

Secure Document Circulation: An Architecture for e-Health

Shane Bracher, Padmanabhan Krishnan Centre for Software Assurance, Bond University, Gold Coast, Australia

Abstract

We present an architecture for the secure circulation of electronic medical records. The architecture considers two issues prevalent in e-health - inter-operability and security and privacy - and is designed for inter-organisational information flow. We focus our attention on the protection of patient privacy and discuss how privacy policies are applied and enforced on medical records. We also consider privacy protection based on trust relationships formed by the patient. A key feature of the architecture is that privacy policies are not assumed to be complete. For cases where policies do not contain sufficient information to make a privacy-related decision, we show how a simple reasoning scheme can be used based on the "need to know" principle.

Keywords: Electronic medical records, privacy, policies, workflow

1. Introduction

Information technologies are revolutionising the medical profession by increasing the efficiency of health care provision and reducing costs [1]. Notable technologies attracting widespread attention are portable medical monitoring devices for the home and electronic medical records [2]. But as "e-health" becomes more pervasive within the medical industry, one prediction is that health care technologies will shift focus "from a central, hospital-based system to a patientcentred system, where patients will be the manager and owner of their health information" [2].

Despite the current inroads being made into e-health, serious problems remain. Two major issues include *inter-operability* and *security and privacy*. The lack of mandated or cooperation between inter-operability standards in the industry are hampering efforts to improve the provision of health care. For e-health to advance, it is imperative that inter-operability standards exist; for example, to allow monitoring devices to communicate with one another, to allow electronic medical records to be compatible with the information systems in all medical institutions, etc. Inter-operability itself raises security and privacy concerns. For example, it is unclear how the confidentiality and privacy of medical information could be safeguarded in the situation where electronic records are circulated between medical professionals affiliated with different organisations.

Related to data circulation are workflows. Workflows are a standard way to model and manage coordination and interaction between businesses and customers. Workflow Management Systems (WfMS) control the execution of workflows. WfMS operate by interpreting process definitions, interacting with the workflow participants and invoking external software applications. Although these systems once focused on automating isolated office tasks, nowadays, there is a shift towards WfMS managing inter-organisational information flows (to achieve automation of the entire business process) [3]. For workflows which spread across multiple organisations, *web services* have emerged as a promising implementation option considering their suitability for heterogeneous and Internet-based environments.

Although there are many specific languages to describe workflows [4], some standards, especially in the context of web services such as WS-BPEL (originally called BPEL4WS) [5], have emerged. When such technology is applied to medical systems, the availability of standards such as Health Level 7 (HL7) and DICOM have not fully helped in overcoming the complexity associated with medical workflows [6]. This is particularly true in developed countries where there are two logical components, viz., administrative aspects and actual medical services.

The situation is not clear even if one focuses only on the security aspects of workflows in medical systems. In general, the most widely recognised web services security technology for providing access con-

The *electronic Journal of Health Informatics* is an international journal committed to scholarly excellence and dedicated to the advancement of Health Informatics and information technology in healthcare. ISSN: 1446-4381

[©] Copyright of articles is retained by authors; originally published in the *electronic Journal of Health Informatics* (http://www.ejhi.net). This work is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 2.5 License (http://creativecommons.org/licenses/by-nc-sa/2.5/au).

trol is eXtensible Access Control Markup Language (XACML) [7]. The use of role based access control (RBAC) for medical systems has been explored [1] to model issues such as trust and privacy. They show how multiple roles arranged in a hierarchy can be used to represent trust and jurisdiction. While RBAC can be used to implement privacy requirements by controlling access to data, the privacy guaranteed by a particular RBAC set of policies is not always clear. Agrawal et al. [8] discuss privacy issues at a higher level of abstraction (i.e., independent of implementation language) and a proposed architecture to implement it. Hooda et al. [9] describe the security of a records system which is HL7 compliant. But as noted [6] security features are often missed when authorisation is used as an enforcement mechanism for privacy. It is possible to use temporal logic [10] to describe privacy related security constraints directly (such as separation of duty) - but such specifications are hard to use in practice (from both the health practitioners' and the software engineers' perspectives). Modelling and verification tools can be used instead to apply the logic indirectly.

