Security Middleware Infrastructure for Medical Imaging System Integration and Monitoring

Weina Ma, Kamran Sartipi

Department of Electrical, Computer and Software Engineering, University of Ontario Institute of Technology, 2000 Simcoe St N, Oshawa, Ontario, Canada

{Weina.Ma, Kamran.Sartipi}@uoit.ca

***Abstract*— With the increasing demand for electronic medical records sharing, it is a challenge for medical imaging service providers to protect the patient privacy and IT infrastructure security in an integrated environment. In this paper, we present a novel security middleware infrastructure for seamlessly and securely linking legacy medical imaging systems, diagnostic imaging web applications as well as mobile applications. In this infrastructure, software agents such as user agent and security agent have been integrated into medical imaging domains that can be trained to perform their tasks. The proposed security middleware utilizes both online security technologies such as authentication, authorization and accounting, as well as post security operations to discover system security vulnerability. By integrating with the proposed security middleware, both legacy system users and Internet users can be uniformly identified and authenticated; access to patient diagnostic images can be controlled based on patient's consent directives and other access control polices defined at a central point; relevant user access activities can be audited at a central repository; user access behavior patterns are studied by utilizing data mining techniques; the explored behavior patterns provide system administrators valuable knowledge to refine existing security policies; behavior-based access control is enforced by capturing user's dynamic behavior and determining their access rights through comparing with the discovered knowledge of common behaviors. A case study is presented based on the proposed infrastructure.**

***Keyword*—Behaviour Pattern, Data Mining, Medical Diagnostic Imaging, Middleware, Security**

I. INTRODUCTION

MORDEN Diagnostic Imaging (DI) solutions maintain and manage patient radiology images (e.g., CT scans, X-ray, MRI, ultrasound), and corresponding diagnostic reports in digital formats, for the purpose of diagnosis, treatment improvement and medical science research. Over the past decades, Picture Archiving and Communication Systems (PACS) have taken a dominant role in the workflow of DI solutions in a single hospital or radiology department. A federated DI domain allows for a centralized capture, long-term archiving and non-proprietary sharing of radiology information across a large distributed network. A central diagnostic imaging repository (DI-r) provides common services to the participating hospitals. According to the status of DI-r projects across Canada [1], 19 provincial DI-r's have been developed or being developed to reliably maintain, deliver and share DI information to consumers within the electronic health record (EHR) systems. Meanwhile, mobile health information technology (mHealth) is increasingly important in telemedicine, but traditional security infrastructure deployed in PACS and DI-r systems is not ready for accessing DI records through mobile devices.

Integrating the Healthcare Enterprise (IHE) has developed a number of integration profiles [2], [3] that address security requirements to improve the way computer systems in healthcare share information. These security control requirements are achieved through a trusted model where each local medical imaging system is responsible for ensuring that the personal health information is adequately protected. A key challenge with this trusted model is the lack of federated capabilities: i) access control rules are local to each system, which means consistency of access rules across all systems has to be managed manually; ii) patient consent directives and their impact on access control are not communicated automatically to each system; iii) user authentication is local to each system that imposes a significant administrative burden to ensure that individuals are uniformly identified in each system; iv) access to data is audited in each local system which also imposes a significant burden to investigate inappropriate access or monitor security breaches.

Middleware is a software layer that lies between service providers and consumers in a distributed computer network. Our proposed security middleware enables secure radiology image sharing among different provincial DI-r's, heterogenous PACS systems in distributed hospitals, as well as web clients and mobile clients. The main objective of this study is to propose an infrastructure for development of security middleware that provides: online security mechanism

Manuscript received October 12, 2015. This work is sponsored by an Ontario Research Fund (ORF) grant, and a follow-up of the invited journal to the accepted out-standing conference paper of the 17th International Conference on Advanced Communication Technology (ICACT2015). Grant ID: RE-05-073. Project: Secure Intelligent Content Delivery System for Timely Delivery of Large Data Sets in a Regional/National Electronic Health Record. This research was conducted with collaboration of Dr. David Koff and Dr. Peter Bak at MIIRCAM Centre of McMaster University.

