J Med Syst
DOI 10.1007/s10916-010-9639-0

ORIGINAL PAPER

Redactable Signatures for Signed CDA Documents

Zhen-Yu Wu · Chih-Wen Hsueh · Cheng-Yu Tsai · Feipei Lai · Hung-Chang Lee · Yufang Chung

Received: 24 October 2010 / Accepted: 7 December 2010

Abstract The Clinical Document Architecture, introduced by Health Level Seven, is a XML-based standard intending to specify the encoding, structure, and semantics of clinical documents for exchange. Since the clinical document is in XML form, its authenticity and integrity could be guaranteed by the use of the XML signature published by W3C. While a clinical document wants to conceal some personal or private information, the document needs to be redacted. It makes the signed signature of the original clinical document not be verified. The redactable signature is thus proposed to enable verification for the redacted document. Only a little research does the implementation of the redactable signature, and there still not exists an appropriate scheme for the clinical document. This paper will investigate the existing web-technologies and find a compact and applicable model to implement a suitable redactable signature for the clinical document viewer.

Keywords Clinical document architecture · Health level seven · XML signature · Redactable signature · Implementation

Z.-Y. Wu (✉) · C.-W. Hsueh · C.-Y. Tsai · F. Lai
Department of Computer Science and Information Engineering,
National Taiwan University,
Taipei, Taiwan
e-mail: d96922021@ntu.edu.tw

F. Lai
Department of Electrical Engineering,
National Taiwan University,
Taipei, Taiwan

H.-C. Lee
Department of Information Management, Tamkang University,
Taipei, Taiwan

F. Lai
Graduate Institute of Biomedical Electronics and Bioinformatics,
National Taiwan University,
Taipei, Taiwan

Y. Chung
Department of Electrical Engineering, Tunghai University,
Taipei, Taiwan

C.-W. Hsueh
Graduate Institute of Networking and Multimedia,
National Taiwan University,
Taipei, Taiwan

Introduction

With the rapid development of the Internet, various technologies for applications are maturing, leading to the digitization and electronic orientation of our daily-life activities, such as e-commerce, e-medicine, e-banking, e-government, and e-society. In region of e-medicine, these developments, including the use of the healthcare smart cards, digital certificates and signatures for medicine applications, electronic patient records and prescription records, and so on, have come to mature and been used in most hospitals or medical institutes during the last decade. This field, nonetheless, remains as one of the most popular researches.

A clinical document, one type of electronic patient records contains highly sensitive personal health information, its authenticity and integrity need to be ensured while the data are exchanged over insecure networks. This has increased the high attentions on secure medical data exchange among hospitals or other healthcare organizations. The Clinical Document Architecture (CDA) proposed by the Health Level Seven International (HL7) [1]

Published online: 23 December 2010

nowadays has been the standard for interchanging clinical documents among heterogeneous systems in the area of healthcare.

There already exists a technology called digital signature for secure data exchange. A digital signature is basically a scheme for assuring the authenticity and integrity of a digital message or document. It means that a message or document recipient can verify the message or document by his individual signature and then confirm messages not being interpolated during transit. Such technologies have also been applied to CDA documents by using the XML Signature (XML-DSig) published by W3C [2].

Nevertheless, appropriate alterations of some signed clinical documents should be allowed because privacy requirements are there other than the integrity of the document. Two scenarios about protecting the privacy of patients are considered. First is the question of morality. From an empathy perspective, the condition or illness of patients should not be publicly disclosed, no matter what disease they have or might have (especially sexually transmitted diseases, and mental illnesses); they would certainly like to keep them private, other than from their physicians. Secondly, stemming from privacy protection, a patient can be assured of the security of his clinical data for the public use for medical research by institutions in order to find better medical treatment to treat the illness [3, 4]. This may be the biggest advantage that can be gained from ensuring patients' privacy [5].

Therefore, a clinical document of a patient concealed some personal or private information is required, i.e. the de-identification process. However, as the procedures of de-identification are done digitally, the signed XML-DSig of the original document will not be verified correctly for the de-identified document. The same situation would also happen while partial contents of a document are given to another physician for some other diagnostic opinions. These are named the digital document sanitizing problems as well [6]. Some discussions about these problems are proposed in [7] and [8].

The redactable signature, a signature allows users to execute redaction technology for protecting the privacy but can still confirm the authenticity of digital documents [9, 10], intends to model a situation where a censor can delete certain substrings of a signed document without destroying the ability of the recipient verifying the integrity of the resulting document, also named redacted document [11]. More specifically, the redactable signature scheme allows the censor to remove parts of the document and then inserts a special symbol representing the location of the deletion so that the document can still be verified.

Some related research about redactable signatures is introduced in the following. In fact, redactable signatures are examples of homomorphic signatures which were introduced by Rivest in his talks on Combridge College [12] and formalized by Johnson et al. Micali and Rivest also proposed a transitive signature scheme as the first construction of homomorphic signatures [13]. The notions on homomorphic signatures could be traced back to incremental cryptography, introduced by Bellare, Goldreich and Goldwasser [14, 15]. Subsequently, Johnson et al. [11] introduced redactable signature schemes which enabled the verification of a redacted signed document. Signature scheme with similar property has also been proposed for XML documents [16]. And various redactable signatures and related signature schemes are continually proposed [7, 8, 17–19].

The majority of research on redactable signatures and related signature schemes focus on the theoretical definitions and models, and only a little research has been done on the implementation. For achieving the goal of using the redactable signatures to protect privacy in reality and not research by theoretical interests, the main purpose of this paper is to investigate a feasible solution to implement the applicable redactable signature scheme for the CDA document. Moreover, it is undertaken to understand how modern web technology could help improve the medicine environments to suggest practical implications of this area.

With regard to the needs of the implementation, since the redactable signature requires lots of expensive computations, the CPU speed, the space of memory or hard disks, and the network bandwidth are suggested to be as good as possible. For example, the Intel Core i3-550 CPU, 4 G memory, and 320 GB hard disk are used to start the implementation in this study. Moreover, Python, a popular programming language nowadays, is chosen to develop the signature and run on the Windows 7 operation system. The more details will be introduced in Section "Proposed methodology".

The rest of this paper is organized as follows. Section "Preliminary" introduces the CDA standard and the related background knowledge of redactable signatures. Section "Proposed methodology" illustrates the proposed methodology for implementing the redactable signature satisfying the needs of privacy and security. Following, evaluations of the proposed approach are shown in Section "Analysis of proposed approach", and conclusions are drawn in Section "Conclusions and future work".

Preliminary

This section first provides a brief overview of the CDA standard, and some background knowledge of public-key cryptosystem and digital signature. Then, some information about the W3C recommendation of XML-Signature syntax and processing is followed. Finally, the concept of redactable signature and its related knowledge are described.

Clinical document architecture

The CDA is an XML-based markup standard intending to specify the encoding, structure, and semantics of clinical documents for exchange [20]. A CDA document uses the XML schema to wrap its contents and is defined to have the following six characteristics by HL7.

1. Persistence: A clinical document continues to exist in an unaltered state, for a time period defined by local and regulatory requirements.
2. Stewardship: A clinical document is maintained by an organization entrusted with its care.
3. Potential for authentication: A clinical document is an assemblage of information that intends to be legally authenticated.
4. Context: A clinical document establishes the default context for its contents.
5. Wholeness: The authentication of a clinical document applies to the whole and does not apply to portions of the document without the full context of the document.
5. Human readability: A clinical document is human readable.

```
<ClinicalDocument>
   . . . CDA Header . . .
   <structuredBody>
      <section>
         <text>. . .</text>
         <observation>. . .</observation>
         <substanceAdministration>
            <supply>. . .</supply>
         </substanceAdministration>
         <observation>
            <externalObservation>. . .
            </externalObservation>
         </observation>
      </section>
      <section>
            <section>. . .</section>
      </section>
   </structuredBody>
</ClinicalDocument>
```

Fig. 1 Major components of a CDA document

A CDA document has the ability to contain any kinds of clinical contents, such as discharge summaries, referrals, care provision information notes, and so on [21]. Figure 1 shows an example of major components of a CDA document.

