

---

## Deflation-secure web metering<sup>1</sup>

---

**Rob Johnson**

Computer Science Department,  
Stony Brook University,  
Stony Brook, NY 11794-4400  
E-mail: rtjohnso@cs.sunysb.edu

**Jessica Staddon\***

Palo Alto Research Centre,  
Palo Alto, CA 94304  
E-mail: staddon@parc.com  
\*Corresponding author

**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 (CDs) now have an incentive to underreport the size of their audience. Existing metering protocols protect only against the inflation of audience size. We introduce the first protocols for audience metering that protect against deflation attempts by CDs. The protocols trade-off the amount of additional information the CDs must distribute to facilitate metering with the amount of infrastructure required and are applicable to internet radio, web plagiarism, and software license enforcement.

**Keywords:** Bloom filters; broadcast encryption; metering.

**Reference** to this paper should be made as follows: Johnson, R. and Staddon, J. (xxxx) 'Deflation-secure web metering', *Int. J. Information and Computer Security*, Vol. x, No. x, pp.xx-xx.

**Biographical notes:** Rob Johnson is a PhD candidate at the University of California, Berkeley, and an Assistant Professor at the State University of New York, Stony Brook. His research interests span all areas of computer security, including secure system design, software security, network security, cryptography and cryptanalysis. He has published several articles in these areas. He was an intern at PARC when some of this work was done. Previously he was a PhD student in Mathematics.

Jessica Staddon is a Research Scientist and Manager at the Palo Alto Research Centre. She specialises in applied cryptography and is particularly interested in digital rights management and privacy. She serves on the editorial boards of the *Int. J. Information and Computer Security* and the *J. Computer Security*. Jessica received her PhD in Mathematics from the University of California, Berkeley, in 1997.

<sup>1</sup>Some of the results of this paper appeared in the 2002 ACM Workshop on Digital Rights Management.

## 1 Introduction

Internet Content Distributors (CDs) 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 CDs have an incentive to inflate these numbers. Franklin and Malkhi (1998) introduced the inflation-secure metering schemes to prevent this type of fraud, and many variations and improvements have subsequently been proposed (e.g. Naor and Pinkas, 1998; Ogata and Kurosawa, 2000; Masucci and Stinson, 2001). The most secure of these protocols are token-based, meaning clients possess special tokens that are hard for the CD to generate. When clients request a 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 CD cannot generate the tokens itself, it is unable to inflate its client count.

With the advent of per-listener royalty fees for internet radio (Greenman, 2002) and the growth of web content plagiarism (Dyck, 2002), CDs now have an incentive to report artificially small audiences, but no existing protocol prevents such behaviour 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 the client tokens to lower its client count.

We present three new client-metering protocols that prevent CDs from reporting artificially low audience sizes. The protocols leverage the relative anonymity of the internet to enable an auditor to monitor the CD’s interactions with the clients. Essentially, the auditor poses as a client in his interactions with the CD. Since the CD cannot distinguish the auditor from any other client, it cannot disregard the metering 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 are not deflation-secure, we use other techniques to decrease the communication overhead. 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 CD simply maintains a Bloom filter (Bloom, 1970) that encodes the client IDs (anonymised 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 anonymised 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 (TP) randomly allocates a subset of a global set of keys to each client. The CD 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 CD 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 (i.e. other than the encrypted content) on the part of the CD, but does not completely preserve the privacy of the clients.

The third protocol (see Section 5) again uses encryption to offer protection against both inflation and deflation, and requires essentially no additional communication

between the auditor and the CD. The encryption keys are derived from polynomials, and the size of the ciphertexts is correlated with the degree of the polynomial used. As the number of clients grows, the CD is forced to use polynomials of increasingly higher degree to encrypt the content; hence the audience size can be inferred from the size of the ciphertexts. This protocol offers more privacy to the clients and only uses constant storage by the clients. Table 1 summarises the main features of our protocols.

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

<i>Scheme</i>	<i>Protocol 1</i>	<i>Protocol 2</i>	<i>Protocol 3</i>
Deflation protection	Yes	Yes	Yes
Inflation protection	No	Yes	Yes
Privacy preserving	Yes	No	Partially
Communication overhead	$O(n)$	$O(1)$	Varies
Client storage	$O(1)$	Varies	$O(1)$
Counts cumulative audience	Yes	Yes	Yes
Counts current audience	Yes	No	No

Finally, we observe that deflation security essentially requires the CD 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 CDs 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 technologies (see, for example, Shields and Levine, 2000) may support perfect anonymity and unlinkability in the near future. Thus, we analyse our protocols in the context of perfect anonymity, and believe that they degrade gracefully in the current internet.

This paper is organised as follows. We discuss related work in Section 1.1. Our model is described in Section 2. We present and analyse an easily implemented deflation-secure protocol in Section 3, a deflation-secure protocol with constant overhead in Section 4 and a deflation-secure protocol with constant client storage in Section 5. We conclude in Section 6 and discuss open problems. In Appendix A, we provide a Bayesian analysis of the protocol of Section 3.

### 1.1 Related work

One of the first methods for counting the number of visitors to a website is developed by Franklin and Malkhi (1998). Naor and Pinkas (1998) present a protocol with stronger security guarantees (Franklin and Malkhi, 1998). Ogata and Kurosawa (2000) identify flaws in the Naor and Pinkas scheme and propose their own. The Naor–Pinkas model has been generalised and analysed extensively (De Bonis and Masucci, 2000; Masucci and Stinson, 2001; De Bonis, Blundo and Masucci, 2002; Nikov et al., 2003). In a similar vein, Kuhn (2000) 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-mentioned papers. The simplest audience measurement technique counts the number of entries in the server’s log files (Coffey, 2001). Since it is easy for the server administrator to delete or 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 (FAST/ARF). In this arrangement, the ad agency can underreport the number of ads it serves, thus lowering the advertising fees it pays. Reiter, Anupam and Mayer (1998) propose a scheme for detecting this sort of fraud. Conversely, Anupam, Mayer, Nissam, Pinkas and Reiter (1999) 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 ‘Inc. MeasureCast, Nielsen Media Research’. These numbers can be fairly accurate, but this method is expensive. Some audience measurement services combine log analysis and consumer surveys ‘Inc. MeasureCast, comScore Networks’. 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 misreporting ‘IPSOS-RSL Broadcast Division, Nielsen Media Research’.

All the audience measurement techniques described earlier are designed for determining the advertising rates and thus are concerned only about the attempts made by the CD to inflate the audience size. In all the schemes above except the survey and diary methods, the CD can easily deflate the size of the audience. In the context of advertising, CDs 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 CD from deflating the audience size.

Finally, we note that secure voting (see, for example, Chaum, 1988; Schoenmakers et al., 1996) 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 do not believe that these techniques are directly applicable to the CD setting.