In this paper, we propose an architecture for the circulation of electronic medical records amongst medical professionals, with a focus on protecting patients' privacy. We also show how a realistic version of the "need to know" principle can be implemented. The architecture is designed to support inter-organisational information flow, and for that reason, it considers inter-operability issues as well as security and privacy issues. As in any inter-organisational information flow scenario involving document circulation, a major challenge faced is enforcing access control when the document is outside the security realm of the data owner. This challenge exists as inter-organisational workflows assume no centralised global authority to enforce security. Instead, each security realm which the document flows through is separately administrated by its respective organisation.

In the proposed architecture, privacy conditions are governed by policies. We discuss two types of policies for protecting patients' privacy. Global privacy policies list the privacy rules that apply to all medical records. Patient privacy policies are unique for each patient and list the patient's own preferences in terms of privacy.

Technically it is possible to implement privacy using RBAC, Chinese wall policies etc. We use the RBAC model, together with encryption, to enforce privacy conditions. In any case, privacy is the main policy driver of our research - RBAC is simply one of many possible implementation options that can be used.

Problems arise if the privacy conditions do not adequately describe the situation under which the medical record is being used. For example, the policy rules may not be explicit enough to provide an accurate decision. Policies across organisations may be such that they do not allow access to data. For instance, we do not assume that the definition of roles and access control are defined in a uniform fashion. It is possible for particular RBAC definitions to prevent the completion of a particular task.

To address such situations, we apply an extension to the architecture that allows users to provide a reasoning to justify their usage of the medical record. If their reasoning is valid, permission is granted for the user to perform their requested action in accordance with the privacy conditions.

Additionally, the patient may specify in their privacy policy the trust relationships that it shares with various medical practitioners. If the requesting user is trusted by the patient, then permission can be granted based on the conditions of the trust relationship.

Issues such as privacy can be associated with both the flow of information as well as the tasks performed. For instance, the requirement that a patient's medical record will not be sent to the hospital administrator is related to the flow of information. On the other hand, a doctor treating a patient may need to know some details to complete the task on hand.

The remainder of this paper is organised as follows: Section 2 provides an abstract overview of the architecture; Section 3 describes how we apply the architecture to e-health and the technology used in its implementation; Section 4 reports on the case study; and finally, Section 5 concludes the paper.

2. Architecture Overview

2.1. Document Model

The formation of the circulation document used in the proposed architecture is shown in Figure 1. At the core of this model is the document data and the document metadata. The document data contains data values, and these data values are assigned by anyone editing the document. The document data is conceptually considered as a graph where data values (or groups of data values) are organised into nodes. This approach is inspired by the Document Object Model (DOM) [11] but we specialise it for the purpose of secure document circulation in the context of privacy and health records.

Meta information is assigned to individual nodes. This information is contained in the document metadata and includes such details as: structure restrictions (applied to the data organisation structure within the node); content restrictions (the valid set of data values for the node); data security (confidentiality and data integrity protection); data ownership (the identity of the owner of the node); and attributes (other miscellaneous attributes of the node).

The integration of data security into the model serves to provide finer granularity support. Typically, a document is secured by encrypting it as a whole. With finer granularity support, individual data values within the document can be separately secured without affecting the rest of the document. For this model, this level of granularity is necessary as data ownership is assigned to nodes rather than to the entire document itself. Additionally, it is necessary for providing decentralised access control. The need for decentralised access control exists as the model assumes that there is no single authorisation authority. To deal with this, encryption must be used to render protected nodes unreadable to unauthorised users. For authorised users, decryption keys are required to access protected nodes. The decryption key determines the node that the user is permitted to access, whereas access to decryption keys is determined by the access control rules.

The document data is governed by policies. The types of policies supported in the model are as follows:

- Global Policy specifies the flow control rules for the document as well as global access control conditions. Each document instance has a single global policy. e.g. "The document can only be forwarded to users of role x after it has been received by at least one user of role y."
- Data Owner Policy specifies the access control conditions applied to a node in the document data. This policy is defined by a data owner and it applies to all nodes owned by that particular data owner. Each data owner associated with the document data has their own data owner policy. e.g. *"Users of role x have 'read only' permission for node n."*



Figure 1: Document model for secure document circulation architecture.

