# An Auditable Metering Scheme
# for Web Advertisement Applications

Liqun Chen and Wenbo Mao

Hewlett-Packard Laboratories, Bristol
Filton Road, Stoke Gifford, Bristol, BS34 8QZ, the UK
*{liqun_chen, wenbo_mao}@hp.com*

**Abstract.** *This paper proposes a cryptographic mechanism for metering the duration and/or the number of instances of running a data process. This mechanism has the following property: knowing a secret, one can validate a piece of metering evidence in constant units of time while without the secret the job of generating a valid piece of evidence requires time indicated by a value in the evidence. Because the mechanism utilises a well-known computational complexity problem, the meter based on it can be implemented in software yet is tamper-resistant.*

*We will address the use of this mechanism in building an auditable metering scheme for finding the popularity of web sites. The scheme is suitable for rapidly increasing web advertisement applications. We will also discuss the related security issues and mention some other applications, which can benefit by using the mechanism.*

**Keywords:** *Auditable metering, Web advertisement, Tamper-resistant software, Lightweight security.*

## 1 Introduction

Use of the Internet, especially the world-wide-web (the web), is rapidly growing. Accompanying this growth is a speedy emergence of new electronic services offered over the Internet. Many of such new services require metering in terms of the length of time or the number of instances that the services are used.

In this paper we propose an auditable metering scheme that is suitable for web advertisement applications. In our model, there are three entities involved: an advertisement content provider (we equate it to the advertiser), a service provider (e.g. Internet Service Provider (ISP)) and a client (i.e., a customer). The ISP runs a web advertising service by displaying web-based ad pages for the advertiser, and the client visits the web pages and browses the ads. In general, both the ISP and the advertiser will naturally like to encourage the client to spend more time reading the ads. For this reason, the client does not have to pay for browsing the ad pages. Under this model, we assume the following payment agreement between the ISP and the advertiser: the fee for an ad paid by the advertiser to the ISP is dependent on the popularity of the web page that contains the ad, i.e., the amount of time that this advert has been accessed for by the clients. It is important, therefore, for the advertiser to be able to measure the popularity of a web page in order to determine an appropriate charge for an advert.

For the purposes of building such a web advertising service and payment model, we will propose a cryptographic mechanism for metering the duration and/or the number of instances of running a data process. This mechanism has an attractive property: knowing a secret, one can validate a piece of metering evidence in constant units of time while without the secret the job of generating a piece of valid evidence requires time indicated by a value in the evidence. Because the mechanism utilises a well-known computational complexity problem, the meter based on it can be implemented in software yet resists tampering.

## 1.1 Motivation

The electronic commerce potential of the Internet, in particular, that of the web, has brought forward a new business of offering free access to the Internet. Organisations, such as Geocities (www.geocity.com), Dixons (www.freeserve.net), TESCO (www.tesco.co.uk/indexn.htm), Yahoo! (www.yahoo.com), and BT (www.btinternet.com), are a few examples of free Internet Service and/or Content Providers (ISPs/ICPs). Web advertisement is considered one of the sources of revenue for free ISPs/ICPs. The current figures show that advertisement on the Internet is a multi-billion dollar industry in the year 2000 [5]. Compared with the traditional hardcopy-printing-based advertisement, the web-based version is cost effective, speedy and can be conveniently connected to shopping over the Internet. For example, after viewing an advert, a client can order goods right away. Another important advantage of the web-based over the paper-based advertising is the ease of collecting data relating to clients' purchasing behaviour and of mining that information. Such information is a valuable commodity for a seller.

As a rapidly growing application, the web advertisement requires more research effort in order to identify and formulate suitable service and payment models and various related metering problems, which will in turn stimulate new research topics in computer security.

## 1.2 Previous Work

There are many existing commercial enterprises that try to sell services for measuring the activity of web sites (a partial list of these, in [6], includes companies like I/PRO, Nielsen, NetCount, RelevantKnowlegde, and others). These companies use mainly two methods: sampling the activities of a group of web clients, and installing an audit module in web sites. Naor and Pinkas [6] argued that sampling could be very inaccurate. On the other hand, an audit module installed in web sites cannot prevent the web servers from forging the auditing results. Thus, neither is suitable for the web advertisement applications.

