Deflation-Secure Web Metering*

Rob Johnson†
University of California at Berkeley
rtjohnso@cs.berkeley.edu

Jessica Staddon
Palo Alto Research Center
staddon@parc.com

Abstract

As a result of recent changes in the policy governing Internet content distribution, such as the institution of per listener royalties for Internet radio broadcasters, content distributors now have an incentive to under-report the size of their audience. Existing metering protocols only protect against inflation of audience size. We introduce the first protocols for audience metering that protect against deflation attempts by content distributors. The protocols trade-off the amount of additional information the content distributors must distribute to facilitate metering with the amount of infrastructure required and are applicable to Internet radio, web plagiarism, and software license enforcement.

1 Introduction

Internet content distributors often need to prove to a third party that they have a certain number of visitors or listeners. Such information is usually used to set advertising rates, so content distributors have an incentive to inflate these numbers. Franklin and Malkhi introduced inflation-secure metering schemes [15] to prevent this type of fraud and many variations and improvements have subsequently been proposed (e.g., [23, 20, 24]). The most secure of these protocols are token-based, meaning clients possess special tokens that are hard for the content distributor to generate. When clients request content they provide the distributor with a token and the distributor is able to use the tokens to generate a short "proof" that it has the claimed number of clients. Because the content distributor cannot generate the tokens itself, it is unable to inflate its client count.

With the advent of per-listener royalty fees for Internet radio [17] and the growth of web content plagiarism [13], content distributors now have an incentive to report artificially small audiences, but no existing protocol prevents such behavior since the distributor can simply ignore the portions of its interactions with clients that are necessary for metering. For example, with a token-based metering scheme the distributor may simply discard client tokens to lower its client count.

We present two new client metering protocols that prevent content distributors from reporting artificially low audience sizes. Both protocols leverage the relative anonymity of the Internet to enable an auditor to monitor the content distributor's interactions with clients. Essentially, the auditor poses as a client in its interactions with the content distributor. Since the content distributor cannot distinguish the auditor from any other client, it cannot disregard the metering

*A preliminary version of this paper appeared in the 2002 ACM Workshop on Digital Rights Management.

†Rob Johnson was an intern at PARC while conducting this research.

portion of the protocol without risking detection. Previous metering schemes use secret-sharing to distill a short proof of audience size from all the client protocol interactions. Since these techniques aren't deflation-secure, we use other techniques to decrease communication overhead. Both of our protocols are quite practical, requiring only a few bytes per client interaction.

Our first protocol (see Section 3) requires essentially no additional infrastructure. The content distributor simply maintains a Bloom filter [5] that encodes the client IDs (anonymized to preserve privacy) of all clients who have requested the content. The Bloom filter is small in applications such as micro-broadcasting. The protocol offers protection against deflation because the auditor can verify that its anonymized ID was one of the inputs to the filter. This protocol cannot detect inflation, but it can be combined with a standard inflation-secure scheme to detect both types of cheating.

The second protocol (see Section 4) uses encryption to offer protection against both inflation and deflation. A trusted party randomly allocates to each client a subset of a global set of keys. The content distributor makes the content publicly available (e.g. by posting a file on the web) in encrypted form using an encryption key known to all of its clients. If the keys are allocated according to a well-chosen distribution, then the auditor can estimate the number of clients based only on the encryption key the content distributor is using. Hence, the encryption key serves the same purpose as the Bloom filter in the previous protocol; both encode the client requests. This protocol requires essentially no additional communication (that is, other than the encrypted content) on the part of the content distributor, but doesn't completely preserve the privacy of the clients. Table 1 summarizes the main features of our protocols.

Finally, we observe that deflation security essentially requires the content distributor to prove the *lack* of knowledge of client requests and so is an inherently harder problem than inflation security which is solvable by requiring proofs of knowledge of client requests. Hence, we believe that any solution to this problem will involve the imperceptible monitoring of content distributors for protocol compliance, and therefore, anonymous networks, as our protocols do. Note that the current Internet offers relative anonymity and, by virtue of dynamically assigned addresses and dial-up connections, relative unlinkability. Further, emerging peer-to-peer technologies may support perfect anonymity in the near future. Thus, we analyze our protocols in the context of perfect anonymity, and believe they degrade gracefully in the current Internet.

Overview. This paper is organized as follows. We discuss related work in Section 1.1. Our model is described in Section 2. We present and analyze an easily implemented deflation-secure protocol in Section 3 and a deflation-secure protocol with constant overhead in Section 4. We conclude in Section 5 with open problems.

1.1 Related Work

One of the first methods for counting the number of visitors to a web site is due to Franklin and Malkhi [15]. Naor and Pinkas [23] present a protocol with stronger security guarantees [15]. Ogata and Kurosawa [24] identify flaws in the Naor and Pinkas scheme, and propose their own. The Naor-Pinkas model has been generalized and analyzed extensively [6, 20, 7, 28]. In a similar vein, Kuhn [19] presents a scheme by which an auditor can efficiently verify the number of unique signatures on a document, with applications to digital petitions and web metering.

The methods currently used to measure audience size are far more primitive than anything proposed in the above papers. The simplest audience measurement technique counts the number of entries in the server's log files [9]. Since it is easy for the server administrator to delete or

Scheme	Protocol 1	Protocol 2
Deflation protection	Yes	Yes
Inflation protection	No	Yes
Privacy preserving	Yes	No
Communication overhead	$O(n)$	$O(1)$
Counts cumulative audience	Yes	Yes
Counts current audience	Yes	No

Table 1: The main features of the schemes presented in this paper. The number of clients is denoted by n.

insert entries into the log files, these numbers cannot be trusted. In the specific case of counting the number of visitors that see an advertisement, the trustworthiness of the measurements can be improved by having the advertising agency serve the ad directly [14]. In this arrangement, the ad agency can under-report the number of ads it serves, thus lowering the advertising fees it pays. Reiter, Anupam, and Mayer [26] propose a scheme for detecting this sort of fraud. Conversely, Mayer, Nissam, Pinkas and Reiter [3] describe general attacks for inflating the number of ads that appear to be served through a given web page.

The size of a particular website's audience can also be gauged by consumer surveys and focus groups [2, 27]. These numbers can be fairly accurate, but this method is expensive. Some audience measurement services combine log analysis and consumer surveys [2, 10]. Similarly, audience size can be measured by having web surfers keep a diary of the sites they visit, although these numbers are prone to accidental error as much as malicious mis-reporting [12, 27].

All the audience measurement techniques above are designed for determining advertising rates and thus are only concerned about attempts by the content distributor to inflate the audience size. In all the schemes above except the survey and diary methods, the content distributor can easily deflate the size of her audience. In the context of advertising, content distributors have no incentive to do so, hence this has not been a problem. This is not the case when the audience size is being measured to determine royalty fees. Ours are the first schemes we know of that attempt to prevent the content distributor from deflating her audience size.

Finally, we note that secure voting (see for example, [8, 25]) is also concerned with accurate audience measurement. However, voting protocols tend to be fairly heavyweight due to the requirements of that setting (e.g. public verifiability) hence we don't believe those techniques are directly applicable to the content distributor setting.

