Serving Ads from `localhost` for Performance, Privacy, and Profit

Saikat Guha[†], Alexey Reznichenko[†], Kevin Tang[‡], Hamed Haddadi[†], Paul Francis[†]
[†]Max Planck Institute for Software Systems and [‡]Cornell University
[†]{sguha,areznich,hamed,francis}@mpi-sws.org, [‡]kt258@cornell.edu

ABSTRACT

Online advertising is a major economic force in the Internet today. Today's deployments, however, erode privacy and degrade performance as browsers wait for ad networks to deliver ads. In this paper we pose the question: is it possible to build a practical private online advertising system? To this end we present an initial design where ads are served from the endhost. It is attractive from three standpoints — privacy, profit, and performance: tracking the user's profile on their computer and not at a third-party improves privacy; better targeting and potentially lower operating costs can improve profits; and relying more on the local endhost rather than a distant central third-party can improve performance. In this paper we explore whether such a system is practical with an eye towards scalability, costs, and deployability. Based on a feasibility study conducted over traces from over 31K users and 130K ads, we believe our approach holds much promise.

1. INTRODUCTION

Online advertising is a key economic driver in the Internet economy. While advertisers would like to increase the amount of personalization they use to target ads, they are hampered by both privacy concerns [3, 5] and the cost of using personal data to quickly deliver targeted ads [6]. In this paper we explore whether it is feasible to build an online advertising system that is both private and scalable.

To this end we have been designing a practical private online advertising system, which we call Privad. The core idea behind Privad is simple: all private information needed to target an ad is kept on the user's computer. Deciding what ads to show, as well as serving the ads is performed purely locally by the user's computer. This is made possible by pushing (many or all) ads to users in advance. Reports about which ads are viewed or clicked are transmitted in such a way that user privacy is preserved while still allowing the advertising network to detect and defend against click-fraud. We describe this approach in more detail in Section 2.

One key challenge in Privad is scaling to the large numbers of ads in the Internet today while preserving the advertising model that advertisers have come to expect. For instance, advertisers today can expect ads to be shown to users moments after uploading, and can specify daily budgets, which when exhausted deactivates the ad for the day. These two requirements result in what we call "ad churn" where the set of active ads changes constantly. Ad churn naturally has implications on how to scalably push ads to users, and the associated network and storage costs. We address these design issues in Section 3. Additionally, we study the month long behavior of 130K ads part of Google's search advertising network, and the online activity of 31K CoDeeN [9] users to better guide our design decisions.

Another key challenge is incentivising deployment. User privacy alone is not sufficient for convincing advertising networks or users to adopt Privad. Advertising networks care most about maximizing revenue through better targeted ads and lower operating costs. Privacy guarantees allows Privad to freely perform behavioral targeting, which has been point of contention between existing online advertising networks and regulatory bodies [4]. Second, in contrast to existing approaches, Privad eliminates the need for expensive infrastructure for targeting ads to millions of users in real-time by outsourcing the task to the endhost; the tradeoff, however, is the cryptography overhead as we discuss in Section 3.

Incentivising users to install Privad requires first that Privad does not degrade user experience in any way. We can ensure this by only showing ads in the same ad boxes that are common today (unlike previous adware, which employed disruptive advertising). Second, especially early on there must be some positive incentive for users to install it. This could done through adware-style software bundling, shopping discounts, toolbars, or other incentives. As we show in Section 4, the behavior of CoDeeN users lends some credibility to such a deployment model. Finally, it requires that privacy advocates (e.g. EFF, ACLU, and government agencies) endorse Privad. This at least prevents anti-virus software from actively removing Privad from clients. Ideally, it even leads to privacy-conscious browser vendors (e.g. Firefox) or operating systems installing it by default, or by governments mandating that existing advertising companies deploy Privad technology.

Overall, Privad represents an argument that highly-targeted practical online advertising, and good user-privacy are not mutually exclusive. Our rough design and cursory feasibility studies notwithstanding, we believe this approach is well worth exploring.

2. PRIVAD ARCHITECTURE

We begin with the Privad model and rough protocol, followed by a brief discussion on the privacy characteristics of Privad. Due to space constraints, we present only a small part of the larger Privad system [2] with the aim of orienting the reader for the scalability issues discussed in the rest of this paper.