## 2 Model

Our protocols involve a CD, clients, and in the case of the protocol of Section 4, a 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 website ‘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 Properties

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 maximise the gain in accuracy that comes with each decrease in efficiency.

### 2.1.1 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 influenced 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 these sections by increasing the amount of information in the encodings.

### 2.1.2 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 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 is on the order of the number of clients. Our second protocol has constant overhead but requires significant client storage and infrastructure. Finally, our third protocol requires infrastructure and may have significant communication overhead, but it requires only constant client storage.

### 2.1.3 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. If an auditor mimics typical client behaviour (e.g. in its content-request patterns), unlinkability is not required.

## 2.2 *The challenge of web metering*

Cryptographic web-metering protocols typically involve distributing the secret information amongst the clients. When clients request content from the CD, they provide the CD with some secret information. The CD's actions reflect the amount of secret information the CD knows, and hence the number of clients. Because of this structure, web-metering protocols are inherently vulnerable to deflation attacks; the CD simply ignores additional secret information and colludes with clients to ensure a content distribution channel through other means. In addition, in many of the applications that motivate our work (e.g. internet radio) clients have a strong incentive to engage in such collusion to reduce the CD's apparent audience size. Finally, a web-metering protocol is able to *estimate* the audience size based only on the CD's actions. To meet all these challenges we require an anonymous communication channel (making collusion more difficult) and we obtain bounds on the probability of a CD's actions as a function of the number of clients. Nevertheless, we cannot claim that our protocols, or indeed, *any* web-metering protocol, is a complete solution. In the following we discuss the challenges in more detail.

### 2.2.1 *Collusion between CDs and clients*

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 later. Thus, the auditor can be sure that the radio station has only a few listeners *on that port*. Our protocols cannot prevent the radio station from broadcasting to a large number of users from a second port that is spread by word-of-mouth among faithful fans.<sup>1</sup>

Hence, to protect against client and CD collusion, a reliable scheme for tracking all relevant CDs is needed. This is a hard problem that we do not address in this paper. We do note, however, that in order for the attack described earlier 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 imposes a natural limit on the scale of this sort of cheating.

### 2.2.2 *Inflation attacks*

The client anonymity we require can also be used against the CD. The auditor (or any other client) can potentially inflate the audience size artificially by repeatedly requesting the content as a new client. Our protocols do not explicitly protect against this. One possible remedy is to insert a TP between the distributor and the clients, with anonymous communication only between the TP and the CD. If the CD suspects this attack is underway, the TP's logs can be examined. Of course, requiring a TP 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 protocols in Sections 4 and 5), then this approach is worth considering.

### 2.2.3 *Bayes' theorem and web-metering*

At a high level, all web-metering schemes work in the same way: they allow an auditor to use an observed event to infer audience size. More precisely, the auditor has information about the probability of the observed event, given a conjectured audience size. If the

event has low probability for audience sizes greater than a value  $N$ , the auditor may use this as indirect evidence of an audience size less than  $N$ . Audience size can be estimated only in this manner because of Bayes' theorem (see, e.g. Motwani and Raghavan, 2000), which can be expressed as:

$$P(\text{event} / n \text{ clients}) = \frac{P(n \text{ clients} / \text{event})P(\text{event})}{P(n \text{ clients})}$$

where the quantity the auditor knows is  $P(\text{event} | n \text{ clients})$ . Hence, in order to bound the quantity that *directly* infers the audience size,  $P(n \text{ clients} | \text{event})$ , some information on the distribution of clients is necessary. In general, we do not believe the auditor is likely to have such information, so our protocols primarily rely on using the auditor's knowledge of the distribution of  $(\text{event} | n)$  to estimate the audience size; however, in Appendix A, we demonstrate how such additional information can improve the analysis of the protocol in Section 3.

### 2.3 Applications

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

#### 2.3.1 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; for example, 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 organisation such as the RIAA.

#### 2.3.2 Screen-scraping

Websites that provide a useful service, such as Yahoo's real-time stock prices, often get 'screen-scraped' by other web services (Dyck, 2002). 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 behaviour 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.

#### 2.3.3 Distribution of licensed content

Consider a website 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.

### 2.3.4 Web advertising

As described in Section 1.1, some web advertisers serve their ads directly, and hence can underreport 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.

## 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).

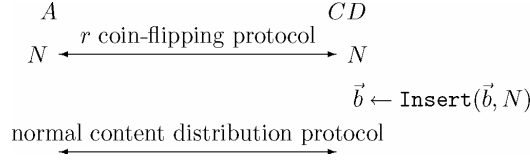
The protocol uses Bloom filters (Bloom, 1970), so we give a brief introduction to them here. A Bloom filter is a lossy representation of a set and consists of a bit-vector  $\vec{b}$  of length  $m$  and  $s$  independent hash functions  $h_1, \dots, h_s : \{0, 1\}^* \rightarrow \mathbb{N}$ .<sup>2</sup> 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  $\vec{b}[h_1(x) \bmod m] = \dots = \vec{b}[h_s(x) \bmod m] = 1$  (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  $\vec{b}[h_1(x) \bmod m] = \dots = \vec{b}[h_s(x) \bmod m] = 1$ . Note that this test can lead to false positives; this is why the Bloom filter is termed ‘lossy’. If  $\vec{b}[h_i(x) \bmod m] = 0$  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  $1 - (1 - 1/m)^{ns}$ . Hence, the expected weight of a Bloom filter given  $n$  insertions is  $m(1 - (1 - 1/m)^{ns})$ . One can derive an estimate of the actual number of insertions by setting  $m(1 - (1 - 1/m)^{ns}) = w(\vec{b})$  and solving for  $n$ . This estimate of  $n$  is denoted  $I(\vec{b})$ , where  $I(\vec{b}) = \ln(1 - w(\vec{b})/m) / s \ln(1 - 1/m)$ . In Appendix A, we use Bayesian analysis to discuss the accuracy of this estimate in detail; here we just note that to minimise the probability of a false positive,  $s$  should be chosen so that  $s = (\ln 2)m/n$ , which gives a false positive rate of  $(1/2)^{(\ln 2)m/n} \approx (0.6185)^{m/n}$ . 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 CD maintains a Bloom filter of width  $m = cn$ , where  $n$  is the average number of requests seen by the CD each week and  $c$  is a parameter agreed upon in advance. In practice,  $c = 8$  works well as discussed earlier. When a client sends a request to the CD, the CD and client engage in a coin-flipping protocol to agree on an  $r$  bit nonce  $N$  and the CD inserts  $N$  into the Bloom filter. Any standard coin flipping protocol will work (see, for example, Goldwasser and Bellare, 1999). They then proceed with their normal protocols. Each week, for example, the CD

sends the Bloom filter to the auditor and then starts again with a fresh filter. The auditor checks that the Bloom filter it receives,  $\vec{b}$ , contains any nonces it has negotiated with the CD. For example, if the auditor has sent  $k$  requests to the CD, the auditor checks that all their nonces,  $N_1, \dots, N_k$ , are present in the Bloom filter that the CD submits for that interval. Provided these conditions are satisfied, the auditor computes an upper bounds on the number of requests,  $n$ , such that the probability of a Bloom filter of weight  $w(\vec{b})$  given  $n$  requests is less than  $\epsilon$ , for some  $0 < \epsilon < 1$ .