2 Model

Our protocols involve a content distributor (CD), clients, and in the case of the protocol of Section 4, a trusted party (TP). We call a client that is interested in ascertaining the CD's audience size, an *auditor*. The goal of our protocols is the production of a trustworthy measurement of the number of client requests the CD has received even though the CD has an incentive to deflate this measurement. We typically denote the actual number of clients over a specified interval by n, and the maximum number of clients over the same duration by n_{max}. Although we describe our protocols in the context of measuring client requests over an extended time interval (as is done, for example, with web site "hit" counts) the protocol of Section 3 can easily be adapted to measure the number of currently active clients (or streams, in unicast applications).

2.1 Requirements

Our goal is to achieve security guarantees that are comparable to those of the ideal model in which a trusted third party ensures accurate, deflation-secure audience metering, but with substantially more practical protocols. This suggests a trade-off between efficiency and accuracy and we believe this is unavoidable. Hence, we require that our protocols are *tunable* to the desired level of accuracy. Our objective is to maximize the gain in accuracy that comes with each decrease in efficiency.

Tunable Deflation-Security and Accuracy. In most of the applications we consider, it makes sense to assume that CDs and clients are aligned against the auditor, hence we need to protect against attempts by the CD and the clients to conduct their transactions "under the table", and other collusion attacks. We offer such protection by monitoring CD/client interactions to check for protocol compliance. Consequently, one immediate way to increase the level of deflation-security is to increase the number of content requests by the auditor. We show in Section 3 that in practice a high level of deflation-security can be achieved with a reasonable number of content requests by the auditor. In addition, we note that the CD should have to collude with a large number of clients and negotiate a new protocol in order to achieve significant deflation. Such a high degree of collusion may be detectable by the auditor because of the resulting discrepancy between the apparent audience size and the congestion at the site.

Accuracy is also impacted by the method of encoding client requests, that is, the more lossy the encoding, the less accurate the resulting client count. Increasing accuracy by improving the quality of the encoding is hence, very protocol-specific. In Sections 3 and 4 we discuss ways in which accuracy can be improved in the protocols of those sections by increasing the amount of information in the encodings.

Efficiency. We measure efficiency in terms of the communication overhead, client storage and infrastructure requirements of the protocols. As mentioned earlier, our goal is to get as close to the the ideal model as possible (i.e. no additional communication overhead or client storage) with a far more practical protocol. Our first protocol comes with no additional client storage and essentially no infrastructure requirements, but incurs a communication cost that's on the order of the number of clients. Our second protocol has constant overhead but requires significant client storage and infrastructure.

Privacy. A deflation-secure client metering protocol should preserve the client's privacy since all that is being measured is a count. We note, however, that there is some advantage to providing client anonymity while allowing request linking as this allows the CD to detect efforts at artificially inflating its client counts. Our protocols only require anonymity, not unlinkability.

2.2 Limitations

Our protocols prevent compliant content distributors from reporting deflated audience sizes. As long as the clients and content distributors use our protocols, we can establish tight bounds on the probability that they can cheat successfully. However, our protocols are not complete solutions to this problem. To see why, consider a small internet radio station that broadcasts music on a well-known port. The radio station faithfully executes one of the deflation-secure metering schemes described below. Thus, the auditor can be sure that the radio station only has a few listeners *on that port*. Our protocols cannot prevent the radio station from boadcasting to a large number of users from a second port that is spread by word-of-mouth among faithful fans[1].

We can view the broadcaster described above as two separate radio stations. Now the problem of accurately tracking audience size splits into two problems. First, we need a reliable scheme for measuring the size of each content distributor's audience. Our schemes address this problem. Second, we need a reliable scheme for tracking all relevant content distributors. This is a hard problem that we do not address in this paper. We do note that in order for the attack described above to be effective, the radio station must collude with a large number of listeners to ignore our protocols. The logistics of maintaining a large, secret network will impose a natural limit on the scale of this sort of cheating.

The client anonymity we require can also be used against the content distributor. The auditor (or any other client) may artificially inflate the audience size by repeatedly requesting the content as a new client. Our protocols do not explicitly protect against this. One possible remedy is to insert a trusted party between the distributor and the clients with anonymous communication only between the trusted party and the distributor. If the content distributor suspects this attack is underway, the trusted party's logs can be examined. Of course, requiring a trusted party for the sole purpose of protecting against this attack is suboptimal, however if the protocol is such that a trusted party is already required (as is true of the protocol in Section 4) then this approach is worth considering.

2.3 Applications

There are a number of settings in which client metering protocols that are secure against deflation are necessary.

Internet Radio. The Internet has given rise to hobbyist Internet radio broadcasters which have extremely small audiences. For example, according to live365.com, there are over 1000 Internet radio stations with less than 100 listening hours per month; e.g. these stations have an average of less than one listener tuned in for 3 hours each day. A client metering protocol may be used to prove this fact to an organization such as the RIAA.

Screen-Scraping. Websites that provide a useful service, such as Yahoo's real-time stock prices, often get "screen-scraped" by other web services [13]. The scraping service simply fetches the information from the original service, parses the desired data out of the returned web page, repackages it in a new format, and finally presents it to the client. As long as the screen-scraping service does not overuse the original service provider, this behavior can be tolerated. If the scraping service agrees to use one of our protocols, then the originating web service provider can audit the scraping service to ensure that it is not abusing the original service provider.

Distribution of Licensed Content. Consider a web site that holds a limited distribution license for content (e.g. movies, music files or software). Our protocols can be used to ensure that the distributor does not exceed the license.

Web Advertising As described in Section 1.1, some web advertisers serve their ads directly, and hence can under-report the number of ads they serve in order to reduce the fees they must pay to carrying websites. Our protocols can detect this type of fraud.

[1]Listeners that take advantage of this back door may lose their anonymity, but we cannot guarantee the anonymity of clients that don't use our protocols.

3 Estimating Audience Size with Minimal Infrastructure

Our first protocol is very easy to adopt and can be adapted to support either total request counting or current client set counting. Its main drawback is that the bandwidth required is linear in the size of the audience, but this protocol is quite efficient for scenarios in which the number of clients is small, as is the case for several of our intended applications (e.g. Internet radio micro-broadcasters).

The protocol uses Bloom filters [5], so we give a brief introduction to them here. A Bloom filter is a lossy representation of a set and consists of a bit-vector $\vec{b}$ of length m and s independent hash functions $h_1, \ldots, h_s : \{0,1\}^* \rightarrow \mathbb{N}$. [2] In the literature of Bloom filters, m is called the *width* of the filter. Initially, the bit vector is all zeros. To insert an element x into the set represented by the Bloom filter $\vec{b}$, set the bits $\vec{b}[h_1(x) \bmod m] = \cdots = \vec{b}[h_s(x) \bmod m] = 1$ (if a bit is already set to 1 then it remains 1). To test whether x is an element of the set represented by Bloom filter $\vec{b}$, test that $\vec{b}[h_1(x) \bmod m] = \cdots = \vec{b}[h_s(x) \bmod m] = 1$. Note that this test can lead to false positives; this is why the Bloom filter is termed "lossy". If $\vec{b}[h_i(x)] = 0$ for some i, then x cannot be in the set. Bloom filters do not support item removal.