In recent years, considerable concerns have been arisen over the use of the CDA among different applications and heterogeneous system environments [22–24]. Specifically, the CDA is used for exchanging the clinical documents between different hospitals and healthcare organizations.

Public-key cryptosystem

The concept of public-key cryptosystems was first proposed by Diffie and Hellman in 1976 [25], which opened a new direction on cryptography development. Since then, many researchers started proposing various types of public-key cryptosystems.

Public-key cryptography is a kind of asymmetrical encryption technique. Each party in such cryptosystem holds two keys; one is the public key used for encrypting a datum and the other is the private key used for decryption. Usually, a cryptosystem is used to protect the data transmitted through the Internet from being tampered by an illegal third party. For instance, a document is encrypted by a receiver's public key before it is sent out. Thus, the encrypted document can only be derived by a receiver who uses his own private key. It is very difficult for an illegal party to decrypt the contents of the document, except when the receiver's private key is obtained and used for decrypting the message. Although the cryptosystem can make data transmission become more confidential and convenient, both the sender and the recipient must hold each other's key sets at the same time in order to perform the encryption and decryption tasks. This is not practical enough. Therefore, a public-key infrastructure (PKI) method is proposed to solve this impractical problem [26].

A PKI scheme is constructed on the basis of the public-key cryptosystem framework so as to offer all the security requirements, including authentication, confidentiality, message integrity, and non-repudiation. Certificate authority (CA) is a part of the PKI scheme. The CA is a Trusted Third Party (TTP) in the scheme that manages and issues the certificates to the requesters and provides services such as keeping public keys, offering directory service, and issuing certificates. Under the PKI scheme, both parties are capable of exchanging information securely and safely with each other on the network.

Digital signature

All specific digital contents are capable of being encrypted and decrypted to ensure their integrity and non-repudiation. With the agreement from all related parties, digital signatures are valid to be used in private communication. The concept of digital signatures originally coming from cryptography is a way to encrypt or decrypt senders' text messages by applying a hash function to keep the messages secure when being transmitted [9, 10, 27].

A one-way hash function is a mathematical algorithm, which takes any length of a text message as the input and gives an output in a specific length. Its main function keeps the encrypted output very difficult being derived by a third party [26]. Based on one-way hash functions, a digital signature scheme can be done as follows.

Based on a Public-key cryptosystem, a sender firstly uses a one-way hash function to convert an electronic record into a text message of a specific length, which is called the message digest [10]. Then, the sender will use his private key to sign on the message digest generating a digital signature. As the recipient receives the message and its signature, he can use the sender's public key to verify the signature for the message through the hash function. If the calculated message is not the same as the message itself, it is possible that a wrong document is outputted because of tampering. On the contrary, it is the valid document that the recipient wants. Obviously, this method can help to ensure data transmission security. Figure 2 demonstrates the procedure of signing a digital signature.

XML signature

The XML signature, also called XMLDsig, XML-DSig, and XML-Sig, is a W3C recommendation that defines XML syntax for the digital signature [2]. As shown in Fig. 3, the XML digital signature is represented by the *Signature* element which has the following structures, as "?" denoting zero or one occurrence, "+" denoting one or more occurrences, and "*" denoting zero or more occurrences.

Generally speaking, the XML signature has the following features:

1. The XML signature can sign any kinds of resources which are addressable by a Uniform Resource Identifier (URI) [28]. The resource could be elements within the same XML document (in this specification, a 'same-document' reference is defined as a URI-Reference consisting of a hash sign '#' followed by a fragment or alternatively consisting of an empty URI) or elements in an external XML document identified by the URI, even a non-XML resource, e.g. image. The XML signature is also applied to everything which can be located by the URI. In other words, the signature target could be dynamic.
2. The XML signature can sign the entire XML document, or only the selected elements.
3. The XML signature can support both asymmetric key algorithms (Digital Signatures) and symmetric key algorithms (HMAC) for signing.

From above descriptions, it clearly shows that the XML signature supports signing the various external resources of

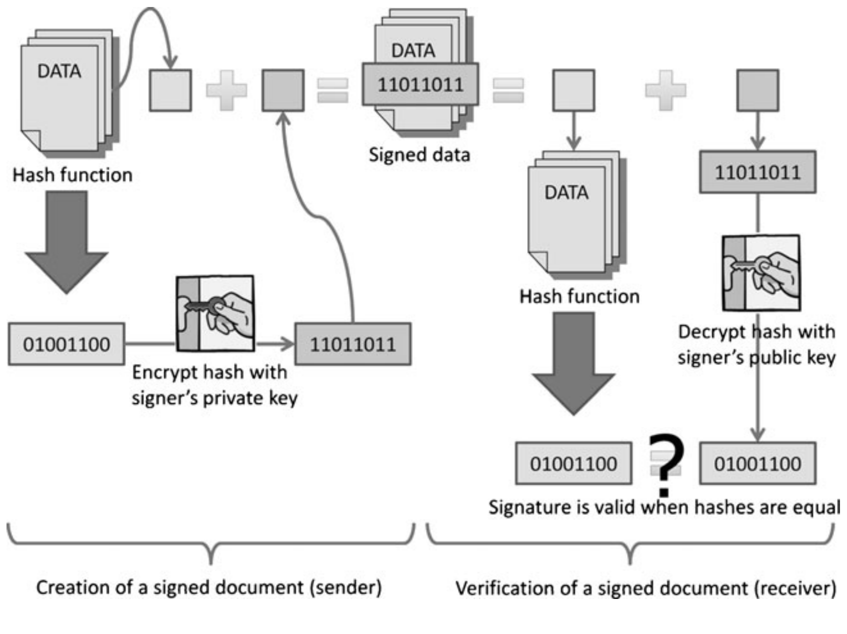

DATA
Hash function
01001100
Encrypt hash with signer's private key
11011011
DATA
11011011
Signed data
DATA
Hash function
11011011
Decrypt hash with signer's public key
01001100
?
01001100
Signature is valid when hashes are equal
Creation of a signed document (sender)
Verification of a signed document (receiver)

Fig. 2 The procedure of signing a digital signature

```
<Signature ID?>
  <SignedInfo>
    <CanonicalizationMethod/>
    <SignatureMethod/>
    (<Reference URI?   >
      (<Transforms>)?
      <DigestMethod>
      <DigestValue>
    </Referecne>)+
  </SignedInfo>
  <SignatureValue>
 ( <KeyInfo>)?
 ( <Object ID?>)*
</Signature>
```

Fig. 3 Major components of an XML signature

the XML documents, the authenticity and integrity of the CDA document and is thus guaranteed.

Redactable signature

The redactable signature is a signature allowing users to execute redaction technology for protecting the privacy but still being able to confirm the authenticity of digital documents [9, 10]. In other words, the redactable signature incorporates the features of redaction technology and the properties of digital signatures. The signature intends to model a situation where a censor can delete certain substrings of a signed document without destroying the ability of the recipient to verify the integrity of the resulting (redacted) document [11]. More specifically, the redactable signature scheme allows the censor to remove parts of the document and then inserts a special symbol representing the location of the deletion that the document can still be verified.

