Towards Lightweight Secure User-Transparent And Privacy-Preserving Web Metering

Alarifi, Fahad Abdulkareem

Awarding institution:
King's College London

Towards Lightweight Secure User-Transparent And Privacy-Preserving Web Metering

by Fahad Alarifi

A thesis submitted in partial fulfillment for the degree of Doctor of Philosophy

in the
Department of Informatics
in the
Faculty of Natural & Mathematical Sciences

October 2015

Abstract

Privacy is an issue today as more people are actively connecting and participating in the Internet. Problems arise when such concerning issue is coupled with security requirements of online applications. The *web metering* problem is the problem of counting the number of visits done by users to a webserver, additionally capturing data about these visits. There are trade-offs between designing secure web metering solutions and preserving users' privacy. There is also a dilemma between privacy-preserving solutions versus accuracy of results. The problem becomes more difficult when the main interacting party, the user, is not inherently interested to participate and operations need to be carried out transparently.

This thesis addresses the web metering problem in a hostile environment and proposes different web metering solutions. The web metering solutions operate in an environment where webservers or attackers are capable of invading users' privacy or modifying the web metering result. Threats in such environment are identified, using a well established threat model with certain assumptions, which are then used to derive privacy, security and functional requirements. Those requirements are used to show shortcomings in previous web metering schemes, which are then addressed by our proposed solutions. The central theme of this thesis is user's privacy by user-transparent solutions.

Preserving users' privacy and designing secure web metering solutions that operate transparently to the user are two main goals of this research. Achieving the two goals can conflict with other requirements and such exploration was missed by former solutions in the literature. Privacy issues in this problem are the result of the dilemma of convincing interested parties of web metering results with sufficient details and non-repudiation evidence that can still preserve users' privacy. Relevant privacy guidelines are used to discuss and analyse privacy concerns in the context of the problem and consequently privacy-preserving solutions are proposed. Also, improving the usability through "securely" redesigning already used solutions will help into wider acceptance and universal deployment of the new solutions. Consequently, secure and privacy-preserving web metering solutions are proposed that operate transparently to the visitor.

This thesis describes existing web metering solutions and analyses them with respect to different requirements and desiderata. It also describes and analyses new solutions which use existing security and authentication protocols, hardware devices and analytic codes. The proposed solutions provide a reasonable trade-off among privacy, security, accuracy and transparency. The first proposed solution, transparently to the user, reuses Identity Management Systems and hash functions for web metering purposes. The second hardware-based solution securely and transparently uses hardware devices and existing protocols in a privacy-preserving manner. The third proposed solution transparently collects different "unique" users' data and analyses fingerprints using privacy-preserving codes.

Acknowledgements

First and foremost, I want to thank my supervisor Professor Maribel Fernández for her valuable time, patience and continuous feedback. I learned many things, from how to better research to how to write. I am also extremely grateful to Dr. Steve Barker; he will always be remembered. I also want to thank Professor Muttukrishnan Rajarajan and Professor Daniel Dougherty for their valuable feedback and insights. I also want to thank Dr. Waleed Alrodhan for his generous feedback and insights. I also want to thank Mr. Jacob Jathfa at University of Cape Town for speeding up the learning curve and for his inspiring methods.

I want to thank my cousin Dr. Fahad Suliman and my brothers Dr. Hani Alarifi and Mr. Nasser Alarifi for their moral support and encouragement. I also want to thank Professor Keith Martin and Dr. Alan Tomlinson at Royal Holloway for their feedback and suggestions. I also want to thank Mr. Bandar Alateyyah and Mr. Mansour Albussili for tests verification and optimisation.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Contents

This chapter provides the motivation and description for the web metering problem. This chapter also describes the research problems, the contributions and structure of the thesis.

1.1 Motivation

Web metering became a valuable measurement tool when "online advertising" services started playing an important role in the Internet. Web metering addresses the problem of charging for advertisements in a fair way for all involved parties. Looking at the history of advertising, advertisements have long been placed in areas that are very unlikely to be missed by the visitor[1]. For example, in 1900, one brand sign was placed next to a waterfall of Niagara Falls [41]. From the beginning of advertising, ensuring the delivery of the advertisements (and implicitly the ability to count unique viewers) was paramount. Traditional advertising methods cannot be directly applied here because there are differences between Internet-based services and the traditional ones. One aspect in this regard is the automation capability in Internet-based services [27]. Also, charging and billing for technology-based services has other different difficulties than the traditional ones [58].

The tendency in some current online business models is to offer webserver content and services free to visitors and the cost of this service has to be valued. Once the offered service is measured, another service, like online advertising, can cover up the cost. Web metering schemes attempt to measure the exposure of webserver content and services and provide web metering evidences[2] to other interested parties.

Also, advertisements based on Pay Per Click (PPC) are widely deployed today. However, there are security issues with the PPC model [87], and the model restricts payment to visitor clicks which is not an accurate translation of a web metering interaction. In addition, a clicks behaviour report by comScore and Starcom shows that 8% of Internet visitors account for 85% of all clicks[3], which questions the accuracy and integrity of such a scheme. Further, heavily visited webpages can have a wider range of advertisements based on web metering evidence, whereas rarely visited pages might display

[1]We use the terms visitor and user interchangeably in this thesis.

[2]Despite word evidence is non-countable, plural form is used here to refer to the generated pieces of information which each represents web metering evidence.

[3]www.comscore.com/Insights/Press-Releases/2009/10/comScore-and-Starcom-USA-Release-Updated-Natural-Born-Clickers-Study-Showing-50-Percent-Drop-in-Number-of-U.S.-Internet-Users-Who-Click-on-Display-Ads

fewer carefully targeted advertisements. This can also support the special case of contextual advertising, where online advertisements are displayed based on the content of webpages by matching available advertisements to the metered webpages.

The motivation for web metering is not limited to charging models. Measuring exact statistics of Internet visits can also help webmasters improve their website's performance. Webmasters can dedicate more effort to highly visited webpages, and such statistics can help them optimise the flow and interaction of website processes. In addition, knowing the popularity of webpages can change the way they are reached. For example, when a visitor browses a news webpage, he might be motivated to know the exposure of the *technology* section, which is displayed in terms of number of visits and is similar to ordering items based on bestselling criterion. The hotel problem [109] can be addressed by running a scheme to these single pages of the webserver rather than the "whole" webserver for that limited period of time. Consequently, inferences about all webserver's pages cannot be simply made from the aggregate of single pages, for that particular period of time. If advertisements are targeted to webpages rather than the whole webserver, *bounce* data would be a desired metric (despite its privacy concerns) to show which pages have not *bounced* users out (and give them more value).

Other potential applications for web metering, besides charging for advertisements, are for search engines and Content Distribution Networks (CDN). Typically, visitors find relevant websites through Internet search engines, which crawl and index webservers. Based on the visitor's search keywords, search engines return relevant results. In case the number of returned results is not small, it is important to rank them based on criteria such as their popularity [26]. Web metering schemes can measure webpage popularity, which allows search engines to order their results. However with the private information typically obtained, number of requests by third parties to Google about users' data kept on increasing while granted responses with some data slightly decreased[4].

Also, web metering can help CDN with evidence of visits to justify the cost of the content delivery to visitors as described in [112]. Typically, content distributor (or the webserver here) takes offline large content from the content producer in order to deliver it in an efficient way, geographically closer to the visitor. Then, content distributors need to communicate evidence of the incurred visits to content producers for payment or maintain such evidence in case there is a a conflict upon a payment.

1.2 Web Metering Description

This section describes the problem, defines four requirements and scope. It starts by describing the problem and the solution. Then, it defines two security (more precisely, privacy and integrity) and two functional (transparency and accuracy) requirements. In addition, it describes validation methods that can be used to check the achieved security properties. Last, it shows some challenges and the scope.

Web metering is a method to measure the interactions done between the visitor and the webserver over a specific period of time. The web metering model consists of the following four entities: a visitor, a webserver, Web Metering Enquirer Party (WMEP), and an audit agency or a Web Metering Service Provider (henceforth abbreviated to metering provider). The webserver is a device and an application that can host a website and provide services remotely. The visitor is a person that uses a platform to access the webserver through a network[5]. (Automated visits done by machines are outside the scope of this research partly because the research is motivated by online advertising.)

[4] www.google.com/transparencyreport/userdatarequests/

[5] Network is not restricted to Internet, for example, the visitor can access a webserver in Local Area Network (LAN).

WMEP is an entity interested in measuring the interactions between the visitor and the webserver and it is authorised to submit a query and receive web metering results. Audit agency is a trusted party which performs web metering operations, or can be referred to upon conflict regarding web metering evidence or for other information (e.g. verification keys of visitors). Metering provider is a third party which can provide a web metering service and its trust level can be elevated to be an audit agency. Trusted relationship in this problem domain is crucial to fairly provide evidence for the occurred web metering interaction. Figure 1.1 shows the web metering model consisting of these four entities and their connections.

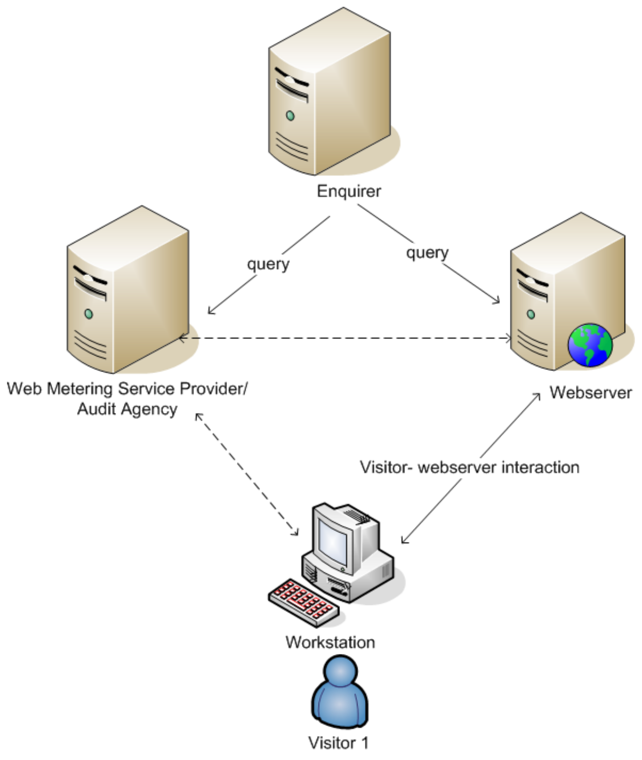

Enquirer
query
query
Web Metering Service Provider/
Audit Agency
Webserver
Visitor- webserver interaction
Workstation
Visitor 1

FIGURE 1.1: Web Metering Model

More formally, web metering entities are defined as follows.

Web Metering Entities. Assume there are m webservers that provide some online services. And there are n visitors that visit any webserver. Also, there are j optional third party entities which can be WMEP, audit agency or metering provider. S = { S_1 ,...,S_m } is the set of webservers, V= { V_1,...,V_n } is the set of visitors and TP = { TP_1,...,TP_j } is the set of third party entities. Sets S and V are disjoint and sets V and TP are disjoint too. However, sets TP and S have no restrictions among each other and can be intersecting.

A generic web metering protocol can be constructed as follows (AA denotes audit agency, MP denotes metering provider and T denotes time frame):

Web Metering Protocol.

1. **Visitor** $\leftrightarrow$ **Webserver** : an interaction between visitor and Webserver
2. **WMEP** $\rightarrow$ **Webserver/AA/MP** : Webserver ID, T
3. **Webserver/AA/MP** $\rightarrow$ **WMEP** : result, evidence, T

Initially, the visitor connects to the webserver at some point of time and sends a request. The webserver processes the request and delivers a human readable response[6] and this process repeats into further interactions. After capturing the visitor interaction, WMEP submits a query including an identifier of a specific webserver and requested time frame to audit agency, metering provider or webserver. Time is relevant here because web metering evidence is presented with respect to time. The recipient processes the received query and returns an answer. A typical example of such query is an online retailer sending the following request to a webserver hosting its advertisement: "On 27th of February, how many visits have been made to the webserver in the morning?". The webserver replies with the number of visits and evidence to support its response. Web metering evidence is defined as follows.

[6]An example of a human readable request-response is a visitor requesting to view a picture and the webserver responds with an html file containing the picture.

Web Metering Evidence. Web metering evidence is a token that proves an interaction between the visitor and the webserver took place as part of the visits, which is proof that the involved parties ideally cannot deny.

Such web metering evidence is provided to WMEP to back up the web metering results. Evidence can be challenged and there are various issues and levels for evidence, which will be described later in the section.

The problem is metering the interactions done between two entities: the visitor and the webserver. As shown in Figure 1.2, the visitor first connects to the webserver at some point of time and requests a service. Then, the webserver answers the visitor request completing a visit. The Data Capturer Component is a simple abstraction of the web metering function, which can be distributed among all the three involved entities. More precisely, web metering problem and its solutions (i.e. web metering schemes) are defined as follows.

Web Metering Problem And Scheme. The web metering problem is counting the number of visits done by the visitor to the webserver and additionally capturing data about these visits. A web metering scheme provides the number of visits and possibly captures the visits' data between the visitor and the webserver in order to provide web metering result and evidence. The web metering scheme goes through initialisation, interaction and evidence verification phases.

Web metering evidence can have different levels depending on the web metering scheme to back up the produced web metering result. A non-repudiation service can be used to enable an entity in a communication channel to prevent another party from denying having taken a particular action [42]. A non-repudiation service is mainly concerned here with providing such evidence that an interaction (e.g. a mouse click between the visitor and the webserver took place). Also, how much detailed such evidence should be to resolve disputes, is a problem in web metering environment.

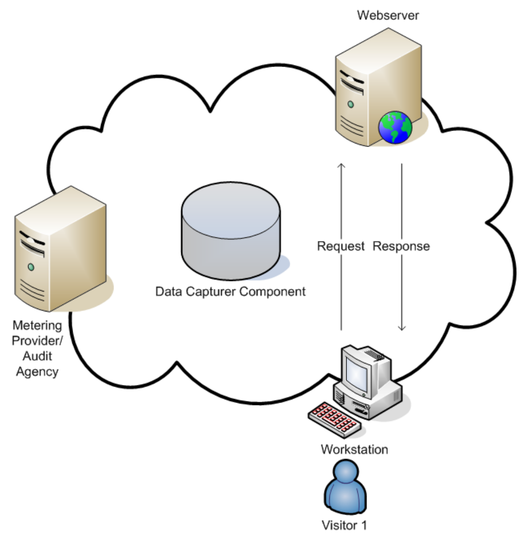


FIGURE 1.2: Basic Web Scenario

Privacy.

There are various definitions for privacy but we consider the following definition.

Definition 1. ([66])

Privacy is the right of individuals to control or influence what information related to them may be collected and stored and by whom; and to whom that information may be disclosed.

Such privacy in web metering context is concerned with the exposure of private information of the visitors during the web metering scheme operation. Web Metering schemes should typically not collect information about the visitor and possibly bind this information to his identity.

Integrity.

The integrity requirement for web metering schemes is defined as follows.

Integrity. A web metering scheme satisfies integrity requirement if its web metering operations are executed as expected per its specifications and cannot be affected by an adversary, preserving stored and transferred evidences and data.

This integrity requirement is needed to prevent actions that can change the web metering result and evidence. Also, on the extreme end of integrity, web metering evidence can be looked at as a non-repudiation token. A non-repudiation of origin service provides means by the sender of a message to prevent him from denying having sent a message [42]. For example, web metering evidence can be a non-repudiation of origin token that the visitors cannot deny in case they have done the visit[7]. That is, it is the evidence against visitors falsely denying the sent of the requests (the visit event). Depending on the intended application of web metering, this level of non-repudiation can be targeted.

Validation

When evaluating the security of a protocol or a web metering scheme, we can relate to known computational problems e.g. discrete logarithm problem. Similarly, the scheme can depend on pre-image resistance property of a hash function e.g. a webserver given a hash code is unlikely to be able to compute a hash input. In addition, if the scheme requires one entity to encrypt a message using the public key of intended receipt, the adversary in Dolev-Yao model [46, 84] cannot guess the message or private key from the observed message. However, such model assumes that the adversary has control over the open network. Fortunately, there are available tools that can automatically validate security claims e.g. Scyther [40].

[7]On the other hand, a non-repudiation of delivery service is needed when the webserver is motivated to deflate the number of visits [75].

Scyther can be used to check that a particular security property claimed by a web metering scheme is satisfied and detect possible attacks. We start by showing the scheme steps e.g. using a message sequence chart. Such digram can be obtained from the scheme description. Then, we build a model by defining roles of events. As from the web metering description earlier in the section, there are three participating entities in any web metering scheme. Consequently, there can be at most three roles which are the user, the webserver and the audit agency. When Scyther runs, an agent executes the role definition. For example, if the user is sending an encrypted nonce to the webserver using a pre-shared symmetric key, Scyther validates that the claimed secrecy property. Scyther confirms that the claim is reachable if an adversary cannot learn the nonce. Otherwise, Scyther shows the attacks in trace patterns graph. Verification is completed by tracing the scheme and showing occurred events. The produced graph shows the steps an adversary can do in order to carry out the attack or that the claim is reachable.

Besides the encryption example, there are other predefined types in Scyther like nonces or hash functions. Predefined or new types make it easier to map the scheme steps and eventually claim the security property. For example, $Claim(user, Secret, x)$ is a claim in the user role of the secrecy of the value x after possibly send or receive events to other entities. Once the secrecy claim is reachable, it is proved that an adversary cannot impersonate the message sender and increase number of visits. Otherwise, the scheme can be fixed by including a relevant security mechanism (e.g. encryption), so that the scheme is no longer vulnerable. With our assumptions (e.g. denial of service attacks are not considered), we disabled maximum number of bounds in the tests and Scyther still reports no attacks. Along Scyther tests, we also formulated prepositions and achieved properties, and provided proofs and justifications. Both types of evaluation confirm that the security properties can be met.

Transparency.

A visitor transparency [80] property is defined as follows.

Transparency. A web metering scheme is transparent if it executes inside or behind another existing action or property in web interaction so it does not require a new explicit action from the visitor.

Accuracy.

Web metering scheme accuracy is defined as follows.

Accuracy. Web metering scheme accuracy is the level of precision at which the scheme produces in its web metering result and evidence. A further level of the scheme accuracy is granularity, which concerns how detailed the produced result and evidence were.

The overall problem can be abstracted as a charging model for content. The charging model has to capture various properties including security, accuracy and usability. However, when current models are applied today, there are trade-offs among conflicting properties. In this thesis, we propose new web metering schemes addressing these conflicting properties. Our results, of proposed schemes and achieved properties, are summarised in Table 1.1. (To balance simplicity and accuracy of the properties presentation without overloading the terms, integrity notion is reused here instead of security.) One major application for charging for content is online advertising. It is assumed that the content has limited value and, consequently, a lightweight web metering security approach [57] can be followed. Such an approach can still restrict the webserver from faking visits, as the required cost exceeds the gained benefits.

The web metering problem cannot be viewed as an entirely mathematical problem seeking a specific solution using only methods of mathematics simply due to practicality and deployment issues. As a result, the problem can be initially approached as a set of

Table 1.1: Properties Trade-off And Proposed Schemes

	Privacy	Integrity
Accuracy	Conventional Analyser (Chapters 8 & 9)	IDMS + Hash (Chapter 7)
Integrity	Hardware (Chapter 6)	▨

desiderata that require a solution (e.g. using mathematics or technology) to a desired level of satisfaction. From a different perspective, web metering can be viewed as a need that will provide advances for today's web (or Internet). Historically such web development will shape its path by solutions or products that prove themselves in the market and such need's parameters cannot be prespecified due the evolving nature of the web. Besides, the problem is interdisciplinary among cryptography, computer science, psychology, forensic science and web technology.

The following two areas are not considered in this research. Bandwidth Metering research addresses metering the Internet bandwidth usage for pricing purposes, which uses the same web metering terminology. Some Internet Service Providers (ISPs) are interested in bandwidth usage to accordingly bill their customers. It is the opposite concept of flat rate web access where the user pays a fixed amount regardless of their actual web usage [23]. Also, there are some instances when metering some webservers do not constitute a web metering problem in special environments. For example, metering a banking website where the visitor needs credentials issued by a trusted party (i.e. the bank itself) can be easily achieved by current web technology means. On the other hand, metering a newspaper website is a typical scenario[8] here that makes web metering a challenge today.

[8]A scenario is "an imagined or projected sequence of events, especially any of several detailed plans or possibilities" [dictionary.com].

1.3 Research Problems And Contributions

The following are two research questions. The solutions mainly require either a designed balance among identified trade-off (or conflicting) properties or an exploration of a new web interaction property.

1. **Privacy Trade-offs**. Can we design a web metering scheme which can satisfy the granularity requirement (by producing web metering evidence of certain quality) or the integrity requirement, and still satisfy the privacy requirement?

   The basic web metering evidence reveals number of visits with no information about the visits themselves (e.g. time of each visit). However, determining quality of captured visits can be a requirement for some web metering applications where "rich" evidences, containing additional information about visits, are needed. For example, in basic evidence, a casual visitor, who is not interested in the webserver content or services, can increase the number of visits. However, the number of visits should only be increased with "meaningful" visits, captured by more detailed evidence. Consequently, such evidence can reveal number of unique users and their number of visits. Also, this granularity of metered data can help in dispute resolution regarding web metering evidence. However, it is a trade-off, the more information collected about the web interaction, the more likelihood of privacy invasion of visitors. Similarly, having entity authentication for visitors helps significantly in evidence verification (that the captured visitors' messages are authentic). However, entity authentication and privacy can be conflicting requirements.

2. **Transparent Integrity.** Can we design a web metering scheme that can satisfy both visitor transparency and integrity requirements?

   The majority of published web metering schemes (e.g. [93, 62, 10]) assumed special setup which is not needed during a typical visitor-webserver interaction. This resulted in further researching secure web metering schemes (e.g. [17, 21, 113]) which are not transparent to visitors. Such visitor transparency requirement does not require an explicit action from the visitor particularly performed for the web metering scheme. The integrity requirement is the reliability of the web metering operation to produce web metering evidence and the data integrity of evidence. Solving this problem requires finding an environment where already used solutions can be redesigned to produce web metering evidence transparently to the visitor, and still satisfy integrity requirement. Once both requirements can be satisfied, other requirements (like accuracy and privacy) can be further addressed.

Contributions.

This research examines previous and current web metering schemes and proposes alternative and better approaches that can provide privacy-preserving and "usable" schemes. The contributions of this research are to enhance the privacy and the usability of secure web metering schemes, as follows.

- Design and analysis requirements, and a classification

  We provide, in Chapters 2 and 3, a set of security and functional requirements and a classification to address design and analysis issues regarding web metering schemes. The requirements are used to address a scheme in a systematic way as without such requirements, new web metering schemes continue on previous work in an ad hoc way and their unjustified contributions are naturally set without clear definition of purpose. This work defines the environment, states assumptions, identifies threats and classifies existing web metering schemes.

- Privacy-preserving schemes

  We provide, in Chapters 6 and 8, two novel privacy-preserving web metering schemes, with different scenarios. This work addresses the web metering problem in the context of privacy. It explores the dilemma of convincing interested parties of web metering results with sufficient details that can still preserve visitor's privacy. The methodology is to find and properly reuse potential properties and data to achieve the respective requirements and desiderata. Established privacy guidelines are also used to analyse and show how current schemes can invade visitor's privacy, and consequently new schemes (and approaches) can be proposed to enhance that aspect. The analysis along various privacy metrics are used to compare web metering schemes with each other and consequently privacy policies are provided for web metering schemes to comply with. A secure web metering scheme is proposed in Chapter 6, by using an "anonymous" signature scheme, that can provide a proof of visitor secret possession in an enhanced privacy-preserving manner. The other scheme is proposed in Chapter 8 to improve the accuracy of web metering results in an enhanced privacy-preserving manner. Furthermore, scenarios and variations are proposed in Chapters 8 and 9 utilising existing communication protocols to run transparently to the visitor.

- Transparent and accurate schemes with integrity

  We provide, in Chapter 7, two novel web metering schemes, with different scenarios, that run transparently to the visitor (an aspect of usability) and satisfy the integrity requirement. To obtain such novel schemes, the methodology mainly consisted of exploring the solution space for alternative and better approaches than previous ones, to securely provide web metering evidence. In particular, this work researches environments where existing web solutions or properties can be securely redesigned to provide a transparent web metering function. Also, the resulted schemes could possibly integrate previous work for web metering purposes. Two existing solutions are reused here. Also, it explores a possible collaboration among the web metering entities that can be done transparently to the visitor. The start of this research was particularly exploring approaches that can use current authentication and communication protocols to transparently provide web metering evidence. There is one proposed transparent redeemable voucher scheme that uses authentication protocols, and another simpler scheme that uses password hashes solution to provide web metering evidence.

1.4 Publications

Parts of Chapters 2 and 7 were published in the 6th international conference on Information Security and Assurance (ISA 2012) [1]. Parts of Chapters 2 and 7 were published in International Journal of Security and Its Applications [2]. Parts of Chapters 4 and 6 were published in First Workshop on Secure Smart Systems (SSS 2014), SECURECOMM 2014 [4]. Parts of Chapters 5 and 6 were published in EAI Endorsed Transactions on Security and Safety [5]. Parts of Chapters 8 and 9 were published in World Symposium on Computer Applications & Research (WSCAR 2015) [3].

1.5 Thesis Outline

The outline of this document is as follows. Chapter 2 describes threats in the web metering environment and proposes design and analysis requirements from security and functional perspectives. Chapter 3 proposes a classification for web metering schemes. Chapter 4 describes previous web metering schemes and provides gap analysis with regard to the proposed requirements and desiderata. Chapter 5 further researches privacy issues and solutions, and analyses web metering schemes. Chapter 6 proposes a novel privacy-preserving scheme using user hardware. Chapter 7 proposes transparent schemes that securely reuse authentication protocols. Chapters 8 and 9 propose a privacy-preserving scheme with different scenarios, that run transparently to the user and enhances the accuracy of web metering results. Chapter 10 provides a summary and conclusions.

Part I

Web Metering Framework And Privacy

Chapter 2

Threats To And Requirements For Web Metering Schemes

Contents

This chapter first identifies threats in the web metering environment. Then it proposes a set of requirements to address design and analysis issues for web metering schemes. We use threat trees and Dolev-Yao model to derive requirements and desiderata for web metering schemes. Two security requirements are derived to address the web metering *threats and two functional requirements are proposed to address validity and usability of web metering schemes.*

2.1 Introduction

Threat tree analysis[1] is an initial high level analysis to learn more about the security of an application. In Figures 2.1 and 2.2, we provide two simple examples using threat trees to show "what can go wrong" during a "hostile" web metering operation. Such dialogue is essential in problem solving techniques [104] to better understand the real problem (and consequently state any assumptions). This process will then be followed by deriving the requirements and desiderata, that the solutions should satisfy. As the webserver is inherently motivated to change metering evidences for its benefits, the webserver can change the visits criteria by inflating (or deflating) visits, or invade the privacy of visitors, as in Figure 2.1. An adversary can do the same (or infer) statistics about certain webserver e.g. peak hour of visits. However, typically the threat to webserver statistics is of a lower probability and such data is assumed to be public. For each entity, attacks were listed within the framework. Further attacks were then derived with their likelihood in a typical web metering environment. Other examples and scenarios with different assumptions are provided in Appendix B.

On the other hand, a non-motivated visitor may not cooperate to behave according to a web metering scheme presumed operations, as shown in Figure 2.2. The visitor can inflate their visits or any captured visit property, or do the opposite and hide his presence. Typically, the second threat has higher probability than the former, because it is in the visitor interest to hide his visit for their privacy rather than inflate their visit number.

[1]www.schneier.com/paper-attacktrees-ddj-ft.html

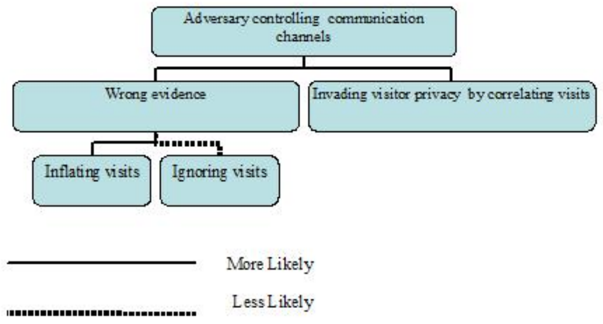

Adversary controlling communication channels
Wrong evidence
Invading visitor privacy by correlating visits
Inflating visits
Ignoring visits
More Likely
Less Likely

Figure 2.1: Threat Tree for Adversary Controlling Channels

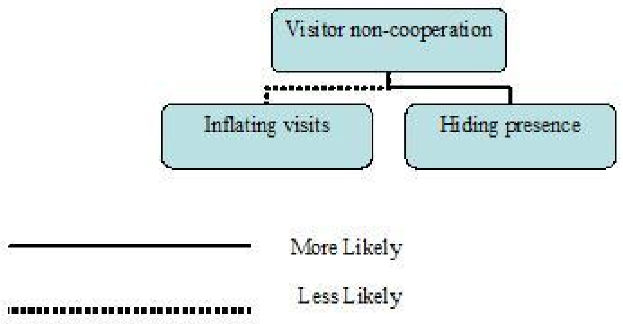


Figure 2.2: Threat Tree for Non-cooperating Visitor

2.2 Threats

We consider here the following threats: threats to web metering scheme (which includes metering operation and metering result), threats to communication channels and threats to visitor privacy. We define here an adversary as a corrupt webserver or any entity that does harm to the web metering scheme. We generally assume that number of corrupt visitors is much smaller than total number of metered visitors [11]. It can be assumed that there are virtually no corrupt visitors in visitor-centric schemes, presented in section 3.2.1, where visitors either control the scheme or it is in their interest to be honest. Also, WMEP is assumed not to be able to take leverage of query space to get private information about a particular visitor by "smartly" querying the audit agency. This assumption can be looked at as a way similar to "securely" querying statistical databases.

- Threats to web metering scheme

  - Threat to metering operation

    This threat regards a corrupt webserver which does not follow the required web metering operations (in the scheme). A corrupt webserver is inherently motivated to change number of visits. The change by webserver can be hit inflation or hit deflation [75]; however, we only consider hit inflation in this document. Also, a corrupt webserver can be motivated to change some metering operations without changing number of visits. For example, a corrupt webserver intentionally changes a webpage identifier, which is going to be recorded in the web metering evidence, to a different webpage that charges higher fees for advertisements. The same attacks can be carried out by a corrupt visitor, whenever applicable.

  - Threat to metering result

    After the web metering operation, evidence can be stored and referenced from webserver, metering provider or audit agency. We consider here evidence stored at webserver (we assume stored evidence at metering provider or audit agency is secure.) Threat to web metering result concerns a corrupt webserver that changes the evidence. For example, a corrupt webserver changes the time of visit included in web metering evidence.

- Threats to communication channels

  Using the Dolev-Yao threat model, an adversary here has control over data in the communication channels with below assumptions. The communication channels here include the following three channels: visitor to webserver channel, visitor to metering provider or audit agency channel, and webserver to metering provider or audit agency channel.

  We assume communication channels to WMEP (from metering provider, audit agency or webserver) are secure (satisfying both integrity and privacy requirements later described in 3.1). Also, we assume that webserver phishing attacks are not possible here, where the adversary is initially able to impersonate the intended webserver, because the visitor initiates the connection by sending a direct request to the intended webserver. Also, we assume that metering provider and audit agency cannot be impersonated (i.e. an adversary cannot send messages as if they originated from either party) as both are equipped with countermeasures outside the scope of this document. Also, we assume that the adversary cannot block communications to a web metering entity (but he still can modify it) and such denial of service attacks are resolved e.g. each entity anticipating a message sends a timeout message until the message is received. As a result, the adversary here is able do any of the following:

  – The adversary can obtain data sent in the communication channels.

- The adversary can become a legitimate entity in the web Metering scheme where he can initiate a connection.

- The adversary can receive data from other entities.

- The adversary can send data to other entities impersonating another entity.

Using the above adversary capabilities and assumptions, a successful attack enables him to execute any of the following three scenarios:

- Replay attack

  A replay attack where an adversary captures data sent from visitor to metering provider, audit agency or webserver and sends the data again. Similarly, an adversary captures data sent from webserver to metering provider (or audit agency) and sends the data again. And if such action is not detected, the visits number is increased.

  For example, an adversary captures a message sent from visitor to metering provider. Then, the adversary sends the same captured message to metering provider. The metering provider checks and accepts the message and increases the number of visits.

- Impersonation attack

  An adversary, who is more powerful here than the first replay attack scenario where attack effect is limited to captured data, creates fake data and sends it to metering provider or audit agency impersonating a valid web metering entity (i.e. webserver or visitor). Or an adversary creates a fake request to a webserver impersonating a valid visitor. If such actions are not detected, the visits number is increased or the data has invalid properties.

For example, an adversary learns the format of messages sent from visitor to metering provider. Then, the adversary creates a message with same observed format for a fake webserver and sends the message to metering provider as if it was originated from the visitor. The metering provider checks and accepts the message and increases the number of visits for the fake webserver.

– Man in the middle attack

  An adversary receives data from visitor or webserver not intended to him and modifies it before forwarding it to the intended party. If such actions are not detected, the visits number is increased or the data has invalid properties.

  For example, an adversary receives a message from visitor intended to metering provider. Then, the adversary changes the message to include a fake webserver identifier and sends the message to metering provider as if it was originated from the visitor. The metering provider checks and accepts the message and increases the number of visits for the fake webserver.

- Threats to visitor privacy

Web metering evidence carries information about an interaction between a visitor and a webserver. We assume the webserver content and services are public and we are not concerned about webserver privacy. A corrupt webserver which has access to web metering evidences can invade visitors privacy by correlating different evidences together. Also, a corrupt webserver can increase the requested information from the visitor side.

Another aspect of visitor privacy is when an adversary (using his capabilities shown in previous point -threats to communication channels-) impersonates a valid visitor and receives replies from metering provider (or audit agency) that contain private data about the visitor. Or the adversary captures (and possibly correlate) data sent in visitor to metering provider (or audit agency) channel or in visitor to webserver channel.

A summary of the assumptions we followed in this thesis are shown below:

- Visitor is inherently not motived to participate in the web metering scheme.
- Limited value of the online content (affecting the designed cost for webserver faking evidence).
- Number of corrupt visitors is far less than the total number of metered visitors.
- Webserver phishing attacks are not possible, and metering provider and audit agency cannot be impersonated.
- Communication to WMEP is "secure" and outside the web metering problem.
- Denial of service attacks are not considered here as they can be detected and minimised using other countermeasures.
- The web metering environments considered are where the visitor privacy is a concern.
- The clocks of all involved entities are synchronised.

Table 2.1: Web Metering Requirements

Security	Integrity	Privacy
Functional	Accuracy	Usability

2.3 Web Metering Requirements

The following are four web metering requirements: integrity, privacy, accuracy and usability. There is a trade-off between some of these web metering requirements; for example, usability can be sacrificed for better integrity. The first two requirements (integrity and privacy) address threats described in section 2.2 and the next two requirements (accuracy and usability) specify how web metering schemes should operate, as summarised in Table 2.1. We use these requirements and desiderata to analyse previous and proposed schemes. Although the presentation of the analysis is not strictly consistent across all schemes, all requirements and desiderata are covered. The ordering and emphasis of the analysis parts are due to their differently achieved properties and contributions.

2.3.1 Integrity

Integrity of web metering is defined as the integrity of the metering process used to produce web metering evidence and the integrity of the evidence. This requirement is needed against intentional (and accidental) actions that can change the web metering evidence. For example, once integrity of web metering is achieved, a corrupt webserver should not be able to construct or change evidence of a visit. We use the terms security and integrity interchangeably when the other non-functional requirement (privacy) is clearly stated.

Integrity of web metering consists of two aspects: reliability of metering process and data integrity, as follows.

- Reliability

  This integrity aspect is concerned with the reliability of the process used for generating web metering evidence. Reliability can be defined as the consistency of measurements which are taken using the same method under normal or hostile conditions on the same subject [76]. That is, if the same environment conditions are met again, the measurements results should be the same. Reliability in web metering is defined as follows.

  Reliability Of Web Metering Scheme. A web metering scheme is reliable if its web metering operations are executed as expected per its specifications and cannot be affected by an adversary to eventually provide consistent results and evidence.

  The reliability requirement is needed as a countermeasure to the threat to web metering operation and threats to communications channels (replay and impersonation attacks) described in section 2.2. In other words, to preserve reliability of a web metering scheme, there have to be no possible malicious that can change the metering process. Particularly, in an unreliable web metering scheme, a corrupt webserver can change its required metering operation to inflate number of visits or maliciously change a characteristic captured in web metering evidence. Once reliability requirement is met, it means that the web metering scheme cannot be changed by an adversary. We do not consider accidental actions that could change the reliability of a web metering scheme e.g. missed legitimate visits through a conditional failure.

- Data integrity

  Data integrity is a property that counters threats to the validity of data [42]. Once this property is satisfied, it provides protection against unauthorised modification or destruction of data. Data integrity in web metering refers to the integrity of stored and transferred evidences and data as follows.

- Evidential integrity

  This type of integrity assures that evidences are kept as they were originally produced and stored. That is, once evidences are generated, they have to maintain their exact state and not be changed (maintaining evidences state includes intentional and accidental changes). Also, this integrity includes stored data that requires post processing work to produce the final web metering evidence. The evidential integrity requirement is needed as a countermeasure to the threat to metering result described in section 2.2.

- Communication integrity

  This integrity aspect is concerned with integrity of the communication channels used for transferring web metering data. Transferred web metering data refers to pieces of data transmitted around different web metering entities that can be used to constitute the web metering evidence. Communicating this data has to be done in a way that if the data is changed en route, the change is going to be detected. Communication integrity requirement is needed as a countermeasure to man in the middle attack described in threats to communications channels described in section 2.2.

2.3.2 Privacy

Unless there is an explicit visitor consent, visitor's privacy and visitor being unidentifiable are requirements to privacy-preserving web metering schemes. This privacy requirement is to stop an adversary from invading the privacy of visitors (as in threats to visitor privacy described in section 2.2) and to prevent a legitimate web metering entity knowing private information about identified visitors. Similarly, *unlinkability* can be an additional requirement where visits cannot be linked.

However, stateful connections are generally required today in Web applications (e.g. shopping cart at a retailer website) which in turn makes unlinkability a difficult requirement and not achievable in some cases. Once privacy (and *anonymity*) are provided, lightweight version of unlinkability can be targeted in web metering (e.g. visits can only be linked for a session of one minute).

All privacy, anonymity and unlinkability are requirements for web metering schemes. Basically, the scheme has to let the visitor have control over his personal information (privacy), and does not identify (anonymity) or link the visitor's actions (unlinkability). Throughout the rest of the document, on a high level we refer to those concepts as privacy and the later details will show what aspect of privacy. Further details are provided in Chapter 5.

2.3.3 Accuracy

Accuracy is a degree of how close web metering evidence is to the actual number of visits. Besides this measurement accuracy aspect, web metering schemes should have universal properties manifested in platform independence, which is the level of scheme dependency on visitors of specific platforms. Platform is generic here, and dependency can refer to lower attributes than the platform; for example, dependency upon certain situations or conditions. In the web metering context, it is the complete coverage of the multiple common anticipated visitors' situations or setups. In essence, "all" visits should be captured regardless of their situations and platforms. This property depends on the audience of the web metering scheme, and no scheme can be completely platform independent due to different web interactions; however, it can address a reasonable spectrum of visits. In case the scheme is applicable to certain environments, all assumed platforms or conditions should be specified. The degree of universality is difficult to define without analysing the scheme in question because prospective visitors with their accessing platform cannot be predefined and generalised to all schemes. Another relevant aspect of platform dependency is the maturity of the underlying techniques used in a web metering scheme, which is the extent of the maturity and "wide acceptance" (with reasonable assumptions) of the used web metering approach.

Web metering evidences can have two properties: the degree of how valid and how detailed they are, as follows.

- Accuracy Of Web Metering Result

  Validity in web metering is defined as whether the web metering scheme "truly" measures the specified web metering interaction. In other words, validity is concerned on whether the scheme is capable of providing accurate web metering results as a result of the accuracy of its operations.

  Accuracy of web metering result is the degree of which the web metering result is close to the actual visit criteria. For example, a web metering result produced using visitors signatures scheme to reveal the number of unique visitors is accurate. However, a web metering result produced using Internet Service Provider, Internet Protocol (IP) address scheme is not accurate compared to the previous signature scheme.

- Granularity Of Web Metering Data

  One further desirable property of accuracy in web metering is how much detailed the produced result is. Granularity is the degree to which the web metering evidences exhibit web metering details. Greater level of web metering details with supporting evidences open up new applications and agreements in the domain of web metering. Also, greater level of web metering details can back up claims regarding the collected web metering evidences and provide more information about the quality of visits. Quality of a visit is a property that can distinguish the value of visits e.g. a visit done by a returning customer to an online retailer versus a visit from a young person from a different geographic location.