Let $w(\vec{b})$ denote the Hamming weight of $\vec{b}$. The probability that a bit is 1 in a Bloom filter of width m after n insertions using s hash functions is $1 - (1 - \frac{1}{m})^{ns}$. So given a filter $\vec{b}$, we can estimate the number of insertions which have been performed on $\vec{b}$ by $I(\vec{b}) = \frac{\ln(1-w(\vec{b})/m)}{s\ln(1-1/m)}$. To minimize the probability of a false positive, s should be chosen so that $s = (\ln 2)m/n$, which gives a false positive rate of $\left(\frac{1}{2}\right)^{(\ln 2)m/n} \approx (0.6185)^{m/n}$. Hence, by varying the width of the Bloom filter we can tune the accuracy of the protocol, with obvious consequences for the communication overhead incurred. For example, if $m/n = 8$, the false positive rate using $s = 5$ is 0.0216 and the overhead is $O(n)$ on each request. Finally, if $\vec{b_1}$ and $\vec{b_2}$ are two Bloom filters of the same width, then we say $\vec{b_1} \leq \vec{b_2}$ if $\vec{b_1}[i] \leq \vec{b_2}[i]$ for all i.

The protocol is illustrated in Figure 1. Each content distributor maintains a Bloom filter of width $m = cn$, where n is the average number of requests seen by the content distributor each week and c is a parameter agreed upon in advance. In practice, $c = 8$ works well as discussed above. When a client sends a request to the content distributor, the content distributor and client engage in a coin flipping protocol to agree on an r bit nonce N and the content distributor inserts N into the Bloom filter. Any standard coin flipping protocol will work [16]. They then proceed with their normal protocols. Each week, for example, the content distributor sends the Bloom filter to the auditor and then starts again with a fresh filter. The auditor checks that the Bloom filter it receives, $\vec{b}$, has $w(\vec{b}) \leq 2m/3$ and that any nonces it negotiate with the CD are present in the filter. For example, if the auditor has sent k requests to the CD, the auditor checks that all their nonces, $N_1, \ldots, N_k$, are present in the Bloom filter that the CD submits for that interval. Provided these conditions are satisfied, the auditor computes an estimate of the number of requests seen by the content distributor via $I(\vec{b}) = \frac{\ln(1-w(\vec{b})/m)}{s\ln(1-1/m)}$. The requirement that $w(\vec{b}) \leq 2m/3$ is a technical constraint necessary to guarantee that the estimate $I(\vec{b})$ is sufficiently accurate (see Theorem 1).

For small content distributors, this scheme is very efficient. Using the ratio $m/n = 8$ mentioned above, the content distributor must send the auditor about 1 byte per join. So, for example, a content distributor that receives 20 requests each day would only have to send a

[2]The hash functions need not be cryptographically secure. They are just used to map the universe of objects down to integers.

$$
\begin{array}{ccc}
A & & CD \\
N & \xleftrightarrow{\text{r coin flipping protocol}} & N \\
 & & \vec{b} \leftarrow \texttt{Insert}(\vec{b}, N) \\
 & \xleftrightarrow{\text{normal content distribution protocol}} &
\end{array}
$$

Figure 1: The cumulative request counting version of the Bloom-filter protocol. The content distributor is denoted by CD. The client, A, must be anonymous, and N is the result of executing a coin flipping protocol for r coins.

140 byte message to the auditor each week. Thus this scheme is completely feasible for small to medium content distributors. Even a relatively large content distributor with around 150 requests per day would only have to send a 1K weekly message to the auditor. In the context of Internet radio broadcasters, these overheads are very small since the average audio stream takes at least 2K/s.

3.1 Analysis

The quantity $I(\vec{b})$ is an accurate estimate of the size of the content distributor's audience. The following theorem implies that if we use $I(\vec{b})$ as an estimate of the number of requests received by the content distributor then, with extremely high probability, the actual number of requests will differ from our estimate by at most $\alpha\sqrt{m}$ for a small value of α.

Theorem 1 *Fix $n_{max} < \frac{m \ln s}{s}$ and $W < (1 - \frac{1}{s})m$. Let X be a random variable representing the set of nonces received by the content distributor. We model X as taking on values at random from the set $\{\{x_1, \ldots, x_n\} | x_i \in \mathbb{Z}/2^r\mathbb{Z}, 0 \leq n < n_{max}\}$. Let $\vec{B}[X]$ denote the Bloom filter representation of X, and $w(X) = w(\vec{B}[X])$. Then*

$$\Pr[||X| - I(\vec{B}[X])| \geq \alpha\sqrt{m} \mid w(X) = W] = O\left(\sqrt{m}\exp\left(\frac{-(\alpha-1)^2}{2}\right)\right).$$

Proof. By Bayes' Theorem,

$$\Pr[|X| = n \mid w(X) = W] = \frac{\Pr[w(X) = W \mid |X| = n]\Pr[|X| = n]}{\sum_{i=0}^{M}\Pr[w(X) = W \mid |X| = i]\Pr[|X| = i]}.$$

Since we are estimating $|X|$ from $w(X)$, we assume that $|X|$ is uniformly distributed. [3] Letting $K = \sum_{i=0}^{M}\Pr[w(X) = W \mid |X| = i]$ and simplifying gives

$$\Pr[|X| = n \mid w(X) = W] = \frac{\Pr[w(X) = W \mid |X| = n]}{K}.$$

Except for the factor of K, the LHS of this equation is just the well-known occupancy distribution derived from tossing n balls into m bins. Let $\mu(i) = E[w(X) \mid |X| = i] = (1 - (1 - \frac{1}{m})^{is})m$. When $\mu(i) < (1 - \frac{1}{s})m$ (or, equivalently, when $i < \frac{m \ln s}{s}$), then $\frac{d\mu}{di} > 1$.

[3] This is a common but controversial assumption in Bayesian analysis. The controversy arises because the validity of the analysis depends on this assumption, but the assumption cannot be verified statistically. For the purposes of bounding the tail probabilities, the uniform distribution is a relatively pessimistic choice, hence we believe it is a safe one. A similar situation arises in Section 4.

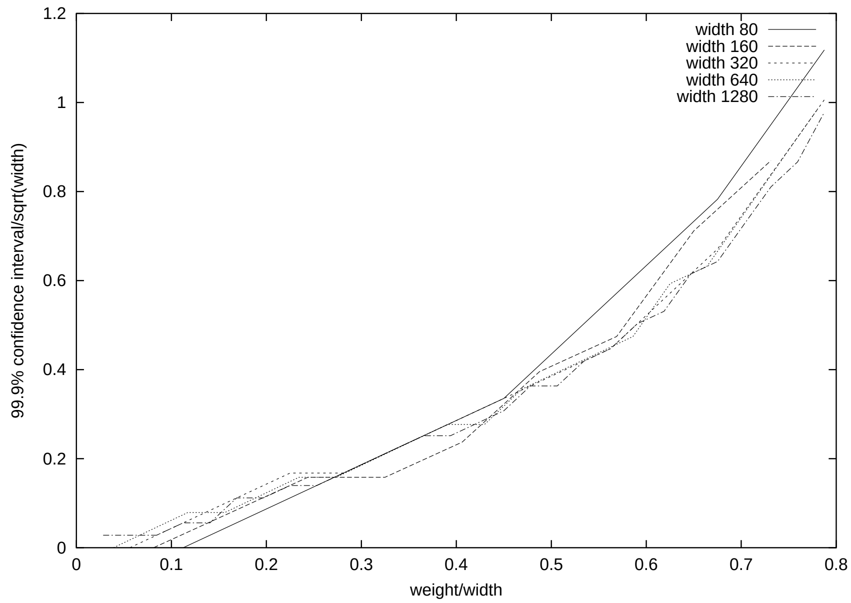

Figure 2: The accuracy of using $I(x)$ to estimate the number of insertions performed on a Bloom filter. Note that the confidence intervals have been normalized to $\sqrt{m}$. Since our protocol requires that content distributors submit Bloom filters $\vec{b}$ with $w(\vec{b}) \leq \frac{2m}{3}$, we can conclude that with 99.9% confidence, the actual number of requests received by the content distributor differs from $I(\vec{b})$ by at most $\frac{4\sqrt{m}}{5}$.

