# Protecting Privacy while Sharing Medical Data Between Regional Healthcare Entities

**Tyrone Grandison[a], Srivatsava Ranjit Ganta[b], Uri Braun[c], James Kaufman[a]**

[a] *IBM Almaden Research, 650 Harry Road, San Jose, California 95120*
[b] *Pennslyvania State University, University Park, PA 16802*
[c] *Harvard University, Cambridge, MA 02138*

**Abstract**

*Economies of scale, corporate partnerships and a need to increase the efficiency of Information Technology in the Healthcare sector are leading to the construction of Regional Health Information Organizations (RHIOs) across the United States. RHIOs are normally aligned by service provision given by particular healthcare payers (e.g. Blue Cross-Blue Shield, PacifiCare etc.) in particular geographies. Globalization has created a transient workforce that may require their healthcare provider access their patient data across several sovereign RHIOs. The barrier to enabling RHIO to RHIO collaboration lies in the need to respect the data disclosure policy of each RHIO, to adhere to the geography-specific healthcare legislation and also to not violate the express privacy wishes of the patient(s) involved. In this paper, we propose a data-level control called "Sticky Policy Enforcement" which allows sharing to occur across RHIOs, while adhering to the concerns mentioned.*

***Keywords: Privacy, Healthcare Systems, Collaboration***

## Introduction

Beyond the move to transform the American healthcare landscape by leveraging Information Technology to deliver better care, to reduce medical errors and to improve the quality of life, the existing healthcare topology dictates that computer networks be built that preserve the established business alliances that exist between payers and providers. This motivates the formation of connected information centers called Regional Health Information Organizations (RHIOs). The mandate to create a National Healthcare Information Network (NHIN) in the United States is based on the emerging existence of these RHIOs [1, 2]. A NHIN is a realization of a collaborative network of Healthcare Information Systems.

Formally, a RHIO is an independent regional collective that facilitates the development, implementation and application of secure health information exchange among participating care providers. Each RHIO has independent policies regarding the privacy of health records stored within the RHIO. Figure 1 presents the typical RHIO environment. A limited number of RHIOs exist today in the US and they vary in the ways they approach data sharing. The formation of a RHIO is based on the understanding that all the stakeholders agree to follow a specific set of guidelines.

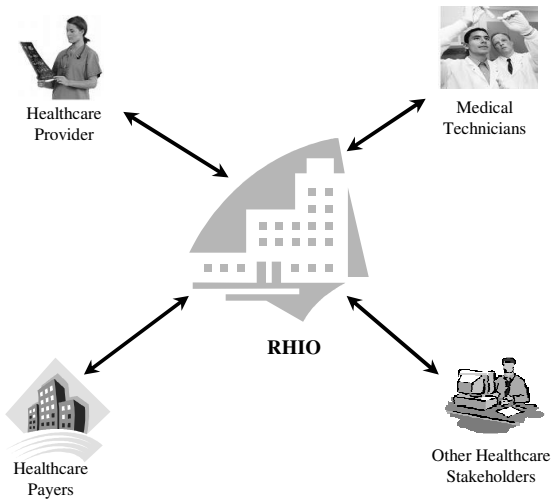


*Figure 1: Regional Health Information Organization (RHIO)*

These guidelines detail policies regarding access, dissemination and processing of the patient data in the RHIO. Currently, the mechanism for cross-RHIO information sharing differ so greatly that collaboration across RHIOs to deliver care is still a daunting problem, but critical for the successful adoption of RHIOs.

One of the key challenges is the protection of patient privacy when clinical documents are shared. This challenge is compounded by the facts that there can be no assumption of a central authority; that enforcement may involve multiple privacy policies based on source, destination and the documents involved in the transfer and the fact that data can be forwarded to an entity with additional rights, such as remote update rights.

This paper presents a first step towards addressing these concerns and marches healthcare systems towards achieving inter-RHIO collaboration while adhering to data disclosure constraints (e.g. privacy and security concerns). The technology is called *Sticky Policy Enforcement* and provides a way to en-

sure that policy constraints are enforced wherever patient data travels.

In this paper, we will provide a discussion on the related work in the field, highlight the *Sticky Policy* architecture, demonstrate the technology by walking through the enforcement steps and provide concluding remarks.

## Related Work

The first mention of *Sticky Policies* appeared in computer science literature at the start of the 21st century [3]. It emerged from IBM's work on Enterprise Privacy Authorization Language and was recognized as a concept that is important for privacy preservation in distributed computer systems. The underlying notion behind *Sticky Policy Enforcement* is that the policy applicable to a piece of data travels with it and is enforceable at each point it is used. Though identified over a half a decade ago as a critical problem, application-independent solutions that were technically feasible and scaleable were not realized.

