The Dark Side of the Web: An Open Proxy's View

Vivek S. Pai, Limin Wang, KyoungSoo Park, Ruoming Pang, and Larry Peterson
Department of Computer Science
Princeton University

Abstract

With the advent of large-scale, wide-area networking testbeds, researchers can deploy long-running services that interact with other resources on the Web. While such interaction can easily attract clients and traffic, our experience suggests that projects accepting outside input and interacting with outside resources must carefully consider the avenues for abuse of such services. The CoDeeN Content Distribution Network, deployed on PlanetLab, uses a network of caching Web proxy servers to intelligently distribute and cache requests from a potentially large client population. Due to CoDeeN's non-commercial nature, content is not pushed/advertised by content providers, but instead is pulled by clients who have configured their browsers to use CoDeeN.

In effect, CoDeeN is one of the largest "open" proxy networks in the world, and therefore draws unwanted attention from malicious users. This paper discusses our experiences with undesirable traffic on CoDeeN, the mechanisms we developed to curtail it, and the future directions for such work. We believe that this work provides a safe alternative to open proxies and will encourage others to deploy similar systems. Some of the security mechanisms we are developing are suitable for ISPs to deploy on their own networks to detect misbehaving customers before problems arise. Finally, other research projects that allow "open" access to Web resources may face similar situations, and may be able to adopt similar mechanisms.

1 Introduction

Researchers developing large-scale, wide-area network projects often require real traffic to test their systems, and one tempting option is to allow open access to their systems on the Internet. Especially for systems that can act as conduits for data, the ability to interact with the rest of the Internet provides a simple means for bootstrapping demand for such services and traffic. However, we have observed that this approach can invite traffic from malicious users. In this paper, we describe our experiences with CoDeeN, an academic Content Distribution Network (CDN) using a network of Web proxies on PlanetLab.

Caching Web proxy servers, commonly known as proxies, are widely used across the Web due to their ability to serve repeated requests from disk. Content consumers, such as ISPs, schools, and large organizations, use them as *forward proxies* to serve common requests locally and reduce the load on their slow/expensive wide-area network links. Content providers use them as *reverse proxies* to offload work from busy Web servers. Commercial content distribution networks use them, coupled with custom redirection logic, to widely replicate content providers' web pages to cache content closer to clients.

CoDeeN uses proxies in all of the capacities described above – each CoDeeN node is capable of acting as a forward proxy, a reverse proxy, and a redirector. When a client connects to a CoDeeN proxy and sends its Web requests, the node first acts as a forward proxy and tries to satisfy the requests locally. Requests not cached locally are handled by the redirector logic, which uses policies based on our recent work on CDN robustness [12]. Using request locality and system load information, the redirectors forward most requests to other CoDeeN nodes, which now act as reverse proxies for these requests. Requests still not satisfied are sent to the appropriate origin servers.

Going against conventional wisdom and standard practices, we deployed CoDeeN nodes as "open" proxies, allowing client access from outside the hosting organization. This decision was simpler than determining all of the configuration issues required, and we assumed that an unpublicized, experimental research network would not be of much interest to anyone. Additionally, since we wanted to eventually allow anyone to use CoDeeN, we did not see any benefit in disallowing access while we were developing and testing the system. We underestimated how long it would take for others to discover our system, and the scope of activities for which people seek open proxies.

Network administrators generally consider open proxies to be unwise for various reasons. Open proxies often *increase* bandwidth usage, since WAN-connected users double the proxy's WAN usage when fetching and forwarding cache misses. Of greater concern is that all outbound requests originate from the proxy, implicating it in any abuse-related complaints. In CoDeeN's early development, a small amount of outside traffic was helpful, since we could observe behavior under real load. As CoDeeN grew more stable, we noticed traffic increasing daily, even without any publicity on our part.

Within days of CoDeeN becoming stable, the PlanetLab administrators began receiving complaints. With over 40 CoDeeN nodes, we had focused on stability and reliability, and had not begun serious work on monitoring. As a result, the complaints we received were sometimes the first indication of abuse. These experiences led us to improve our own monitoring, and probably accelerated development of PlanetLab's traffic logging.

Unfortunately, we were unable to find any work or tools to "partially close" a proxy, where it could still be used by anyone but without fear of malicious behavior. Proxies were strictly binary in nature – either completely restricted and useful, or completely open and rife with abuse. With the growing rate of problems from CoDeeN's openness, we shut down the system while we analyzed the problems and developed solutions. Since relaunching the system in early June, thousands of users have accessed CoDeeN, and have generated only two abuse complaints. Below we describe some of the problems we encountered, our solutions to partially close CoDeeN, and our experiences with the results.