The main difference between a conventional digital signature scheme and a redactable signature scheme is that the hash function is applied to the entire document in the former scheme, whereas in the latter, the document is firstly split into redactable parts and the hash function is then applied to each of them. Finally, the signature is generated through encrypting those hashes using the signer's private key. The signing workflow of a redactable signature scheme is shown in Fig. 4.

How the redaction works in the redactable signature is further shown in Fig. 5. The basic idea of redaction is that (a) the censor may delete some portions of a document which belongs to the redactable parts and (b) replace them with the corresponding hashes. Since the redactable parts are filled with specific hash values from Step (b), the recipient can successfully execute the whole verification process by using the same signature without errors. In Contrast with the common digital signature, if some portions of the document are removed, the hash function could not be calculated and thus the original signed signature is not verifiable with the redacted document.

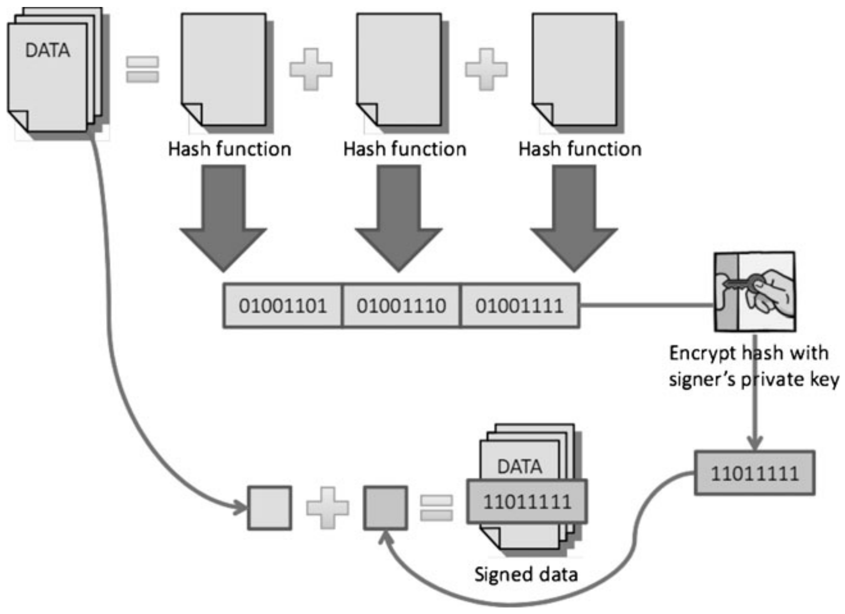


Fig. 4 Signing workflow of a redactable signature scheme

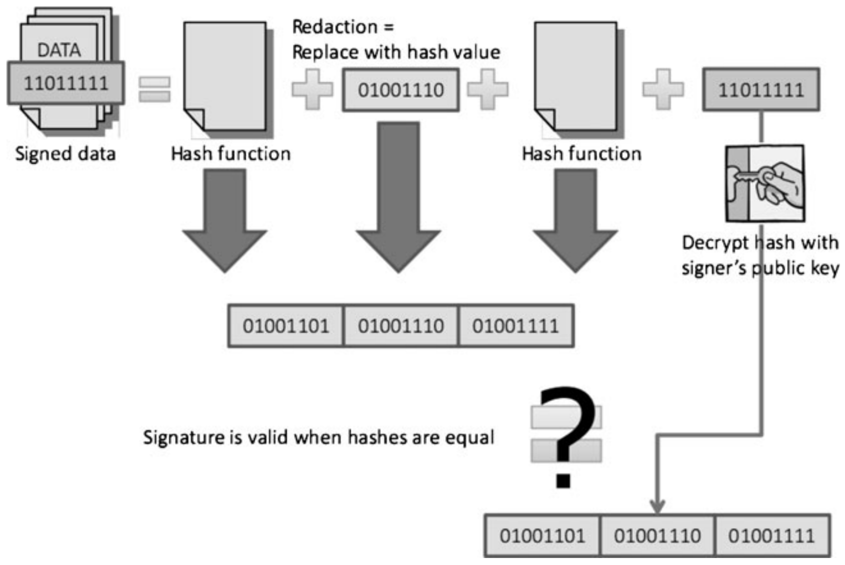


Fig. 5 Demonstration of a redactable signature being verified

Proposed methodology

In this section, the implementation of the applicable redactable signature, i.e. a CDA document viewer, which is capable of using the redactable signature to sign CDA documents, is demonstrated. Firstly, various possible implementation approaches are discussed. Then, some useful advantages among them are picked for the integration. Finally a feasible model for implementing the redactable signature is proposed.

Existed possible approaches

There are basically two kinds of approaches to implement the redactable signature, namely conventional program approach and browser-based approach. The implemented programming languages, such as C, C++, C#, Java, Perl, PHP, Python, and so on, are classified as the conventional program approach. The conventional program approach has high flexibility since designers have full control of the environment when running the program. Another approach, the browser-based approach, contains the approaches with the use of browser, e.g. the web application, Adobe Flash, or the Google Native Client [29]. This approach has high portability since web browsers are available for various platforms.

To let the redactable signature model possess high portability and flexibility, after investigating the different approaches above mentioned, an appropriate signature architecture which can integrate both the conventional program and browser-based approaches is proposed in this study. Python, an excellent programming language available on every platform and able to maintain the acceptable performance, is thus decided to be a tool for practicing the signature. And, other relevant technologies are also applied to satisfy the implementation needs. All of them will be introduced in the following sections.

Implementation

The basic idea of the proposed methodology to implement the CDA document viewer that is capable of dealing with a redactable signature is shown in Fig. 6.

The proposed viewer has the following components.

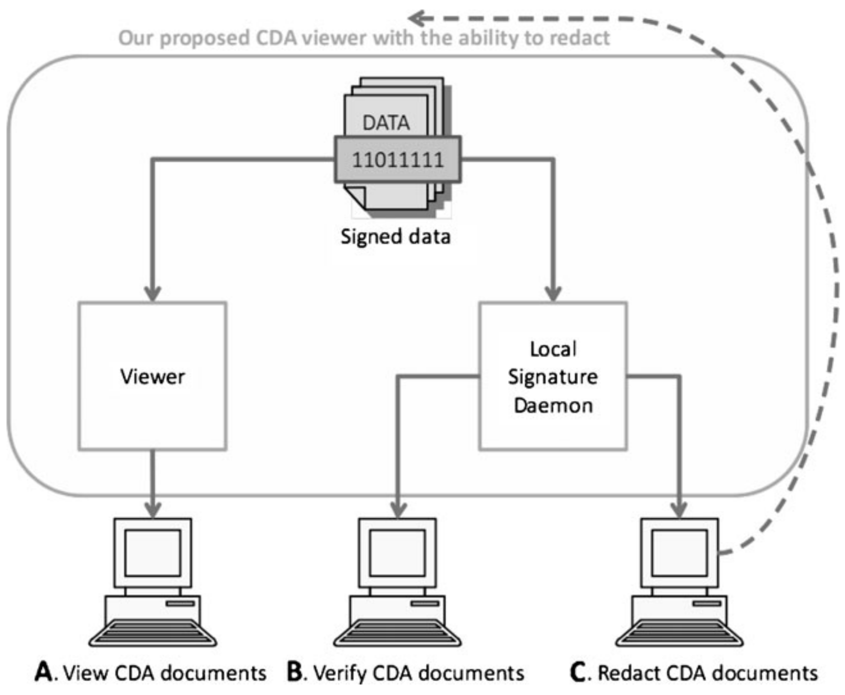
1. Viewer, it is for converting CDA documents into html format that browsers could handle with.
2. Local signature daemon, the local signature daemon is basically a tiny web server that handles the requests from local browser and then generates digital signatures or message digests for the given requests.