Rivest and Lampson's work [4] embodies the earlier efforts in this space. Their focus was on the establishment of trust for a single disclosure object with a single policy. A data recipient is either granted access to the entire document, or must request authorization from the source. In healthcare environments, this is not sufficient. *Sticky policy* functionality should handle data disclosure to a party with well-defined constraints that allow data release to less privileged parties without requiring the originator's involvement. This avoids the potential pitfall of having to contact a (potentially) large number of third parties before making a decision to disclose a specific piece of information.

The work by the Trusted Computing Group (TCG) consortium [5] represents another, more popular, approach to establishing the trust in single object, single policy environments. The general concern with the traditional approaches is that they require targeted application development and are not application and data agnostic, which is a mandatory requirement for situations with a complex web of pre-existing infrastructure, which are likely to be from differing vendors and running different, even proprietary, systems. Additionally, an ideal approach to *Sticky Policy Enforcement* should account for the fact that data changes occur frequently. It is not clear how approaches like that of the TCG would handle this without incurring a severe performance penalty.

## System Description

The goal of our system is to enable distributed privacy policy enforcement. The difficulty in achieving this lies in the fact that there is no single entity with *a priori* access to all the policy constraints applicable to a document in a given state of the system. Our solution to this problem involves identifying the applicable privacy policy constraints for a document(s) to be shared and sticking them together, forming a single entity of transfer.

In taking the approach of packaging policy with data, we maintain centralized decision making in a distributed enforcement. As only policy constraints that apply to the disclosed data are transferred, the communication impact is relatively small and the system does not require prior agreement among all medical organizations, states and patients.

### HIPPOCRATIC DATABASE TECHNOLOGY

Our solution approach to the distributed privacy policy enforcement problem leverages the Hippocratic Database (HDB) Active Enforcement (AE) technology [6], which provides cell-level, policy-based disclosure management functionality, such that databases only return data that is compliant with company policies, applicable legislation, and customer preferences. The AE component ensures that enterprise applications accessing a database adhere to fine-grained data disclosure policies. These policies, which may be security policies, privacy policies or data management policies, are distilled from the company's own policy, legal and regulatory requirements, customer preferences, and opt-in and opt-out choices. The component automatically rewrites user requests (i.e. queries) and returns only data that is consistent with these policies, allowing applications to enforce disclosure policies on arbitrary data elements at query execution time. A detailed description of HDB AE can be found in [7]. A quick overview of its operation is that, for a centralized or federated data system, HDB AE allows the definition of fine-grained disclosure policy, the creation of user preferences, the resolution of conflicts between preferences and policy and the enforcement of all the applicable constraints in an architecture that is application and database agnostic.

HDB Active Enforcement technology was chosen as our platform because: (i) it offers a general platform for handling and codifying privacy policy and preference information; (ii) its enforcement mechanism is transparent to enterprise applications (integration currently assumes a database interface such as ODBC or JDBC); (iii) it is agnostic to underlying database technology; (iv) it allows policy changes without any modifications to the applications in use; and (v) in the typical case, it improves query processing speed.

### STICKY PRIVACY POLICY

The format of our *Sticky Policy* package (Figure 2) consists of three parts: Data, Policy and Audit information. The Data segment contains the health documents requested. The Policy segment embodies the policy constraints applicable to the documents to be made available. The semantics of the policy entries are:

1. Requestor: The entity requesting access to part(s) of clinical document(s) from the source. The values for this entry could be taken from the roles mentioned in CDA standard.
2. Recipient: The entity that will be the final consumer of the data. The domain of possible values is similar to the set used for a requestor.
3. Purpose: The purpose for which the document(s) is being requested.

4. Retention: Time period until which access to the data is allowed. This could be computed based on various organizational policies.
5. Copy-forward: The condition specifying whether the recipient is entitled to forward the requested document(s) to a third party after copying. The set of possible values are:

| **Copy forward** | |
|---|---|
| ***Yes w/notification*** | *May copy and forward the data with a notification to the sender* |
| ***Yes w/o notification*** | *May copy and forward the data without any notification* |
| ***No*** | *May not forward at all* |
| ***Ask*** | *Must ask the sender on forward* |

6. Append/Modify: The Boolean condition specifying whether the recipient can append/modify the document.

| **Append/Modify** | |
|---|---|
| ***Yes w/notification*** | *May append/modify with a notification* |
| ***Yes w/o notification*** | *May append/modify without any notification* |
| ***No*** | *May not append/modify at all* |
| ***Ask*** | *Must ask on append/modify attempt* |

The Audit section of the sticky policy consists of information including the source, requestor, a timestamp and digital signature to verify the authenticity of the sticky policy.