While our immediate motivation has been to secure the CoDeeN network from the problems associated with open proxies, we believe our techniques have broader application. The most obvious beneficiaries are people who want to deploy open proxies for some form of public good, such as sharing/tolerating load spikes, avoiding censorship, or providing community caching. Since ISPs generally deploy forward proxies transparently, our techniques would identify customers abusing other systems, before the ISP receives complaints.

2 Problems

In this section, we discuss some of the problems we encountered during the early development and testing of CoDeeN. For the purposes of discussion, we have broadly classified the problems into those dealing with spammers, bandwidth consumption, high request rates, content theft, and anonymity, though we realize that some problems can fall into multiple areas.

2.1 Spammers

The conceptually simplest category of CoDeeN abuser is the spammer, though the mechanisms for spamming using a proxy server are different from traditional spamming. We encountered three different approaches – SMTP tunnels, CGI/formmail POST requests, and IRC spamming. These mechanisms exist without the use of proxies, but gain a level of indirection via proxies, complicating investigation. When faced with complaints, the administrators of the affected system must cooperate with the proxy administrators to find the actual spammer's IP address.

SMTP tunnels – Proxies support TCP-level tunneling via the CONNECT method, mostly to support end-to-end SSL behavior when used as firewalls. After the client specifies the remote machine and port number, the proxy creates a new TCP connection and forwards data in both directions. Our nodes disallow tunneling to port 25 (SMTP) to prevent facilitating open relay abuse, but continually receive such requests. The prevalence and magnitude of such attempts is shown in Figure 1. As a test, we directed these requests to local honeypot SMTP servers. In one day, one of our nodes captured over 100K spam e-mails destined to 2,000,000 addresses. Another node saw traffic jump from 3,000 failed attempts per day to 30,000 flows in 5 minutes. This increase led to a self-inflicted denial-of-service when the local system administrator saw the activity spike and disconnected the PlanetLab node.

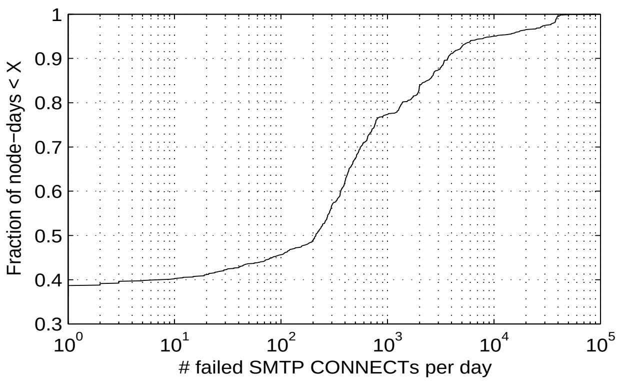

Figure 1: **CONNECT activity for 38 nodes – Almost 40% of the samples show no activity, while 20% show over 1000 attempts/day. The maximum seen is over 90K attempts to one node in one day.**

POST/formmail – Some web sites use a CGI program called *formmail* to allow users to mail web-based feedback forms to the site's operators. Unfortunately, these programs often store the destination e-mail address in the form's hidden input, relying on browsers to send along only the e-mail address specified in the form. Spammers abuse those scripts by generating requests with their victims' e-mail addresses as the targets, causing the exploited site to send spam to the victim.

IRC – IRC networks are targets for spammers due to their weak authentication and their immediate, captive audience. Most proxies allow CONNECTs to ports above the protected port threshold of 1024, which affects IRC with its default port of 6667. IRC operators have developed their own open proxy blacklist [2], which checks IRC participant IP addresses for open proxies. We were alerted that CoDeeN was being used for IRC spamming, and found many of our nodes blacklisted. While the blacklists eliminate the problem for participating IRC networks, the collateral damage can be significant if other sites begin to refuse non-IRC traffic from blacklisted nodes.

2.2 Bandwidth Hogs

CoDeeN is hosted on PlanetLab nodes, with the hosts absorbing the bandwidth costs. Since most nodes are hosted at high-bandwidth universities, they attract people performing bulk data transfers. Due to lack of locality, such transfers provide no benefit to other CoDeeN users – they cause cache pollution and link congestion.