**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



Note that because the CD's storage is just  $cn$ , this protocol requires less storage and less communication overhead than a protocol in which the CD stores and sends *all* the nonces. Such a protocol incurs  $O(n \ln n)$  storage and communication overhead, whereas our protocol incurs  $O(n)$  storage and communication overhead. In addition, for small CDs, this scheme is very efficient. Using the ratio  $m/n = 8$  mentioned earlier, the CD must send the auditor about 1 byte per join. So, for example, a CD that receives 20 requests each day would have to send only a 140-byte message to the auditor each week. Thus this scheme is completely feasible for small to medium CDs. Even a relatively large CD with around 150 requests per day would have to send only a 1-kB 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 2 kB sec<sup>-1</sup>.

### 3.1 Analysis

The following lemma gives an upper bound the auditor may use to estimate the audience size when a filter of weight  $w(\vec{b})$  is received.

**Lemma 1:** *If  $n < \frac{\ln(1 - \frac{\epsilon t}{m})}{s \ln(1 - 1/m)}$  then  $P(w(\vec{b}) \geq t | n) < \epsilon$*

*Proof.* Recall that  $E(w(\vec{b}) | n) = [1 - (1 - 1/m)^{ns}]m$ . By Markov's inequality (see, for example, Motwani and Raghavan, 2000):

$$P(w(\vec{b}) \geq t | n) \leq \frac{E(w(\vec{b}) | n)}{t} = \frac{[1 - (1 - 1/m)^{ns}]m}{t}$$

Setting this quantity to be less than  $\epsilon$  and solving for  $n$  yields the statement of the lemma. □

The bound given by this lemma is a fairly coarse one. A more refined analysis is possible using Bayesian analysis; however, since this involves making the controversial assumption that the number of clients is uniformly distributed, we leave this analysis to Appendix A.

In general, the CD 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  $\vec{b}'[h_i(N_j)] = 0$ . Proposition 1 describes the CD's optimal strategy and bounds his chances of success.

**Proposition 1:** *Suppose the CD receives  $n$  requests, but wishes to report only  $L < n$  of those requests. Let  $\{J_1, \dots, J_n\}$  be the set of nonces generated by servicing the requests, and  $\vec{b}$  be the Bloom filter generated from  $\{J_1, \dots, J_n\}$ . Then the CD's optimal strategy is to report a Bloom filter  $\vec{b}'$  containing the largest subset  $S \subseteq \{J_1, \dots, J_n\}$  such that  $I(w(\vec{b}')) \leq L$ . If  $w(\vec{b}) - w(\vec{b}') = D$  and the auditor sent  $k$  requests to the CD, then*

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

*Proof.* The CD 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 CD's chances of being caught. Hence the CD's optimal strategy is to report a Bloom filter  $\vec{b}'$  containing some subset  $S \subseteq \{J_1, \dots, J_n\}$ .

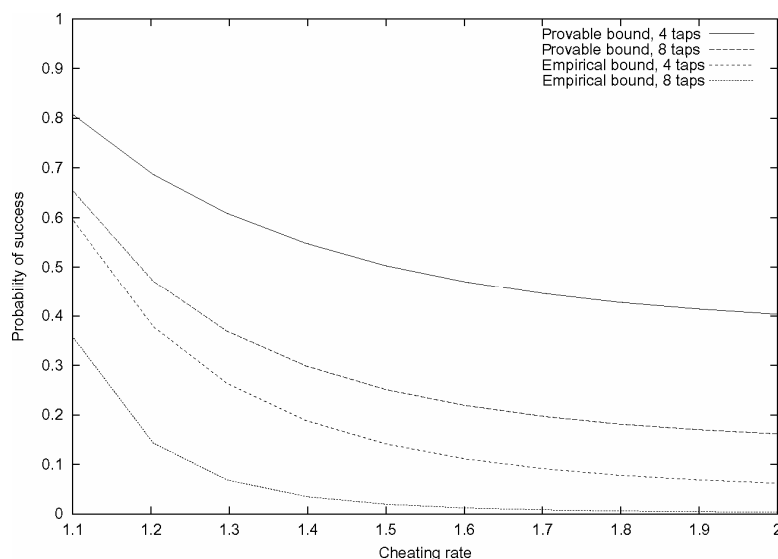
To decrease the weight of the Bloom filter by  $D$  one must remove at least  $D/s$  items, since each item can decrease the weight of the filter by at most  $s$ . Since the CD cannot distinguish the auditor's requests, his best strategy is to select the largest  $S$  such that  $w(\vec{B}[S])$  is below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

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 CD trying to cheat by finding the optimal subset  $S$  described in the above proposition. Based on our experiments, the CD has to remove at least  $D/2$  items from  $\{J_1, \dots, J_n\}$  in order to decrease the weight of his Bloom filter by  $D$ . Figure 2 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 CD 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

CD, artificially inflating the audience size. Since this scheme provides total listener anonymity, the CD cannot identify the attacker. Also, a CD 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 CD's audience.

**Figure 2** The probability that a CD can fool the auditor, assuming  $m = 1024$ ,  $s = 5$ , and the CD 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 CD cannot fool the auditor with greater probability than these curves indicate. So, for example, if a CD receives  $1.3 \times 128$  requests, and the auditor sent eight auditing requests, then the CD's chances of successfully convincing the auditor that he received only 128 requests is less than 10%.



We have described this scheme in terms of request-counting, but it can also be used to count the current audience. Suppose the auditor wants to know the current audience size at each minute. Then the CD 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 the content from the CD 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 further be improved by using compressed Bloom filters (Mitzenmacher, 2001) 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 website that provides software or movies in encrypted form available for download and decryption with payment. When used with real-time content, the CD must be using the network as a broadcast channel in order for assuring 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 TP. In the initial phase of the protocol, the TP sends all the keys to the CD. 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 the clients probabilistically, the auditor (who may be the same as the TP) when requesting the content anonymously<sup>3</sup> (e.g. by visiting the distributor's website), can infer the audience size from the encryption key in use.

The TP assigns keys to the clients as follows. First, the entire set of keys is partitioned into  $t$  sets,  $S_1, \dots, S_t$ . 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 CD does not have any 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 3 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.

