# Enabling Robust Information Accountability in E-healthcare Systems*

Daisuke Mashima
*Georgia Tech Information Security Center*
*Georgia Institute of Technology*

Mustaque Ahamad
*Georgia Tech Information Security Center*
*Georgia Institute of Technology*

## Abstract

In the United States, the transition from traditional paper-based healthcare records to electronic healthcare record (EHR) systems is being promoted aggressively. While EHR systems offer a number of benefits, they will introduce new security and privacy concerns. A significant fraction of such threats, at least in part, arise due to actions of insiders of healthcare organizations, either accidentally or intentionally. We believe information accountability, which allows us to securely identify how a health record reached a certain consumer and who was involved in its sharing or transfer, is a key to discourage such threats. In this work, we propose a patient-centric scheme to establish robust information accountability in electronic healthcare record sharing systems. We also present a prototype implementation and show that it does not impose unacceptable performance overhead.

## 1 Introduction

According to U.S. Department of Health & Human Services (HHS) [3], a large number of information breaches reported in healthcare settings were accidental and often resulted from loss or theft of computers belonging to healthcare organizations, for example [6] and [7]. Moreover, inadvertent disclosure could be a result of P2P file sharing by employees [10]. Other insider threats include intentional breach of healthcare data when dishonest employees motivated by monetary gain leak the data to external parties. One such example is the case of Cleveland Clinic in 2006 [1]. An insider in the clinic obtained patients' healthcare records and sold them to her cousin, who ran a medical claims company. He then filed a number of fake Medicare claims for monetary gain. As a result of an incident like this, patients whose records are misused could suffer financial loss. Also, such incidents would erode patients' and medical professionals' confidence in E-healthcare systems, which could limit the effectiveness of electronic health records.

Accidental or intentional data breach due to insiders can be discouraged by robust accountability of actions that access, share or transfer data in E-healthcare systems. In this work, we define information accountability (or simply accountability) as providing patients with assurance about how their healthcare records get to consumers who utilize health information to provide care or to support operations such as billing. Accountability, when misuse is detected, enables patients and healthcare organizations to identify and punish malicious insiders who may be engaged in illegal sharing (or leakage) of electronic health records. For example, in the Cleveland Clinic example mentioned earlier, accountability would make the involvement of the malicious insider and her cousin visible to patients, who can alert Medicare investigators and provide the evidence necessary to punish the dishonest employee. By ensuring actionable accountability, disallowed behaviors are discouraged under rules and punishments against healthcare professionals defined in HIPAA [10], as discussed in [17]. There are already a variety of schemes designed to detect and prevent insider attacks, but we believe that information accountability can effectively complement such schemes both in deterring attacks and for providing information that can facilitate investigation once incidents are reported. In addition to the mitigation of insider threats, patients can further benefit from information accountability since they can become aware of how their healthcare records are shared and who may have the copies of them.

Although current health systems utilize auditing, they do not necessarily provide sufficient level of accountability and do not support patient's awareness of how and when patient data is shared and used. Nowadays many healthcare organizations implement logging and access control mechanisms for a healthcare record repository,

*This research was supported in part by the National Science Foundation under grant IIS-NetSE-0905493.

but often such schemes are not comprehensive and could allow unauthorized actions [12]. Also, identifying compromised or malicious insiders may not be possible even when such security mechanisms are properly configured and enforced. For instance, in case a number of employees in a healthcare organization accessed the same record and one of them leaked or illegally shared a copy of the data with an external, and possibly malicious, entity that eventually misuses the health data, it is not possible to correctly identify which employee is responsible for it. Similar situations can also arise in case of accidental disclosure. Moreover, if copies of health records are shared with other healthcare organizations, which is common in E-healthcare systems, exactly identifying the source of a breach would be even more complicated.

In this paper, we explore a way to establish robust information accountability in healthcare record sharing. Specifically, we design a scheme to securely attach metadata called an "accountability tag" to each copy of an electronic healthcare record. Such tags carry cryptographically-verifiable evidence of entities that are involved in the sharing of health information that carries the tags. Under our system, accountability tags are verified and logged by a patient-controlled online agent that allows a patient to determine how her healthcare data is shared and consumed. Although we focus only on accountability from a patient's point of view in this work, a similar approach can also be used by a healthcare organization to track how information owned by it is shared.

This paper is organized as follows. In Section 2, we discuss related work. After that, in section 3, we discuss key assumptions and the scope of this work. The design of the system using accountability tags and the associated protocols are presented in section 4. Then, section 5 evaluates the correctness, and how various security threats are handled is discussed in section 6. A prototype implementation and its performance are presented in section 7. Finally, section 8 concludes the paper.

## 2 Related Work

The use of information accountability as deterrence against misuse of electronic data and its effectiveness are discussed by Weitzner et al. [17]. According to them, "Information accountability means the use of information should be transparent so it is possible to determine whether a particular use is appropriate under a given set of rules and that the system enables individuals and institutions to be held accountable for misuse." Our definition is closely related to this, but focused on sharing of electronic healthcare information. Namely, our system allows patients to be aware of how their healthcare records propagate in the distributed E-healthcare environment so they can determine whether sharing is appropriate and can identify which entities are involved in the health record sharing or disclosure when misuse is suspected.

Provenance of electronic data [15] [8] has similarity with information accountability pursued in this work. Provenance of data can be defined as "the process that led to the data" [11], and in addition to the origin of the data and chain of ownership, it covers what operations were performed on the data. Such information is derived based on a set of assertions made by services or processes that touch the data. Typically, provenance systems require a centralized repository, which is called a provenance store, that stores assertions to later answer queries from users or third-party auditors. However, in a distributed setting involving multiple organizations, secure aggregation of assertions by an external entity in the presence of certain threats is often not possible and also could lead to privacy issues in the healthcare setting, as discussed in [11]. In a multi-domain electronic healthcare record sharing environment, our scheme accomplishes information accountability, which can answer a set of provenance like queries about EHR sharing. Moreover, since collected information is stored on an entity chosen (or managed) by each patient, privacy concerns are minimal.

The idea of attaching an accountability tag to data may sound similar to digital steganography schemes often used in digital rights management systems. Use of digital watermarking to preserve ownership of healthcare records is explored, for example in [9]. As done in [9], typically a watermark is embedded by a data owner and is immutable during its lifetime. However, only embedding the identity of a content owner (or issuer) is not sufficient for accountability. Moreover, we need to attach information about who shares (or discloses) a certain health record with whom as information is shared across many entities. Thus, accountability information needs to be attached dynamically as sharing occurs. Because of these reasons, digital steganography is not a suitable solution for our context.