Webcam Trackers – Sites such as SpotLife.com provide a simple means to use digital cameras as auto-updating web cameras. This *subscription-based* service allows the general public to broadcast their own "webcams". We noticed heavy bandwidth usage of the SpotLife site, with individual IP addresses generating multiple image requests per second, far above the rate limits in the official SpotLife software. SpotLife claims to bundle their software with over 60% of digital cameras, and a community of high-rate downloaders has formed, to SpotLife's consternation. These users clearly have enough bandwidth to access webcams directly, but use CoDeeN to mask their identity.

Cross-Pacific Downloads – CoDeeN nodes in Washington and California received very high bandwidth consumption with both source and destination located along the Eastern rim of Asia. The multi-megabyte downloads appeared to be for movies, though the reason that these clients chose a round-trip access across the Pacific Ocean is still not clear to us. A direct connection would presumably have much lower latency, but we suspect that these clients were banned from these sites, and required high-bandwidth proxies to access them effectively. Given the high international bandwidth costs in Asia, Western US proxies were probably easier to find.

Steganographers – While large cross-Pacific transfers were easy to detect in access logs, others were less obvious. This class had high aggregate traffic, spread across uniformly-sized, sub-megabyte files marked as GIFs and JPEGs. Large images sizes are not uncommon in broadband-rich countries such as South Korea, but some size variation is expected given the unpredictability of image compression. We downloaded a few of these large files and found that they generated only tiny images on-screen. From the URL names, we assume that these files contain parts of movies stuffed inside image files to hide their actual payload. However, we have not found the appropriate decryption tools to confirm our guess.

2.3 High Request Rates

TCP's flow/congestion controls mitigate the damage that bulk transfers have on other CoDeeN users. In contrast, another class of users generated enough requests that we were concerned that CoDeeN might be implicated in a denial-of-service attack.

Password Crackers – We found an alarming number of clients using CoDeeN to launch dictionary attacks on Yahoo, often via multiple CoDeeN nodes. At one point, we were detecting roughly a dozen new clients per day. Since Yahoo can detect multiple failed attempts to a single account, these users try a single password across many accounts. The attacks appear to be for entertainment, since any victim will be random rather than someone known to the attacker. The problem, again, is that the requests appear to come from CoDeeN, and if Yahoo blocks the IP address, then other PlanetLab services are affected.

Google Crawlers – Like password crackers, we found a number of clients performing Google web/image searches on a series of sorted words. These were clearly mechanical processes working from a dictionary, and their requests were evenly spaced in time. We speculate that these clients are trying to populate their own search engines or perhaps build offline copies of Google, but cannot understand the direct benefit of such an approach.

Click-Counters – Ad servers count impressions for revenue purposes, and rarely do we see such accesses not tied to actual page views. The one exception we have seen is a game site called OutWar.com. Points are obtained when people click on a player's "special link", which delivers a Web page containing ad images. The system apparently counts hits of the player's link instead of ad views, which seems to invite abuse. We have noticed a steady stream of small requests for these links, presumably from players inflating their scores.

2.4 Content Theft

The most worrisome abuse we witnessed on CoDeeN was what we considered the most sophisticated – unauthorized downloading of licensed content.

Licensed Content Theft – Universities purchase *address-authenticated* site licenses for electronic journals, limited to the IP ranges they own. PlanetLab's acceptable use policies disallow accessing these sites, but CoDeeN unintentionally extended this access worldwide. We discovered this problem when a site contacted PlanetLab about suspicious activity. This site had previously experienced a coordinated attack that downloaded 50K articles. Unfortunately, such sites do not handle the *X-Forwarded-For* header that some proxies support to identify the original client IP address. Though this header can be forged, it can be trusted when *denying* access, assuming nobody would forge it to deny themselves access to a site.

Intra-domain Access – Many university Web pages are similarly restricted by IP address, but are scattered within the domain, making them hard to identify. For example, a department's web site may intersperse department-only pages among publically-accessible pages. Opportunities arise if a node receives a request for a local document, whether that request was received directly or was forwarded by another proxy.

2.5 Anonymity

While some people use proxies for anonymity, some anonymizers accessing CoDeeN caused us some concern. Most added one of more layers of indirection into their activities, complicating abuse tracking.

Request Spreaders – We found that CoDeeN nodes were being advertised on sites that listed open proxies and sold additional software to make testing and using proxies easier. Some sites openly state that open proxies can be used for bulk e-mailing, a euphemism for spam. Many of these sites sell software that spreads requests over a collection of proxies. Our concern was that this approach could flood a single site from many proxies.