W. M. is with Department of Electrical, Computer and Software Engineering, University of Ontario Institute of Technology, 2000 Simcoe St N, Oshawa, ON L1H 7K4, Canada (corresponding author, phone: +1-905-721-8668, fax: 905-721-3178, e-mail: Weina.Ma@uoit.ca).

K. S. is with Department of Electrical, Computer and Software Engineering, University of Ontario Institute of Technology, 2000 Simcoe St N, Oshawa, ON L1H 7K4, Canada (e-mail: Kamran.Sartipi@uoit.ca).

such as common authentication and authorization methods; post security mechanism that assists system administrators in exploring user access behavior patterns by mining audit logs; and applying behavior based access control by capturing user's dynamic behavior, and determining access rights through comparing with the discovered common behaviors. In this context, the main contributions of this paper include: i) designing middleware architecture for seamlessly and securely integrating legacy medical imaging systems; ii) proposing a behavior-based technique which allows to detect outlier behaviors and enhance the system's access control policies; iii) presenting a new method to measure behavior similarity and outlier degree; and iv) introducing generic software agents which can be customized and trained to perform the assigned tasks (e.g., access control, or auditing).

The remaining of this paper is organized as follows: Related work is discussed in Section II. Section III presents the proposed infrastructure of security middleware, and user behavior monitoring. Section IV is allocated to a case study, and finally conclusion is presented in Section V.

II. RELATED WORK

IHE is an initiative by healthcare professionals and industry which aims at setting up consolidated healthcare information sharing through standards based approaches [4]. It guides enterprises in using established standards to achieve interoperability based on existing IT infrastructure. However, the IHE suggested trust model in cross-enterprise domains lacks federated capabilities. Also, the small and medium scale medical service providers lack the proper skills and technology to make reliable and accurate authorization decision independently, especially in cloud and mobile computing environments. In such context, we introduce a security middleware that provides one common method for integrating a broad range of medical service providers.

A software agent is a program that acts on behalf of an agency for different users or other programs. The notion of generic and lightweight agent that resides at client side to be utilized by different service providers is introduced in [5]. The agents can be customized and trained based on the service provider generated role description and knowledge to perform the assigned tasks. This technology is an extension of the service-oriented architecture (SOA) model that allows for providing personalized services and maintaining client privacy through processing client's data locally. In our proposed architecture, we use cooperative-agents that reside at both client side and service provider side to interact with the security middleware and perform the assigned tasks.

In an earlier work [6] and [7], we proposed a general and secure infrastructure for sharing medical images between PACS and EHR systems. The proposed environment in that work was based on federated authentication and authorization techniques (OpenID and OAuth) [8], and cooperative agents with dedicated tasks to provide both action-based and behaviour-pattern based access control. As for legacy PACS systems, an agent-based approach [9] is proposed allowing for capturing PACS communication messages, identifying PACS users and extracting user actions to feed into an action-based access control mechanism.

Most of the existing access control models deal only with static systems. Behaviour-based access control for distributed healthcare systems is initially introduced in [10]. The proposed access control model captures the dynamic behavior of the user, and determines access rights through comparing with the expected behavior. Ideally, the distance between observed behavior and expected behavior is significant if the user acts abnormally. This model is also applied in security sharing of medical images [6]. In our proposed architecture, we define a behavior pattern as: *consistent observations of a sequence of actions that a user or a group of users conducted in a common context during a specific time interval (e.g., a session, a day, a week)*. Our work enhanced the behaviour-based access control by proposing a new behaviour similarity metric to determine the closeness between the observed dynamic behaviour and discovered common behaviour, and introducing an outlier degree to detect outliers.

Despite the placement of security mechanisms such as authentication, authorization and secure communication in most systems, authorized users, intended or carelessly, exhibit risky behaviours that may cause data leakage or damage to protected resources. Examining human behaviour among authorized users is helpful in assisting security professionals to make access control decisions. Our proposed security middleware provides: online security services to identity and authorize user access; and post security services to monitor and analyse the authorized user's access behaviour patterns. Such an acquired knowledge can lead administrators to security policy enhancements.