2.3.4 Usability

Early web metering schemes in the literature (e.g. using secret sharing schemes) assumed users' involvement, which placed a burden on users[2]. Usability in web metering is concerned with efficiency and visitor transparency aspects. Efficiency in web metering is concerned with carrying out the web metering process with the least amount of work and time. The speed of a web metering scheme is typically affected by the computational and communication resources the scheme uses. We consider here four usability aspects: transparency, computational, communication and storage.

Scalability, which is the capability to extend web metering audience domain at a reasonable cost, is not considered here as a web metering efficiency requirement. Threshold schemes stop the web metering operation after some limit e.g. length of a hash chain or a polynomial degree, when the scheme has to be re-initialised by the trusted third party. Generally, scalability in distributed systems refers to a system's ability to grow [76], and such property was pointed out in an early secret sharing web metering scheme [93] that showed an improvement for unlimited number of time frames. However, we regard here re-initialisation of the scheme as an efficient solution especially as this only applicable to threshold schemes which imposed this limit by design to trap the web-server from inflating the number of visits and limit the effect of unfairly changing the web metering operation to that number. With such re-initialisation overhead, the web metering scheme can still grow and "securely" serve extra visitors than previously anticipated. Despite the ability to have parameters that can be used at the visitor side to

[2]www.sites.google.com/site/yuriyarbitman/Home/on-metering-schemes

re-initialise a new challenge without the trusted third party involvement, we consider such scalability as a “nice to have” property in that context.

Across web metering entities, the overall efficiency of a scheme depends (to different degrees) on the efficiency of all participating entities, starting with the visitor, then the webserver, and finally the metering provider or the audit agency. The metering provider (and audit agency) is assumed to be equipped with a more efficient setup and hardware than existing capabilities of the webserver or the visitor. As a result, for efficiency reasons, metering operations can be outsourced to a metering provider or audit agency. In addition, the webserver is assumed to be equipped with more efficient hardware than the visitor. Computational and communication requirements are applied here to all involved web metering entities, as they can reflect the speed of the scheme. However, a storage requirement is applied to webserver and visitor only, and a transparency requirement is applied to the visitor only. On the entity level, satisfying some of these efficiency aspects can affect the others. For example, if a webserver is using more computational resources to solely execute metering operations, communication resources are minimised since constructing pieces of evidence do not have to be communicated.

- Transparency

  This usability aspect is concerned with schemes that require special new hardware, software or action from the visitor. Such a desirable, transparent scheme does not require the visitor to change his browser “structure” or browsing behaviour to access the webserver, at any time frame, including the scheme initialisation phase. Consequently, the visitor finds it easy to use by not being required to have non-existent hardware or software not needed currently or visionary during a “normal” interaction between the visitor and webserver. In addition, the transparent scheme does not require the visitor to execute an explicit human action (e.g. clicking on a specific button). Luckily, the web metering result can still be provided with such “opaque” schemes for the uninterested visitor. One way of approaching this requirement is by only using current standard software (mostly demonstrated in the web browser) and hardware existing at visitor.

This visitor transparency requirement is one aspect of a broader usability requirement. A usability requirement can be defined as “the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use” [67]. Therefore, usability in web metering can be defined as the extent to which a web metering scheme can be used to produce evidence for number of visits, while accounting for visitor satisfaction.

Visitor satisfaction can be affected by factors other than transparency. The other efficiency requirements are important aspects for visitor satisfaction that can affect usability. For example, a scheme that requires heavy computations on the visitor side becomes not usable because it changes the visit experience and, hence, the visitor satisfaction.

- Computing Resources

  Computing resources used to execute the web metering scheme include Central Processing Unit (CPU), cache memory and main memory. Some examples of work that requires these resources includes summarisation, cross referencing, database lookup or mathematical function execution.

  When the scheme execution starts, metering instructions and relevant data are loaded into main memory or cache memory. Then, CPU fetches the instructions into processor registers for processing. Any wasteful or extended use of these resources can make the scheme computationally inefficient.

- Communication

  This efficiency aspect is concerned with required communication resources during web metering scheme execution. The cost of transporting information from one entity to another includes communication processes (software or procedural) and communication resource (measured in needed or consumed network bandwidth). In one way, communications among web metering entities can be looked at as InterProcess Communication (IPC) in Computing where processes are exchanging data with each other.

  Web metering schemes can be designed in a communication efficient way by minimising number of sent messages or packing messages together. An example of making the most of sent messages is "piggybacking" approach where metering information is added to outgoing requests or responses. Piggybacking has an advantage over separate communication messages because it can minimise the cost of communication processes and used media.

- Storage

  This usability aspect is concerned with the mount of data that has to be stored on the permanent storage (e.g. hard disk drives). There are other storage units (cache memory and main memory) closer to CPU which are covered in previous computational efficiency requirement. A web metering scheme can become inefficient storage wise if it satisfies any of the following two points:

  – The scheme requires considerably high volume of storage that can be an obstacle for some visits. For example, the webserver has to store all visited objects and visitors actions on them.

  – If required amount of stored data increases dramatically with the number of visits. For example, the size of log file supporting web metering evidences doubles by each visit.

Chapter 3

Classification For Web Metering Schemes

Contents

This chapter proposes a classification for web metering schemes. We use a tree structure with certain parameters to classify web metering schemes. There are two main categories based on the availability of the third party during the web metering operation where each has three entity-centric subcategories.

3.1 Introduction

The two main categories for web metering schemes depend on whether a third party is required during the visit. Ultimately, there has to be a trusted party to validate web metering evidence after the visitor-webserver interaction or possibly assist in generating it “online” during the web metering operation. Furthermore, similar schemes are grouped together here based on where the metering operation is centred. That is, web metering schemes are classified based on the party that eventually controls measurement results.

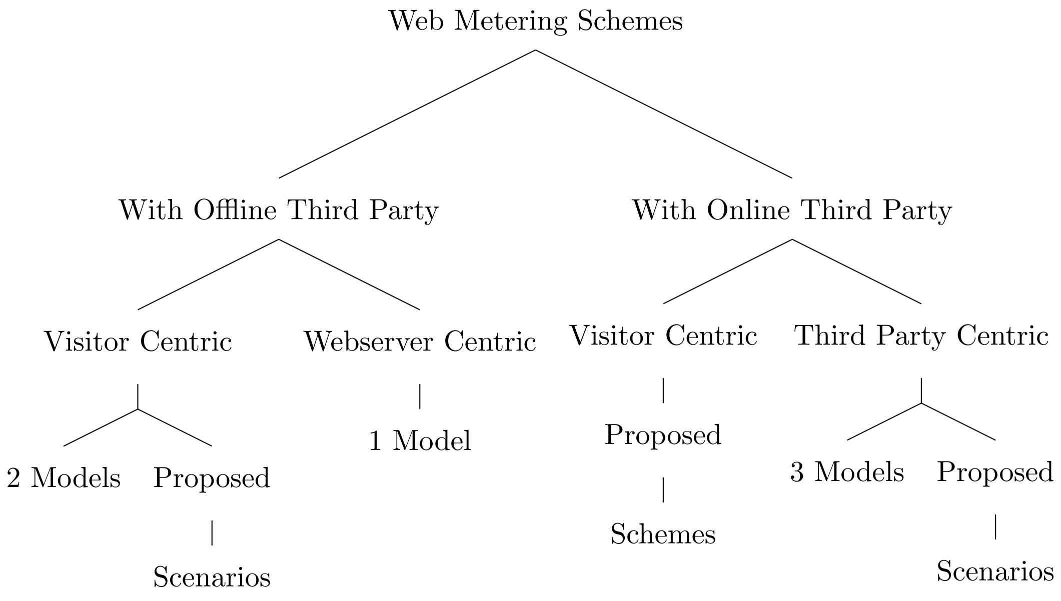
Web metering schemes are grouped into the following: visitor-centric, webserver-centric and third-party-centric. Entity-centric here means one of the following two points:

1. The major part of scheme is controlled by the entity. For example, in a visitor-centric scheme, the visitor possesses a private key which the scheme depends on.
2. It is in the interest of the entity to execute the scheme. For example, in a visitor-centric scheme, it is in the visitor interest to redeem received coupons at webservers and receive free services. This point has to be considered pragmatically, otherwise webserver can be argued to always be the interested party.

The ordering of the above points is important; if the first point is not enough to classify a scheme, the second point is used. For example, in distributed execution where place of control is not clear, a scheme is classified based on the entity that in its or his interest to execute the scheme.

The majority of web metering work is centred around the visitor than the other two parties. One reason of the bias towards visitor-centric schemes is that visitors are generally more trusted to carry out measurements operations compared to webservers. Consequently, the focus of this document is visitor-centric which is evaluated with more depth. Initially, each web metering group is described, starting with visitor-centric, then webserver-centric and last third-party-centric schemes. Then, relevant schemes are described. Last, there is a gap analysis which analyses the group with respect to each requirement.

The following tree shows the number of previous models and our proposals under each category.



Two models are proposed for visitor-centric with offline third party web metering schemes and there are proposed scenarios that follow the two models. Despite the inherent motivation for the webserver to fake web metering evidence, there are few attempts to construct webserver-centric schemes which all follow one model with offline third party. It can be noted that there is no third-party-centric model under offline third party category, as the third party cannot "run" the web metering scheme while being offline.

To the best of our knowledge, there is no previous model under visitor-centric with online third party category and novel schemes are proposed. Similarly, there is no previous webserver-centric model with online third party. Online third-party-centric schemes can address the corrupt webserver problem by shifting the web metering role to the third party. Three models are proposed in this category, and there is a proposed scheme with different scenarios.

3.2 Web Metering Classification

3.2.1 Visitor-Centric

Visitor-centric web metering schemes are ones where the visitor is the controlling or the most interested entity that communicates with both the webserver and a third party resembling the V letter. There has to be three entities in this V-Model: the webserver, the visitor and the third party which can be a metering provider or the audit agency which execute some metering operations. This web metering model is divided into two subcategories: signature and voucher schemes.

The third party does not initially intercept connections to the webserver rather be invoked by either the visitor or possibly the webserver later on the metering process. Then, the third party needs to interact directly with the visitor. Figure 3.1 shows the protocol flow for V-Model.

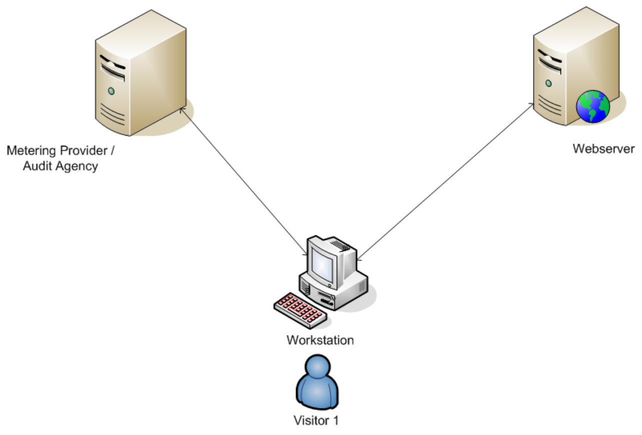


Figure 3.1: Message Flow in V-Model

3.2.1.1 Signature-based Web Metering

Signature-based web metering schemes are schemes where the visitor uses a private key to sign messages, so that a web metering evidence can be constructed that it is bound to the visitor identity (at least one message has to be signed). In such schemes, the visitor signs a message during the web interaction. Then, the webserver collects those signed messages to constitute web metering evidence.

The visitor has to produce a signature on prespecified data. This signature can construct a non-repudiation token for some visit property to webserver. For example, the visitor produces a signature on the time of the interaction (for prespecified data) and sends the signature to the webserver. The webserver can use the signature to claim the occurred visit, which reveals the visitor identity and optionally the time of the interaction.

The visitor has to have a digital signature program and a signing key to compute the required signature. On the other side, the webserver needs a signature verification program and a verification key to check the validity of received signatures.

Figure 3.2 shows the different entities in basic signature-based web metering. First, the visitor connects to the webserver at some point of time and submits a

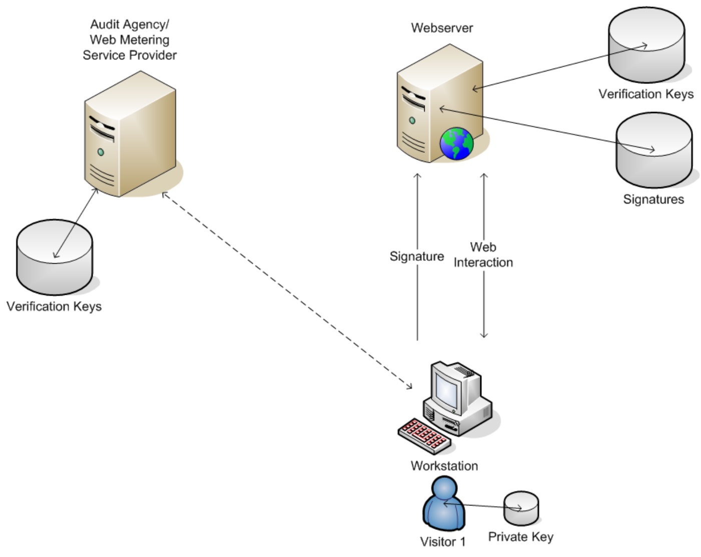


Figure 3.2: Message Flow in Signature-based Model

signature. Then, the webserver checks the signature and stores it before resuming visitor-webserver interaction. And this process is repeated for different time frames and different visitors. As shown, webserver and other entities reference different Verification Keys databases existed at their location. However, Verification Keys database can be a centralised database located at a specific, neutral party.

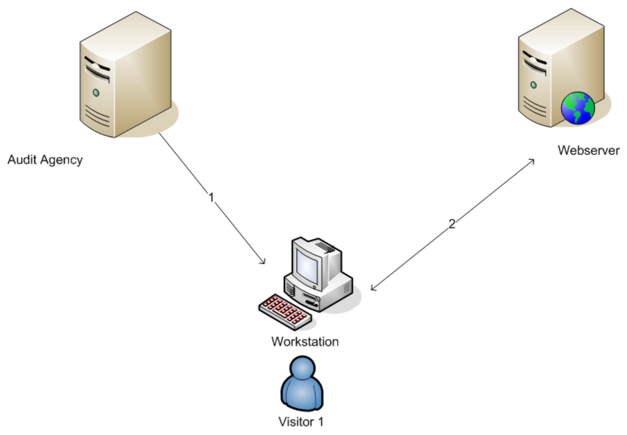


Figure 3.3: Message Flow in Voucher-based Model

3.2.1.2 Voucher-based Visitor Version

Voucher-based web metering schemes are schemes where the audit agency has to distribute vouchers to visitors (or webservers). A voucher is a piece of information that is sent to one entity so that it can be redeemed at another entity. For example, those vouchers can be processed by visitors and the result is sent to the webservers upon their visits (it can be that the visitor simply forwards the voucher to the webserver without any computations). Figure 3.3 shows voucher-based web metering model.

3.2.2 Webserver-Centric

This model layout consists of a visitor, a webserver and a third party which the webserver has to interact with during the metering process. There is no necessary prespecified order of which entity sends the first message. Figure 3.4 shows the protocol flow for webserver-centric web metering. This model is applicable to schemes where the webserver requires additional resources from a third party. Also, this model facilitates implementation as a metering provider is referenced whenever needed without the visitor interference.

Voucher-based model was shown in section 3.2.1.2 and the model can be used from the webserver perspective as follows. The audit agency here distributes coupons to webservers. Webservers forward the received coupons to visitors during the web interaction. Typically, visitors possession of the coupons will serve as web metering evidence and usually the coupons are forwarded back to the audit agency.

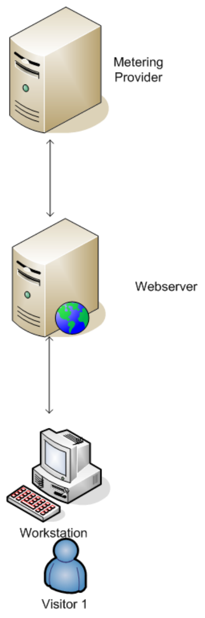


FIGURE 3.4: Message Flow in Webserver-centric Model

3.2.3 Third-Party-Centric

3.2.3.1 Script-based Web Metering

This model layout consists of a visitor, a webserver and a third party (typically a metering provider but can be the audit agency) that controls the metering operations. The webserver uses a script submitted to the visitor to shift the web metering task to a third party. That is, the model works by placing a tracking code into each of the metered pages. This tracking code is run at the visitor platform which calls and then executes a metering component which exists at the third party. The advantage of this model is that metering operations can be outsourced to a metering provider where the script at the webserver references. Figure 3.5 shows the protocol flow for script-based metering.

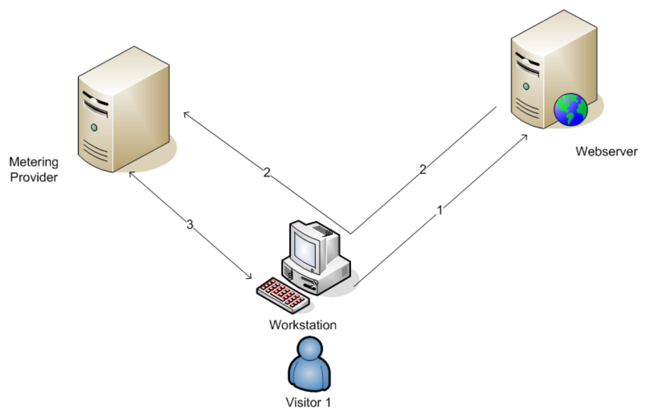


FIGURE 3.5: Message Flow in Script-based Model

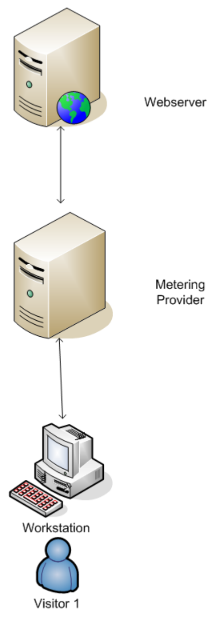


FIGURE 3.6: Message Flow in In-line Model

3.2.3.2 In-line Web Metering

This model layout constitutes of a visitor, a webserver and a third party that controls the metering operations. Here the third party (typically a metering provider) intercepts all connections between the visitor and the webserver. Figure 3.6 shows the protocol flow for In-line metering. This model is applicable to schemes where the metering provider is capable of forcing communications between the visitor and the webserver. This model overcomes the problem of trusting the webserver to refer to the metering provider because the metering provider communicates directly with the visitor.

3.2.3.3 Audit-based Web Metering

In this model, there is an audit agency in a strong position that has control over the webserver, the visitor or both. The audit agency can enforce requirements or policies on the webserver or the visitor. Also, depending on the scheme, the audit agency can capture the web interaction or information at either entities. The difference between visitor-centric or webserver-centric models to this one is the shift of scheme control to the audit agency.

The advantage of audit-based model is that web metering scheme is relatively simple, because it exists at the audit agency that enforces required metering operations at the webserver and the visitor. The extreme end of this model can make the web metering problem trivially solved as both the visitor and the webserver are willing to be controlled by a third party. In such conditions, where an audit agency controls other entities, the environment is special and may not apply to Internet visits. Figure 3.7 shows the protocol flow for audit-based model. The dotted line denotes that the audit agency can access web interaction information.

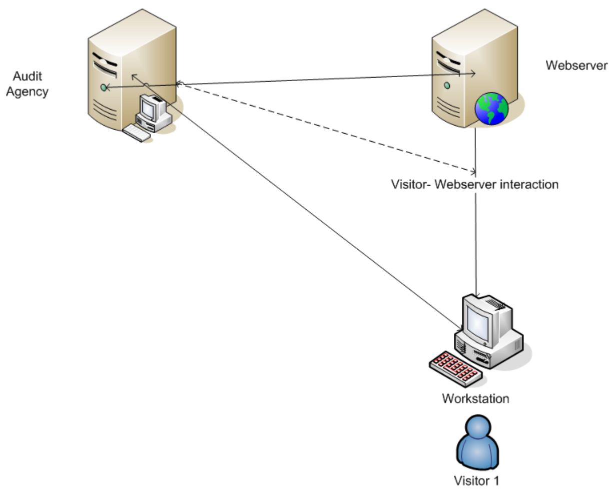


FIGURE 3.7: Message Flow in Audit-based Model

Chapter 4

Previous Work

Contents

This chapter describes in detail and analyses existing web metering schemes. While studying the existing schemes using the requirements and desiderata, we have identified a gap and later proposed solutions. The first part describes signature and voucher-based schemes in which the visitor communicates a signature *or a voucher respectively to the webserver during the interaction. Then, the described web metering schemes are analysed using the web metering requirements. After that, webserver-centric schemes are described and analysed. (Due to the fundamental flaw of webserver-centric schemes regarding the inherent motivation for the webserver to fake visits, no webserver-centric scheme is proposed.) Then, existing third-party-centric web metering schemes are described in detail and analysed under three models: script, inline and audit-based.*

4.1 Offline Third Party User-Centric

4.1.1 Signature-based Schemes

4.1.1.1 Basic Signature-based Scheme

A straightforward signature-based scheme would be the visitor signing the webserver identifier and the time of the interaction as an evidence of a visit. In the scheme setup, each visitor creates a signature key and publishes the verification key. Although this simple web metering scheme was not proposed in the literature or publicly deployed, we believe it can be practically deployed if visitors are already using digital signature programs and visitor privacy is not a concern.

In that scheme, the visitor extracts the webserver identifier upon their initial connection to the webserver. Then, the visitor combines webserver identifier and optionally current time and uses his private key to compute a signature on them. During the visitor-webserver interaction, the visitor sends the signature and his identifier to the webserver. The webserver verifies the received signature and checks that the webserver identifier and the time are correct. If the checks are correct, the webserver allows the visit and stores signature and visitor identifier as a web metering evidence for the current time frame.

After the visitor-webserver interaction, the webserver presents signatures and visitors identifiers for a specific time to interested parties. Interested parties can verify the signatures by using the published visitors verification keys.

4.1.1.2 Signed Hash Chain

A web metering scheme can use digital signatures and hash chaining to construct non-repudiation evidences of visits as proposed by Harn and Lin [62]. To exempt the user from producing a costly signature for each visit, a hash chain is proposed. The scheme is based on Lamport hash chaining technique for password authentication [83].

The scheme works as follows. Let $H(x)$ be a one way function and $H^m(x)$ is the successive application of $H(x)$ m times. Initially, the visitor generates a random number s and computes $H^m(s)$ where m is maximum number of visits and then registers the result at the webserver. Also, the visitor hashes the result with additional information like the webserver identifier and sends a signature of the hash to the webserver.

For the next m visits, the visitor submits the previous corresponding hash input in the hash chain. That is, in the chain (V_0 ... V_m), each value V_i of (V_1 ... V_m), $V_i = H\ (V_{i-1})$. So in the first visit, the visitor submits to the webserver, V_{m-1} which is $H^{m-1}(s)$. The webserver checks that the one way function of received value V_{m-1} is the previous received value $(H^m(s))$. If so, the webserver allows the visit and stores both values, and so on. If the webserver has V_0 and V_m and the signature of V_m, it proves that the webserver has been visited m times.

On the previous two schemes, the visitor used a private key to provide a signature during the web interaction. The webserver checked and stored the received signature, and consequently allowed the visit. The received signature was used to constitute a web metering evidence as it is bound to the visitor identity. The audit agency was involved in the initialisation and verification of the scheme but not during the visitor-webserver interaction. Digital signature schemes and hash functions (and other compression functions like message authentication codes) can be grouped here together, as typically they take variable length input at the visitor side and produce a specified length output.

4.1.2 Voucher-based Schemes

Secret sharing schemes are classified under this category because the visitor receives a secret from the audit agency which can be regarded as a voucher. The webserver here needs to receive a specific number of shares from visitors (threshold) upon their visits to be able to compute a required result. The result of the computation proves the webserver has indeed received the secrets (the visits).

The majority of web metering schemes is based on secret sharing schemes. One of the early published web metering schemes in literature is by Naor and Pinkas [93] which was based on Shamir Threshold secret sharing scheme [111]. In this scheme, the webserver needs to receive a specific number of shares from visitors to be able to compute a required result using Shamir scheme as evidence of the visits. To provide web metering evidence that a webserver has been visited a specific number of times, Secret sharing schemes can be used where its threshold denotes the specific number of visits. Further research continued on Naor and Pinkas work for example, Masucci work on web metering schemes [21, 18, 85, 16, 20, 17, 22].

More formally, Shamir secret sharing works as follows. The secret is divided into n parts where the knowledge of k parts or more is adequate to compute the secret. Let randomly chosen $k-1$ coefficients be a_1, a_2, ... a_{k-1} and a_0=secret. Then, let the polynomial f(x) $=a_0+a_1x+a_2x^2$ ... $a_{k-1}x^{k-1}(mod p)$ where p is a large prime number. After that, n random inputs $(r_1...r_n)$ are fed into the $f(x)$ to give each visitor v a pair $(r_v,\ f(r_v))$. Once the webserver receives k pairs from visitors, the secret, a_0, can be computed using *Lagrange* basis polynomials [14]. The required number of received pairs can be used to reconstruct the secret, by computing $p(x)$, as follows.

$$l_{vi}(x) = \frac{\prod_{j=0}^{k-1}(x - x_j)}{\prod_{i=0}^{k-1}(x_i - x_j)}$$

$$p(x) = \sum_{c=0}^{k-1} l_{vc}(x) f(r_{vc})$$

Also, Asmuth-Bloom threshold scheme [9] can be used as a web metering scheme as in the following two phases: the formation and secret recovery phases [82]. In the secret formation phase for the secret S, the audit agency chooses a prime p and numbers d_1, $d_2..d_n$ which satisfy the following five conditions.

1. For any chosen d_i, $d_1 < d_2 .. < d_n$.
2. For any d_i and d_j, where i does not equal j, the Greatest Common Divisor of d_i and d_j is 1.
3. p must be a prime number i.e. the Greatest Common Divisor of d_i and p is 1 for all i.
4. The following equation holds:

$$\prod_{i=1}^{h} d_i > p \prod_{i=1}^{h-1} d_{n-i+1}$$

5. Prime p is greater than the secret S.

After that, let M be the sum of the product of d_i where i is from 1 to h and let r be a random integer between 0 and $\lfloor (M/p) - 1 \rfloor$. The audit agency computes

S_b which is $S + rp$ and computes s_i for each visitor which is S_b mod d_i and distributes s_i and d_i pair to each visitor i. So that h shares received by the webserver enable it to compute S_b and recover the secret S while $h - 1$ visitors do not have sufficient information to recover S. In Secret Recovery phase, having h pairs (s_{i1},d_{i1}),,(s_{ih},d_{ih}) by the visited webserver, S_b can be computed using Chinese Remainder Theorem of (M1, d_{i1},..,d_{ih}, k_{i1},...,k_{ih}) mod $M1$ where $M1$ is the sum of di up to h threshold. Having calculated S_b, the secret S can be computed by S_b mod p.

Similarly, a distributed signatures scheme was also proposed as an application for web metering [113] where a set of entities can sign a message. The webserver combines the signatures to provide a proof to the audit agency. Such scheme follows secret sharing schemes approach, as the audit agency has to initially set up the scheme with the visitors who then need to provide part of the signature to the webserver.

4.1.3 Gap Analysis

Reliability

In both signature and voucher schemes, reliability of metering process depends on the secret the visitor possesses. And as we assumed there are no corrupt visitors in V-Model, a corrupt webserver is not able to generate web metering evidence fraudulently and consequently the two schemes are reliable in this regard. Also, as secret parameter is kept safely at the visitor side, non-repudiation against visitors denying the visits can be achieved in the two schemes.

However, security of underlying process affects reliability in terms of how hard for an adversary to forge a valid signature in signature schemes without owning the signature and create a valid voucher in voucher schemes without owning the voucher. In such case where there is backdoor for the scheme, the effect in secret sharing schemes is limited to maximum number of visits k. Besides, a typical Secret sharing scheme, where the webserver needs k shares to construct a value, sent shares from a visitor can be captured by an adversary and used fraudulently to construct the required result. Regarding signature schemes, the effect of integrity breach is comparably unlimited until the signature key is revoked.

Webserver identifier has to be included in the signature or the voucher as a countermeasure against an impersonating webserver. Signed hash chain in section 4.1.1.2 satisfies this by hashing the result of $H^m(s)$ and webserver identifier and then sending the signature of the hash value. For voucher schemes, the adversary is unlikely to be able to impersonate a webserver once webserver ID is "hard-coded" in the voucher (e.g. through a signature) and hard to be changed.

For visitor impersonation, the adversary is unlikely to be able to impersonate a visitor in signature schemes because of published verification keys. For voucher schemes, visitor can be impersonated unless webserver recognises and expects the exact format of vouchers. So, in secret sharing schemes, the webserver could end up collecting wrong shares from an adversary. Further details are provided in data integrity analysis.

It is hard for an adversary to have a successful replay attack in signature schemes once real or logical time or a unique value is signed. In signed hash chain, depending on the security of used hash function and assuming visitors do not reuse their hash chains, it is highly unlikely to have a value which hashes to the same value so that an adversary resends it to increase the number of visits. Also, replay attack is not possible in voucher schemes once the voucher includes expiry date or usage trials.

Data integrity

Digital signature in signature schemes provides data integrity for the signed data using authentic visitors verification keys (assuming private key is securely generated). Webserver verifies the received signature and if the signature is valid, communication integrity is preserved, that is there was no possible change for the signed data en route. Also, evidential integrity is preserved if signature verification for the stored signed data is valid.

As a result, man in the middle attack is unlikely to happen in signature schemes because the adversary is unlikely to be able to produce a valid signature on the modified message. In signed hash chain, the visitor signs the hash of $H^m(s)$ and webserver identifier and webserver uses with $H^m(s)$ and s as an evidence for m visits. Depending on the security of used hash function, the adversary is unlikely to be able to find another value (than the intercepted one) that hashes to the previously sent hash value (collision resistance property) and sent it to the webserver before the upcoming message. This attack on changing the hash values messages tends to be meaningless because the attacker changes the messages but not the effect. Also, it is hard for an adversary to find a valid signature of the hash of a modified $H^m(s)$ and webserver identifier.

Secret sharing schemes do not provide evidential or communication integrity. As a result, man in the middle attack is possible in voucher schemes because webserver is not able to check whether received vouchers have been changed en route. However, once the webserver is able to construct the intended secret, communication integrity was achieved. Also, additional actions can be added to some voucher schemes to satisfy communication integrity as follows. When the webserver receives a voucher, the webserver immediately cross references received voucher with issued one at the issuing party.

Privacy

In the initialisation of voucher schemes, there has to be some sort of visitors authentication to audit agency as a countermeasure against the webserver receiving vouchers fraudulently. Also in signature-based, during the scheme operation, the public key of visitor has to be published. As a result, both signature and voucher schemes do not provide privacy for visitors.

Accuracy

- By using standard signature and secret sharing schemes, it is hard for the adversary to forge a valid signature in signature schemes or create a valid voucher in voucher schemes. So, signature schemes produce accurate number of visits and number of unique visitors based on the number of received signatures and the data origin authentication for visitors (provided from the signatures) respectively. Also, voucher schemes provide accurate number of visits based on the number of received vouchers.
- Regarding granularity of metered data, signature schemes can provide number of visits and number of unique visitors and voucher schemes can only provide the number of visits. However, both categories can be designed so that signed data or voucher includes further static (e.g. visitor's operating system) or dynamic (e.g. current time) information.

Efficiency

- Transparency

  Signature schemes require a signature program at the visitor side and can only be considered transparent if the visitor was already using the required signature program prior to the metered interaction. Secret sharing schemes are not transparent to visitor because they require the visitor to submit a share for each visit.

- Computing Resources

  Creating a signature key pair in signature schemes requires computational resources at the visitor side. Also, signature schemes require visitors to use their computational resources to sign required pieces of information for each visit (or communication session). Depending on the used digital signature type, computational resources are accordingly needed (e.g. a random number has to be generated in randomised digital signature). Also, Digital Signature Algorithm (DSA) schemes can be used over RSA-type ones because the former can be optimised for fast signing.

  From the webserver perspective, computing resources are required to verify received signatures. Schemes can use RSA signing algorithm instead of DSA because RSA-type schemes can be optimised for fast verification.

  In secret sharing schemes, the visitor has to execute the shares generator which requires computing resources for the following: randomly choosing $k-1$ coefficients and n inputs, and computing the polynomial $f(x)$ as described in section 4.1.2 (and randomly choosing a_0 and prime number if not prespecified). (It can be that audit agency performs the previous computations for the visitor and securely forwards the shares to him.) Also, secret sharing schemes require shares collection and Lagrange polynomial calculation at webserver side. As a result, many computations at signature and voucher schemes require the use of CPU and main memory resources.

- Communication

  In signature schemes, the verification key has to be securely communicated to webserver (and possibly audit agency or metering provider). During the interaction, an extra piece of information (the signature) has to be affixed to the signed data as done in appendix signature schemes compared to the more bandwidth efficient, message recovery signature schemes. The second scheme, where the message is recovered from the signature, will save communication overhead. So in signed hash chain web metering scheme described in section 4.1.1.2, message recovery scheme is more bandwidth efficient because the visitor has to send both signed data and signature. However, in schemes where signed data does not have to be sent, appendix signature scheme is more bandwidth efficient.

In the initialisation of voucher schemes, vouchers or initialisation parameters (polynomial parameters sent by audit agency to the visitor so he can issue vouchers) have first be delivered to visitors. Then, visitors send a voucher to the webserver during each visit or session. Voucher size is typically limited e.g. a share sent by the visitor in secret sharing schemes should be practically not a large number as not to increase the needed computations at the webserver.

– Storage

In signature schemes, the webserver has to store verification key, signed metered data and signature. Required storage can be reduced if message recovery signature scheme is used as the signature is only stored and metered data can be revealed from it. On the other side, the visitor does not have to store the sent signatures but he has to store his private key.

In secret sharing schemes, all received shares have to be stored at the webserver until the required secret is constructed. The visitor does not need to keep previous sent shares but he might need to store precomputed values for shares generation or shares themselves. Both signature and voucher schemes are considered storage efficient, because the storage requirements do not increase with the number of visits and both schemes are limited to relatively small values typically bound by further required computations.

4.2 Offline Third Party Webserver-Centric

A webserver-centric web metering scheme was proposed in [74], which uses e-coupons as an attempt to map traditional advertisements models into the electronic ones. The web metering evidence is generated if the visitor explicitly passes the received e-coupon (or voucher) from the webserver back to the issuing party. Such schemes can be used when a corrupt webserver is motivated to deflate number of visits because the visitor in this scheme will forward the voucher to audit agency without webserver involvement.

Initially, the audit agency or an interested party combines an agreement text and a webserver identifier. This agreement text contains information about the requested webserver services. Then, an e-coupon is sent to the webserver which includes the previous combination and a signature of them. The webserver checks the validity of the received e-coupon and submits it to visitors upon their visits. After the web interaction, visitors forward this e-coupon to the audit agency or the interested party for redemption (this final step is not shown in Figure 3.4). Those received e-coupons from visitors are checked and if valid, they serve as evidence for visits done to the relevant webserver. Improvements have been proposed in [44] to address these issues in environments where the adversary is motivated to deflate number of visits. However, we only consider hit inflation attacks in this document.

Another category is puzzle-based web metering schemes, which depends on a task solution requiring certain amount of work or beyond the sole capability of the webserver or the visitor. A task is created and can be submitted to either the webserver or the visitor. If the audit agency is the task issuer, it submits the task to the webserver which requires certain amount of visitors resources to provide a solution. If the webserver can provide a solution, it can be regarded as evidence of the visits.

It can be that the webserver is the task issuer once the process of generating tasks are agreed upon with the audit agency. In this case, the webserver issues and submits the task to the visitor, which requires certain amount of resources or capability to provide a solution. If the visitor is able to provide a solution, the webserver can use the solution as an evidence of the existing visitor resources or capability. Such schemes exploit visitors resources unnecessarily (and likely excessively) for a typical visit, and assume that webservers and visitors have limited amount of resources and once they are given tasks, they are only capable of executing them to a certain threshold. That is, the webserver and the visitor have access to known limited resources that can be used to solve issued problems and the task solution can be used to measure the resources used throughout the web interaction.

There is an initialisation phase needed when a task of certain difficulty is created either by the webserver or assigned to the webserver. Then at the beginning of the web interaction, this task is submitted to the visitor which requires a direct solution. The visitor uses his resources to solve the task and sends the solution to the webserver as evidence of the visits.

A webserver-centric *processing-based* scheme was proposed by Chen and Mao [32], which uses computational complexity problems including prime factorisation, presumed difficulty of computational Diffie Hellman and one way hash functions. These computational problems attempt to force the webserver to use the visitors' resources in order to solve them and consequently provide a web metering evidence via the produced result. The scheme can be categorised as webserver-centric as the webserver receives a computational challenge from an audit agency to solve it with the help of visitors. The challenge can be looked at as a voucher that a webserver can redeem once it is able to attach a specific computational value to it.

Transparently to visitor, the scheme uses two algorithms, Timing Algorithm and Auditing Algorithm to provide web metering evidence with the help of processing capabilities that exist at the visitors. The Timing Algorithm is used to increase the cost of inflating the visits criterion by producing the amount of time the visitor spent on the webserver and Auditing Algorithm is used for evidence verification.

The Timing Algorithm, shown in Algorithm 1, is run during the web interaction and produces the amount of time the visitor spent on the webserver, which works as follows. Let $n = p * q$ where p and q are two large prime numbers. Also, x and e are two random integers less than n where e is odd and $h(x)$ is a one way hash function. The output of the Timing Algorithm is (t, a) which satisfies $a = h(x)^b (mod n)$ where $b = 1 + e + e^2 (mod Q(n))$. $Q(n)$ is the least common multiple of $p - 1$ and $q - 1$, that is $Q(n) = lcm(p - 1, q - 1)$. The web metering evidence is (t, a, x, e, n).

This Timing Algorithm will calculate the amount of time the visitor is connected to the webserver. A program can be sent to the visitor once he requested the webpage. Then, the program keeps iterating and updating the values of t and a. Once the visitor leaves the page, the pair (t, a) is sent back to the webserver. Then, the webserver combines (t, a) with (x, e, n) as an evidence of the visit. The Auditing Algorithm is used then for evidence verification, as shown in Algorithm 2.

Algorithm 1 Timing Algorithm

```
procedure ($n$,$x$,$e$)
    $y$=$h(x)$;
    $a$=$y$;
    $t$=1;
  while there is no stop do
    $a$=$ya^e$ $(mod\ n)$;
    $t$=$t$+1;
  end while
    return $(t, a)$
end procedure
```

Algorithm 2 Auditing Algorithm

```
procedure ($t$,$a$,$x$,$e$,$n$)
    $y$=$h(x)$;
    $E$=$e^{t+1}$ $(mod\ Q(n))$;
  if $a^{e-1}$=$y^{E-1}$ $(mod\ n)$ then return $True$
  else
    return $False$
  end if
5: end procedure
```

Also, in this research area [50], Internet visitors need to do computations and be involved in the metering process. That is, the visitor here is required to commit part of his computing resources during the web interaction. An example of such hard but tractable task is extracting square roots modulo a prime number. The web metering evidence here depends on this task difficulty in a way that the task remains feasible to the visitor to compute.

The following is a description of a scheme of extracting square roots modulo a prime number. For a relatively small prime number p and some x, let $F(x) = \sqrt{x} mod(p)$. Solving $F(x)$ requires $logp$ multiplications. If the visitor is able to solve $F(x)$ during his interaction with the webserver, it is an evidence that the visitor carried out $logp$ multiplications. Assuming the visitor computing power is known, those $logp$ multiplications can be translated into the time the visitor spent on the webserver.

There is a variation of other visitor puzzles [120] which can be used for web metering proposes. However, it is not typically the desired way of metering due to the security flaws [89] with Completely Automated Public Turing Test to Tell Computers and Humans Apart (CAPTCHA) [118] and wasted computational power.

4.2.1 Gap Analysis

Reliability

Reliability here depends on both the webserver and the visitor cooperating to forward the voucher back to the issuing party. Also, security of underlying process affects reliability in terms of how hard for an adversary to forge a valid voucher. In case the used vouchers are signed by the issuing party or the computational challenge is hard, it is unlikely for an adversary to be able to produce a valid voucher for a fake visit.

For visitor impersonation, there is an authentication between the visitor and the issuing party, securely done outside the scope of the web metering scheme, which makes it harder for an adversary to impersonate a visitor. It can be assumed that there are no corrupt visitors where they forward their authentication credentials to an adversary. Such attacks can be useful to an adversary whenever visitors, who redeem vouchers, receive benefits from the issuing party. Visitor impersonation is possible in processing-based schemes; however, it is assumed the cost of doing so is larger than the benefit. Also, the webserver is hard to be impersonated as the webserver name is included in the voucher in voucher schemes and it is assumed that the webserver is authenticated to the audit agency in processing-based schemes. However, it is possible for an adversary to have a successful replay attack unless time is secured in the voucher or the voucher has been already redeemed.