- Local Validation Policy specifies the pre-conditions and post-conditions of a document for it to become accepted by a document recipient. Each document recipient defines their own validation policy. e.g. *"The document can only be received from a user of role x."*
- Trust Policy specifies the trust relationships shared between entities (e.g. document recipients).
 When policy rules (of the above policies) are trust-based, this policy provides the trust predicates that are used to evaluate the policy rules.

The document data is intended to be machine interpretable to allow for automated processing. So that the document data can be easily read by users, each document is associated with a template which describes the presentation layout for the document. As part of the document processing, a document view is computed (taking the document data and the template) to make the document human readable. We also assume that the various policies are consistent. While it is possible to analyse the policies for consistency, this is outside the scope of this research.

2.2. Architecture Model

A simplified view of the architecture is given in Figure 2. The document flow begins with the *issuing authority* releasing a new document instance to a *recipient*. The document instance consists of the document data and the document metadata. The document data and the document metadata are initialised by the issuing authority (e.g. the issuing authority decides on the structure of the document data and the initial data values, etc.).

When the recipient performs an action on the document (e.g. read the data value of a node), this action is first verified by a *verifier*. The verifier checks the action against the policies associated with the document, and informs the recipient of the verification result (i.e. either action permitted or action denied). Once a recipient has processed the document, it can continue the document flow by forwarding the document to another recipient.

The issuing authority, recipient and verifier are all types of agents. Each user associated with the document flow interact with the system via an agent. Each agent has a set of behaviours that it is capable of performing on behalf of the user it represents. These behaviours are subject to policy conditions. If a requested action by the user is found to be illegal (i.e. the verifier replies with an action denied response), the agent refuses to execute the action. In other words, agents attempt to ensure that only legitimate actions are performed on the circulating document.



Figure 2: Secure document circulation architecture.

Agents alone cannot prevent illegal actions from occurring. For example, the user could bypass the agent and directly perform an illegal action on the document data. Therefore, although it is not possible to prevent illegal actions, the architecture ensures that such actions are detectable to others via verification and document validation. This is ensured by *always* invoking the verifier before every operation.

3. Secure Document Circulation for e-Health

We achieve secure document circulation for e-health by adapting the previously discussed architecture to support the circulation of electronic medical records amongst medical professionals. We take a patient-centric approach by declaring the patient as the owner of all data in their medical record. The idea is for the medical record to be easily transportable between different medical professionals, health institutions, etc. to provide better integration in the medical industry.

In this paper, we focus only on protecting patients' privacy (despite the architecture's applicability for a wide class of properties extending beyond security and privacy). We specialise the architecture to include a patient privacy policy. This allows the patient to declare their own privacy rules based on the disclosure of their personal health data. The global policy in the architecture also expresses privacy rules. When an action is verified, if none of the patient's privacy rules are applicable, the global privacy rules are used.

The agent types introduced in section 2.2 are represented by the following entities in the e-health architecture:

- Issuing Authority: A recognised, medical governing authority (possibly affiliated with the government).
- Recipient: The patient and any medical professional involved in treating the patient.
- Verifier: A trusted third party.

The document data for the medical record could include the following sections: patient specific data (e.g. name, address, insurance number, emergency contact), patient's medical history, patient's medication history, etc. There are a number of component types that could also be included in the document data (e.g. prescriptions, referrals, specialised services such as X-rays, ultra-sounds, etc.). Each data item may also have a start time (or creation time) and an optional expiry time. This can follow the HL7 specifications.



Figure 3: Action verification process for electronic medical records.

The action verification process for the medical record is illustrated in Figure 3. The recipient performing the action starts by instructing its agent of the acton to perform. In this case, the recipient is a General Practitioner. The agent follows a two-step procedure to process the action. The first step is to verify the action. The agent sends an action request to the verifier. The verifier verifies the action request against the patient's privacy policy and the global policy. In the situation where the medical record has multiple data owners, the data owner's policy is also used. Although multiple data owners are supported in the architecture, we concentrate on the patient as the only data owner. For the purposes of this paper, the patient's privacy policy is the same as the data owner's policy.