Acquiring decent user access behaviour patterns is crucially important in our approach. We analysed the audit logs of distributed PACS systems, and extracted sequencing, association and timing constraints to represent a behavior pattern: sequencing requires that a series of steps occur in a certain order; timing limits the occurrence frequency of certain values; and association identifies the cases where two or more system values occur at the same time. We employ data mining techniques in user access behaviour discovery. Association rules mining was originally introduced by Agrawal [11], aiming at analyzing customer purchase habits by finding association relations between items in the customer shopping baskets. Sequential pattern mining was also proposed by Agrawal [12], detecting frequently occurring ordered events or subsequence as frequent patterns. There are many applications involving sequenced data, such as customer shopping sequences, web click streams, and biological sequences. Clustering is a method of grouping objects in a way that objects in one cluster are very similar to each other but they are dissimilar to the objects in other clusters [13]. Similarity-based clustering methods define and utilize similarity metrics to determine the closeness between the pairs of objects [14]. We proposed a behavior model based on association, sequencing and time constraints, which utilizes association mining, sequential pattern mining and similarity-based clustering techniques to explore user behaviors from audit logs.

An obvious measure of the closeness of two sequences is to find the maximum number of identical items in those two sequences (preserving the symbol order), which is defined as Longest Common Subsequence (LCS) of the sequences [15]. Formally, let $X=(x_1, x_2, \ldots, x_m)$ and $Y=(y_1, y_2, \ldots, y_n)$ be two sequences of lengths m and n, respectively. A common subsequence *cs* of X and Y represented by $cs(X, Y)$ is a subsequence that occurs in both sequences. The longest common subsequence *lcs* of sequence X and Y, $lcs(X, Y)$ is a common subsequence of both sequences with maximum length. The length of $lcs(X, Y)$ is denoted by $R(X, Y)$. Solving $R(X, Y)$ is to determine the longest common subsequence for all possible prefix combinations of the two sequences X and Y. Let $r(i, j)$ be the length of the *lcs* of x_i and y_i, where $x_i = (x_1, x_2, \ldots, x_i)$ and $y_i = (y_1, y_2, \ldots, y_j)$. Then $R(X, Y)$ can be defined recursively as following [14]:

$$r(i,j) = \begin{cases} 0 & if\ i = 0\ or\ j = 0 \\ r(i-1, j-1) + 1 & if\ x_i = y_j \\ \max\{r(i-1, j), r(i, j-1)\} & if\ x_i \neq y_j \end{cases} \quad (1)$$

III. PROPOSED INFRASTRUCTURE

The overall architecture of the proposed security middleware infrastructure for medical imaging system integration and monitoring is shown in Figure 1 and its detailed workflow is shown in Figure 2.

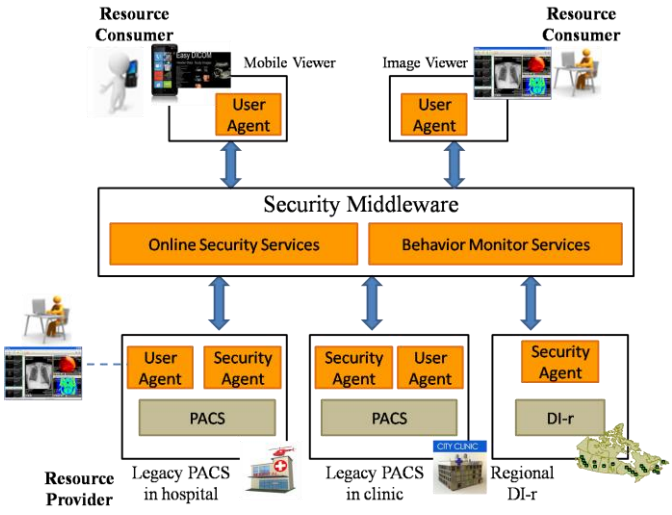

Fig. 1. Architecture for security middleware integration with legacy PACS, DI-r's and client applications

A. Architecture

Figure 1 illustrates the proposed architecture, where the client's access requests can be authorized under different access control models in legacy PACS and DI-r domains, but they are ruled according to the unified access control policies. The Security Middleware monitors and analyses user access behaviour patterns and assists the system administrators in consolidating existing access control policies based on the acquired knowledge from the extracted behaviour patterns. The components of the architecture are as follows.