Franklin and Malkhi [2] initiated the work of metering via a tamper-resistant software approach (a revised version of the paper is in [3]). In their paper, Franklin and Malkhi addressed the problem of artificial client visits. They provided a lightweight security solution, which is based on a timing function that makes a large number of artificial visits very costly. Every time a client visits a web page, the client's computer will run the timing function, and then the result of the computation will be sent to an auditing proxy. Since the timing function used in their scheme is simply repeating the construction of a hash function value and it is infeasible to invert such a hash function value (using MD-5 or SHA), to verify correctness of such a piece of evidence accurately must use the same procedure for the evidence generation. To reduce the computation of verification, they introduced two approaches: a statistical auditing function and an approximate auditing function. Both of these functions check a part of evidence values and provide a probabilistic verification result. The drawback of their scheme is that in order to ensure that the verification is reasonably accurate, the auditing process cannot be done in constant units of time.

In Eurocrypt'98, Naor and Pinkas proposed a secure but inefficient metering scheme [6]. In that paper, they studied methods for performing secure web metering using robust secret sharing schemes. Their approach provides computationally secure metering, which relies on the involvement of a trusted third party (an auditing agency) to compute and distribute periodically secret shares to both the web server and all potential clients. A simplification of their method is as follows: a secret is distributed by the audit agency to all $N$ clients, and the server needs to receive requests from at least $K$ clients in order to reconstruct the secret. The secret is then sent by the web server to the audit agency as a proof of having received at least *K* hits during the audit period. The main drawback of this proposal consists in the fact that clients need to be initialised for each audit period, and this forces them to communicate with the audit agency. Furthermore, this scheme cannot be used to a system, which is open to all clients who do not need a membership to use it. Clearly, this is not an open system, and therefore it is not suitable for a web advertisement application.

Bergadano and Mauro, [1], proposed a hardware-based tamper-resistance solution. They require the web server to be equipped with a special tamper-proof hardware that is provided by the certification agency. The drawback of this proposal is that the site's web server will have to be modified.

### 1.3 This Work

This paper proposes a software-based cryptographic mechanism, which utilises a well-known computational complexity problem, for metering the length of time or the number of instances of a data processing system.

We make use of this mechanism to solve the metering problem in a web-based advertisement service and payment model. The main difference between this model and the previous works is that we will first argue that an advertiser is an important entity in the web advertising and billing services, so that we designate this entity to act as an auditor. It is however not necessary for the advertiser to be on-line during the time when a piece of metering evidence is generated. We only need the advertiser to be involved in the system initialisation stage for the generation of some primary parameters for the system. The verification of the metering evidence will be carried out in an off-line manner.

Similar to the auditable metering scheme of Franklin and Malkhi [3], our scheme can be implemented without changing any existing structure of web servers or browsers, and our solution can provide a timing measure for client visits in a lightweight security way. By lightweight security in a web advertisement application, we mean that it is possible for someone to forge a small amount of valid timing evidence but none of the party involved will have an incentive to do so. However, without knowing a secret, the job of generating a huge amount of valid evidence requires time that is indicated in the evidence. As it was argued in [3], lightweight security (which has been discussed for many different applications and the first example was in [4]) is a good strategy for metering web popularity.

Compared with Franklin and Malkhi approach, our scheme provides a more rigorous and more efficient verification procedure. The verification procedure used in our auditable metering mechanism gives an accurate result instead of a probabilistic one as in their solution. Furthermore, in our scheme, any piece of metering evidence, no matter how big the amount of the time duration or the number of instances indicated in the metering evidence is, can be verified in constant units of time. The time complexity for verifying a piece of metering evidence is independent of the amount of the time that the evidence has been generated.

### 1.4 Organisation

The rest of this paper is organised as follows. In Section 2, we present our auditable metering mechanism. In Section 3, we first identify a web-based advertisement service and payment model; secondly address some security requirements, and finally discuss implementation issues. In Section 4, we will introduce a few other applications that benefit from using the auditable metering mechanism as well. We conclude in Section 5.

