Secure Access Accounting for Content Distribution Networks

Radu Sion
Department of Computer Sciences
Stony Brook University
Stony Brook, NY 11794
Phone: (631) 849-3700
Email: sion@cs.stonybrook.edu

Mikhail Atallah
Department of Computer Sciences
Purdue University
West Lafayette, IN 47907
Phone: (765) 494-6017
Email: mja@cs.purdue.edu

Abstract

Location aware Content Distribution Networks (CDNs) have become the de-facto standard for online content dissemination. One common revenue model in CDNs, requires their 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 becomes essential. In this paper we introduce a provable secure verification mechanism for access accounting in this framework. Our solution exploits one of the common enabling mechanisms of CDN location awareness, namely DNS redirection. We discuss several variations and analyze associated attacks. We experimentally validate the proposed solution.

Keywords

Secure Access Accounting, Content Distribution Networks

Secure Access Accounting for Content Distribution Networks

Radu Sion* Mikhail Atallah†

Abstract

Location aware Content Distribution Networks (CDNs) have become the de-facto standard for online content dissemination. One common revenue model in CDNs, requires their 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 becomes essential.

In this paper we introduce a provable secure verification mechanism for access accounting in this framework. Our solution exploits one of the common enabling mechanisms of CDN location awareness, namely DNS redirection. We discuss several variations and analyze associated attacks. We experimentally validate the proposed solution.

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 [8] [15] 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.

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.

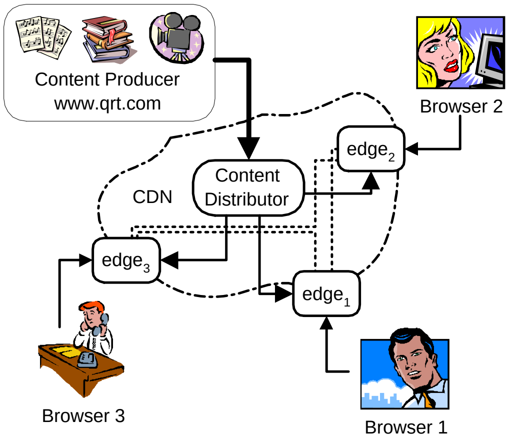

Figure 1. Content flow in a traditional CDN.

Client location awareness for (ii) is commonly implemented through a modified naming resolution protocol (DNS) [5, 7, 9] such that clients (e.g. web browsers) that attempt to access a certain site *"www.qrt.com"* are receiving different, location-aware IP address answers 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 (see Figure 2). The CDN front-ends are thus named "edge-servers" [1].

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,

*Computer Sciences, Stony Brook University, Stony Brook, NY 11794, sion@cs.stonybrook.edu

†Computer Sciences, Purdue University, West Lafayette, IN 47907, mja@cs.purdue.edu

[1] Please see [6] for an overview of CDN redirection schemes.

but also decrease internal accounting costs (now externally guaranteed).

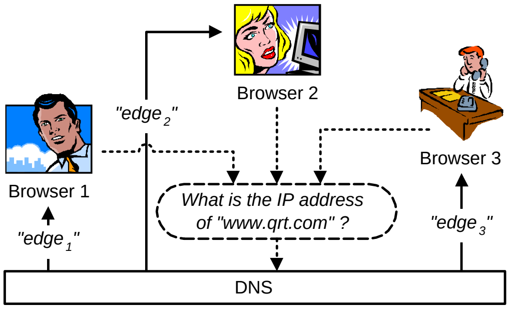

Figure 2. Location-aware DNS redirection.

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 a non-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 the proposed solution and evaluates dispute scenarios and various other attacks. Section 4 places our work in the context of related work and Section 5 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 Sampling

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 name-space 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 including:

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).

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 content 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.

2.2 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.