By Kamath, Motwami, Palem, and Spirakis' Occupancy Bound [18],

$$\Pr[|w(X) - \mu(|X|)| \geq \theta\mu(|X|)] \leq 2\exp\left(\frac{\theta^2\mu(|X|)^2(m-1/2)}{m^2 - \mu(|X|)^2}\right).$$

By combining this bound with the Bayesian equation above and unenlightening algebraic manipulation, one can derive that

$$\begin{aligned}\Pr[||X| - I(W)| \geq \alpha\sqrt{m} \mid w(X) = W] &\leq \frac{4\sqrt{m}}{K}\sum_{i=\alpha}^{\infty}\exp\left(\frac{-(i-1)^2}{2}\right) \\ &= O\left(\sqrt{m}\exp\left(\frac{-(\alpha-1)^2}{2}\right)\right)\end{aligned}$$

The only tricky part of the derivation is to use that $|i - I(W)| \leq |W - \mu(i)|$, which holds because $\frac{d\mu}{di} > 1$. $\square$

In practice, $I(\vec{b})$ is a much better estimate of the number of requests than this theorem predicts. Figure 2 shows the width of the 99.9% confidence interval for several choices of m. As the figure shows, as long as $w(\vec{b}) \leq 2m/3$ as required by our protocol, then with 99.9% confidence, $|I(\vec{b}) - |X|| \leq \frac{4\sqrt{m}}{5}$. So for example, using a Bloom filter $\vec{b}$ with $m = 640$, if $w(\vec{b}) = 320$, then with 99.9% confidence, the actual number of insertions performed on the filter is between 80 and 100.

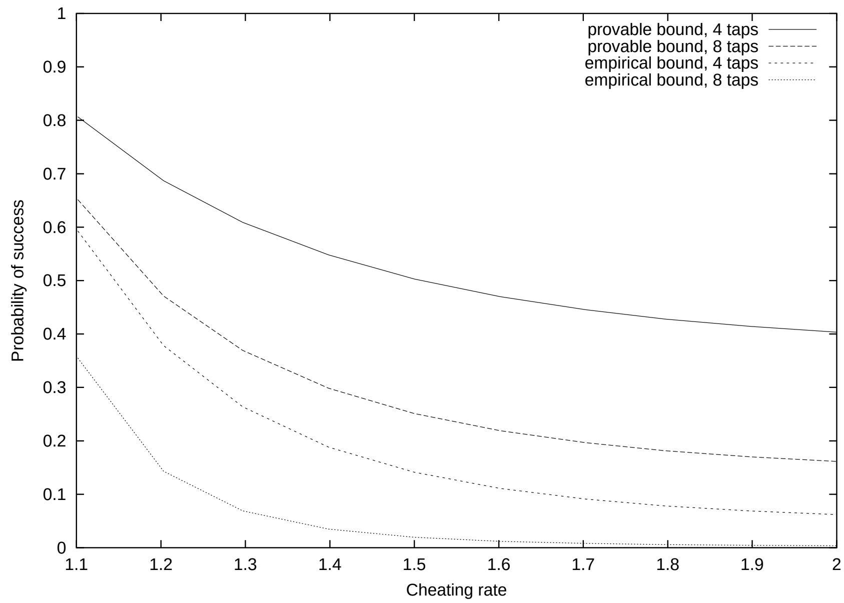

Figure 3: The probability that a content distributor can fool the auditor, assuming $m = 1024$, $s = 5$, and the content distributor is allowed to report Bloom filters with weight at most 512, which corresponds to 128 requests. The top two curves are provable bounds: a content distributor cannot fool the auditor with probability better than these curves indicate. The bottom two curves are empirical bounds: based on computer simulations, we believe that a content distributor cannot fool the auditor with greater probability than these curves indicate. So for example, if a content distributor receives $1.3*128$ requests, and the auditor sent 8 auditing requests, then the content distributor's chances of successfully convincing the auditor that he only received 128 requests is less than 10%.

In general, the content distributor can attempt to cheat during an auditing period by reporting a Bloom filter $\vec{b'} < \vec{b}$, where $\vec{b}$ is the correct Bloom filter containing all requests for the auditing period. The auditor detects this cheating if there exist i and j such that $\vec{b'}[h_i(N_j)] = 0$. The following Proposition describes the content distributor's optimal strategy and bounds his chances of success.

Proposition 2 *Suppose the content distributor receives n requests, but wishes to report only $L < n$ of those requests. Let $\{J_1, \ldots, J_n\}$ be the set of nonces generated by servicing the requests, and $\vec{b}$ be the Bloom filter generated from $\{J_1, \ldots, J_n\}$. Then the content distributor's optimal strategy is to report a Bloom filter $\vec{b'}$ containing the largest subset $S \subseteq \{J_1, \ldots, J_n\}$ such that $I(w(\vec{b'})) \leq L$. If $w(\vec{b}) - w(\vec{b'}) = D$ and the auditor sent k requests to the content distributor, then*

$$\Pr[\text{content distributor succeeds}] \leq \frac{\binom{n-k}{D/s}}{\binom{n}{D/s}}$$

Proof. The content distributor gains nothing by reporting a Bloom filter $\vec{b'} \not\leq \vec{b}$, since it does not decrease his chances of being caught. If there exist i, j such that $\vec{b'}[h_i(J_j) \bmod m] = 0$, then setting $\vec{b'}[h_{i'}(J_j) \bmod m] = 1$ for $i' \neq i$ does not decrease the content distributor's chances of being caught. Hence the content distributor's optimal strategy is to report a Bloom filter $\vec{b'}$ containing some subset $S \subseteq \{J_1, \ldots, J_n\}$.

To decrease the weight of the Bloom filter by D, one must remove at least D/s items, since each item can decrease the weight of the filter by at most s. Since the content distributor cannot distinguish the auditor's requests, his best strategy is to select the largest S such that $w(\vec{B}[S])$ is below the allowed threshold. We may assume that for any $J_j \in \{J_1, \ldots, J_n\} \setminus S$, there exists an i such that $h_i(J_j \bmod m) = 0$ since otherwise the content distributor could add J_j to S without affecting the weight of $\vec{B}[S]$. So cheating successfully requires selecting (at least) D/s items from $\{J_1, \ldots, J_n\}$ without selecting one of the k requests sent by the auditor. The probability of doing this is $\frac{\binom{n-k}{D/s}}{\binom{n}{D/s}}$.