TCP over HTTP – Other request traffic suggested that some sites provided HTTP-to-TCP gateways, named *http2tcp*, presumably to bypass corporate firewalls. Other than a few archived news articles on Google, we have not been able to find more information about this tool.

Non-HTTP Port 80 – While port 80 is normally reserved for HTTP, we also detected CONNECT tunnels via port 80, presumably to communicate between machines without triggering firewalls or intrusion detection systems. However, if someone were creating malformed HTTP requests to attack remote web sites, port 80 tunnels would complicate investigations.

Vulnerability Testing – We found bursts of odd-looking URLs passing through CoDeeN, often having the same URI portion of the URL and different host names. We found lists of such URLs on the Web, designed to remotely test known buffer overflow problems, URL parsing errors, and other security weaknesses. In one instance, these URLs triggered an intrusion detection system, which then identified CoDeeN as the culprit.

3 Solutions

Our guiding principle in developing solutions to address these security problems above was to allow users at PlanetLab sites as much access to the Web as they would have without using a proxy, and to allow other users as much "safe" access as possible. To tailor access policies, we classify client IP addresses into three groups – those local to this CoDeeN node, those local to any site hosting a PlanetLab node, and those outside of PlanetLab.

3.1 Rate Limiting

The "outside" clients face the most restrictions on using CoDeeN, limiting request types as well as resource consumption. Only their GET requests are honored, allowing them to download pages and perform simple searches. The POST method, used for forms, is disallowed. Since forms are often used for changing passwords, sending e-mail, and other types of interactions with side-effects, the restriction on POST has the effect of preventing CoDeeN from being implicated in many kinds of damaging Web interactions. For the allowed requests, both request rate and bandwidth are controlled, with measurement performed at multiple scales – the past minute, the past hour, and the past day. Such accounting allows short-term bursts of activity, while keeping the longer-term averages in control.

To handle overly-aggressive users we needed some mechanism that could quickly be deployed as a stopgap. As a result, we added an explicit blacklist of client IP addresses, which is relatively crude, but effective in handling problematic users. This blacklist was not originally part of the security mechanism, but was developed when dictionary attacks became too frequent. We originally analyzed the access logs and blacklisted clients conducting dictionary attacks, but this approach quickly grew to consume too much administrative attention.

The problem with the dictionary attacks and even the vulnerability tests is that they elude our other tests and can cause problems despite our rate limits. However, both have fairly recognizable characteristics, so we used those properties to build a fairly simple signature detector. Requests with specific signatures are "charged" at a much higher rate than other rate-limited requests. We effectively limit Yahoo login attempts to about 30 per day, frustrating dictionary attacks. We charge vulnerability signatures with a day's worth of traffic, preventing any attempts from being served and banning the user for a day.

Reducing the impact of traffic spreaders is more difficult, but can be handled in various ways. The most lenient approach, allowing any client to use multiple nodes such that the sum does not exceed the request rate, requires much extra communication. A stricter interpretation could specify that no client is allowed to use more than K proxies within a specified time period, and would be more tractable. We opt for a middle ground that provides some protection against abusing multiple proxies.

In CoDeeN, cache misses are handled by two proxies – one acting as the client's forward proxy, and the other as the server's reverse proxy. By recording usage information at both nodes, heavy usage of a single proxy or heavy aggregate use can be detected. We forward client information to the reverse proxies, which can then determine that a client is using multiple forward proxies. While forwarding queries produces no caching benefit, forwarding them from outside users allows request rate accounting to include this case. So, users attempting to perform Yahoo dictionary attacks (which are query-based) from multiple CoDeeN nodes find that using more nodes does not increase the maximum number of requests allowed. With these changes, login attempts passed to Yahoo have have dropped by a factor of 50 even as the number of attackers has tripled.

3.2 Privilege Separation

Protecting licensed content required more work, since it requires identifying what content is licensed. Using Princeton's e-journal subscription list as a starting point, we extracted all hostnames and pruned them to coalesce similarly-named sites, merging journal1.example.com and journal2.example.com into just example.com. We do not precisely associate subscriptions with universities, since that determination would be constantly-changing and error-prone. For clients trying to access licensed content, those that are local to the CoDeeN proxy are permitted, while others are currently given an error message. In the future, when dealing with accesses to licensed sites, we may redirect clients from other CoDeeN sites to their local proxies, and direct all "outside" clients to CoDeeN proxies hosted at sites without any subscriptions.