Resource Consumer, is a medical imaging viewer (including mobile image viewer) that provides quality diagnostic images to the end users. According to the definition of SOA, both provider and consumer are roles that are played by software agents on behalf of their owners.

Resource Provider, is a medical imaging system that provides electronic image storage and convenient access to images from multiple resource consumers.

User Agent, is a software agent that is deployed at the client side to perform authentication request on behalf of the client application (e.g., image viewer) against the Security Middleware.

Security Agent, is a generic agent that is deployed at the server provider side for making access control decisions and collecting information about the user activities. Security Agent is customizable and trainable for different authorization models. The security middleware sends control information (access control polices), training data (authorization model) and assigned tasks (collecting user activity events) to customize and train a Security Agent. Based on the acquired training, assigned tasks, and user's data, Security Agent acts as a local access control mechanism. It also performs some filtering operations on the collected local user activities to allow for the behavior monitoring services at the Security Middleware.

Security Middleware, is an infrastructure that utilizes both online security technologies such as authentication, authorization and accounting, and post security procedures such as association and sequential pattern mining and pattern extraction to monitor users' behaviors.

Online Security Services, supports a set of centralized user directories and provides a common service that handles all user authentication requests. It also provides centralized access control policy management and a set of authorization models. The existing IT infrastructure in legacy domains is operating based on different technologies, procedures and models. It is not necessary to employ exactly the same access control mechanism across these domains, but it is necessary that they agree at the policy level.

Behaviour Monitor Services, provides the mechanism for monitoring the activities within the resource consumer and medical imaging systems. Data mining engines are employed to assist the system administrators obtain deep insight into the user access behavior patterns. With the system administrator's agreement, the discovered behavior pattern knowledge (common behavior) is sent to Security Agent as training data. At the same time, a behavior based access control task is assigned to Security Agent. Security Agent monitors the users' dynamic behaviors and compares with the common behaviours. Security Agent notifies the system administrator if any user behaves significantly different from the identified common behaviours.

A typical PACS system contains: image acquisition devices namely modalities (e.g., CT scan, MRI system); image archives where the acquired images are stored; and workstations where radiologists view the images. Both User Agent (serving workstations and modalities) and Security Agent (serving image archives) are deployed at each PACS system. The DI-r provides registry services for querying patient's medical images from legacy systems, and repository

service for storing and retrieving medical images. Security Agent is deployed to each DI-r system serving such services.

B. Workflow Model

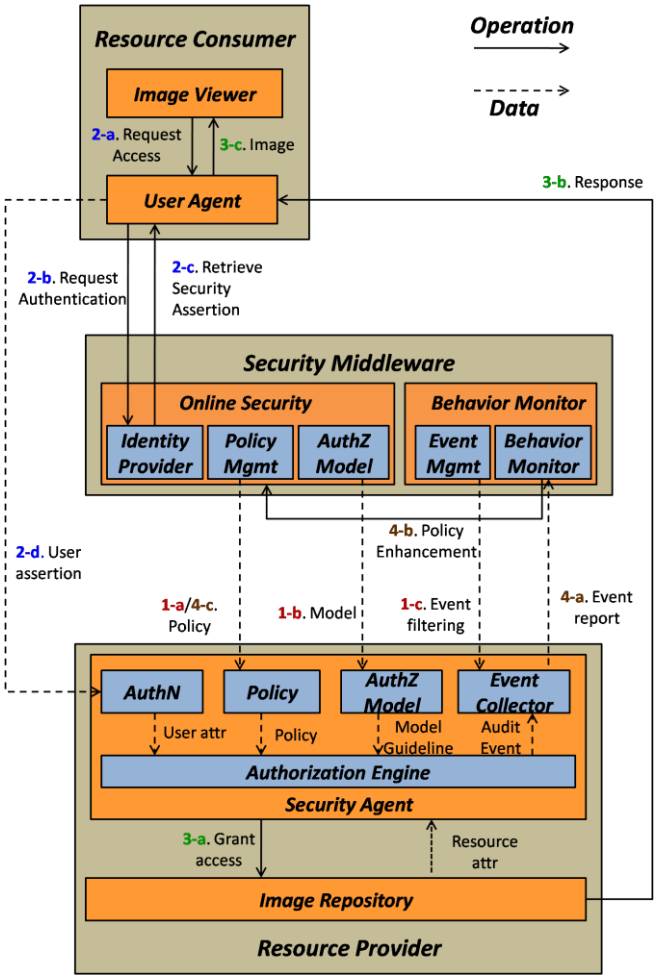

Fig. 2. Authentication, authorization and user behavior monitoring workflow for the proposed security middleware