In the earlier work [13], we designed a scheme to ensure patients' awareness of when and by whom their healthcare records are used or updated. Although it helps patients notice suspicious events, it alone is not sufficient to establish information accountability pursued in this work. Namely, even if patients could detect misuse cases, they (and also other entities) can not exactly identify who was involved in the healthcare record sharing that resulted in the misuse.

## 3 Scope, Assumptions, and Goals

In our design of a health information accountability scheme, we make the following assumptions. While all of them may not be readily met today, we believe that, to

safeguard electronic healthcare records against emerging threats, the health IT infrastructure must evolve in the future, which will make these assumptions more realistic.

**(1) A trusted service on an online entity chosen or controlled by a patient.** It is used to deploy an "accountability agent" to process and log accountability tags on behalf of a patient. The agent must be available when patient data is consumed by some entity. It can be run at a trusted third party or the patient's own server hosted in a cloud. Techniques used to protect traditional security services (e.g., authentication servers) can be used to protect such an agent and are not discussed in this paper.

**(2) Public key infrastructure (PKI) for healthcare entities with trust anchors.** We believe this will be reasonable in future healthcare record sharing systems. For instance, NHIN Direct standards [2] involve PKI established with a regional healthcare information organization (RHIO) or other trusted entities as trust anchors. We assume that every participant in the EHR sharing is assigned a public/private key pair, which is authorized by one of the trust anchors so that other parties can trust it.

**(3) Private keys reliably authenticate the legitimate owners.** We assume that patients and others employ reasonable safeguards to protect private keys. Also, it is not easy to distinguish the case where a system user intentionally uses his private key from a case where the private key is misused by an unauthorized party, including malware installed on the insider's device. In this work, we treat them similarly since there is no distinction from the patient's perspective, and thereby the healthcare organization and the owner of the key should be held responsible in both cases anyway.

**(4) Adversaries can obtain meaningful gain only by presenting healthcare records to legitimate consumers.** Meaningful gain typically includes, in healthcare settings, financial gain and medical services, which are usually primary goals of cyber criminals. Financial or medical services are provided by legitimate entities, such as hospitals and insurance companies, so we believe this assumption holds. In addition, we assume that such legitimate consumers are motivated to verify the integrity and authenticity of data before taking actions based on it. This is also reasonable because such verification is beneficial for them to avoid fraud cases. Although data disclosure just for embarrassment or snooping celebrities' healthcare information out of curiosity are not covered by our scope, our system is still effective against the serious risk of medical identity theft [4] [10].

Our approach introduces an "accountability tag" that is attached by a health record repository to each copy of a healthcare record. Such a tag is checked and verified, when the record is shared and used, by an agent trusted by a patient who owns the data. To achieve information accountability, tags and associated protocols must satisfy the following properties.

**(a) Verifiability:** An accountability tag needs to contain the identity of the repository where the record is stored. Also, when the data is shared, identities of the source and destination of the health record sharing must be included in the tag. All of these identities must be carried in the tag in a cryptographically-verifiable way.

**(b) Unforgeability:** An accountability tag needs to be integrity protected to avoid tampering en route. In addition, a tag must be securely bound with its healthcare record. A malicious entity should not be able to modify or forge the tag to defeat our accountability goal without being detected.

**(c) Revocability:** A patient or a healthcare provider should be able to revoke an accountability tag. This property is required to minimize the risk posed by a stolen/compromised tag as well as misbehaving insiders.

**(d) Non-repudiation:** After an accountability tag is issued, the issuing repository and other entities involved in the health record sharing chain should not be able to repudiate their involvement in transfer of health data.

**(e) Assurance of Accountability:** Last but not least, it must be guaranteed that health records can not be meaningfully consumed by legitimate consumers without ensuring information accountability.

## 4 System Design

In this section, we first explain the high-level idea of the primary components of our design and then present the detailed protocol definition.

### 4.1 Patient-centric Monitoring Agent

As discussed, we need a trusted agent that works for each patient to verify accountability tags when the patient's healthcare records are consumed and also to log the information contained in the tags. To implement such an entity, we introduce a *patient-centric monitoring agent* and protocols that enforce mediation by the monitoring agent when healthcare records are updated and consumed (*Accountable Update* and *Accountable Usage*) [13]. We provide a brief overview of the agent and the operations it executes when data is accessed and updated. Detailed discussion can be found in [13].

*Accountable Update* protocol is executed when a health record issuer, for instance a doctor, creates a new healthcare record or submits updates to a health record repository. The architecture and protocol is shown in Figure 1. The record issuer creates a health record or update and makes his digital signature on it. This signature is encrypted with the monitoring agent's public key. Next, it sends the record and the encrypted signature to

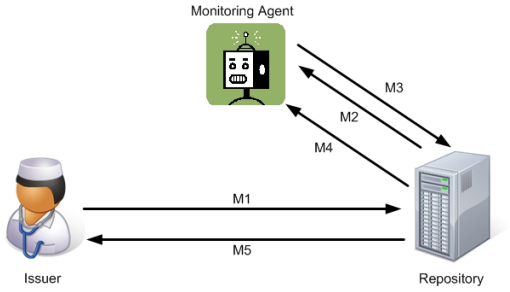


Figure 1: System Architecture and Message Flow in *Accountable Update*

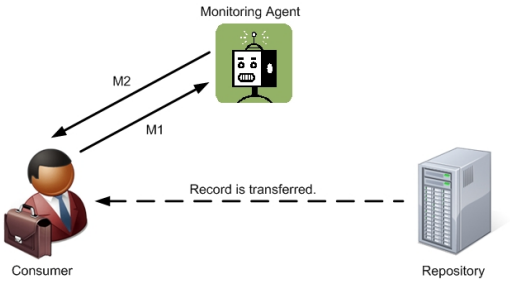


Figure 2: System Architecture and Message Flow in *Accountable Usage*

the repository (M1). The repository then contacts the patient's monitoring agent to obtain the proof of patient's authorization (M2 and M3). After accepting the record, the repository issues verifiable receipts to the monitoring agent as well as the issuer (M4 and M5).