2.1 Model

There are five players in Privad: user, publisher, advertiser, broker, and dealer. User, publisher, and advertiser are identical to today's model: users visit publisher webpages; advertisers wish their ads to be shown to users; for each ad viewed or clicked the advertiser is charged and the publisher is paid a commission. The *broker* brings together advertisers, publishers, and users much like today (e.g. Google). The key distinction, however, is that in Privad the task of profiling the user, and targeting and serving ads is outsourced to a *client*[1] running on the user's computer.

The *dealer* is the key to privacy in Privad. All communication between the client and the broker is proxied anonymously by the dealer. The dealer is run by an organization that is itself untrusted with user profile information, but is nevertheless unlikely to collude with the broker. This could for instance be prominent privacy advocacy groups (e.g. EFF or ACLU) or a government regulatory agency. The dealer would be funded by a special tax levy on the broker.

The dealer serves two roles. For the user the dealer ensures anonymity by hiding the user's identity (e.g. IP address) from the broker, but itself does not learn any profile information about the user since all messages between the client and broker are encrypted. Unfortunately, when clients are hidden from the broker, the broker is less able to protect itself against click-fraud. Therefore, the dealer also helps the broker defend against click-fraud, but in a way that preserves user privacy. Additionally, the dealer helps protect against application-level DoS at the broker by rate-limiting client messages.

[1]The client is, in fact, an untrusted black-box monitored by a trusted reference monitor, for instance the user's anti-virus software.

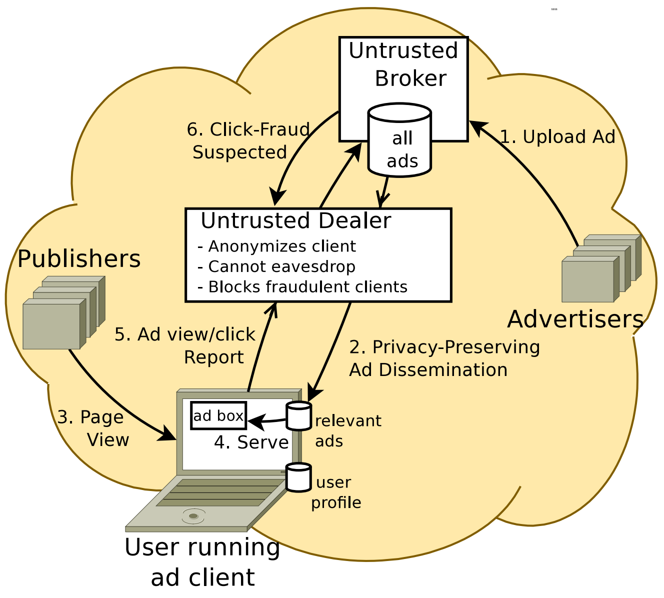

Figure 1: The Privad architecture

2.2 Protocol

In this paper we discuss three Privad mechanisms: ad dissemination, view/click reporting, and click-fraud defense.

Ad Dissemination: The client builds up a personalized user profile capturing such attributes as each user's demographics, interests, or keywords. This is done by monitoring the user's activity, for instance, on online social networking websites, shopping sites, and other applications. Advertisers upload ads to the broker (Figure 1, message 1), which the broker then sends to clients. The ads are filtered locally by the client based on the user's profile and stored until needed.

A privacy-preserving Pub-Sub mechanism between the broker and client is used to disseminate ads to the client. The Pub-Sub channels are nested interest categories such as sports, tennis, rackets, Wilson rackets, as well as limited demographics such as geographic region or gender. The client sends a subscription message to the broker; the message is relayed by the dealer to hide the client's identity. The message is encrypted with the broker's public key to prevent the dealer from eavesdropping. The subscription message contains a symmetric key chosen by the client. The symmetric key is used by the broker to encrypt ads published to that client (Figure 1, message 2). Keys for different subscriptions by the same client are different to prevent the broker from linking together a client's interests. In Section 3.2 we slightly modify this mechanism for scalability.

Ad View/Click Reporting: The client serves an ad from the local store when the user visits a publisher webpage with an ad box. A view/click report containing the ad ID and publisher ID is sent to the broker. As before, the message is encrypted with the broker's public-key and relayed by the dealer (messages 3–5). The dealer attaches a unique record ID to each report before relaying it; a mapping between this record ID and the user's identity (e.g. IP address) is stored temporarily at the dealer to assist in click-fraud defense.