# 2 An Auditable Metering Mechanism

We present two algorithms. The first is called Timing Algorithm, and the second, Auditing Algorithm.

## 2.1 The Timing Algorithm

Let $n = pq$ be an integer with two large prime factors $p$ and $q$ which are roughly the same size; $x$ and $e$ be two random integers less than $n$ where $e$ is odd. Timing Algorithm inputs *(n, x, e)* and outputs *(t, a)* with the following computations.

**Algorithm 1.** *Timing(n, x, e)*

```
        $y \leftarrow h(x)$; $a \leftarrow y$; $t \leftarrow 1$;
        while there is no “stop” interruption {
                $a \leftarrow ya^e \ (mod\ n)$;
                $t \leftarrow t+1$;
        }
        return (t, a);
    end,
    where
        h() denotes a secure one-way hash function that the system has agreed;
        symbol ← means “is made equal to” (for example, in $y \leftarrow h(x)$, variable “y” is
        made equal to the value of integer “h(x)”).
    Upon termination, the tuple (t, a, x, e, n) constitutes a piece of metering evidence.
```

Algorithm 1 iterates its computation while a computer platform is executing a task (for instance, the client is browsing a web page which contains an ad). Each iteration represents one “tick”, which may in turn represents a number of fixed units of time. The number of ticks accumulated is represented by the value of *t*. The output pair (*t, a)* satisfies:

$$a = h(x)^{b} \ (mod\ n), \qquad (1)$$

where

$$b = 1+e+e^{2}+ \cdots + e^{t} \ (mod\ \lambda(n)). \qquad (2)$$

Here $\lambda(n) = lcm(p-1, q-1)$, the least common multiple of $p-1$ and $q-1$.

Clearly, to generate a valid piece of metering evidence, Algorithm 1 needs to execute *t* iterations, each contains a modulo exponentiation. We should emphasise that without knowing the factorisation of *n*, the *t* exponentiation seems to be necessary: for *n»t*, to date no algorithm exists that can generate a piece of valid metering evidence using steps fewer than *t*.

## 2.2 The Auditing Algorithm

Let *(t, a, x, e, n)* be a piece of metering evidence generated by Algorithm 1. Auditing Algorithm inputs the tuple *(t, a, x, e, n)* and outputs *YES* or *No*.

**Algorithm 2.** *Auditing(t, a, x, e, n)*

```
        $y \leftarrow h(x);$
        $E \leftarrow e^{(t+1)} \pmod{\lambda(n)}$ ;
        if ($a^{(e-1)} \equiv y^{(E-1)} \pmod{n}$){
                return (YES);
        }
        else{
                return (NO);
        }
    end.
```

Clearly, with $\lambda(n)$, Algorithm 2 will terminate in three modulo exponentiations and answer *YES* or *NO.* The time complexity is independent of *t* in the metering evidence.

## 2.3 Analysis of the Mechanism

The above mechanism has the following properties.

1. *Without factoring n, there exists no algorithm that can generate (t, a) from (n, x, e) in less than t exponentiations mod n.*

Let exponentiation modulo $n$ take one tick (representing some given units of time). Then, generating a valid pair *(t, a)* using Algorithm 1 takes *t* ticks.

2. *Algorithm 1 is intrinsically sequential.*

Clearly, every successive tick can only be performed on the result of a previous tick; there is no obvious way to parallelise the procedure using multiple processors aiming at saving time. One may compute the exponent $b'$ first and then perform one exponentiation modulo $n$. However, without knowing the order of $x \bmod n$, the exponent

$$b' = 1+e+e^2+ \cdots + e^t \tag{3}$$

is not compact, which means that the size of $b'$ is $t|n|$ (here $|n|$ means the bit size of $n$ in the binary representation). Therefore, the exponentiation using the large exponent $b'$ shall still take $t$ ticks. A disadvantage of doing so is that a huge space is required while no time is saved.