**Data**
- Clinical Document 1
- ..
- Clinical Document n

**Policy**
- Requestor
- Recipient
- Purpose
- Retention
- Copy forward
- Append Notification

**Audit**
- From
- To
- Timestamp
- Verifiable Signature

*Figure 2: Format of Sticky Privacy Policy*

The source and requestor information is used by the auditor to track the data while the timestamp is used to determine the causal ordering. The digital signature serves two purposes: 1) to maintain the integrity of the healthcare documents, i.e. guaranteeing that the document(s) are not tampered with, and, 2) to ensure the non-repudiation of the sticky policy.

## ARCHITECTURE & MODEL

In solving the distributed policy enforcement problem, we assume a trust model where authenticated users are honest-but-curious. Thus, our attack model focuses on the user who inspects the data they receive and attempts to gather data that they are not entitled to, and does not address malicious users who attempt to gain access to data they have not received even if it violates the policies.

Figure 3 illustrates the conceptual architecture of our system and showcases the creation and management of sticky policies.

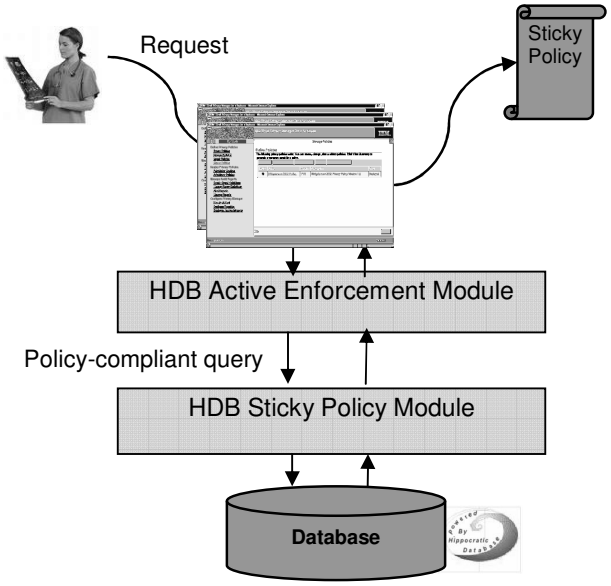


*Figure 3: Sticky Policy Enforcement Architecture*

Our enforcement model consists of two approaches:

***Proactive enforcement*** involves prevention of unauthorized disclosure before it occurs by blocking operations or suppressing results that may lead to a privacy violation.

***Reactive enforcement*** involves detection of the violation through audits. This is based on an optimistic assumption that the environment is non-malicious.

These approaches differ in their points of execution. Proactive enforcement eliminates violations before they occur, which has limited use in scenarios when *a priori* knowledge of all the possible access situations is not possible. For example, in an emergency situation, violations should be allowed as long as they are auditable and necessary for the delivery of care. Reactive enforcement achieves this by tracking all the access information and assuming the existence of a trusted auditor system. The auditor must be able to access data from the source and recipient including any intermediaries between those parties. For the US healthcare environment, this could be the responsibility of the Department of Health and Human Services or its delegates. In the case of a violation, the system presumes the node to be guilty and demands records to certify innocence.

For distributed environments, proactive and reactive enforcement can be achieved through either a centralized or a federated approach leveraging a set of cooperating enforcers.

In our design, we perform proactive enforcement through HDB Active Enforcement technology. In the case of reactive enforcement, a centralized approach employs a single auditor which is trusted and authorized to investigate all aspects of a suspected policy violation. In a decentralized model, a set of cooperating auditors can be employed with each team responsible for a specific set of nodes, data or both.

Beyond the issues of proactive and reactive enforcement, there is a question of where the enforcement occurs. We consider two possible locations: at the source and at the recipient. Enforcement at the source is simpler because it relies on the source's controls. Enforcement at the recipient places trust in all recipients. However, some enforcement can only occur at the recipient. For example, restricting the recipient from forwarding the result on to others is not something the source can enforce. Our approach is to attempt to perform all enforcement at the source and only rely on enforcement at the recipient when no alternative exists.