For better performance of the conventional program approach while retaining the portability of the browser-based approach at the development of viewer and local signature daemon, the objective is to write programs which do one thing and do it well (the UNIX philosophy); technologies such as the XPath [30], JSON [31], and the principle of least privilege [32] are also used in the development. Those two components will be introduced in detail in the following sections.

Long polling

Asynchronous JavaScript and XML (AJAX) is a group of interrelated web development techniques used in the client-side to create interactive web applications. With AJAX, web applications can retrieve data from the server asynchronously in the background without interfering with the display and behavior of the existing web page.

Fig. 6 The proposed CDA viewer architecture



The AJAX technology is applied in the viewer component that the user experience depends on the response time of corresponding operation. Figure 7 shows that, with polling, an AJAX-enabled application will poll the server for data regardless whether the data on the server side are changed or not.

Taking online chat rooms as an example, clients poll the server every *N* seconds to see if new messages are available. More specifically, the browser requests the server for updating the contents during every set interval. One problem might occur when no new message comes into the server, the response of the request will not contain any data and become meaningless.

For solving such problem, a better technique, called long polling, is shown in Fig. 8. Long polling is an AJAX request in which the request and the response are both treated as asynchronous operations. Instead of accepting the request and immediately processing to return a result, the server accepts the request and holds it until some other event occurs, providing data to send as a response to the original request.

Taking the web-based chatroom for example as well, clients would open a persistent connection waiting for the

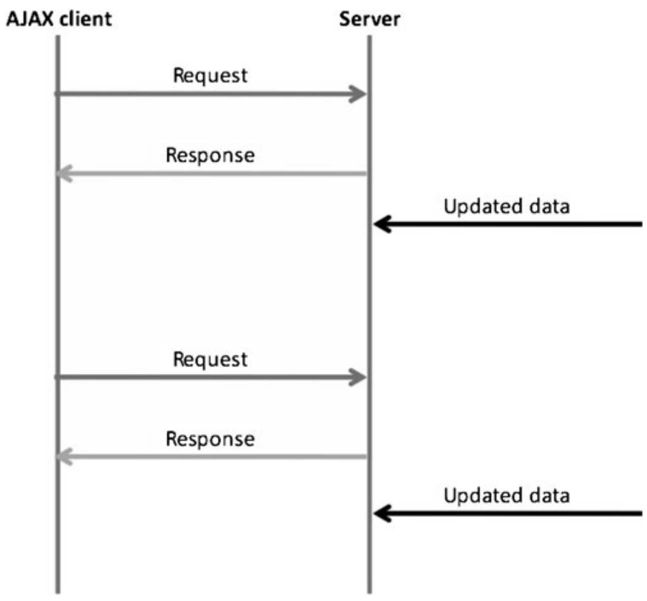


Fig. 7 AJAX with polling

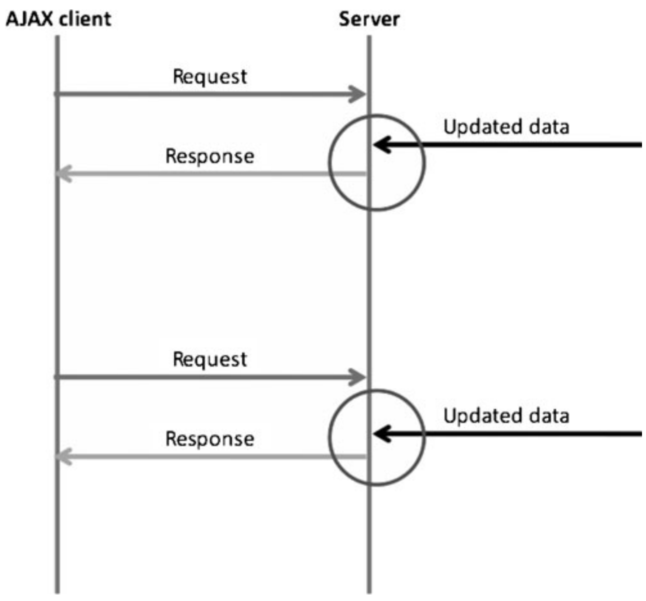


Fig. 8 AJAX with long polling

server to push data till it is available, regardless whether the browser sends out the request or not.

Viewer component

In terms of the implementation of the viewer component, the technology of Extensible Stylesheet Language Transformations (XSLT), which is an XML-based language used for the transformation of XML documents, is applied. It will transform the XML documents into some other different structured documents such as HTML or XHTML that the browsers can support. Note that some JavaScript code shall be inserted into the resulting documents during the transformation process, which lets the browser be able to interact with the loaded CDA documents and enable to verify and redact the selected parts of the original documents.

As shown in Fig. 9, the use of XSLT will change a XML-based CDA document into a HTML web page displayed in the browser. On the contrary, a CDA document without transformation is presented as Fig. 10.

Local signature daemon component

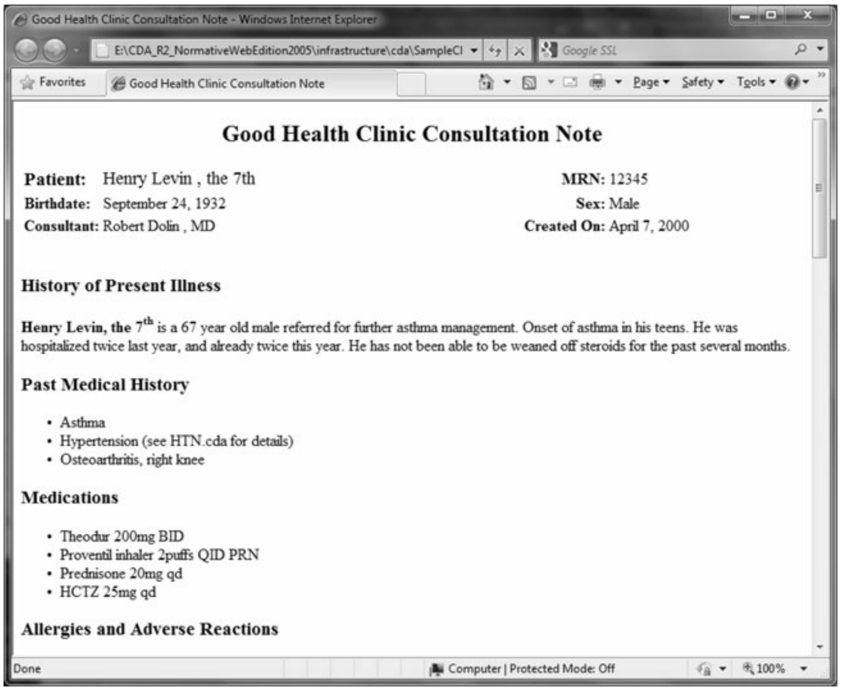
One applied format and principle should be introduced first in this section. JSON (JavaScript Object Notation), a lightweight data-interchange format, is easy for humans to read and write as well as easy for machines to parse and generate. It is based on a subset of the JavaScript programming language, standard ECMA-262 3rd edition. JSON is a text format, a completely independent language, but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON a good data-interchange format [1]. When users apply JSON during AJAX requests, the transmission costs are reduced. In addtion, JSON has good interoperability with other structured formats, such as XML that can be used in the redaction process.