3. *Verification of the metering evidence is efficient with knowledge of the secret $\lambda(n)$ .*

To verify the metering evidence *(t, a, x, e, n)* using Algorithm 2 only takes three ticks, since computing

$$E = e^{(t+1)} \pmod{\lambda(n)} \tag{4}$$

and

$$a^{(e-1)} \equiv h(x)^{(E-1)} \ (mod\ n) \qquad (5)$$

only involves three modular exponentiations. Algorithm 2 also has a constant space complexity of $|n|$, since $E$ has the same size of $n$. Thus, the auditor is able to check a large amount of evidence efficiently. The efficiency is obtained as a result of the auditor's knowledge of the secret $\lambda(n)$, which allows him to compute $E$ in (4) in a compact manner.

4. Let $a(t)$ denote the output value $a$ of Algorithm 1 after $t$ ticks. Then w*ithout factoring n, it is infeasible to compute a(t-1) from a(t).*

It is obvious that any one is able to compute $a(t)$ from $a(t-1)$, $x$, $e$, and $n$ by running one "tick" in Algorithm 1. However, computing in the backward direction from $a(t)$ to $a(t-1)$ involves extracting $(e+1)$*-th* root and inverting hash function.

5. *Optionally, the value of e can be fixed.*

In Algorithm 1, $e$ is a random number. Alternatively, using a fixed value $e$ instead of a random number should be fine, as long as $e$ is a positive odd number less than $n$.

6. *The number x can include some special information if required.*

The basic requirement on the number $x$ is unique and fresh. For the purposes of meeting different requirements, the number $x$ can actually include some special information.

Example 1: this number can be constructed by using more than one random numbers; respectively each random number is contributed by one entity, e.g., the server, the client or the auditor. Thus, every contributor is able to check the freshness of the metering evidence.

Example 2: this number can include some information to specify particular servers, auditors or clients if required, such as the names or other identifiers of these entities.

7. *With the secret (p, q), a valid piece of metering evidence can be constructed in a constant length of time.*

In Algorithm 2 we have shown an efficient procedure for the auditor, who knows the secret of factorisation of $n$, to validate a piece of metering evidence. Furthermore, he can actually construct such a piece of metering evidence without running Algorithm 1. Based on $E = e^{(t+1)} \ (mod\ \lambda(n))$ and $a^{(e-1)} \equiv h(x)^{(E-1)} \ (mod\ n)$ in Algorithm 2, in order to construct the value $a$, the auditor needs to extract the $(e-1)$*-th* root of $h(x)^{E-1} (mod\ n)$, which is $a$. Knowing the factorisation of $n$, such roots can be computed in polynomial time in the size of $n$.

Since the auditor is able to forge the metering evidence, this auditable metering mechanism is only suitable for applications where we can trust that the auditor has no incentive to forge a piece of metering evidence. This is the case in our web-based advertisement services to be described in the next section, where the auditor is an advertiser, who is supposed to pay for the time that his ads are browsed based on the metering evidence generated.

Note that Property 7 can actually be used to benefit some applications. As to be discussed in Subsections 4.2 and 4.3, "Rent-to-Own" and "Pay-per-Play" are two examples of such services.

8. *Without factoring n, any verifier given a(t) is able to validate a(t), if he also knows that a(t+1) is a correct value of evidence .*

This is obvious. Note that Property 8 can be used to benefit some applications. In Section 4 we will provide some examples to use this property.

For the purpose of security and performance, we should choose a proper length of size of the number *n,* such as a similar length of a secure RSA module [7]. We recommend using *n* of the length of 1024 bits.

# 3 Web-based Advertisement with Auditable Metering

In this section, we introduce our auditable metering scheme for web advertisement applications. We start with identifying a web advertising service and payment model with three entities involved, and then discuss requirements on security and implementation of the metering scheme. After that, we will present the metering scheme itself.

## 3.1 The Model

There are three entities in our model of the web advertisement services: the Internet Service Provider (ISP), the advertiser and the Client.

The advertiser provides the content of an advert. The ISP runs an advertising service for this advert in a web page via the Internet. The Client obtains the information of the advert by visiting the ISP's web site. Both the ISP and the advertiser would naturally like to encourage the client to spend more time viewing the advert. The client does not have to pay for accessing the web pages. The ISP and the advertiser have a payment agreement: the advertising fee for an advert, which is paid by the advertiser to the ISP, is proportional to the popularity of the web page that contains the advert, i.e., the amount of the time duration when this page has been accessed for by the clients.