Protecting hosting sites from outside exposure cannot use the coarse-grained blacklisting approach suitable for licensed content. Otherwise, entire university sites and departments would become inaccessible. To address this problem, we use multiple proxies at different locations to de-escalate request privilege. We determine if a request to example.edu originates locally at example.edu, and if so, the request is handled directly by the CoDeeN node. Otherwise, the request is forwarded to a CoDeeN node at another hosting site. To eliminate the exposure caused by forwarding a request to a site where it is local, we modify our forwarding logic – no request is forwarded to a CoDeeN proxy local to the origin server.

Since our security mechanisms depend on comparing hostnames, we also disallow accesses to machines identified only by their numerical IP addresses. After implementing this approach, we found that some requests using numerical IP addresses were still being accepted. In the HTTP protocol, proxies receive requests that can contain a full URL, with hostname, as the first request line. Additional header lines will also identify the host by name. We found some requests were arriving with differing information in the first line and in the Host header. We had not observed that behavior in any Web browser, so we assume such requests were custom-generated, and modified our redirector to reject such abnormal requests.

4 Results

In the first eight weeks after its relaunch, CoDeeN has operated with only brief outages for software upgrades, and has serviced over 24 million requests from over 59000 unique IP addresses. Our traffic is growing, as shown in Figure 2, and we have become the most heavily-used service on PlanetLab. We have received a handful of queries/complaints from system administrators at the local PlanetLab sites, and all but one have been false alarms. Most queries have been caused by system administrators

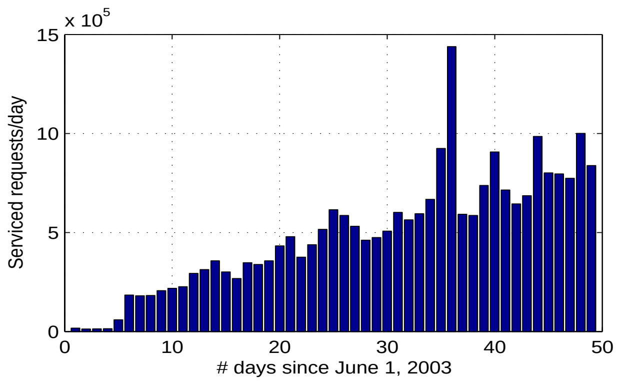

Figure 2: **Daily traffic on CoDeeN has been steadily increasing, and approaches one million serviced requests per day.**

or others using/testing the proxies, surfing through them, and then concluding that they are open proxies.

We have been using CoDeeN daily, and have found that the security restrictions have few effects for local users. Using non-Princeton nodes as our forward proxy, we have found that the restrictions on licensed sites can be overly strict at times. We expect that when we complete its implementation, bouncing requests to completely unprivileged nodes, the special handling for those sites will not be noticeable. By changing the configuration information, we have also been able to use CoDeeN as an outside user would see it. Even on our high-speed links, the request rates limits have not impacted our daily browsing.

Restricting outside users from using POST does not appear to cause significant problems in daily use. Searches are commonly handled using the GET method instead of the POST method, and many logins are being handled via HTTPS/SSL, which bypasses the proxy. The most noticeable restrictions on outsiders using POST has been the search function on Amazon.com and some chat rooms. In eight weeks of use, local users have generated fewer than 300 POST requests, with the heaviest generator being software update checkers from Apple and Microsoft.

Our security measures have caused some confusion amongst malicious users. We routinely observe clients testing proxies and then generating requests at very high rates, sometimes exceeding 50K reqs/hour. However, rarely do CoDeeN nodes see more than 20K valid reqs/hour. Some clients have generated over a million unsuccessful requests in stretches lasting longer than a day.

5 Related Work

Commercial content distribution networks, such as Akamai [1], Mirror Image [6], and Speedera [9], commonly deploy reverse proxies to replicate content, but their content is restricted to the Web sites of their customers. By using DNS-based redirection, or explicit URL modification [5], access to these systems is performed without explicit cooperation of the end user. Some concerns regarding malicious behavior are common with CoDeeN – clients could presumably request a large, previously uncached document via multiple proxies, causing a surge of demand at the origin servers. However, we are not aware of any public information on what mechanisms are used to handle such instances.