Principle of least privilege, defined by the Department of Defense Trusted Computer System Evaluation Criteria, is a declaration that requires each subject in a system to be granted the most restrictive set of privileges (or lowest clearance) needed for the performance of authorized tasks. The application of this principle can limit the damage which results from accident, error, or unauthorized use [24]. The least privilege approach provides the following benefits. In addition to the reduced risk from attack by malicious software, these benefits include:

1. Increased security.
2. Increased manageability.
3. Increased productivity.
4. Reduced costs.
5. Reduced piracy and legal liability issues.

Clearly, when a program follows the principle of least privilege, it is limited in terms of changing a system. The stability and security of the system should be increased so as to obtain various benefits mentioned above.

The suitable conventional program approach is applied to implement the component of local signature daemon. Although the approach benefits full control, the security


Good Health Clinic Consultation Note - Windows Internet Explorer
E:\CDA_R2_NormativeWebEdition2005\infrastructure\cda\SampleCI
Google SSL
Favorites
Good Health Clinic Consultation Note
Page
Safety
Tools
Done
Computer | Protected Mode: Off
100%

Good Health Clinic Consultation Note

Patient: Henry Levin , the 7th — **MRN:** 12345
Birthdate: September 24, 1932 — **Sex:** Male
Consultant: Robert Dolin , MD — **Created On:** April 7, 2000

History of Present Illness

Henry Levin, the 7th is a 67 year old male referred for further asthma management. Onset of asthma in his teens. He was hospitalized twice last year, and already twice this year. He has not been able to be weaned off steroids for the past several months.

Past Medical History

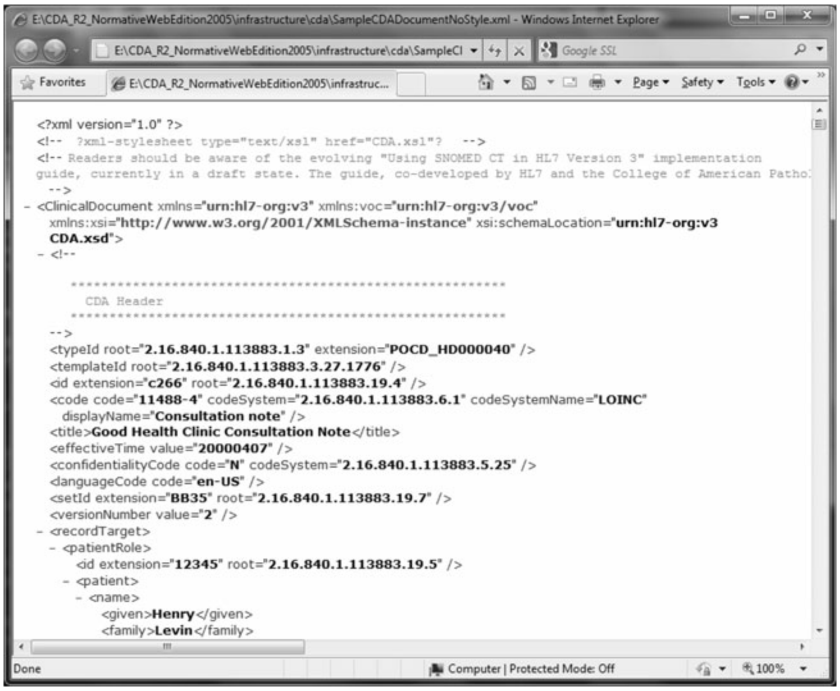
- Asthma
- Hypertension (see HTN.cda for details)
- Osteoarthritis, right knee

Medications

- Theodur 200mg BID
- Proventil inhaler 2puffs QID PRN
- Prednisone 20mg qd
- HCTZ 25mg qd

Allergies and Adverse Reactions

Fig. 9 A transformed CDA document


E:\CDA_R2_NormativeWebEdition2005\infrastructure\cda\SampleCDADocumentNoStyle.xml - Windows Internet Explorer
E:\CDA_R2_NormativeWebEdition2005\infrastructure\cda\SampleCl
Google SSL
Favorites
E:\CDA_R2_NormativeWebEdition2005\infrastruc...
Page
Safety
Tools
Done
Computer | Protected Mode: Off
100%

```
<?xml version="1.0" ?>
<!--  ?xml-stylesheet type="text/xsl" href="CDA.xsl"?   -->
<!-- Readers should be aware of the evolving "Using SNOMED CT in HL7 Version 3" implementation
guide, currently in a draft state. The guide, co-developed by HL7 and the College of American Patho
  -->
- <ClinicalDocument xmlns="urn:hl7-org:v3" xmlns:voc="urn:hl7-org:v3/voc"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="urn:hl7-org:v3
    CDA.xsd">
  - <!--

      **********************************************************
        CDA Header
      **********************************************************
    -->
    <typeId root="2.16.840.1.113883.1.3" extension="POCD_HD000040" />
    <templateId root="2.16.840.1.113883.3.27.1776" />
    <id extension="c266" root="2.16.840.1.113883.19.4" />
    <code code="11488-4" codeSystem="2.16.840.1.113883.6.1" codeSystemName="LOINC"
      displayName="Consultation note" />
    <title>Good Health Clinic Consultation Note</title>
    <effectiveTime value="20000407" />
    <confidentialityCode code="N" codeSystem="2.16.840.1.113883.5.25" />
    <languageCode code="en-US" />
    <setId extension="BB35" root="2.16.840.1.113883.19.7" />
    <versionNumber value="2" />
  - <recordTarget>
    - <patientRole>
        <id extension="12345" root="2.16.840.1.113883.19.5" />
      - <patient>
        - <name>
            <given>Henry</given>
            <family>Levin</family>
```

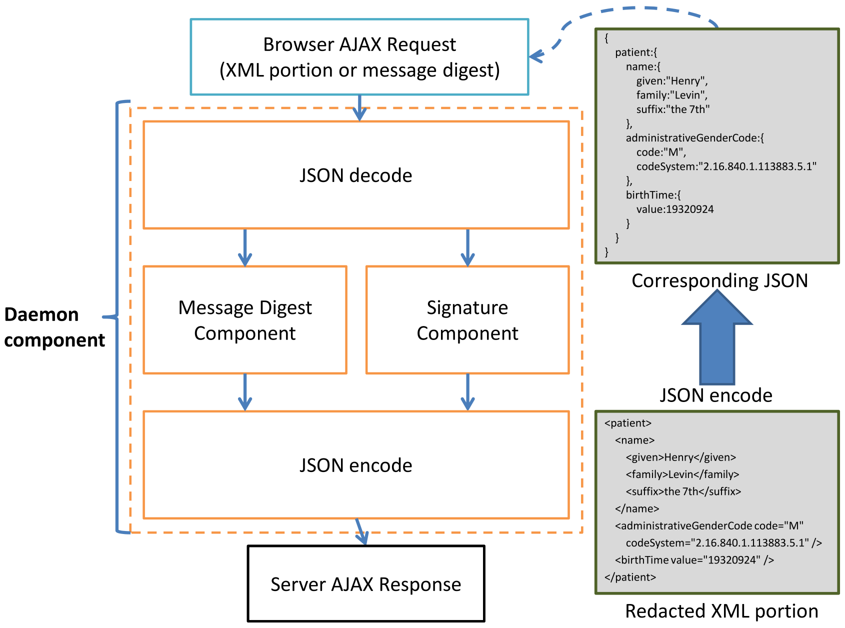
Fig. 10 A CDA document without transformation

concerns need to be dealt with properly. Hence the implementatioin shall follow the principle of least privilege that rises the stability and security of the component. In addition, since the local signature daemon acts as a web server, its address is bound to 127.0.0.1, a local loopback interface so that threats from the Internet may be completely avoided.