Data integrity

Issuing party signature on the voucher, whenever used in voucher schemes, provides data integrity, both evidential and communication integrity, for the voucher. As a result, a man in the middle attack is unlikely to happen because the adversary is unlikely to be able to change a valid voucher. For processing-based schemes, communication integrity between the webserver and the audit agency is satisfied by checking whether received value at audit agency is correct or not. However, man in the middle attack is possible but it is assumed the cost of doing so is larger than the benefit.

Privacy

Privacy cannot be satisfied in webserver-centric schemes because the visitor has to get authenticated to the issuing party for each visit or session upon redeeming the voucher. However, generally privacy is satisfied in processing-based schemes unless the webserver can figure out used resources at the visitor which can lead to their identity. Unlinkability is not satisfied because the visitor is continuously providing his computing resources to the webserver.

Accuracy

– By using a standard signature scheme to sign the vouchers, webserver-centric schemes can provide an accurate number of visits based on the number of redeemed vouchers at the issuing party. Also, the issuing party can check whether a visitor has already accessed the webserver or is a unique visitor when he forwards the voucher. As for processing-based schemes, they can provide accurate results based on the computed result done by the participating visitors. However, such an approach, which requires excessive usage and widespread of such scheme, can trigger issues regarding its effects on the environment [92] and eventually can affect its universality.

– The voucher can include further information about the expected visit time (e.g. hour and date) which satisfies granularity of metering requirement. However, there is no granularity in processing-based schemes.

Efficiency

– Transparency

  Voucher schemes do not run transparently to the visitor because they require a program at the visitor side to accept the vouchers and forward them to the issuing party. However, processing-based schemes can run transparently to the visitor as the visitor does not need to take an action.

– Computing Resources

  The visitor needs computing resources for checking whether a received voucher is valid or not e.g. verifying the voucher signature. Also, computing resources are needed for the visitor to get authenticated to the issuing party upon delivering the voucher. From the webserver perspective, computing resources are required to verify and agree on received vouchers. Also, such resources are needed in case specific vouchers are sent to visitors based on specific criteria e.g. specific time. Regarding the issuing party, initially it has to construct and agree on vouchers with webservers. Also, it has to check received vouchers from visitors against its repository.

For processing-based schemes, computing resources are used both at the webserver and the visitor sides depending on the computations required by the challenge. Also, the audit agency needs computing resources to construct the challenge and check its result.

– Communication

The voucher size should be typically small. The signature can be attached to the voucher as done in appendix signature schemes or recovered from the voucher as in message recovery signature schemes. The second scheme, where the voucher is recovered from the signature, will save communication overhead. Initially, the voucher has to be communicated to webserver. During the visitor-webserver interaction, the webserver sends the voucher to the visitor who typically sends it again to the issuing party. Typically in processing-based schemes, only the computational result, which is relatively small, has to be communicated among the web metering entities.

– Storage

The webserver has to store received vouchers before submitting them to visitors or forward them on-the-fly to visitors. Similarly, the visitor has to store the received vouchers before submitting them to the issuing party or directly forward the received voucher to the issuing party. Processing-based schemes do not require storage requirements as the required information is typically communicated online.

4.3 Online Third-Party-Centric

4.3.1 Script-based

One example of script-based scheme is Google Analytics (GA) [60] which is a service offered by Google to analyse visits to webservers. In order for webservers to use the service, they need to incorporate a JavaScript code in their webpages which is called every time the webpage is visited. Then, web metering evidence and visits analysis are displayed in a GA webpage. During the visitor-webserver interaction, GA runs the script and captures information about the visitor. GA follows script-based model, because a third party (Google) has a web metering component and tracking code has to be placed in each of metered pages.

Pay Per Click (PPC) model requires visitors to make an action of mouse clicking on the displayed advertisements or links as evidence of the visit. Once the visitor click is done, the advertised websites will receive a request and can capture a specific referrer. Another similar technique on assumed visitor's participation is Alexa[1], which requires the visitors to install a toolbar in order to meter the visits. The technique can be classified as script-based, as a third party (e.g. Alexa or Google Adwords[2]) will capture a visit or a click to a particular webserver.

On the other hand, Cost Per Impression (CPI) is a model where the cost of the advertisement depends on the number of the views rather than number of clicks as in PPC. The webserver here is referring to a metering provider during the view evidence generation (e.g. Google Adwords). There are many variations of the model but we consider the following CPI description. Initially, the visitor connects to the intended webserver. Then, an advertisement is displayed to the visitor prior to accessing the intended content and the visitor is required to perform an

[1]www.alexa.com
[2]www.google.co.uk/adwords

action or wait a specified amount of time before accessing the web content. Once either condition is satisfied, the third party will record the advertisement view. This advertisement can be originated by the metering provider and sent by the webserver to the visitor.

4.3.2 In-line

One In-line scheme was proposed in [10] which is based on user tracking by HTTP proxy usage. The HTTP proxy here adds advertisement and relevant code to the passed HTML pages so that the JavaScript can track visitors actions like mouse movements and keyboard strokes. Such scheme requires additional involvement from a metering provider (e.g. Internet Service Provider (ISP)) to capture the required traffic. And the scheme follows In-Line third party model as the HTTP proxy intercepts visitors connections and communicates them back to the webserver.

Initially, the visitor is set to access an HTTP proxy to access webservers. Whenever the visitor connects to a webserver, the HTTP proxy handles the request and passes it to the intended webserver. The webserver processes the visitor request and sends the reply to the HTTP proxy. The HTTP proxy adds tracking code to the reply and passes them to the visitor. The visitor receives the reply and executes the added HTTP proxy code.

4.3.3 Audit-based

One example of audit-based scheme is a closed network where visitors access local webservers in a controlled environment like a campus network. In this scheme, there exists an audit agency capable of monitoring the web interaction and can further have access to either party. A special case of such audit-based scheme is the attempt to only control the webserver side. The use of a physical hardware box attached to the webserver was proposed in [13]. Despite the hardware box being physically located at the webserver, the scheme is classified as an audit-based third-party-centric as the audit agency still remotely controls the box. In such scheme, the webserver connects to an audited hardware box which intercepts visitors' requests and stores a log. Randomly, the box produces a Message Authentication Code (MAC) on a visitor request which is then redirected to the audit agency for an additional verification step. The audit agency verifies the MAC code and the request and if valid, the received request is redirected back the webserver.

Another example of audit-based scheme was proposed in [19]. In this scheme, the visitor has to register with the audit agency. Then, audit agency initialises the scheme by submitting the metering parameters to both the visitor and the webserver. The scheme can also be classified as offline trusted party centric but the involved third party is inherently more powerful here and can intercept the web metering operation.

In the beginning of this scheme, the visitor contacts the audit agency to access a particular webserver for a particular number of visits m. Once the audit agency registers the visitor, the audit agency uses hash chaining method for the web interaction as follows. Let $H(x)$ be a one way function and $H^m(x)$ is the successive application of $H(x)$ m times. The audit agency generates a random number s and computes $H^m(s)$. Then, the audit agency sends $H^m(s)$ and visitor identifier to the webserver. Also, the audit agency sends to the visitor m, s and the visitor registered identifier. The visitor uses m and s to calculate $H^m(s)$.

On the first visit, the visitor submits $H^{m-1}(s)$ to the webserver and for the next m visits, the visitor submits the previous hash value in the hash chain. The webserver uses the last received hash value, number of visits and visitor identifier as web metering evidence. The difference between this scheme and signed hash chain in 5.1.1.2 is that in this scheme, the visitor has to rely on the audit agency.

4.3.4 Gap Analysis

Reliability

Initially, reliability in script-based schemes depends on the webserver “redirecting” the visitor to the metering provider or the audit agency. Then, while the script runs between the third party and the visitor, reliability is concerned of how hard for an adversary to change an operation executed by the script e.g. amount of time a visitor spent on a webpage. Similarly, PPC suffers from corrupted webservers that can launch hit inflation attack to increase their revenue.

For both In-line and audit-based schemes, reliability of metering process depends on the audit agency or the metering provider. Specifically, audit-based schemes are assumed to be run in a controlled environment. A replay and visitor impersonation attacks are possible in In-line and script-based schemes. Also, man in the middle attack can be achievable here as there is no redundancy check between the the visitor and the third party. In In-line schemes, webserver impersonation can be done as the visitor is not authenticated to the intercepting third party. However, it is hard for an adversary to launch a successful webserver impersonation attack in script-based schemes because there is an authentication token between the webserver and the third party.

Data integrity

Data integrity, in third-party-centric schemes, depends on the metering method used by the third party. Whenever the third party is trusted, evidential integrity is satisfied. However, typically communication integrity is not satisfied in both script and In-line schemes unless a redundancy check mechanism is added. On the other hand, it is assumed that the environment is controlled in audit-based schemes where communication integrity can be preserved.

Privacy

Privacy cannot be satisfied in audit-based schemes as the audit agency typically can have access to private information. Also, in In-line schemes, all data sent from the visitor has to go through the audit agency or the metering provider. Regarding, script-based schemes, visitor privacy issues depend on the script operation e.g. GA captures IP address of the visitor which does not satisfy privacy property.

Accuracy

- The accuracy of web metering results here depends on the metering operation performed in third-party-centric schemes. Particularly, third parties in script-based schemes are involved after the initial visit to capture data about the visit or the visitor. GA provides accurate information by capturing IP address of the visitor and tracking him through a cookie. However, Alexa technique depends on visitors who only took the action of installing their tool, which renders the technique not to be an accurate, universal measurement. Similarly, the server side “secure” log approach, described in audit-based model, lacks accuracy as simply the scheme is a collection of “visitors” requests and webserver replies.
- The third party in In-line and audit-based schemes is capable of capturing more than “basic” information about the visitor as all information passes through the metering provider in in-line schemes and the audit agency is assumed to be controlling party in audit-based schemes. Granularity of the script-based schemes

depend on the information captured by the script e.g. GA satisfies granularity as it attempts to capture the visitor location.

Efficiency

- Transparency

  Although all traffic passes through the third party on the way to the webserver in in-line schemes, it is not considered transparent because the visitor has to explicitly set the intercepting third party. Also, typically in audit-based schemes, the visitor has to perform some explicit actions with the audit agency. On the other hand, script-based schemes can run transparently where the visitor is "redirected" to the third party without an explicit action.

- Computing Resources

  Depending on the third-party-centric scheme, computing resources at the visitor and webserver are needed accordingly. Script-based schemes should require more computing resources at the visitor than audit-based schemes because a script is executed at the visitor side. There are no computing resources required for In-line schemes, besides the codes and their replies, as the third party intercepts the visitors' requests.

- Communication

  There are no additional communication messages required at In-line schemes as the normal traffic is captured at the intercepting party. Generally, depending on the audit-based scheme, additional messages have to be communicated to the webserver and the visitor to initialise the scheme. Similarly, depending the executed script at script-based schemes, messages are sent to the metering provider accordingly. Calling and executing such tracking code require the visitor to have separate inbound and outbound connections to the third party.

- Storage

  There is no storage requirements for both In-line and script-based schemes. However depending on the audit-based schemes, there can be storage requirements at the visitor side e.g. storing the hash chain described in [19].

Chapter 5

Web Metering In The Context Of Privacy

Contents

This chapter shows privacy issues with web metering schemes and highlights some findings. We first show privacy problems with web metering schemes and describe different privacy levels. We then describe established privacy guidelines and principles that can be applied to web metering schemes. We compare web metering schemes with each other using some privacy criteria. After that, we provide our observations regarding users' privacy and web metering schemes. Last, we show privacy countermeasures and provide a summary for the used privacy guidelines and policies.

5.1 Introduction

Further to Definition 1, in Chapter 1, the following is a description for two privacy levels that can be used by web metering schemes. Anonymity is an assurance to a subject doing an action that he is not identifiable within a set of subjects [101]. In the web metering context, anonymity is an assurance to the user that he is not identifiable to webserver, metering provider, audit agency, WMEP or adversary. Another stronger level in user's privacy is unlinkability (it can be that anonymity is provided to the user without satisfying unlinkability). Unlinkability of two or more items of interests is that these items of interests are no more and no less related after the attacker observation [101]. In the web metering context, unlinkability of two or more user's visits requires that the observer (including any web metering entity or an adversary) cannot determine if the visits are related or not.

We discuss different aspects of the web metering problem in the context of privacy. We also use privacy guidelines and standards to discuss privacy issues with web metering schemes. We discuss Asia-Pacific Economic Cooperative (APEC) principles, Organisation For Economic Co-operation And Development (OCED) and Code of Fair Information Practises and how they affect web metering schemes. The reason for the chosen privacy frameworks is their influential effect on privacy laws across the globe [115]. These guidelines share many principles e.g. there is Collection Limitation principle in the three guidelines. We start with APEC principles since they are more detailed than OECD and Code of Fair Information Practises. This discussion can be used to formulate privacy requirements to web metering schemes, that is, an aggregate of these principles would constitute a privacy policy for web metering schemes. Besides the privacy-preserving web metering scheme design, there has to be a policy and guidelines regarding the practices of stored results. We refer here to the webserver as the main entity that collects private information but the same applies to metering provider and audit agency.

5.2 Privacy Problems

An adversary can impersonate a valid user and can receive replies from the audit agency that contain private data about the user. Or the adversary can capture (and possibly correlate) data sent from the user. On the other hand, a corrupt webserver can store non-private data that could be correlated at a later stage and invade the user privacy. Also, a corrupt webserver can increase the requested information from the user side and receive private data. Further discussions are provided in section 2.3.2.

Furthermore, balancing the users' privacy right with the conflicting [28] webservers' and interested parties' freedom of expression right, complicates this interdisciplinary problem for privacy-preserving web metering schemes. Without closer analysis and specific metrics, service providers could pragmatically argue that information about visits to the webservers can be published as an exercise of their right of freedom of expression and for public interest.

There are two cases dealing with private information in web metering context. The first web metering scheme stores private information about users for a later operation (the stored private information is part of the web metering evidence or the private information is used to further generate the evidence). The second type uses the private information (on-the-fly) during the metering interaction without storing them to produce richer evidence based on the used private information. Depending on the scheme, we believe that operating on private data to produce processed information (second type) is still a privacy concern because used private data is at risk of being captured by an adversary. However, the quality of the evidence produced and existing security measures can further be analysed to give better decisions regarding this privacy aspect.

Besides, differentiating between unique users and returning ones poses a privacy concern. This is due to the fact that users were classified returning ones after being technically tracked or possibly identified by the webserver. Schemes capable of differentiating among their users should state so and how this can be achieved. This knowledge capability can be suppressed by linkability property in web metering schemes.

Authentication, in web metering schemes, can help in accountability of visits but it can contradict with privacy property. As a result, schemes may tend not to authenticate their users and preserve their privacy. Such schemes are vulnerable to threats that can make an entity impersonates another trusted entity affecting the integrity of the web metering scheme. In an unauthenticated web metering, fake visits by an adversary can be launched to inflate the number of visits which affects the integrity of the scheme. Also, any collected evidences of visits cannot serve as non-repudiation tokens.

There has to be some sort of authentication for users in secret sharing web metering schemes, described in 5.1.2, to stop an adversary from inflating number of visits. This authentication requirement cannot be done without revealing information about the user identity so that the webserver cannot impersonate a valid user and have the required shares. In case metering provider is generating the shares, the user needs only to be authenticated to the metering provider to receive the shares. The metering provider is trusted to execute metering operations (generating and sending the shares to the authenticated user). However, a questionable metering provider has to be trusted as well not to collude with the webserver to link user identity with visits after the user-webserver interaction. A possible approach to solve such authentication problem would be using pseudonymity, which is the use of pseudonyms as identifiers [101] with various forms to enhance users privacy in these web metering schemes. Similarly, an adversary can observe and correlate user authentication data with the visits. Also, the users identities have to be revealed to audit agency to solve dispute about collected shares by the webserver which can potentially be linked to the visits.

Also, HTTP proxy scheme, in section 4.3.2, adds tracking code to the requested HTML pages that track users' actions (like mouse movements and keyboard strokes). Consequently, all visits (including users' requests) have to go through the intercepting HTTP proxy which does not preserve users' privacy. Similarly, Google Analytics (GA), described in section 4.3.1, is one current web metering scheme which can satisfy the granularity requirement by providing more information than just number of visits. However, GA does not provide privacy for users because it captures Internet Protocol (IP) address of users to provide geographic results and uses cookies to track them. To preserve users' privacy, Google provides a "hotfix" plugin[1] to users which once explicitly installed, GA does not work (and collect private data). Such tracking and data

[1]www.googlepublicpolicy.blogspot.co.uk/2010/05/google-analytics-more-choice-for-users.html

collection could raise other problems (like secondary privacy damage [79]) as data collected about some users can affect other users' privacy. Also, an opt out option has the potential to make its user to stand out, as in anti-tracking observation in section 5.4.2.

The following three privacy classes [115] interrelate with privacy in web metering: *information privacy*, *territorial privacy* and *communications privacy*. Information privacy is concerned with collecting and handling personal information. This class is the most affected with web metering schemes as they collect information about the user in order to formulate web metering evidence. Territorial privacy is concerned with the ability to invade into an individual's environment which can be affected here by capturing user IP address and consequently knowing his location. Communications privacy is concerned with the means of an individual correspondence (e.g. email) which also can be captured by web metering schemes as well. The described classes show the different private data that web metering schemes are potentially capable of obtaining.

Besides the privacy problems, the webserver (and audit agency) has to manage the following four trade-offs privacy risks [115]. The first two privacy risks can result in tagging the webserver as a corrupt.

1. Legal Compliance

   The webserver has to comply with the laws and commitments set in its privacy policy for web metering activities.

2. Reputation

   The webserver should protect its reputation by taking considerable security measures against attackers intending harm to the web metering operation or evidence and respecting its privacy-preserving web metering policy.

3. Investment

   The webserver has to cover the costs for the required privacy-preserving web metering operation that will typically be cost effective investment for web metering applications like online advertising. Also, the webserver has to plan an investment strategy for the aggregate web metering results.

4. Reticence

   The webserver has to use the web metering results properly to compete with other webservers. For example, a webserver can leverage an advertised minimal captured information for web metering purposes and back it up with evidence without disclosing potential web metering applications.

Last, the following are four potential categories for privacy solutions [39, 105].

1. In this category, the privacy solution guarantees that the web interaction cannot be linked to user's identifying information e.g. user's location through his IP address. An example of such solution is TOR network countermeasure described in section 5.5.
2. The solution here filters deliberately revealed private information e.g. user's email address. We find such category of solutions the least interesting as the user can control his submitted data.
3. The solution in this category creates pseudonyms that are not related to the user.
4. A policy based solution here reveals information based on the user's preferences e.g. using *P3P policy* in browsers.

5.3 Privacy Guidelines and Standards

5.3.1 APEC Privacy Principles

APEC Privacy Subgroup was established in February 2003 to develop a framework for privacy practises [8]. As we are only concerned with potentially private information about the user, we are not differentiating between personal (private) information and publicly available information as followed in APEC framework. The following are nine principles proposed in the APEC framework.

1. Preventing Harm

   This principle is concerned with the misuse of private information that can lead to harm to individuals. Also, used measures to counter harm should be according to the likelihood and severity of the harm.

   In the web metering context, an adversary should be prevented from misusing information collected, used or transferred by web metering schemes. Also, there should be measures to counter these threats relative to likelihood and outcome of the harm.

2. Notice

   Private information collectors should provide clear notice regarding the private information. Notice statements should include that private data is being actually collected, the purpose of collection, entities that can access these information, means of contacting private information collector and means for individuals to request limitations, access or correction. This notice should be provided in a reasonably time manner and may not be required in case the "private" information is publicly available.

In the web metering context, webservers have to clearly show to users details about collected private information when a web metering scheme is in place. The notice includes declaration of using a web metering scheme, the purpose of the scheme, potential WMEP(s) and metering providers/audit agency (if any), means of contacting webserver about the scheme and to request limitations, access or correction of users private information.

3. Collection Limitation

   This principle states that the collection of information has be relevant to the purposes of collection. In web metering, there has to be measures to limit schemes (and the adversary) to only collect information relevant to constructing an accurate web metering evidence.

4. Uses of Personal Information

   Private information has only to be collected for the purposes of collection except the individual consent on doing so or information required to provide a service to the individual or requested by a "superior" trusted party. In the web metering context, webservers have only to collect information for the purpose of constructing an accurate web metering evidence. However, the webserver can collect more information, if it received a consent from a user to do so or the additional information required to provide a requested service by the user or a superior trusted party requested to collect the information.

5. Choice

   Private information collectors should provide individuals, wherever appropriate, the ability to make choice regarding collection, usage and disclosure of private information. In the web metering context, webservers have to notify users about their available choices regarding private information collection, usage and disclosure and how to achieve that.

6. Integrity of Personal Information

   Private information collected should be accurate, complete and up to date. Web metering schemes should collect, use and report accurate web metering data and should keep track of the collection time.

7. Security Safeguards

   Private information collectors should use safeguards to protect the private information in proportion to likelihood and outcome of the threats. In the web metering context, web metering schemes should have security measures against privacy attacks by an adversary relative to likelihood and outcome of the harm.

8. Access and Correction

   Individuals should be able to confirm whether private information collector holds their private information. If so, individuals should be able to access the information after a reasonable time from the access request, with a non excessive charge (if any), using a reasonable method and in a format easy to understand. Individuals should be able to challenge the accuracy of the accessed information to modify it or delete it, whenever possible. This ability to access and to correct is not granted if the expense is not reasonable or not in proportion to the individual privacy risk, information cannot be disclosed for legal or security reasons or when other individuals privacy is invaded. If an individual is refused access or correction, he should be provided with the reasons and how to challenge the decision.

   In the web metering context, users should be able to confirm with webservers if they hold private information about them. Once confirmed, users should be able to access that information in a timely manner, with a non excessive charge (if any), using a reasonable or a normal method (e.g. HTTP GET) and in a format easy to understand. Also, users should be able to challenge the accuracy of the metering information and be able to modify it or delete it, whenever possible.

Webservers can deny such access (or correction) if the expense is not reasonable (or not in proportion to the user privacy risk), information cannot be disclosed for legal or security reasons or when other users privacy is invaded. If so, webservers should provide the users with the reasons and how to challenge the decision.

9. Accountability

   Private information collectors should be accountable for complying with these APEC principles. Whenever private information collectors send private information, they should take reasonable steps and do due diligence to protect the information at the recipient. In the web metering context, webserver is accountable for complying with these APEC principles. Also, the webserver has to take reasonable steps (e.g. an agreement) to ensure that results at WMEP are protected.

5.3.2 OECD, Code of Fair Information And Directive 95/46/EC

In this section, we first describe OECD principles and then discuss Code of Fair Information Practises from a web metering perspective. OECD guidelines on the protection of privacy include eight basic principles [97], which can be used for as an auditing policy and measurements of web metering schemes as follows.

- Collection Limitation Principle

  This principle is concerned with the limit of private data collection, and consent or knowledge from the user for doing so.

- Data Quality Principle

  This principle is concerned with the relevancy of collected data by the web metering scheme and how accurate this procedure is.

- Purpose Limitation Principle

  The purpose of private data collection has to be specified prior to the collection action and subsequent usage has to stick to the purpose.

- Use Limitation Principle

  Collected data has to be used with respect to specified purposes (as in Purpose Limitation Principle) unless agreed by the user or the audit agency.

- Security Safeguards Principle

  This principle is concerned with the security safeguards used by the web metering scheme against risks such as loss or unauthorised access.

- Openness Principle

  There has to be a policy of openness at the webserver about developments, practices and policies regarding the collected private data.

- Individual Participation

  The user has the right for the following:

  1. Obtain a confirmation from the webserver whether it has private data about the user.
  2. If there is a confirmation of private data existence, the private data have to be available to the user in reasonable time, charge and manner.
  3. If 1 or 2 is denied by the webserver, reasons have to be stated and the user is able challenge the denial.
  4. If the challenge is successful, private data about the user has to be modified, erased or completed.

- Accountability Principle

  The webserver has to be accountable to comply with the measures above.

Code of Fair Information Principles

Most of Code of Fair Information Principles cover the following [52, 30]:

1. Openness

   The webserver should publicly declare to users the use of the web metering scheme and the collected data.

2. Individual Participation

   Users have the right to view all web metering results and be able to launch a dispute to modify or remove the results in case they are not accurate. This users' right is crucial as it affects the accuracy of the used web metering scheme.

3. Collection Limitation

   Web metering schemes should be limited to only collect necessary information (either approved by or known to the user) to construct a valid web metering evidence. Consequently, such measures will limit an adversary from collecting additional information about the user. Also, web metering schemes should dispose private information after an agreed time. For example, if a webserver states that private information is only collected during the visit, the webserver should delete the collected information once user leaves the webserver.

4. Data Quality

   Web metering schemes have to only collect information that is relevant to constructing an accurate web metering evidence.

5. Finality

   After web metering operation, schemes should be limited to only submit relevant web metering evidence to WMEP in the same way the private information was collected in principle 3 (Collection Limitation).

6. Security

   Web metering schemes have to have security measures to stand privacy attacks from an adversary and to prevent accidental actions that can lead to disclose private information (e.g. loss of private user information).

7. Accountability

   Webservers have to be accountable to comply with the above six privacy principles.

EU Data Protection Directive (95/46/EC) [115]

Directive 95/46/EC is a law to protect privacy of European Union citizens. It shares many principles of the earlier Code of Fair Information Principles described above. Relevant European Directive principles in web metering context are as follows.

- Legitimate purpose and fair processing

  Users' private data may be obtained explicitly for only web metering purposes. Also, users have to be fairly informed of any ongoing web metering operation.

- Notice

  Besides the fair processing requirement, involved users have to be "adequately informed".

- Data quality

  Obtained users' data have to be accurate, and updated whenever applicable.

After the Directive 95/46/EC has been passed in 1995, other updated directives followed e.g. Privacy and Electronic Communication Directive (2002/58/EC) [115]. This *E-Privacy Directive* is more focused towards online marketing practices. The following are the new introduced principles.

- Users can easily opt-out from receiving any online advertisements.

- Cookies are used more transparently with the option of rejecting them.
- Information about any activity of publishing users' data is clearly provided with the option of rejecting it.
- Users are notified of any additional *value-added* online services and asked for their permission.

5.3.3 Digital Advertising Alliance Guideline for Behaviour Advertising

Online behaviour advertising is an advertising method that attempts to learn users' interests through their "behaviour" on webserver(s). Recent effort has been made to address user's privacy expression and disputes particularly towards online tracking for advertising purposes. Digital Advertising Alliance program is supported by many "influential" online advertising entities. Besides outlined principles, the program encourages webservers to display easily recognisable specific icons to notify their users of the existing web metering and advertising scheme. The following are seven principles for online behaviour advertising [7].

1. Education.

   There should be entities (e.g. audit agency or visited webserver) that educate users and third parties about existing online behaviour advertising. This principle will show users' options and ensure relevant material are available.

2. Transparency[2].

[2]This principle should not be confused with transparency property in section 2.3.4.

Data collection and uses must be clearly published. This principle will show how data is collected and used by the webserver or other parties. In particular, personally identifiable information and easy to use users' choices mechanism should be clearly described.

3. User Control.

   This principle will provide mechanisms that enable the user to control their data collection and uses. Any users' data that are not agreed on must not be collected or used.

4. Data Security.

   There must be data security for collected and used data. This should include anonymising personally identifiable information and not disclosing used vulnerable (e.g. reversible) anonymising mechanisms. Also, there must be a timeout period for collected data, conforming to reasonable business needs or applicable laws.

5. Material Changes.

   Any changes to collected data or uses must be published and consent has to be sought in case data or uses practises are changed in less privacy-preserving way.

6. Sensitive Data.

   Different data collected about users has to be classified differently (e.g. financial and health data are likely to be more private than visited webservers). Consequently, different data classification requires different practises.

7. Accountability.

   Policies must be developed so that involved entities are accountable for users' data. Mechanisms must also be developed to ensure that the policies are being followed.

5.3.4 P3P Categories

The World Wide Web Consortium (W3C) Platform for Privacy Preferences Project (P3P) [37] provides a framework regarding privacy issues in accessing websites by allowing them to express their privacy practises in a standard format. P3P is an attempt to classify various web interaction data based on the users preferences for different privacy levels. We used the different user's data categories without involving the users or using them as a policy in users' browsers. We have analysed web metering schemes according to relevant metrics described in P3P. A summary of an analysis of current schemes is shown in Table 5.1.

From two extremes, a particular private information can be either *required* by the scheme or *protected*. We use the symbol ✗to denote that the scheme cannot operate without the corresponding required private information in order to provide web metering result or evidence. On the other hand, we use the symbol ✔to denote that the private information can be protected and not accessed by the adversary under secure user setup. Such setup can be achieved with countermeasures that can prevent the adversary from getting the private information. The countermeasures can be provided by the scheme itself or can be potentially provided by other techniques. An example of outside countermeasures that can prevent the adversary from getting protected information could be a user behind a firewall with anonymous browser settings. Further details are provided in section 5.5. We use the symbol $\nmid$ to denote that the private information is not always or necessarily required by the web metering scheme; however, it is *available* and can still be captured by the adversary. Such available information can still be captured due to an efficient implementation (or a variation) of the scheme.

The following are the relevant P3P categories.

TABLE 5.1: Privacy Comparison

Scheme	**Identifiers**	State	Interactive	Location	Computer	Navigation
Digital Signature [62]	✗	∤	✓	✓	✓	✓
Secret Sharing [93]	✗	∤	✓	✓	✓	✓
Webserver Voucher [74]	✗	∤	✓	✓	✓	✓
Processing-based [32]	✓	✗	✓	✓	∤	∤
Webserver Hardware [13]	✓	∤	✗	✓	✓	✓
HTTP Proxy [10]	∤	∤	∤	∤	✓	∤
Google Analytics [60]	∤	✗	✓	∤	∤	∤

- **Identifiers** are issued to users by a third party, which can be the Government or generally a "trusted" third party, to identify the user e.g. a username. There can be various levels of identifiers. For example, it is reasonable not to consider the action of capturing an IP address as identifying as authenticating an audited hardware decoder. Each user has a private and public key pair that is used to produce and verify the signature in signature-based schemes. Also, GA assigns an identifier to the user browser after capturing user's IP address. In HTTP proxy, all visits have to go through the proxy, which does not preserve users' privacy. Particularly, unencrypted traffic that goes through HTTP proxy, including identifiers, can be captured. In secret sharing schemes, secrets, submitted by the user, cannot be used as identifiers; however, users may have to be authenticated to get or verify the secrets. Also, users, in webserver voucher schemes, have to be authenticated to ensure that the webservers are not forwarding the coupons themselves.

- **State Management Mechanisms** are used to maintain the state of the connection to the webserver e.g. cookies. If the state of the user is required or can be captured, unlinkability cannot be provided by the scheme. In signature-based schemes, the user continuously submits a signature (or a hash value) that can link his visits. In typical secret sharing schemes, each user continuously submits a share to the webserver for each visit (or session) which link them up until the threshold value is computed. Similarly, in webserver voucher schemes, the state of the user can be tracked as the user frequently forwards the e-coupons. The state of the user is continuously tracked in processing-based schemes because of the required participation from the user side. Also, user state can be figured out in HTTP Proxy schemes as all traffic goes through the proxy. Also, GA uses cookies to track the user state.

- **Interactive Data** includes data generated on-the-fly during user-webserver interaction e.g. a query to the webserver. Also, in webserver-centric hardware schemes, users' queries are occasionally captured and *MACed* before the result is forwarded to the audit agency. HTTP Proxy schemes can capture interactive data from the user as all traffic goes through the HTTP proxy.
- **Location Data** category covers information regarding the users location e.g. users' GPS location. Location of users is not directly captured in HTTP Proxy schemes, however, depending upon the location of the HTTP Proxy, location of users can be tracked. Also, GA, by default, captures IP address (in full or part) of the user, which can provide information about the user location.
- **Computer Information** is any information about the user computer e.g. IP address. Processing-based schemes and GA can capture computer information by analysing the time needed to return the solution (e.g. estimated CPU speed during the visit) and capturing the IP address respectively.
- **Navigation and Click-stream Data** covers data about the user browsing behaviour e.g. user clickstream. Depending on the implementation of the processing-based scheme, navigation data can be required (e.g. user presence is determined by mouse movements). Also, HTTP proxy returns to the user a tracking code that captures the user mouse movement. Also, GA captures navigation data about the user through the type of referral which is stored in the cookie.

It can be observed from the analysis summary that the categories identifiers and state are the least satisfied privacy categories. In particular, all schemes require or can capture the state of the user. Furthermore, once a user is tracked, other private information can be captured. For example, user's preferences can be captured from the webserver content or some navigational data. Such attack is when user's preferences are involuntarily learned by the adversary. If a scheme was able to capture the users' state but was not able to identify the user (or derive his preferences), the captured state alone is not considered a privacy concern. Such type of potentially private data can get more subtle. For example, if a user is accessing the webserver using *Internet Explorer* browser, derived data could be that the user prefers Microsoft-based products.

Identifier information is used to achieve security requirements; however, such authentication information inherently conflicts with privacy. Identifier information can be the determinant metric to provide a privacy-preserving scheme. Processing and webserver hardware-based schemes are the most satisfying privacy-preserving schemes as they do not require nor can capture users' identifiers.

Time is important in web metering to present results with respect to time but the continuity and the linkability of the visits (i.e. state) is an additional desired property. However, if a webserver was able to only capture the user state but was not able to identity him, the captured state alone may not be considered as a privacy concern. The same can be applied to navigation and interactive data e.g. capturing interactive data about a user and not being able to identify him. We provide improvements in that regard in Chapters 8 and 9. Also, following the unlinkability property described in Chapter 2, we can consider that capturing the user state alone is not a "major" privacy concern under some applications e.g. checking whether the user has spent enough time to properly view a webpage.

To further investigate that issue (and from its extreme extent), we follow here the definition of personal information described in the Personal Information Protection Act (Law No. 57 of 2003 which became effective on April 1, 2005) as shown below.

Personal Information. "Personal information" means the information about a living individual which can identify the specific individual by name, date of birth or other description contained in such information (including such information as will allow easy reference to other information and will thereby enable the identification of the specific individual).

From personal information definition, identifiers category is the only one which is considered direct personal information. Care must be taken not to allow other "non-personal" yet private information to eventually identify the user. For example, MAC address (as computer data category) could eventually lead to user identification (and tracking). We took that subtle issue into consideration during designing our proposed solutions so that potential identifiers are never used (compared to other categories like navigation data).

5.4 Privacy Observations

5.4.1 Engineering Privacy Setup

Despite the desired web metering granularity and adversary attacks, the concept of using privacy as an economic rationality [48] can be applied in the context of web metering. That is, web metering evidence can be generated by trading services to the user in exchange for information. This approach inherently has a limited scope because it assumes users wish to participate in the web metering scheme and therefore, it still has questionable efficiency. However, when such benefits outweigh the risks, users tend to accept and adjust to metered interactions [61]; such an approach requires a web metering privacy policy for webservers to be able to fairly trade their services. Consequently, the following observation holds.

Users' Benefits Tendency Observation. Users can adjust to new settings while interacting with the webserver if privacy benefits are proven to them.

Particularly, using User-Managed Access (UMA) [64], the user can trade its specific information available at different webservers with the webserver currently visiting. For example, a user visiting an academic blog, the blog asks the user to share university album in order to access the blog content or services. Getting the user information could be designed in new products so the activity happens transparently to the user.

5.4.2 Tracking

We generically define the activity of tracking a user as follows:

User Tracking. User tracking is a set of actions that enable an entity to associate users' visits over a period of time that exceeds a prespecified session.

Further to guidelines and policies described in section 5.3 particularly Digital Advertising Alliance program, user tracking can be the default setup during user-webserver interaction [86]. The audit agency has to be involved in tracking to ensure users' privacy is preserved and to counter corrupt webservers faking evidence.

Following anti-fingerprinting discussion in [51], we further state the following generic anti-tracking observation.

Anti-Tracking Observation. Across a large set of random users, a user who intentionally changes his captured fingerprints as a deterrent to tracking is highly likely to be distinguished from the rest of the set and potentially become unique and more trackable.

Rationale. Let n be number of users with each leaving m fingerprints, where m is a large number. A user deploying an anti-tracking scheme will intentionally generate "different" fingerprints up to all of m. As n increases, the anti-tracking scheme presence is more likely to be spotted and user's fingerprints matching another user likelihood decreases.

With such thought experiment, one can infer the increased chances for some anti-tracking schemes to generate unique fingerprints. However, it has to be hard for an adversary to track a user by randomly changing the unique fingerprints. Consequently, a web metering scheme can still be designed to address the dilemma of producing accurate web metering results in a privacy-preserving manner.

5.4.3 User's IP Address

IP addresses can be used to figure out users' physical locations. Techniques that can figure out users' locations, like *round-trip time (RTT)*, *Domain Name Server (DNS) hostname*, *traceroute* and others [98, 91], can be used by an adversary. For example, a traceroute to a user's IP address could reveal routing hosts near the user's physical location. (As a countermeasure, an Onion Network privacy solution is described in section 5.5.2.) There are even accessible third party location services that use various techniques to map IP addresses into the location. By experimenting with free PHP tools like *IP2Location*[3], we were able to freely get representatively accurate location data for few tests from different locations and devices. The proximity of the location results in UK were accurate in most

[3]www.IP2Location.com

cases and the service provider keeps on maintaining a database of IPs versus their physical locations. Consequently, the following observation holds.

Physical Users' Locations Observation. Despite captured IP addresses do not inherently contain geographical location, it is highly likely for an adversary to be able to successfully figure out physical locations of “common” visiting users, with low costs.

Elsewhere, *Secondary Privacy Damage* [78] is a privacy invasion to other users connected to the user accessing the webserver. Even if there is a relationship between two users whose data is captured by the proposed schemes, in most scenarios, secondary image damage is unlikely to occur. For example, computer information like browser data or *clock skews* [77] have no inherent link between two related users accounts. However, there has to be a scale of certainty levels for the different collected data to not eventually lead to user's identification.

5.5 Secure User Setup And Privacy Countermeasures

In this section, we describe two privacy countermeasures, namely nat and onion network. There is a trend for other privacy-preserving tools (e.g. Disconnect[4]); however, for all opting out solutions, care must be taken with regard to anti-tracking observation described in section 5.4.2.

5.5.1 Network Address Translation

An example of countermeasures that can prevent the running web metering scheme from getting *Protected* information in P3P analysis in section 5.3.4 could be a user behind a firewall with anonymous browser settings. In particular, using

[4]disconnect.me

Network Address Translation (NAT) [114], a unique address is changed to a different global one when communicated to another network. A NAT operation can be simplified as follows. A user inside a network makes a request to a webserver over the Internet. The user's request is first sent over the local network to the gateway (e.g. a router). The router changes the request machine IP address and source port into the global IP address and a new source port respectively. The router also records the request machine IP address and source port along the new assigned port. Then, the router does other required checks (e.g. integrity) before sending the user's request to the destination IP address and port specified in the request. When the router receives a reply, the router searches its record for the reply destination port address. If there is a match, the router extracts the corresponding user's IP address and port. Then, the router forwards the received reply to the extracted address and port.

5.5.2 Onion Network

An *Onion Network* [45] is an alternative privacy-enhancing solution to proxies and home NATing devices especially in case the proxy is not trusted or not directly connected to the user (and consequently an adversary can observe the proxy's received requests). In such networks, the route from the user to the webserver is randomly set where each node en route only knows its predecessor and successor. The schemes proposed in this thesis can work with Onion Network and still balance privacy with accuracy trade off properties. The use of TOR (The Onion Router) browser is not different from typical ones and can currently work on major platforms. Using such network can protect essential user's private data particularly his location. The following are the protocol steps [45].