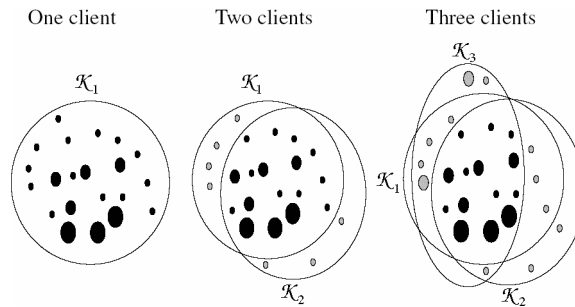
##### 4.1 Basic protocol

This protocol takes a positive integer  $m$  representing the number of keys in the system, a positive integer  $t$  and positive integer  $s_1, \dots, s_t$  such that  $s_1 + s_2 + \dots + s_t = m$  as inputs. The keys are partitioned into  $t$  sets,  $S_1, \dots, S_t$ , such that for each  $i$ ,  $|S_i| = s_i$ , where  $s_1 > s_2 > \dots > s_t$ . For each  $i = 1, \dots, t$  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  $p_1 < p_2 < \dots < p_t$ . Numbers  $\epsilon_1, \epsilon_2, 0 < \epsilon_1, \epsilon_2 < 1$ , are also input to provide a gauge of the accuracy of the audience measurements. These parameters imply an upper bound,  $n_{\max}$ ,

on 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,  $k_1, \dots, k_m$ , and sends them to the CD.
- 2 Upon contacting the CD, a client,  $u_i$ , receives a set of keys  $K_i \subseteq \{k_1, \dots, k_m\}$  from the TP. For  $j = 1, \dots, m$ ,  $k_j \in K_i$  with probability  $pr$  if  $k_j \in Sr$ . The TP sends the CD the ID numbers of the client's keys.<sup>4</sup>
- 3 To distribute content to clients  $u_{j1}, \dots, u_{jr}$ , the CD chooses a key  $k \in T = K_{j1} \cap \dots \cap K_{jr}$  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 the content and notes the key,  $k$ , that the CD is using in Step 3. There exists  $i \in \{1, \dots, t\}$  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  $n$ ,  $\left[ \frac{|T \cap S_i|}{|T|} \mid n \right]$ , to be within a confidence level of  $1 - \epsilon_1$ . Using this distribution, the auditor determines a range  $[n_1, n_2]$  such that, for each  $n \in [n_1, n_2]$ ,  $P(k \in S_i \mid n) \geq \epsilon_2$ , and estimates 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).
  - If the auditor has contacted the CD 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 CD is not following the protocol.

**Figure 3** The black ovals represent keys in the set  $T$  when there are one, two and three clients. That is,  $K_i$  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 is selected from  $T$  reflects the audience size



This protocol relies on the CD's inability to distinguish between the keys in the intersection  $T$ . The CD 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  $S_i$  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 CD 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 CD 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 CD 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 CD has to use a key that is not known to several clients, in which case the distributor may indeed be able to reach only a small audience.

Finally, the CD can potentially benefit from collusion with clients or other CDs. If the TP is using the same global set to allocate keys to clients of different CDs (which is a desirable practice because it can allow clients to ‘surf’ multiple distributors without needing to repeat the initialisation phase), then the distributors (and users) may be able to distinguish between keys that they would not have been able to use otherwise. However, as mentioned earlier, this may only be 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.

## 4.2 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  $[(|T \cap S_i| / |T|) | n]$ . Let

$$\beta_{x,i} = s_1 p_1^x + \dots + s_{i-1} p_{i-1}^x + s_i + 1 p_{i+1}^x + \dots + s_t p_t^x.$$

**Lemma 2:** Let  $0 < \delta < 1$ . For  $i = 1, \dots, t$  and  $n = x$ ,  $P(k \in S_i | n = x)$  is at least as large as

$$\left\{ [(1-\delta)s_i p_i^x] / [(1+\delta)\beta_{x,i} + (1-\delta)s_i p_i^x] \right\} \text{ and at most as large as}$$

$$\left\{ [(1+\delta)s_i p_i^x] / [(1-\delta)\beta_{x,i} + (1+\delta)s_i p_i^x] \right\} \text{ with probability at least } 1-\epsilon_1, \text{ when}$$

$$\left[ e^\delta / (1+\delta)^{1+\delta} \right] s_i p_i^{n_{\max}} \leq \left\{ [1 - (1-\epsilon_1)^{1/t}] / 2 \right\} \text{ and } \leq e^{-\delta^2 s_i p_i^{n_{\max}} / 2} \leq \left\{ [1 - (1-\epsilon_1)^{1/t}] / 2 \right\}.$$

*Proof.* For  $i = 1, \dots, t$ , 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, Motwani and Raghavan, 2000), 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  $\left[ \frac{e^\delta}{(1 + \delta)^{1+\delta}} \right] s_i p_i^{n_{\max}} \leq \left[ \frac{e^\delta}{(1 + \delta)^{1+\delta}} \right] s_i p_i^{n_{\max}} \leq \left\{ [1 - (1 - \epsilon_1)^{1/t}] / 2 \right\}$   
and  $e^{-\delta^2 s_i p_i^{n_{\max}}} \leq e^{-\delta^2 s_i p_i^{n_{\max}} / 2} \leq \left\{ [1 - (1 - \epsilon_1)^{1/t}] / 2 \right\}$  Hence,  $P(k \in S_i | n = x) =$   
 $|T \cap S_i| / |T| = (|T \cap S_i|) / (|T \cap S_1| + \dots + |T \cap S_t|)$  is in the interval stated in the lemma  
with probability at least.  $\left\{ (1 - 2)[1 - (1 - \epsilon_1)^{1/t}] / 2 \right\}^t = 1 - \epsilon_1$ .  $\square$

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 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 1:** *To satisfy step 4 of the basic protocol, it suffices (but is not necessary) to choose  $n_{\max} \leq \ln(c(\epsilon_1, \delta, t) / s_i) / \ln p_i$  and  $s_i \geq [c_i(\epsilon_1, \delta)] / p_i^{n_{\max}}$  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  $\sum_{i=1}^t \frac{[c_i(\epsilon_1, \delta)]}{p_i^{n_{\max} - 1}}$ .*

*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 2:  $\left[ \frac{e^\delta}{(1 + \delta)^{1+\delta}} \right] s_i p_i^{n_{\max}} \leq \frac{[1 - (1 - \epsilon_1)^{1/t}]}{2}$  and

$e^{-\delta^2 s_i p_i^{n_{\max}}} / 2 \leq \frac{[1 - (1 - \epsilon_1)^{1/t}]}{2}$  It follows that  $c_i(\epsilon_1, \delta) =$   
 $\max \left\{ \left[ 2 \ln \left( \frac{1 - (1 - \epsilon_1)^{1/t}}{2} \right) \right] / -\delta^2, \left[ \ln \left( \frac{1 - (1 - \epsilon_1)^{1/t}}{2} \right) / \ln \left( \frac{e^\delta}{(1 + \delta)^{1+\delta}} \right) \right] \right\}$ .