Note that in the UNIX-like systems, it requires the root privilege to bind port numbers if they are smaller than 1024. To avoid the similar problem in the implementation, the port number of the daemon is greater than 1024 and made no conflicts in the existing daemons. Figure 11 demonstrates the basic concept of the porposed daemon. The workflow is introduced in Section "Workflow".

Workflow

This section describes how the proposed approach works. The user interface of the proposed viewer component is shown in Fig. 12. When the user executes the redaction in a clinical document, a redacted document is presented as Fig. 13. Figure 14 shows a portion of the redacted CDA



```
{
  patient:{
    name:{
      given:"Henry",
      family:"Levin",
      suffix:"the 7th"
    },
    administrativeGenderCode:{
      code:"M",
      codeSystem:"2.16.840.1.113883.5.1"
    },
    birthTime:{
      value:19320924
    }
  }
}
```

```
<patient>
  <name>
    <given>Henry</given>
    <family>Levin</family>
    <suffix>the 7th</suffix>
  </name>
  <administrativeGenderCode code="M"
    codeSystem="2.16.840.1.113883.5.1" />
  <birthTime value="19320924" />
</patient>
```

Fig. 11 Local signature daemon component of proposed CDA viewer

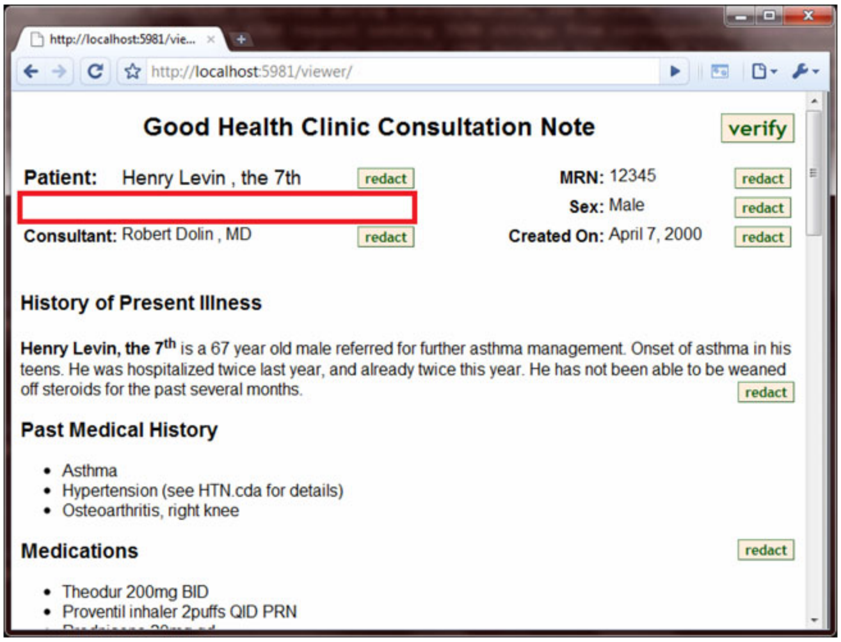
document without transformation. The following is the flow of the whole process.

[The redaction process]

Step 1: The user receives a CDA document signed by the HIS, as Fig. 10 shows.
Step 2: The user puts the received CDA file to the specified location enabling the local signature daemon to have access rights for it.
Step 3: The user opens the browser and navigates to http://127.0.0.1:port/viewer/.
Step 4: The user can view the well-transformed clinical document on the browser, as Fig. 9 shows.
Step 5: Redaction is triggered when the user clicks on the button labeled *redact*.
Step 6: The JavaScript function is inserted during the transformation handling the button event. It shall issue an AJAX request and send the encoding JSON strings corresponding to the portion of the original CDA document (in XML form) to the local signature daemon, as Fig. 11 shows.
Step 7: The request is decoded to strings of XML form, and a message digest is then calculated by the local signature daemon. Finally, an AJAX response, including the message digest, is sent back to the browser (note that the long-polling technology is used that once the digest calculation is done, the viewer will get the result immediately). The flowchart is also shown in Fig. 11.
Step 8: The corresponding message digest is then exchanged with the part the user desires to redact, as Fig. 14 shows.
Step 9: End.

[The verification process]

Step 1: The verifier receives a CDA document that is redacted by the other user.
Step 2: The verifier puts the received CDA file to the specified location enabling the local signature daemon to have access rights for it.
Step 3: The verifier opens the browser and navigates to http://127.0.0.1:port/viewer/.
Step 4: The verifier can view the well-transformed clinical document on the browser, as shown in Fig. 13. Note that the corresponding parts of redaction done by other user would be masked.
Step 5: Verification is triggered when verifier clicks on the button labeled *verify*.
Step 6: The JavaScript function is inserted during transformation handling the button event. It shall issue an AJAX request and send the encoding JSON strings corresponding to message digests of the redacted parts of CDA document to the local signature daemon.
Step 7: The request is decoded to the message digests. These message digests will combine with other parts message digests hashed from non-redacted parts of the original CDA document to form a completed message digest for verification by the local signature daemon. Then, the daemon can verify the validity of



Good Health Clinic Consultation Note verify

Patient: Henry Levin , the 7th redact
MRN: 12345 redact
Sex: Male redact
Consultant: Robert Dolin , MD redact
Created On: April 7, 2000 redact

History of Present Illness

Henry Levin, the 7th is a 67 year old male referred for further asthma management. Onset of asthma in his teens. He was hospitalized twice last year, and already twice this year. He has not been able to be weaned off steroids for the past several months. redact

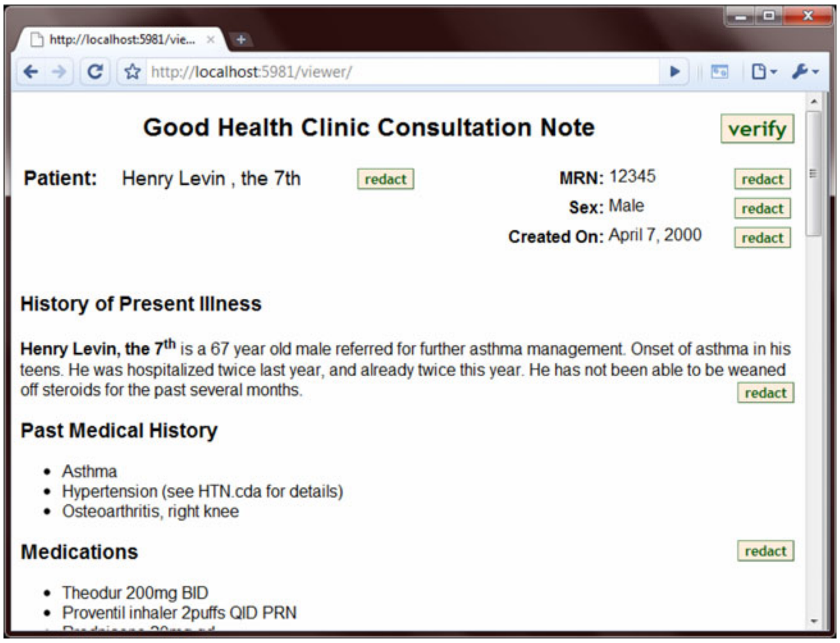
Past Medical History

- Asthma
- Hypertension (see HTN.cda for details)
- Osteoarthritis, right knee

Medications redact

- Theodur 200mg BID
- Proventil inhaler 2puffs QID PRN

Fig. 12 The User Interface of proposed CDA viewer



http://localhost:5981/vie...

