 1$) or a union of $l > 1$ single inputs (i.e., $r = 1$), the condition holds as long as p_{out} is below some threshold.

When l and r are allowed to grow beyond 1, the quick combinatorial explosion of the number of hypotheses to test by our system requires that the ratio $p_{\text{out}}/p_{\text{in}}$ decreases exponentially fast, but detection remains possible. For $l = r = 3$, a relatively complex case, we can still detect targeting even when 2% of accounts outside the target received the ads. Figure 12 presents

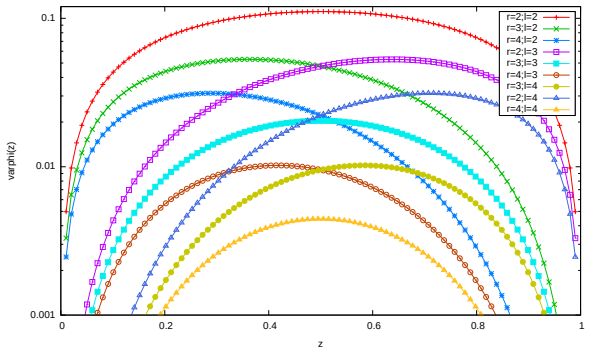


Figure 12: The function $\varphi_{l,r}(z)$ for $l > 1$ and $r > 1$.

the value of the RHS defining the necessary condition, as a function of the variable $z = (1-x)^{\frac{1}{l}}$. We observe maxima for different values of l and r.

A.2.5 Beyond detection, computing the core family

So far, we have shown that, after computing the family made with inputs of account receiving ads, looking for an x_intersecting subset of this family with size l is a correct test with high probability whenever we have a logarithmic number of accounts. If this test determines that targeting does not take place, there is no other explanation to find. However, if targeting occurs, one would also like to deduce from this test *which* combination of inputs are used for targeting this ad, or computing exactly the core family of the function f.

Here we show that under the same condition as detection, we can compute the core family. There are multiple algorithms to do so, each one potentially better depending on what is known about the targeting. They all use a common result that we draw below:

Lemma 7 *We assume Eq.(1) and targeting occurs with a core family $S^{(core)}$. There is $C > 0$, $0 < x < 1$ such that for any $\varepsilon > 0$, polynomial P, and combination $\mathscr{C}$, if $m \geq C \cdot (\ln(n) + \ln P(n) + \ln(1/\varepsilon))$, then with probability $(1 - \frac{\varepsilon}{P(n)})$ exactly one of the following claims holds:*

(i) $\mathscr{C}$ contains a combination from the core family

$$\textit{i.e.}, \exists \mathscr{S} \in S^{(core)}, \mathscr{S} \subseteq \mathscr{C}.$$

(ii) an x_intersecting subset of size l exists for

$$\Delta^{(ad)}(\mathscr{C}) = \left\{ \mathscr{S} \cap \overline{\mathscr{C}} \;\middle|\; \mathscr{S} \in S^{(ad)}, \mathscr{C} \subseteq \mathscr{S} \right\}.$$

This result combines all lemmas used in the proof of the detection test. In fact, with the convention $S^{(\text{core})} = \emptyset$ used to denote non-targeting, it contains the proof of detection test as a particular case with $\mathscr{C} = \emptyset$. But its strength is to be applied to multiple different combination $\mathscr{C}$ as a building block to determine $S^{(\text{core})}$.

Proof: First we prove (i) implies (ii) cannot hold which is the easy part of the result. If a combination of the core is contained in $\mathscr{C}$, then any account that contains $\mathscr{C}$ as part of its input is in the target and hence it receives an ad with probability p_{in}, and this holds irrespectively of all other inputs. One deduces that $\Delta^{(\text{ad})}(\mathscr{C})$ in that case is a Bernouilli family, we can then apply Lemma 4 and conclude that (ii) may only occur with probability $\varepsilon/P(n)$.

We now show that if (i) does not hold, then (ii) does.

$$\text{Let } \Delta^{(\text{ad,in})}(\mathscr{C}) = \left\{ \mathscr{S} \cap \overline{\mathscr{C}} \;\middle|\; \mathscr{S} \in S^{(\text{ad,in})}, \mathscr{C} \subseteq \mathscr{S} \right\},$$

$$\text{and } \Delta^{(\text{core})}(\mathscr{C}) = \left\{ \mathscr{S} \cap \overline{\mathscr{C}} \;\middle|\; \mathscr{S} \in S^{(\text{core})} \right\}.$$

Note that since no combination of the core family is included in $\mathscr{C}$, no element of $\Delta^{(\text{core})}(\mathscr{C})$ is empty. Observe that, by definition a combination in $S^{(\text{ad,in})}$ should contain a combination of the core. This directly implies that a combination in $\Delta^{(\text{ad,in})}(\mathscr{C})$ necessary contains a combination from $\Delta^{(\text{core})}(\mathscr{C})$, which is by consequence a 1_intersecting family of $\Delta^{(\text{ad,in})}(\mathscr{C})$.