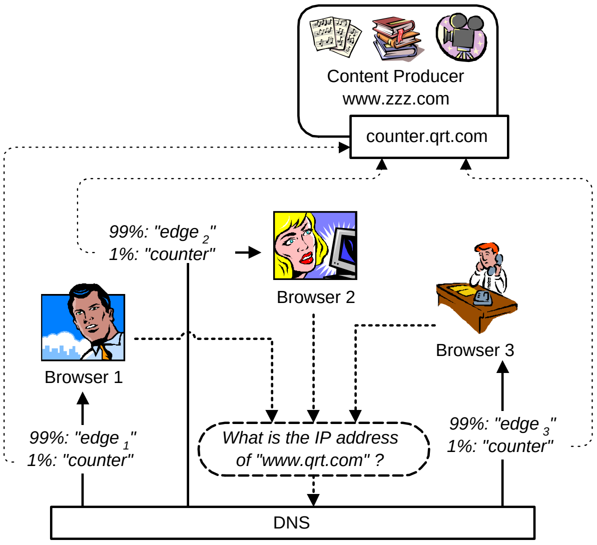

Figure 3. Probabilistic DNS redirection.

Recall 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 "sampling" server, 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 sampling server can estimate the total number of CDN handled requests with 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 outside of the CDN, 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. Is it safe to assume that the sampling server is able to handle the associated load ? Fortunately, this load is controllable. By making p arbitrarily small (but statistically relevant) the document request rate at the sampling server'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. See Section 3.4 for a quantitative validation experiment.

3 Discussion

3.1 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 sampling 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.

3.2 Disputes

While the above solution offers counting ability at the content-provider site, in (probably rare) cases, disputes may still arise if the CDN operator claims significantly different numbers than estimated by the counter server. In this scenario, both of the parties "own their truth" [2].

To avoid such sampling disputes between the content provider and the CDN operator, we propose placing the sampling server in the custody of a (possibly specialized) trusted third party ("sampling custodian"), for example a web service hosting provider that can operate a content providing server (see Figure 4). The statistics provided by the counter server, being out of reach for both the CDN operator and the content provider, can be trusted [3].

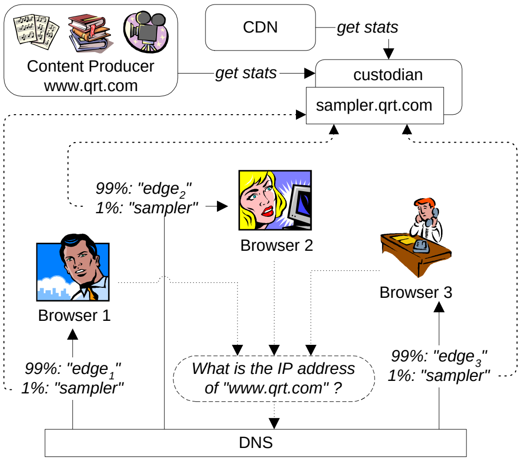

Figure 4. Probabilistic DNS redirection. To avoid disputes, the sampling server is placed under the control of a "sampling custodian".

3.3 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 [2] 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.

3.4 Experiments: Client Load Per Lookup

Another (apparent) issue to be considered is the fact that the client load for each DNS lookup varies largely. A lookup from a location likely to be information savvy with respect to the given content (a location that targets the content well) will be higher than the average. While this is a valid argument for a singular, short-term experiment, in the proposed scheme, accounting goes on continuously. The actual client load is sampled over a significantly longer period (e.g. a month's worth). The law of large numbers applies here and, on average, the load patterns are going to be proportionally similar at the counter server with respect to the actual CDN sites. to validate this assertion, we performed sampling experiments over 40 days (between 12/18/2003 and 01/27/2004), 3236534 million individual hits worth of weblog data from our departmental web server support this argument. A 2% HTTP isolated session-level sample resulted in roughly 2.007% of the total number of page accesses and 1.992% of the total server load in terms of bytes transferred. The Law of Large Numbers indeed applies.

4 Related Work

Somewhat related efforts can be found on topics such as secure hit-metering and network traffic accounting [1] [3] [4] [6] [10] [11] [12] [13] [14]. Many of these and other industry efforts (e.g. ipro.com, interse.com) are based on the idea of introducing a third party auditing module integrated at server side, (hopefully) trusted by all paying clients. While there might be some validity to these ideas, the significant economic incentive of the server operator to break into and alter such a module, especially if hosted at the server side, is not to be overlooked. A design that would result in a trusted and distributed auditing mechanism would be more desirable.