http://localhost:5981/viewer/

Good Health Clinic Consultation Note verify

Patient: Henry Levin , the 7th redact

MRN: 12345 redact

Sex: Male redact

Created On: April 7, 2000 redact

History of Present Illness

Henry Levin, the 7th is a 67 year old male referred for further asthma management. Onset of asthma in his teens. He was hospitalized twice last year, and already twice this year. He has not been able to be weaned off steroids for the past several months. redact

Past Medical History

- Asthma
- Hypertension (see HTN.cda for details)
- Osteoarthritis, right knee

Medications redact

- Theodur 200mg BID
- Proventil inhaler 2puffs QID PRN

Fig. 13 A redacted CDA document

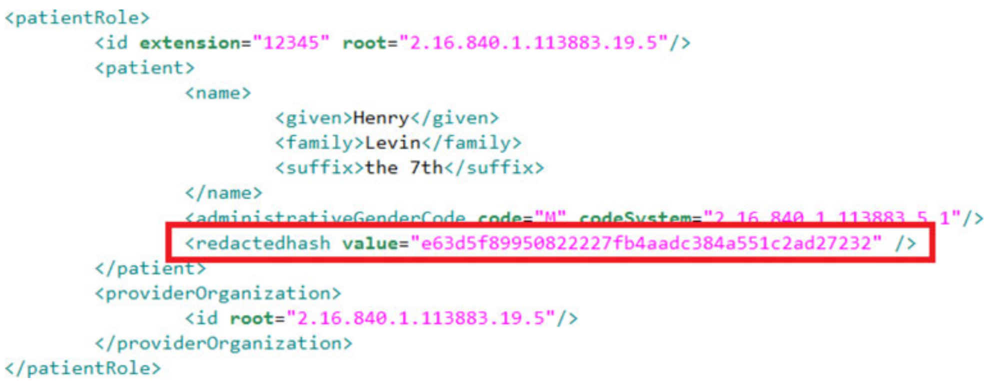
the signature (the detailed process is shown in Fig. 4). Finally, an AJAX response, containing the verification result, is sent back to the browser (note that the long-polling technology is used that once the signature verification is done, the viewer will get the result immediately).

Step 8: The validity of signature is verified.

Step 9: End.

Below shows an example to explain the whole processes of the redactable signature scheme. First of all, if some patient wants to conceal some private information appeared in the signed XML-based CDA document, he would execute the redaction process. He can easily open the document by web browser since the viewer component of the scheme has converted it into html format as Fig. 12 shows. Supposing the patient would not like other people to know who his consultant is, he can click the *redact* button to eliminate the information about the consultant. The component of local signature daemon will automatically find the specific hash value and replace the consultant's name with it to complete the redacting action. The result is shown in Figs. 13 and 14, and all information about the consultant is gone and replaced by the specific value that common users are hard to understand.

When a verifier receives the redactable document, he can execute the verification process to confirm the correctness of the document by using the original signature, i.e. knowing the document indeed comes from the signer. He opens the document yet does not view the information of the redactable parts, as Fig. 13 shows**.** He clicks the *verify* button, and then the component of local signature daemon automatically verifies the document. The component will combine all message digests including redacted and non-redacted parts of the CDA document to form a completed message digest for verification. Then, the daemon can check the combined digest whether it is equal to the original digest decrypted by using the signer's public key. If



```
<patientRole>
        <id extension="12345" root="2.16.840.1.113883.19.5"/>
        <patient>
                <name>
                        <given>Henry</given>
                        <family>Levin</family>
                        <suffix>the 7th</suffix>
                </name>
                <administrativeGenderCode code="M" codeSystem="2.16.840.1.113883.5.1"/>
                <redactedhash value="e63d5f89950822227fb4aadc384a551c2ad27232" />
        </patient>
        <providerOrganization>
                <id root="2.16.840.1.113883.19.5"/>
        </providerOrganization>
</patientRole>
```

Fig. 14 A portion of the redacted CDA document without transformation

Fig. 15 Partial codes about implementing redactable signatures

```
def getCDAHash(filename):
        from lxml import etree
        import hashlib

        parser = etree.XMLParser(remove_comments=True)
        tree = etree.parse(open(filename, 'rb'), parser)
        cdaxml = tree.getroot()
        sha1 = hashlib.sha1()
        list = []

        for elements in cdaxml:
                for element in elements.iter():
                        elementtext = ''.join([element.tag, str(element.a
ttrib), (element.text or '').strip()])

                        sha1.update(elementtext)
                        list.append(sha1.hexdigest())

        sha1.update(''.join(list))

        return sha1.hexdigest()

signHash(getCDAHash('cda.xml'))
```

both of them are the same, the validity of the signature and document are guaranteed.

Analysis of proposed approach

In this section, the proposed approach is evaluated through three aspects, including the portability and flexibility of the program and the complexity to implement the signature and its demanding CDA viewer. The higher portability of program the approach gains, the more platforms the web browser can be available. The development costs are thus decreased. The same situation, the higher flexibility of program the approach owns, the more control rights can be kept by the designers. The bigger programming problems would be relatively reduced. Final, the lower complexity, i.e. the difficulty of implementation, the approach guarantees, the lower time and money cost the developers need to pay out.

Portability

The proposed CDA viewer has two significant components as viewer and local signature daemon. The portability of viewer and local signature daemon would be discussed respectively. In terms of viewer, the property of portability should not be the issue since there are various web browsers available on different platforms. As to the implementation of the local signature daemon, the designed programming language chooses Python which can be available on every platform and maintain the acceptable performance, this will add the daemon's productivity and lower its maintenance costs. Therefore, the portability is achieved in the CDA viewer.

Flexibility

When designers implement the redactable signature by the conventional program approach, their program will have high flexibility as they have full control of the environment during running the program. But in terms of a browser-based approach, its flexibility is usually limited by the corresponding web browser or browser plug-ins such as Adobe Flash Player.

To make this approach possess the properties of portability and flexibility, the local signature daemon component of the proposed viewer is implemented based

Table 1 Comparison table with other implementing approaches

	Portability	Flexibility	Complexity
Our approach	High	High	Low
Web Application	High	Medium	High
Conventional Program	High variance	High	High variance
Adobe Flash	Medium	Medium	High variance
JavaScript	High	Medium	High

on both the conventional program approach and the browser-based approach. Therefore, the approach can do anything that a conventional program could do. Furthermore, performance-intensive functionalities can also be implemented into the daemon, which then is executed through AJAX calls.

Difficulty of implementation

The whole architecture of CDA viewer contains the viewer component and the local signature daemon component (which deals with business logic). The viewer component is basically a defined XSLT, and all browsers would deal with the transformed XHTML or HTML file well. Without considering the fancy user interface, the difficulty of the implementation for the viewer component is eliminated. As to the local signature daemon component, it only needs to deal with signing and message digesting. The programming difficulty thus depends on the chosen programming language, i.e. Python, an excellent programming language available on every platform and able to maintain the acceptable performance. Figure 15 shows the partial codes about implementing the proposed redactable signature by Python programming language. The main function of these codes will combine with each hash function responded to the redactable part of the CDA document to form a redactable signature.

Summing up the above descriptions, a comparison table with other implementing approaches was drawn, as shown in Table 1. C, C#, C++, Java, and etc. are the conventional program approach. It has high flexibility since designers have full control of the environment when running the program. However, other properties, such as portability and complexity are high variance. A high efficiency might be presented; but unfavorable results could possibly appear. On another side, the web application, Adobe Flash, or JavaScript are the browser-based approach. This kind of approaches combines with the use of browser, and is available for various platforms so as to have high portability. But their flexibility and complexity outperform other approaches. Ours, on the contrary, with the analysis of the three concerns mentioned above shows the implementation being practicable.