The goal of *Accountable Usage* protocol is to allow patients to know when and by whom their records are meaningfully consumed (i.e. used only after verification of record issuers' signatures). This scope corresponds to assumption (4) mentioned in Section 3. The protocol is designed in such a way that the monitoring agent must be involved in the record verification process as shown in Figure 2. When receiving a healthcare record, a consumer needs to contact the monitoring agent (M1) because the signature is encrypted. The monitoring agent logs the usage and then decrypts the signature. Instead of returning the signature in plain text, it creates a non-transitive, designated signature using Universal Designated Verifier Signature scheme [16] so that only the specific consumer can be convinced with validity of the issuer signature. Then the designated signature is returned along with a transaction proof (M2). Finally, the consumer can verify the designated signature with the issuer's public key.

## 4.2 Accountability Tag

Next, we discuss the construction and handling of accountability tags. $A$ denotes an insider who is an employee of a healthcare organization that stores a patient $P$'s healthcare record. $A$ intends to share the record with an external entity $B$. $Repo_A$ denotes the repository of $A$'s organization. Since $B$ is external to the organization, it does not have direct access to $Repo_A$. By $[data]_{entity}$, we mean that $data$ is signed with $entity$'s private key. At the high level, construction of accountability tags and sharing of healthcare records are done as follows. The scheme is also illustrated in Figure 3.

1. $A$ authenticates itself to $Repo_A$ to request $P$'s record. The credential for this authentication can be different from $A$'s private key.

2. $Repo_A$ creates *PreTag* $= [CERT_A, M]_{Repo_A}$, where $CERT_A$ is $A$'s public key certificate and $M$ represents the metadata of the corresponding record, including the record's hash value. $M$ is stored on the repository with the record when *Accountable Update* was executed in the past.

3. $A$, before sharing the record with an external entity $B$, signs *PreTag* with its own private key along with $B$'s identity as destination, namely *Tag* $= [CERT_B,$ *PreTag*$]_A$. We call this task *tag activation*.

4. $A$ sends the record accompanied by its metadata $M$ and *Tag* to a recipient $B$ via encrypted and authenticated channel established with $A$ and $B$'s keys. Upon its receipt, $B$ can check if the entity that activated the tag, $A$ in this case, is actually the party sending the record and tag.

5. $B$ signs *Tag* before using it. We call this step *tag confirmation* and denote the resulting tag as *CTag*. When $B$ uses the health record at some consumer or submits the shared record to its repository, it needs to present *CTag* with the record.

Each accountability tag carries verifiable identities of the repository that released the copy of the healthcare record, the source of the sharing that downloaded a record from its repository, and the destination of the sharing (e.g., requesting entity in another organization). Three stages of a tag denoted as *PreTag*, *Tag*, and *CTag* correspond to these three identities that are to be verified. Accountability tags must be verified and logged when *Accountable Update* and *Accountable Usage* are performed, and the set of collected tags allows the patient's agent (and the patient) to construct the entire sharing path, as discussed in Section 4.4 and 5 in detail.

Intuitively, the way in which accountability tags are generated and handled is analogous to a personal check. Usually, personal checks are issued by a bank, whose name is printed on each copy along with an account holder's identity. When the account holder wants to

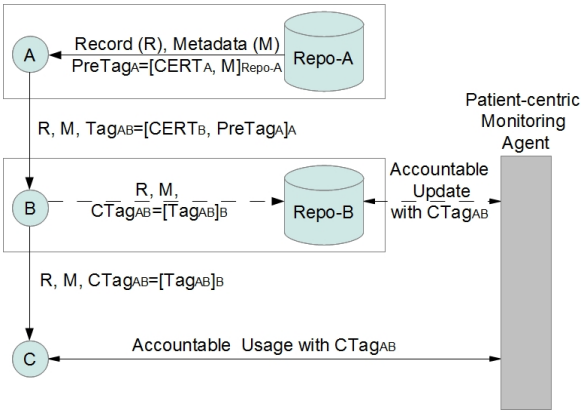


Figure 3: Overview of an accountability tag. $A$ downloads a record with *PreTag* from the repository, and shares the record and *Tag* with $B$ after tag activation. $B$, after tag confirmation, can either submit the record to its own repository (dotted arrows) or present the record and tag to a legitimate consumer ($C$). At the end of *Accountable Update*, the monitoring agent adds *Repo-B* to its *repository list*.

make payment, he specifies the recipient's name and makes his signature on it. Then, the check is handed to the recipient. Before cashing the check at a bank, the recipient needs to endorse the check. When the check is eventually presented at the bank, the bank can verify the chain of identities from the issuing bank to the recipient.

## 4.3 List of Authorized Repositories

We also introduce a *repository list* maintained by a patient's monitoring agent. It keeps track of the repositories authorized by the patient to store a copy of a certain healthcare record, i.e., repositories that successfully executed *Accountable Update* protocol with the monitoring agent; see also Figure 3. The repository list contains a list of $CERT_r$, which is a repository's public key certificate issued by a trust anchor, for each record, which can be uniquely identified by its hash value. We can use the list to check if a repository that issues *PreTag* is a legitimate place to store the corresponding record. Specifically, in *Accountable Update*, when a monitoring agent receives an authorization request from a repository, it checks whether the specified record is already stored in any other repository. If this is the case, assuming that the submitted record is shared from another entity, the monitoring agent additionally verifies that the request is accompanied by *CTag* that contains *PreTag* signed by one of the repositories in the list. On the other hand, a brand-new healthcare record does not require an accountability tag when being submitted.

In the *Accountable Usage* protocol, when the monitoring agent receives a verification request from a consumer, it confirms the presence of *CTag* and validity of signatures on the tag. In addition, it checks if the pair of the repository's identity, which is in *PreTag*, and the hash value of the record exists in its repository list. This verification against the repository list ensures that the record can be used only when *PreTag* is issued by an authorized repository. If any of these conditions are violated, the monitoring agent rejects the request and thereby the record can not be meaningfully used by the consumer. The information in *CTag* is logged by the patient's monitoring agent for later reference.

