**CERIAS Tech Report 2003-06**

# PROVABLE ACCESS ACCOUNTING FOR CONTENT DISTRIBUTION NETWORKS

by Radu Sion and Mikhail Atallah

Center for Education and Research in
Information Assurance and Security,
Purdue University, West Lafayette, IN 47907

# Provable Access Accounting for Content Distribution Networks [*]

Radu Sion
Computer Sciences & Center for Education and Research
in Information Assurance
Purdue University
(sion@cs.purdue.edu)

Mikhail Atallah
Computer Sciences & Center for Education and Research
in Information Assurance
Purdue University
(mja@cs.purdue.edu)

## Abstract

*One common revenue model in Content Distribution Networks (CDN), requires the CDN operator to provide access statistics (e.g. hits, transferred bytes) to the content provider, who in turn is expected to deliver payment dependent on these reported values. An implicit assumption of self-regulated truthfulness of the CDN operator governs this process. The content provider has to trust that the content distributor provides accurate numbers and does not artificially "inflate" them. This type of one-sided accounting is not tolerated well in two-party business interactions. An independent accuracy proof is preferred.*

*Here we present a provable secure verification mechanism for access accounting in this framework. Our solution exploits one of the common enabling mechanisms of CDN location awareness, dynamic DNS. We discuss several variations and analyze associated attacks.*

## 1 Introduction

Content Distribution Networks promise improved access times to online content. Given that most of the final content customers are human users, it is known [4] [6] that there exist upper bounds on the user patience-behavior with respect to content display times. A user waits only so much for a certain webpage to be displayed before it cancels and visits elsewhere (possibly a competitor). Thus it is natural to attempt to improve response times for web services.

[*]Portions of this work were supported by Grants EIA-9903545, ISS-0219560, IIS-9985019 and IIS-9972883 from the National Science Foundation, N00014-02-1-0364 from the Office of Naval Research, and by sponsors of the Center for Education and Research in Information Assurance and Security.

Various bottlenecks are to be found in any content producing web-application. An important first one is the request processing ability of the delivery front-end. A second bottleneck is directly related to the available front-end bandwidth at the content producing site.

Content Delivery Networks address both these issues by (i) content distribution and (ii) client nearest-location awareness. (i) is usually achieved by the deployment of an entire set of front-end machines that are effectively caching the same content for their clients while load-balancing the incoming request load, see Figure 1.

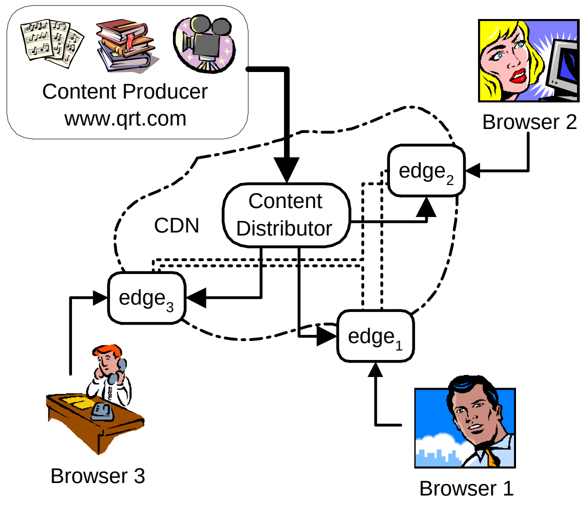

**Figure 1. Content flow in CDNs.**

Client location awareness for (ii) is commonly implemented through a modified naming resolution protocol (DNS) [2, 3, 5] such that clients (e.g. web browsers) that attempt to access a certain site *"www.qrt.com"* [1] are

[1]This domain name was chosen for exemplification purposes only. If in use, we claim no affiliation whatsoever.

receiving different, location-aware IP address answers [2]. In other words, different clients are "told" to connect to the "nearest" (in some metric of distance, e.g. client to server bandwidth) front-end. The CDN front-ends are thus named "edge-servers".