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

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

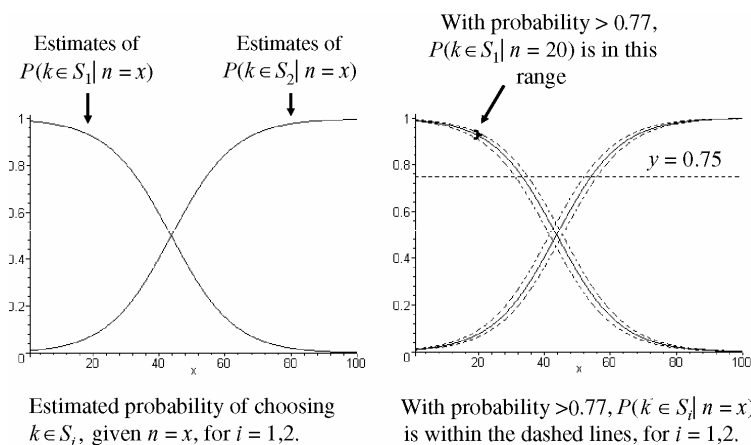
For illustrative purposes,<sup>5</sup> we conclude this section with a small example.

#### 4.2.1 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  $s_1 = 37,000$ ,  $p_1 = 0.6$ ,  $s_2 = 370$ ,  $p_2 = 1$  and  $n_{\max} = 13$ . Because  $|T \cap S_2| = 370$  with probability 1, we need find only a confidence interval for  $|T \cap S_1|$  and this will imply confidence intervals for  $|T \cap S_1|$  and  $|T \cap S_2| / |T|$ . Setting  $\delta = 0.2$ , by the proof of Lemma 3 we need the following inequality

to hold:  $(0.98)^{s_1 p_1^{13}} < \frac{\epsilon_1}{2}$ . Solving for  $\epsilon_1$  yields,  $\epsilon_1 \geq 0.75$ . If we choose  $\epsilon_2 = 0.75$ , then with at least 0.75 confidence, it follows by solving the inequality,  $[(1-\delta)37000(.6)^x]/[(1-\delta)37000(.6)^x + 370] \geq 0.75$  for  $x$ , that  $P(k \in S_1 | n \leq 6) \geq 0.75$ . Similarly,  $370/[(1+\delta)37000(.6)^x + 370] \geq 0.75$ . 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 4.<sup>6</sup>

**Figure 4** 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  $p_1 = 0.6$ ,  $p_2 = 1$ ,  $s_1 = 37000$ ,  $s_2 = 370$ ) 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 0.75.



In this example, we expect a client to store 22,570 keys. If the keys are each 64-bits long, this represents 0.17 megabytes of keying material. While this is significant, it is a fraction of the space required by most media players (for example, it is about 0.09 of the download size of WinAmp.com’s ‘full’ player). Viewed differently, after listening to streaming music at a data rate of 28.8 kilobytes per second for less than 20 min, the keying material is less than 0.0425 of the audio data that has been downloaded.

Since a client will typically have more than half of the 37,370 keys in this example, the TP can tell the CD the keys the client *does* not 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.

### 4.3 Extensions

#### 4.3.1 Multiple content distributors

The basic protocol is easily modified to allow the TP to use a single set of keys for multiple CDs. 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.

### 4.3.2 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 receive the same number of keys in  $S_i$  with high probability, the particular keys they tend to receive may partly be a function of their sex. Hence, the auditor may be able to infer the predominant sex of the audience from the CD's choice of encryption key in  $S_i$ .

### 4.3.3 Measuring the current audience

The protocol described earlier is best suited to estimate the *cumulative* audience size, for example, the number of hits received by a website 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 CD typically is informed only of the new additions to the multicast group and is unlikely to know when a member leaves (Baudes and Zabele, 1993). Hence, by observing the CD's behaviour, or by querying directly, it may be possible to learn only the cumulative audience. In this case, behavioural patterns may be used to infer current audience size from cumulative data.

It may also be possible to modify the basic protocol to measure the audience size directly. The key idea is that if the auditor can observe the content for long enough<sup>7</sup> 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 CD gains some ability to distinguish the 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 CD 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 CD may seek to mislead the auditor by using only  $k'$ . However, if the CD is required to change the 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 CD 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 Estimating audience size with constant client storage

In the following protocol, the auditor is able to infer the audience size from the size of a message that is sent with the (encrypted) content. The message enables the clients to recover the content encryption key and, subsequently, the content. As with the protocol of Section 4, the protocol requires a keying infrastructure, but we use polynomial interpolation to reduce the client storage requirements.

The main idea of the protocol is to force a CD to send increasingly longer messages with increases in the number of clients while allowing less communication overhead than with unicast (i.e.  $O(n)$ ) communication. We achieve this by providing each client with a point on one of the several polynomials. The actual polynomials vary according to their degree and the clients are more likely to receive a point on a polynomial of low degree than one of high degree. The size of the message that is sent by the CD is correlated with the degree of the smallest polynomial for which all of the clients know a point. Hence, when the number of clients is small, the CD is more likely to be able to send a small message than when the number of clients is high. The CD cannot artificially reduce the size of the message and thus mislead the auditor, without distributing extra keys to clients, and this is a risky practice because, as in the previous protocols, the auditor often poses anonymously as a client. This protocol is most useful when it suffices to prove that audience size is below a certain bound.

Before stating the protocol we introduce a piece of notation. Let  $p(x) = a_0 + a_1x + \dots + a_tx^t$  be a polynomial in  $F_q[x]$  for some field  $F_q$  with  $q$  elements and generator  $g \in F_q$ . We use the notation,  $g^{p(x)}$  to denote the  $(t+1)$ -tuple,  $(g^{a_0}, \dots, g^{a_t})$ .

### 5.1 Basic protocol

This protocol takes positive integers,  $T, t_1, \dots, t_T$  as inputs as well as probabilities,  $p_1 > p_2 > \dots > p_T$ , and  $\epsilon$ .