1. **User → Node 1 :** $ENC_{pubkey}\ [x1 = g^{s1}]$

2. **Node 1** $\rightarrow$ **User** : $y1 = g^{s2}$, $HASH$ $[K1 = g^{s1*s2}]$
3. **User** $\rightarrow$ **Node 1** $\rightarrow$ **Node 2**: ENC_{K1} $[x2 = g^{s3}]$
4. **Node 2** $\Rightarrow$ **Node 1** $\rightarrow$ **User**: ENC_{K1} $[y2 = g^{s4}]$, $HASH$ $[K2 = g^{s3*s4}]$
5. **User** $\rightarrow$ **Node 1** $\rightarrow$ **...** $\rightarrow$ **Node x**: $ENC_{K1,K2,.,Kx}$ [Connect To Webserver *identifier*]
6. **Node x** $\rightarrow$ **Webserver** : *TCP handshake*
7. **Node x** $\rightarrow$ **...** $\rightarrow$ **Node 1** $\rightarrow$ **User**: ENC_{Kx} [Successful Connection]
8. **User** $\rightarrow$ **Node 1** $\rightarrow$ **...** $\rightarrow$ **Node x**: $ENC_{K1,K2,.,Kx}$ [Get Webserver Object]
9. **Node x** $\rightarrow$ **Webserver** : Get Webserver Object
10. **Node x** $\rightarrow$ **...** $\rightarrow$ **Node 1** $\rightarrow$ **User**: ENC_{Kx} [Requested Webserver Object]

The user creates a secret key and negotiates Diffie Helmann parameters [43] with the first random node (Node 1). In step 1, the user sends $x1 = g^{s1}$ to Node 1 encrypted using Node 1 public key. In step 2, Node 1 sends back g^{s2} and a hash of the agreed Diffie Helmann key (g^{s1*s2}). Furthermore, from the agreed key, a key can be derived for each direction. In step 3, the user sends a request to Node 1 to extend the connection to Node 2. The request contains $x2$ for a new secret exponent, encrypted using the symmetric key $K1$. Once Node 1 receives the request, it decrypts it and forwards $x2$ to the next random node (Node 2). In step 4, Node 2 generates a secret ($s4$) and sends $y2$ and a hash of the agreed key ($K2$) to Node 1. Node 1 encrypts the received response using $K1$ and sends the encrypted message to the user. Then, the user calculates the new agreed key ($K2$) and checks the hash. Keys sharing and their forwarding in step 3 and 4 are repeated for further nodes e.g. Node 3. In step 5, the user sends a connect request to Node 1 encrypted using all agreed symmetric keys. In step 6, the end node (Node x) negotiates *TCP handshake* with the intended webserver, without encryption. In step 7, Node x sends a successful connection status message to the previous node in the path i.e. Node x-1, encrypted using the user and Node x agreed symmetric key. Similarly, the status message gets further encrypted down the path to user. Last, the user decrypts the received message using all agreed symmetric keys. In step 8, similar to step 6, the user sends webserver access request to Node 1 encrypted using all agreed symmetric keys. In step 9, Node x sends the received request to the webserver. In step 10, similar to step 7, Node x sends to the user the received webserver reply encrypted using the agreed key. Last, the user decrypts the received reply using using all agreed symmetric keys to reveal the requested webserver object.

The latest version of TOR (The Onion Router) browser should be used with its recommended settings. For example, Java[5] should be disabled so that Java circumventing attacks [91] are not successful. Otherwise an adversary's Java applet could surpass the onion network (or proxy) setup by making a direct connection to the webserver. A version of TOR browser makes both the screen and the window width and height always the same, which is distinctive[6]. Following anti-tracking observation, a user with unusual equivalence of certain browser characteristics will appear distinctive and reveal the usage of TOR browser. Further details are provided in section 8.7.2.

The following are two User Agents captured by a webserver for UK-based users with and without TOR browsers. The examples highlight the information that webservers can obtain that can be used with the proposed privacy-enhancing schemes (e.g. hardware based scheme in Chapter 6). The location of end nodes showed user's location is instead Amsterdam (Netherlands). The operating system is the generic Windows NT 6.1 instead of MAC. Also, the browser was Mozilla

[5]www.java.com

[6]https://trac.torproject.org/projects/tor/ticket/10833?cversion=0&cnum_hist=8

instead of *Safari*. Both user's location and computer categories were protected during a typical use of TOR browser.

```
% 31.5...141 is the user's IP address from a location in UK using
Safari browser on iPad to access the webserver. The following is
the first user's request to get the webserver homepage.

31.5...141  - - [22/Apr/2014:12:40:33 -0400]
"GET / HTTP/1.1" 200 1897 "-" "Mozilla/5.0 (iPad;
CPU OS 7_0_4 like Mac OS X) AppleWebKit/537.51.1
(KHTML, like Gecko) Version/7.0 Mobile/11B554a Safari/9537.53"


% The following is the first user's request to get the webserver
homepage using a TOR browser. 77.2...162 is the IP address
of the end node in Amsterdam. The end node's browser is
instead Mozilla and operating system is Windows NT 6.1.
The real IP address and different computer data can be
protected and can still work in schemes which do not
depend on such private data.

77.2...162  - - [22/Apr/2014:12:41:17 -0400]
"GET / HTTP/1.1" 200 1897 "-" "Mozilla/5.0
(Windows NT 6.1; rv:24.0) Gecko/20100101 Firefox/24.0"
```

5.6 Summary

As per international guidelines, privacy is a right that many governments are supporting. Following all major privacy guidelines, we believe our decision of protecting user's privacy while serving others' needs (i.e. advertising) is the first step in the right direction. A privacy policy has to be enforced in web metering context. Web metering entities (excluding the users) are likely to be motivated to comply with privacy rules and regulations, as a public compliance announcement and to avoid potential penalties. Even if user's privacy is preserved, the user still has the right to vote on web metering policies. Consequently, opting out should still be an option in all cases and schemes. For example, taking users' preferences into consideration was a goal in P3P. That is, the user can vote on all published web metering guidelines and the user has an option of actively opting out while interacting with webservers. In this research, we proposed different options and a political decision can promote any of them. In such decision, all options are evaluated using the policies and guidelines, and the judgment will promote the best one(s).

In this chapter, we showed privacy problems of web metering schemes from different perspectives. We referred to established guidelines and used relevant criteria to compare web metering schemes with each other. We then stated our observations from different types of experimentation. Last, we discussed secure countermeasures to users to preserve their privacy.

Part II

Privacy-Preserving And Secure User-Centric Schemes

This part proposes three user-centric web metering schemes that satisfy security, and privacy from different levels. The first scheme uses a hardware device, the second scheme uses hash functions and the third scheme uses identity management systems.

Chapter 6

A Privacy-Preserving Web Metering Scheme Via User-Centric Hardware

Contents

This chapter proposes a novel web metering scheme, utilising user-centric hardware, to provide web metering evidence in an enhanced privacy-preserving manner. We also analyse current web metering schemes using an established privacy guideline and show how web metering can conflict with privacy. This new proposal uses existing user hardware to provide web metering evidence. The methodology was then to include an existing mathematical problem to ensure security while preserving privacy. We first describe background information for the proposed scheme then we describe the novel scheme using a specific protocol. Then, we analyse the scheme from security and privacy perspectives. After that, we provide our implementation results and last our conclusion.

6.1 Introduction

The novelty of the proposed scheme is as follows. We propose a new web metering scheme that uses a hardware device at the user side to provide web metering evidence in a privacy-preserving manner. To the best of our knowledge, the proposed scheme is the first generic hardware-based user-centric web metering scheme. We show that the proposed scheme has the required security properties and enhances the privacy of users. In addition, we show that, aside the presence of the hardware component, the scheme can be implemented in a way that makes web metering transparent to the user. We also use privacy measurements to analyse and compare existing web metering schemes, showing the benefits of the proposed scheme.

The user can also be motivated to be involved in a web metering scheme as follows. The user might prefer an anonymous subscription to certain webservers offering particular content e.g. streaming films. The user would then have to purchase the subscription which also executes the privacy-preserving web metering scheme.

Related Work On Secure User-centric Hardware: A relevant application that uses hardware decoders but not for web metering purposes, is pay television. Here, the user has to have hardware decoders to get multimedia content sent by a broadcasting server. Only authorised users' decoders can decrypt the broadcast content, using the embedded decryption keys. The server encrypts the broadcast content, which will be decrypted using the corresponding decryption key, inside the hardware decoder. The technique can also have other security properties like a tracing capability to detect rogue decoders that share the decryption keys [34].

The remainder of this chapter is organised as follows. Section 6.2 proposes a generic web metering scheme and provides an analysis covering assumptions, goals and practical aspects. Section 6.3 describes techniques to implement the generic scheme. Section 6.4 analyses the proposed scheme from security and privacy perspectives. Section 6.5 provides a proof-of-concept implementation analysis. Section 6.6 concludes the chapter. Parts of this chapter were published in [4, 5].

6.2 Web Metering Via User-Centric Hardware

6.2.1 High Level Description Of Proposed Scheme

Inspired by the webserver-centric hardware-based web metering scheme in [13] and the use of secure user-centric hardware-based broadcasting technique (e.g. pay television) in [34], we propose here a new web metering scheme that relies on a hardware device at the user side.

Definition 2. A **secure device** is an abstraction for an integrated circuit that can securely store a secret value. To access that secret value, a processor is needed which can be inside that hardware device or inside an attached computing platform. The device has to be equipped with a technique (e.g. *zeroization*) so that the secret key cannot be extracted.

The device contains a secured secret key used for authentication, and has the ability to store another signature secret key, inside or outside the device. Examples of such hardware devices are a smart card or an enhanced version e.g. a Trusted Platform Module (TPM) [65]. In addition to TPMs, two factor authentication is an authentication mechanism in which the user uses two different credentials e.g. a password and a hardware token. Banks two factor authentication token[1] is a non-transparent example of a hardware device distributed to the user for a secure webserver visit. The adversary could still *purchase* hardware devices for "fake" users' identities. The cost should typically be higher than the gained benefits, as in [57]. Our generic web metering scheme operates in an environment which consists of a webserver, a user, who owns a hardware device, and an audit agency. The three parties follow the protocol specified below. First, we define hardware authentication which will be used as a step in the generic scheme as follows.

Definition 3. Hardware authentication is a unilateral authentication [42] in which the audit agency is assured of the claimed communicating user's identity.

The following is a generic protocol for the proposed web metering scheme.

1. **User** $\rightarrow$ **Webserver** : Access request
2. **Webserver** $\rightarrow$ **User** : Certificate request
3. **User** $\rightarrow$ **Audit Agency** : Hardware certificate
4. **User** $\leftrightarrow$ **Audit Agency** : Hardware authentication

[1]www.hsbc.co.uk/1/2/customer-support/online-banking-security/secure-key

5. **User** $\rightarrow$ **Audit Agency** : New key
6. **Audit Agency** $\rightarrow$ **User** : Certificate for new key
7. **User** $\leftrightarrow$ **Webserver** : Certificate & signature
8. **Webserver** $\leftrightarrow$ **Audit Agency** : Verification key & evidence

The protocol for the proposed web metering scheme consists of eight steps, as follows.

1. User sends an access request to webserver.
2. Webserver replies with a certificate request.
3. User sends the certificate for the secret key, inside the hardware device, to audit agency.
4. Audit agency checks the received hardware device certificate. If the certificate is valid, audit agency authenticates the communicating user by checking whether he can access the corresponding hardware device.
5. If authentication succeeds, the user generates a signature key pair and sends public part of it to audit agency.
6. Audit agency signs the received public key and sends a certificate back to user.
7. User signs webserver URL and time using the new signature key and sends the signed URL and certificate received in step 6 (or a form of it) to webserver.
8. Webserver checks the certificate and the signature, and stores them as evidence.

In step 1, the user sends an access request to an object in the webserver. In step 2, the webserver checks whether the user has submitted a valid (attestation) certificate. If not, the webserver requests a certificate signed by the audit agency.

In step 3, the user checks if he holds a valid certificate. If so, step 7 is instead executed. Otherwise, the user sends to the audit agency, the certificate for the secret key embedded in the hardware device. For example, the *hardware certificate* can be a signature by a *hardware authority agency* (e.g. Intel) for a public key, where its corresponding private part is embedded in the hardware device. In step 4, the audit agency checks the validity of the received certificate (e.g. not revoked). If the certificate is valid, the audit agency checks whether the user holds the corresponding secret key in relation to the certificate. For this step, the user is asked to encrypt fresh nonces using the embedded secret key. In step 5, if the user is authenticated, he generates a new signature key pair and sends the public part of it (verification key) to the audit agency. This step can be executed for x number of key pairs. In step 6, the audit agency signs the received verification key (blindly if privacy is required) using its signature key and sends the produced signature (requested certificate) to the user. In step 7, the user forwards the received certificate in step 6 to the visited webserver or convinces the webserver that he has obtained a certificate. The user also sends his verification key to the webserver if it is not included in the submitted certificate. The user also signs a webserver identifier (e.g. URL) and possibly other information (e.g. time) and sends the signature to the webserver as evidence of the visit. For efficiency reasons, the webserver could periodically publish reference numbers (or pseudonyms) that can be linked to the webserver and time instead of concatenating URL and time for the *evidential signature*. In step 8, the webserver checks that the certificate was somehow *signed* using audit agency signature key. The webserver also checks (possibly using a privacy-preserving protocol) that the received signature was signed by the user's new signature key. If both checks succeed, the webserver stores the certificate and signature as web metering evidence.

The following is an example of the proposed scheme. Assume some webservers can only be accessed if the users own web metering hardware devices. Once the user accesses a webserver at some point in time, the user gets redirected to an audit agency. The audit agency first ensures that the user owns a valid hardware device. Then, the user generates and sends a session key and asks for a certificate. The audit agency produces the certificate using different possible schemes and sends it to the user. The user gets redirected back to the webserver when he submits the certificate and a new signature message.

6.2.2 Security And Privacy Assumptions And Goals

We assume that number of corrupt users is small as done in [11]. In particular, the webserver cannot convince significant number of users to collude with it, to create fake web metering evidence. The rationale behind this assumption here is that the number of users captured by web metering evidence should typically be large and unlikely for the webserver to be able to cost-effectively motivate a considerable number of users into colluding. Also, colluding participants have to risk losing the functionality of their hardware devices once tagged as rogue.

User-centric hardware-based web metering schemes have a potential to overcome user impersonation attacks and can be designed to preserve users' privacy. This can be achieved by involving the audit agency in the user setup or increasing the cost of webserver faking visits, as followed in the lightweight security approach in [57]. The hardware introduction is used here to increase the cost for a corrupt webserver to fake visits by requiring it to own a hardware device for each fake user. At the same time, the scheme has to ensure that it is impossible for a corrupt webserver with one authentic hardware device to be able to generate an unlimited number of evidences e.g. using a periodic hardware authentication with a limit of issued certificates. Therefore, we need a hardware device at the user side containing a secret key. Also, the secret keys certificates and public cryptographic values have to be available to the audit agency as they are required in step 3. In steps 3 and 7, the user is assumed to be securely redirected and may not necessarily be aware of this ongoing web metering operation, if a privacy-preserving scheme is being used in a *transparent* mode.

A summary of the assumptions we followed in this chapter are as follows.

1. Number of corrupt users is far less than the total number of metered users.
2. User owns a secure hardware device (as in Definition 2).
3. The audit agency can obtain a list of valid devices certificates (e.g. from Intel) and recognise revoked or expired ones. Alternatively, users could be incentivised to register their authentic hardware devices for privacy-preserving browsing.
4. The web metering environment is where the user's privacy is a concern.
5. There is limited value of the online content (affecting the cost for webserver owning hardware devices).

In the rest of this section, we further describe attacks that can happen during a hostile web metering operation and then highlight the required security goals to counter such attacks. We derive the following security attacks from the adversary capabilities described in Dolev-Yao threat model [46]: replay, impersonation and man in the middle attacks. Further details are provided in Chapter 2.

Replay Attack. A replay attack occurs when an adversary captures data sent from the user to the metering provider, the audit agency or the webserver and sends the data again. Similarly, an adversary captures data sent from the webserver to the metering provider or the audit agency and sends the data again.

If a replay attack is not detected, the visits number may be increased.

Impersonation Attack. An adversary in an impersonation attack (which is more powerful than the replay attack scenario where attack effect is limited to captured data), creates fake data and sends it to the metering provider or the audit agency impersonating a valid webserver or user. Or an adversary creates a fake request to a webserver impersonating a valid user.

If an impersonation attack is not detected, the visits number may be increased or the evidence data may have invalid properties.

Man In The Middle Attack. Man in the middle attack occurs when an adversary receives data from the user or the webserver not intended to him and modifies it before forwarding it to the intended party.

If such attacks are not detected, the visits number may be increased or the data have invalid properties.

Besides the three communication attacks, there is also a threat that a corrupt webserver may not follow the required **web metering operations**. A corrupt webserver is inherently motivated to change the number of visits. Also, a corrupt webserver can be motivated to change some metering operations without changing number of visits. For example, a corrupt webserver intentionally changes a webpage identifier, which is going to be recorded in the web metering evidence, to a different webpage that charges higher fees for advertisements.

To counter such attacks, there have to be data integrity of the web metering results and secure web metering operation (we define these concepts below). Data integrity is a property that counters threats to the validity of data [42]. Once this property is satisfied, it provides protection against unauthorised modification or destruction of data. Data integrity in web metering refers to the integrity of stored and transferred evidences and data, as follows.

Evidential Integrity Goal. Evidential integrity assures that evidences are kept as they were originally produced and stored. That is, once evidences are generated, they have to maintain their exact state and not change (maintaining evidences state includes intentional and accidental changes). Also, this integrity includes stored data that requires post processing work to produce the final web metering evidence.

Evidential integrity requirement is needed as a countermeasure to the web metering operation and stored web metering result attacks.

Communication Integrity Goal. Communication Integrity is concerned with integrity of the communication channels used for transferring web metering data. Transferred web metering data refers to pieces of data transmitted between users, webservers and audit agency that can be used to constitute the web metering evidence. Communicating this data has to be done in a way that if the data is changed en route, the change is going to be detected.

Communication integrity requirement is needed as a countermeasure to man in the middle communication channels attack.

The following security requirement is needed as a countermeasure to communication channels (replay and impersonation) and web metering operation attacks.

Security Goal. A web metering scheme is secure if its web metering operations are executed as expected per its specifications and cannot be affected by an adversary, to eventually provide consistent results and evidence.

High Level Analysis Of Proposed Scheme. To ensure that an impersonation attack is not feasible, step 3 has been included as only users who have valid hardware devices will be authenticated (because the key cannot be extracted from the hardware device). As a result, an impersonation attack for imaginary set of users will require an adversary to own a hardware device for each impersonated user, which is not feasible. To ensure that man in the middle attack is not successful, step 4 has been included as an adversary listening to communications will not be able to satisfy the required authentication. Similarly, an adversary interfering the certificate in step 6 will not posses the corresponding secret part. To ensure that replay attack is not successful, time should be included in the signed messages. We provide a more detailed analysis of security properties of the scheme in section 6.4.1.

To preserve user's privacy, in step 6, the audit agency has to blindly sign the new user's key and send the blind signature (i.e. certificate) to the user. Owing to the blind signature production, the audit agency does not know the user's key. In step 7, the user submits a form of the received signature or proves to the webserver that she possesses an audit agency signature on the new web metering signature key. The webserver would store the signatures as evidence for number of visits that are done by users carrying authentic hardware devices. We provide a more detailed analysis of privacy properties of the scheme in section 6.4.2.

6.2.3 Practical Aspects

The use of hardware devices is common today. Commercial hardware tokens[2] can be used in the proposed scheme as long as they hold a *zeroizable* secret for authentication. In case the user is not motivated to explicitly participate in the web metering scheme but still have an applicable hardware device, the scheme can still be run transparently to the user, where a program (or a script[3]) anonymously attests the user. One current application requiring TPMs are digital wallets [33]. In a typical application, the user uses a virtual wallet program on a

[2]www.safenet-inc.com/uploadedfiles/about_safenet/resource_library/resource_items/product_briefs_-_edp/safenet_product_brief_ikey_4000.pdf

[3]www.cometway.com

Table 6.1: Limitations

Strength	Weaknesses/Challenges
• Privacy	• Transparency
• Security	
• Accuracy	

device to make a transaction. Potential motivations for such a wallet over credit cards could be finding better deals or further authenticating communicating users with customised information set in the wallet. Recent commercial devices (e.g. iPhone 6[4]) use Near-Field Communication (NFC) technology for such digital wallet. NFC devices can use TPMs [63]. On the other hand, an organisation might want to restrict accesses to their local network once users have certain hardware devices in a fashion similar to Virtual Private Network (VPN) connections. For example, the distributed hardware devices can provide the required connectivity and privacy-preserving web metering results. On a larger scale another non-transparent scenario could be to distribute free zeroizable devices to users (e.g. USB storage sticks).

There is also a trend of developing extra hardware devices (rather than traditional Personal Computers or mobile phones) for various desirable functions e.g. Google Glass[5]. Along the main functions like cameras or games, accessing certain webservers can be an additional function using a privacy-preserving web metering scheme.

6.2.4 Summary

The proposed scheme depends on discrete logarithm problem. Transparency (and other efficiency requirements) can also be satisfied. A challenge for the proposed scheme is the existence of a zeroizable hardware device at the user side. We

[4]www.apple.com/iphone-6/
[5]www.google.com/glass/

showed in section 6.2.3 the different options considering the trend of such secure hardware devices. Table 6.1 provides a summary.

6.3 Techniques To Implement Proposed Scheme

In this section, we start by describing mechanisms to implement each step in the proposed generic scheme.

Steps 1 and 2 can be implemented using standard mechanisms for issuing requests e.g. HTTP requests.

Steps 3 and 4 address the identification and authentication of the hardware device. As mentioned in section 6.2.1, a TPM can be used as a web metering hardware device for the required hardware authentication step. A *trusted computing* platform is a device which has an embedded TPM, which has Endorsement Key (EK) and a certificate on the public part of it to prove the platform is genuine. We can follow with such hardware device the lightweight security approach, where it is still possible for an adversary to construct fake web metering evidence but its cost does not offset the earned benefit.

Steps 5, 6 and 7 are included in the proposed scheme to take into account the privacy requirements. Step 8 is optional depending on whether the webserver needs to contact the audit agency for certificates or evidence redemption.

In the rest of this section, we describe existing protocols and schemes that can be used to implement steps 5, 6 and 7 in the web metering scheme in section 6.2.1. Using them, we give a technique to implement the scheme, satisfying both the security and users' privacy requirements. Table 6.2 outlines the mechanisms, satisfying the different requirements.

TABLE 6.2: Mechanisms Comparison

Security Without Privacy	Security And Privacy
Key transport protocol	Zero-knowledge protocol
Audit agency signature	Audit agency blind signature
User non-repudiation signature	User evidential signature of knowledge

Security Without Privacy. To implement step 5, the following secure but *non-privacy-preserving* key transport protocol [24] can be used, where the audit agency and the user have public key pairs.

1. **User → Audit Agency** : $ENC_{AuditAgency}[identifier, key, count]$
2. **Audit Agency → User** : $ENC_{key}[count{,}nonce]$
3. **User → Audit Agency** : $SIG_{User}[AuditAgency, H(nonce{,}count{,}key)]$

In the first message, the user encrypts, using a standard encryption scheme, his identifier, a new signature key and its count number using audit agency public key and sends it to the audit agency. The audit agency decrypts the message and encrypts the received count number and a nonce using the new key and sends it to the user. The user decrypts the message to reveal the nonce to hash it with the count number and the new key. Then, the user signs, using a standard signature scheme, the hash code along the audit agency identifier with his public key and sends the signature to the audit agency. This signed message can be used to link the new signature key to the user's identity. For step 6, once the new signature key is securely sent to audit agency, the audit agency signs it with its private key and sends the signature to the user as a certificate. For step 7, the user produces evidential signatures using the new signature key and uses the audit agency certificate to confirm its validity.

Security And Privacy. By contrast, to provide a privacy-preserving web metering scheme, the user has to commit to a new signature value for step 5 in the generic scheme, for example using Pedersen commitment scheme [100]. For the next step, an audit agency has to blindly sign the committed value (once the user is authenticated) and allow the user to prove its possession, without revealing it. For step 7, the user uses the new signature value, without linking it to the former authenticated credential.

A general view of the privacy-preserving technique required in step 5 can be two interacting entities in which one can prove to the other that it holds a secret without revealing it. New secrets can be generated with the help of a trusted third party while the former secret is "buried away" in another value. For example, using *Schnorr* zero-knowledge protocol [108], a secret s can be embedded in a smart card and used for signing such that $y = g^s mod\ p$ where g is a group generator and p and q are two large prime numbers such that q is a divisor of $p-1$ (y, g, p and q are public values). A commitment scheme can be used in constructing a zero-knowledge protocol. In the web metering context, the user can convince the audit agency that the interacted messages are correctly formed using zero-knowledge proof of knowledge of a discrete logarithm. The following are the corresponding steps for Schnorr protocol, that can be used for step 5.

1. **User → Audit Agency** : User chooses a random r and sends $a = g^r mod$ p
2. **Audit Agency → User** : Audit agency sends a challenge c
3. **User → Audit Agency** : User sends $b = r + c * s\ mod$ q

In step 1, the user chooses a random value r where $1 \leq r \leq q-1$ and sends the commitment $a = g^r mod$ p. In step 2, the audit agency sends a challenge c where $1 \leq c \leq 2^t$ for some security parameter t. In step 3, the user sends to audit agency $b = r + c * s\ mod$ q. The audit agency checks whether $a * y^c = g^b$. If the check is correct, the audit agency is convinced that the user knows the secret s. The result of this check can be used as a proof that the user knows s without revealing it. For implementing step 6 in the generic scheme, the audit agency has to document the result as a "redeemable" privacy-preserving certificate. Then, for step 7, the zero-knowledge protocol has to run again between the user and the webserver.

In the following sections, we first show a secure technique that can be used to implement the generic scheme (without preserving the users' privacy). We then show a technique that can be used to implement the generic scheme, satisfying both the security and users' privacy requirements.

6.3.1 Privacy Certification Authority

In this part, we show a secure technique that can be used to implement the generic scheme (without preserving the users' privacy). The first attestation method published by Trusted Computing Group (TCG)[6] was to use Privacy Certification Authority (CA). Privacy CA has the same role as the audit agency. The following are seven steps using this attestation method for web metering purposes. In step 1, the user sends an access request to the webserver. In step 2, the webserver sends a request for attestation to the user, to enable the webserver to reply to the user request and reliably record web metering evidence. In step 3, the user submits his hardware certificate (for EK) to Privacy CA when the Privacy CA cross checks it with published certificates. All hardware certificates are initially published by the hardware authority agency and their status can be updated by the Privacy CA. In step 4, Privacy CA validates the used EK with respect to the submitted hardware certificate. In step 5, if the checks are correct, the user generates Attestation Identity Key (AIK) and sends the public part of it to the Privacy CA. In step 6, Privacy CA signs the received AIK and sends a signed AIK certificate. In step 7, the user uses AIK private key for

[6]www.trustedcomputinggroup.org

producing evidential signatures to the webserver and the received AIK certificate to authenticate himself. Figure 6.1 shows the message flow for the described steps.

The privacy problem with using Privacy CA method is as follows. The webserver would send to Privacy CA (for example, on conflict resolution) the received signatures along the corresponding AIK certificate. Privacy CA can link the self-issued AIK certificate along the non-repudiation signatures to the corresponding used TPM's EK.

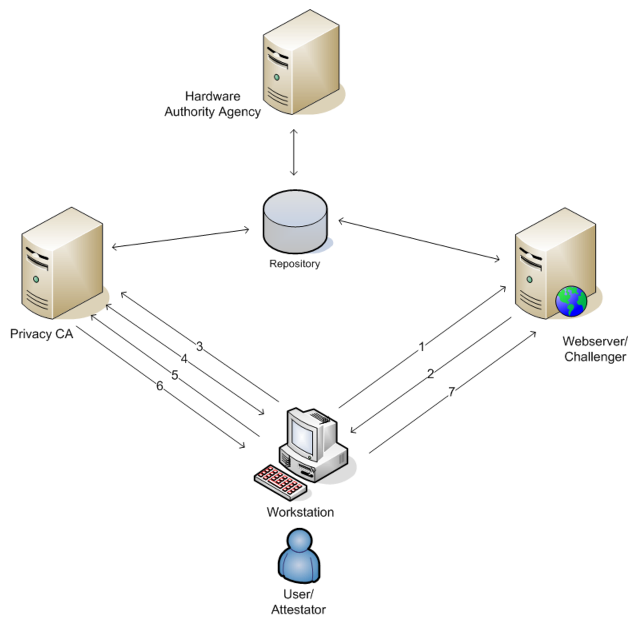


FIGURE 6.1: Attestation Using Privacy Certification Authority (CA)

6.3.2 Direct Anonymous Attestation Protocol

Direct Anonymous Attestation (DAA) protocol [25] can fortunately provide the needed public commitment, signature scheme and zero-knowledge proofs techniques. DAA protocol uses Camenisch-Lysyanskaya signature scheme [29] to provide a blind signature on the committed value and allow the user to prove its possession, through a zero-knowledge proof of knowledge of the committed value. As described earlier in the section, Schnoor zero-knowledge protocol can be used as an example to provide both security and privacy properties. As a result, once the user is authenticated, he can convince the audit agency of a newly chosen value without revealing it. Then, this new value (known only to the user) is used to interactively provide web metering evidence at the webserver side.

According to DAA protocol described in [25], communication between user and audit agency can be done using *Join Protocol* and communication between user and webserver can be done using *Sign/Verify Protocol*. Figure 6.2 shows the message flow for both protocols. Step *a* corresponds to steps 3 and 4 in the generic scheme. The rest of steps refer to the same order.

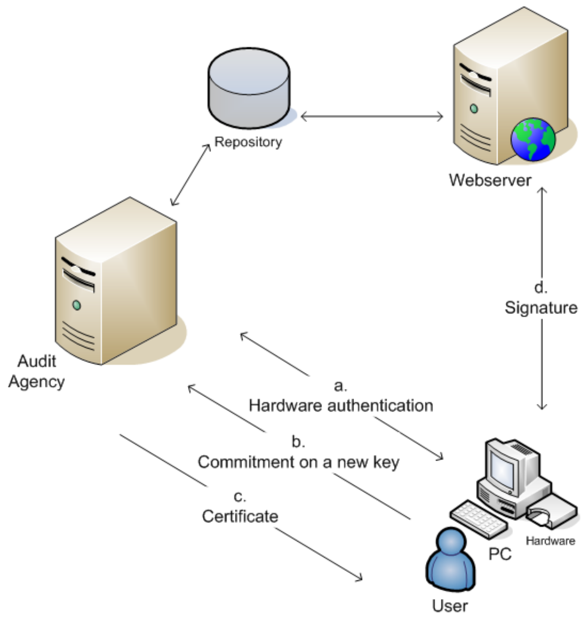


FIGURE 6.2: Anonymous Attestation Via User Hardware

The user gets authenticated to audit agency using EK (steps 3 and 4 in the generic scheme) and then receives a certificate as follows. In step 5, during Join Protocol, the user generates a secret key f and computes $U = z^f x^{v1} mod\ n$ where

$v1$ is used to blind f and (n, x, y, z) is public key of audit agency. (z can be set-up as $x^{r2} mod\ n$ where $r2$ is random number so that the audit agency chosen random number will be multiplied by the secret f and added to the blind $v1$). Also, the user computes $N = Z^f mod\ p$ where Z is derived from audit agency identifier and p is a large prime. Then, the user sends (U, N) to the audit agency and convinces the audit agency that they are correctly formed using a proof knowledge of a discrete logarithm. We assume that the challenges and messages are securely chosen and constructed as specified in [25]. Then, in step 6 in the generic scheme, the audit agency computes $S = (y/(Ux^{v2}))^{1/e} mod\ n$ where $v2$ is random number and e is a random prime. Then, the audit agency sends $(S, e, v2)$ to the user to have (S, e, v) as a TPM certificate where $v = v1 + v2$. More than one secret can be generated here to guarantee unlinkability in case the audit agency is offline. The join phase is the heavy work phase of the scheme and can be periodically done for different requirements.

In step 7 in the generic scheme, during Sign/Verify Protocol, the user signs messages using the secret key f and audit agency certificate (S, e, v) received in Join Protocol. The user also computes $N2 = Z_2^f mod\ p$ where Z_2 is a group generator that can be configured for a required anonymity level. Z_2 can be fixed for a limited period of time in synchronisation with audit agency certificate issuance to determine unique number of users. For example, to determine unique users for a period of one hour, the audit agency has to keep a record of hardware authentications so the user cannot generate another key f, and Z_2 has to be fixed, for that period of time. Also, Z_2 can be chosen by the webserver, reflecting its identity e.g. a code for its URL. The b bit can be specified in DAA protocol to indicate that the signed message was chosen by the user. Such feature makes DAA more flexible to different desirable web metering applications than ecash [31]. Furthermore, a signed hash chain as in [62] can be used to efficiently know the length of the visited session and set it to a desired length. Consequently, the user can use the hash chain result as a generator. If the user is still online after the session ends, a new signed message (of a new hash chain) is created. This new message cannot be linked to the previous one as the webserver is just convinced that these messages were signed by user secret keys generated during the Join Protocol without the need to know them (proof of knowledge). The different generators capturing webserver URL and time with different certificates can guarantee (and tune) such required unlinkability and session length.

The user can provide a proof that she has a certificate for the secret values (f and v) by providing a zero-knowledge proof of the secret values, such that the following equation holds: $S^e z^f x^v \equiv y \ mod \ n$. Then, the user sends the signature to the webserver and convinces the webserver that he knows f, S, e and v. The webserver checks the signature and if valid, the webserver stores $N2$ along the result of the zero-knowledge proof as web metering evidence, proving interactively the communicated user's TPM was genuine.

6.4 Security And Privacy Analysis Of Proposed Scheme

Part of the proposed scheme's security depends on the physical hardware device attached to a user's platform. The device contains a secured secret key used for authentication, and has the ability to store another signature secret key, inside or outside the device. The new generated signature key, used for producing web metering evidence, can still be retrieved (or extracted) by the user. Then, to preserve the user's privacy, the zero-knowledge proof guarantees that the new signature key cannot be related to the original key inside the hardware device, which was previously used for authentication. The used DAA protocol [25] guarantees that there is no anonymity revocation and consequently the audit agency is not able to identify the user as it cannot link the first key (i.e. EK) to the new key.

6.4.1 Security Analysis Of Proposed Scheme

Proposition 1. An adversary capturing all communicated messages, but not owning the device, cannot:

1. create fake web metering evidence (i.e. N2, see section 6.3.2);
2. cannot impersonate an existing user.

Therefore, the proposed scheme achieves integrity and security goals.

Proof.

We assume that the user owns a secure hardware device and number of corrupt users is small (as in section 6.2.2). Thus, hardware authentication (as in Definition 3) can only succeed by interactively proving the ownership of the physical hardware device containing the built-in secret key. Without a successful hardware authentication, valid evidence cannot be created in the absence of the subsequent committed signature key in step 5 (i.e. f). Consequently, the adversary has to own a hardware device in order to create valid web metering evidence. Moreover, we assume that the challenges and messages in steps 5, 6, and 7 are securely chosen and constructed as specified in section 6.3. Therefore, evidential integrity goal is achieved.

Depending on the audit agency setup, x certificates can be issued to the user after the successful hardware authentication, and valid for a limited period and cannot be reused. We assume that user's secret keys are used to encrypt nonces or time stamps, as specified in section 6.3, to ensure freshness as a countermeasure against impersonation and replay attacks for an observed user. While producing the zero-knowledge proof in proposed DAA-based scheme, the user has to interactively convince the other entity (audit agency or webserver) of the knowledge of secret values, using freshly provided challenges. Any captured messages that are resent again during Join Protocol will be rejected by audit agency as they will not fit in the current window of acceptable responses. Similarly, captured and resent messages during Sign/Verify Protocol will not enable webserver to construct new valid evidence N2 as they will not fit in the required window. Therefore, security goal is achieved.

Using zero-knowledge proof of a discrete logarithm [108], the adversary will not be able to learn the built-in secret key to pass the required authentication in Join Protocol nor be able to learn the corresponding secret signature key in Sign/Verify Protocol. Therefore, observing messages sent by a user will not enable the adversary to get the secret values to impersonate a valid user or hijack the session. Therefore, communication integrity goal is achieved.

□

6.4.2 Privacy Analysis Of Proposed Scheme

Proposition 2. The proposed DAA-based web metering scheme protects any identifying information captured from the authentic certificate of the user's hardware secret key.

Proof. By Definition 3, after hardware authentication, the audit agency is assured that the communicating user can securely access the secret key inside the hardware device and consequently can confirm the user's identity. Then, the zero-knowledge protocol [108] is used to convince the audit agency that constructed commitment messages were formed correctly without disclosing the secret value

f. Once the audit agency is convinced, the audit agency produces a certificate S for the user's committed secret value f which is later anonymously used during Sign/Verify Protocol.

We assume during Sign/Verify phase, the user keeps the audit agency certificate (S, e, v) secret and only uses it to convince the webserver of the knowledge of the chosen secret key f. Otherwise, the audit agency can match the hardware certificate identifier to user's visits as follows. The audit agency records all issued certificates for the received hardware certificates during the "blind" signature production. Then, once the webserver has the exact audit agency certificates along users' signatures, the webserver forwards them to audit agency. The audit agency can check and match the self-issued certificates to the recorded hardware certificates.

Similarly, there has to be a non-predictable difference in time or no pattern between user committing to a new signature key and using it. This is initially achieved by the two roles of audit agency and webserver when their involvement is separated by time. For example, when the user machine boots up, new keys are generated, approved and stored. For the next immediate visit, the user can either use previously approved signature keys or have to wait a random time before using the new keys. Then, the user is always set to contact the audit agency for new signature keys once the number of user's available keys goes below a threshold, say two. The random delay should be minimal as not to affect the user browsing experience. Also, to limit the effect of an impersonation attack, we can assume the scheme needs to re-run daily or every hardware boot up to limit the number of fake evidences. Therefore, the proposed scheme protects any captured user's identifying information.

$\square$

During the Join Protocol, the user computes and sends U which is "embedded" in the audit agency certificate S. The user then convinces the webserver the knowledge of S along other secret values without disclosing S. The user also computes and sends N as a provision for recording and possibly revoking the approved commitments, as proposed in [25]. If such *linkability* feature is not required and lifetime of all approved fs is designed to be limited, user's computation and submission of N can be skipped.

The proposed scheme does not stop an adversary from capturing other non-required private information about the user (e.g. IP address). A solution for such problem would be to use relevant security countermeasures (e.g. Network Addressing Translation [114] and Onion Network [45]) to prevent the capture of unrelated private data. Network Address Translation and Onion Network were described in section 5.5.

Scyther Validation.

Scyther tool can be used to automatically validate a security protocol and detect attacks, if any. Further details are provided in section 2.1. Scyther was used to validate the proposed scheme as follows.

```
usertype exponent;
protocol AAauthenUser(user,AA)
{
        role user
        {
        var AAnonce: Nonce;
        const g;
        secret exp : Function;
        fresh f : exponent;
        #User encrypts AA fresh nonces to prove possession of TPM secret key
        recv_1(AA,user, AAnonce);
        send_2(user,AA, {AAnonce}sk(user));
```

```
            /* Adversary cannot impersonate a valid user is proven by the secret
            claim of the SK (key inside TPM). */
            send_3(user,AA, exp(g,f));
            claim_user(user,Secret,f);
            }

            role AA
            {
            var commit;
            var password: Nonce;
            fresh AAnonce: Nonce;
            send_1(AA,user, AAnonce);
            recv_2(user,AA, {AAnonce}sk(user));
            # AA receives the user commitment on chosen secret f
            recv_3(user,AA, commit);
            }
}
```

Figure 6.3 characterises the roles using Scyther.

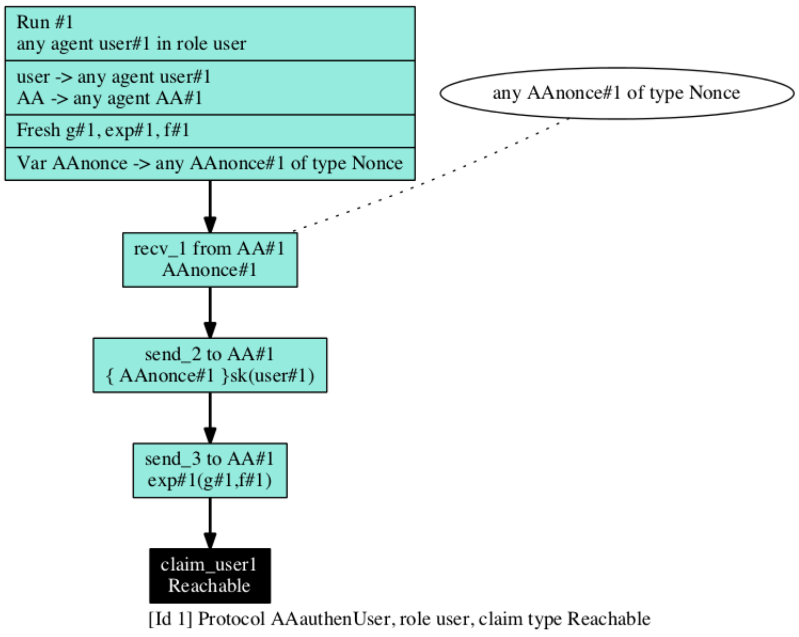


Figure 6.3: A Scyther User Trace Patterns

6.4.3 Privacy Analysis And Comparison

The Privacy Preferences Project (P3P) [38] is reused here to analyse and compare web metering schemes with the proposed one. Further details are provided in section 5.3.4. The following are the relevant P3P categories and detailed schemes analysis.

– **Identifiers** are issued to users by a third party, which can be the Government or generally a "trusted" third party, to identify the user e.g. a username. User hardware decoders techniques require users to have decryption keys upon subscription. Users' identifiers are required in the proposed scheme during hardware authentication in the Join Protocol. However, during the Sign/Verify Protocol, users' identifiers are preserved as previously used identification information, during Join Protocol, cannot be related thanks to the used zero-knowledge protocol.

– **State Management Mechanisms** are used to maintain the state of the connection to the webserver e.g. cookies. In user hardware decoders techniques, the state of the user can be tracked while decrypting on-the-fly broadcast content. The state of the user can be tracked in the proposed scheme depending on the key expiry date and the hash chain whenever used.

– **Interactive Data** includes data generated on-the-fly during user-webserver interaction e.g. a query to the webserver. User hardware decoders techniques can capture users' queries when encrypting particular responses (e.g. pay-per-view).