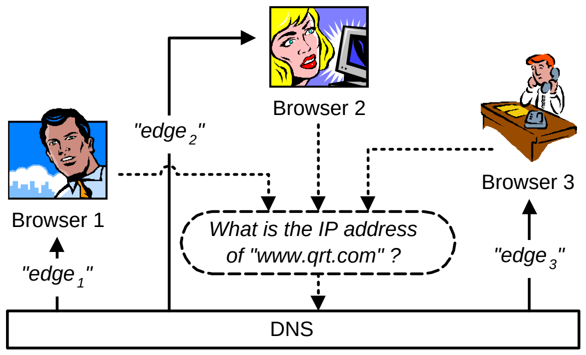

**Figure 2. DNS based location-aware redirection.**

In the past years CDNs have become enormously successful, in no small part due to the fact that they indeed deliver. Most revenue models in CDNs require maintaining access statistics (e.g. at the CDN operator site), that are to be used in the billing process to compute (possibly non-linear) proportional payment from the content provider (i.e. CDN customer). The content provider finds himself in the position to trust the CDN operator with respect to the delivered statistics. Providing an independent proof of data accuracy would not only boost CDN customer confidence, thus probably increasing its customer base, but also decrease internal accounting costs (now externally guaranteed).

For security and privacy reasons it is unreasonable to assume the CDN operator is going to be "opening up" its internals, giving access to (necessarily) *all* its customers to composing edge-servers and distribution logistics so that statistics can be accounted for directly by all parties. Other secure but less intrusive avenues need to be explored.

In this paper we introduce an un-intrusive, provable secure verification mechanism for access accounting for CDNs and associated hosted content. Our solution exploits the CDN location aware DNS mechanism by modifying it to provide a content producer direct access to provable access statistics. We discuss several variations and analyze associated attacks.

The paper is structured as follows. Section 2 introduces the main contribution and a related variant. Section 3 discusses a dispute scenario and various other attacks and associated issues. Section 4 concludes.

## 2 A Solution

The DNS infrastructure is one of the most valuable and important components of the Internet. Not only do the main Web protocols rely on DNS but also most of the existing communication and data distribution mechanisms require an assumption of availability, safety and consistency of the naming infrastructure.

When an attacker (Mallory) attempts to mount a name resolution infrastructure (DNS) attack it has to be contain-able and localized. Fortunately the hierarchical structure of the DNS facilitates these properties making it easier to both structurally contain and quickly localize faulty points (e.g. domain name hijackings). Top level domains in DNS (e.g. ".com", ".edu") are hosted in physically secure environments. A high level of redundancy is guaranteed by multiple mirrorings providing secure alternate naming authorities.

Our solution builds on this assumption of DNS upper-level security. We trust the DNS authority to perform non-maliciously. This trust is not a restrictive assumption as it derives from the conclusions above and does *not* extend to the actual communication protocol (e.g. DNS query transport). The assumption only stipulates that the DNS servers are not hijacked.

### 2.1 DNS Lookup Counting

Web page requests are usually preceded by an associated DNS lookup. Thus the first idea that naturally comes to mind is to perform access counting at the actual local DNS name authority. In other words, count all DNS requests for a specific URL and keep these counts at the DNS server site. The local namespace DNS serving can be delegated to a truly trusted third party, hired by both the CDN operator and the content provider for this very purpose (or specialized in providing such services). This effectively transforms the DNS lookup into a trusted counting machine for the accessed content, satisfying the requirements set in the beginning of this paper. Unfortunately there are important problems with this scenario:

**Accuracy:** The DNS lookup mechanism does not guarantee accuracy in terms of page accesses. Client DNS caching is the main reason for this. In other words, it is possible that a single DNS query is performed for an arbitrary number of document requests (e.g. HTTP requests). Often this behavior is also browser and content dependent, making it even more complex. At the extreme it can be entirely unrelated to the actual content access rate (e.g. proxying and full DNS caching).

One possible work-around, would be an additional step, estimating for each content provider, the ratio of total data transferred (or number of accesses) to number of DNS queries. That ratio naturally varies from one provider to the next, but for a particular provider that ratio (i) may be arguably fairly time-invariant, and (ii) may lend itself to estimation through periodic random sampling.

In other words, the ratio of DNS lookups to total data transferred for a particular content provider would be estimated every 3 months using a single "snapshot" random sampling experiment. After that, this ratio could serve as a conversion factor from number of DNS queries

[2]For more details on Internet naming conventions and resolution protocols see RFCs 1034, 1035, 2694, 2673, 2672, 2671, 2606 etc

to total data transferred. From then on all we'd have to do is count the DNS queries and use the conversion factor to estimate total data transferred.

While appealing in theory, this idea suffers from additional practical deployment problems. When and by whom is the "snapshot" experiment performed ? How does this cope with varying interests to the same content provider's data within a short time-frame (e.g. news providers) ?

Another apparent solution would be the disabling of DNS caching. But upon careful inspection, this becomes an impossible task, as, in this scenario, a DNS server query needs to be performed for each document access. The DNS infrastructure is not designed for this type of massive request floods. This will surely cause starvation if not provisioned for. Provisioning for such a case would incur unforeseen additional costs and major global-scale changes, probably out-weighting the benefits.