## 3.2 Requirements

Our design requires the following.

For all parties:

1. The scheme should provide lightweight security (see Subsection 1.3 for the meaning of the lightweight security in a web advertisement application). We assume that there is no incentive for the advertiser to forge a piece of metering evidence. It must not be cost effective for the ISP, without colluding with a great number of clients, to create a significantly large volume of artificial visits.

2. For this scheme to be widely deployed, both the computation and the storage requirements of the client, of the ISP and of the advertiser sides should be reasonable.

For the ISPs:

3. Any client visits to the advertising web pages should be recorded by the corresponding ISP, whether the visits are from the ISP itself or from any proxy server. In this way, repeated visits by clients behind a firewall can also be captured.

4. With related to use of this scheme, no modification of any existing structure of the web-site contents is required.

For the clients:

5. Privacy: No client registration is needed. This also ensures the openness of the system.

6. Compatibility: No any modification is needed on client computer system in order for the client to use this system. The scheme should be deployable in the commercially available web browsers.

For the advertisers:

7. Free of forging: The scheme must make no sense for the advertiser to forge metering evidences.

8. Off-line: It should not be necessary for the advertiser to be on-line during the time of metering evidence generation. The advertiser should only be active in the system setting up time and in the time of verifying metering evidence.

## 3.3 The Scheme

Our scheme has the following three parts.

**Part 1: parameter generation.**

The advertiser generates the value *n*, as described in Algorithm 1, forwards it to the ISP, and keeps *p* and *q* secret. This may be done in advance of any advert downloading request or on demand. Optionally, if the scheme makes use of a fixed value *e*, as analysed in Property 5 of Subsection 2.3, the value *e* may be generated by the advertiser and then sent to the ISP together with the value *n*.

**Part 2: metering procedure.**

1. The client "clicks" on a hyperlink to download from the ISP a web page containing an advert. This causes the browser to generate a request message, which is sent to the ISP.

2. Upon receipt of the request from the client, the ISP generates a mobile code of Algorithm 1 that includes the numbers of *n*, *e* and *x*. The ISP then sends the advert web page and the mobile code back to the client.

3. Upon receipt of the above, the mobile code Algorithm 1 starts to execute. In the meantime, the web browser of the client displays the web page.

4. When the client leaves the web page, the mobile code stops the execution and outputs the metering result *(t, a, x, e, n)*, which is then sent back to the ISP.

5. Upon receipt of the metering result *(t, a, x, e, n)*, the ISP stores it for future billing of the advertiser.

**Part 3: auditing procedure.**

Later in the payment time, if the advertiser wishes to validate the metering evidence, he can do so by running Algorithm 2.

## 3.4 Security Analysis

Our metering scheme has the following two security features.

*1. This scheme provides a lightweight security solution, which is ensured by using the auditable metering mechanism described in Section 2.* An ISP may try to create the metering evidence with the intention to over-charge an advertiser. However, as has already been described above, the only way for the ISP to generate a piece of metering evidence is to behave as a client, which is very labour-intensive if a significant amount of evidence needs to be generated. Indeed, the time and resources taken to generate the evidence would cost the ISP more than what would be gained from the increased advertising revenue.

*2. Because the metering mechanism used in this scheme utilises a well-known computational complexity problem (difficulty of factorisation), the scheme can be implemented in pure software yet resists tampering.* We assume that there is no incentive for the clients to attack the advertising web page in order to bypass the mobile code of Algorithm 1. This is a reasonable assumption for our web advertising service and payment model, because the clients in this model do not pay for visiting the advert web pages.

# 4 Other Applications

In this section, we show some other applications, which may also benefit from using the auditable metering scheme.

## 4.1 Software Rental Service

In this application, the content provider is a software rental company. The software to be rented is stored by the ISP ready for downloading by clients.

The client first selects the software he wishes to rent by clicking on an appropriate hyperlink. Upon receipt of the request, the ISP sends the client the requested software code accompanied by mobile code of Algorithm 1 and the three numbers *(n, x, e)*, When the client runs the software, he runs the meter too, in Algorithm 1. The meter runs for as long as the client uses the software. When the client finishes using the software, he “exits” and the meter transmits the metering result back to the software rental company.