- 1 For  $i = 1, \dots, T$ , the TP randomly chooses  $t_i$  pairs  $(a_{i1}, b_{i1}), \dots, (a_{it_i}, b_{it_i}) \in F_q \times F_q$ . Also, the TP randomly generates polynomials,  $f_1(x), \dots, f_T(x) \in F_q[x]$  such that for  $i = 1, \dots, T$ , the degree of  $f_i(x)$  is  $t_1 + \dots + t_i$  and for all  $j \leq i$  and  $r \leq t_i$ ,  $f_i(a_{jr}) = b_{jr}$ . Hence, a point  $(a_{jr}, b_{jr})$  is on polynomials  $f_j(\cdot), \dots, f_T(\cdot)$ . The TP gives these polynomials to the CD.
- 2 A client,  $u_j$ , receives a point  $(a_j, b_j) \in F_q \times F_q$  from the TP, where for  $i = 1, \dots, T$ ,  $(a_j, b_j) \in \{(a_{i1}, b_{i1}), \dots, (a_{it_i}, b_{it_i})\}$  with probability  $p_i$  and an index<sup>8</sup>  $v_j$  such that  $(a_j, b_j)$  is a point on a polynomial  $f_{v_j}(x)$ .<sup>9</sup> When requesting content,  $u_j$  sends  $v_j$  to the CD.
- 3 To distribute content to the clients  $u_j, \dots, u_{j_r}$ , the CD chooses a minimum index  $j \in \{1, \dots, T\}$  such that each client knows a point on the polynomial,  $f_j(x)$ , random values  $r, s \in \{1, \dots, q\}$  and encrypts the content encryption key,  $g^s$ , as the broadcast  $B = \{g^{rf_j(x)+s}, g^r\}$ . A fresh  $s$  should be chosen regularly. (The frequency with which this is done provides a way to tune the accuracy of the protocol.)
- 4 When a fresh key is announced, the auditor can request the content and observe the size of the broadcast,  $wq$ . Since  $w = \text{degree}(f_j(x)) + 2$ , the auditor can recover the degree of the polynomial used and from that determine a value  $n_1$  such that for  $n > n_1$ ,  $P(|B| = w \log q | n = n_1) < \epsilon$ , and estimate the audience size as being less than  $n_1$ .

In the final step of the protocol, the auditor determines a lower bound on  $n$  such that the observed broadcast size is very unlikely for a client count exceeding the lower bound. The following lemma provides such a lower bound in statement 1. Estimating audience size in this way results in an overestimate; however, the second claim suffices to bound

the error of this estimate. Claims 3 and 4 provide alternate ways of estimating the audience size. Relying on these last two claims may not be as satisfying to the auditor since depending on the parameter values,  $P(|B| = b \log q | n = x)$  may never be close to 1 (see the example at the end of this section).

**Lemma 3** Let  $b = t_1 + \dots + t_w + 2$ , for some integer  $w \in \{1, \dots, T\}$ ,  $0 < \epsilon < 1$  and  $n > 1$ . The following bounds can be used to estimate the number of clients of the CD given a broadcast size of  $b \log q$ .

- 1 If  $x > \log(\epsilon) / \log(1 - p_{w+1} - \dots - p_T)$ , then  $P(|B| = (b \log q | n = x) < \epsilon$ .
- 2 If  $P(|B| = (b \log q | n = x) < \epsilon$ , then  $x > \log \left[ \epsilon \left( \frac{p_1 + \dots + p_w}{p_w} \right) \right] / \log(1 - p_{w+1} - \dots - p_T)$ .
- 3 If  $x < \log \left[ (1 - \epsilon) \left( \frac{p_1 + \dots + p_w}{p_w} \right) \right] / \log(1 - p_{w+1} - \dots - p_T)$ , then  $P(|B| = (b \log q | n = x \text{ clients}) > 1 - \epsilon$
- 4 If  $P(|B| = (b \log q | n = x) < 1 - \epsilon$ , then  $x < \log(1 - \epsilon) / \log(1 - p_{w+1} - \dots - p_T)$ .

*Proof.* To see the first claim note that the broadcast size is  $b \log q$  if and only if none of the clients have points on  $f_{w+1}(\cdot), \dots, f_T(\cdot)$  and at least one client has a point on  $f_r(\cdot)$ . Hence,  $P(|B| = b \log q | n) = P(\exists j \in [n] \text{ s.t. } f_w(a_j) = b_j | \forall j \in [n], \forall r = w+1, \dots, T, f_r(a_j) \neq b_j; n) P(\forall r = w+1, \dots, T, f_r(a_j) \neq b_j | n) = \{1 - [1 - (p_w / (p_1 + \dots + p_w))]^n\} [1 - p_{w+1} - \dots - p_T]^n$ . The

expression in the lemma comes from noting that when  $n \geq 1$ :  $\{1 - [1 - (p_w / (p_1 + \dots + p_w))]^n\} [1 - p_{w+1} - \dots - p_T]^n < [1 - p_{w+1} - \dots - p_T]^n$  and solving for  $n$  in the following inequality:  $[1 - p_{w+1} - \dots - p_T]^n < \epsilon$ . The second claim follows from the fact that when  $n \geq 1$ :  $P(|B| = (b \log q | n \text{ clients}) = \{1 - [1 - (p_w / (p_1 + \dots + p_w))]^n\} [1 - p_{w+1} - \dots - p_T]^n > [p_w / (p_1 + \dots + p_w)] (1 - p_{w+1} - \dots - p_T)^n$ . Hence,  $P(|B| = (b \log q | n \text{ clients}) < \epsilon$  implies that  $[p_w / (p_1 + \dots + p_w)] (1 - p_{w+1} - \dots - p_T)^n < \epsilon$ , and solving for  $n$  yields the second statement of the lemma.

Claims 3 and 4 follow in similar fashion by reusing the above inequalities.  $\square$

Another important aspect of this protocol is communication overhead. The communication overhead depends on the number of clients,  $n_{\max}$ , that the protocol can detect, which in turn depends on the values  $\{p_i\}_{i=1}^T$  and  $\{t_i\}_{i=1}^T$ . The following corollary makes the relationship between the expected broadcast size and these parameters precise.

**Corollary 2:** The expected broadcast size given  $n = x$  is  $(t_1 + \dots + t_v + 2) \log q$ , where  $V = \sum_{w=1}^T w (1 - [1 - (p_w / (p_1 + \dots + p_w))]^x) (1 - p_{w+1} - \dots - p_T)^x$ .

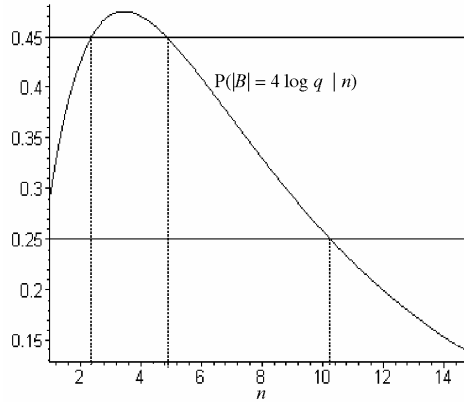
*Proof.* This follows from the fact that the probability polynomial  $f_w(\cdot)$  which will be used to compute the broadcast given  $n = x$  clients, is  $(1 - [1 - (p_w / (p_1 + \dots + p_w))]^x) (1 - p_{w+1} - \dots - p_T)^x$   $\square$

We conclude this section with a small example.