It is an immediate consequence of Lemma 5 that under the condition above, $|\Delta^{(\text{ad,in})}(\mathscr{C})|/|\Delta^{(\text{ad})}(\mathscr{C})| \geq x$ holds with probability $1 - \varepsilon/P(n)$. Therefore $\Delta^{(\text{core})}(\mathscr{C})$ is an x_intersecting family of $\Delta^{(\text{ad})}(\mathscr{C})$, proving (ii). □

The result above shows that under the same conditions as those used for detection, one can design a provably correct test to decide whether a combination $\mathscr{C}$ is a superset of a combination in the core. This test resembles the previous one, it looks for an intersecting subset of size l that does not use the inputs of $\mathscr{C}$ among the accounts containing $\mathscr{C}$. It uses no more than $O(N^{l+1})$ operations with a naive exhaustive search. What remains to be shown is how one can conduct multiple tests on various combinations $\mathscr{C}$ to compute $S^{(\text{core})}$. There are multiple ways:

Agglomerative algorithms: Assume an upper bound $l_{\max}$ is known. A simple (costly) search looks for the results of all tests for all combinations. For instance, one can maintain a current core $S^{(\text{core})}$ initialized to be empty, and a queue of combinations remaining to be checked, which is initialized to contain $\mathscr{C} = \emptyset$. The first test in effect tests whether targeting occurs. Whenever a combination $\mathscr{C}$ is at the head of the queue, we update it as follows: (1) if the combination already contains one combination identified in $S^{(\text{core})}$, simply drop it otherwise run the test; (2) if the test finds an intersecting subset of size $l \leq l_{\max}$, conclude that $\mathscr{C}$ does not contain a combination of the core, and add $N - |\mathscr{C}|$ combinations to the queue constructed as $\{\ \mathscr{C} \cup \{i\}\ \mid i \notin \mathscr{C}\ \}$, while avoiding those already in the queue; (3) if the test concludes that $\mathscr{C}$ contains a combination of the core, add $\mathscr{C}$ to $S^{(\text{core})}$.

It's possible to run the queue infinitely, stopping whenever $l_{\max}$ combinations have been identified, or when all combinations of order $r_{\max}$ have been checked, assuming such bound is known. This uses at most $O(N^{l_{\max}+r_{\max}+1})$ operations.

Removal algorithms: There are two drawbacks in the precedent algorithm: it tests a large number of combinations, and if the bound $r_{\max}$ is loose, and $l < l_{\max}$ it will test absolutely *all* combinations of size $r_{\max}$ before concluding, which seems very costly. We now present another algorithm that does not assume any bound on r, and prevents this exhaustive search.

It works as follows: let us assume we already identified a family of some combinations in the core, $S \subseteq S^{(\text{core})}$. If we assume we start from a combination $\mathscr{C}$ that (1) is not a superset of a combination in S, and (2) contains a combination from the core family $S^{(\text{core})}$, then we are guaranteed to find another combination of $S^{(\text{core})}$ using at most $|\mathscr{C}|$ tests. In fact, one can update $\mathscr{C}$ as follows: order all inputs from $\mathscr{C}$ arbitrarily, and for each one do the following: first remove the input from $\mathscr{C}$ and run the test to determine whether it still contains a combination of the core. If the test indicates that a core combination remains, this removal is permanent, otherwise, it proves that, for the remainder of inputs left, this one is "critical" and we put it back in $\mathscr{C}$. After we do that for all inputs, the ones remaining in $\mathscr{C}$ form a combination of the core.

One can start with $S = \emptyset$ and $\mathscr{C}$ containing all inputs, as this is guaranteed to find a core combination $\mathscr{C}_1$. At any time, S contains at most l combinations of at most r inputs, which means there are at most r^l subsets constructed by taking all inputs and removing at least one inputs from each of the combinations of S. All those subsets satisfy property (1) above, but not necessarily (2). In fact, we can consider them in any order, and run the test of property (2). If one of them does satisfy it, it can be used to find a new combination of the core, and the process repeats with a new value of S. Otherwise, if all of those subsets are shown not to contain any more combination, we can conclude that S contains all combinations of the core. There could not be more than l combinations in S, hence this algorithms uses at most lr^lN tests, which hence uses in total $O(N^{l_{\max}+2})$ operations.