Click-fraud Defense: There are two parts to defending against click-fraud: first, detecting click-fraud, and second, taking corrective action. The broker may use a range of passive and active techniques to detect click-fraud including, for instance, expectations based on historical data, statistical tests, "bait" ads, and honeyfarms. When click-fraud is suspected, the broker presents the dealer with the record IDs of the fraudulent clicks (message 6). The dealer maps the record IDs to the user(s) responsible, and for users implicated more than some threshold number of times, the dealer stops relaying subsequent click reports.

Everything else: Due to lack of space we cannot discuss the full set of mechanisms in Privad including attribute design, user profiling, ad auctions, click anonymization, click-fraud detection, and the reference monitor framework. Interested readers should see [2].

2.3 A Brief Word on User Privacy

While this paper focuses on the feasibility and scalability aspects of the Privad architecture, for completeness we present here a short discussion of user privacy. For a more in-depth treatment of privacy and click-fraud issues see [2].

In Privad, the definition of privacy we are particularly concerned with is *unlinkability.* A user's personalization profile contains many attributes (e.g. demographics, what websites were visited, etc.). Our first design objective is: no single player should be able to link the *identity* of the user with any piece of the user's personalization profile. Second, no single player should be able to link together more than some limited and well-known threshold number of pieces of personalization information for a given user.

Privad meets these constraints by splitting the data between the dealer and broker in such a way that compromising a single party does not violate user privacy. For instance, through the reporting mechanism, the dealer learns `User A` viewed/clicked on *some* ad, while the broker learns *someone* viewed/clicked on `Ad X`. Second, the broker cannot link together multiple reports from the same client preventing the broker from reconstructing the user's profile over time. Third, the broker cannot single out a user by feeding it bogus information during ad dissemination since the client can detect this attack by establishing multiple subscriptions, which the broker cannot link together.

The protocol as described is susceptible to collusion between the dealer and broker. One defense is to chain multiple dealers; dealer N keeps secret the mapping between the record ID at dealer $N-1$ and itself, and a corresponding change is made to the click-fraud defense mechanism. This doesn't solve the collusion problem, but raises the bar by requiring all dealers and the broker to collude.

Ultimately the goal is not to produce a bullet-proof system, but rather to be much more private than existing systems (admittedly a low bar). Defending against governments that legally subpoena all involved parties is explicitly a non-goal. Rather, Privad defends against third-parties, for example an insurance company, that today can enter into a secret agreement with a single player and trawl for users matching a particular profile.

3. SCALING PRIVAD

In this section we discuss scalability concerns in the three Privad mechanisms, and explore potential solutions. The goal of this section is not to arrive at a final solution but to lay out the design space that we are in the process of exploring through various measurement studies and a prototype implementation. Overall, we have not yet found any issues that would fundamentally limit scalability.

3.1 Trace Data

We gathered two sets of traces to guide our design decisions: Google search ads, and CoDeeN click-stream.

Google data: We sampled Google search ads for a month-long period. We selected 1300 words uniformly at random from a dictionary consisting ~100K words built from a webpage catalog [8]. We then issued a Google search query for each word roughly every 30 minutes and recorded the ads served. While we took a number of measures to discover as many ads as possible (e.g. issuing queries from a geodiverse set of planetlab nodes), we cannot determine what fraction of all ads for the chosen set of queries we managed to discover. Nevertheless, we believe the data captures the qualitative characteristics of ads (in the US), and serves only as a rough guide in quantitative matters.

CoDeeN data: We collected the click-stream pertaining to existing ad networks (Google, Doubleclick, etc.) for anonymized CoDeeN users over a month-long period. We use CoDeeN's bot-detector tool [7] to restrict our analysis to the approximately 31K human users in the dataset. We noticed, however, some false