– **Location Data** category covers information regarding the users location e.g. users' GPS location. From the described user hardware decoders technique, we infer that users' location data cannot be captured.

Table 6.3: Privacy Comparison

Scheme	**Identifiers**	State	Interactive	Location	Computer	Navigation
Digital Signature [62]	✗	†	✔	✔	✔	✔
Secret Sharing [93]	✗	†	✔	✔	✔	✔
(User Hardware Decoder) [34]	✗	†	†	✔	✔	✔
Webserver Voucher [74]	✗	†	✔	✔	✔	✔
Processing-based [32]	✔	✗	✔	✔	†	†
Webserver Hardware [13]	✔	†	✗	✔	✔	✔
HTTP Proxy [10]	†	†	†	†	✔	†
Google Analytics [60]	†	✗	✔	†	†	†
Proposed scheme (DAA-based [25])	✔	†	✔	✔	✔	†

- **Computer Information** is any information about the user computer e.g. IP address. Users' computer information can only be captured in the proposed scheme during the Join protocol by figuring out information regarding the platform from the hardware certificate.
- **Navigation and Click-stream Data** covers data about the user browsing behaviour e.g. user clickstream. Depending on the proposed scheme implementation, navigation data can be captured. For example, if the signed webserver URL references various levels within the webserver content, navigation data can be captured during the lifetime of the used key f.

A summary of the P3P analysis is shown in Table 6.3. From two extremes, a particular private information can be either *required* by the scheme or *protected*. We use the symbol ✗ to denote the scheme cannot operate without the corresponding required private information in order to provide web metering result or evidence. On the other hand, we use the symbol ✔ to denote that the private information can be protected and not accessed by the adversary under secure user setup.

6.5 Proof-Of-Concept Implementation Analysis

We tested[7] the proposed scheme from the user side. We did various optimised simulation tests on a user machine with 2727578 high resolution performance counter frequency, using standard *math*, *gmp* and *OpenSSL* libraries[8] and the bit length specified in [25]. The prime numbers were of 1024 bits and modulus and generators were of 2048 bits. The public key values can be precomputed and stored at the hardware device, or securely downloaded to the user.

The tests showed feasible results. It took around 1650 nanoseconds to execute the first part of the public commitment (U) and around 515 nanoseconds to execute the second part (N). Then, the certificate production equation (S) needs only to be executed at the audit agency side, possibly by utilising Montgomery reduction algorithm [88] since the equation requires floating number exponentiation and *div* operations. From the results, the operations throughout the scheme phases can be executed in a short time so that they are not noticeable to the user.

We started by small bit length to cross check by hand that the calculations are correct. Speed can be increased when the recommended strict security properties were not taken into consideration. It took 250 ns to execute the commitment U once bit length was reduced to 128 bit. Despite being six times faster, the bit length reduction has obvious security concerns. On the other hand, by including a multiplication of two 1024 bit primes to get the modulus using less optimised code, the commitment took at most around 18000 ns. Taking into consideration that operations like MUL and EXP are slower than simpler ones like XOR and AND, resource limited devices could store precomputed values (e.g. inside the TPM) instead of computing the complete modular exponentiation. Further details about the code are provided in Appendix A.

[7]at National Center for Digital Certification (NCDC): Research & Development. www.ncdc.gov.sa
[8]http://www.openssl.org/docs/

The kind of secure hardware device required by the scheme is commonly used today. For example, the BitLocker program[9] uses the TPM public key for disk encryption, allowing the decryption (by TPM private key) if baseline platform measurements are met again. Furthermore, all Windows systems are planned to be shipped with TPMs in order to pass hardware certification of the latest operating system [121]. The scope of the required hardware devices is not limited to TPMs as discussed in section 6.2.3.

6.6 Conclusion

Privacy-preserving web metering is a challenging problem, especially with current necessary trade-offs and in environments where the audit agency could collude with webservers to identify users and link their visits. In this chapter, we proposed an alternative and new user-centric web metering scheme using hardware to enhance users' privacy. We described a generic web metering scheme of a straightforward protocol and a special case using an existing protocol. We also used established privacy guidelines to analyse and compare representative categories of web metering schemes and showed the gained privacy benefits of the proposed scheme. The proposed scheme can provide different security countermeasures and users' privacy settings. However, denial of service attacks are still possible.

We built a proof of concept implementation on a traditional computer to evaluate the efficiency and transparency of the running web metering operations. Transparency is the most problematic property to satisfy compared to other efficiency requirements like communication efficient property. Besides the operational cost from audit agency and webserver sides, the main barrier for a wide deployment

[9]www.technet.microsoft.com/en-us/library/cc162804.aspx

is that users should accept the hardware device. However, in many contexts, the gain in privacy will offset the costs. We discussed how the user hardware assumption can be realistic in today's and future computing devices and showed different options. The tests show that the usability requirement can be satisfied considering the trend of required hardware devices. It also runs unnoticeable to the browsing experience with typical computing resources; however, in order to guarantee faster implementation, the user may need to store additional values.

Future work includes exploring techniques for discovering rogue hardware devices, and implementing the scheme with different settings to provide the evidential signature e.g. hash chaining. Various options for counting the number of unique users can be further explored for different advertising applications. Future work also includes analysing the performance of the proposed scheme using handheld devices. Complete formal automatic validation of the proposed scheme is left for future work as well.

Chapter 7

User-Centric Schemes Using Third Party Authentication

Contents

This chapter proposes two web metering schemes that utilise already used authentication protocols to carry web metering evidence. The first scheme uses a hash-based authentication application and the second scheme uses an identity management system.

7.1 A New User-Centric Scheme Using Hash Functions

7.1.1 Introduction.

Hash functions can be used for web metering purposes in conjunction with other security mechanisms like digital signatures. For example, a web metering scheme can use a low cost signed hash chain [62]. In this section, we propose a web metering scheme utilising a property in some hash functions.

7.1.2 Background

We use here a standardised iterative hash function H as follows. Let IV be an L bit initialisation vector. Also, $B1$ bit of the data x is the input to the round function R while $B2$ bit is the output from the previous round. Last, O bit is $H(x)$ where $O \leq B2$. The hash operation starts by padding where x is padded by a multiple of $B1$ bit. Then, x is divided into $B1$ blocks. Then, IV and first block of x (i.e. x1) are fed into R. Then, R takes the next $B1$ bit input block and previous $B2$ bit hash output ($H1$) and the process is iterated for the rest of the blocks, as shown in Figure 7.1.

Extensibility Property. ([42])

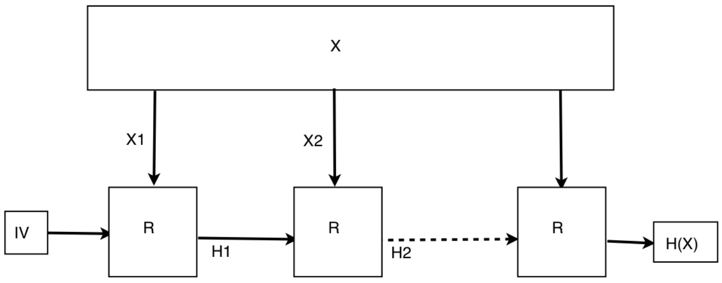


Figure 7.1: Basic Iterative Hash Function

Iterative hash functions, which use a simple padding scheme and no output transformation, can possess the extensible property in which given $H(x)$, $H(x||y)$ can be computed for any y without the need to know x.

7.1.3 Proposed Scheme

Assume the following simple web authentication application. The user possesses a password (or generally a secret) which is also known to the "trusted" audit agency. The webserver has the hash value of the password. The authentication application starts by hashing the submitted user's password. The webserver then compares the received hash value with the stored one.

The main idea of the proposed scheme is to that the webserver sends verifiable nonces to the user to be an input along the user's password, to a hash function. The webserver generates nonces (possibly periodically) and sends them to the user. Alternatively, the nonces could be generated by the audit agency and given to the webserver. Then, the user replies with the following: user's identifier, $H(password||nonce)$, $H(password \oplus nonce)$ and $ReferenceNo$. The user's identifier is how the user can be identified e.g. a username, $H(password||nonce)$ is the hash value of the password concatenated with the nonce sent by the webserver, $H(password \oplus nonce)$ is the evidential hash value of the password *XORed* with the nonce and $ReferenceNo$ is an optional pointer to a particular visit to the webserver. Initially during user account creation, the chosen user's password is hashed and stored at the password hashes repository at the webserver, possibly by involving the audit agency.

Proposition 3. If H is an extensible hash function and x is only known to particular parties, then $H(x||y)$ can be computed by any party for some y without the need to know x while $H(x \oplus y)$ can only be computed by those parties which know x.

Proof. Let y be an arbitrary value such that any entity with the knowledge of the public $H(x)$ and y can compute $H(H(x)||y)$. Then, by *Extensibility Property* definition, $H(H(x)||y) = H(x||y)$. $H(x \oplus y)$ can only be computed, even in iterative hash functions, by first *XORing* x and y and then running the round function R for the XOR blocks result to produce the hash value. Therefore, $H(x \oplus y)$ can only be computed by the entities knowing x while $H(H(x)||y)$ or $H(x||y)$ can produced by any entity. □

The following are the protocol steps in the proposed web metering scheme.

1. **User** → **Webserver** : Access request
2. **Webserver** → **User** : Authentication request: script & nonce
3. **User** → **Webserver** : ID & hashes
4. **Webserver** : ID & hash check

The following are the corresponding steps for the proposed scheme.

1. User sends an access request to webserver.
2. Webserver replies with an authentication request containing a script and a nonce.

3. User writes his identifier and password. The script produces the two hashes and sends them along the user's identifier to webserver.
4. Webserver checks the received user's identifier and the verifiable hash. If checks are correct, webserver grants user's access request and stores relevant hash as evidence.

In step 1, the user sends an access request to the webserver, for example, an HTTP request. In step 2, the webserver replies to the user with a script and a nonce. In step 3, the script gets the input user's identifier and password, and produces the two hashes: $H(password||nonce)$ and $H(password \oplus nonce)$. Then, the script sends the identifier and two hashes back to the webserver. In step 4, the webserver cross checks the user's identifier and extracts the corresponding stored password hash from its repository. The webserver then *pads* the extracted password hash to the sent nonce and compares the result with the received $H(password||nonce)$. If the check is correct, the webserver stores $H(password \oplus nonce)$ as evidence. The webserver can give received evidence along users identifiers (or pseudonyms) to interested parties where they can cross check with audit agency. Figure 7.2 shows the message flow for the described steps.

7.1.4 Security Analysis

The proposed hash-based web metering scheme uses a typical simple password application, that can be run transparently to the user, and still satisfy security requirement when producing web metering evidence. By Proposition 3, $H(password \oplus nonce)$ can only be calculated by the user or the audit agency and will serve as evidence of the user's visit. Also, $H(password||nonce)$ can be validated by the webserver despite not knowing the actual password. The scheme can also be used with salted hash password storage at the webserver utilising the

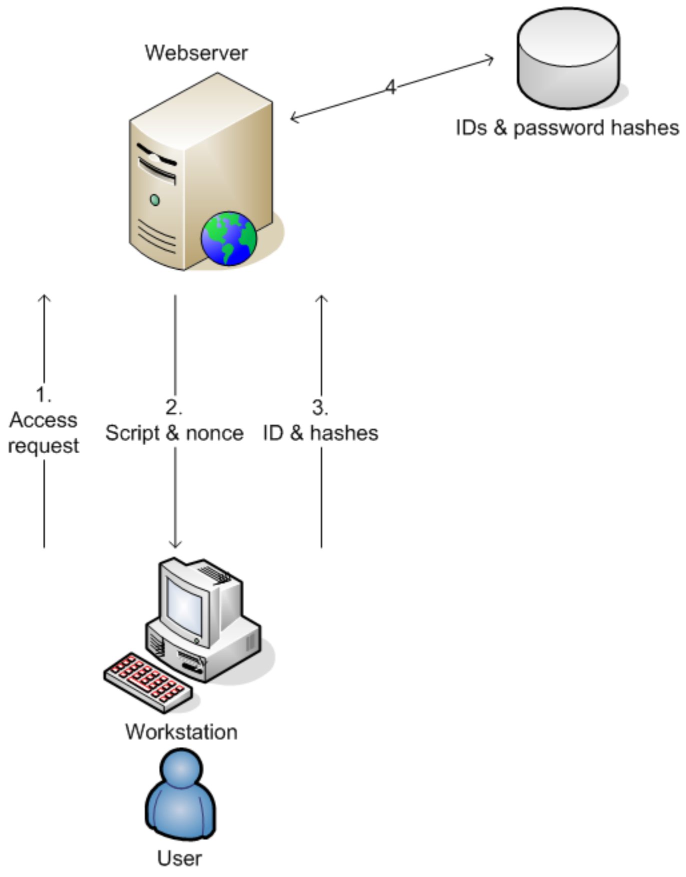


FIGURE 7.2: Hash-based Web Metering Scheme

Extensibility Property. However, users' privacy can be an issue as users' visits can be inherently linked to registered users' identities.

The security of the proposed scheme depends on the preimage resistance property [42] in hash functions. As a result, the adversary is unlikely to be able to invert the hash code to reveal $password \oplus nonce$. Further to such attack of faking evidence, the knowledge of $password \oplus nonce$ can enable the adversary to know the password using a set of XOR properties [36], as follows. The adversary *xors* the inverted $password \oplus nonce$ with the nonce. By associativity property, $(password \oplus nonce) \oplus nonce = password \oplus (nonce \oplus nonce)$. By nilpotence property, $password \oplus (nonce \oplus nonce) = password \oplus 0$. Since 0 is the neutral element, the result of the XOR operation will reveal the password.

The two hashes provide a form of data integrity and the adversary is unlikely to be able to find a new value that is equal to $H(password \oplus nonce)$. Also, if $H(password||nonce)$ is changed, the resulted hash will not match with the stored user's identifier.

Man in the middle and replay attacks are not successful here due to the use of nonces. An impersonation attack can be regarded as a denial of service attack as follows. An adversary can still change $H(password \oplus nonce)$ and the webserver cannot detect it. The adversary, with the knowledge of $H(x)$, can properly compute $H(x||n)$ for any n, but not $H(x \oplus n)$. The adversary would then send (or store) a correct $H(x||n)$, but wrong $H(x \oplus n)$ which will be rejected eventually by the audit agency. In environments where such denial of service attacks are still possible or not countered, the audit agency has to be set online during the web metering operation in order to check the received $H(x \oplus n)$ and allow or reject the visit accordingly. In such environments, the computation and submission of $H(x||n)$ can occasionally be skipped. However, given the nature of the simple authentication mechanism, the adversary could be limited to the webserver and the assumption of unlikelihood of such denial of service attacks can be regarded as reasonable.

7.1.5 Scyther Validation

Scyther tool was used to validate the proposed scheme as follows.

```
usertype name;
protocol protocol0(user,webserver)
{
        role user
        {
                fresh password: Ticket;
                fresh username: name;
                hashfunction H;
                var wNonce: Nonce;
                secret xor : Function;
                send_0(user,webserver,username, H(password));
                recv_1(webserver,user, wNonce);
                /* The random nonces are unlikely to be the same,
                and thus replay attack is unlikely to successful and
                will eventually be spotted by the audit agency. */
                send_2(user,webserver, H(H(password),wNonce));
                send_3(user,webserver, xor(password,wNonce));
                claim_user(user,Secret,password);
        }

        role webserver
        {
                var password: Ticket;
                hashfunction H;
                var username: name;
                fresh wNonce: Nonce;
                var evidence,receivedhash;
                macro checkticket =H(H(password),wNonce);
                recv_0(user,webserver,username, H(password));
                send_1(webserver,user, wNonce);
                recv_2(user,webserver, receivedhash);
                match(checkticket,receivedhash);
                recv_3(user,webserver, evidence);

        }
}
```

Figure 7.3 characterises the roles using Scyther.

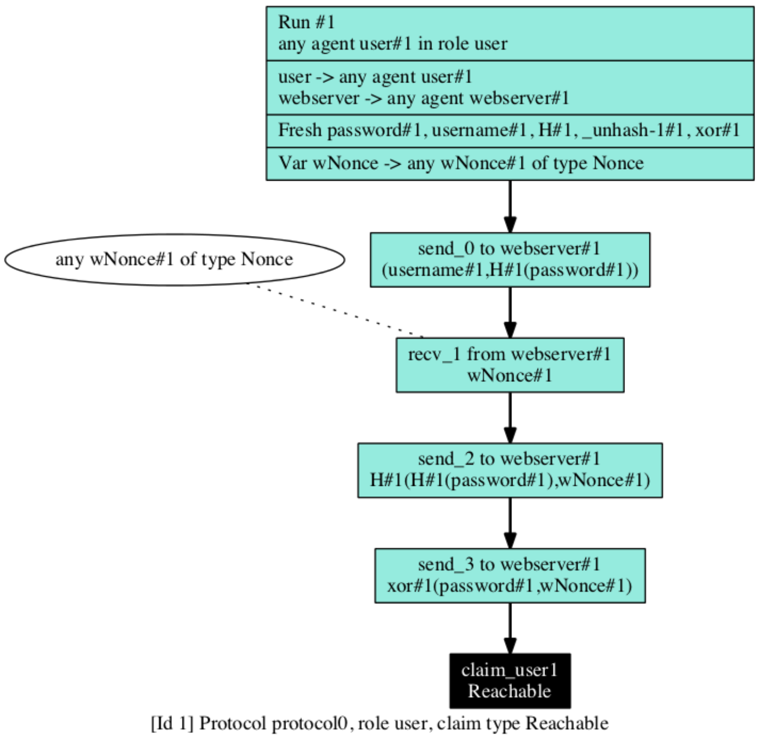

Run #1
any agent user#1 in role user
user -> any agent user#1
webserver -> any agent webserver#1
Fresh password#1, username#1, H#1, _unhash-1#1, xor#1
Var wNonce -> any wNonce#1 of type Nonce
any wNonce#1 of type Nonce
send_0 to webserver#1
(username#1,H#1(password#1))
recv_1 from webserver#1
wNonce#1
send_2 to webserver#1
H#1(H#1(password#1),wNonce#1)
send_3 to webserver#1
xor#1(password#1,wNonce#1)
claim_user1
Reachable
[Id 1] Protocol protocol0, role user, claim type Reachable

FIGURE 7.3: A Scyther User Trace Patterns

7.1.6 Scenarios

In this section, we provide alternative options for the used hash function, audit agency involvement and potential applications. Instead of computing $H(password \oplus nonce)$, *HMAC* [12] can be used as follows. A hash function is used here to get Message Authentication Code (MAC) for the nonce message using the secret password. $H(x1||H(x2||nonce))$ can be computed where $x1$ and $x2$ are two different values derived from the password. However, XOR operation has to be used again to compute x1 and x2 as follows. $x1 = password \oplus opad$ and $x2 = password \oplus ipad$ where *opad* and *ipad* are prespecified values.

If the trust level provided by the typical script is not sufficient (e.g. the password can be captured plaintext), there can be a redirection to the audit agency using HTTPS to download the authentic script and possibly a nonce. The HTTPS connection is used to ensure integrity of the connection. The redirection can still be done over HTTP to download a *signed* script[1]. Once the script is securely downloaded at the user side, it sends the required hashes to the webserver. On evidence verification, audit agency cross checks generated nonce at specific times (or timestamps) against ones recorded in web metering evidence. Figure 7.4 shows the web metering steps in such scenario.

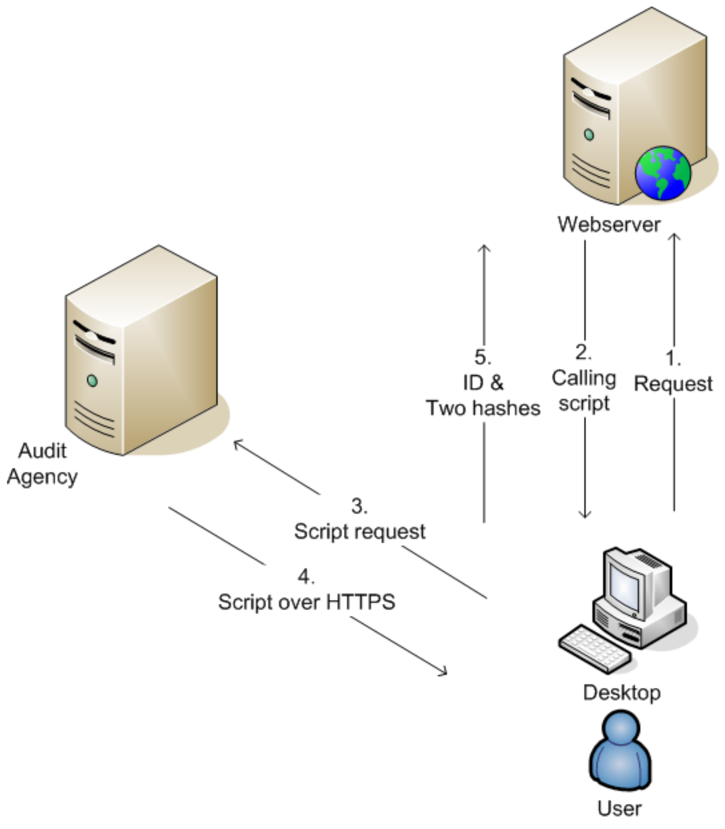


FIGURE 7.4: Redirection to Audit Agency in Hash-based Scheme

For communication efficiency, the user can periodically use timestamps instead of nonces, and possibly sends them along the two hashes. Steps 2, 3 and 4 can be repeated once the script is not running at the user side.

[1]www.mozilla.org/projects/security/components/signed-scripts.html

Reusing such current applications for web metering purposes can be looked at as an example to further generate web metering schemes and ideas that transparently reuse other existing applications (possibly proprietary). However, other options can be used to improve the security of the proposed scheme. In none-transparent scenarios, where the user is involved in the web metering operation, a verified script can be used. The user can securely store the web metering script or a program (e.g. a browser *plugin*) during account creation. The installed plugin executes scripts that are only signed by the audit agency. Alternatively, the user stores all verified scripts signatures. Once the script is downloaded from the webserver, the plugin computes its signature and compares it with the stored signature. If the check is correct, the plugin runs the script. A secure script can also force an Advanced Digest Authentication/Digest Access Authentication [70]. The webserver, which could be Windows Server 2003, has to be audited as well. The log of all requests originating from the webserver has to indicate the use of *digest authentication*.

7.1.7 Proof-Of-Concept Implementation Analysis

It is a tradeoff, the scheme can be secure by enforcing non-transparent solutions e.g. using cryptography. A reuse of existing "secure" applications is a way to satisfy such usability requirement. We captured the usability requirement by proof-of-concept implementation of the scheme using a hash function and XOR operation. A program can run transparently to the user and produce evidence. We tested the proposed scheme on a user's machine with 4 GB memory and 2.53 GHz processor. The proof-of-concept implementation test showed feasible results with a fast execution, that can be run transparently to the user using *Javascript*. It took around 2000 nanoseconds to compute $H(password||nonce)$

and $H(password \oplus nonce)$. We used open source *Javascript*[2] for Secure Hash Algorithm, SHA-256 [68] for a password and a nonce of six characters. Each of the produced web metering hashes are of 256-bit length, making the scheme communication efficient. There are no storage requirements at the user side.

7.1.8 Conclusion

In this section, we proposed a transparent web metering scheme using a simple authentication mechanism. The security of the scheme depends on the security of the used hash function. As for the privacy of the scheme, the user is interactively identified and authenticated by the webserver (or the audit agency). Further data, than such user's credentials, (e.g. IP address) are not required by the scheme. This scheme was the first step that enabled us to design a web metering scheme with more security features, as in section 7.2.

7.2 A New User-Centric Scheme Using Identity Management Systems

This section proposes a new web metering scheme using identity management systems. The methodology was to explore existing security solutions that can be reused for web metering purposes. Along such a transparent reuse, there is a consideration for a balance among the rest of requirements, which were validated through our analysis and using an automatic verification tool. First, we provide background information covering details for some representative identity management systems. Then, we describe the proposed web metering scheme, showing possible approaches and using a specific representative identity management system. Last, we analyse the proposed scheme and provide our conclusion.

[2]www.movable-type.co.uk/scripts/sha256.html

7.2.1 Background

Identity management is defined as a set of functions and capabilities used for assuring identity information, identity of an entity and enabling business and security applications [69]. We consider here a web-based identity management system and we choose a generic claim-based identity management system, OpenID and Kantara Initiative as representative examples for identity management system categories. A generic claim-based identity management system, OpenID and Kantara Initiative can be categorised as Information Card-based Identity Management, Single Sign-On and Federated Identity Management respectively as described in [6]. A web-based identity management model consists of a user (i.e. visitor), a service provider (i.e. webserver) and a third party called the *Identity Provider* (IdP) that partly performs metering operations.

In a web-based identity management model, the visitor initially sends a request to the webserver. The webserver asks for a security token (or a ticket) from a specific trusted IdP in order to grant the request. The IdP authenticates the visitor and either sends the security token directly to the webserver or forwards it through the visitor. The webserver checks the received token and, if valid, the webserver responds to the visitor request. An identity management system can be regarded as a visitor-centric web metering solution as the visitor has control over a secret. The three representative identity management systems follow a generic identity management system message flow shown in Figure 7.5.

Step 1 is included to request a token and can be implemented using standard mechanisms for issuing requests e.g. HTTP requests. **Step 2** is included to interactively deliver the token request and can be implemented using standard mechanisms e.g. HTTP requests. **Step 3** is included so that the webserver cannot impersonate a valid user and consequently receive an assertion and can be implemented using standard authentication mechanism e.g. username and password. **Step 4** is included as evidence of the visit vouched by the audit agency and can be implemented in SAML. **Step 5** is included to forward the requested token so the user is granted access to the webserver.

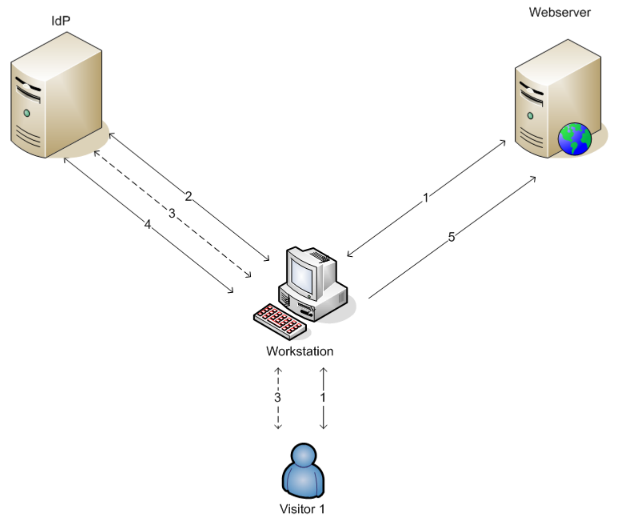


FIGURE 7.5: Generic Identity Management System

Security Assertion Markup Language (SAML) is a format for stating security information regarding a subject (i.e. the visitor). It is an XML-based standard which can be used for exchanging authentication messages between a webserver and the IdP. Simple Object Access Protocol (SOAP) can be used to carry SAML assertions within the HTTP protocol. A SAML assertion can carry any of the following security statements [96]:

– Authentication Statement: a statement from the IdP which specifies the authentication method used and the time, in the event that the visitor has been authenticated.

– Attribute Statement: a statement which contains extendable attribute information about the visitor.

– Authorisation Statement: a statement which recommends a decision regarding an access request. Authorisation decision statement is not available in SAML 2.0.

7.2.1.1 Claim-based Identity Management System

A claim is an assertion that certain identifying information belongs to the user. The mapping of physical IDs is an XML file stored on the visitor machine containing information that represents him, known as Information Card.

The following are steps in a claim-based identity management system. In the first step, after the visitor request, an *identity selector* at the visitor side retrieves and processes the webserver security policy. The identity selector checks which IdP the webserver trusts and extracts claims that need to be asserted and the security token type. Then, the identity selector finds Information Cards that match those requirements and asks the visitor to choose one, if there is more than one match. In step 2, the identity selector sends a security token request to the IdP and retrieves the IdP security policy. In step 3, the IdP authenticates the visitor. In step 4, after the visitor is authenticated, the IdP sends the requested security token. In step 5, the identity selector forwards the received security token to the webserver. The webserver checks the security token and, if valid, the visitor is authenticated to the webserver. The following is the message flow (identity selector is abstracted as simply the visitor).

1. **Webserver $\rightarrow$ Visitor** : Visitor receives and processes the webserver security policy
2. **Visitor $\leftrightarrow$ IdP** : Visitor requests security token and retrieves IdP security policy
3. **Visitor $\leftrightarrow$ IdP** : Visitor Authentication
4. **Visitor $\leftrightarrow$ IdP** : IdP sends the requested security token

5. **Visitor** $\rightarrow$ **Webserver** : Visitor forwards the received security token

7.2.1.2 OpenID

OpenID is an open source identity management system where a visitor registers with an IdP and receives an identifier [56]. The visitor can use this identifier, which is usually a Uniform Resource Identifier (URI), to access an OpenID enabled webserver. The message flow for OpenID is as follows.

1. **Visitor** $\rightarrow$ **Webserver** : Visitor submits OpenID
2. **Webserver** $\rightarrow$ **Visitor** $\rightarrow$ **IdP** : Visitor is redirected to IdP
3. **Visitor** $\leftrightarrow$ **IdP** : Visitor Authentication
4. **IdP** $\rightarrow$ **Visitor** $\rightarrow$ **Webserver** : IdP sends the requested security token

First, the visitor submits his OpenID identifier to the webserver. Then, webserver uses the identifier to get the identity of relevant IdP and optionally shares a secret with the IdP. In step 2, webserver redirects the visitor to the discovered IdP. Then in step 3, the visitor gets authenticated to the IdP. Last, the visitor is redirected back to the webserver with the requested security token.

7.2.1.3 Kantara Initiative

Kantara Initiative[3] was founded chiefly by Liberty Alliance Project whose goal is to build open specifications for federated identity management. Liberty model uses Circle of Trust concept where two IdPs with their webservers can be federated [119]. The message flow for the original Identity Federation Framework (ID-FF) Browser POST profile is as follows.

[3]kantarainitiative.org

1. **Visitor** → **Webserver** : Visitor requests access
2. **Webserver** → **Visitor** → **IdP** : Visitor is redirected to IdP
3. **IdP** → **Visitor** → **Webserver** : IdP sends the requested security token

In the first step, the visitor accesses the webserver and the webserver gets the identity of IdP. In step 2, the visitor is redirected to IdP where he is authenticated and then redirected back to the webserver in step 3.

7.2.2 Proposed Web Metering Scheme

This section proposes a web metering scheme utilising existing authentication mechanisms used by the visitor to incorporate a web metering function. The scope of interesting web metering functions that could benefit from the leverage of authentication providers is more than a participation in the web metering operation. Possible general approaches are described then there is a web metering scheme description which uses a specific identity management system to provide web metering evidence. This work was partially published in [1, 2].

7.2.2.1 Possible Approaches

The proposed scheme depends on protocols that are already used between the visitor and the metering provider and have padding capability to carry web metering evidence transparently to the visitor. Such "flexible" ticket-based protocols are authentication protocols described in ISO/IEC 10181-2 and ISO/IEC 9798 where tickets (or vouchers) can be redesigned to carry the evidence. For example, identity management systems and Kerberos protocol fit in this category and we use here a current identity management system. In these authentication protocols, a third party authenticates the visitor and embeds the web metering evidence in a ticket which is sent to the webserver. The addition of a web metering feature is possible here because ISO/IEC allows additional information to be carried in *text* fields in authentication protocols [42]. The scope of these possible solutions is limited e.g. a casual visitor who is not using an authentication mechanism is excluded.

7.2.2.2 Identity Management Systems

While studying many identity management systems, we observed background communications between webserver and IdP can be used as or to carry web metering evidences for visits to webserver. These web metering evidences are based on the collaborative work and communications done between the webserver and the IdP. The IdP is regarded here as a web metering service provider that provides visit evidence transparently to visitors.

The proposed scheme can also utilise the ability to extend attribute statement in SAML messages. This feature affects accuracy requirement as well because granularity of metered data can be increased by webserver extending the requested attributes in the attribute statement. Regarding integrity of web metering, there is a cryptographic binding of assertion and IdP through a digital signature and since IdP is a trusted entity, the signature will enhance the reliability of the metering process.

7.2.2.3 Proposed Solution

The following are the three phases for the proposed web metering scheme using a claim-based identity management system.

Initialisation Phase

– A visitor has to register at the IdP and obtain an Information Card. This procedure requires communication messages between the visitor and the IdP and later storing the Information Card at the visitor computing device.

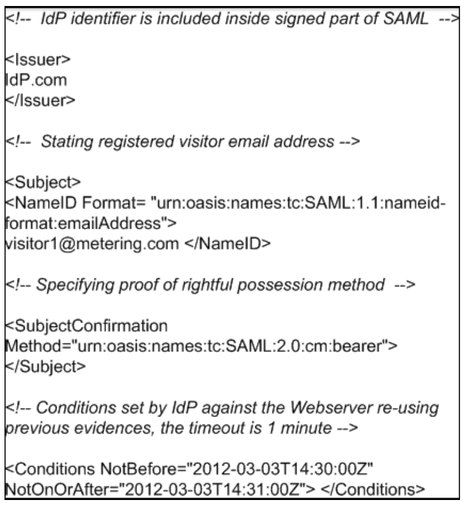
– Security policies at the webserver and the IdP have to include the required authentication information. That is, policy at the webserver has to detail the claims it needs, token format and what IdPs are trusted. The token format could include an unpredictable identifier chosen by the webserver. Requested visitor attributes are variable (e.g. date of birth can be requested to determine the quality of visits).

– Both the webserver and the IdP create public and private key pairs and then their public keys are authentically exchanged.

Metering Phase

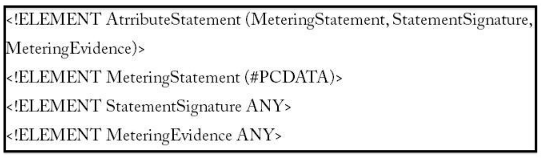
In step 1, the webserver policy, listing the requested claims inside the object tag (e.g. "trusted IdP is Yahoo!" and "visitor date of birth is required"), is downloaded into the visitor device. (Relevant web metering information can be provided by extending the attribute statement in SAML as described in 7.2.1.) The visitor chooses the right Information Card and sends a token request to the relevant IdP in step 2.

In step 3, the visitor is authenticated to the IdP using the stored username and password for additional visitor transparency (or X.509 certificate if available). In step 4, the IdP creates a SAML 2.0 token and lists all the requested claims and fields in the attribute statement. After that, the IdP signs the token using the IdP private key. The relevant SAML fields inside the assertion are shown in Figure 7.6 and a Document Type Definition (DTD) for the web metering tags is shown in Figure 7.7. In step 5, the visitor forwards the received token to the webserver. The webserver checks the signature and, if valid (which means that the token was sent by IdP and there was data integrity), the webserver stores the token.



```
<!--  IdP identifier is included inside signed part of SAML  -->

<Issuer>
IdP.com
</Issuer>

<!--  Stating registered visitor email address -->

<Subject>
<NameID Format= "urn:oasis:names:tc:SAML:1.1:nameid-
format:emailAddress">
visitor1@metering.com </NameID>

<!-- Specifying proof of rightful possession method  -->

<SubjectConfirmation
Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
</Subject>

<!-- Conditions set by IdP against the Webserver re-using
previous evidences, the timeout is 1 minute -->

<Conditions NotBefore="2012-03-03T14:30:00Z"
NotOnOrAfter="2012-03-03T14:31:00Z"> </Conditions>
```

Figure 7.6: Web Metering information in SAML Token



```
<!ELEMENT AtrributeStatement (MeteringStatement, StatementSignature,
MeteringEvidence)>
<!ELEMENT MeteringStatement (#PCDATA)>
<!ELEMENT StatementSignature ANY>
<!ELEMENT MeteringEvidence ANY>
```

Figure 7.7: DTD for Web Metering Tags

Post Processing Phase

After the webserver receives and stores the assertion, the webserver constructs the web metering evidence in a readable format, preserving the visitor privacy as specified. After that, the WMEP queries the webserver as it has the web metering results and relevant IdPs can be referenced in case the webserver results are not trusted.

There are two requested fields in an attribute statement: MeteringStatement and StatementSignature. The MeteringStatement field represents a statement from the IdP, assuring that the webserver has been visited at a specific time. Optionally, the statement can include further details about the visitor. The StatementSignature is a signature on the Metering Statement using the IdP private key. Those two statements can be used to reveal web metering results. The Metering Statement can be published by the webserver to interested parties such as WMEP and the Statement Signature serves as an evidence. An example of web metering evidence extracted at the webserver is shown below.

– **MeteringStatement** = *IdP.com assures that a visitor over 18 years old has accessed webserver1.com at 14:29 on* 3^{rd} *of March 2012.*
– **StatementSignature**= Sig_{IdP} (MeteringStatement)
– **MeteringEvidence**= MeteringStatement | StatementSignature

7.2.3 Analysis

In this section, we analyse the proposed scheme using the following four web metering properties: integrity, privacy, accuracy and efficiency. The first two security properties (integrity and privacy) address the web metering threats and the next two requirements (accuracy and efficiency) specify how web metering schemes should operate.

7.2.3.1 Integrity

Proposition 4. An adversary cannot produce a valid web metering assertion for a fake visit.

The following is the proof for Proposition 4.

Proof. We assume that number of corrupt users is small as done in [11]. By that assumption and using standard users' certificates, an adversary is not able to impersonate a valid user because he is not able to access or frequently guess the user authentication credentials. Also, webserver cannot be impersonated as the intended webserver name is already included in the signed assertion. Also, a replay attack is not successful as the time (real or logical) is included in the assertion. Therefore, an adversary cannot successfully produce a signature using a standard XML- signature scheme [47] for a web metering statement of a fake visit.

□

Data Integrity

The IdP signature on the token provides data integrity, both evidential and communication integrity, for the token. On the other hand, there is no communication integrity for downloaded security policy to visitor; however, identity selector is triggered after the webserver identified itself using its public key. Also, there is no communication integrity for the token request sent from visitor to IdP. As a result, a man in the middle attack is unlikely to happen in the proposed scheme because the adversary is unlikely to be able to produce a valid assertion for a modified message.

Scyther Validation.

In addition to the previous analysis, Scyther tool was used to further validate the proposed scheme as follows.

```
/* Validated protocol for a webserver claiming a fresh nonce included in
a assertion by audit agency after user being authenticated. */
usertype Bool;
protocol AAauthenUser(webserver,user,AA)
{
        role webserver
        {
        fresh webserverID: Nonce;
        var assertion;
        /* Webserver generates a nonce per its security policy during
        the initialisation phase in section 7.2.2.3. */
        send_0(webserver,user,{webserver,webserverID}pk(AA));
        /* Webserver sends the nonce to the user encrypted using
        AA public key as part of the webserver policy. */
        claim_webserver(webserver,Secret,webserverID);
        /* Webserver claims that the chosen webserverID is included in
        a signed assertion following user authentication. */
        recv_4(user,webserver, {assertion}sk(AA));
        }

        role user
        {
        var assertion;
        var authenticated: Bool;
        var webserverID: Nonce;
        var AAnonce: Nonce;
        recv_0(webserver,user,{webserver,webserverID}pk(AA));
        recv_1(AA,user, AAnonce);
        // User receives a nonce from AA.
        send_2(user,AA, {AAnonce}sk(user),{webserver,webserverID}pk(AA));
        /* User encrypts the received nonce using his private key and
        forwards the received webserver token. */
        recv_3(AA,user, authenticated, {{assertion}sk(AA)}pk(webserver));
        send_4(user,webserver, {assertion}sk(AA));
        }

        role AA
        {
        fresh assertion;
        fresh authenticated: Bool;
        var webserverID: Nonce;
        fresh AAnonce: Nonce;
        send_1(AA,user, AAnonce);
        recv_2(user,AA, {AAnonce}sk(user),{webserver,webserverID}pk(AA));
        send_3(AA,user, authenticated, {{assertion}sk(AA)}pk(webserver));
        /* AA formulates an assertion by signing a timely metering
        statement including webserverID and sends the assertion to
        the user. */
        }
}
```

Figure 7.8 shows an overview of the roles and their connections. Figure 7.9 characterises the roles with the webserver claim of having a chosen secret identifier to be embedded in an assertion.

This analysis can also be readily reused to lead to more attacks in other frameworks with different assumptions. Although the model can be generally fixed to eliminate attacks under stronger requirements by including necessary cryptographic mechanisms, such introduction will affect the transparency of the schemes. However, reusing existing solutions can guarantee that usability aspect of the scheme. Both types of the used analysis mainly confirm the security benefits of properly using digital signatures and interactively authenticating the user.