Depending on the nature of the software that is being rented, it may be preferable for the software to incorporate the functionality of the mobile code, rather than having the mobile code as a separate piece of functionality. An advantage of this approach is that it is difficult to separate the operation of Algorithm 1 from use of the software.

## 4.2 Rent-to-Own Service

In an alternative form of the above service, the software rental application may be arranged in the following way. After a certain time (e.g. T units of time) of paid use, the client can use the software without further charge. Such an arrangement is sometimes known as “rent-to-own”.

For example, after T units of time of paid use, the client gets a certificate automatically issued by the programme. The certificate includes two data tuples: the first tuple is *(t=T+1, a=f(x, T+1), x, e, n)* where *a=f(x, T+1)* can be constructed by a meter owner in the way described in Property 7 of Subsection 2.3; the second tuple is *(t=T, a’=f(x, T), x, e, n),* where *a’=f(x, T),* is the output of Algorithm 1 specified in Section 2. For the purpose of message authentication, the first tuple *(t=T+1, a=f(x, T+1), x, e, n)* may have been signed by the meter owner, such as either the content provider or the ISP. This certificate proves that the client has run the programme at least *T* units of time, and that is universally verifiable.

With this certificate, the client can run the software on any different computing platform, which is able to verify the validation of the certificate with only one "tick" by using Algorithm 1 in the way analysed in Property 8 of Subsection 2.3. More specifically, once the computer has access to Algorithm 1, it can substitute the values ($T$, $a'=f(x, T)$, $x$, $n$, $e$) into the algorithm and then simply iterate it one more time. If the iteration results in $a=f(x, T+1)$, then the computer can be arranged to understand that the client has genuinely acquired the right to use the software without further payment.

### 4.2 Pay-per-Play Service

In this application, a client is able to play games using a "pay-per-play" service. In a similar fashion to the previous two applications, the client can select and download from an ISP a computer game provided by a game content provider. Accompanying the downloaded game is a metering process, which runs on the client's computer when the client starts playing the game, and continues for as long as the client continues to play the game.

During the time of playing the game, for example, the correct values of metering evidence (say, one piece of evidence for each session) have to be sent to either the content provider or the ISP, who is able to validate the evidence in the way specified in Properties 7 and 8 of Subsection 2.3, to keep the game running continuously. Based on the verification of metering evidence, the content provider may charge the client and the ISP may charge the content provider.

## 5 Conclusion

This paper has proposed a new auditable metering mechanism based on a well-known computational complexity problem. The paper has also discussed how to use this mechanism to implement a pure software-based temper-resistant metering scheme, which can be used in web advertising services and other Internet services. Compared with some previous metering approach with similar designing requirements, this scheme offers a more rigorous and efficient procedure for the evidence verification.

## References

[1] F. Bergadano and P. De Mauro. Third party certification of HTTP service access statistics. In *the Proceedings of the 1998 Cambridge International Workshop on Security Protocols*, Cambridge, England, 1998.

[2] M. K. Franklin and D. Malkhi. Auditable metering with lightweight security. In *the Proceedings of Financial Cryptography '97*, LNCS 1318, pp 151-160, Springer, Berlin, 1997.

[3] M. K. Franklin and D. Malkhi. Auditable metering with lightweight security. *Journal of Computer Security.* 6(4): 237-255, 1998.

[4] S. Glassman, M. Manasse, M. Abadi, P. Gauthier, and P. Sobalvarro. The Millicent protocol for inexpensive electronic commerce. In *the Proceedings of the 4th International World Wide Web Conference*, pp 603-618, December 1995.

[5] M. Kinsman. Web advertising 1997: market analysis and forecast. *Cowles/Simba Information*. Stamford, Connecticut. May 1997.

[6] M. Naor and B. Pinkas. Secure and efficient metering. In *the proceedings of EUROCRYPT '98*, LNCS 1403, pp 576-590, Springer, Berlin, 1998.

[7] R.L. Rivest, A. Shamir, and L.M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*. 21(2): 120-126, 1978.