**Example:** Consider an instantiation of the protocol in which  $T = 3$  and  $p_1 = 0.5$ ,  $p_2 = 0.25$ , and  $p_3 = 0.125$ . In this instantiation, a single point on each polynomial is distributed and so the broadcasts are quite small (but with limited revocation capability as discussed later). If the auditor detects a broadcast of size  $4 \log q$ , then it can conclude that a polynomial of degree 4 is being used and can calculate  $P(|B| = 4 \log q | n) = [1 - (2/3)^n](7/8)^n$ . Figure 5 demonstrates that the probability of a broadcast of that size, given an audience of sizes 3 or 4, is more than 0.45, and the probability of a broadcast of that size given an audience of size 11 or higher, is less than 0.25.

Finally, we note that by choosing larger values of  $t_1, \dots, t_T$ , than in the above example, it may be possible to efficiently revoke clients from the system. The idea is that when there are several points on a polynomial to choose from, the likelihood that the good clients will share polynomial points with revoked clients is reduced, and thus fewer good clients need to be rekeyed when the polynomial is changed to exclude revoked users.

**Figure 5:** In this example,  $T = 3$ ,  $p_1 = 0.5$ ;  $p_2 = 0.25$ ;  $p_3 = 0.125$ ,  $b = 4 = t_1 + t_2 + 2$ .  $P(|B| = 4 \log q | n) < 0.25$  when  $n > 10$ , and  $P(|B| = 4 \log q | n) > 0.45$  when  $n \in \{3, 4\}$ .



$n$	Expected Broadcast size (rounded to nearest integer)
1	$2 \log q$
2	$3 \log q$
3	$4 \log q$
4	$4 \log q$
5	$4 \log q$
6	$4 \log q$
$n \geq 7$	$5 \log q$

## 6 Conclusion and directions for future research

We have introduced the first metering protocols that provide security against deflation attempts by the CD. Our protocols add a modest amount of additional interaction between the clients and the CD to facilitate the measurement of an accurate lower bound on the

number of clients. The first protocol needs essentially no infrastructure and provides privacy-preserving deflation security at a cost of  $O(n)$  overhead, for  $n$  clients. The final two protocols achieve improved overhead and inflation protection, but at the cost of increased infrastructure and reduced client privacy.

The approach we have taken to achieving deflation security involves a particular new technique: anonymous monitoring of the CD. Indeed, we believe that deflation security *requires* different techniques than those used for inflation-secure metering due to the need to force CDs to ‘commit’ in some way to client counts. Thus, other approaches may also be possible. We end with a discussion of two directions in which this research may be taken.

### 6.1 Open problems

We highlight two problem areas to explore. The first is deflation-secure web-metering under weaker assumptions. Our protocols require that an auditor be able to anonymously request the content from the CD. Although we are optimistic that this will be easy to achieve in the future (and it is already achievable to some extent today), it is a strong assumption it would be interesting to explore what other assumptions can yield deflation security. One possible way to avoid this assumption is with increased reliance on a TP. It would be interesting to determine what solutions lie between the extremes of a TP who monitors all interactions and a fully anonymous channel that facilitates more efficient protocols.

The second area in which research would be useful is in improving the performance of protocols that follow the same basic structure as ours. For example, 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. In addition, it would be interesting to analyse how these protocols perform over time. That is, to precisely characterise (and thus hopefully, optimise) the trade-offs between the maximum audience size that is measurable, the client storage and communication overhead.

### Acknowledgements

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

### References

- Anupam, V., Mayer, A., Nissim, K., Pinkas, B. and Reiter, M., K. (1999) ‘On the security of pay-per-click and other Web advertising schemes’, *Computer Networks (Amsterdam, Netherlands)*, Vol. 31, pp.1091–1100.
- Baudes, R., and Zabele, S. (1993) RFC 1458: Requirements for multicast protocols, May 1993. Status: INFORMATIONAL.

- Bloom, B. (1970) 'Space/time trade-offs in hash coding with allowable errors', *Communications of the ACM*, Vol. 13 pp.422–426.
- Chaum, D. (1988) 'Elections with unconditionally-secret ballots and disruption equivalent to breaking RSA', *Advances in Cryptology Eurocrypt*, Vol. 330, pp.177–182.
- Coffey, S. (2001) 'Internet audience measurement: a practitioner's view', *Journal of Interactive Advertising*, Vol. 1, pp.65–75.
- comScore Networks, net-score product description, Available at <http://www.comscore.com/>
- De Bonis, A. and Masucci, B. (2000) 'An information theoretical approach to metering schemes', Paper presented at the *ISIT 2000*. In proceedings.
- De Bonis, A., Blundo, C. and Masucci, B. (2002) 'Bounds and constructions for metering schemes', Paper presented at the *Communications in Information and Systems 2002*. In proceedings.
- Dyck, T. (2002) 'Yahoo chief scientist describes web attacks', *EWeek*, July
- FAST/ARF, Available at <http://www.fastinfo.org/measurement/pages/index.cgi/audiencemeasurement>
- Franklin, M.K. and Malkhi, D. (1998) 'Auditable metering with lightweight security', *Journal of Computer Security*, Vol. 6
- Goldwasser, S. and Bellare, M. (1999) Lecture notes on cryptography. Summer Course 'Cryptography and Computer Security' at MIT, 1996–1999.
- Greenman, C. (2002) 'Royalty fees threaten web stations', *New York Times*, April
- IDzap, Available at: <http://www.idzap.com>
- Ipsos-RSL Broadcast Division. 'An introduction to the ipsos-rsl broadcast division', Available at: <http://www.rslmedia.co.uk/broadcast/experience.html>
- Kamath, A.P., Motwani, R., Palem, K. and Spirakis, P. (1995) 'Tail bounds for occupancy and the satisfiability threshold conjecture', *Random Structures and Algorithms*, Vol. 7, pp.59–80.
- Kuhn, M.G. (2000) 'Probabilistic counting of large digital signature collections, 2000', Paper presented at the *9th USENIX Security Symposium*. In proceedings.
- Masucci, B. and Stinson, D.R. (2001) 'Efficient metering schemes with pricing', *IEEE Transactions on Information Theory*, Vol. 47, pp.2835–2844.
- Inc. MeasureCast. 'An analysis of streaming audience measurement methods', Available at: <http://www.measurecast.com/docs/Audience, Measurement, Methods.pdf>
- Mitzenmacher, M. (2001) 'Compressed bloom filters', *ACM Symposium on Principles of Distributed Computing*, p.144–150.
- Motwani, R. and Raghavan, P. (2000) *Randomized algorithms*. Cambridge University Press.
- Naor, M. and Pinkas, B. (1998) 'Secure and efficient metering', *Advances in Cryptology – Eurocrypt '98*, Vol. 1403, pp.576–589.
- Nielsen Media Research. 'Audience measurement services – the global leader for actionable internet information', Available at: <http://www.nielsen-netratings.com/marketing/advertising/audience measurement/>
- Nikov, V., Nikova, S., Preneel, B. and J. Vandewalle. (2003) Paper presented at the 'Applying general access structure to metering schemes, 2002' *International Workshop on Coding and Cryptography (WCC 2003)*. In proceedings.
- Ogata, W. and Kurosawa, K. (2000) 'Provably secure metering scheme', *Advances in Cryptology – Asiacrypt '00*, Vol. 1976, pp.388–398.
- Reiter, M.K., Anupam, V. and Mayer, A. (1998) 'Detecting hit shaving in click-through payment schemes', Paper presented at the *3rd USENIX Workshop on Electronic Commerce*, pp.155–166. In proceedings.
- Schoenmakers, B., Cramer, R., Franklin, M. and Yung, M. (1996) 'Multi-authority secret ballot elections with linear work', *Advances in Cryptology – Eurocrypt '96*, Vol. 1070, pp.72–83.