After the verifier verifies the action, the recipient's agent is notified of the verification result. If the action has been denied, the agent notifies the recipient. The medical record remains unchanged. If the action has been cleared, the agent performs the action and the medical record can change state.

3.1. Privacy Enforcement Framework

The RBAC model can be used to enforce the privacy rules expressed in the policies. Typically, access control is enforced within a single security realm. One of the main characteristics of the proposed architecture is that it supports inter-organisational information flow, and for that reason, it cannot assume a single authority for enforcing access control. Subsequently, the architecture uses encryption to implicitly enforce access control. Private data items in the medical record are secured using encryption and access to these data items require possession of the designated decryption key(s).

The use of privacy policies enforced by access control mechanisms is fine when matching policy rules exist (in other words, when the policies provide a clear definition of the circumstances for the action request). If the circumstances for the medical record usage are different to all of the conditions listed in the policies, this creates a problem. Essentially, this issue relates to incomplete information. We cannot assume that the policies will fully capture all possible circumstances where a particular medical record will be used. For this reason, we incorporate a simple reasoning scheme into the architecture based on the "need to know" principle. The need to know approach is difficult to implement effectively as:

- It requires that the behaviour of every agent in the system be fully captured. This is not possible as many agents are human who use their intuition. They also encounter many differing situations that are not formally specified.
- Even after capturing the entire reasoning process, the verification process can be tedious. Even if an automatic technique such as model checking is applicable, one needs to

show that all required behaviours would need the particular information (or in other words, that there is no legal behaviour that does not use this information). Specifying this in general is non-trivial.

Introducing a reasoning scheme into the architecture extends the policy evaluation procedure during the action verification process. The action request must satisfy the following principles:

- If the request satisfies the patient's privacy rules, the data item can be released to the requesting agent.
- If the request satisfies the global privacy rules, the data item can be released to the requesting agent.

• If both of the above situations are invalid, the requesting agent must specify a reason. If this reason satisfies the privacy rules, the data is released to the agent. The reason provided by the agent is then stored in the document metadata for auditing purposes.

The agent can formulate a reason by writing basic expressions that incorporate a suitable fragment of logic. Assertions are used to encode observations made by the agent, and these assertions form the agent's justification for access to the data item. Although the verifier cannot automatically check the accuracy of any assertions made by the requesting agent in its reasoning, the logical consistency can be implicitly verified. The reason is later stored in the document metadata so that any assertions (including observations of the patient's health status, readings of vital signs, etc.) can be manually verified during auditing procedures. We show an example reasoning scenario in the next section as part of our case study.

In the situation where the action request concerns trust-related privacy rules, trust predicates (specified in the policies) are used in the verification process. If the requesting agent is considered trustworthy according to the trust predicates, the trust-related privacy rules are consequently satisfied and the data item can be released to the requesting agent.



Figure 4: Implementation setup for the case study.

3.2. Implementation

The medical record (i.e. the document data and the document metadata) is represented in XML. The benefits of XML are that it is platform independent, machine interpretable, standardised and extensible. These characteristics make XML suitable for automated processing and interorganisational information flow. XML is also compatible with the standardised Document Object Model (DOM) - a platform and language neutral interface for dynamically accessing and modifying the content and structure of XML data. The DOM presents XML data in a tree structure. As the medical record data is conceptually a graph of data items, we can model this data structure in XML.

For protecting private data items, we use XML Encryption and XACML. As its name suggests, XML Encryption provides a standardised way for encrypting XML elements (without restrictions on the choice of encryption algorithm, key length, etc.). XACML provides a policy language and an access control decision request/response language (both are encoded in XML). Access control decisions result in one of four possible values: *Permit, Deny, Indeterminate* (error occurred) or *Not Applicable* (no valid policy rules found). An XACML implementation requires a Policy Enforcement Point (PEP) and a Policy Decision Point (PDP). The PEP sends an XACML request to the PDP for an access control decision on the user's request. The PDP processes the XACML request, using the policies to make an access control decision which is later enforced by the PEP.