The overall workflow model is shown in Figure 2. The steps of the model's operations are as follows.

Step 1) ***Security Agent customization*** (1-a to 1-c, red colour). Security Middleware generates the required training knowledge to train the generic Security Agent. The training knowledge is defined as a set of: 1-a) role based *access control polices* that are applicable to the protected resources; 1-b) *authorization model* that defines the access control procedure, and information-provider servers such as user attribute provider and resource-attribute provider; 1-c) *event filtering criteria* to be used for collecting user's access to resources. Security Agent receives the provided knowledge as well as the relevant Resource Provider's context, and then modifies the general authorization process and event collection task for the purpose of behavior analysis.

Step 2) ***Authentication*** (2-a to 2-d, blue colour). User Agent is a software agent deployed at the Resource Consumer. Image Viewer employs User Agent to fulfil the authentication flow (2-a). *Identity Provider* is an identity authentication server that is capable of authenticating the end users (2-b) and provides "security assertions" containing authentication statement and user attribute statement (2-c). A user assertion is communicated between User Agent and Security Agent for exchanging authentication and authorization data (2-d). Authentication statement confirms that the user has been identified and approved by the authentication server; the attribute statement asserts that the user is associated with certain attributes. These asserted attributes feed Security Agent to make access control decisions.

Step 3) ***Authorization*** (3-a to 3-c, green colour). Resource Provider sends instructions to Security Agent to perform authorization. Security Agent constitutes the following components: Authorization Engine that evaluates applicable policies and renders an access control decision; AuthN that provides the user's associated attributes; Policy that contains security middleware assigned policies and sends relevant policies to Authorization Engine for a specified target; AuthZ Model that guides Authorization Engine to fulfil the agreed-on authorization procedure; Event Collector records the authorization decisions. If this access request is granted, Security Agent sends an access request to Image Repository (3-a). Image Repository serves the request and returns its response (e.g., requested image) to User Agent (3-b). User Agent forwards the requested resource (image) to Image Viewer (3-c).

Step 4) ***Behaviour pattern mining and policy enhancement*** (4-a to 4-c, brown colour). User behaviour pattern is defined as consistent observations of a sequence of actions performed by the same user, under certain environment and during a specific time interval. Event Collector sends the collected data (i.e., event-log data) to Behavior Monitor component after filtering out the uninterested events (4-a). A knowledge driven behavior pattern discovery process is applied to orchestrate user's common behaviour patterns. Finally, the system administrators explore the opportunities to refine existing security policies by means of analysing salient features and characteristics of the discovered behaviour patterns (4-b). Finally, the consolidated polices are dispatched to the corresponding Security Agent to take effect (4-c), which closes an access control policy loop.

C. Behavior Anomaly Definition

Behavior anomaly is widely classified into the following three categories: i) *point anomaly*: where an individual data instance is considered as anomalous with respect to the rest of dataset; ii) *contextual anomaly*: where an individual data instance is considered as anomalous in a specific context, but might be considered as normal in a different context; and iii) *collective anomaly*: where a collection of related data instances is considered as anomalous with respect to the rest of dataset; however, the individual data instances in the collection may not be anomalous [15]. We propose a new method to detect contextual collective anomalies. In the followings, we define the data instance as an event that constitute a set of attributes; and define contextual collective anomalies as outlier behaviours that are dissimilar with the common behaviours in a specific context.