## Scenario

Initially, a request for clinical documents is placed through an application interface, which issues a query or a set of queries. On receiving the query, the proactive enforcement module rewrites the query to account for all applicable disclosure policies. In a traditional HDB-enabled system, the rewritten query would continue directly to the data system's native query execution engine. However, for our system, the rewritten query is redirected to the sticky policy module where the query is further modified to create a sticky policy result set. A digital signature is computed on the source using a User-Defined Function (UDF) and included in the sticky policy. The final sticky policy is then transferred to the recipient in its entirety. Figure 4 provides a sample healthcare sticky policy.

For this prototype, we chose to use XML as the data format for representing the sticky policy for multiple reasons. The interoperability among various EMR, HER and PHR systems used in the healthcare industry has been limited based on the proprietary interfaces and standards [8]. Although, XML processing as in the case of any text processing involves a lot of overhead, it offers the much needed features of platform independence and simplicity demanded by the healthcare industry. When the XML processing overhead becomes unbearable, specialized hardware and software can be used to mitigate this concern.

For the purposes of this work, we assumed an agreement on vocabulary among interoperating parties. We also assume that proactive enforcement is achieved by leveraging HDB Active Enforcement. On the recipient, the AE component accepts the received sticky policy, assigns a unique id to the policy and stores an unaltered copy of policy for the purposes of auditing. The policy elements are then de-coupled and the corresponding data and policy constraints are extracted. The policy rules are then entered into the HDB metadata tables. The document(s) or their part(s) are then stored in a database and links are created from the entries in the HDB metadata

Reactive enforcement is achieved by traversing the sticky policy audit logs to find a violation or to prove innocence. An audit begins with the suspicion of a privacy violation. We presume that any party with access to the data but without a sticky policy is guilty. In essence, the sticky policy is a certificate of innocence. This is similar to not having a license demonstrating legal ownership of a software product.

The auditor searches the database for the data item for which enforcement is presumed as violated. Once identified, the auditor checks the HDB metadata, and identifies the relevant policy and archive entries. If the ability to identify these entities does not exist (i.e. there is no policy or the archive entry is missing), then a violation has occurred. The auditor then tries to verify the signature on the sticky policy stored in the archive table. Again if the signature is not valid, a violation has occurred.

**<ClinicalDocument 1 …>**
<patient><name><given>James</given><family>Woods</family></name>…
<assignedPerson><name><given>Sarah</given><family>Beach</family><suffix>PhD.</suffix></name></assignedPerson>…
<section><title>History of Present Illness</title><text>James Woods has suffered from calcific bursitis.</text>…
<title>Physical Examination</title>…
<title>Vital Signs</title><text>The patient's height, weight, and body mass index were measured to be 2.29489 meters, 400.05671738536824 pounds, and 34.45 kilograms per meter squared, respectively.</text>…
**<ClinicalDocument 2 …> …**

**Requestor** – Alice/ Admin Role/ ArizonaCare
**Recipient** - Bob/ Staff Physician/ ArizonaCare
**Purpose** – Emergency Case
**Retention** – 30 days
**Copy-Forward** – Yes With Notification
**Append/Modify** - No

**From** – Trina/ Admin Role/ CalShield
**To** – Alice/ Admin Role/ Arizona Care
**Timestamp** – Nov 19 2006
**Signature - …**

*Figure 4: Sample Sticky Policy for Healthcare RHIO Sharing*

If everything has been successful so far, the auditor checks to make sure the sticky policy and data content agree. The auditor compares the policy in the policy table to its counterpart in the sticky policy and similarly compares the data in the sticky policy with that in the database. The auditor also verifies that the HDB metadata tables cover all the data included in the sticky policy. Even if everything checked so far is OK, the audit is not stopped.

It is possible for an enforcement breach to have occurred before the current node even received the sticky policy. The auditor then traverses through the sticky policy to identify the node that forwarded the sticky policy. The auditor continues the audit up the chain to the originators until he reaches the bounds of his jurisdiction or the first sender.

## Performance Discussion

We ran experiments to evaluate the overhead cost of sticky policy generation and sticky policy consumption. In the interest of terseness, we will just provide the results here.

Our experimental platform used a synthetically generated dataset based on the Clinical Document Architecture [9]. All experiments were run using IBM DB2 UDB 8.2. The operating system was Microsoft Windows XP with Service Pack 2. The hardware consisted of a PC with Pentium-4 2.4GHz processor and a 60GB disk. The buffer pool was set to 1 MB. All other DB2 default settings were used.