## 4.4 Protocol Details

Table 1: Notations used in Protocol Description

| Notation | Description |
|---|---|
| $PKE_k(p)$ | Encrypts a plain text $p$ using a public key $k$ under an asymmetric encryption scheme. |
| $PKE_k^{-1}(c)$ | Decrypts a cipher text $c$ using a private key $k$ under an asymmetric encryption scheme. |
| $SKE_k(p)$ | Encrypts a plain text $p$ using a secret key $k$ under a symmetric encryption scheme. |
| $SKE_k^{-1}(c)$ | Decrypts a cipher text $c$ using a secret key $k$ under a symmetric encryption scheme. |
| $Hash(d)$ | Computes a message digest of $d$ under a secure cryptographic hash function. |
| $Sign_k(d)$ | Computes a message digest of $d$, $Hash(d)$, and then signs it using a private key $k$. |
| $Verify_k(s)$ | Computes a message digest of data signed (omitted in the notation) and then verifies a signature $s$ using a public key $k$. |
| $SignEnc_{sk,pk}(d)$ | Sends $d$ via a secure and authenticated channel established by using a sender's private key $sk$ and a receiver's public key $pk$. |
| $RL-ADD(cert, H)$ | Adds the given certificate ($cert$) to the repository list of the record corresponding to the specified hash value ($H$). |
| $RL-LOOKUP(cert, H)$ | Returns whether the given certificate ($cert$) is included in the repository list of the record corresponding to the hash value ($H$). |
| $Matching(M, Tag)$ | Returns whether the given pair of healthcare record metadata ($M$) and accountability tag ($Tag$) is valid. Verification is done by comparing a hash value included in $M$ and one in $Tag$. |

Notations used in the protocol description are summarized in Table 1. The protocol uses two types of public/private key pairs. One is a key pair issued by a trust anchor, such as RHIO, called a *master key pair*. We denote a patient's (i.e. a record owner's) pair by $\{SK_o, PK_o, CERT_o\}$, an issuer's by $\{SK_i, PK_i, CERT_i\}$, a consumer's by $\{SK_c, PK_c, CERT_c\}$, a health record repository's by $\{SK_r, PK_r, CERT_r\}$, and a patient's monitoring agent's by $\{SK_m, PK_m, CERT_m\}$. $CERT_o, CERT_i, CERT_c$, and $CERT_r$ are issued by a trust anchor. $CERT_m$ is issued by the patient (i.e., signed with $SK_o$) and contains the location of the monitoring agent so that any party can interact with it. The other type of key pair is created under Universal Designated Verifier Signature (UDVS) scheme [16], and they are denoted as $\{pub_i, priv_i\}$ and $\{pub_c, priv_c\}$ respectively. Each of UDVS public key is signed with the entity's master private key, resulting in $cert_i$ and $cert_c$. The primitives in UDVS are summarized in Table 2.

The procedure to issue an accountability tag is shown in Figure 4. *Accountable Update* and *Accountable Usage* protocols with information accountability assurance are presented in Figure 5 and 6. The message flow and the architecture correspond to ones found in Figure 1 and 2 re-

**Protocol 4.1:** TAGGENERATION

$(\mathbf{M1})\mathbf{Src} \rightarrow \mathbf{Repository}: ID_{src},\ Credential_{src},\ RecordID$

$(\mathbf{P1})\mathbf{Repository}:$
$\begin{cases} \textbf{if } \{ID_{src},\ Credential_{src}\}\ is\ not\ valid \\ \quad \textbf{then } Abort \\ Load\ CERT_{src}\ based\ on\ ID_{src} \\ Load\ \{C_D,\ M,\ Auth\}\ based\ on\ RecordID \\ PreTag \leftarrow Sign_{SK_r}(M,\ CERT_{src}) \end{cases}$

$(\mathbf{M2})\mathbf{Repository} \rightarrow \mathbf{Src}:$
$SignEnc_{SK_r, PK_{src}}(\{C_D,\ M,\ Auth\},\ PreTag)$

$(\mathbf{P2})\mathbf{Src}:$
$\begin{cases} \textbf{if } Verify_{PK_r}(PreTag) = \textbf{false} \\ \quad \textbf{then } Abort \\ Load\ CERT_{dst}\ based\ on\ ID_{dst} \\ Tag \leftarrow Sign_{SK_{src}}(CERT_{dst},\ PreTag) \end{cases}$

$(\mathbf{M3})\mathbf{Src} \rightarrow \mathbf{Dst}:$
$SignEnc_{SK_{src}, PK_{dst}}(\{C_D,\ M,\ Auth\},\ Tag)$

$(\mathbf{P3})\mathbf{Dst}:$
$\begin{cases} \textbf{if } Verify_{PK_{src}}(Tag) = \textbf{false} \\ \quad \textbf{then } Abort \\ \textbf{if } Verify_{PK_r}(PreTag) = \textbf{false} \\ \quad \textbf{then } Abort \\ CTag \leftarrow Sign_{SK_{dst}}(Tag) \end{cases}$

Figure 4: Generation of Accountability Tag

spectively. The record metadata $M$ used in the protocols is defined as $\{C_S, C_{H_D}, H_{C_D}, cert_i, CERT_i, CERT_o, CERT_m, CERT_r\}$. $C_S$ is the encrypted issuer signature, and $C_{H_D}$ is the hash value of the record $D$, called $H_D$, encrypted with the monitoring agent's public key. $C_D$ is generated by encrypting $D$, using $H_D$ as the key. The hash value of $C_D$ is denoted as $H_{C_D}$ [13].

Table 2: Primitives of UDVS Scheme

| Notation | Description |
|---|---|
| UDVS-S($sk_s, m$) | Given $sk_s$ and a message $m$, outputs a publicly verifiable signature $sig$. |
| UDVS-PV($pk_s, m, sig$) | Given $pk_s, m$, and the corresponding $sig$, outputs the verification result. |
| UDVS-DS($pk_s, pk_v, m, sig$) | Given $pk_s, pk_v$, and the pair of $m$ and $sig$, generates a designated verifier signature $DVS$. |
| UDVS-DV($sk_v, pk_s, m, DVS$) | Given $pk_s, sk_v$, and the pair of $m$ and $DVS$, returns the verification result. |

In Figure 4, let $Src$ denote an insider who has access to the health record repository and $Dst$ be an external entity with whom $Src$ chooses to share a health record. Also, we assume that the repository implements a user authentication mechanism that reliably authenticates the requester and has access to the requesting insider's public key certificate. The last line of (P2) corresponds to tag activation. It is not unlikely that $Src$ wants to share the same records with multiple entities, instead of just one. In such a case, $Src$ can prepare a different tag for each recipient. The recipient ($Dst$), after receiving the tag and the record, makes a signature on the tag in (P3), which corresponds to tag confirmation. This is required before $Dst$ executes *Accountable Update* or *Usage*.