7.2.3.2 Privacy

Privacy is not satisfied for the proposed scheme because the visitor has to register and later get authenticated to the IdP for each visit. But by limiting requested

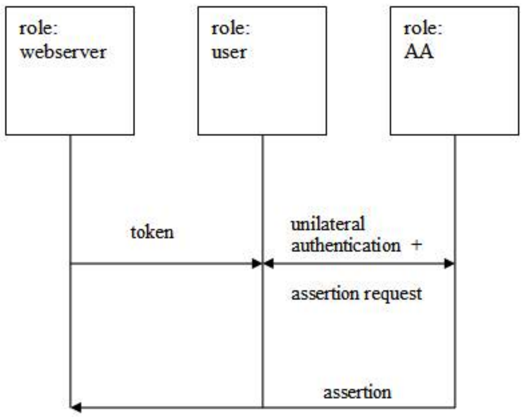


Figure 7.8: Roles And Messages

claims, the visitor can still be anonymous to the webserver. As for the adversary, he is unlikely to be able to obtain assertions sent from IdP to visitor or from visitor to webserver because used channels are encrypted.

7.2.3.3 Accuracy

Accuracy of Evidences

By using a standard signature scheme in the proposed solution, the solution provides an accurate number of visits based on the number of received assertions.

Granularity of Metered Data

The assertion in the proposed scheme can include further information about the visit (e.g. date) which satisfies granularity of metering requirement. Furthermore, the attribute statement can be used to include any further information regarding the visit or the visitor. For example, identity identifiers (like visitor age) can be requested by the webserver to be included in the assertion.

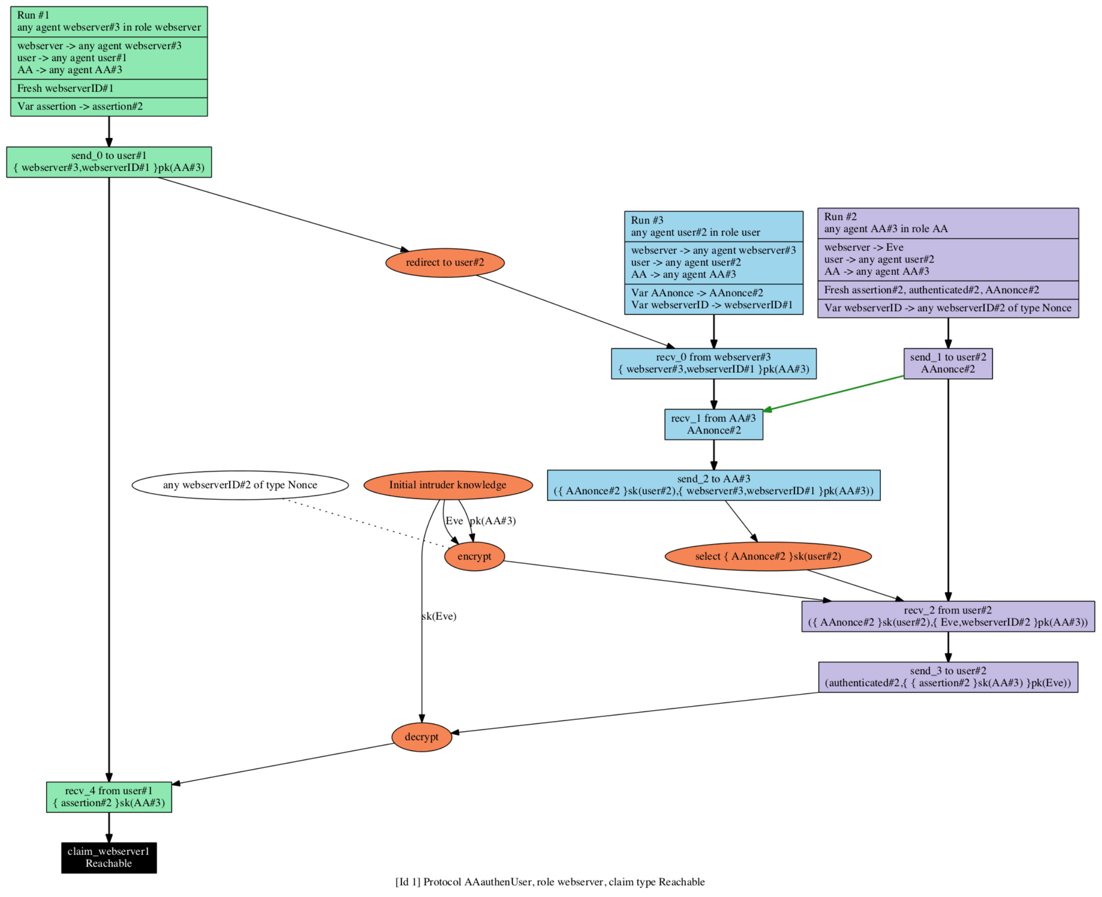

Run #1
any agent webserver#3 in role webserver
webserver -> any agent webserver#3
user -> any agent user#1
AA -> any agent AA#3
Fresh webserverID#1
Var assertion -> assertion#2
send_0 to user#1
{ webserver#3,webserverID#1 }pk(AA#3)
redirect to user#2
Run #3
any agent user#2 in role user
webserver -> any agent webserver#3
user -> any agent user#2
AA -> any agent AA#3
Var AAnonce -> AAnonce#2
Var webserverID -> webserverID#1
Run #2
any agent AA#3 in role AA
webserver -> Eve
user -> any agent user#2
AA -> any agent AA#3
Fresh assertion#2, authenticated#2, AAnonce#2
Var webserverID -> any webserverID#2 of type Nonce
recv_0 from webserver#3
{ webserver#3,webserverID#1 }pk(AA#3)
send_1 to user#2
AAnonce#2
recv_1 from AA#3
AAnonce#2
any webserverID#2 of type Nonce
Initial intruder knowledge
send_2 to AA#3
({ AAnonce#2 }sk(user#2),{ webserver#3,webserverID#1 }pk(AA#3))
Eve
pk(AA#3)
encrypt
select { AAnonce#2 }sk(user#2)
sk(Eve)
recv_2 from user#2
({ AAnonce#2 }sk(user#2),{ Eve,webserverID#2 }pk(AA#3))
send_3 to user#2
(authenticated#2,{ { assertion#2 }sk(AA#3) }pk(Eve))
decrypt
recv_4 from user#1
{ assertion#2 }sk(AA#3)
claim_webserver1
Reachable
[Id 1] Protocol AAauthenUser, role webserver, claim type Reachable

FIGURE 7.9: A Scyther User Trace Patterns

7.2.3.4 Usability

We tested a simplified version of the scheme on a server with 1024 MB memory and 1.6GHz processor. This server was hosting both the webserver and the IdP which run OpenID. The signature on the metering statement is produced using open source PKCS#1 (v2.1) RSA compliant library[4]. Execution time from IdP side is around 40 seconds and generated evidence is 204 characters. The inefficiency of the proposed scheme centres around the on-the-fly digital signature from IdP side and the dynamic text (as time is included) that has to be signed.

[4]www.phpseclib.sourceforge.net

As a result, the usability can be improved by signing a smaller pointer message (e.g. a number instead of the full metering statement) which can be used as a signed reference at the webserver side. Alternatively, to avoid any delay of the transparent browsing experience from audit agency side, the webserver could allow the visit and periodically rerun the scheme. We detail in the rest of this section the four different usability aspects.

Transparency

A web metering scheme is transparent if it executes inside or behind another action or property in web interaction so it does not require an explicit action from visitor. The proposed scheme is transparent to the visitor because he does not need to be aware of the metering operation. However, the user has to explicitly register with an IdP prior to the metering operation.

Computing Resources

The visitor needs computing resources for the following actions. Initially in a claim-based identity management system, the identity selector has to process the webserver request and uses the included webserver policy to match Information Cards from which the visitor chooses. The webserver request has to be stored in memory until it is included in the request to the IdP. Then, the identity selector constructs a security token request to the IdP. Then, the identity selector enables the visitor to get authenticated to IdP. After that, optionally, the received assertion is shown to the visitor for his consent to be forwarded to the webserver. As shown, there are many required computations at the visitor side but the relevant metering computation for this scheme is the extra computation regarding requested metering data in webserver policy.

After the webserver receives the visitor request, the webserver identifies itself using its public key and sends the webserver policy. Lastly, the webserver checks the signature of the received assertion and decides on whether to allow the visitor to access the requested resource. If granted, the webserver extracts the fields which will be used as web metering evidence. Assuming small number of requested claims, relevant metering computations for webserver are limited to specifying and later extracting requested claims and web metering evidence.

Regarding the IdP, it checks the visitor request against its policy and authenticates a visitor. Then, the IdP populates the token (e.g. a java program which extracts relevant fields from visitors database and fills the token). Also, the IdP constructs MeteringStatement, StatementSignature and MeteringEvidence. After that, the IdP produces a signature on the token. For OpenID, processing time at IdP was around 40 seconds.

Communications

The visitor needs to send and receive the following data. Initially, the visitor retrieves webserver and IdP policies and then sends a token request to IdP (webserver policy includes the additional metering request). After that, the visitor sends authentication credentials and then receives an assertion from IdP. Last, the visitor forwards the assertion to the webserver. The received and forwarded assertion contains the additional web metering evidence.

Regarding the webserver, it receives initial visitor request to access the webserver. Then, the webserver sends the webserver policy to the visitor. Last, the webserver receives an assertion from the visitor.

Regarding IdP, it sends its policy and later receives a token request from the visitor. Then, the IdP receives authentication information from the visitor and then sends the required assertion. MeteringEvidence of 204 characters containing MeteringStatement and StatementSignature has to be communicated from the IdP to the visitor and then to the webserver. As a result, the proposed scheme is communication efficient as the sizes of sent and received messages are typically small and have to be carried inside SOAP messages.

Storage

The visitor in the proposed scheme has to store Information Cards at his machine. Also, the visitor may need to store authentication information e.g. an X.509 certificate. The webserver has to store the webserver policy, assertions of 204 characters as evidences and its public key. The scheme is storage efficient as required size of outlined elements are typically small and limited and the storage requirements do not increase dramatically with the number of visits.

7.2.3.5 Summary

The proposed scheme can still run transparently and utilise security properties available in identity management systems or generally authentication protocols. An obvious limitation of the scheme is the existence of an authentication protocol between the user and the audit agency that can be used for web metering purposes. The on-the-fly signatures of assertions can affect the scheme efficiency with regard to required computing resources and consequently running time. Privacy is a concern in environments where the audit agency is colluding with webservers to link users' visits. Table 7.1 provides a summary.

Table 7.1: Limitations

Strength	Weaknesses/Challenges
• Security	• Privacy
• Accuracy	
• Usability (to some extent)	

7.2.4 Conclusion

In this section, we proposed a web metering scheme by utilising an identity management system to transparently carry a web metering evidence. We described a scheme that uses a generic claim-based identity management system to incorporate a web metering function. The scope of other interesting web metering functions that could benefit from the leverage of identity providers is more than the produced Statement Signature e.g. identity providers can be audit agencies that publish web metering results.

Transparency is the main obstacle in usability requirement as the user needs to have a prior relationship with the IdP. Required computing resources at the user side and exchanged messages were minimal which can provide fast unnoticeable implementation. There is also no new required storage requirements at the user side.

We used an established threat model to point to the gap between previous schemes and desired ones that motivated the proposed scheme. However, we believe that visitor privacy is still a challenge especially in environments where service providers are not trusted not to collude with webservers to link visitors identities to their visits. We formally analysed the proposed scheme using Scyther and we further confirmed the results by going through each achieved property. We plan to make a complete automatic validation of all security properties with different assumptions. Future work also includes configuring the proposed solution with different requirements. For example, to improve the scheme efficiency in the case when WMEP can only query IdP for web metering evidence, a signature on the Metering Statement is not required.

Part III

Privacy-Preserving Schemes Using Third-Party-Centric Analytics

This part proposes two transparent web metering schemes that improve privacy while providing "good enough" accurate results, compared to previous schemes. The first chapter proposes a generic scheme that securely captures different data about the users, in different scenarios, in a privacy-preserving manner. The second chapter proposes a special case scheme that collaboratively captures data about users using an embedded content technique.

Chapter 8

A New Scheme Using A Privacy-Preserving Conventional Data Analyser

Contents

This chapter describes third party web metering schemes, that transparently collect data about the user. The methodology was to explore the different types of users' data to enhance the accuracy of web metering result in a privacy-preserving manner. Relevant published results were also highlighted to be reused for web metering purposes. This chapter starts by providing background information and describing the problem and the contributions. Then, related schemes, in third party web metering scripts, mining user interaction and potentially privacy-preserving fingerprints, are described. Then, a new scheme, with its three scenarios of using users' "non-private" data, is described. After that, analysis is provided from three perspectives. Last, a conclusion is provided.

8.1 Background

It is desirable to have a web metering scheme, that can transparently produce accurate number of unique users and can still preserve users' privacy. In order to achieve such balance, the scheme has to collect more data about users in a privacy-preserving manner. Such desirable accuracy requires a higher scale of evidence. In an ideal scenario, users are authenticated to webservers and consequently the scheme can securely determine (and claim) number of unique users. However, entity authentication and privacy are inherently conflicting requirements.

Certain webservers (e.g. electronic mail providers) already have "authentic" users so that they can figure out number of unique users. To further confirm authenticity of users, it is common today for webservers to interactively verify users' accounts using other personal identifiable information (like mobile numbers). In some extreme cases, they further request a scanned version of user's photo ID[1]. The problem of figuring out number of unique users becomes challenging when the users do not have verifiable accounts with the webserver.

Third party analytics web metering schemes depend on the following three techniques: fingerprinting, probing and tracking. We define fingerprinting and probing as follows.

Fingerprinting And Probing. Fingerprinting is a passive activity that examines the after-interaction remains while probing is an active communication to extract specific user's information, which can still be executed transparently to the user.

For example, a probed *Media Access Control (MAC)* address is an actual data about a certain user's computer. However, mouse movements over a webpage is "fingerprints" left after a user's visit.

User tracking is defined in section 5.4.2.

Privacy Trade-offs. Basic web metering schemes reveal number of users' visits with no information about the visits themselves. However, determining quality of the visits is desirable for various applications e.g. advertising. However, it is a trade-off, the more information collected, the more likelihood of invading user's

[1]https://en-gb.facebook.com/help/385569904840341

privacy. Similarly, authenticating users during web metering operation helps significantly in evidence verification against number of unique users. However, entity authentication and privacy are inherently conflicting requirements.

Cookies. A cookie [73] is a text file set by the webserver or audit agency in the user's device. The user then sends the cookie back to the webserver or audit agency in every request. The webserver would recognise the user by an ID number placed within the cookie. We do not consider *non-persistent* cookies in our proposed schemes. The following are five attributes in *Set-Cookie* header: cookie name, receiving domain and its path, *Secure* flag to require SSL and expiry date. Secure flag can be used against corrupt hijacking webservers problem, described in section 2.2. Another countermeasure is the use of *HTTPOnly* flag to protect against cross site scripting attacks. However, if this flag is not set, there can be cross domain requests using Javascript. There are also *Local Shared Objects*[2] which store extensive data about the user on a common directory that can be accessed by different browsers.

Cross-Origin Resource Sharing. The same origin policy at users' browsers would securely allow external webserver requests, only if they are from the same domain using the same protocol and port. Using Cross-Origin Resource Sharing (CORS) [117], the user's browser would send the request to audit agency with *Origin* tag (or header) set to the webserver. The audit agency cross checks the Origin tag and the requested content. If there is a match, the audit agency grants the request.

The scheme can transparently involve the user with the audit agency during his interactions with the webserver. If the audit agency receives a request from a user

[2]helpx.adobe.com/flash-player/kb/disable-third-party-local-shared.html

referred by the webserver script (specified in the origin tag) asking for a particular content, the audit agency checks the webserver and the type of request. There could be various levels of requests, with the highest requiring the webserver to redirect the user to the audit agency e.g. the webserver believes that the users are unique and have made too many visits.

A simple example for such pseudocode is as follows.

```
If a ‘‘normal" user visit, webserver references a basic capture
API at audit agency.

Else if
a revisiting user (e.g. cookie), webserver
references an API to capture further data about user
e.g. possible fingerprints about used device.

Else if
other unique users with repeated visits,
webserver references a redirection API and based on response,
webserver redirects user to audit agency.
```

In order for webserver to count number of users, the webserver needs to “invite” the audit agency during user’s presence. The web metering code execution has been forced to transparently run at the user side instead of the untrusted webserver.

8.2 Problem Description

Analytics web metering schemes (e.g. GA) collect data about the user in order to provide web metering results. This category of web metering schemes is third-party-centric since a third party (e.g. Google) collects users’ data and shares the results with the visited webserver. In addition, Google collects various device information for its services including hardware model and International Mobile

Station Equipment Identity (IMEI)[3]. Besides the security problem of corrupt webservers inflating number of visits, there are privacy concerns regarding the collected data and used technology. For advertising purposes, a desirable web metering result would be the ability to additionally tell an accurate number of unique users rather then the mere number of visits. However, to obtain such result, users' privacy still has to be preserved. On one extreme, some current schemes capture all possible data about users[4]. Then, however, the scheme attempts to identify unique users through coupling the announced access device with the browser. We believe the leverage of such basic identification method is worth further analysis.

There are various "new" devices capable of accessing a webserver, for example, Microsoft Xbox game console has an Internet browser to watch *Netflix*. There can be occasions where capturing some of such devices' data helps more towards accurately identifying a unique user compared to common devices like laptops. Elsewhere, Online Advertisers (like Doubleclick[5]) use similar analytic schemes and procedures to track users and flag corrupt webservers using cookies and users ' private data (e.g. IP addresses).

One reason for the many previous schemes addressing privacy and accuracy dilemma for major players (e.g. Google) is that those attempts would affect the effectiveness of their existing online advertising [107]. Consequently, it is hard for new schemes, with different assumptions, not to affect the profitable advertising business without providing convincing solutions that address that aspect. Also, there are various technical difficulties and security issues for interactions between webserver and audit agency, with the user in the middle. The *same-origin policy*

[3]www.google.com/policies/privacy/#infocollect
[4]www.heapanalytics.com
[5]www.google.com/doubleclick/

prevents scripts (e.g. delivered by the webserver) to access content at another domain (e.g. the audit agency), while collecting web metering evidence.

8.3 Contributions

In this chapter, we explore web technology options and look at different ways to implement "accurate" web metering analytics schemes in a privacy-preserving manner. The flow of different proposals started by embedding content (and code) from another cooperating webserver or audit agency (Chapter 9), to other sharing methods to finally a complete redirection to the audit agency. During designing the proposed web metering scheme, we explored various user's data that are "unlikely" to identify the user yet can improve the accuracy of web metering results in a transparent way. Furthermore, the proposed scheme does not require redesigning the referenced existing protocols. In the proposed schemes, we explored different scenarios of capturing many "non-private" data about the users and simple actions (like mouse clicks). We used simple users' fingerprints to detect distinctive data to improve the ability of securely counting number of unique users in a privacy-preserving manner. (Exploring more sophisticated OS fingerprints or signatures is left for future work).

We explored approaches and options with different merits so they can be incorporated into existing schemes (e.g. GA). We also analysed referencing techniques like Cross-Origin Resource Sharing (CORS) from web metering perspective and compared it to other methods like scripts calls (including JavaScript Object Notation with padding [JSONP]) and popups. We are excluding issues of the webserver manipulating the execution of a third party code (in case webserver is running the script itself) and one naive solution could be auditing the webserver to ensure some parameters are not changed e.g. $HKEY\_LOCAL\_MACHINE$.

We also explored different users' data that can be captured and analyse them from privacy perspective. The exploration of such data and capturing techniques addresses the trade-off between accuracy and privacy as the more accurate data collected, the likelihood of privacy invasion. In that regard, we explored different implementation scenarios with different security benefits. The proposed scheme and findings of the research can be used to securely enhance the privacy and accuracy of existing web metering schemes. Parts of this chapter were published in [3].

8.4 Related Work

8.4.1 Google Analytics

Google Analytics (GA) [60] is a web metering service offered by Google to analyse users' visits to webservers. Further to the description in section 4.3.1, the script references a library at Google-Analytics.com to be used at the webserver. During users' visits, referenced web metering code is loaded into the webserver script domain. The code is executed under the webserver control, setting a *webserver-owned* cookie [106] to track returning users to the webserver and not Google-Analytics.com. Then, the code extracts the user's assigned identifier in a cookie (set earlier or updated by the running script) and captures some user's data, all to be sent back to Google-Analytics.com.

Despite the privacy improvement of webserver-owned cookie of not figuring out users visiting different webservers incorporating GA script, returning users will still be identified to the webserver and google-analytics.com. Besides the cookie issue, the referenced code captures private data about the user e.g. user's Internet Protocol (IP) address.

8.4.2 Piwik

Piwik[6] is an open source service similar to GA, which captures data including user's Operating System (OS), browser type, location, keywords queried and visited webpages. Unlike Google Analytics, Piwik script sends the web metering results to the visited webserver, instead of Piwik servers. Such approach assumes no trusted entity (i.e audit agency) and shifts the trust from the "strategically and potentially powerful" metering provider to solely the visited webserver. Despite Piwik being recognised as compliant with the French and German data protection laws [7], [8], further analysis and improvements can be done.

Three French-compliant privacy recommendations were highlighted as follows. Piwik allows deleting used cookies, user's choice of being opted out and anonymises captured IP addresses. Piwik has the option of automatically deleting old users' data. Old data can be used in an aggregate fashion to provide some desirable web metering services e.g. unique users over a year. Piwik recommends deleting users' data logs of six months and over. To maximise the privacy benefits for "un-necessary" stored users' data, the period can be set as long as needed to generate the web metering result (e.g. 30 seconds) instead of the minimum purge value of one day. Evidence part of the generated result may contain private information or get correlated, and therefore should be set to be deleted as soon as used to convince enquirers. Piwik gives a choice to the user to opt out. A cookie is used to tell the visited webserver that the user does not wish to run Piwik. Although the cookie does not contain an identifier, the cookie still allows the adversary to track opt out users and could further identify users using data sent in the opt out request (e.g. User Agent and IP address). Alternatively, in order not to use

[6]www.piwik.org

[6]www.cnil.fr/vos-obligations/sites-web-cookies-et-autres-traceurs/outils-et-codes-sources/la-mesure-daudience/

[7]piwik.org/blog/2011/03/piwik-can-be-used-in-compliance-with-data-protection-laws/

opting out cookies and create a user profile, user's opting out decision should be done for each session, in "non-interfering" way (e.g. the webserver homepage clearly lists opting out link). Or, the webserver could ask users (e.g. as a popup) if they would prefer to opt out (if the user does not respond to the opting out preference request after a timeout, the web metering script positively does not execute for that session).

There has to be a third party to confirm or help producing the web metering results. Piwik with its current setup can be used for a webserver that would prefer to get web metering results without proving to others. That is a different goal, the goal we are looking for in this research is that the webserver can prove to the enquirers with the option of seeking help from an audit agency or metering provider. All our three scenarios achieve that goal with different properties. Trust can be elevated by choosing an audit agency over a metering provider.

Piwik anonymises users' IP addresses by not storing the most significant bytes, up to three bytes. However, with such privacy improvement, the "new" IP addresses cannot be used alone to provide accurate number of unique users because of the substantial increase of IP addresses collisions. Latest version of Google Analytics also supports such IP anonymity feature, by not storing the most significant byte[9]. Furthermore, Piwik has an option not to use cookies or users profiles. Anonymising users' IP addresses and disabling cookies together further affect the accuracy of unique users.

8.4.3 Interaction Mining

Users' tracking and mining users' interactions can be promising underlying techniques for web metering schemes, if executed in a privacy-preserving manner. A

[9]developers.google.com/analytics/devguides/collection/gajs/methods/gaJSApi_gat#_gat._anonymizeIp

relevant framework for mining user interaction was proposed in [110, 35]. In such web metering scheme, the webserver installs a tracking mechanism in the user's browser (or a cookie) which sends a request to a third party in a manner similar to script-based model. The third party assigns an identifier to the user and analyses the web usage as the user interacts with the webserver. There are seven steps for tracking the user described in [110], as shown in Figure 8.1.

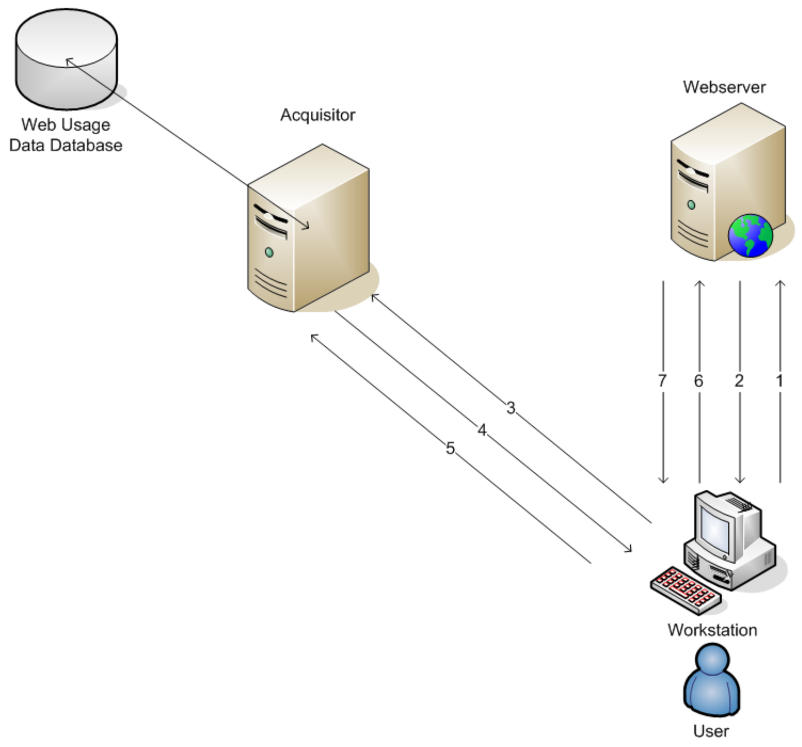


Figure 8.1: User Tracking And Interaction Mining Scheme

In step 1, the user sends a request to the webserver. In step 2, the webserver responds with the requested page including a tracking agent to the user's platform. In step 3, the on-the-fly installed agent connects to the acquisitor and requests an identifier. In step 4, the acquisitor replies with a unique identifier and the agent tracking applet is executed. In step 5, once the user sends another request to the webserver, the agent intercepts it and sends first the collected data about the previous delivered page to the acquisitor. In step 6, the user request is sent to the webserver and in step 7 the webserver responds with the requested page.

8.4.4 Browser Fingerprints

There are some distinct and unique characteristics of users' browsers. In some browsers, the uniqueness of their plugins exceeded 90% [51]. Even on tests of users accepting cookies, the fingerprints keep on changing over long periods e.g. 24 hours. (Despite certain properties keep changing, care must be taken into account, in privacy-preserving web metering schemes, so that the new changed version is unlikely to be linked to the previous version e.g. using randomisation techniques.) In some of their results, it has been shown that 37.4% of returning users had a change in their browser fingerprint. Such "non-aggregate" result has good implication in privacy (without the need of cookies) as fingerprints still seem not to possess "enough" uniqueness to be used as long term identifiers.

94.2 % of users in a sample [51] were determined if they were unique or not. It is unlikely of 94.2 % of sampled users to have the same browser plugins fingerprints matching another user's fingerprints [51]. Such "changing" uniqueness result about users' plugins helps significantly and is promising to be used in designing web metering schemes to count number of unique users. However, despite the designed or "natural" unpredictability, users can still be efficiently tracked without the need of cookies [95] e.g. by using *Flash* and capturing *navigator* and *screen* objects. Unfortunately, as per a sample [51], detecting changes in browser plugins had 65% success rate. Further to the browser plugins, in a manner similar to processing-based web metering schemes described in section 4.2, time required to execute certain instructions can be used to detect users' browsers [90]. There were also interesting cases with forged User Agents which could make unique fingerprints (which was a catalyst for anti-tracking observation in section 5.4.2). Also, from the P3P analysis summary in section 5.3.4, state metric is the least satisfied. Based on that data, the following observation holds.

Subtle "Re-identification" Observation. It is more likely for an adversary to be able to track a user than identify him. Without even persistent techniques (e.g. cookies), the user can still be tagged and tracked.

8.5 Proposals Outline

We propose a generic web metering scheme and three different scenarios. We explore different fingerprints of user's device and operating system. The exploration has been initially inspired by different approaches for example, Honeynet Project[10] has collected a relevant database of different operating systems fingerprints. In some situations, audit agency could use active probes similar to *nmap* if a repeated user is suspected of being actually a corrupt webserver e.g. *TCP Window Size* was changed. These fingerprints and scenarios with the existence of audit agency are covered in section 8.6.

– Redirection scenario will allow the audit agency to directly interact with the user (e.g. setting a first party cookie); however, it is assumed that the user visit experience does not change throughout the redirection. The redirection is desired so audit agency can confidently capture packets sent by the user and (passively) analyse them for possible fingerprints.

– Embedded scenario, detailed in Chapter 9, utilises a feature commonly used today in many forms. The proposal can further benefit from the ideas extensively explored by the discontinued Google *wave* product. Embedded

[10]www.honeynet.org

scenario follows the same script-based web metering model (e.g. GA) with the additional benefits of involving more referenced webservers compared to a single web metering script or content sharing methods (like CORS [117]) which can provide better security.

8.6 Proposed Web Metering Scheme

The proposed scheme involves users, a webserver and an audit agency which assumes webserver might fake web metering results. (The audit agency can be discarded if the webserver cannot be corrupt. However, the web metering problem we want to solve here involves an independent third party that can help producing and confirm webserver results. Further details are provided in sections 8.2 and 8.4.2.) In this section, we start by describing the proposed generic conventional[11] web metering scheme. Then, we further show three possible scenarios. For each scenario, we outline the underlying techniques. We fully describe and extensively analyse one scenario in Chapter 9.

8.6.1 Generic Privacy-Preserving Conventional Scheme

The following are the protocol steps for the proposed generic scheme.

1. **User** $\rightarrow$ **Webserver** : Access Request
2. **Audit Agency AND Webserver** : Obtaining User's Data
3. **User** $\rightarrow$ **Audit Agency AND Webserver** : Requested User's Data
4. **Webserver** $\rightarrow$ **User** : Result Of Access Request

In step 1, the user sends an access request to the webserver.

[11]We use the term conventional to mean "conforming or adhering to accepted standards, as of conduct or taste" [dictionary.com].

In step 2, The audit agency obtains user's data along the webserver to protect against corrupt webservers. The following are three alternatives for step 2 (however there could be more options).

- The audit agency and the webserver *captures* users' data and possibly stores anonymised data.
- The audit agency and the webserver analyses *fingerprints* from the user's request.
- The audit agency and the webserver *probes* the user's for further data.s

In step 3, the user sends any requested data. In order to provide accurate number of unique users, the web metering scheme depends on the following users' data (whenever applicable).

1. Announced OS. In HTTP packet, User Agent explicitly shows OS information.
2. Further OS Fingerprints. User's HTTP packets can contain information that could reveal information about the underlying OS e.g. TTL.
3. *Clock Skews* [77]. User's devices have distinctive characteristics e.g. the difference between reported time and real time as skews in the CPU clock.
4. Browser. User Agent explicitly shows user's browser. Besides, further browser fingerprints can be obtained as shown in section 8.4.4.
5. IP and possible source port range. IP header includes user's IP information (the proposed schemes does not store any users' IP addresses). The IP address could be used by the audit agency exceptionally to check corrupt webservers. TCP header includes port information, which can be used to further detect user's OS.
6. Internet Service Provider (ISP). ISP is the service provider the user uses to access the webserver. This information is included inside the *hostname*.

TABLE 8.1: User's Data Example

Data	OS	Fingerprints	Clock Skews	Browser	IP & Port	ISP	Timestamp	Peripherals	Nonce
Value	Windows NT 6.0 en-US	TTL=128 TCP Timestamp=0	N/A	Mozilla Firefox 5.0 Gecko	86...226 @ 2000	BT	17/07/03 16:17:07 -0400	(30,80) Onmousemove	723964

7. Timestamp. Timestamp is the time of user's visit to webserver or audit agency.

8. Peripherals.

   During typical user-webserver interactions, there are "universal" actions from users' peripherals (e.g. mouse movements), which can be used as a lightweight measure of the "liveness" of the user on a webserver [49]. Such typical actions, including mouse clicks, mouse movements and keyboard presses, indicate the user presence on the webserver and can serve a lightweight web metering evidence of the visit. Particularly, capturing the time spent by the user on a webpage is a step towards better figuring out its importance and potential marketing campaigns. Also, further developed concepts (e.g. eye tracking techniques [59]) can be deployed for web metering purposes to check whether the user has actually looked at an advertisement or not.

   Only mouse or keyboard peripherals' data are obtained. For example, in some scenarios, the user's mouse pointer coordinates at a particular moment. "High information load" peripherals that can be used for a browsing activity (e.g. a microphone) are excluded. Google App[12] accepts commands through voice, which potentially carries more data about the user than simple mouse movements.

9. Nonce. Nonce is a random number generated by the audit agency.

Table 8.1 is an example for the users' data.

[12] www.google.com/mobile/ios/

8.6.1.1 Counting Number Of Unique Users:

Previous privacy-preserving web metering schemes (e.g. [32]) allow to count number of visits but a more challenging problem would be counting number of unique users. In order to address the challenging problem of counting number of unique users, we propose the use of a formula. The formula computes a solution, based on user's data, and compares it with previous ones to determine whether the user is returning, consequently identifying unique users. The audit agency and the webserver can assign weights to obtained data. The weights can be used to calculate the probability whether the user's captured data can classify the user as possibly returning or unique. Such concept of giving each user's attribute a weight (excluding nonces) can be used for determining how much we are confident that the user is unique; the higher the result, the more likelihood of user's uniqueness. Both audit agency and webserver can refer to a specific user at specific time by their results of calculation. A similar "good enough" approach of capturing similar users' data with more advanced weights assignment has been used in [15]. If webserver result is different from the audit agency, the audit agency disregards that web metering evidence for that user.

The proposed scheme will require both webserver and audit agency to compute a result of a formula that includes certain user's variables (with weights to potentially determine a threshold). Some of the variables in the formula include user's browser fingerprints, operating systems fingerprints, clock skews and possibly Internet Protocol addresses.

From *Subtle Re-identification Observation* in section 8.4.4, some browser fingerprints (and other users' data) should not be stored by the webserver, rather hashed, adjusted and used at that time to decide number of unique users. Otherwise, the fingerprints can be inferred by an adversary and linked to a particular user. Even with such attack, the user is still not identified; however, the user can be tracked using captured fingerprints without cookies.

In order to implement the proposed scheme in a secure and privacy-preserving manner, there are issues that need to be addressed by implementing different techniques and functions, as follows.

- *Corrupt Webserver Check.* This audit agency check concerns a webserver impersonating users. The audit agency ideally stores a hash of user's browser objects, Flash and masked IP, and maintains an anonymised record of "recent" users. The privacy-preserving check can be more complex and changes the requested users' data frequently to make it costly for webservers to fake users.

  Flash can be used in Corrupt Webserver Check technique to get user's OS data. After Corrupt Webserver Check technique *flags* a possible fake user, audit agency cross checks its hashes database for the next users. If there is a match, the audit agency increments a counter for the returning user and uses flash to get specific user's data e.g. OS build number. Once the counter reaches a threshold within a specified period, the audit agency further investigates possibility of the corresponding webserver being corrupt.
- *User's Data Obtaining.* This technique concerns obtaining pre-specified user's data (Table 8.1) without permanently storing them. The technique is done in a way that ensures user's privacy is preserved yet a version of the obtained data can be later used to tell the number of unique users.
- *Unique User Check.* This technique attempts to figure out number of unique users within a specific period (e.g. 10 minutes) without invading users' privacy. With small number of corrupt users assumption [11], we also assume that users would not alternate between their typical browsers with TOR

browser within that particular timeframe. The audit agency and webserver on-the-fly calculate weights for the result of User's Data Obtaining technique in order to compute a specific formula. The following is a simple algorithm for such check to produce number of unique users.

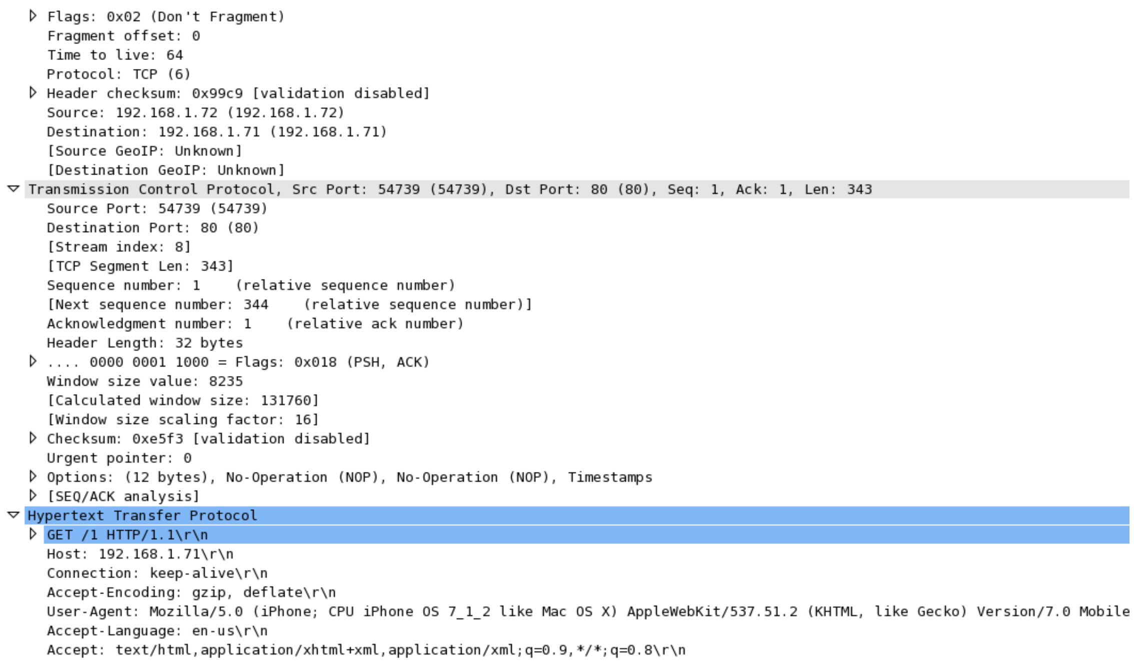
```
1: procedure UNIQUE_USER($Threshold$)
       $Calculated\_Weight$=0
       $Register$=First obtained record
2:     while $Calculated\_Weight > Threshold$ OR
       there is a record do
3:         if $Register$ = corresponding stored record then $Register \leftarrow$ next
   obtained record
4:         else $Calculated\_Weight = Calculated\_Weight$ - $Lookup\_Weight$
       $Register \leftarrow$ next obtained record;
5:         end if
6:     end while
7:     if $Calculated\_Weight > Threshold$ then $User\_Counter = User\_Counter$
   +1
8:     end if
9: end procedure
```

On a high level, the algorithm works as follows. First, threshold, *Threshold*, is set to prespecified value (say -4) and proposed formula weight, *Calculated_Weight*, is reset. Also, first obtained user's record (e.g. OS) is loaded into a memory register for processing. Then, while the weight does not exceed the threshold or there is still current user's data that can be compared to previous ones, the following conditional check is executed. If current user's record equates the stored one, the algorithm obtains the next user's record for comparison (without affecting the weight). Otherwise, the weight is decremented by the corresponding weight in Table 8.2, *Lookup_Weight*, and the algorithm obtains the next user's record for comparison. Last, if there is a user's data that matches the current user's data satisfying the formula, within a threshold, the user's counter field is incremented by one.