The implementation setup for our case study is illustrated in Figure 4. We model the recipient's agent (the requesting agent) as the PEP and the verifier as the PDP. As encryption is used to safeguard private data items, we include a Kev Distribution Centre in the setup (to provide a key retrieval service). A SAML token enclosed in the XACML response authenticates the verifier's access control decision to the Key Distribution Centre. Alternatively, key distribution could be handled by the verifier (in that case, the decryption key(s) could be sent with the XACML response).

For implementing the reasoning scheme, the choice of verification tool depends on the language selected for encoding the reasons. While the exact language has not been finalised, it will follow the structure of the medical record. PVS [12] offers a suitable specification language based on classical, typed higher-order logic. This tool also provides a theorem prover for validating the reasonings. An XML to PVS translator can be constructed to link the document with the verifier. An alternative verification tool that is more closely aligned to XACML is Margrave [13]. The specification language used in Margrave is based on the Scheme programming language.

4. Case Study

We consider a medical emergency situation for our case study. The global privacy policy defines the standard permissions for responding to an emergency situation. Additionally, the patient has its own preferences for responding to an emergency situation (stored in the patient privacy policy). A doctor treating the patient wants to read the patient's medical record. The rules governing the use of the medical record in an emergency situation are as follows:

The global policy for data access is

 \forall (*d*: Doctor, *p*: Patient): isEmergency(*d*, *p*) \supseteq canReadRecord(*d*, *p*)

A particular patient's (say p_1) personal policy is

 $\forall (d: \text{Doctor}, p: \text{Person}, \text{gp: GP}): (\text{isEmergency}(d, p_1) \land \text{emergencyContact}(p_1, p) \land \text{isGp}(\text{gp}, p_1)) \supset (\text{notify}(d, p) \land \text{notify}(d, \text{gp}))$

The global policy rule permits any doctor to read the entire medical record if the patient is in a state of emergency. The patient's policy rule allows any doctor to notify the patient's emergency contact (e.g. the next of kin) and preferred general practitioner of the patient's medical condition if the patient is in a state of emergency.

4.1. Scenarios

We look at two scenarios in this case study:

- 1) The patient is involved in an emergency.
- 2) The patient has a rare condition that results in an emergency.

The first scenario is relatively straight forward as there is a matching policy rule for when the patient is involved in an emergency. The doctor's agent sends an XACML request to the verifier, this request is checked against the global and patient policies, and the doctor is granted permission to read the medical record.

The second scenario is more complex. It is unclear from the policy rules if the patient's condition constitutes an emergency situation. As the condition is rare, no explicit policy covering the condition is found. Because of this, the doctor must provide a reason. Apart from giving the doctor access to the medical record, the reasoning also aims to fill the information gap caused by the missing policy rules.

An example reasoning that the doctor could provide is as follows:

 \forall (*p*: Patient): (hasCondition(d_1, p, c_1) \land hasCondition(d_1, p, c_2)) \supset

 $hasRareCondition(d_1, p)$

 \forall (p: Patient): hasRareCondition(d_1, p) \supseteq isEmergency(d_1, p)

combined with the observations that the patient p_1 has conditions c_1 and c_2 which is formally specified as

hasCondition $(d_1, p_1, c_1) \land$ hasCondition (d_1, p_1, c_2)

The doctor asserts that any patient suffering from both of these conditions implies a rare condition. Subsequently, a rare condition implies an emergency situation. As a policy rule for an emergency situation exists, the verifier uses this rule for making the access control decision. Although the doctor has clarified how the patient's rare condition constitutes an emergency situation, the reasoning does not explicitly explain why the doctor *needs to know* the patient's medical record information.