We need to consider two cases for *Accountable Update*. One case is when an issuer creates and submits a new record, and the other case is when a copy of a healthcare record shared by another entity is submitted. Because the former is almost identical to the *Accountable Update* protocol defined in [13] and only difference

**Protocol 4.2:** ACCOUNTABLEUPDATEWITHACCOUNTABILITYTAG

$(\mathbf{P1})\mathbf{Issuer}':$
$\begin{cases} Receive\ C_D,\ M,\ Auth,\ and\ CTag \\ Load\ CERT_{r'},\ and\ CERT_{i'} \\ \textbf{if } Verify_{PK_m}(Auth) = \textbf{false} \\ \quad \textbf{then } Abort \\ M' \leftarrow \{M,\ CERT_{r'},\ CERT_{i'}\} \\ MAC_{i'} \leftarrow Sign_{SK_{i'}}(M',\ Timestamp_{i'}) \end{cases}$

$(\mathbf{M1})\mathbf{Issuer}' \rightarrow \mathbf{Repository}':$
$SignEnc_{SK_{i'}, PK_{r'}}(C_D,\ M',\ MAC_{i'},\ Timestamp_{i'},\ CTag)$

$(\mathbf{P2})\mathbf{Repository}':$
$\begin{cases} \textbf{if } Verify_{PK_{i'}}(MAC_{i'}) = \textbf{false} \\ \quad \textbf{then } Abort \\ \textbf{if } (Hash(C_D) = H_{C_D}) = \textbf{false} \\ \quad \textbf{then } Abort \\ \textbf{if } Verify_{PK_o}(CERT_m) = \textbf{false} \\ \quad \textbf{then } Abort \end{cases}$

$(\mathbf{M2})\mathbf{Repository}' \rightarrow \mathbf{MoA}:$
$SignEnc_{SK_{r'}, PK_m}(M',\ MAC_{i'},\ CTag,\ Timestamp_{i'})$

$(\mathbf{P3})\mathbf{MoA}:$
$\begin{cases} \textbf{if } Verify_{PK_{i'}}(MAC_{i'}) = \textbf{false} \\ \quad \textbf{then } Abort \\ \textbf{if } Verify_{PK_i}(cert_i) = \textbf{false} \\ \quad \textbf{then } Abort \\ H_D \leftarrow PKE^{-1}_{SK_m}(C_{H_D}) \\ S \leftarrow PKE^{-1}_{SK_m}(C_S) \\ \textbf{if } UDVS - PV(pub_i, H_D, S) = \textbf{false} \\ \quad \textbf{then } Abort \\ \textbf{if } Verify_{PK_{i'}}(CTag) = \textbf{false} \\ \quad \textbf{then } Abort \\ \textbf{if } Verify_{PK_{src}}(Tag) = \textbf{false} \\ \quad \textbf{then } Abort \\ \textbf{if } Verify_{PK_r}(PreTag) = \textbf{false} \\ \quad \textbf{then } Abort \\ \textbf{if } RL - LOOKUP(CERT_r,\ H_D) = \textbf{false} \\ \quad \textbf{then } Abort \\ \textbf{if } Matching(M,\ CTag) = \textbf{false} \\ \quad \textbf{then } Abort \\ Log\ Timestamp_{i'},\ MAC_{i'},\ and\ CTag\ with\ M' \\ M'' \leftarrow \{C_S,\ C_{H_D},\ H_{C_D},\ cert_i,\ CERT_i, \\ \quad CERT_o,\ CERT_m,\ CERT_{r'}\} \\ Auth \leftarrow Sign_{SK_m}(M'') \end{cases}$

$(\mathbf{M3})\mathbf{MoA} \rightarrow \mathbf{Repository}':$
$SignEnc_{SK_m, PK_{r'}}(Auth,\ M'')$

$(\mathbf{P4})\mathbf{Repository}':$
$\begin{cases} \textbf{if } Verify_{PK_m}(Auth) = \textbf{false} \\ \quad \textbf{then } Abort \\ M \leftarrow M'' \\ Save\ C_D,\ M,\ and\ Auth \\ Rcpt_{r'} \leftarrow Sign_{SK_{r'}}(M, Timestamp_{r'}) \end{cases}$

$(\mathbf{M4})\mathbf{Repository}' \rightarrow \mathbf{MoA}:$
$SignEnc_{SK_{r'}, PK_m}(Rcpt_{r'}, Timestamp_{r'})$

$(\mathbf{M5})\mathbf{Repository}' \rightarrow \mathbf{Issuer}':$
$SignEnc_{SK_{r'}, PK_{i'}}(Rcpt_{r'}, Timestamp_{r'})$

$(\mathbf{P5})\mathbf{MoA}:$
$\begin{cases} \textbf{if } Verify_{PK_{r'}}(Rcpt_{r'}) = \textbf{false} \\ \quad \textbf{then } Abort. \\ RL - ADD(CERT_{r'}, H_D) \\ Log\ Rcpt_{r'}\ and\ Timestamp_{r'}\ with\ M' \end{cases}$

Figure 5: Accountable Update with Accountability Tag

is just updating the repository list, we omit the detailed definition due to space limitation. Since the submitted record is not stored on any other repositories, no tag is required to complete the protocol in this case.

Figure 5 defines the way in which a recipient of a shared healthcare record and *CTag* submits the healthcare record to its own repository. In this figure, $i$ and $r$ denote the original record issuer and the repository. On the other hand, $Issuer'$ (and $i'$) denotes a second-hand issuer who is submitting a received copy of the record to his repository, $Repository'$ (and $r'$). Verification of a tag by the patient's monitoring agent ($MoA$) is done at (P3), and its repository list is updated at (P5). The reposi-