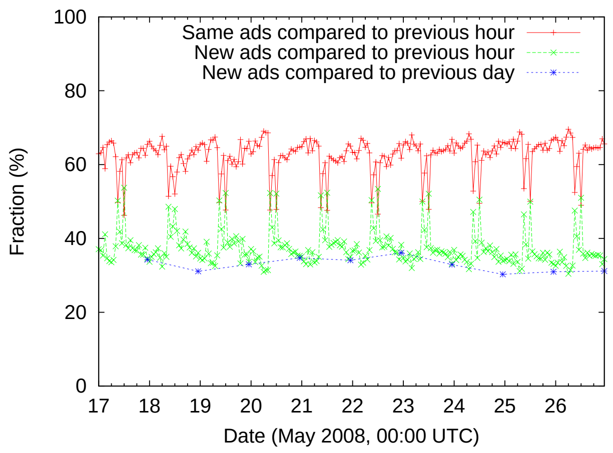

Figure 2: Ad churn in Google search ads

positives for bots that have evolved since the tool was first developed. We retain these bots in the data to account for bot activity one might normally expect. Altogether, the data allows us to analyze how users interact with ads (views/clicks etc.). Note that the data is inherently biased as CoDeeN users are more technically inclined than average web users[2].

In our analysis below, we point out where uncertainty or bias in our datasets affects results.

3.2 Ad Dissemination

Push vs. Pull: The simplest approach to disseminating ads is to flood all ads to all users. While this is ideal from a privacy perspective, we found it is not scalable. The problem is not so much the large number of ads (which could otherwise be disseminated using P2P), but rather ad churn. Figure 2 plots the fraction of ads that change from day-to-day, and hour-to-hour. Between 30%–40% ads on a given day/hour differ from those the previous day/hour. There is a daily spike at around 4am US east-coast time, when we suspect the daily budget is reset and ads that expired the previous day are reactivated. Additionally, 5%–10% of ads are permanently replaced every hour (i.e. the old ad was never again seen, and the new ad was never seen before). Thus even with optimal caching, we estimate flooding updates would require 2GB/month of compressed ad data, which is unacceptably high considering the margin for error in our estimates.

A pull based mechanism is more attractive. Ad churn does not increase client load, and additionally, turnaround for new ads is quick since the broker directly controls each ad sent. The question then is how Privad scalability compares to existing ad networks, which also use a pull model.

[2]Using CoDeeN for instance requires manual browser configuration

Request-Response vs. Pub-Sub: Existing ad networks use a request-response protocol where ads must be disseminated in real-time. This places a hard limit (few hundred milliseconds) in which to find candidate ads that match the user's profile, select the best ads, and return the results.

In contrast, a Pub-Sub approach does not require such fast responses. The broker can send out matching ads well in advance of the user visiting a publisher since the client can store the ad until needed. Further, Pub-Sub requires half the number of messages compared to request-response since the subscribe message is sent only once.

Personalization in the cloud vs. at the edge: Existing ad networks personalize ads to individual users based, for instance, on past browsing history, search history, and user preferences (some networks). Since each user is different, processing and memory requirements grow with the number of users.

Pub-Sub channels tradeoff network for less memory and processing at the broker. Since the fine-grained personalization is performed at the client, the broker only needs to perform very coarse-grained filtering — i.e. classifying ads into channel categories — to reduce network overhead. Thus processing and memory requirement are driven primarily by the (fixed) number of channels and is largely independent of the number of users (modulo maintaining channel membership, which amounts to a few bytes per online user).

The tradeoff is added network traffic since more ads must be sent as compared to existing networks, although with fairly fine-grained channels, this can be reduced to a degree. That being said, the use of crypto for the privacy aspects amplifies the cost of sending extra ads.

Cryptography Overhead: Every ad sent is encrypted by the broker at some cost, however, since we use symmetric key encryption, and there exists hardware accelerators that can perform at line-speeds, this is not a scalability concern per se. We are, however, looking at options to reduce this overhead should it prove to be a serious concern.

One option that we considered, but ultimately rejected, was using a Tor-like model [1] for privacy-preserving ad dissemination. In such a model the encryption cost would be borne out by the Tor exit-node (i.e. distributed across clients). Our problem with Tor, aside from the added complexity, is that it is of single-purpose use in Privad; it can be used for preserving privacy in ad dissemination, but cannot be used for private ad reporting since the exit-node cannot help defend against click-fraud. The dealer, in contrast, preserves privacy for both mechanisms, and additionally, is extremely simple. Fortunately, there are other options for reducing the crypto overhead.

In our Google dataset we observe the number of impressions for ads is highly skewed — a small fraction of ads (10%) garner a disproportionate fraction of impressions (80%). Second, these ads are correlated with the more generic queries in our sample set (e.g. Kitchen). Third, these generic queries have higher than average advertiser competition, which we assume results in higher costs per click. This suggests high advertising budgets and broad targeting are a likely cause, and as such are a characteristic of online advertising that can be leveraged.