□

Again, the bounds in this proposition are not as tight as possible. In practice, the content distributor will have to omit considerably more than D/s requests in order to reduce the weight of the reported Bloom filter below the allowed threshold. To get a better idea what the real chances of cheating successfully are, we wrote a computer program to simulate a content distributor trying to cheat by finding the optimal subset S described in the above proposition. Based on our experiments, the content distributor has to remove at least $D/2$ items from $\{J_1, \ldots, J_n\}$ in order to decrease the weight of his Bloom filter by D. Figure 3 compares the probability of successfully cheating estimated from the above proposition and the probability of success derived from our experiments. As the graph shows, the actual probability of cheating is much lower than the proposition indicates.

This scheme preserves audience anonymity. The content distributor and client use a coin flipping protocol to agree on the nonce to be placed in the Bloom filter. Since this nonce is generated randomly, it cannot reveal anything about the identity of the client. This strong guarantee of privacy has a downside: a malicious client can send many requests to the content distributor, artificially inflating the audience size. Since this scheme provides total listener anonymity, the content distributor cannot identify the attacker. Also, a content distributor and a group of cooperative clients can agree to always generate the same nonce, hence all the clients would appear to be just one client, deflating the content distributor's audience.

We have described this scheme in terms of request-counting, but it can also be used to count current audience size. Suppose the auditor wants to know the current audience size at each minute. Then the content distributor simply inserts the IDs for all its active clients into a Bloom filter every minute and sends this off to the auditor. To audit, the auditor anonymously requests content from the content distributor and verifies that it is counted among the active streams. Although the reporting overheads are obviously increased in such a scheme, they are still quite low. For example, an Internet radio station with 20 listeners will have to send the auditor about 20 bytes of data every minute, which is quite modest. The above accuracy and security analysis apply directly to this scheme, too.

Finally, this scheme can be further improved by using compressed Bloom filters [21] to reduce the false positive rate without increasing the size of messages sent to the auditor.

4 Estimating Audience Size with Constant Overhead

In the following protocol, the auditor is able to infer the audience size from a constant number of bits that are associated with the (encrypted) content. The protocol offers security against both deflation *and* inflation of audience size. It is most naturally applicable to the distribution of fairly static content, for example, a web site that provides software or movies in encrypted form available for download and decryption with payment. When used with real-time content, the content distributor must be using the network as a broadcast channel in order for the auditor to be assured the measurements are accurate. The drawback of the protocol is that it requires a keying infrastructure. As in Section 3, the basic protocol is essentially a metering scheme in that it counts *hits* (or, joins). In Section 4.2, we discuss extensions to the basic protocol that allow demographic information to be extracted from the content and the current audience size (i.e., not just the cumulative audience) to be estimated.

In this protocol, each client stores a set of encryption keys issued by a trusted party (TP). In the initial phase of the protocol, the TP sends all the keys to the content distributor. When a client requests the content, the TP gives some subset of the keys to the client and sends the ID number of each of the client's keys to the CD. To distribute content to the current set of clients, the CD forms the intersection of the clients' key sets, T, and chooses a key from T for encrypting the content. Because the TP assigns keys to clients probabilistically, the auditor (who may be the same as the TP) when requesting the content anonymously[4] (e.g. by visiting the distributor's web site), can infer the audience size from the encryption key in use.

The TP assigns keys to clients as follows. First, the entire set of keys is partitioned into t sets, $S_1, \ldots, S_t$. Each client receives any particular key with a fixed, independent probability. For keys in the same set S_i, this probability is the same. By choosing the sets $\{S_i\}_{i=1}^{t}$ to be of decreasing size (as i increases), but with increasing associated probabilities, the TP can control the proportion of keys in T that are in any S_i given the audience size. More precisely, if the audience is small, T is dominated by keys from S_1, but as the audience grows, the proportion of keys in T that are in S_1 will be far less than the proportion that are in S_i for $i > 1$. Hence, because the content distributor doesn't have any a priori knowledge of the composition of the sets $\{S_i\}_i$, the distributor is unable to distinguish between the keys in T and so the choice of $k \in T$ is a reflection of the distribution of T, and by inference, the audience size. Figure 4 demonstrates how T, may change over time. For illustrative purposes, keys with higher probabilities are indicated by larger ovals.

The following makes the protocol more precise.

Basic Protocol. This protocol takes as input a positive integer m representing the number of keys in the system, a positive integer t, and positive integers $s_1, \ldots, s_t$ such that $s_1+s_2+\ldots+s_t = m$. The keys are partitioned into t sets, $S_1, \ldots, S_t$, such that for each i, $|S_i| = s_i$, where $s_1 > s_2 > \ldots > s_t$. For each $i = 1, \ldots, t$ and any $k_j \in S_i$, there is a probability p_i that the TP will assign key $k_j \in S_i$ to any given client (keys are assigned independently), where $p_1 < p_2 < \ldots < p_t$. Numbers ϵ_1, ϵ_2, $0 < \epsilon_1, \epsilon_2 < 1$, are also input to provide a gauge of the accuracy of the audience measurements. These parameters imply an upper bound, n_{max}, on

[4] Receiving the content anonymously also allows the auditor to determine that the content distributor isn't distributing keys to clients (to maintain the appearance of a small audience) or abusing the protocol in some other way. For applications in which the surreptitious distribution of keys to clients by the content distributor is a real concern, a simplified version of the analysis in Section 3 can be performed to calculate the frequency with which the auditor should request the content.

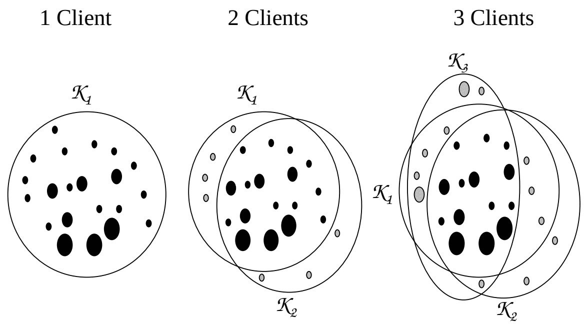

Figure 4: The black ovals represent keys in the set T when there are 1, 2 and 3 clients. That is, $\mathcal{K}_i$ denotes client i's key set, for $i = 1, 2, 3$. The larger ovals correspond to keys that are more likely to be assigned to any given client. As the number of clients grows the proportion of large ovals in T increases. In this way, the key that's selected from T reflects the audience size.

the number of joins that can be accurately measured by the system. The variable n is used to denote the actual number of joins. The protocol consists of the following steps:

1. The TP randomly generates m keys, $k_1, \ldots, k_m$, and sends them to the content distributor.
2. Upon contacting the content distributor, a client, u_i, receives a set of keys $\mathcal{K}_i \subseteq \{k_1, \ldots, k_m\}$ from the TP. For $j = 1, \ldots, m$, $k_j \in \mathcal{K}_i$ with probability p_r if $k_j \in S_r$. The TP sends the content distributor the ID numbers of the client's keys[5].
3. To distribute content to clients $u_{j_1}, \ldots, u_{j_r}$, the content distributor chooses a key $k \in T = \mathcal{K}_{j_1} \cap \ldots \cap \mathcal{K}_{j_r}$ and encrypts the content (or perhaps, a key that is used to encrypt the content) with k. A fresh key should be chosen regularly.(The frequency with which this is done provides a way to tune the accuracy of the protocol.)
4. Periodically, the auditor requests content and notes the key, k, that the content distributor is using in Step 3. There exists $i \in \{1, \ldots, t\}$ such that $k \in S_i$. The auditor calculates the distribution of the random variable that measures the proportion of keys in T that are in S_i as a function of n, $(\frac{|T \cap S_i|}{|T|} | n)$, to within a confidence level of $1 - \epsilon_1$. Using this distribution, the auditor determines a range $[n_1, n_2]$ such that for each $n \in [n_1, n_2]$, $P(k \in S_i | n) \geq \epsilon_2$, and estimates[6] the audience size as being in this range.
   - To increase the likelihood of inferring audience size correctly, the auditor can monitor the content through several key changes. In addition, accuracy can be tuned by requiring the CD to choose new keys more frequently (e.g. with each new song).