Shields, C. and Levine, B. (2000) ‘A protocol for anonymous communication over the internet’, Paper presented at the *ACM Conference on Computer and Communications Security (CCS)*. In proceedings.

## Appendix

### *Bayesian analysis of the bloom filter-based protocol*

In this appendix we discuss how Bayes theorem (see, for example, Motwani and Raghavan, 2000) can be used to estimate the number of requests seen by the content distributor via  $I(\vec{b}) = \lceil \ln(1-w(\vec{b})/m) \rceil / \lceil s \ln(1-1/m) \rceil$ . For this analysis we impose a technical requirement that  $w(\vec{b}) \leq 2m/3$  in order that the estimate  $I(\vec{b})$  is sufficiently accurate (see Theorem 7). The following theorem implies that if we use  $I(\vec{b})$  as an estimate of the number of requests received by the CD then, with extremely high probability, the actual number of requests will differ from our estimate by at most  $\alpha\sqrt{m}$  for some small value,  $\alpha$ .

**Theorem 1:** Fix  $n_{\max} < m \ln s / s$  and  $< [1 - (1/s)]m$ . Let  $X$  be a random variable representing the set of nonces received by the CD. We model  $X$  as taking on values at random from the set  $\{\{x_1, \dots, x_n\} | x_i \in \mathbb{Z}/2^r\mathbb{Z}, 0 \leq n < n_{\max}\}$ . 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<sup>10</sup>.

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  $\mu(i) = E[w(X) \mid |X| = i] = [1 - (1 - (1/m))^i]m$ . When  $\mu(i) < [1 - (1/s)]m$  (or, equivalently, when  $i < (m \ln s / s)$ ), then  $(d\mu/di) > 1$ .

By Kamath et al.’s occupancy bound [18],

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

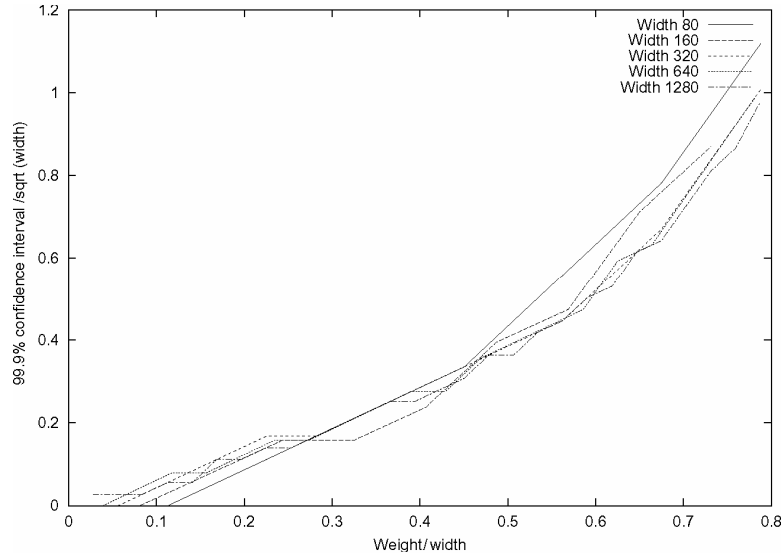
By combining this bound with the Bayesian equation above and unenlightening algebraic manipulation, one can derive that

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

The only tricky part of the derivation is to use that  $|i - I(W)| \leq |W - \mu(i)|$ , which holds because  $(du/di) > 1$ . □

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

**Figure A** The accuracy of using  $I(x)$  to estimate the number of insertions performed on a Bloom filter. Note that the confidence intervals have been normalised to  $\sqrt{m}$ . When the CDs submit Bloom filters  $\bar{b}$  with  $\omega(\bar{b}) \leq (2m/3)$ , we can conclude that with 99.9% confidence, the actual number of requests received by the CD differs from  $I(\bar{b})$  by at most  $(4\sqrt{m}/5)$ .



## Notes

- <sup>1</sup>Listeners that take advantage of this backdoor may lose their anonymity, but we cannot guarantee the anonymity of clients that do not use our protocols.
- <sup>2</sup>The hash functions need not be cryptographically secure. They are just used to map the universe of objects down to integers.
- <sup>3</sup>Receiving the content anonymously also allows the auditor to determine that the CD is not distributing keys to the 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 the clients by the CD 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.
- <sup>4</sup>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 CD.
- <sup>5</sup>In general, it is unwise to choose  $p_2 = 1$  and  $t = 2$ , because the CD then knows that any key  $k$  that is not stored by all the clients is in  $S_1$  with probability 1. However, even in this example it is arguable that using key  $k$  yields a successful attack, since we expect  $k$  to be stored only by around seven clients ( $0.6n_{max}$ ) which is already very close to the six client audience that the auditor will infer from the usage of  $k$ .
- <sup>6</sup>Note that the confidence intervals hold up to  $n = 13$  only.
- <sup>7</sup>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.
- <sup>8</sup>The indices may be permuted so that the client does not learn anything about the degree of the polynomial on which it knows a point.
- <sup>9</sup>If the CD knows precisely what points the clients have, there is a small chance that the CD will be able to create a polynomial of low degree which allows all clients to decrypt, even when some clients have points on polynomials of high degree. Hence, we hide the points from the CD. Abuse of this requirement is detectable by the auditor by repeated anonymous content requests, as in earlier protocols.
- <sup>10</sup>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.