Specifically, a small number of broadly targeted ads is particularly well suited to the push model discussed earlier. Supplementing the Pub-Sub mechanism with a P2P-based push (e.g. Gossip) has the potential of reducing crypto costs. Ads would be classified as *persistent* or *ephemeral*, where persistent ads are those with a broad reach and high daily budgets. The broker would use the Pub-Sub mechanism to seed persistent ads to a small number of clients that would then gossip it onward; this would significantly reduce the number of crypto operations at the broker without adding much overhead to clients due to the small number of such ads. Ephemeral ads would continue to be distributed using only the Pub-Sub mechanism. Of course, with P2P comes additional privacy concerns and attack vectors that we must consider.

Overall, we believe building a scalable privacy preserving ad dissemination mechanism is feasible. Scalability at par with existing ad networks is likely to be achievable given the benefits of Pub-Sub over Request-Response, and personalization at the edge rather than in the cloud, despite the symmetric key cryptography overhead. In addition, the possibility of reducing crypto costs further by enlisting client help looks promising.

3.3 Ad View/Click Reporting

Cryptography Overhead: The overhead of public-key operations at the broker is the primary scalability concern in view/click reporting. The problem is much more severe for view reports than for click reports since views far outnumber clicks and have a lower profit margin. There are two approaches that we believe can be used to scale reporting: sampling, and offloading computation to clients.

The first option is to simply send a view report with a uniform $1/p$ probability; the broker adjusts view counters accordingly for each report. This results in an estimate of the number of views with a factor p reduction in public key operations.

The second option farms out public-key operations to idle clients to reduce load at the broker. This is made possible by the following protocol: each client periodically generates a public-private key pair; the public key is sent to the broker, for instance by piggybacking it onto a subscribe message. When a client wishes to send a view/click report, it first requests the broker for a random public key to encrypt the report with. It then sends the encrypted report, the public-key used, and a hash of the original report contents to the broker. The broker forwards the encrypted report to the client that holds the corresponding private key, which decrypts the report for the broker. The broker can validate the decryption by verifying the hash, which is much cheaper than a public key operation.

For privacy, both the request/response for the public key, and the encrypted report are proxied by the dealer. This prevents the broker from linking different reports from the same user. However, since the objective is to reduce crypto operations, the request/response for the public key cannot itself be encrypted. This poses a potential man-in-the-middle attack at the dealer. One solution is for the client to request the key through one dealer and send the encrypted report through a different dealer. A stronger defense against colluding dealers (but more expensive for the client) is to request multiple keys through N dealers and iteratively encrypt the report with all N keys, thereby raising the bar by requiring all N dealers to collude. Note, however, that clients performing the decryption cannot compromise user privacy (even if they collude among themselves) since they learn no more than the broker would have in our original private reporting scheme.

That being said, this protocol is still preliminary and subject to further analysis. Nevertheless, we believe the fundamental notion of harnessing clients to reduce public-key operations at the broker without compromising user privacy is tractable.

3.4 Click-Fraud Defense

Storage Overhead: The only scalability concern with the click-fraud defense is storage at the dealer. The length of time the mapping between the record ID and client identity must be stored depends on the time it takes the broker to detect click-fraud. Depending on the complexity of the statistical analysis, we expect the number to be between a few days, to a few weeks. In any event, since disk is cheap, this is not a big concern.

4. DEPLOYING PRIVAD

There are two main challenges in deploying Privad: getting users to install it, and deploying the dealer function. The key to both is that Privad is verifiably private.

The user deployment for Privad is, admittedly, adware. The historical failure of adware is not due to its

	Users	Ad Views	CTR	3rd-Party Toolbars	Ad Blockers
China	7308	39K	0.5 %	22 %	12 %
Saudi Arabia	6710	56K	2.7 %	40 %	9 %
United States	1420	19K	0.9 %	13 %	17 %
U.A.E	1322	8K	1.7 %	35 %	8 %
Germany	956	5K	1.5 %	7 %	19 %
Worldwide	30987	189K	2.5 %	21 %	12 %

Table 1: Online ads viewed (for top-5 ad networks), click-through rate (CTR), and the use of third-party toolbars and ad blockers by CoDeeN users.

showing ads per se: Google and others have proven that ad-based businesses are viable. Rather, arguably the problems of adware were two-fold. First, it degraded user experience, either by showing ads in a disruptive way, or by consuming excessive system resources. Second, much adware was simply sleazy: installed without user knowledge, extremely difficult to uninstall, or outright spied on user data. This rightfully led to anti-virus companies identifying and destroying adware.