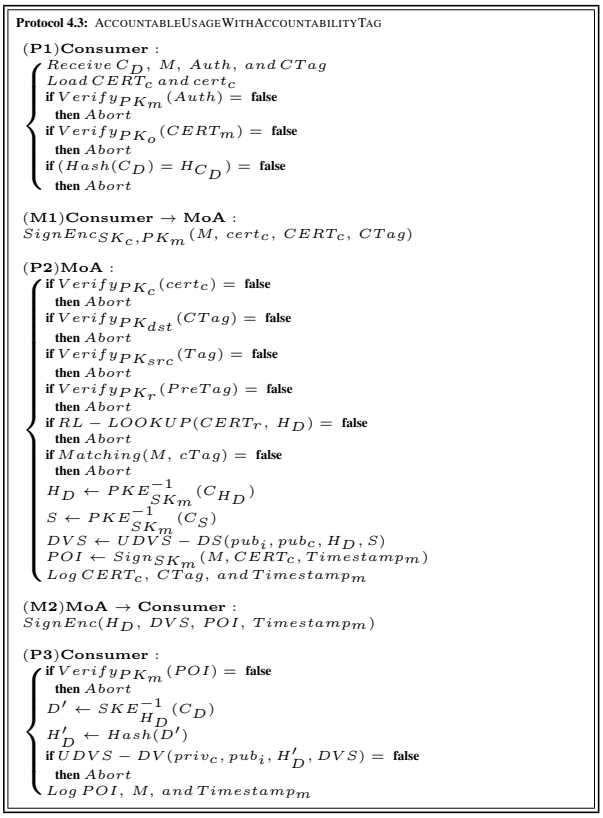


Figure 6: Accountable Update with Accountability Tag

tory list is updated after receiving a valid receipt from the repository, which implies that a repository that does not issue a valid receipt is not considered as an authorized repository for the corresponding record. In (P3), *CTag* is verified with $PK_{i'}$ because $Issuer'$ must be equal to $Dst$ of the tag. In addition, note that at the end of (P3), the record metadata is re-defined as $M''$ by using the certificate of $Repository'$. This step is necessary to replace the repository information while preserving the original issuer information. This modification does not conflict with the definition of a repository list since it is defined based on hash values of healthcare records, instead of the metadata $M$.

Regarding Figure 6, which is the protocol that a legitimate consumer of health records, such as Medicare, needs to follow when it verifies the authenticity and integrity of health records. $POI$ is a transaction proof used in [13] and is not directly related to the accountability tag scheme. The verification of *CTag* by the patient's monitoring agent is done at (P2). As discussed earlier, the repository list is checked in this step so that the request is processed only when the pair of the identity of the repository, which signs *PreTag*, and the record hash value is found on the list. Note that, $PK_{dst}$ and $PK_{src}$ are extracted from *CTag* while $PK_r$ comes from $M$.

In case $Dst$ needs to further share the data with another party, it must first insert the record to its (or its organization's) repository by using the protocol shown in Figure 5. This step is required for it to obtain a valid *PreTag* issued with its identity. Once the record is in its repository, $Dst$ needs to execute the protocol in Figure 4 as $Src$ to generate *Tag*. If $Src$ itself, who downloaded a record from a repository, intends to use the record, it can do so by specifying itself as the destination of *Tag* and confirming it with its own master private key.

## 5 Correctness

We show that the protocols presented in the previous section guarantee the properties that were identified for accountability tags in Section 3. Since the identities of the repository and the source and the destination of the sharing can be verified via digital signatures on *PreTag*, *Tag*, and *CTag* respectively, the property **(a) Verifiability** is satisfied. Likewise, digital signatures protect the integrity of the tag and do not allow repudiation by participating entities. Thus **(b) Unforgeability** and **(d) Non-repudiation** properties are also met. In addition, since a tag includes the metadata of the record ($M$), it is bound to a specific record and thereby can not be replayed with any other healthcare record.

**(c) Revocability** can be achieved by adding functionality to a patient's monitoring agent that acts on tags received by it. Specifically, a patient can create a revocation list of tags on her monitoring agent so that *Accountable Usage* or *Accountable Update* requests with certain tags can be denied at (P3) of Figure 5 and (P2) of Figure 6. Our design supports denial rules for identities of the source, destination, and/or repository. By using this feature, if misbehavior or private key compromise is reported, a patient can reject tags issued by the corresponding entity. Also, in case some repository is removed (e.g., when an organization running it is shut down), a patient can delete the repository from the repository list on her monitoring agent.

To understand how **(e) Assurance of Accountability** is satisfied, consider the setting discussed in Figure 3. In order for $B$ to successfully present a record to $C$, $B$ needs to have a valid *CTag* whose destination is $B$. In addition, *PreTag* in it must be signed by a repository on the repository list, which endorses $A$ as the source of sharing. Thus, the monitoring agent can know, when *Accountable Usage* is executed by the consumer $C$, that the record originated from *Repo-A* and is shared by $A$ with $B$, which satisfies our accountability goal. In case any of these signatures is invalid or absent, the request is not processed. Another way for $B$ to use a record is to submit the record to his repository once and then prepare a tag whose destination is set to itself. In this case, since the monitoring agent's repository list indicates that the same

record is already stored in *Repo-A*, $B$ needs to present a valid $CTag$ when submitting it to its repository. In this case, $CTag$ will allow the monitoring agent to learn that the record is shared from *Repo-A* to *Repo-B* via $A$ and $B$. If $B$ omits executing *Accountable Update*, *Repo-B* is never added to the repository list, and thereby it can not issue a valid *PreTag* for the corresponding record afterwards. Thus, we can again establish accountability and identify who is involved. It would be possible that $B$ forges a record. However, $B$ needs to submit the record to a repository to later use it. When such update is performed, the event is brought to patient's attention through her monitoring agent. If, for instance, $B$ is not a doctor, the update event could appear suspicious to the patient.

We now discuss accountability in a scenario where data is shared by multiple entities along a path. In the context of Figure 3, if $B$ wants to share a record further with another entity, say $D$ (not in the figure), who eventually presents the record to $C$, $B$ needs to execute *Accountable Update* (Figure 5) with *Repo-B* and then must prepare a tag to share the record with $D$. At this point, *Repo-B* is added to the repository list. As discussed above, the monitoring agent can learn how the record reached *Repo-B*. Then, with the record and *Tag* given by $B$, $D$ can successfully use the record at $C$ after generating *CTag*. When $C$ executes *Accountable Usage* (Figure 6) using *CTag* provided by $D$, the monitoring agent can additionally learn the sharing from $B$ to $D$. By combining the information obtained, the monitoring agent can know the path from *Repo-A* to $C$ via $A$, $B$, *Repo-B*, and $D$. Thus, accountability for the full sharing path can be attained. The same holds when additional hops are involved in the record sharing path. In this way, accumulated tags at the monitoring agent allow patients to re-construct the entire sharing path. User-friendly retrieval and presentation of the information logged at the monitoring agent are part of our future work.