We did a few simple tests for visits from seven users of different platforms, with some as returning. The sample covered visits from different times with recurrent ten minutes web metering period. In order to confirm the results, we used network sniffing tools particularly Wireshark[13]. The use of network sniffers (compared to webserver log) show all traffic en route to the webserver without any possible "eliminations" (e.g. through firewalls). The weights calculation achieved high success rates for determining number of unique users. Seven different platform tests achieved 100 % accuracy from typical visits in different settings. However, a normal visit followed by another (within the timeframe) using a TOR browser would fail the simple calculation. It is reasonable to assume that if a user uses a TOR browser, he is unlikely to frequently keep toggling with other browsers (within a short period of time e.g. 10 minutes). However, with small number of corrupt users assumption [11], another check can be done for probable visits using TOR browsers. This additional check could be occasionally executed to flag possibly altering disguised users within a specific timeframe (in Corrupt Webserver Check technique). A screenshot for a network sniffer's typical result on a webserver is shown in Figure 8.2. The tool first shows the different TCP flags and ends with the user agent string. The outlined TCP flags' fingerprints are an example of users' data that can be captured for a limited period of time without identifying the users. Some of our (particularly) early observations are inspired by experimental tests to check signs of users'

[13]www.wireshark.org/

uniqueness with potentially users' "public" data. We used webserver network log, a network sniffer and data obtained through JavaScript to check potential unique data.



```
  ▷ Flags: 0x02 (Don't Fragment)
    Fragment offset: 0
    Time to live: 64
    Protocol: TCP (6)
  ▷ Header checksum: 0x99c9 [validation disabled]
    Source: 192.168.1.72 (192.168.1.72)
    Destination: 192.168.1.71 (192.168.1.71)
    [Source GeoIP: Unknown]
    [Destination GeoIP: Unknown]
▽ Transmission Control Protocol, Src Port: 54739 (54739), Dst Port: 80 (80), Seq: 1, Ack: 1, Len: 343
    Source Port: 54739 (54739)
    Destination Port: 80 (80)
    [Stream index: 8]
    [TCP Segment Len: 343]
    Sequence number: 1    (relative sequence number)
    [Next sequence number: 344    (relative sequence number)]
    Acknowledgment number: 1    (relative ack number)
    Header Length: 32 bytes
  ▷ .... 0000 0001 1000 = Flags: 0x018 (PSH, ACK)
    Window size value: 8235
    [Calculated window size: 131760]
    [Window size scaling factor: 16]
  ▷ Checksum: 0xe5f3 [validation disabled]
    Urgent pointer: 0
  ▷ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
  ▷ [SEQ/ACK analysis]
▽ Hypertext Transfer Protocol
  ▷ GET /1 HTTP/1.1\r\n
    Host: 192.168.1.71\r\n
    Connection: keep-alive\r\n
    Accept-Encoding: gzip, deflate\r\n
    User-Agent: Mozilla/5.0 (iPhone; CPU iPhone OS 7_1_2 like Mac OS X) AppleWebKit/537.51.2 (KHTML, like Gecko) Version/7.0 Mobile
    Accept-Language: en-us\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n
```

FIGURE 8.2: Webserver Captured Traffic

A sporadic experimentation covering different operating systems' updates, for almost a year, further confirmed the findings. We experimented, with the user's data, in a top-down fashion as from the general overloading and inconsistent data, down to the specific, and last to the fingerprints. The User-Agent is initially parsed[14] to specify all user's attributes. (Many values, within the User-Agent, can confirm the announced operating system. For example, *AppleWebKit* along *Safari* consistently confirm *Mac* operating system.) Then, other direct (e.g. probed) values (e.g. browser plugin details) are captured to narrow the obtained results from the User-Agent.

[14]The parsing should not be simply to search and extract the engine's identifier as some browsers might pretend to be "as" another e.g. "like Gecko" term.

TABLE 8.2: User's Data And Weights

Data Type	OS	TCP			Browser			ISP
		TTL	Timestamp	DF	$Hash$(plugins)	Version	Engine	
Weight	2	0.5	0.5	1.5	1.5	1	0.5	0.5

TABLE 8.3: Users' Data During Last 10 Minutes Example

Data Type	OS	TCP			Browser			ISP
		TTL	Timestamp	DF	$Hash$(plugins)	Version	Engine	
User 1	Linux	255	N/A	OFF	N/A	31	Gecko	Sky
User 2	Windows	128	N/A	ON	N/A	9	N/A	BT
User 3	Linux	64	N/A	OFF	N/A	34	Blink	Virgin

Last, fingerprints (like TTL) are used to further confirm previous findings. For example, if User-Agent lists MSIE 9.0, then the TTL should be less than 64. Further details about relevant tests for the generic scheme and some of its scenarios are provided in section 8.7.3.4.

The following is an example for the described technique. Assume the audit agency has already a record (e.g. a database), say currently 20 users. After capturing user's data, audit agency and webserver do the following operations offline. The audit agency and webserver compare current captured user's data with the stored record. If there is a match, audit agency and webserver compare the next data type. Once there is a mismatch, audit agency and webserver decrement a counter by the corresponding data type weight. Audit agency and webserver repeat the described comparison until the counter reaches a prespecified value (say -4) and then move to the next user. Once there is a case where the counter did not go below or equal to -4, audit agency and webserver both flag the current user as returning. Then, audit agency and webserver calculate weights formula to produce web metering evidence. Table 8.2 is an example for executing Unique User Check technique.

Assume audit agency and webserver have a record of 3 users, as shown in

Table 8.4: User's Data Codes And Evidence

Data	OS			TCP			Browser			ISP	
Value	Windows	Linux	Hex(others)	TTL	Timestamp	DF	$Hash$(plugins)	Version	$Hash$(Engine)	BT	Hex(others)
Code	2	3	1	1	2	3	1	2	3	2	1

Table 8.3. Audit agency and webserver already executed capturing user's data (steps 2 and 5 correspondingly). Assume captured current user's data are as follows: OS=Windows, TCP TTL=128, TCP DF=ON, browser version=10 and ISP=BT. Unique User Check counter for User 1 would be -6. As a result, Unique User Check counter is executed for the next user, User 2. Unique User Check counter for User 2 would be -1. So, User 2 is flagged as returning.

Table 8.4 is a simplified example for codes set by the audit agency after weights calculation. If the captured data is a numeric value then it is multiplied by the corresponding code. Otherwise, the code represents listed options or the data type is converted into hexadecimal. The code default value is set to 1. The following is the result of audit agency applying the codes to some captured data in Table 8.1. 2 (OS Windows) + 128 (TCP TTL) + 3 (DF) + 20 (browser version) + 2 (ISP BT) + 1707 (minutes and seconds of timestamp) + 723964 (nonce) = 725826. Both audit agency and webserver store 725826 as web metering evidence for User 2.

– *Result Storing.* This technique stores the result of the formula calculation by audit agency and webserver, in a privacy-preserving manner.

Steps 2 and 3 in the proposed generic scheme can be achieved in different ways and therefore we give three scenarios of which the audit agency and webserver can use. We instantiate the proposed generic scheme, by describing steps and techniques for each scenario. Such scenarios are possible variation and potential transparent applications of the proposed generic scheme. The three scenarios share many underlying techniques for example obtaining users' data. Examples of such obtained data and results of the tests are provided in Table 8.1, Figure 8.2, section 8.7.3.4 and Appendix A.

8.6.2 Scenario 1: Redirection-based

A summary of the assumptions we followed in this scenario is as follows.

1. Audit agency and webserver both can do web metering operations and correlate their results.
2. Redirection occurs transparently to the user.

The user will be redirected from webserver to audit agency, optionally through a *cross domain* cookie, and the formula result will later be cross checked between both entities to determine validity of collected evidence. (Redirection can be implemented by placing specific tags in HTML headers.) The following are the protocol steps in the proposed redirection-based conventional web metering scheme for steps 2 and 3 in the generic scheme.

1. **Webserver** → **User** → **Audit Agency** : Redirection
2. **Audit Agency** : Obtaining User's Data: Capturing \Fingerprinting \Probing User's Data
3. **Audit Agency** → **User** : Cookie
4. **Audit Agency** → **User** → **Webserver** : Redirection
5. **Webserver** : Obtaining User's Data: Capturing \Fingerprinting \Probing User's Data

The following are the corresponding steps for the proposed scheme.

1. Webserver redirects the user.

2. Audit agency obtains user's data.
3. Audit agency inserts a nonce into a cookie at the user's machine (possibly encrypted).
4. Audit agency redirects the user back to webserver.
5. Webserver obtains user's data and reads the cookie (possibly by decrypting it).

Besides the generic outlined techniques described in section 8.6.1.1, the following is an additional technique that the proposed redirection-based scheme can use.

Nonce Sharing. This technique complements Corrupt Webserver Check, further anonymises users' stored data and optionally organises web metering subscriptions with webservers. It includes setting or reading a cross domain cookie (if possible) containing a nonce by audit agency. The audit agency generates a nonce and stores it at the user side through a cookie. In case the cookie cannot be set by the audit agency (e.g. user's browser preferences), audit agency sends the nonce directly to the corresponding webserver. To prevent an adversary hijacking the cookie, the nonce can be encrypted using a symmetric key with the redirecting webserver. In case the nonce cannot be shared through the user (even through CORS), the audit agency sends it directly to the webserver upon the redirection. The included nonce will anonymise the web metering stored evidence.

Figure 8.3 shows message flow for the described steps. **Step a** corresponds to step 2, **step b** corresponds to step 4 and **step c** corresponds to step 5 in the described scenario. **Step a** is included to involve the audit agency in the web metering scheme and can be implemented using standard HTTP status codes e.g. *3xx* redirection status code. **Step b** is included to let the audit agency "track" or share values with webserver using standard HTTP headers e.g. Set-Cookie (further details are provided in section 8.1.) **Step c** is included to transparently return the user to the intended visit and can be implemented using the same mechanism of **step a**.

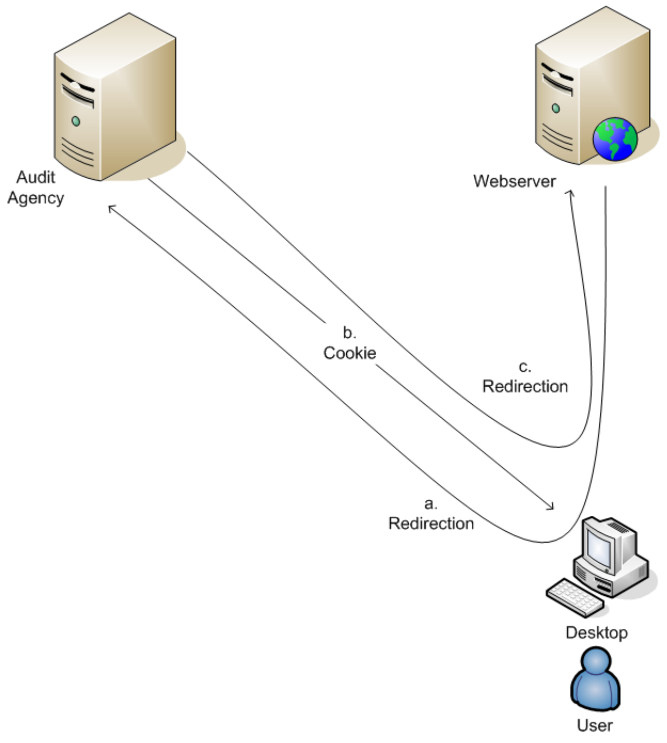


FIGURE 8.3: Conventional Redirection-based Web Metering

In step 1 in the proposed scenario, the webserver transparently redirects the user to the audit agency. In step 2, the audit agency executes Corrupt Webserver Check and User's Data Obtaining techniques. The audit agency generates a nonce and obtains data about the user to calculate a specific equation (Unique User Check technique can be done offline but without permanently storing the user's data.) In step 3, the audit agency executes Nonce Sharing technique. The audit agency encrypts a nonce using a pre-shared symmetric key and places the encrypted version in a cross domain cookie at the user's machine. In step 4, the audit agency redirects the user back to the webserver. In step 5, the webserver executes Nonce Sharing and User's Data Obtaining techniques. The webserver reads the cookie to decrypt the nonce. The webserver also obtains specific data about the user in order to execute the specific equation (Unique User Check technique can be done offline but without permanently storing the user's data).

8.6.3 Scenario 2. Script-based

A summary of the assumptions we followed in this script-based scenario is as follows.

1. There is a referenced web metering script at audit agency.
2. User's platform allows the execution of the web metering script e.g. JavaScript.

In this scenario, compared to scenario 1, the audit agency can be the main web metering provider. The webserver in that case does not need to redirect the user to the audit agency rather "securely" references a script at the audit agency side (in the same way Google Analytics operates as described in section 8.4.1). (Alternatively, in a likely non-transparent way, the webserver can send a secure script e.g. signed.) The script collects "non-private" data about users and produces evidence for number of unique users to interested enquirers. The nonce does not need to be generated by the audit agency and cross domain cookies are not required (as in redirection-based scenario). As in User's Data Obtaining, users' peripherals data (e.g. mouse movements and browsing behaviour) can be obtained through the script to check the probability of the user actually seeing an advertisement and produce evidence. Further details about script-based scenario are provided in sections 4.3.1 and 8.4.2.

Besides the generic outlined techniques, the following is a summary of two additional techniques.

1. *Secure Script Loading.* <script> tag or CORS can be used in case the script is referenced at audit agency. Other "non-transparent" sharing and tracking

mechanisms could still be used [106] e.g. as popups. Also, the script could be sent by the webserver e.g. Piwik.

2. *Hash Storing.* This technique on-the-fly hashes some browser objects, OS and possibly masked IP, and stores the result, as web metering evidence.

The following are the protocol steps for the described scenario for steps 2 and 3 in the proposed generic scheme.

1. **Webserver** → **User** : Script reference
2. **Audit Agency** \Webserver : Obtaining User's Data

The following are the corresponding steps.

1. Webserver sends to the user a reference for a web metering script.
2. Audit agency or webserver obtains data about user by running the script.

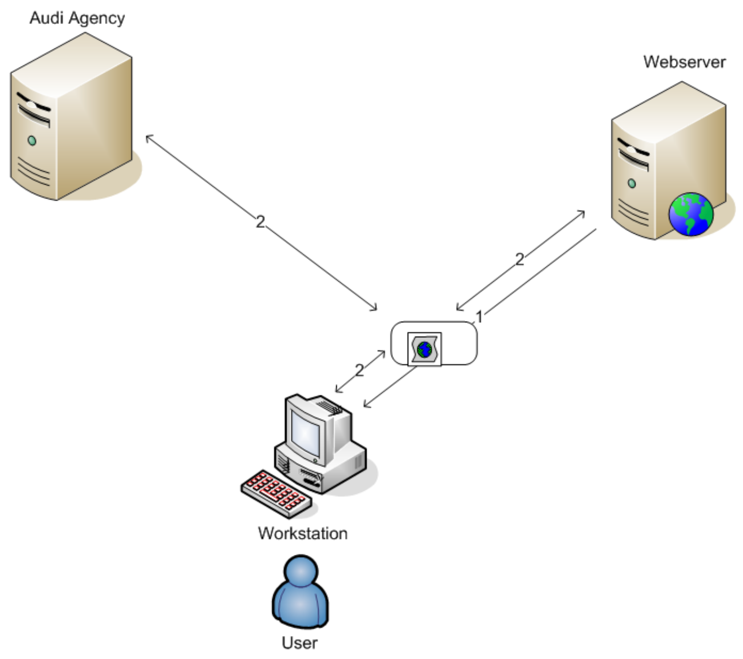


FIGURE 8.4: Conventional Script-based Web Metering

In step 1, the webserver executes Secure Script Loading technique. The webserver delivers the requested content, referencing a script at the audit agency side. In step 2, the script executes User's Data Obtaining, Unique User Check and Hash Storing. The user runs the script and the requested data are sent to the audit agency, the webserver or both. Figure 8.4 shows the message flow for the described steps.

Different ways were explored to securely implement a whitelist of web metering scripts. The script can be run using <script> tag or CORS. A form of a singed script can also be used, if run transparently. The script on-the-fly is parsed and compared to secure hashes, possibly over HTTPS.

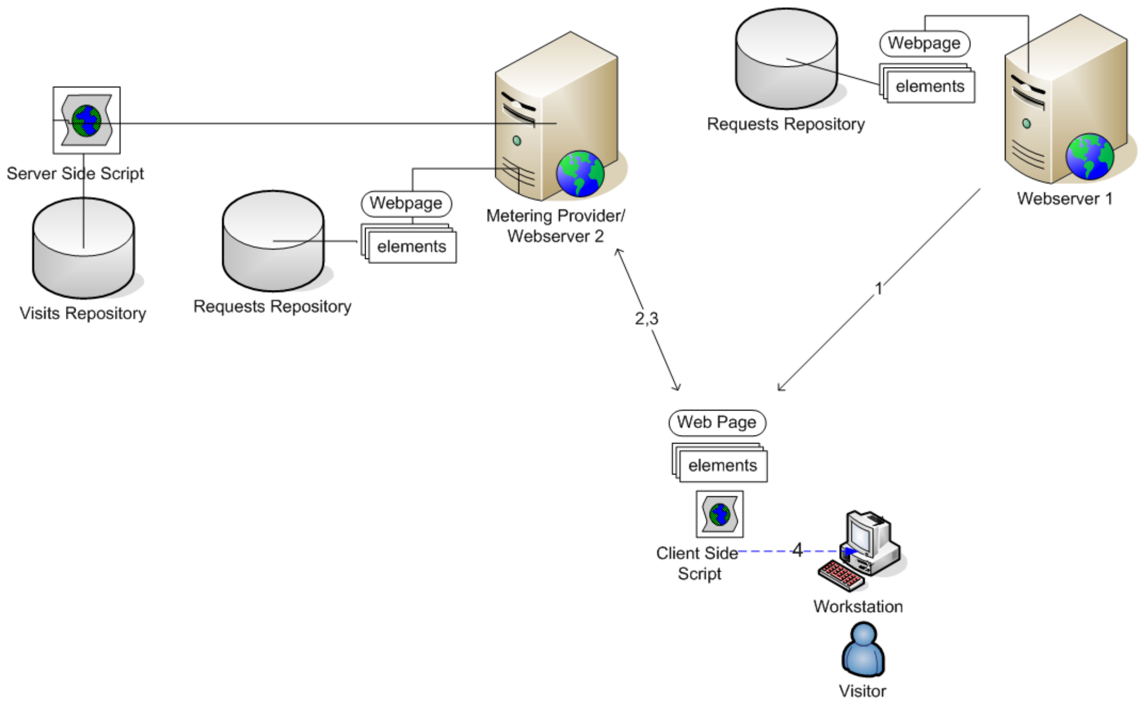


Figure 8.5: Conventional Embedded Web Metering

8.6.4 Scenario 3. Embedded-based

A summary of the assumptions we followed in this embedded-based scenario is as follows.

1. There are relevant embedding and web metering codes at referenced webserver(s).
2. User's platform allows execution of embedding and web metering codes e.g. JavaScript.

In this scenario, the web metering scheme utilises an embedding facility. Similar to script-based scenario, it starts by the original target webserver serving the users with its own webpages containing links to other webservers, as shown in Figure 8.5. Then, either automatically or triggered by the user, the links get rendered into embedded content. The embedded content codes periodically check user status (e.g. by capturing mouse movements using JavaScript) and executes User's Data Obtaining technique. The HTTP requests captured by the original webserver and user's data obtained by the embedded content webserver (and possibly the original webserver) are later correlated to provide the web metering result and evidence.

Besides the generic outlined techniques, the following is a summary of two additional techniques.

1. *Secure Codes Loading.* In this technique, embedding and web metering codes are downloaded transparently to the user.
2. *Hash Storing.* This technique on-the-fly hashes some browser objects, OS and possibly masked IP and stores the result, as web metering evidence.

The following are the protocol steps for the embedded-based for steps 2 and 3 in the proposed generic scheme.

1. **Webserver → User** : Requested webpage containing web metering and embedded codes
2. **User → Referenced Webservers** : Webpages requests
3. **Referenced Webservers → User** : Requested webpages containing web metering codes
4. **Webserver/Referenced Webservers** : Obtaining User's Data

The following are the corresponding steps for the proposed scheme.

1. Webserver responds with the requested webpage containing its web metering code and embedded code from other webservers.
2. Embedded code at the user side sends requests to the referenced webservers.
3. Referenced webservers respond with the requested content and web metering code.
4. The original webserver along the referenced webservers' web metering codes obtain non-private data.

Further details about embedded-based scenario are provided in Chapter 9.

8.7 Analysis

8.7.1 Security Analysis

The proposed generic scheme, with its three scenarios, can also utilise IP addresses, particularly Internet Protocol version 6 (IPv6) [94], for the IP data collected in Table 8.1, as follows. IPv6 has a potential to identify internet users from the source address field on the 40 bytes header where the source address field can optionally have Media Access Control (MAC) address. Consequently,if a user changes the network, the 128 bit will be changed. Such announced MAC address will enable audit agency to uniquely identify users and bar corrupt webservers from faking visits. Audit agency can correlate and use such data in Corrupt Webserver Check technique. Besides, obtaining simple IP addresses is not a secure approach as a low cost Honeyd[15] can be used to generate many fake IP addresses. Consequently, more accurate approaches that are based on checking "liveness" of the user (ranging from transparent navigational data to $CAPTCHAs$) can address such attacks.

Analysing fingerprints of users' requests in the proposed generic scheme, with its three scenarios, can help web metering schemes to uniquely identify users and possibly flag corrupt webservers (in Corrupt Webserver Check technique). An example of such fingerprint is the source port range, like Linux operating systems source port assignment which starts from 32768. Also, depending on the subsequent port assignment, some specific patterns could indicate an automated change. In the case where the audit agency is trusted, the use of *audit agency-owned cookies* can be a security measure set by the audit agency against corrupt webservers. Depending on the dominance or popularity of the audit agency, it might flag some returning users as potentially corrupt groups of webservers.

With the increased usage of *Flash* [122], the audit agency can also use Flash to collect data about users for web metering purposes. To improve the accuracy of the proposed web metering scheme, data collected from Flash can be combined with other captured "non-private" users' data in a privacy-preserving manner. Furthermore, Flash is commonly used today along JavaScript to fingerprint users' actions for fraud detection [95]. In particular, Adobe Flash was identified as a way to capture data about users. The proposed scheme can be leveraged for such other uses as the web metering script resides at the audit agency capturing "non-private" data about users. Also, besides NATing and Onion Networks

[15]www.honeyd.org

countermeasures described in section 5.5, users can use certain security browser extensions [122] that do not allow hidden Flash content to be executed.

With low adversary capability assumption [57], it is highly unlikely for an adversary to be able to predict the semi *random walk* [99] the user would take. We view the user's browser behaviour as a potential "random" event. This can get speculative with all current browsers and their different architecture and rendering techniques.

In redirection-based scenario, the cross domain cookie can be encrypted for the following payment scenario. Audit agency and webserver can share a symmetric key based on a web metering service subscription with an expiry date. Audit agency changes the nonce every redirection to ensure freshness and it is recorded along the web metering result and evidence. Nonces are used just to award that particular redirection visit to the corresponding webserver. In a less privacy-preserving manner, the audit agency can keep a record in the Unique User Check technique and set a lifetime for the nonce (e.g. 15 minutes), so it can link visits during this period. Alternatively, the audit agency can set two nonces in case the webserver has to calculate the formula for returning users before the redirection. The two nonces enable the possibility of the webserver to start the web metering process and only redirect the user if the webserver suspects a returning user. In such case, the webserver uses the first nonce for the current visit while the second one for the following visit formula's calculation.

8.7.2 Privacy Analysis

Property 1. A web metering scheme that captures certain unique users' properties, that by themselves "unpredictably" keep on changing, will not enable an adversary to track the users.

Justification.

An example of an unpredictable change is an automatic installation of a non-standard font upon a user's visit to a webserver having a new language. As the user continuously changes his "unique" fingerprints to a typical appearance, an adversary is unable to correlate such random changes. (Following anti-tracking observation in section 5.4.2, obtained properties with random changes have to be rounded to typical new settings). A simple example of such anti-tracking scheme is the random screen resolution upon *torbutton*[16] startup. The concept can be further illustrated as follows. The browser first reads previous settings and preferences upon its startup. Then, it generates random values and calculate differences with the stored ones. The browser applies the changes which should be minimal to not affect the user's browsing experience. In such"stenographic" appearance, the audit agency or webserver would collect unreal (blinded) yet useful users' information for web metering purposes. For example, the browser generates a random value between 0 and 0.1 and subtract the stored browser size from the generated value. The changes must be "random" (in a range) and yet have high probability of matching typical users configurations. As a result, it is still unlikely for the new changes to match another current user (within a short timeframe) and, hence, number of unique users can be determined.

Changing browser values can be detected since the browser rendering operation is still the same. Tests for changing user agents in *Internet Explorer* have successfully changed or masqueraded the values but still can be fingerprinted from the browser behaviour [55].

A summary of the P3P analysis for the proposed scheme is shown in Table 8.5. Further details about P3P categories are in section 5.3.4. We use a ✗to denote the scheme *requires* the corresponding data while a $\nmid$ indicates that a variation of

[16]torproject.org/projects/torbrowser.html.en

the scheme might require it. We use ✔ if the data type can be *protected* (i.e., the scheme does not need to obtain the corresponding data in order to implement the web metering techniques). There are three groups of possible web metering schemes: user-centric, webserver-centric and third-party-centric. For example, a non-transparent user-centric web metering scheme [93] can use Shamir secret sharing scheme [111] to provide accurate number of visits to a certain threshold.

Proposition 5. Proposed scheme protects users' identifiers, state and physical locations, and is still highly likely to produce accurate number of unique users.

The following is the justification for Proposition 5.

Proof sketch.

The proposed generic scheme with its three scenarios does not assign any sort of identifiers (or tags) to visited users e.g. using cookies. (The cookies can be used by the audit agency to agree on nonces with visited webservers.) Following Property 1, the common changing behaviour of users' browser plugins and a hash of users' data and a random nonce (even stored), the adversary will not able to track the user. The randomness makes two different users highly unlikely to generate same fingerprints across long number of bits (e.g. 20) of identifying information [51]. "Manipulating" the weights of captured data can further provide different levels of traces of captured state. Therefore, the scheme protects users' state information.

In the proposed scheme, users' data that could potentially identify users are captured users' fingerprints, computer data and Internet Service Provider (ISP). Users' fingerprints include users' communication and browser fingerprints which neither has identifying information, with the anonymising nonces and hashes. Users' computer data, including OS and browser, are hashed again with new nonces. ISP data only contains generic data about the provider, which ideally

Table 8.5: Privacy Comparison

Scheme	**Identifiers**	State	Interactive	Location	**Computer**	**Navigation**
Digital Signature [62]	✗	∤	✓	✓	✓	✓
Secret Sharing [93]	✗	∤	✓	✓	✓	✓
User Hardware [4] (DAA-based [25])	✓	∤	✓	✓	✓	∤
Webserver Hardware [13]	✓	∤	✗	✓	✓	✓
Processing-based [32]	✓	✗	✓	✓	∤	∤
HTTP Proxy [10]	∤	∤	∤	∤	✓	∤
Google Analytics [60]	∤	✗	✓	∤	∤	∤
Proposed Scheme	✓	✓	∤	✓	✗	✗

serves large sums of users. Such "heuristic" scheme uses the obtained users' data to distinguish visiting users and be able to figure out number of unique users. Since the proposed scheme cannot track the user and following Subtle Re-identification Observation, the same user is unlikely to be re-identified (as the hashes are highly likely to be different). Therefore, the scheme protects users' identification information.

Furthermore, a desirable, new property can be provided here regarding a user profile for a limited period. Any users' data (including a hash of the IP address and a nonce) will be discarded on-the-fly (e.g. after a session of 3 minutes) and, hence, there is no stored user profile. Such feature can tell how many unique users within that period of time (3 minutes). As users' IP addresses are anonymised (and not stored), the proposed scheme protects users' physical locations. Therefore, the proposed scheme protects users' identifiers, state and physical locations, and is still highly likely to produce accurate number of unique users.

In the proposed scheme with its three scenarios, each variable can have a weight in order to estimate the probability whether the user is returning or new. For example, a close result may indicate the user has visited the webserver 10 minutes ago but currently is using a different browser. Such conclusion can help in "anonymously" confirming the user is using different browsers as 67% of sampled users use more than one browser [122]. Using such weights, we can also address the *privacy vs accuracy problem*, that is, we ideally want to count unique users while preserving their privacy. With the assumption of small number of corrupt users [11], the proposed scheme can achieve higher accuracy level of web metering results in a privacy-preserving manner.

The proposed web metering script at the audit agency only collects browser and computer data with no IP addresses or tracking cookies usage. Various fingerprint privacy issues were pointed out in [122] particularly when captured browser information achieved 62.01% precision in identifying users. However, in the rest of the tests, users' IP addresses were captured as well. The proposed web metering script can also anonymise captured data on-the-fly e.g. hashing captured users' browser plugins. The different stages of anonymisation, from tables of users' data to the hashes, provide privacy-enhancing results than persistent techniques like cookies.

Accuracy And Granularity of Metered Data.

The web metering result and evidence show the number of unique users with respect to a particular interval of users visits e.g. five minutes. The proposed scheme can further collect additional data about users' peripherals including keyboard presses and mouse movements. Such data can be used to check the probability of the user seeing an advertisement and to determine how long the user spent on a webpage. No other users' data (e.g. users' ages) than Table 8.1 are requested.

We explored options for accessible browser objects (e.g. *navigator* and *screen* [122, 95] that can be captured for web metering purposes. For example, navigator.appVersion is more accurate about detecting user's browser than navigator.appName and navigator.appCodeName. We found that other objects including *availHeight* can also be used in the web metering script.

The Unique User Check can tell the number of unique user for very limited periods e.g. 10 minutes. Aggregate number of unique users for larger periods cannot accurately be added up (hotel problem [109]). In order to solve that, there have to be two phases: active check and post-counting. During the active check (and before *flushing* the table), hashes of the users' tuples have to be stored for longer periods (say 2 hours). In post counting phase, while adding up number of unique users (say periods of 12), duplicate hashes will be combined. Smaller hashes can be used here to further "anonymise" the hashed results and not be used as identifiers which unfortunately will increase the likelihood of collisions. *Salting* techniques could also be used to further anonymise the stored hashed results with a random number being changed frequently (e.g. every day).

8.7.3 Usability

8.7.3.1 Computing Resources

The user needs computing resources to reply to probing requests. The concerned probing requests are Flash and browser's plugins. Audit agency and webserver have to be equipped with a large memory in order to primarily avoid storing any users' data and secondly efficiently run the web metering code. For example, Piwik efficient recommendation for memory is 1 GB, which is common webservers' requirements today. PHP cache can further be used to optimise the webserver (or audit agency) memory management. The active users' comparison in Unique User Check can be executed in the background. Using a webserver of with 1024 MB memory and 1.6GHz processor, it took on average 4 milliseconds to parse a simple table (similar to Table 8.3) of 100 users.

8.7.3.2 Storage

The user in the redirection-based scenario might store a cookie that includes a nonce. The use of hash functions decreases the storage requirements at audit agency and webserver.

8.7.3.3 Communication

The largest data the user needs to send are data about browser's plugins and screen. Using few tests, both sent data are in average 5000 to 6000 bytes on traditional computers and around 1500 bytes for browsers' data on mobile devices.

8.7.3.4 Proof-Of-Concept And Transparency

In this scheme, we investigated accuracy and privacy trade-off requirements. In addition, there was a dilemma between implementing the security technique (Corrupt Webserver Check) and preserving users' privacy. As for scheme usability, the proof-of-concept tests particularly regarding the uniqueness of obtained data show the potential of such scheme to provide comparable results. Details about the used code are provided in Appendix A.

We did various tests to check whether the different scenarios can be done transparently to the user and check the accuracy of different captured users' data. Such invisibility addresses usability issue. The following is a one line log for HTTP requests captured by a webserver.

```
81.1..172 - - [24/Jul/2013:09:55:27 -0400]
"GET / HTTP/1.1" 200 1897 "-" "Mozilla/5.0 (Macintosh;
Intel Mac OS X 10_6_8) AppleWebKit/534.59.8 (KHTML,
like Gecko) Version/5.1.9 Safari/534.59.8"
```

The following is an explanation for the recorded log. 81.1..172 is the user's source IP address. 24/Jul/2013:09:55:27 is the date and time of the visit. -0400 is used to represent Eastern Time Zone (ET), short from Universal Time Coordinated (UTC) - 04:00. Per [54], "GET / HTTP/1.1" 200 means an HTTP request was made for the home page and was successful (200 is a code for OK). The user's operating system is Mac, version number 10_6_8. The user's browser engine is AppleWebKit 534.59.8 and the browser is Safari.

Users' platforms have different privacy and security settings. For example, browser plugins, in devices with iOS 8 operating system, cannot be obtained using JavaScript. Similarly, if flash or Java is disabled, user's system fonts cannot be obtained. Also, if JavaScript is disabled, it will be announced to the webserver (which has implications as in anti-tracking observation). We used user's Time Zone to infer his location. For example, a Time Zone of -60 (coupled with *HTTP_ACCEPT Headers* of fr-fr value) can also be used to infer that the user is browsing from France. (However, we encountered a case that updating the operating system from 8.0.2 to 8.1.2 changed the Time Zone value from -60 to 0.)

Meta refresh tag can be used to tell the user's browser to refresh the webpage after specific number of seconds. By adding the below straightforward tag at HTML header with content value equals zero, a redirection to the audit agency will occur immediately.

```
<meta http-equiv="Refresh" content ="0;
url=http://AuditAgencyExample.com">
```

The same tag and method have to be applied at the audit agency to redirect the user back to the webserver, after successfully capturing user's data. Unlike GA or Piwik, redirection is used here instead of audit agency script and webserver-owned cookie.

Table 8.6: Limitations

Strength	Weaknesses/Challenges
• Privacy	• Accuracy
• Usability	• Security (to some extent)

Redirection can also happen between cooperating yet unlikely colluding webservers instead of the audit agency. However, the redirection should be done in away that is not "disorienting" to the user[17]. Audit agency can additionally collect more data about the user, possibly private data to prevent the webserver from faking users. A signed script can be used and approved in major browsers too. Audit agency might also periodically perform other actions not published to webservers e.g. audit agency could use a script from another audit agency.

Existing privacy-preserving browser toolbars can be reused for such web metering services. For example, ShareMeNot[18] is a browser plugin that blocks cookies from being sent to embedded content entities. Such tools (that provide privacy benefits to the user) can be redesigned to provide web metering results. For example, the blocked entities are in a good position to offer a web metering service in an enhanced privacy-preserving manner (enforced by the tool) to other webservers.

8.7.4 Summary

The proposed scheme runs transparently to the user in a privacy preserving manner. Security of the scheme depends on making it harder for corrupt webservers faking visits while not invading users' privacy in the process. Similarly, there is a dilemma for obtaining users' data while preserving their privacy. Table 8.6 provides a summary.

[17]www.w3.org/TR/WCAG10CORETECHS/#auto-page-refresh
[18]www.sharemenot.cs.washington.edu/

8.8 Conclusion

Third party Analytics tools are commonly used today by webservers with Google Analytics [60] as a leading example. In this chapter, we proposed a web metering scheme with different settings for different scenarios, where its novel underlying techniques can be used to securely improve the privacy and accuracy of existing schemes. In particular, the proposed scenarios and underlying techniques can be used as improvements to the privacy of existing schemes (like Google Analytics) while providing accurate number of unique users. To the best of our knowledge, the proposed scheme is the first solution to address counting number of unique users in a privacy-preserving manner, in different scenarios. We concentrated on a few published findings to address the privacy and accuracy dilemma. Navigation and anonymised computer data were the only user data transparently obtained, compared to personally identifiable information in previous schemes (e.g. Google Analytics). This user's unawareness feature can be further genuinely designed with advertising activities for better privacy-preserving browsing experience. We used a simple example of 10 minute periods; however, the interval has to be feasibly evaluated.

We addressed some raised smaller issues from different perspectives, and relevant desiderata. We explored the use of various kinds of user data to produce web metering results in a privacy-preserving way. We promote the use of fingerprints over actual data to get privacy-enhancing results and to avoid the privacy attack of potentially deriving user's preferences (or worse identification) from any navigational data. Users' browsers plugins data is a promising type of data that can be used to count unique users. However, an anonymised version of the data should still be used as evidence e.g. using hashing to counter any possible linkage of plugins changes. We believe such obtained data can still achieve comparable accuracy and our results of exploring potentially distinctive user's data can be a starting point for future work to further explore more distinctive yet privacy-preserving data. We showed how we can still address the desirable property of counting number of uniquer users during a specific limited period (e.g. 10 minutes) without the need for persistent data after that session. We analysed the security of the proposed scenarios, and privacy of obtained data and outlined the benefits. We believe a published script by a neutral entity (i.e. audit agency) is a reasonable assumption to improve its trustworthiness. Such open source perspective is common today as noticed in web metering services similar to Piwik.

Compared to proposed schemes in Chapters 6 and 7, the scheme here satisfies all aspects of usability as it does not require any hardware device at the user side or a prior relationship. With such flexibility, the scheme is also communication and storage efficient and with minimal computing resources to execute the proposed techniques.

Future work includes full-scale implementation or adoption by open source tools (e.g. Piwik) for testing against large number of users. Future work also includes further researching and specifying all trackable browser plugins so they are excluded in Data Obtaining Technique. Also, for particular devices with certain configurations (e.g. an outdated iPad), the Corrupt Webserver Check technique could request (or force) an optimised view of landscape and deliver the corresponding page. Such additional check is used to confirm the communicating device.

Chapter 9

A Third-Party-Centric Web Metering Scheme Using Embedded Content

Contents

This chapter provides a case study of one scenario of the conventional web metering scheme described in Chapter 8. The proposed web metering scenario, from the generic scheme, attempts to shift the role of the trusted party to collaborative online webservers. Such a new look at the elevation of trust by other collaborative entities can be used to provide "good enough" web metering results and evidence. The proposed embedded web metering scenario also introduces the use of lightweight web metering techniques transparently to the user in an enhanced privacy-preserving manner. Furthermore, an audit agency can be involved to *transparently check the metered users. This chapter starts by providing relevant background information. Then, it describes the proposed web metering scenario. Last, it analyses the proposed scenario from different perspectives.*

9.1 Introduction

Embedding objects for contents or services of other webservers is common nowadays. The embedded content has various shapes including but not limited to embedded videos, pictures and advertisements. A further developed concept is *Web Widgets*, which is estimated to be used by more than 20 % in major online content and services providers[1]. Further research addressed advertisements displayed based on the content, where such advertisements can be chosen properly by a third party and get displayed at the advertising webserver through some mechanism (e.g. advertisements can be associated on-the-fly to webpages using genetic programming [81].) This type of schemes provides a shift from having a self-developed advertising model into one where many parties are involved. Thus, fortunately such research can improve schemes based on collaborative actions, and provide better auctioning information for webservers in which web metering schemes can provide decisive information regarding that. For example, in [53] it is described how auctions can be carried out between the publisher and the advertiser for a required level of advertisement exposure. Also, auctioning deals, among cooperating webservers, for displaying embedded advertisements would even motivate corrupt webservers to act honestly and cooperate with others. Alongside those embedded videos, advertising and social networking functions, an additional web metering function can be feasibly added.

[1]royal.pingdom.com/2012/06/18/how-many-sites-have-facebook-integration-youd-be-surprised/

User's Data Obtaining technique is shown in Figure 9.1. In step 1, the originally visited webserver delivers the requested webpage to the user, including web metering scripts to other webservers. In steps 2 and 3, the loaded script executes the techniques described in section 8.6.4 and sends the data back to referenced webservers. The user's data can be sent as custom parameters in the URL, similar to Google Analytics [106].

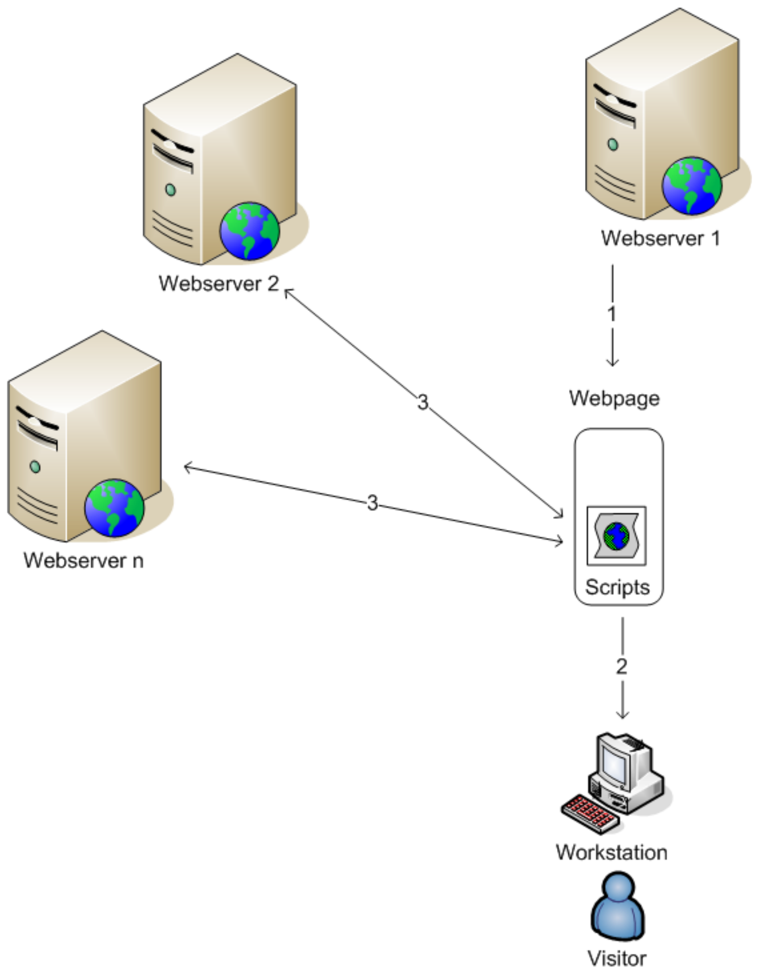


FIGURE 9.1: User's Data Obtaining in Embedded Scenario

The proposed scheme can also utilise a syndication protocol like Atom Publishing Protocol (APP) or its predecessor Really Simple Syndication (RSS). At the beginning of the research, we observed how web syndication activities were picking momentum and such evolving webservers' interconnectivity can be used for web metering purposes. Typically in APP, a request can be sent to a webserver to retrieve an object and the webserver sends back a reply for that request [72]. The protocol works on the atom feed entries and consider them as resources, as described in [71]. Also, the APP uses HTTP requests to deal with these resources and the HTTP response codes to keep track of sent requests.