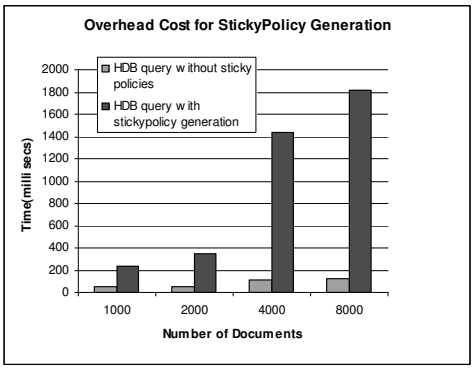


*Figure 5: Overhead Cost for Sticky Policy Generation*

We examined document sharing for sets of 1000, 2000, 4000 and 8000 documents, which is well over the current limits for healthcare document sharing. It was observed that the overall cost introduced by sticky policy generation over the privacy preserving query processing in HDB is acceptable (Figure 5) considering that the generation is done using XML.

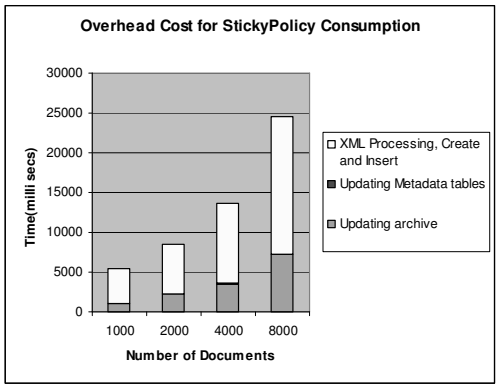


*Figure 6: Overhead Cost for Sticky Policy Consumption*

For policy consumption, the time elapsed in updating the metadata tables and the archive is less than 30% of the overall policy consumption cost (Figure 6). As pointed earlier, XML hardware appliances may be used to reduce the overall consumption cost.

## Future Work

Our future efforts will focus on 1) further deployment and enhancement of the current *Sticky Policy* functionality, and 2) innovating new approaches to enable *Sticky Policy Enforcement* in distributed systems with a central repository. On the first task, we plan to start by including technology that increases the transmission security strength. i.e. removing the assumption that the channel is inherently secure and encrypting the shared documents before transfer. Then we will remove the assumption of an honest-but-curious user and consider hostile environments; touching on the difficulties in considering Byzantine failures or collusion among several participants. For the second task, we will assume a system of shared policies and construct mechanisms to provide privacy guarantees when data is transferred.

## CONCLUSIONS

The construction of RHIOs and the sharing of information between them is an important pre-requisite for the successful creation and deployment of a National Healthcare Information Network (NHIN). Very little attention has been placed on technology to enable this RHIO to RHIO collaboration in a privacy preserving manner, till now. In this paper, we present *Sticky Policy Enforcement* technology, which provides mechanisms to perform proactive and reactive enforcement when sharing clinical documents between RHIOs.

## References


[1] The Goals of Strategic Network. http://www.hhs.gov/healthit/goals.html.

[2] The California Regional Health Information Organization. http://www.calrhio.org/.

[3] Stufflebeam, William, Antón, Annie I., He, Qingfeng, and Neha Jain. Specifying Privacy Policies with P3P and EPAL: Lessons Learned. Proceedings of the 2004 ACM workshop on Privacy, Washington DC, USA (2004).

[4] Rivest, Ronald L., and Lampson, Butler. SDSI – A Simple Distributed Security Infrastructure. Working Document v1.1

[5] TCG Specification Architecture Overview. Revision 1.2 Retrieved 2006 from https://www.trustedcomputinggroup.org/specs/IWG/TCG_1_0_Architecture_Overview.pdf

[6] K. Lefevre, R. Agrawal, V. Ercegovac, R. Ramakrishnan, Y. Xu, D. DeWitt. "Limiting Disclosure in Hippocratic Databases". Proc. of the 30th Int'l Conf. on Very Large Databases (VLDB 2004), Toronto, Canada, August 2004

[7] IBM Almaden Research. Hippocratic Database Active Enforcement: User's Guide. Retrieved Nov. 28, 2006 from http://www.almaden.ibm.com/cs/projects/iis/hdb/Publications/papers/HDBEnforcementUserGuide.pdf

[8] Srivatsava Ranjit Ganta, Eishay Smith, Sarah Knoop, Sondra Renly, James Kaufman. "The Eclipse Open Health Framework", 5th International Conference on Healthcare Technology and Management (HCTM 2006), Chicago, August 2006.

[9] Dolin RH, Alschuler L, Boyer S, Beebe C, Behlen FM, Biron PV, Shabo Shvo A, "HL7 Clinical Document Architecture, Release 2", Journal of American Medical Informatics Association (JAMIA), 2006