The key to Privad deployment is avoiding all the pitfalls of adware. Privad ads can be placed where they already exist—in banner ad boxes on web sites. The user won't notice any difference in the browsing experience. The fact that relatively few users install ad blockers supports the supposition that most users are ok with existing ads. Indeed, among CoDeeN users, who are presumably more technically savvy on average, fewer than 12% users use ad blockers (Table 1). Of course, the client must be implemented in such a way that it doesn't consume noticeable CPU, memory, or bandwidth, but our scaling features should make this very possible. If privacy advocates can be convinced that Privad is not only not bad, but in fact a good alternative to privacy-compromising cloud-based advertising, then there is a real chance that the anti-virus companies can be won over.

Assuming that the historic pitfalls of adware can be overcome, getting users to install it is relatively straightforward. For instance, a surprising (to us) number of users are willing to install plugins that offer minimal value to the user, for instance a toolbar or performance enhancer. In the CoDeen dataset, 21% of users installed one or more third-party toolbars (e.g. Alexa). For non-technical users this number could well be higher.

Bundling the Privad client with various freeware applications ranging from the low-end (screen-savers) to the high-end (Firefox browser) can bring in more users, albeit at some per-user expense. Ultimately, however, bundling with computer manufactures or operating system companies will yield the most users. Again, the key to this is winning over privacy advocates and possibly even regulatory agencies.

Unlike the broker, the dealer is not motivated by profit. We envision that the dealer would be funded through a special tax levy imposed on the broker, and operated either by privacy advocacy groups, government agencies, or some combination of both (i.e. operated by one with oversight by the other). Again, the key is in convincing privacy advocacy groups of the value of Privad relative to the status quo.

5. SUMMARY AND FUTURE WORK

This paper presents a preliminary design and analysis of a private online advertising system. The results encourage us to continue this research. As always, however, the devil is in the details, and many details remain unresolved. There are also some high-level issues that this paper does not address. One of them is auction scalability and quality, and how it relates to advertiser privacy. Others include detection mechanisms for click-fraud, and protecting user privacy after the click. While we have candidate solutions for these problems, it remains to be seen how effective they are in practice. In any event, we believe that our biggest challenge is more social than technical. Success requires that we convince privacy advocates, and the largely non-technical popular media, that Privad is not only far better than the status quo, but itself adequately safe. We hope this paper convinces other researchers to study the online advertising problem.

6. REFERENCES

[1] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. In *Proceedings of the 13th USENIX Security Symposium*, pages 303–320, San Diego, CA, USA, August 2004.

[2] S. Guha, B. Cheng, A. Reznichenko, H. Haddadi, and P. Francis. Privad: Rearchitecting Online Advertising for Privacy. Technical Report TR-2009-4, Max Planck Institute for Software Systems, Kaiserslautern-Saarbrücken, Germany, 2009. `http://mpi-sws.org/tr/2009-004.pdf`.

[3] A. Jesdanun. Ad targeting based on ISP tracking now in doubt. In *Associated Press*, Sept. 2008.

[4] D. Kaplan. FTC Issues 'Last Chance' To Ad Industry On Behavioral Targeting. *The Washington Post*, Feb. 2009.

[5] B. Krishnamurthy and C. E. Wills. Cat and mouse: content delivery tradeoffs in web access. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, 2006.

[6] R. Miller. Keynote: Ad networks failed, not news sites, June 2009. `http://tinyurl.com/kuybk4`.

[7] K. Park, V. S. Pai, K.-W. Lee, and S. Calo. Securing web service by automatic robot detection. In *Proceedings of the USENIX Annual Technical Conference*, Boston, MA, USA, May 2006.

[8] `http://www.dmoz.org/`. Dmoz: Open directory project.

[9] L. Wang, K. Park, R. Pang, V. Pai, and L. Peterson. Reliability and Security in the CoDeeN Content Distribution Network. In *Proceedings of the 2004 USENIX Annual Technical Conference*, Boston, MA, June 2004. `http://codeen.cs.princeton.edu`.