The structure of APP can be simplified as follows. The embedded content or entries are grouped into Atom Feeds or Collections. Those collections can be further grouped into Workspaces. The set of Workspaces will be the Service Document, as depicted in Figure 9.2. Any of those components can be referenced by Uniform Resource Identifier (URI) or Internationalized Resource Identifier (IRI).

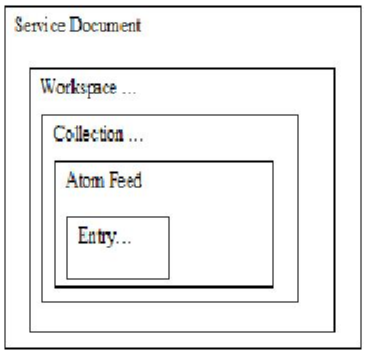


Figure 9.2: Service Document in Atom Publishing Protocol.

The following are four APP operations. The first two have to be executed to get the embedded content.

- Retrieving a Service Document

  The process of getting a Service Document can be done in two steps.

  1. An *HTTP GET* request is sent to the URI of the Service Document.
  2. The webserver responds with the Service Document and a status code.

- Listing Collection Members

  The process of listing members of a Collection can be done in two steps.

  1. An *HTTP GET* request is sent to the URI of the required Collection.
  2. The webserver responds with IRIs of members and a status code.

- Creating a Resource

  The process of creating a resource can be done in two steps.

  1. An *HTTP POST* request of the required member is sent to the URI of the Collection.
  2. The webserver responds with IRI of the created resource and a status code.

- Editing a Resource

  The process of retrieving a resource can be done in two steps.

  1. An *HTTP GET* request is sent to the URI of the resource.
  2. The webserver responds with the resource and a status code.

  The process of editing a resource can be done in two steps.

  1. An *HTTP PUT* request is sent to store a resource.
  2. The webserver responds with a status code.

  The process of deleting a resource can be done in two steps.

  1. A *Delete* request is sent to the URI of the resource.
  2. The webserver responds with a status code.

9.2 Proposed Web Metering Scheme

Further to assumptions, techniques and protocol steps described in section 8.6.4, the proposed web metering scheme tries to provide evidences from the collaborative judgement of the various serving entities in today's typical web infrastructure.

Elevation of trust of the visited webserver is done by the help of other involved webservers as they endorse the web metering results in a manner similar to *web of trust* approach [102]. The cross-scripting collaboration among the involved entities has to be allowed to execute at the user's platform (assumption 2). Furthermore, other forms of scripts can be executed by the involved webservers. For example, user browsing activity on a webserver can be tracked by reloading the whole requested item periodically throughout the interaction while the user is online [103].

As a result of this transparent integration, the webservers have recorded requests and user browsing data in their repository as evidences of visits to the original webserver. Such technique will give an indication of how long a webpage is viewed with no distinction whether visits are done by new or returning users. However, a short-lived tracking feature can be added to know number of unique users for a limited period of time. Having a trusted entity will provide higher degree of confidence for the results against false visits coordinated among colluded corrupt webservers. Additionally, even such requests for the embedded content can carry authentication headers. In such authentication, the forwarding webserver can be registered with the referenced webserver and possess a unique value. Also, the sent request can contain the user ID, which is an added security feature against corrupted webservers but will not make the scheme privacy-preserving.

9.3 Analysis

Further to the analysis and discussion in section 8.7, the following is a more detailed analysis for the proposed scenario against the web metering requirements.

– Integrity

∗ Reliability

Reliability is achieved here by the collaborative agreement on captured visits. Original webserver would publish results of visits with referenced webservers (or a metering provider). Those referenced entities agree to the results if they correspond to their records. Having one referenced audited entity will provide required procedures and security for the other webservers (including the original webserver). For example, whenever applicable and desirable, having a trusted entity embedded in all webpages displaying advertisements, the advertisements payment procedure will require crosschecking both the presented evidences from the original webserver with ones existing at the trusted webserver. Or indeed the advertiser itself can be involved with every page request by embedding extra requests to the advertiser. One application specific issue with this scheme is the correlation of the recorded distributed evidences.

The Corrupt Webserver Check technique will counter adversary's attempts to impersonate a valid user against the collaborative webservers. Also, an adversary cannot impersonate a webserver (as a fake involvement to get shares) because the rest of collaborative webservers will notice discrepancy between results. An adversary can attempt to launch a replay attack to increase the visit data. For example, the adversary can resend user presence data to referenced webserver after the user left the webserver and thus increase the amount of user presence. However, the Corrupt Webserver Check technique will flag repeated users' data and consequently fake visits will score lower value (by not increasing number of unique users), making the attack unsuccessful.

An adversary can launch man in the middle attack and change the obtained visit data. For example, the adversary changes original webserver URL to a fake webserver and increase number of visits for the fake webserver. Depending on the setup, the attack could be meaningless and only achieves denial of service for the intended webserver. If the adversary can get in the middle of the communication channel and change user's data, a successful man in the middle attack could increase number of unique users (but not overall users) among the judging webservers. Such attack has to successfully change user's data so that the calculated weight adds another unique user. However, the Corrupt Webserver Check technique should flag any anomalies regarding number of unique users and further probes more data about the concerned user (e.g. IP address).

    * Data Integrity

      There is no communication integrity for data sent in this scenario. For example, a reply message sent back to the original and referenced webservers carrying horizontal and vertical coordinates of the mouse pointer can be modified by an adversary to the previous value, to falsely record that the user is offline. However, evidential integrity can be achieved by cross referencing evidence among collaborative webservers.

  – Privacy

    Privacy is satisfied here because webservers only collect some prespecified computer and navigational data (e.g. user's mouse movement and keyboard strokes). IP addresses could initially be collected for correlation for visits among webservers and once correlation is completed, IP addresses can then be discarded.

  – Accuracy

    * The proposed scenario provides accurate number of visits based on the agreement of collaborative webservers and collected information about the visits. The use of counter will also solve the new versus returning users problem pointed out in [109]. The number of new users and returning users should be equal to total number of users. Any new user with a counter of more than one will lose the new user status and become returning.
    * Referenced webservers can collect additional data about the user's visit including keyboard presses and mouse movements and clicks through the embedded content code, which satisfies the granularity requirement.
  - Efficiency
    * Transparency

      This proposed scenario is transparent to the user because it captures his normal browsing behaviour.
    * Computing Resources

      From the user side, despite the referenced webserver only captures normal browsing actions, a tracking code, which captures users actions, has to run at the user's platform requiring CPU and memory resources. From the webserver side, after original webserver receives the user's request, it serves its normal webpages with embedded content. Then, once user receives embedded content, referenced webserver starts its tracking operations to capture user's actions.
    * Communication

      User needs to receive the additional tracking code from referenced webserver. Then, the user has to send results of tracking instructions to referenced webserver. Respectively, referenced webserver has to send tracking code and receive its results. Original webserver has to communicate with other referenced webservers for evidence correlation. Efficient procedures for evidences presentation will include methods like collecting sufficient evidences from representative parties. For example, to reduce the cost of correlation, one third party can be involved at a time.

Correlating logs of two webservers located in Canada and America respectively showed difference of around 4 seconds due to their distant locations and different platforms. The logs at the referenced webservers showed records of the originally visited webserver. Below is an example for a log generated at the referenced webserver for the webserver modprime.com.

```
"GET /id/ball.wmv HTTP/1.1" 200 2571623
http://modprime.com/meter/index.html" "Mozilla/5.0 (Windows; U;
Windows NT 6.0; en-US) AppleWebKit/525.19 (KHTML, like Gecko)
Chrome/1.0.154.53 Safari/525.19
```

∗ Storage

The user does not have to store any data; however, webservers store visits' information for later correlation.

Chapter 10

Summary And Conclusions

This chapter provides a global analysis of web metering schemes and a high level view of the problems. This chapter also summarises the contributions and future work.

This research addressed a specific problem posed in the literature almost a decade and a half ago. Earlier research continued on using secret sharing schemes to solve the problem with different assumptions. The internet blackswan [116] with its unexpected shape most likely was the first cause to render such schemes unusable. However, without that early work, recent schemes (e.g. Piwik) and our proposed schemes would not possibly have existed. Despite being an imponderable, that early work could also have provided the foundations for many web applications including online advertising.

At the beginning of the research, we had rough ideas about potential solutions. Our first task was to do an extensive search to find relevant work, both from a theoretical and a practical point of view. We then started by defining the framework and deriving necessary and desirable requirements. Specifying requirements can easily get subjective particularly in an area that survives on business. We believe many of the modest improvements in our proposed schemes compared to previous work are in the right direction. Those applied concepts can be further used in formal methods or full-scale practical deployments. We used Dolev-Yao model where the adversary controls the network. We then explored relevant mathematical problems, potential security solutions and different users' data. We believe those interdisciplinary areas still have the potential for further research for more promising web metering schemes.

The user non-cooperation (or simply disinterest) further enforces low cost solutions, that can provide the web metering needs without the user involvement or breach of his privacy rights. User's lack of awareness of existing schemes is a crucial step towards usable deployments. The start of the research focused on good enough approaches that can become real solutions compared to perfect ones (e.g. secret sharing schemes). Researching such heuristics to improve problem solving results included exploring various transparent web metering techniques that can provide a web metering solution. The privacy problem of many potentially good enough current transparent solutions (e.g. Google Analytics) was also another focus of study.

As shown and discussed, there was a wide spectrum of related work, but we believe this thesis provided a coherent account of a basic framework and problems. Spotting the real problems of old rigorous schemes that did not turn out to be solutions (and further influenced other researchers) was a big step in reasonably deriving requirements and desiderata. We took utmost care that this self-assigned process does get subjective by clearly stating user's lack of interest assumption and protecting his privacy rights. There were visionary cases (backed by commercial publications [95]) which we further investigated. We also used two other published assumptions namely the small number of corrupt users [11] and

TABLE 10.1: Summary Of Proposed Schemes And Achieved Properties

Third Party	Offline		Online		
Centric	User		User	Third Party	
Model	Signature	Voucher	Voucher	Script	In-line
Previous	Signature	Secret Sharing	N/A	GA	Proxy
Proposed	Hardware	Secret Hashes	IDMS	Embedded	Analyser
Properties	Privacy + Integrity	Integrity + Transparency	Integrity + Transparency	Privacy + Transparency	Privacy + Accuracy

the desirable low cost solutions [57]. We believe by looking at the big picture of a quick outline of sketches of different previous schemes and such assumptions, the proposed requirements and desiderata should convincingly be highlighted. We initially explored different web metering ideas and, then, we developed the ones that satisfy the requirements and desiderata into complete solutions. The main proposed schemes were developed in the hope to become (or influence) current and future deployment, satisfying some requirements and striking a balance among trade-off properties.

Proposing web metering schemes might not directly convince commercial players to change their existing ones. Consequently, understanding relevant regulations and policies is a vital part for the proposals to become practical solutions. Representative regulations and policies are provided to be used and enforced in web metering context. There also has to be an element of elegance for the solution to be acquired, and further influence evaluation (and consequently improvement). Reusing existing solutions for web metering purposes is one way to improve the effectiveness of the "tailored" web metering schemes.

In this thesis, a dedicated chapter for each novel web metering scheme (of the main three categories) mainly addresses trade-off properties, transparently to the user. The proposed authentication-based scheme (e.g. using IDMS in Chapter 7) addresses security and accuracy properties. The proposed hardware-based scheme in Chapter 6 addresses security and privacy properties. The proposed analytics-based scheme in Chapters 8 and 9 addresses privacy and accuracy properties. A summary of representative previous work, proposed schemes and achieved properties is provided in Table 10.1. As outlined, user centric schemes were the main focus by addressing their usability (through transparency) and preserving users' privacy.

We highlighted intuitive approaches (signature-based web metering schemes) and showed their merits and problems. We extended existing techniques (e.g. DAA [25], browser fingerprints [51]) and applied them for clearer web metering purposes. For example, a cryptography-based mechanism along a hardware device was used to satisfy the security of one proposed scheme, as in section 6.4.1. At the same time, a privacy-preserving mechanism is used to prove knowledge of secrets without revealing them, as in section 6.4.2. Although the proposed schemes do not track users, we promote the use of a limited hash chain for signature-based categories as in the proposed hardware-based scheme. We also did several proof-of-concept implementation tests for the proposed schemes using different setups. Also, throughout the experimentation, we focused on efficiency and transparency properties while evaluating the accuracy of the results. We have analysed the proposed schemes using the requirements and desiderata, and validated most security properties using Scyther.

Future Work. Future work includes going through each obtainable user's data from different new handheld devices. Such list can be studied to check potential uniqueness and privacy implication. Unique and changing user's data (e.g. certain browser fingerprints) can further be tested through an open source tool like Piwik. We believe adopting the relevant proposed techniques by such tool, as main functions, is the best testing environment to extensively evaluate the accuracy of results for large number of users. Similarly, standardising applicable privacy-preserving web metering extensions on popular browsers (e.g. Mozilla) could be future work. Also, future work could include surveying the diversity of available devices and their inherent distinctive users' requests. For hardware-based scheme, tests are planned to check efficiency of the new Windows-based operating systems. Complete formal validation of the proposed schemes using different models is also left for future work. One possible future work direction is to enable smart devices with privacy-preserving web metering function and to check its efficiency in unreliable network connections.

We have looked at the problem from open network perspective; future work also includes configuring the proposed schemes with different requirements and desiderata. The framework could be refined with special and particular web metering environments e.g. granted public Internet access, or monitored university campus, where user's privacy is not a concern. A lightweight version, yet non-transparent of Completely Automated Public Turing test to tell Computers and Humans Apart (CAPTCHA) work is also planned to be studied, for web metering purposes, particularly from efficiency point of view. We have not considered denial of service attacks and future work could include designing a secure scheme in which an adversary is motivated to shut down the scheme availability on a rival webserver.

Bibliography

[1] Fahad Alarifi. Transparent visitor centric web metering using identity management systems. In *The 6th International Conference on Information Security and Assurance (ISA 2012)*, pages 59–61, April 2012.

[2] Fahad Alarifi. Voucher web metering using identity management systems. *International Journal of Security and Its Applications*, 6(2):391–396, April 2012.

[3] Fahad Alarifi and Maribel Fernández. A new privacy-preserving web metering scheme using third-party-centric analytics. In *World Symposium on Computer Applications & Research (WSCAR'15)*, March 2015.

[4] Fahad Alarifi and Maribel Fernández. Towards privacy-preserving web metering via user-centric hardware. In *International Conference on Security and Privacy in Communication Networks*, volume 153 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering.* Springer International Publishing Switzerland, to appear, October 2015.

[5] Fahad Alarifi and Maribel Fernández. Towards privacy-preserving web metering via user-centric hardware. *EAI Endorsed Transactions on Security and Safety, to appear*, 15(4), 2015.

[6] Waleed Alrodhan. Privacy and practicality of identity management systems. Technical report, Department of Mathematics, Royal Holloway University of London, November 2010.

[7] American Association of Advertising Agencies, Association of National Advertisers, Council of Better Business Bureaus, Direct Marketing Association and Interactive Advertising Bureau. *Self Regulatory Principles for Online bBehavioural Advertising*, July 2009.

[8] Asia-Pacific Economic Cooperation. Singapore: APEC Secretariat. *APEC Privacy Framework*, 2005.

[9] Charles Asmuth and John Bloom. A modular approach to key safeguarding. *IEEE Transactions on Information Theory*, 29(2):208–210, March 1983.

[10] Richard Atterer, Monika Wnuk, and Albrecht Schmidt. Knowing the user's every move: user activity tracking for website usability evaluation and implicit interaction. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, pages 203–212, New York, NY, USA, 2006. ACM.

[11] S. G. Barwick, Wen-Ai Jackson, and Keith M. Martin. A general approach to robust web metering. *Des. Codes Cryptography*, 36(1):5–27, 2005.

[12] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Keying hash functions for message authentication. In *Proceedings of the 16th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '96, pages 1–15, London, UK, UK, 1996. Springer-Verlag.

[13] Francesco Bergadano and P. De Mauro. Third party certification of http service access statistics (position paper). In *Security Protocols Workshop*, pages 95–99, 1998.

[14] Jean-Paul Berrut and Lloyd Trefethen. Barycentric lagrange interpolation. *SIAM Review*, 46(3):pp. 501–517, 2004.

[15] Robert Beverly. A Robust Classifier for Passive TCP/IP Fingerprinting. In *Proceedings of the 5th Passive and Active Measurement (PAM) Workshop*, pages 158–167, April 2004.

[16] Carlo Blundo, Annalisa De Bonis, and Barbara Masucci. Metering schemes with pricing. In *DISC*, pages 194–208, 2000.

[17] Carlo Blundo, Annalisa De Bonis, and Barbara Masucci. Bounds and constructions for metering schemes. In *Communications in Information and Systems 2002*, pages 1–28, 2002.

[18] Carlo Blundo, Annalisa De Bonis, Barbara Masucci, and Douglas R. Stinson. Dynamic multi-threshold metering schemes. In *Selected Areas in Cryptography*, pages 130–143, 2000.

[19] Carlo Blundo and Stelvio Cimato. Sawm: a tool for secure and authenticated web metering. In *Proceedings of the 14th international conference on Software engineering and knowledge engineering*, SEKE '02, pages 641–648, New York, NY, USA, 2002. ACM.

[20] Carlo Blundo, Stelvio Cimato, and Barbara Masucci. A note on optimal metering schemes. *Inf. Process. Lett.*, 84(6):319–326, 2002.

[21] Carlo Blundo, Sebastià Martín, Barbara Masucci, and Carles Padró. A linear algebraic approach to metering schemes. *Des. Codes Cryptography*, 33(3):241–260, November 2004.

[22] Annalisa De Bonis and Barbara Masucci. An information theoretical approach to metering schemes. In *2000 IEEE International Symposium on Information Theory - ISIT 2000*, 2000.

[23] Marc Bourreau, Christian Gacon, and Massimo Columbo. The economics of internet flat rates. *Communications & Strategies*, pages 131–152, 2001.

[24] Colin Boyd and Dong-Gook Park. Public key protocols for wireless communications. In *Proceedings of The 1st International Conference on Information Security and Cryptology, ICSCI '98, Seoul, Korea*, pages 47–57, December 1998.

[25] Ernie Brickell, Jan Camenisch, and Liqun Chen. Direct anonymous attestation. In *Proceedings of the 11th ACM conference on Computer and communications security*, CCS '04, pages 132–145, New York, NY, USA, 2004. ACM.

[26] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the Seventh International Conference on World Wide Web 7*, WWW7, pages 107–117, Amsterdam, The Netherlands, The Netherlands, 1998. Elsevier Science Publishers B. V.

[27] Simon Byers, Aviel D. Rubin, and David Kormann. Defending against an internet-based attack on the physical world. *ACM Trans. Interet Technol.*, 4(3):239–254, 2004.

[28] Calcutt, David. Committee on Privacy and Related Matters. *Report of the Committee on Privacy and Related Matters.* Cm (Series) (Great Britain. Parliament). H.M. Stationery Office, 1990.

[29] Jan Camenisch and Anna Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In *Proceedings of the 22nd Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '02, pages 61–76, London, UK, UK, 2002. Springer-Verlag.

[30] Center for Democracy and Technology. *Privacy Basics: Generic Principles of Fair Information Practises.*

[31] D. Chaum, A. Fiat, and M. Naor. Untraceable electronic cash. In *Proceedings on Advances in Cryptology*, CRYPTO '88, pages 319–327, New York, NY, USA, 1990. Springer-Verlag New York, Inc.

[32] L. Chen and W. Mao. An auditable metering scheme for web advertisement applications. In *ISC*, volume 2200 of *Lecture Notes in Computer Science*, pages 475–485. Springer, 2001.

[33] YuQun Chen, Mohammed Moinuddin, and Yacov Yacobi. Mobile wallet and digital payment, September 30 2011. US Patent App. 13/249,381.

[34] Benny Chor, Amos Fiat, Moni Naor, and Benny Pinkas. Tracing traitors. *IEEE Transactions on Information Theory*, 46(3):893–910, 2000.

[35] Robert Cooley, Bamshad Mobasher, and Jaideep Srivastava. Data preparation for mining world wide web browsing patterns. *Knowl. Inf. Syst.*, 1(1):5–32, 1999.

[36] Véronique Cortier, Stéphanie Delaune, and Pascal Lafourcade. A survey of algebraic properties used in cryptographic protocols. *J. Comput. Secur.*, 14(1):1–43, January 2006.

[37] Lorrie Cranor, Brooks Dobbs, Serge Egelman, Giles Hogben, Jack Humphrey, Marc Langheinrich, Massimo Marchiori, Martin Presler-Marshall, Joseph Reagle, Matthias Schunter, David A. Stampley, and Rigo Wenning. The platform for privacy preferences. W3C Recommendation, November 2006.

[38] Lorrie Cranor, Brooks Dobbs, Serge Egelman, Giles Hogben, Jack Humphrey, Marc Langheinrich, Massimo Marchiori, Martin Presler-Marshall, Joseph Reagle, Matthias Schunter, David A. Stampley, and Rigo Wenning. The platform for privacy preferences. W3C Recommendation, November 2006.

[39] Lorrie Faith Cranor. Internet privacy. *Commun. ACM*, 42(2):28–38, February 1999.

[40] Cas Cremers. *Scyther User Manual*, February 2014.

[41] Alain de Botton. *Status Anxiety.* Penguin; Re-issue edition, 2005.

[42] Alex Dent and Chris Mitchell. *User's Guide To Cryptography And Standards (Artech House Computer Security).* Artech House, Inc., Norwood, MA, USA, 2004.

[43] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[44] Xuhua Ding. A hybrid method to detect deflation fraud in cost-per-action online advertising. In *Applied Cryptography and Network Security, 8th International Conference, ACNS 2010, Beijing, China, June 22-25, 2010. Proceedings*, pages 545–562, 2010.

[45] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. In *Proceedings of the 13th Conference on USENIX Security Symposium - Volume 13*, SSYM'04, pages 21–21, Berkeley, CA, USA, 2004. USENIX Association.

[46] Danny Dolev and Andrew C. Yao. On the security of public key protocols. Technical report, Stanford, CA, USA, 1981.

[47] Joseph Reagle Donald Eastlake and David Solo. *XML-Signature Syntax and Processing.* World Wide Web Consortium (W3C), June 2008.

[48] Paul Dourish and Ken Anderson. Collective information practice: Exploring privacy and security as social and cultural phenomena. *Human Computer Interaction*, 21(3):319–342, 2006.

[49] Heiko Drewes, Richard Atterer, and Albrecht Schmidt. Detailed monitoring of user's gaze and interaction to improve future e-learning. In *HCI (6)*, pages 802–811, 2007.

[50] Cynthia Dwork and Moni Naor. Pricing via processing or combatting junk mail. In *CRYPTO '92: Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology*, pages 139–147, London, UK, 1993. Springer-Verlag.

[51] Peter Eckersley. How unique is your web browser? In *Proceedings of the 10th International Conference on Privacy Enhancing Technologies*, PETS'10, pages 1–18, Berlin, Heidelberg, 2010. Springer-Verlag.

[52] Federal Trade Commission. *Fair Information Practices.*

[53] Jon Feldman and S. Muthukrishnan. Algorithmic methods for sponsored search advertising. *CoRR*, abs/0805.1759, 2008.

[54] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext transfer protocol – http/1.1. RFC 2068, June 1999.

[55] Mark Fioravanti. Client fingerprinting via analysis of browser scripting environment. In *SANS Information Security Reading Room*, 2010.

[56] OpenID Foundation. Openid authentication 2.0., December 2007.

[57] Matthew Franklin and Dahlia Malkhi. Auditable metering with lightweight security. *Journal of Computer Security*, 6(4):237–256, 1998.

[58] Philip Ginzboorg. Seven comments on charging and billing. *Commun. ACM*, 43(11):89–92, 2000.

[59] Kendall Goodrich. What's up? exploring upper and lower visual field advertising effects. *Journal of Advertising Research*, 50(1):91–106, 2010.

[60] Google. Google analytics blog. Official weblog offering news, tips and resources related to google's web traffic analytics service. analytics.blogspot.com.

[61] Jonathan Grudin. Desituating action: Digital representation of context. *Human Computer Interaction*, 16(2-4):269–286, 2001.

[62] Lein Harn and Hung-Yu Lin. A non-repudiation metering scheme. *Communications Letters, IEEE*, 37(5):486–487, 2001.

[63] Michael Hutter and Ronald Toegl. A trusted platform module for near field communication. In *Proceedings of the 2010 Fifth International Conference on Systems and Networks Communications*, ICSNC '10, pages 136–141, Washington, DC, USA, 2010. IEEE Computer Society.

[64] Kantara Initiative. Kantara initiative, user managed access (uma). http://kantarainitiative.org/confluence/ display/uma/Home.

[65] International Organization for Standardization. ISO 11889-1:2009. *Information technology – Trusted Platform Module – Part 1: Overview*, May 2009.

[66] International Organization for Standardization. ISO 7498-2:1989. *Information processing systems – Open Systems Interconnection – Basic Reference Model – Part 2: Security Architecture*, 1989.

[67] International Organization for Standardization. ISO 9241-11:199. *Ergonomic requirements for office work with visual display terminals (VDTs) – Part 11: Guidance on usability*, 1998.

[68] International Organization for Standardization. ISO/IEC 10118-3:2004. *Information technology – Security techniques – Hash-functions – Part 3: Dedicated hash-functions*, 2004.

[69] International Telecommunication Union Telecommunication Standardization Sector (ITU-T). *Y.2720, NGN Identity management framework*, January 2009.

[70] Internet Engineering Task Force, RFC 2617. *HTTP Authentication: Basic and Digest Access Authentication*, June 1999.

[71] Internet Engineering Task Force RFC 4287. *The Atom Syndication Format*, December 2005.

[72] Internet Engineering Task Force RFC 5023. *The Atom Publishing Protocol*, October 2007.

[73] Internet Engineering Task Force, RFC 6265. *HTTP State Management Mechanism*, April 2011.

[74] Markus Jakobsson, Philip D. MacKenzie, and Julien P. Stern. Secure and lightweight advertising on the web. *Comput. Netw.*, 31(11-16):1101–1109, 1999.

[75] Rob Johnson and Jessica Staddon. Deflation;secure web metering. *Int. J. Inf. Comput. Secur.*, 1(1/2):39–63, 2007.

[76] Stephen H. Kan. *Metrics and Models in Software Quality Engineering.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2nd edition, September 2002.

[77] Tadayoshi Kohno, Andre Broido, and K. C. Claffy. Remote physical device fingerprinting. In *SP '05: Proceedings of the 2005 IEEE Symposium on Security and Privacy*, pages 211–225, Washington, DC, USA, 2005. IEEE Computer Society.

[78] Balachander Krishnamurthy and Craig Wills. Privacy diffusion on the web: a longitudinal perspective. In *Proceedings of the 18th international conference on World wide web*, WWW '09, pages 541–550, New York, NY, USA, 2009. ACM.

[79] Balachander Krishnamurthy and Craig E. Wills. Privacy diffusion on the web: a longitudinal perspective. In *WWW*, pages 541–550, 2009.

[80] Rajendra Kumar. *Human Computer Interaction.* Laxmi Publications, 2005.

[81] Anísio Lacerda, Marco Cristo, Marcos André Gonçalves, Weiguo Fan, Nivio Ziviani, and Berthier A. Ribeiro-Neto. Learning to advertise. In *SIGIR*, pages 549–556, 2006.

[82] Chi-Sung Laih, Chun-Ju Fu, and Wen-Chung Kuo. Design a secure and practical metering scheme. In *International Conference on Internet Computing*, pages 443–447, 2006.

[83] Leslie Lamport. Password authentication with insecure communication. *Commun. ACM*, 24:770–772, November 1981.

[84] Wenbo Mao. A structured operational modelling of the dolev-yao threat model. In *Security Protocols Workshop'02*, pages 34–46, 2002.

[85] Barbara Masucci and Douglas R. Stinson. Metering schemes for general access structures. In *ESORICS*, pages 72–87, 2000.

[86] Jonathan R. Mayer and John C. Mitchell. Third-party web tracking: Policy and technology. In *Proceedings of the 2012 IEEE Symposium on Security and Privacy*, SP '12, pages 413–427, Washington, DC, USA, 2012. IEEE Computer Society.

[87] Ahmed Metwally, Divyakant Agrawal, and Amr El Abbadi. Detectives: detecting coalition hit inflation attacks in advertising networks streams. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 241–250, New York, NY, USA, 2007. ACM.

[88] Peter L. Montgomery. Modular multiplication without trial division. *Mathematics of Computation*, 44(170):519–521, 1985.

[89] Greg Mori and Jitendra Malik. Recognizing objects in adversarial clutter: Breaking a visual captcha. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 1:134, 2003.

[90] Keaton Mowery, Dillon Bogenreif, Scott Yilek, and Hovav Shacham. Fingerprinting information in JavaScript implementations. In Helen Wang, editor, *Proceedings of W2SP 2011*. IEEE Computer Society, May 2011.

[91] James Muir and Paul Van Oorschot. Internet geolocation: Evasion and counterevasion. *ACM Comput. Surv.*, 42(1):4:1–4:23, December 2009.

[92] San Murugesan. Harnessing green it: Principles and practices. *IT Professional*, 10(1):24–33, 2008.

[93] Moni Naor and Benny Pinkas. Secure and efficient metering. In *EUROCRYPT*, pages 576–590, 1998.

[94] T. Narten and R. Draves. Privacy Extensions for Stateless Address Autoconfiguration in IPv6. RFC 3041, January 2001.

[95] Nick Nikiforakis, Alexandros Kapravelos, Wouter Joosen, Christopher Kruegel, Frank Piessens, and Giovanni Vigna. Cookieless monster: Exploring the ecosystem of web-based device fingerprinting. In *IEEE Symposium on Security and Privacy*, pages 541–555, 2013.

[96] OASIS Standard Specification. *Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0*, March 2005.

[97] Organisation for Economic Co-operation and Development. *OECD Guidelines on the Protection of Privacy and Transborder Flows of Personal Data*, September 1980.

[98] Venkata N. Padmanabhan and Lakshminarayanan Subramanian. An investigation of geographic mapping techniques for internet hosts. In *SIGCOMM*, pages 173–185, 2001.

[99] K. Pearson. The Problem of the Random Walk. *Nature*, 72(1865):294, 1905.

[100] Torben Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *Proceedings of the 11th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '91, pages 129–140, London, UK, UK, 1992. Springer-Verlag.

[101] Andreas Pfitzmann and Marit Köhntopp. Anonymity, unobservability, and pseudonymity - a proposal for terminology. In *Workshop on Design Issues in Anonymity and Unobservability*, pages 1–9, 2000.

[102] Phil's Pretty Good Software. *PGP(tm) User's Guide. Volume I: Essential Topics*, October 1994.

[103] James Pitkow. In search of reliable usage data on the www. In *Computer Networks and ISDN Systems*, pages 451–463, 1997.

[104] G. Polya and J.H. Conway. *How to Solve It: A New Aspect of Mathematical Method*. Penguin mathematics. Princeton University Press, 2004.

[105] Josyula R. Rao and Pankaj Rohatgi. Can pseudonymity really guarantee privacy? In *Proceedings of the 9th USENIX Security Symposium*, pages 85–96. USENIX, August 2000.

[106] Franziska Roesner, Tadayoshi Kohno, and David Wetherall. Detecting and defending against third-party tracking on the web. In *Presented as part of the 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12)*, pages 155–168, San Jose, CA, 2012. USENIX.

[107] Guha Saikat, Cheng Bin, and Francis Paul. Privad: Practical privacy in online advertising. In *Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation*, NSDI'11, pages 169–182, Berkeley, CA, USA, 2011. USENIX Association.

[108] Claus-Peter Schnorr. Efficient identification and signatures for smart cards. In *CRYPTO*, pages 239–252, 1989.

[109] Nicolae Sfetcu. *Web Design & Development:*. Lulu.com, 2014.

[110] Cyrus Shahabi and Farnoush Banaei Kashani. A framework for efficient and anonymous web usage mining based on client-side tracking. In *WEBKDD '01: Revised Papers from the Third International Workshop on Mining Web Log Data Across All Customers Touch Points*, pages 113–144, London, UK, 2002. Springer-Verlag.

[111] Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.

[112] Radu Sion and Mikhail Atallah. Secure access accounting for content distribution networks, 2005.

[113] Javier Herranz Sotoca. *Some Digital Signature Schemes with Collective Signers.* Dissertation, Universitat Politcnica de Catalunya, 2005.

[114] P. Srisuresh and K. Egevang. IP Network Address Translator (NAT) Terminology and Considerations, August 1999.

[115] Peter Swire and Sol Bermann. *Information Privacy Official Reference For The Certified Information Privacy Professional (Cipp).* IAPP Publication, 2007.

[116] N.N. Taleb. *The Black Swan: Second Edition: The Impact of the Highly Improbable Fragility".* Incerto. Random House Publishing Group, 2010.

[117] Anne van Kesteren. Cross-origin resource sharing. W3C Candidate Recommendation, January 2013.

[118] Luis von Ahn, Manuel Blum, and John Langford. Telling humans and computers apart automatically. *Commun. ACM*, 47(2):56–60, 2004.

[119] Thomas Wason. Liberty id-ff architecture overview- version: 1.2, November 2003. Liberty Alliance Project.

[120] Brent Waters, Ari Juels, J. Alex Halderman, and Edward W. Felten. New client puzzle outsourcing techniques for dos resistance. In *CCS '04: Proceedings of the 11th ACM conference on Computer and communications security*, pages 246–256, New York, NY, USA, 2004. ACM.

[121] Windows Certification Program. *Hardware Certification Taxonomy & Requirements*, April 2014.

[122] Ting-Fang Yen, Yinglian Xie, Fang Yu, Roger Peng Yu, and Martín Abadi. Host fingerprinting and tracking on the web: Privacy and security implications. In *NDSS*, 2012.

Appendix A

Some Web Metering Testing Codes

```
<script type="text/javascript">
var keyboard_press_flag=0;
var mouse_clicked_flag=0;
var mouse_moved_flag=0;
var mouse_movement_x=0;
var mouse_movement_y=0;
var i=1;

// The following function for the keyboard presses

function UpdateKeys(){
keyboard_press_flag=1;
}
```

```
// The following function for the mouse clicks

function UpdateMouseClick(){
mouse_clicked_flag=1;
}

// The following function for the mouse movement

function UpdateMouseMove(){
if (mouse_movement_x != event.clientX || mouse_movement_y != event.clientY){
mouse_movement_x = event.clientX;
mouse_movement_y = event.clientY;
mouse_moved_flag=1;
}
}

// The following function to check if of the three events occurred

function EventCheck(){
if(keyboard_press_flag >0 ||  mouse_clicked_flag >0 || mouse_moved_flag > 0){
keyboard_press_flag=0;
mouse_moved_flag=0;
mouse_clicked_flag=0;
i=i+1;

/*
If a user event occurred, a dummy request to webserver is automatically made
where it is recorded. Otherwise values can be sent using a server side script
or AJAX.
 */


if (i==1){
ChangeImage('images/dummy1.jpg')
}
else{
if (i==2){
ChangeImage('images/dummy2.jpg')
}
                 else{
ChangeImage('images/dummy3.jpg')
}
     }


if (i >2){
i=1;
}
}
}


var c=0;
var t;
function timedCount()
{
```

```
EventCheck(c)

c=c+1;

t=setTimeout("timedCount()",5000);

}


function ChangeImage(img_src) {

document.getElementById("img1").src = img_src;

}


</script>
```

For the proposed hardware-based web metering scheme, we did various tests to execute the required computations on a traditional computer. We used native C with MS visual studio 2010 and GNU MP Library to deal wig Big Number operations. We had many iterations to optimise the code for faster execution. The first statement of computing the value U can be simplified as follows.

```
LONGLONG GetCpuSpeed()

{

LARGE_INTEGER liFrequency;

QueryPerformanceFrequency ( &liFrequency );
printf ( "\n\nHigh resolution performance counter Frequency:
         %I64d (cycles / second)\n\t \n",  liFrequency.QuadPart );

return ( liFrequency.QuadPart );

}

LONGLONG cpuClock = GetCpuSpeed();
```

/* RSA Tool 2 [1] can also be used to get the below p and q primes, and then library is imported and headers files are included. */

```
s = ClockCycles();
BN_mul(n,  _p, _q, ctx);
BN_mod_exp(zf, z, f, n, ctx);
/* A faster modular exponentiation is used compared to BN_exp(). */
BN_mod_exp(xv1, x, v1, n, ctx);
BN_mod_mul(U, zf, xv1, n, ctx);
s = (ClockCycles() - s) / ( cpuClock / 1000LU) ;
```

Computing the second statement of the value N can be simplified as follows.

```
s = ClockCycles();

BN_mod_exp(N, I, f, p2, ctx);
```

[1] www.woodmann.com/collaborative/tools/index.php/RSA-Tool_2

```
s = (ClockCycles() - s) / ( cpuClock / 1000LU) ;
```

There are floating number exponentiation and div operations required to compute the equation (A^e). Therefore, the equation can be computed at a third party to shift any heavy computations from the user side to a more powerful computing device, possibly utilising Montgomery reduction. The computation of the third statement is simplified as follows.

```
s = ClockCycles();

mpf_div(One0e, One, e); // (1/e)

mpf_pow_ui(xv2, x, v2 ); //x^v2

mpf_mul(Uxv2, xv2, U); // (U*x^v2)

mpf_div(zUxv2, z, Uxv2); // (z/ (U*x^v2) )

double zUxv2One0e = pow(mpf_get_d(zUxv2), mpf_get_d(One0e) );

double A = GetMod(zUxv2One0e, m);

Result2 = pow(A, mpf_get_d(e));

Result3 = pow(mpf_get_d(z), f);
```

```
Result4 = pow(mpf_get_d(x), (v1+ (double)v2));


s = (ClockCycles() - s) / ( cpuClock / 1000LU) ;
```

Appendix B

Some Threat Trees

Exceptions can break the web metering operation. In case a problem with the scheme core functions was not handled properly, the problem can affect other web metering properties like the integrity of the scheme. Also under some situations, a conditional failure of some operation may arise which can break the scheme operation. Poor implementation of scheme functions can also technically break the overall scheme operation as shown in Figure B.1. A conditional failure is less likely to occur because determining the exact circumstances of a conditional failure is difficult to specify. Also repetitive disconnections can result in a denial of service attach against the scheme operation as shown in Figure B.2.

We start below with examples of potential problems at the visitor side.

Accidental Problems

- Resources limitation problem at visitor can reduce accuracy of interactions measurement (integrity).
- Visitor is blocking connections (implementation problem).

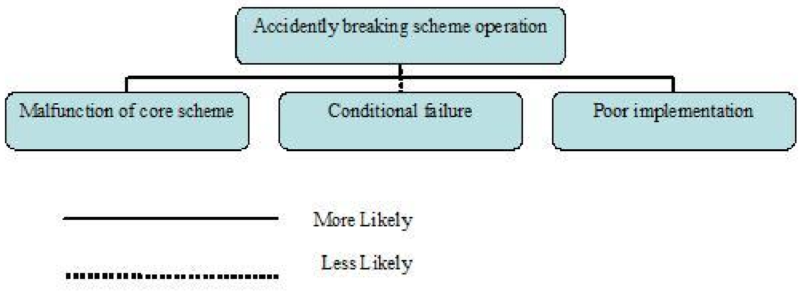

Accidently breaking scheme operation
Malfunction of core scheme
Conditional failure
Poor implementation
More Likely
Less Likely

FIGURE B.1: Threat Tree for Accidentally Breaking Scheme

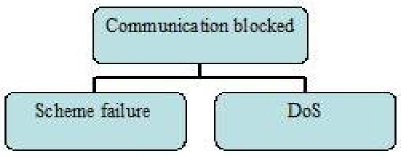


FIGURE B.2: Threat Tree for Communication Blocked

- Visitor to webserver interaction can not be done as a result of the visitor different environment. Such environment differences introduce platform dependency problem.

Intentional Problems

- Visitor does not participate in the metering process by using computing or storage resources. As a result, the scheme is allowed (to some extent) to access such resources.
- The visitor side does not provide required information e.g. allowing accurate presence status to be extracted.
- The visitor side dismounts any component that is required for web metering e.g. uninstalling a hosted web metering program.

- The visitor does not follow predictable browsing behaviour. Visitor browsing behaviour is the progressive way the visitor interacts with the Webserver through a technology interface. An example of such interface is computer mouse clicks.
- The visitor does not initiate the visit to the webserver or intend to access the webserver services excluding unintended ones (e.g. visits by mistake) or forced visits (e.g. redirection).

Examples of potential problems at the webserver side are as follows.

Accidental Problems

Integrity.

- The webserver is not involved properly in the metering process through incomplete participation of metering work, or as a result of a communication problem which triggers faulty connections to other entities.
- The webserver is not behaving according to assumed typical operation due to outside factors e.g. Operating System (OS) errors. Also the webserver may lack the required resources which introduces a different security requirement.

Privacy.

- The webserver is accidentally leaking Web Metering evidences or relevant captured information.