Cooperative proxy cache schemes have been previously studied in the literature [3, 8, 11, 13], and CoDeeN shares many similar goals. However, to the best of our knowledge, the only deployed systems have been based on the Harvest-like approach. Two large-scale cache projects involve providing proxy caches hierarchies. The main differences between CoDeeN and these systems are in the scale, the nature of who can access, and the type of service provided. Neither system uses open proxies. The NLANR Global Caching Hierarchy [7] operates ten proxy caches that only accept requests from other proxies and one proxy cache open to end users. Cache access is password-controlled and requires registration. The JANET Web Cache Service [4] consists of 17 proxies in England, all of which are accessible only to other proxies. Joining the system requires providing your own proxy, registering, and using an access control list to specify which sites should not be forwarded to other caches. Entries on this list include electronic journals.

6 Conclusions & Future Work

In CoDeeN, we have deployed a large-scale network of mostly-open proxies and addressed many of the security-related problems that have historically plagued such systems. These techniques have served us well, allowing us to keep our system running continuously for over four months, without having to worry about nodes being deactivated due to security/policy concerns. We feel that our approach to security, consisting of classification, rate limiting, and privilege separation, provides a model for other Web-accessible services.

The longer-term goal of our work in this area is to automatically detect and classify unusual traffic, and either use some general-purpose rules to handle it, or bring it to the attention of a human for further analysis. Such an approach would have the advantage that it could adapt to newer types of traffic, and would require less monitoring than some of our current techniques. It would also help to be able to identify unusual patterns automatically, in the event that determined malicious users attempt to escalate the sophistication of their attacks.

In this vein, some of our ongoing research involves detecting machine-generated requests and using this information in our policies. Previous work in this area has involved machine-learning techniques performing *post-facto* classification of requests [10]. Our approaches involve dynamically modifying HTML pages and determining how the changes affect the request stream. For example, if hidden links are inserted into pages, and are consistently traversed, chances are high that the request source is a crawler. Similar inferences can be drawn from inter-request timings, request URL similarity, and the downloading of embedded images and style sheets. By quickly classifying automated request generators, rates limits can be adjusted lower more quickly than would be appropriate for human users. In a similar vein, we may be able to permit more POST requests from outside users.

Acknowledgments

This research was supported in part by DARPA contract F30602–00–2–0561 and by in-kind support from Intel, Hewlett-Packard and iMimic Networking. We would like to thank Mic Bowman and everyone managing PlanetLab for their patience with complaints as we developed CoDeeN. We also thank our anonymous reviewers for their valuable comments on improving this paper.

References

[1] Akamai. Content Delivery Network. http://www.akamai.com.

[2] BOPM. Blitzed Open Proxy Monitor. http://www.blitzed.org/bopm/.

[3] A. Chankhunthod, P. B. Danzig, C. Neerdaels, M. F. Schwartz, and K. J. Worrell. A hierarchical internet object cache. In *USENIX Annual Technical Conference*, pages 153–164, 1996.

[4] JANET Web Cache Service. http://wwwcache.ja.net.

[5] D. Karger, A. Sherman, A. Berkheimer, B. Bogstad, R. Dhanidina, K. Iwamoto, B. Kim, L. Matkins, and Y. Yerushalmi. Web caching with consistent hashing. In *Proceedings of the Eighth International World-Wide Web Conference*, 1999.

[6] Mirror Image. http://www.mirror-image.com.

[7] National Laboratory for Applied Network Research (NLANR). Ircache project. http://www.ircache.net/.

[8] M. Rabinovich, J. Chase, and S. Gadde. Not all hits are created equal: cooperative proxy caching over a wide-area network. *Computer Networks and ISDN Systems*, 30(22–23):2253–2259, 1998.

[9] Speedera. http://www.speedera.com.

[10] P. N. Tan and V. Kumar. Discovery of web robot sessions based on their navigational patterns. In *Special Issue of the International Journal of Data Mining and Knowledge Discovery on Web Mining for E-commerce*, 2001.

[11] R. Tewari, M. Dahlin, H. M. Vin, and J. S. Kay. Design considerations for distributed caching on the internet. In *International Conference on Distributed Computing Systems*, pages 273–284, 1999.

[12] L. Wang, V. Pai, and L. Peterson. The Effectiveness of Request Redirecion on CDN Robustness. In *Proceedings of the Fifth Symposium on Operating Systems Design and Implementation*, Boston, MA, December 2002.

[13] A. Wolman, G. M. Voelker, N. Sharma, N. Cardwell, A. R. Karlin, and H. M. Levy. On the scale and performance of cooperative web proxy caching. In *Symposium on Operating Systems Principles*, pages 16–31, 1999.