[5]We suggest that the TP send the keys rather than the client, so that the client cannot cause the audience size to appear larger than it is by sending only a subset of their keys to the content distributor.

[6]Note that the probability that directly infers audience size is $P(n = x | k \in S_i)$. Since the distribution on n is unknown we cannot calculate this probability precisely. However, provided some information on the distribution of n is available, this probability can be derived from the one we know by using: $P(n = x | k \in S_i) = \frac{P(k \in S_i | n=x) P(n=x)}{P(k \in S_i)} \geq P(k \in S_i | n = x) P(n = x)$. For example, if $P(n = x) \geq \alpha$ for all x, then we have an upper bound: $P(n = x | k \in S_i) \geq \alpha P(k \in S_i | n = x)$, and if n is uniformly distributed (as is assumed in Section 3 to achieve analysis benefits that don't seem to occur for this protocol), we have an equality: $P(n = x | k \in S_i) = c_i P(k \in S_i | n = x)$ where $c_i = \Sigma_{y=1}^{n_{max}} P(k \in S_i | n = y)$. Hence, we believe $\{P(k \in S_i | n = x)\}_x$ is sufficient to infer the value of n as being in $[n_1, n_2]$.

– If the auditor has contacted the content distributor previously and received a different set of keys, the auditor should check that k is also in that key set. Alternatively, the auditor can request the content as several different clients and perform the same checks. If any of these checks fail, the content distributor is not following the protocol.

This protocol relies on the content distributor's inability to distinguish between the keys in the intersection, T. The content distributor can gain such an ability in the following ways. First, a key that is *not* known to any of a large set of clients is less likely to be in S_t than a key in T. However, provided the distributor follows the protocol and encrypts the content so that all of the audience can decrypt it, the distributor is unable to make use of this information. The other information the content distributor learns about the keys comes from bills (e.g. licensing royalties). For example, if the distributor is charged less when using key k than when using key k', the distributor knows the index j_k such that $k \in S_{j_k}$ is less than the index $j_{k'}$ such that $k' \in S_{j'_k}$. To remedy this, we suggest that the system be refreshed with every bill (e.g. once a month).

There is also the possibility that the content distributor attempts to cheat in a similar way as in our first protocol, namely by removing some users' key sets from the calculation of the intersection, T, in order to get a larger set from which to draw the encryption key. We argue that it is unlikely this attack will be successful. First, cheating in this way can have the effect of preventing some users from accessing the content (which should generate complaints). Second, it is difficult to guarantee that a small audience will be inferred by the auditor because the key allocation algorithm is probabilistic. That is, if the content distributor chooses a key that is not known to several of the clients then there is still some probability that this key is in S_i for large i, in which case a large audience will be inferred. To guarantee that a small audience will be inferred, the content distributor has to use a key that's not known to several clients, in which case the distributor may indeed only be able to reach a small audience.

Finally, the content distributor can potentially benefit from collusion with clients or other content distributors. If the TP is using the same global set to allocate keys to clients of different content distributors (which is a desirable practice because it can allow clients to "surf" multiple distributors without needing to repeat the initialization phase) then the distributors (and users) may be able to distinguish between keys that they wouldn't have been able to otherwise. However, as mentioned earlier, this may be only of limited value because a key that causes a small audience to be inferred does so because it is only likely to be stored by a small number of clients.

4.1 Analysis

In this section we develop equations that allow the auditor to execute the protocol. First, we find an accurate approximation to the distribution of $(\frac{|T \cap S_i|}{|T|}|n)$. Let $\beta_{x,i} = s_1{p_1}^x + \ldots + s_{i-1}{p_{i-1}}^x + s_{i+1}{p_{i+1}}^x + \ldots + s_t{p_t}^x$.

Lemma 3 *Let $0 < \delta < 1$. For $i = 1, \ldots, t$ and $n = x$, $P(k \in S_i | n = x)$ is at least as large as $\frac{(1-\delta)s_i{p_i}^x}{(1+\delta)\beta_{x,i}+(1-\delta)s_i{p_i}^x}$ and at most as large as $\frac{(1+\delta)s_i{p_i}^x}{(1-\delta)\beta_{x,i}+(1+\delta)s_i{p_i}^x}$ with probability at least $1 - \epsilon_1$, when $(\frac{e^\delta}{(1+\delta)^{1+\delta}})^{s_t {p_1}^{n_{max}}} \leq \frac{1-(1-\epsilon_1)^{1/t}}{2}$ and $e^{-\delta^2 s_t {p_1}^{n_{max}}/2} \leq \frac{1-(1-\epsilon_1)^{1/t}}{2}$.*

Proof. For $i = 1, \ldots, t$, when the number of clients is x, the random variable $|T \cap S_i|$ is binomially distributed with size s_i and probability ${p_i}^x$. Hence, the expected value of $|T \cap S_i|$ is $s_i p_i^x$. Applying

Chernoff bounds (see, for example, [22]), it follows that, $|T \cap S_i| \in [(1-\delta)s_i p_i^x, (1+\delta)s_i p_i^x]$ with probability at least $(1-\epsilon_1)^{1/t}$ when both $(\frac{e^\delta}{(1+\delta)^{1+\delta}})^{s_i {p_i}^{n_{max}}} \leq (\frac{e^\delta}{(1+\delta)^{1+\delta}})^{s_t {p_1}^{n_{max}}} \leq \frac{1-(1-\epsilon_1)^{1/t}}{2}$ and $e^{-\delta^2 s_i {p_i}^{n_{max}}} \leq e^{-\delta^2 s_t {p_1}^{n_{max}}/2} \leq \frac{1-(1-\epsilon_1)^{1/t}}{2}$. Hence, $P(k \in S_i | n = x) = \frac{|T \cap S_i|}{|T|} = \frac{|T \cap S_i|}{|T \cap S_1| + \ldots + |T \cap S_t|}$ is in the interval stated in the lemma with probability at least $(1 - 2\frac{1-(1-\epsilon_1)^{1/t}}{2})^t = 1 - \epsilon_1$. □

From the above lemma, it follows that the auditor needs to find x values such that $\frac{(1-\delta)s_i {p_i}^x}{(1+\delta)\beta_{x,i} + (1-\delta)s_i {p_i}^x} \geq \epsilon_2$ to complete the protocol. In addition, n_{max}, s_i and p_i must be chosen to satisfy Lemma 3, for example, by using the bounds in the following corollary.