We modify the reasoning to show how the need to know principle is used. Before this, a *needs to know* rule is added to the global policy. This rule can be specified as follows:

 \forall (*d*: Doctor, *i*: Info, *p*: Patient): needsToKnow(*d*,*i*) \land memberRecord(*i*, *p*) \supset canReadRecord(*d*, *p*)

The rule states that every doctor is permitted to read the required data from the medical record if the doctor needs to know this data and if the data is contained in the patient's medical record. Using this rule, the following alternate reasoning can be formulated:

 $\forall (p: Patient):$ $(hasCondition(<math>d_1, p, c_1$) hasCondition(d_1, p, c_2)) $(\exists (i: Info):$ $needsToKnow(<math>d_1, i$)

memberRecord(i, p))

The doctor claims that it needs to know the data since the patient has the medical conditions c_1 and c_2 . From the policy rule, provided that the data is contained in the patient's medical record, the doctor is allowed to read this data from the patient's medical record. Depending on the choice of verification tool, one can encode the knowledge needed by the doctor on the patient.

The *needs to know* rule need not be restricted to only existing data in the medical record. An alternative rule could consider the case where the required knowledge can only be derived from existing data in the medical record. Hence, the verifier would need to verify that the required knowledge does indeed depend on the existing data in the medical record.

While a full scale performance evaluation is beyond the scope of the paper, initial results are encouraging. Preliminary experimentation indicates that the overheads associated with the architecture are reasonable. The open source XACML tool from Sun [14] was used to implement the access control aspects of the system. In our test system, the number of roles was fairly limited (five were listed in our test set) although the number of individuals was in the hundreds.

For the first experiment, the PDP was set up with four global policy rules. These rules were specified manually for the purposes of the experiment, where predicates in the policies (such as *isEmergency*) were set up as attributes. The PEP could use these predicates along with the request for a particular type of access. The experiment involved the PEP sending a number of random requests along with (or without) the relevant attributes. The time to execute 200 requests was on average 3.8 seconds (which is about 19 milliseconds per request) with a standard deviation of 0.196 seconds.

For the second experiment, we doubled the number of policy rules but maintained the same number of requests. The average time to execute

200 requests in this extended experiment was 3.9 seconds with a standard deviation of 0.242 seconds. As these results show no significant differences, it would appear necessary for future experiments to evaluate the performance of test systems with much larger test sets and more policy rules.

In terms of the testing platform, the experiments were executed on a virtual machine hosted on an Intel® Xeon[™] CPU 3.00GHz server with 2GB RAM. As our experiments were not memory intensive, we limited the memory of the virtual machine to 256MB. The experiments indicate that performance overheads for using off-the-shelf tools are reasonable for the case study described in this paper. The plug and play nature of the architecture will, in principle, enable the users to replace one subsystem with another. This permits the use of better tools as and when they become available. The implementation of this architecture for large scale examples needs further work. Depending on the application, the various PEPs and PDPs many need to be distributed across various machines with coordinating agents ensuring that only the appropriate access is given.

4.2. Trust

Trust-based rules can also be included in the policies. This extension to the case study was inspired by the work of Jøsang [15] presented at the recent National e-Health Privacy and Security Symposium (ehPASS'06). Jøsang focuses on evaluating trust relationships in the health sector using reputation systems. We consider this idea of trust relationships for expressing policy rule conditions. By referring to trust relationships in the policies, this provides an alternative approach towards enforcing patient privacy.

For example, let us consider a simple trust relationship between a patient and a doctor. The global policy rule states that if the patient trusts the doctor, then the doctor can read the patient's medical record. This is specified formally as:

 \forall (*d*: Doctor, *p*: Patient): trust(*p*, *d*) \supseteq canReadRecord(*d*, *p*)

Each patient can individually specify those doctors which are trustworthy. An example patient privacy policy rule could be the following:

 $\forall (d: \text{Doctor}): \\ \text{memberGroup}(d, g_1) \supset \\ \text{trust}(p_1, d) \end{cases}$

This rule states that if the doctor is a member of group g_1 , then the patient p_1 trusts the doctor. In other words, this rule says that patient p_1 trusts all doctors of group g_1 . It follows from the global policy rule that if the doctor is a member of group g_1 , then the doctor can read the medical record of p_1 . This example shows how trust relationships can be used for making privacy-related decisions.