## 5.1 Application Scenarios

Fist, recall the Cleveland Clinic case mentioned at the beginning of this paper [1]. In this scenario, Medicare is regarded as a legitimate consumer of healthcare records. Thus, Medicare, when processing healthcare records, should (and also is motivated to) verify the authenticity of the records by executing *Accountable Usage* protocol. Patients can know the usage of their records by Medicare via their monitoring agents. We can map the actors in this example to Figure 3 as follows: an insider who leaked the data as $A$, her collaborator (the cousin) as $B$, and Medicare as $C$. As discussed in the previous section, we can guarantee that the identity of the insider and the cousin are known to the patients, who are victims of fraud, when Medicare consumes the records presented by the malicious insider's cousin.

Next, let us consider the setting discussed in "Harmonized Use Case for Electronic Health Records (Laboratory Result Reporting)" by Office of the National Coordinator for Health Information Technology [5]. This use case focuses on the sharing of lab test results among doctors. Lab results are stored in an online repository, which is run by the lab itself or by a regional healthcare information organization (RHIO) and is accessible over a network. Under our scheme, when a doctor downloads the test result from the repository, *PreTag* that includes his identity is attached to the data. If the doctor wants to share it with another doctor, who eventually consumes the data for treating the patient, the sending doctor activates the tag with the destination's identity. A patient's monitoring agent is involved, when *Accountable Usage* is done by the latter doctor, and is informed of both doctors' identities through the accountability tag. Such visibility allows patients to determine whether the sharing is reasonable. Furthermore, when the recipient adds the shared record to his own repository, he must execute *Accountable Update* with the confirmed tag so that the record can be later shared or used. After the completion of the protocol, the patient can know that the copy of the lab result is also stored at the new location.

We can also deal with a scenario with integrated healthcare records from multiple sources, which is often the case with continuity of care documents (CCD). For instance, in Figure 3, $B$ obtains a record ($R_a$) from $A$ and submits it to the repository. $B$ could obtain another record ($R_d$) from another entity, say $D$, in the same way. Later, $B$ might want to share, with another entity $E$, a CCD consisting of $R_a$, $R_d$ and $R_b$ created by $B$. In this situation, we can still ensure accountability for both $R_a$ and $R_d$ by attaching a separate tag for each of them in addition to one for $R_b$. Then, each verification of the issuer signature is done separately by using the corresponding accountability tag, and thereby the patient can know that $R_a$ is released from *Repo-A*, shared by $A$ with $B$, submitted to *Repo-B*, and finally shared with $E$ by $B$. The same holds for $R_d$. $B$ could choose to issue the combined record as a new record by making his own signature on the combined CCD, but it is unlikely because, in that case, $B$ is considered as the issuer of the entire document and must be responsible for all of its contents. In the future, we will explore a scheme to efficiently combine and verify multiple accountability tags.

# 6 Security Discussion

In our security analysis, we start with a trust model that assumes verification of tags solely relies on a trusted patient-centric monitoring agent. Repositories and consumers are required to forward tags to each patient's

monitoring agent when running the protocols. If they become compromised and do not forward the tags, they will not be successful in completing the protocols and the data cannot be meaningfully utilized (directly or after sharing it) by presenting it to legitimate consumers. Repositories are also responsible for issuing *PreTag*, including the requesting insider's identity. If it does not sign *PreTag*, the corresponding healthcare record can not be verified by legitimate consumers. Even if a misbehaving repository specifies another insider's identity as a requester, our scheme would not allow misuse of healthcare data as long as the compromised repository (or an adversary controlling it) does not have access to the claimed entity's private key.

To counter *CTag* misuse, repositories and consumers should verify the identity of the requester against the one claimed in the tag, for example through authenticated communication. By doing so, as long as the private key used to confirm the tag is not compromised, misuse can be prevented at repositories and consumers. We believe legitimate parties are motivated to do so to avoid being involved in fraudulent activities. Setting a lifetime for a tag upon signing, which is enforced by a patient's agent, can provide added protection against *CTag* misuse. Another countermeasure our scheme can offer is revocation of tags. Once informed of a breach, patients can immediately update revocation lists on their monitoring agents to reject tags activated or issued by a certain healthcare organization (or a specific insider in the organization). Also note that, even in case of *CTag* misuse, the usage of healthcare records at consumers are brought to patient's attention so that involved entities are investigated.

Next, we discuss typical threats in E-healthcare systems, namely malicious insiders and lost/stolen devices.

**Malicious Insiders:** As discussed in Section 5, cases where a malicious insider intentionally shares healthcare records with an external entity can be addressed by our scheme. For malicious parties to successfully use the leaked records at legitimate consumers, each of the records must have an associated tag that reveals identities of involved entities to a patient's monitoring agent. In case a malicious insider (or an external adversary) somehow compromises another insider's private key and his identity credential to gain access to his or his organization's repository, healthcare records could be leaked and misused under the identity of the owner of the compromised private key. Regardless whether private keys are compromised or not, our scheme allows a patient to learn whose private keys are involved on the sharing path and thereby are to be investigated. Thus, protection of private keys is crucial for users in our system.

**Lost/Stolen Devices:** Even in case devices or storage used in healthcare organizations are stolen, legitimate consumers will not accept such compromised records as long as they are not accompanied by valid *CTag*s. If valid tags are also stored on the stolen device, the records could be misused by adversaries as discussed above. Thus, *CTag* should not remain on the devices after its usage. Again, resulting misuse of the stolen data will be attributed to the entity that contributed to *CTag*, so we can expect that each honest participant follows such security guidelines. Another possible way for the device thief to misuse stolen healthcare data is to submit it to some repository once. However, if an adversary does so, through *Accountable Update*, the event is reported to the patient, who can alert the repository.

In addition to the threats discussed above, malware infection could be a serious threat. To prevent misuse by malware, again securing private keys and *CTag*s is the key. Such protection can be accomplished, for example, by means of system virtualization, as discussed in [14].

It may be argued that malicious sharing and usage of healthcare data that do not follow the protocols defined in this paper would not be captured. However, legitimate consumers, such as insurance companies, Medicare, and hospitals, are, for their own protection, naturally motivated to run *Accountable Usage* protocol. Moreover, in case unformatted data, e.g., data that is copied or extracted from a healthcare record, is presented, the consumer has no way to verify it, so accepting such a request would be risky. Our goal is not to implement a complete information flow control but to ensure accountability when the data reaches such legitimate consumers.

Lastly, a monitoring agent could become a single point of attack or failure in the healthcare record sharing architecture. However, monitoring agents can be distributed and each patient can have multiple monitoring agents. Moreover, as discussed in [13], a monitoring agent can be legitimately bypassed in case a patient is physically present at a point of care. Thus, disabled monitoring agents will not result in serious consequences.