Corollary 4 *To satisfy step 4 of the basic protocol it suffices (but isn't generally necessary) to choose* $n_{max} \leq \frac{\ln(\frac{c(\epsilon_1,\delta,t)}{s_t})}{\ln p_1}$ *and* $s_i \geq \frac{c_i(\epsilon_1,\delta)}{{p_i}^{n_{max}}}$ *for all i, where $c(\epsilon_1, \delta, t)$ and $c_i(\epsilon_1, \delta)$ are defined below. Provided these inequalities are met, the expected number of keys that a client must store is at least* $\Sigma_{i=1}^t \frac{c_i(\epsilon,\delta)}{p_i^{n_{max}-1}}$.

Proof. The constant $c_i(\epsilon_1, \delta)$ in the upper bound on s_i comes from solving the following two inequalities used in the proof of Lemma 3: $(\frac{e^\delta}{(1+\delta)^{1+\delta}})^{s_i {p_i}^{n_{max}}} \leq \frac{1-(1-\epsilon_1)^{1/t}}{2}$ and $e^{-\delta^2 s_i {p_i}^{n_{max}}/2} \leq \frac{1-(1-\epsilon_1)^{1/t}}{2}$. It follows that $c_i(\epsilon_1, \delta) = \max\{\frac{2\ln(\frac{1-(1-\epsilon_1)^{1/t}}{2})}{-\delta^2}, \frac{\ln(\frac{1-(1-\epsilon_1)^{1/t}}{2})}{\ln(\frac{e^\delta}{(1+\delta)^{1+\delta}})}\}$.

The bound on n_{max} follows similarly with $c(\epsilon_1, \delta, t) = \min\{\frac{2\ln(\frac{1-(1-\epsilon_1)^{1/t}}{2})}{-\delta^2}, \frac{\ln(\frac{1-(1-\epsilon_1)^{1/t}}{2})}{\ln(\frac{e^\delta}{(1+\delta)^{1+\delta}})}\}$.

The lower bound on the expected number of keys per client follows by substituting the lower bound for s_i into the quantity, $\Sigma_{i=1}^t p_i s_i$. □

For illustrative purposes[7], we conclude this section with a small example.

Single Threshold Example. The following example shows how the basic protocol can be used to determine that a threshold number of clients has been achieved. Let $s_1 = 37000$, $p_1 = .6$, $s_2 = 370$, $p_2 = 1$ and $n_{max} = 13$. Because $|T \cap S_2| = 370$ with probability 1, we need only find a confidence interval for $|T \cap S_1|$ and this will imply confidence intervals for $|T \cap S_1|/|T|$ and $|T \cap S_2|/|T|$. Setting $\delta = .2$, by the proof of Lemma 3 we need the following inequality to hold: $(.98)^{s_1 {p_1}^{13}} < \frac{\epsilon_1}{2}$. Solving for ϵ_1 yields, $\epsilon_1 \geq .75$. If we choose $\epsilon_2 = .75$, then with at least .75 confidence, it follows by solving the inequality, $\frac{(1-\delta)37000(.6)^x}{(1-\delta)37000(.6)^x + 370} \geq .75$ for x, that $P(k \in S_1 | n \leq 6) \geq .75$. Similarly, by solving, $\frac{370}{(1+\delta)37000(.6)^x + 370} \geq .75$ we get, $P(k \in S_2 | n \geq 12) \geq .75$. Hence, if $k \in S_1$ the auditor returns the interval $[1, 6]$ for n and if $k \in S_2$ the interval $n \geq 12$ is returned. This is depicted in Figure 5.[8]

In this example, we expect a client to store 22,570 keys. If the keys are each 64 bits long, this represents .17 megabytes of keying material. While this is significant, it is a fraction of the space required by most media players (for example, it's about .09 of the download size of WinAmp.com's "full" player). Viewed differently, after listening to streaming music at a data

[7] In general, it is unwise to choose $p_2 = 1$ and $t = 2$ because the content distributor then knows that any key, k, that's not stored by all the clients, is in S_1 with probability 1. However, even in this example it's arguable that using key k yields a successful attack, since we expect k to only be stored by around 7 clients ($.6n_{max}$) which is already very close to the 6 client audience that the auditor will infer from the usage of k.

[8] Note that the confidence intervals hold up to $n = 13$ only.

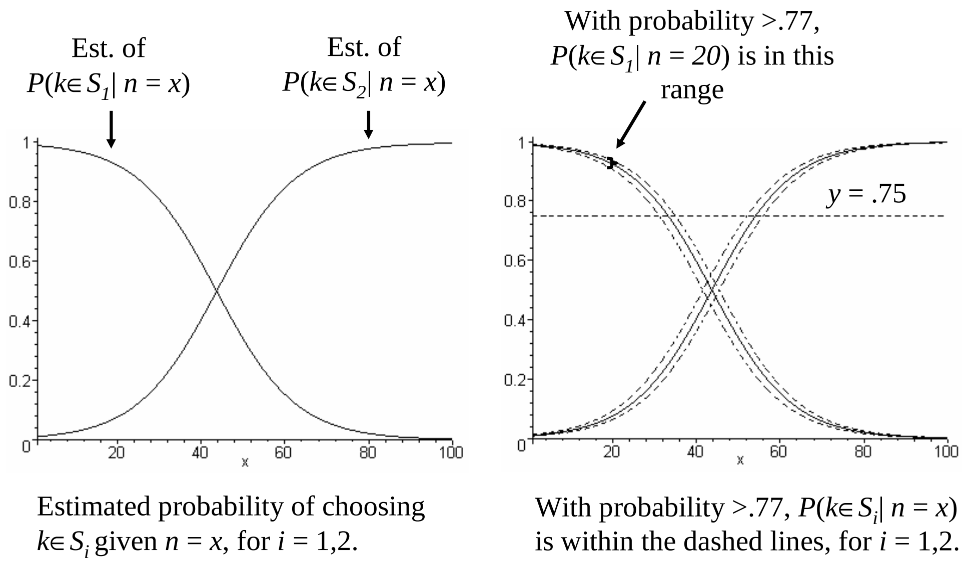

Figure 5: In the left-hand side of the figure we graph, $\frac{p_i^x s_i}{p_1^x s_1 + p_2^x s_2}$ for $i = 1, 2$ (where $p_1 = .6$, $p_2 = 1$, $s_1 = 37000$, $s_2 = 370$) as estimates for $P(k \in S_1|n = x)$ and $P(k \in S_2|n = x)$. $P(k \in S_1|n = x)$ and $P(k \in S_2|n = x)$ are within the distance indicated by the dashed lines of their respective estimates with probability at least .75.

rate of 28.8 kilobits per second for less than 20 minutes, the keying material is less than .0425 of the audio data that's been downloaded.

Since a client will typically have more than half of the 37,370 keys in this example, the TP can tell the content distributor the keys the client *doesn't* have more efficiently than listing the keys the client does have, in step 2 of the protocol. Since the key IDs are less than 16 bits long, we expect this step to require the transmission of at most 29 kilobytes of data. Using compression, this can probably be reduced to only 10 kilobytes. Again, this is only necessary when the client first requests the content.