5. Conclusion

In this paper, we have introduced an e-health architecture to support the secure circulation of electronic medical records. Although our architecture covers multiple issues related to ehealth, we have focused on the protection of patient privacy (and specifically, how to deal with unforeseen privacy situations). We differentiate between standard privacy protections (applied to all medical records) and personalised privacy protections (defined by the patient for their own medical record). When none of the privacy protections fully capture the circumstances for a recipient's usage for a particular medical record, we have shown how a simple reasoning scheme can be applied to close the information gap in the policies. Despite basic scenarios having been tested using the architecture, a large case study needs to be conducted to identify apppropriate tools and implementation strategies.

Acknowledgements

The authors thank Jorge Cuellar from Siemens Research (Munich, Germany) for his assistance with the case study and Xiangpeng Zhao from Peking University for comments on an earlier version of the paper.

References

1. Wilikens M, Feriti S, Sanna A, Masera M. A context-related authorization and access control method based on RBAC. SACMAT 2002: Proceedings of the seventh ACM symposium on Access control models and technologies; 2002; New York, NY, USA. ACM Press; p. 117–124.

2. Kowalenko K. Home as the hub of health care. The Institute; Jul 2006.

3. Stohr E, Zhao JL. Workflow automation: Overview and research issues. Information Systems Frontiers. 2001;3(3):281–296.

4. van der Aalst WMP, ter Hofstede AHM. Yawl: Yet another workflow language. Information Systems, 2005;30(4):245–275.

5. Web Services Business Process Execution Language Version 2.0. The Organization for the Advancement of Structured Information Standards (OASIS); c2007 cited 2008 Mar 10]. Available from: <u>http:// /docs.oasis-open.org/wsbpel/2.0/wsbpelv2.0.html</u> 6. Anzböck R, Dustdar S. Modeling and implementing medical web services. Data and Knowledge Engineering. 2005;55:203–236.

7. eXtensible Access Control Markup Language (XACML). The Organization for the Advancement of Structured Information Standards (OASIS); c2005 [cited 2008 Mar 10]. Available from: <u>http://</u> <u>docs.oasis-open.org/xacml/2.0/</u> <u>access_control-xacml-2.0-core-specos.pdf</u>

8. Agrawal R, Kini A, LeFevre K, Wang A, Xu Y, Zhou D. Managing healthcare data hippocratically. SIGMOD 2004: Proceedings of the 2004 ACM SIGMOD international conference on Management of data; 2004; New York, NY, USA. ACM Press; p. 947–948.

9. Hooda JS, Dogdu E, Sunderraman R. Health level-7 compliant clinical patient records system. SAC 2004: Proceedings of the 2004 ACM symposium on Applied computing; 2004; New York, NY, USA. ACM Press; p. 259–263.

10. Zhao X, Cerone A, Krishnan P. Verifying BPEL Workflows under Authorisation Constraints. In: Dustdar S, Fiadeiro JL, Sheth A, editors. Fourth International Conference on Business Process Management; 2006 Sep; LNCS 4102; Springer Verlag; p. 439–444.

11. Document Object Model. World Wide Web Consortium (W3C); c1997-2005 [cited 2006 Dec 16]. Available from: http://www.w3.org/DOM/ 12. Owre S, Rushby JM, Shankar N. PVS: A prototype verification system. In Kapur D, editor. 11th International Conference on Automated Deduction (CADE), 1992 Jun; Saratoga, NY, USA; LNAI 607; Springer-Verlag; p. 748–752.

13. Fisler K, Krishnamurthi S, Meyerovich LA, Tschantz MC. Verification and change-impact analysis of access-control policies. ICSE 2005: Proceedings of the 27th international conference on Software engineering; 2005; New York, NY, USA. ACM Press; p. 196–205.

14. Sun's XACML Implementation. Sun Microsystems, Inc.; c2003-2004 [cited 2007 May 9]. Available from: <u>http://sunxacml.sourceforge.net</u>

15. Jøsang A. Online reputation systems for the health sector. ehPASS: National e-Health Privacy and Security Symposium; 2006 Oct; Brisbane, Australia; p. 78–88.

Correspondence

Prof. Padmanabhan Krishnan Centre for Software Assurance School of Information Technology Bond University Gold Coast QLD 4229, Australia

Phone: +61 (0)7 5595 3367 Fax: +61 (0)7 5595 1160 http://www.sand.bond.edu.au

pkrishna@staff.bond.edu.au