Event

An *event* records a single user-system interaction (i.e., any communication with the system such as storing and retrieving

a diagnostic image). An event is composed of a set of domain specific attributes. Whenever an attribute value changes, a new event is recorded. For example, an event of PACS system is represented by a tuple of attributes, as follows: *Event = <User, Role, Location, Action, Resource, Patient, Emergency>*. The attributes can be classified into three categorizes:

1) *Actor attributes.* The actor attributes are used to explain the subject of events. For example, *User* is an actor attribute, which identifies an individual who performed the action; *Role* is also an actor attribute which determines a group of people having similar privileges and responsibilities.

2) *Contextual attributes.* The contextual attributes determine the context (or neighbourhood) of events. For example, *Location* can be a contextual attribute which limits the neighbour events happened at the same location or nearby; *Time* can be considered as a contextual attribute which determines the neighbour events happened within a short period of time; *Patient* could be a contextual attribute which explains the neighbor events should be accessing the health records of a specific patient.

3) *Behavioral attributes.* The behavioral attributes define the non-actor and non-contextual characteristics of the events. For example, *Action* is a behavioural attribute, which describes one step of the workflow under a specific scenario; *Location* can also be a behavioral attribute which indicates one location of ward-round by nurses. Behavioral attributes in a dataset may be contextual attributes in another dataset, such as location that is a behavioral attribute in robot moving dataset but a contextual attribute in service accessing dataset.

Behavior

User behavior is extracted from a collection of user-system interactions (i.e., events). We propose a user behavior pattern representation based on association, sequencing and timing rules. *Association* indicates the concurrence of a set of attribute values together. *Sequencing* requires that a series of steps occur in a certain order. *Timing* allows sequencing the events; limits the events' occurrence frequency; and assigns the gaps between successive events.

In our approach, behavior is represented as a quadruple:

Behavior=<Actor, Sequence, Context, Time Interval>

Where *Actor* issues a behavior; *Sequence* is the sequence of steps performed by the Actor; *Context* is the circumstances in which the behavior takes place; and *Time Interval* is the time duration within which the behavior is recovered.

Common Behavior

Intuitively, frequently occurring user behaviors that are discovered from a large event dataset are reasonable to be regarded as user common behaviors. In other words, if a specific behavior is repeatedly performed by a group of people, most probably it is a common behavior. Also, given a large dataset of events, we can expect to discover a collection of common behaviors. The actor of a behavior is extracted from the actor attributes of events to categorize the behaviors. The context of a behavior is extracted from contextual attributes of the events to determine the neighborhood. The sequence of a behavior is extracted from behavioral attributes to explain user's behavioral characteristics. The time interval of a behavior is extracted from the time constraints.

Outlier Behavior

As discussed in subsection *Behavior*, an actor of behavior can be an individual or a group of people that have the same behaviors. We are interested in exploring the common behaviors of individuals or among a group of people. If an individual performs quite differently from his previous behavior, his current behavior is an outlier. If a person is categorized by role, he is supposed to perform similarly with the people who are assigned the same role. If a person has a collection of neighbors who are sharing the same context, he is expected to behave similarly with these neighborhoods. Compared with the anomaly categories discussed at the beginning of section C, the outlier behaviors explored by our approach are contextual collective anomalies.

Dynamic Behavior

The knowledge of extracted common behavior is sent to Security Agent. Security Agent monitors user's dynamic behavior (runtime event traces) and compares it with this user's previous behavior, and with common behaviors of similar actors in specific contexts. Given an outlier degree threshold, the dynamic behavior that is dissimilar to the actor's previous behavior or dissimilar to any common behavior is defined as outlier. Outlier behavior may be abnormal behavior, or maybe not, which requires system administrator's final determination.

D. Common Behavior Mining

Discovering common behavior patterns in a large event dataset (in the range of several hundreds of thousands or millions of events) is a hard problem and sometimes infeasible. To tackle this problem, we partition the search space (event dataset) into clusters of similar events based on their shared attributes using association mining operation. We operate an association mining engine on the event dataset to extract the shared attributes among events. Such shared attributes constitute the contexts of different common behaviours. The association mining engine receives a threshold value that we refer to as "*minsup-assoc"* (i.e., minimum support for association mining, with a value between 0% and 100%). Typically, such a search engine discovers many attribute-sets that occur frequently in the event dataset. A *frequent attribute set* is a collection of attribute values that appears in at least *minsup-assoc* events. Suppose 10% of events of the entire dataset occur at location "L-1" around 12:00 pm (represented as "T-12"). Given a threshold *minsup-assoc* 5%, association mining engine is capable of discovering the frequent attribute set <L-1, T-12> and a collection of events that contain the attribute set. A combination of the number of shared attributes and the number of sharing events measures the similarity between those events. Such an association-based similarity is used for clustering highly related events under a certain context, where each cluster becomes a smaller search space for the next phase.