**Maintainability:** Extensive protocol and implementation modifications are required for this task. External storage needs likely to be deployed to keep counters for each link between the CDN operator and the content0 provider. DNS redundancy needs to be re-designed to account for multiple DNS lookups for the same destination at different DNS server clones etc.

**Multi-Hosts:** A certain domain could be hosting a set of different content-trees (e.g. yahoo.com hosts multiple online "shops"). DNS counting cannot distinguish among these, thus making it impossible to determine request distribution.

DNS counting, while initially tempting, proves not to be usable in this framework.

## 2.2 Probabilistic DNS Redirection

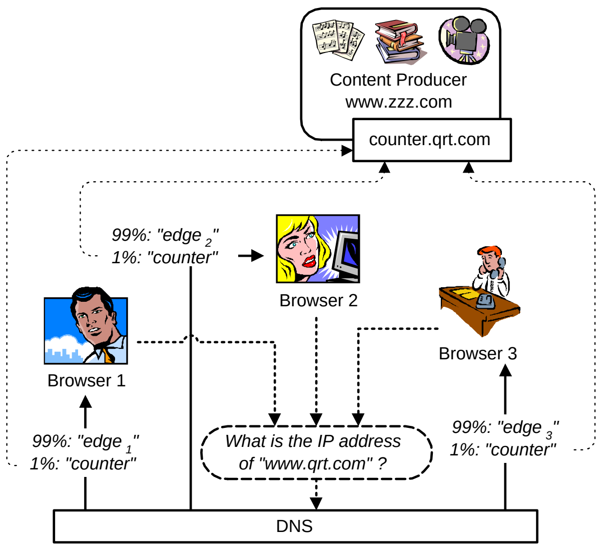

**Figure 3. Probabilistic DNS redirection.**

But surely a document request has to be preceded by a host name lookup. And while at fine-granularity level (i.e. several requests) there seems to be no direct link between number of requests and DNS queries (i.e. due to DNS caching etc.), there naturally exists an association at a lower-granularity level (i.e. large number of document requests). This association can be exploited. Instead of counting singular lookups at the DNS server side, we propose a different approach that aims at solving the above described problems of direct DNS lookup counting.

Remember that our case (i.e. CDNs), the DNS lookup is dynamic and location aware. Lookups for one hostname yield different IP address responses according to "who" (i.e. client) is asking, the aim being to identify a "closer" edge-server that can serve the client request.

We propose the modification of the DNS response pattern in such a way that while preserving its location awareness in most of its lookup responses, a small percentage of lookups (e.g. $p = 1\%$) are directed to a special content-provider operated server ("counter"), that is both able to count the document requests received as well as serve them (see Figure 3).

By knowing $p$ and the number of locally received requests $local\_requests$ the counter can estimate the total number of CDN handled requests with a high accuracy (over a large number of them): $estimate(total\_requests) = local\_requests \times \frac{1-p}{p}$.

One natural concern with placing a document request server at the content-provider site derives from the main raison d'etre of the CDN paradigm. Remember that distributing content to edge servers is rooted in the content producer's inability to handle the corresponding large number of requests and bandwidth requirements. But surely the content provider is able to serve a smaller amount of them [3]. Thus, by making $p$ arbitrarily small (but statistically relevant) the document request rate at the content provider's side can be kept under control.

Not only can we count request rates, but, because the counter server behaves like a clone (i.e. identical content) of any of the distributed CDN edge-servers, we can now also accurately estimate the amount of data transferred (not necessary related to number of requests) from CDN edge-servers to clients, similarly: $estimate(total\_bytes) = local\_bytes \times \frac{1-p}{p}$.

Effectively, the counter server "sees" a virtually identical client load behavior as do the CDN operated edge-servers.

## 3 Discussion

**Disputes.**

This solution offers counting ability at the content-provider site. But in (probably rare) cases, disputes may still arise if the CDN operator claims quite different numbers than estimated by the counter server. In this scenario, both of the parties "own their truth" and apparently we are back, close to square one [4].

[3]As it does this anyway in the interaction with the CDN operator, otherwise it can outsource the task it to a simple web-hosting service.

[4]Not exactly. Now the content provider is more informed by knowing its own estimates.