With this in mind, existing seminal research by Naor and Pinkas in [11] and Pitkow [14] stands out. In these and other efforts, the authors propose an interesting and valid scheme for client-server accounting based broadly on the idea of secret sharing. Clients that are to access a server are given a "share" of a greater secret and, when connecting to the server a client-server exchange is performed in which the server is provided with that particular share. Only when a certain number k of different clients have connected to a particular server, can it re-construct the secret, which effectively can act as a proof for this.

This scheme is definitely superior to server-side auditing modules. Nevertheless, there are a set of drawbacks

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

[3] Note that this mechanism is significantly cheaper, more scalable and secure than having a third party install auditing modules at the CDN site and witnessing each and every incomming web transaction.

that this scheme presents. Each client access has to be preceded by a complex Additionally, in the proposed implementation, each web-page requires the integration of a complex (java) applet as part of it. This is likely not desirable, as the *start* → *run* → *shutdown* cycles of java applications within browsers tend to significantly slow down the average website [16] responsiveness (the display of each initial webpage and subsequent ones will require an applet to be started, run and stopped). Additionally this is also not feasible for clients with no java capabilities at all, such as mobile palm browsers and cell phones, increasingly popular frameworks. The initial scheme also requires the client to periodically connect to a trusted third party to retrieve associated keys and secret shares which complicate the process even more. Because of the actual secret sharing mechanisms deployed, this scheme can be only deployed for a limited number of accesses before the entire auditing process needs to be restarted, hardly an acceptable circumstance in today's dynamic and largely unpredictable web.

Some of these issues in the initial scheme have been partially tackled in follow-up efforts of the authors or others. However, we believe that while in limited scope applications this scheme could be (arguably) deployed, on a large scale, certain requirements are to be observed which might render it less applicable. These are: (i) the accounting process should be entirely transparent to clients/browsers (e.g. no java applets, no additional client load), (ii) there should be no complex client-server interaction required (e.g. no pre-connection secret share distribution), (iii) there should be a minimal level of trust associated with any additional third party or (preferably) no third party at all, (iv) the accounting scheme should be designed so as to be able to run un-interrupted, without periodic re-starts (e.g. no secret share renewal), (v) a mechanism for accounting in such a framework should be as simple as possible but no simpler.

5 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 "sampling" server, placed at a trusted "custodian" third party. We experimentally validated the mechanisms and showed that indeed sampling techniques can be applied to accurately account for incurred server hits.

References

[1] Carlo Blundo, Annalisa De Bonis, Barbara Masucci, and Douglas R. Stinson. Dynamic multi-threshold metering schemes. *Lecture Notes in Computer Science*, 2012:130–131, 2001.

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

[3] Matthew K. Franklin and Dahlia Malkhi. Auditable metering with lightweight security. In *Financial Cryptography*, pages 151–160, 1997.

[4] P. Hallam-Baker. W3c working draft for proxy caches, 1996.

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

[6] J. Kangasharju and K.W. Ross abd J.W. Roberts. Performance evaluation of redirection schemes in content distribution networks. 2000.

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

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

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

[10] J. Mogul and P.J. Leach. Simple hit metering for HTTP (IETF draft), 1997.

[11] Moni Naor and Benny Pinkas. Secure and efficient metering. *Lecture Notes in Computer Science*, 1403:576–576, 1998.

[12] T. Novak and D. Hoffman. New metrics for web media: toward web measurement standards. *World Wide Web Journal*, 2(1):213–246, 1997.

[13] Wakaha Ogata and Kaoru Kurosawa. Provably secure metering scheme. *Lecture Notes in Computer Science*, 1976:388–389, 2000.

[14] J. Pitkow. In search of reliable usage data on the WWW. In *Proceedings of the Sixth International WWW Conference*, 1997.

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

[16] Julie Ratner. *Human Factors and Web Development, Second Edition*. Lawrence Erlbaum Associates, 2002.