Conclusions and future work

In this paper, a compact and applicable model for implementing a redactable signature-capable CDA document viewer by using the existing web-technologies was proposed. In this model, the possible overhead of the viewer caused by the conventional program approach is eliminated via dividing the whole architecture into two parts; the viewer component deals with the presentation, whereas the daemon component processes the business logic. In addition, the smoothness of the user interaction depends on the high speed of local signature daemon component, and the delay problems between AJAX requests are solved by the technique of long polling.

This CDA viewer is also compatible with the applications of the portable CDA for the secure clinical-document exchange that any XML-based applications may have the needs of redactable signatures.

In this approach, the redaction process basically replaces hashes with the corresponding redacted parts, that if a user tends to redact much information, the size of resulting document could become very large. In the future, the information may be redacted based on their common parent so as to make less hashes be generated. Therefore, the proposed methodology can operate more efficiently.

Acknowledgement This work was supported partially by National Science Council, Taiwan under Grants NSC 99-2221-E-029-023.

References

1. The Health Level Seven International home page. Available at http://www.hl7.org/.
2. Eastlake, D., Solo, D., and Reagle, J., XML-signature syntax and processing. first edition of a recommendation, W3C, 2002. Available at http://www.w3.org/TR/2002/REC-xmldsig-core-20020212/.
3. Tsumoto, S., Mining diagnostic rules from clinical databases using routh sets and medical diagnostic model. *Inf Sci* 162(2):65–80, 2004.
4. Hsu, C.-C., and Ho, C.-S., A new hybrid case-based architecture for medical diagnosis. *Inf Sci* 166(1–4):231–247, 2004.
5. Ulieru, M., Hadzic, M., and Chang, E., Soft computing agents for e-Health in application to the research and control of unknown diseases. *Inf Sci* 176(9):1190–1214, 2006.
6. Miyazaki, K., Susaki, S., Iwamura, M., Matsumoto, T., Sasaki, R., and Yoshiura, H., Digital documents sanitizing problem. *IEICE Technical Report ISEC2003-20*, 2003.
7. Miyazaki, K., Iwamura, M., Matsumoto, T., Sasaki, R., Yoshiura, H., Tezuka, S., and Imai, H., Digitally signed document sanitizing scheme with disclosure condition control. *IEICE Trans* 88 (1):239–246, 2005.
8. Slamanig, D., and Stingl, C., Disclosing verifiable partial information of signed cda documents using generalized redactable signatures. In: *e-Health Networking, Applications and Services, 2009. Healthcom 2009*, pp. 146–152, 2009.
9. National Institute of Standards and Technology, "Digital signature standard," 1994.
10. Rivest, R. L., Shamir, A., and Adleman, L., A method for obtaining digital signatures and public-key cryptosystems. *Commun ACM* 21(2):120–126, 1978.
11. Johnson, R., Molnar, D., Song, D. X., and Wagner, D., Homomorphic signature schemes. In: *CT-RSA '02: Proceedings of the The Cryptographer's Track at the RSA Conference on Topics in Cryptology*, pp. 244–262, 2002.
12. Rivest, R., Two new signature schemes, *Presented at Cambridge seminar*, 2001. Available at http://www.cl.cam.ac.uk/Research/Security/seminars/2000/rivest-tss.pdf.

13. Micali, S., and Rivest, R., Transitive signature schemes. In: *CT-RSA '02: Proceedings of the The Cryptographer's Track at the RSA Conference on Topics in Cryptology*, pp. 236–243, 2002.
14. Bellare, M., Goldreich, O., and Goldwasser, S., Incremental cryptography: The case of hashing and signing. In: *Proceedings of advances in cryptology—Crypto 1994*, Vol. 839 of LNCS, pp. 216-233, Springer-Verlag, 1994.
15. Bellare, M., Goldreich, O., and Goldwasser, S., Incremental cryptography and application to virus protection. In: *proceedings of the 27th ACM Symposium on the Theory of Computing*, pp. 45–56, 1995.
16. Steinfeld, R., Bull, L., and Zheng, Y., Content extraction signatures. In *International Conference on Information Security and Cryptology 2001*, Vol. 2288 of LNCS, pp. 163–205, Springer-Verlag, 2001.
17. Ateniese, G., Chou, D. H., de Medeiros, B., and Tsudik, G., Sanitizable Signatures. In *10th European Symposium on Research in Computer Security—ESORICS 2005*, Vol. 3679 of LNCS, pp. 159–177, Springer-Verlag, 2005.
18. Chang, E. C., Lim, C. L., and Xu, J., Short redactable signatures using random trees. In: *CT-RSA '09: Proceedings of the The Cryptographer's Track at the RSA Conference on Topics in Cryptology*, Vol. 5473 of LNCS, pp. 133–147, Springer-Verlag, 2009.
19. Haber, S., Hatano, Y., Honda, Y., Horne, W., Miyazaki, K., Sander, T., Tezoku, S., and Yao, D., Efficient signature schemes supporting redaction, pseudonymization, and data deidentification. In: *Proceedings of the 2008 ACM Symposium on Information, Computer and Communications Security, ASIACCS 2008*, pp. 353–362, 2008.
20. Dolin, R., Alschuler, L., Boyer, S., Beebe, C., Behlen, F., Biron, P., and Shabo, A., HL7 clinical document architecture, release 2. *J Am Med Inform Assoc* 13(1):30–39, 2006.
21. Huang, K.-H., Hsieh, S.-H., Chang, Y.-J., Lai, F., Hsieh, S.-L., and Lee, H.-H., Application of portable CDA for secure clinical-document exchange. *J Med Syst* 34(4):531–539, 2010.
22. Chang, Y., Lai, J., Cheng, P., and Lai, F., Portable cda for the exchange of clinical documents. In: *e-Health Networking, Application and Services, 2007 9th International Conference*, pp. 1–5, 2007.
23. Haomin, L., Huilong, D., Xudong, L., and Zhengxing, H., A clinical document repository for cda documents. In: *Bioinformatics and Biomedical Engineering, 2007. ICBBE 2007,* pp. 1084–1087, 2007.
24. M. Treins, O. Cure, and G. Salzano, "On the interest of using HL7 cda release 2 for the exchange of annotated medical documents," In *Computer-Based Medical Systems, 2006. CBMS 2006. 19th IEEE International Symposium on*, pp. 524–532, 2006.
25. Diffie, W., and Hellman, M., New directions in cryptology. *IEEE Trans Inf Theory* 22(6):644–654, 1976.
26. Stallings, W., Cryptography and network security: principal and practices. Prentice Hall, 4th Edition, 2005.
27. ElGamal, T., A public key cryptosystem and signature scheme based on discrete logarithms. *IEEE Trans Inf* 31(4):469–472, 1985.
28. Berners-Lee, T., Fielding, R., and Masinter, L., Uniform Resource Identifier (URI): Generic Syntax, RFC 3986 (Standard), 2005.
29. Yee, B., Sehr, D., Dardyk, G., Chen, J., Muth, R., Ormandy, T., Okasaka, S., Narula, N., and Fullagar, N., Native client: A sandbox for portable, untrusted x86 native code. In: *Security and Privacy, 2009 30th IEEE Symposium* on, 2009.
30. Kay, M., Chamberlin, D., Robie, J., Fernandez, M. F., Simeon, J., Boag, S., and Berglund, A., XML path language (XPath) 2.0. W3C recommendation, W3C, Jan. 2007. Available at http://www.w3.org/TR/2007/REC-xpath20-20070123/.
31. The JSON format home page. Available at http://www.json.org/.
32. Microsoft Developer Network (MSDN), "Applying the principle of least privilege to user accounts on windows xp," 2006.