4.2 Extensions

Multiple Content Distributors. The basic protocol is easily modified to allow the trusted party to use a single set of keys for multiple content distributors. In step 2, each user sends keys that are computed as the output of a one-way function applied to each of the keys received from the TP concatenated with the CD's ID. Because the CDs have distinct IDs it is computationally infeasible for them to determine which of their received keys are the same.

Privacy and Demographics. Note that this protocol is not completely privacy preserving because the auditor learns something about the clients, namely, that they have key k. However, if there is sufficient separation between the auditor and the TP it will be difficult for the auditor to make use of this information. In addition, we note that it may be possible to use this aspect of the scheme to embed demographic information. For example, although men and women should with high probability receive the same number of keys in S_i, the particular keys they tend to receive may be partly a function of their sex. Hence, the auditor may be able to infer the predominant sex of the audience from the content distributor's choice of encryption key in S_i.

MEASURING THE CURRENT AUDIENCE. The protocol described above is best suited to estimate *cumulative* audience size, for example, the number of hits received by a web site over a certain period of time. In some settings, this may be the only possible measure of audience size. For example, in multicast applications, the content distributor typically only is informed of new additions to the multicast group and is unlikely to know when a member leaves [4]. Hence, by observing the content distributor's behavior, or by querying directly, it may only be possible to learn the cumulative audience. In this case, behavioral patterns may be used to infer current audience size from cumulative data.

It may also be possible to modify the basic protocol to measure audience size directly. The key idea is that if the auditor can observe the content for long enough[9] to gain an accurate estimate of the entire contents of T, then the *current* audience may be inferred. The entire contents of T are necessary because the content distributor gains some ability to distinguish keys from every new client. For example, if k is stored by several clients but k' is only known to a few, then k' may be a cheaper key for the content distributor to use because it may imply a smaller audience in the basic protocol ($k' \in S_i$, $k \in S_j$, where $i < j$). Hence, if the audience shrinks and k' ends up being a key all the current clients know, the content distributor may seek to mislead the auditor by only using k'. However, if the content distributor is required to change keys frequently (e.g., a different key for every few songs) and the auditor listens long enough to determine that k' is the only key in use, an alarm will be raised as the probability that the content distributor would be left with only k' at some point is very low. One problem with this is that a key that is known to clients who are no longer in the audience may be selected as the encryption key.

5 Open Problems

We've introduced the first metering protocols that provide security against deflation attempts by the content distributor. The protocols assume an anonymous network and as this is a strong assumption it would be interesting to explore what other assumptions can yield deflation security.

In addition, each of our protocols requires some a priori knowledge of the maximum audience size. Although this seems like a reasonable assumption for the applications we consider, it would be useful to design a scheme that can efficiently adapt to unanticipated surges in audience size. Ideally, such a protocol would provide content access to only the current set of clients while preserving privacy and enabling efficient auditing.

Finally, inflation-resistant schemes can use secret sharing to produce short proofs of the size of a content distributors audience, but our protocols require either long proofs or large keying infrastructure. Is it possible to construct a deflation-secure metering scheme that uses short proofs with minimal infrastructure?

Acknowledgements

The authors thank Ian Smith for suggesting this problem and Prateek Sarkar, David Goldberg, Dirk Balfanz and Dan Greene for helpful discussions.

[9] This requirement may be easy to meet because the auditor may need to observe the content for a long time in order to preserve anonymity.

References

[1] Anonymizer.com. Company web page. http://www.anonymizer.com.

[2] Arbitron, Inc. An analysis of streaming audience measurement methods. http://www.measurecast.com/docs/Audience_Measurement_Methods.pdf.

[3] Vinod Anupam, Alain Mayer, Kobbi Nissim, Benny Pinkas, and Michael K. Reiter. On the security of pay-per-click and other Web advertising schemes. *Computer Networks (Amsterdam, Netherlands: 1999)*, 31(11–16):1091–1100, 1999.

[4] R. Baudes and S. Zabele. RFC 1458: Requirements for multicast protocols, May 1993. Status: INFORMATIONAL.

[5] B. Bloom. Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, 13(7):422–426, July 1970.

[6] A. De Bonis and B. Masuci. An information thoeretical approach to metering schemes. *Proccedings of ISIT 2000*, 2001.

[7] A. De Bonis C. Blundo and B. Masuci. Bounds and constructions for metering schemes. *To appear in Communications in Information and Systems 2002*, 2002.

[8] D. Chaum. Elections with unconditionally-secret ballots and disruption equivalent to breaking RSA. *Advances in Cryptology Eurocrypt '88*, 330:177–182, 1988.

[9] Steve Coffey. Internet audience measurement: A practitioner's view. *Journal of Interactive Advertising*, 2001.

[10] comScore Networks. net-score product description. http://www.comscore.com/.

[11] IDzap. Company web page. http://www.idzap.com.

[12] Ipsos-RSL Broadcast Division. An introduction to the ipsos-rsl broadcast division. http://www.rslmedia.co.uk/broadcast/experience.html.

[13] T. Dyck. Yahoo chief scientist describes web attacks. *EWeek*, July 2002.

[14] FAST/ARF. http://www.fastinfo.org/measurement/pages/index.cgi/audiencemeasurement.

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

[16] Shafi Goldwasser and Mihir Bellare. Lecture notes on cryptography. Summer Course "Cryptography and Computer Security" at MIT, 1996–1999, 1999.

[17] Catherine Greenman. Royalty fees threaten web stations. *New York Times*, April 2002.

[18] A. P. Kamath, R. Motwani, K. Palem, and P. Spirakis. Tail bounds for occupancy and the satisfiability threshold conjecture. *Random Structures and Algorithms*, 7:59–80, 1995.

[19] Markus G. Kuhn. Probabilistic counting of large digital signature collections, 2000.

[20] B. Masuci and D. R. Stinson. Efficient metering schemes with pricing. *IEEE Transactions on Information Theory*, 47:2835–2844, 2001.

[21] M. Mitzenmacher. Compressed bloom filters. *ACM Symposium on Principles of Distributed Computing*, pages 144–150, 2001.

[22] R. Motwani and P. Raghavan. Randomized algorithms, Cambridge University Press, 2000.

[23] M. Naor and B. Pinkas. Secure and efficient metering. *Advances in Cryptology – Eurocrypt '98*, 1403:576–589, 1998.

[24] W. Ogata and K. Kurosawa. Provably secure metering scheme. *Advances in Cryptology – Asiacrypt '00*, 1976:388–398, 2000.

[25] B. Schoenmakers R. Cramer, M. Franklin and M. Yung. Multi-authority secret ballot elections with linear work. In *Advances in Cryptology Eurocrypt '96*, 1996.

[26] M. Reiter, V. Anupam, and A. Mayer. Detecting hit shaving in click-through payment schemes. In *Proceedings of the 3rd USENIX Workshop on Electronic Commerce*, pages 155–166, 1998.

[27] Nielsen Media Research. Audience measurement services – the global leader for actionable internet information.
http://www.nielsen-netratings.com/marketing/advertising/audience_measurement/

[28] B. Preneel V. Nikov, S. Nikova and J. Vandewalle. Applying general access structure to metering schemes, 2002.

[29] The Cloak. Company web page. http://www.the-cloak.com.