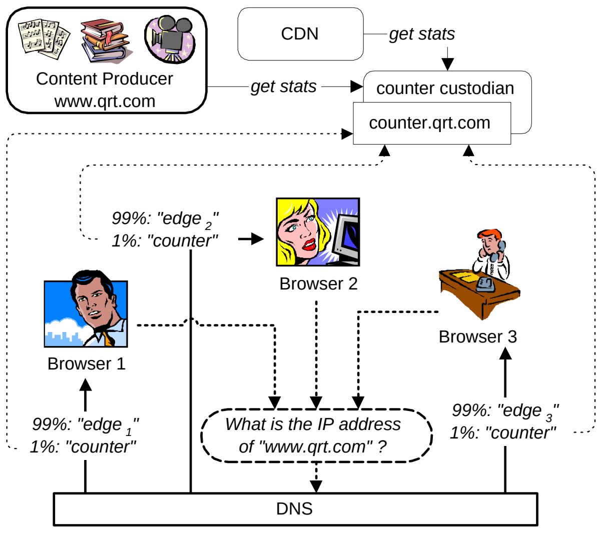

**Figure 4. Counter Custodian site maintains statistics.**

A natural solution here is placing the counter server in the custody of a trusted third party ("counter custodian"), possibly specialized in offering these kinds of services, or just a simple web service hosting provider that can operate a content providing server (see Figure 4). The introduction of this counter custodian is a simple fix for the above dispute issue.

**Attacks.**

Maybe the most important attack that can be mounted against probabilistic DNS redirection is the actual hijacking of the redirecting DNS server and the modification of the configured $p$ value inside the server. This is similar to a collusion attack (between the DNS server operator and the CDN) and it could benefit a malicious CDN operator that can increase $p$ to a higher value $p'$, without the content provider's knowledge. This would in effect raise the proportion of requests going to the content provider, resulting in an overpayment of $(100 \times (1 - \frac{p'}{p}))\%$.

This attack scenario is particularly challenging in that it seems like there is not much one can do about it. DNS integrity is at the foundation of most, if not all, Internet security protocols. Such an attack is likely to be discovered relatively fast. Also, a significant, unexplained increase in received local load at the counter server can be naturally used as an indicator of such a scenario.

Another attack that can be mounted by the CDN operator is the artificial increase of request load on the counter server (and associated revenue) through fake queries. A colluding "friend" can be asked to issue series of fake queries directed at the counter server. Those queries are obviously not yielding any actual business for the content producer but rather just increasing the perceived load. First it is to be noted here that as part of our solution, the CDN operator does not need to know the counter server name. Thus this attack is highly improbable. But even if this knowledge was somehow gained by the CDN operator, a solution could be to configure a set of alternative counter servers and periodically (without the knowledge of the CDN operator) change the name and/or IP address of the current server to which the "counting" requests (i.e. $p \times total\_requests$) are going. The frequency of this alternation does not need to be very high as the CDN operator is not supposed to know the counter in the first place.

**Dynamic Content.**

One special case of content delivery occurs when dynamic content is involved. When the documents delivered are a result of one or a set of underlying database calls, it is hard and often impossible to provide consistent delivery/caching, especially in cases of a high update frequency. This type of content is usually handled directly by the content provider or a set of separate application servers deployed with this specific purpose. Various other caching solutions [1] have been envisioned and there seems to be a trend moving the CDN away from content delivery toward a more generic "application delivery". The proposed solution has to be thus augmented with a mechanism of separation between content that is *not* actually hosted by the CDN itself (e.g. high-update, dynamic) and content that can be "counted" for CDN revenue purposes.

## 4 Conclusions

In this paper we introduced a solution enabling secure and trusted access accounting for content in the framework of CDNs. This is required as part of the revenue model in which content producer payment is proportional (possibly non-linearly) to content distributor load.

Our solution is based on probabilistic DNS redirection, a mechanism in which the DNS redirects a small percent of the content queries to a special "counter" server. We showed how disputes can be solved by placing the counter at a trusted "custodian" third party.

## References

[1] Oracle Corporation, Akamai Technologies, et al. ESI: Edge Side Includes.

[2] A. Iyengar, E. Nahum, A. Shaikh, and R. Tewari. Enhancing web performance, 2003.

[3] J Kangasharju, K. Ross, and J. Roberts. Performance evaluation of redirection schemes in content distribution networks.

[4] M. Koletsou and G. Voelker. The medusa proxy: A tool for exploring user-perceived web performance, 2001.

[5] B. Krishnamurthy, C. Wills, and Y. Zhang. The use and performance of content distribution networks, 2001.

[6] Ramakrishnan Rajamony and Mootaz Elnozahy. Measuring client-perceived response times on the www. In *USENIX USITS*, 2001.