## 7 Prototype Implementation

Table 3: Components in Prototype Implementation

| Name | Description |
|---|---|
| Issuer Tool | This tool is used by either an entity that is creating a new healthcare record or submitting a copy of a healthcare record shared from another entity. This tool is implemented as a Java application. |
| Consumer Tool | When a healthcare record consumer verifies the signature on a healthcare record (i.e. properly consumes a healthcare record), this tool is used. This tool is implemented as a Java application. |
| Repository | This stores and manages electronic healthcare records submitted by issuers. It also handles user authentication and *PreTag* generation. A repository is implemented as a Java Servlet. |
| Monitoring Agent | This is responsible for logging the information obtained in *Accountable Update* and *Accountable Usage* protocols as well as verifying accountability tags. This is implemented as a Java Servlet. |

To demonstrate feasibility and to study performance of requests when accountability tags are processed, we implemented a prototype system consisting of the components shown in Table 3, emulating a setting of NHIN Direct as also done in [13]. We utilized BouncyCastle cryp-

tography library (http://www.bouncycastle.org/) and Java pairing-based crypto library (http://gas.dia.unisa.it/projects/jpbc/). We deployed an *issuer tool* and a *consumer tool* on a laptop PC (Pentium M 750 processor and 2GB RAM) connected to a commercial cable TV Internet while a *repository* and a *monitoring agent* are set up on a machine (Intel Xeon 5150 processor and 8GB RAM) connected to the Georgia Tech campus network.

We summarize the overheads introduced by information accountability in Table 4 through the comparison with the system in [13] that did not support accountability tags. Each number is the average or standard deviation over 20 measurements. In addition, we used two files of different sizes (3MB and 6MB). Regarding *Accountable Update*, the measurements with accountability are made based on the protocol defined in Figure 5. On the other hand, the protocol without accountability [13] requires additional operations at an issuer, such as preparation of UDVS signatures. Thus, we can not compare the results directly. However, we can see that the time to complete *Accountable Update* requests is comparable. For other requests, we can see that overheads introduced for robust accountability are far less than 1 second.

Table 4: Overhead Summary

| Task | Mean (Std. Dev.) w/ Accountability [ms] | Mean (Std. Dev.) w/o Accountability [ms] | Mean Overhead [ms] |
|---|---|---|---|
| Tag Activation | 15.47 (4.08) | 0 | 15.47 |
| Tag Confirmation | 15.18 (4.29) | 0 | 15.18 |
| Acct. Update (3M) | 4530.44 (82.35) | 4957.17 (227.27) | - |
| Acct. Update (6M) | 9117.53 (123.28) | 9834.60 (62.62) | - |
| Acct. Usage (3M) | 1345.83 (106.3) | 1151.33 (113.99) | 194.50 |
| Acct. Usage (6M) | 1792.65 (41.58) | 1560.31 (149.28) | 232.34 |

## 8 Conclusion

We proposed a way to enhance information accountability in electronic health record sharing by means of accountability tags, which are metadata attached to electronic healthcare records. We discussed how accountability tags can be implemented and how they can help establish actionable information accountability for patients when their health records are presented to consumers, such as hospitals and insurance companies.

Full integration of our scheme would incur modifications in existing systems, but systematic support for patient awareness, accountability, and control must be pursued to meet HIPAA/HITECH privacy requirements. We hope our work motivates future research for developing patient-centric E-healthcare systems.

## References


[1] 3rd HIPAA criminal case hints at federal tactics. http://www.ama-assn.org/amednews/2006/10/16/gvsb1016.htm.

[2] Direct Project. http://wiki.directproject.org/.

[3] Health Information Privacy. http://www.hhs.gov/ocr/privacy/hipaa/administrative/breachnotificationrule/breachtool.html.

[4] Medical Identity Theft. http://www.ftc.gov/bcp/edu/pubs/consumer/idtheft/idt10.shtm.

[5] Harmonized Use Case for Electronic Health Records (Laboratory Result Reporting). http://healthit.hhs.gov/portal/server.pt/gateway/PTARGS_0_10731_848103_0_0_18/EHRLabUseCase.pdf, 2006.

[6] Kaiser Permanente Laptop Stolen. http://www.consumeraffairs.com/news04/2006/11/kaiser_laptop.html, 2006.

[7] Patients' Data on Stolen Laptop. http://www.washingtonpost.com/wp-dyn/content/article/2008/03/23/AR2008032301753.html, 2008.

[8] R. Aldeco-Pérez and L. Moreau. Provenance-based auditing of private data use. In *BCS Int. Acad. Conf.*, pages 141–152, 2008.

[9] E. Bertino, B. C. Ooi, Y. Yang, and R. H. Deng. Privacy and ownership preserving of outsourced medical data. In *ICDE*, pages 521–532, 2005.

[10] M. E. Johnson. Data hemorrhages in the health-care sector. In *Financial Cryptography*, pages 71–89, 2009.

[11] T. Kifor, L. Varga, S. Álvarez, J. Vázquez-Salceda, and S. Willmott. Privacy issues of provenance in electronic healthcare record systems. *Journal of Autonomic and Trusted Computing (JoATC)*, 2008.

[12] J. King, B. Smith, and L. Williams. Modifying without a trace: General audit guidelines are inadequate for electronic health record audit mechanisms. In *Proceedings of ACM IHI 2012*, 2012.

[13] D. Mashima and M. Ahamad. Enhancing accountability of electronic health record usage via patient-centric monitoring. In *Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium*, 2012.

[14] D. Mashima, A. Srivastava, J. Giffin, and M. Ahamad. Protecting E-healthcare Client Devices against Malware and Physical Theft. In *1st USENIX Workshop on Health Security and Privacy*, 2010.

[15] L. Moreau, P. T. Groth, S. Miles, J. Vázquez-Salceda, J. Ibbotson, S. Jiang, S. Munroe, O. F. Rana, A. Schreiber, V. Tan, and L. Z. Varga. The provenance of electronic data. *Commun. ACM*, 51(4):52–58, 2008.

[16] R. Steinfeld, L. Bull, H. Wang, and J. Pieprzyk. Universal designated-verifier signatures. *Advances in Cryptology-Asiacrypt 2003*, pages 523–542, 2003.

[17] D. J. Weitzner, H. Abelson, T. Berners-Lee, J. Feigenbaum, J. A. Hendler, and G. J. Sussman. Information accountability. *Commun. ACM*, 51(6):82–87, 2008.