After a clustering phase, sequential pattern mining is applied on each cluster to extract the frequent behavior

sequences. The input to the sequential pattern mining engine is an event sequence dataset and a user-specified threshold "*minsup-seq"* (i.e., minimum support of sequential pattern mining, with a value between 0% and 100%), and the output is a list of frequent sequence patterns that occur in at least *minsup-seq* sequences within the sequence dataset. To perform the sequential pattern mining, we should convert the event dataset to sequence dataset where each sequence is a set of ordered events performed by the same user within one day. Therefore, the discovered frequent sequence patterns can be viewed as user's daily behavior. In the same way, we could explore user's hourly behavior, weekly behavior, and monthly behavior.

As the events within one cluster share rather similar association patterns, the extracted behaviors from one cluster present the common behaviors under similar contexts. The association patterns may: i) include actor attribute *User*; ii) include actor attribute *Role*; or iii) include no actor attribute. If the association pattern includes actor attribute *User*, all behaviors extracted from this cluster belongs to a specific user; other attribute values of the association pattern contribute to the context of the common behaviors. For example, a cluster collects highly related events that share association pattern <U-1, L-1>, so that all behaviors explored from this cluster are common behavior of user U-1 at location L-1(context). If the association pattern includes actor attribute *Role*, the behaviors extracted from this cluster are common behaviors shared among a group of people with the same role. For example, a cluster collects highly related events that share association pattern <R-1, L-1>, so that all behaviors explored from this cluster are common behavior of a group of people that are assigned role R-1 at location L-1(context). If the association pattern does not include any actor attribute, all the attribute values in the association pattern contribute to the context of the common behaviors. The actor of these behaviors can be anyone. For example, a cluster collects highly related events that share association pattern <T-1, L-1>, so that all behaviors explored from this cluster are common behavior at location L-1 around time T-1. Such behaviors have common characteristics under certain context, which are not determined by the privileges and responsibilities of the actors.

E. Formal Representation of Outlier Behavior Detection

First, we formally define the knowledge of common behaviors that are sent from Behavior Monitor to Security Agent. Let $B = \{B_1, B_2, \ldots, B_n\}$ be a set of discovered common behaviors. Let $B_i = \langle B_ia, B_ic, B_is, B_it \rangle$ be a common behavior, where B_ia is actor, B_ic is context, B_is is sequence, and B_it is time constraint. User's dynamic behavior is a trace of events of a specific user. Let $E = \{e_1, e_2, \ldots, e_m\}$ be an ordered event sequence of a single system user, where *Eu* represents the user of the event sequence. If the common behaviors *B* are daily behaviors, Security Agent performs the outlier detection operation once a day. *E* presents the collected events of user *Eu* within one day. A subsequence of *E* is represented as $E_{jk} = \{e_j, \ldots, e_k\}$, where $E_{jk} \subseteq$ E if there exists integers $1 \le j \le k \le m$.

The problem of finding outlier behaviors is defined as follows. Given a collection of common behaviors *B* and user's dynamic behavior E (observed user's event sequence during B_it), an outlier detector is designed based on the dissimilarity between *E* and *B*. Outlier behaviors are three types of observations: i) behave distinct different from his previous behavior; ii) behave quite different from people who have the same privileges and responsibilities; iii) behave quite different from others under certain context. Accordingly, the common behaviors are divided into three categories as shown in formula (2): *Bu* presents a collection of common behaviors of the same user *Eu*; *Br* presents a collection of common behaviors of people who are assigned the same role as *Eu*; *Bc* presents a collection of common behaviors under the same context shared by events in *E*.

$$\begin{aligned} &B = Bu \cup Br \cup Bc \\ &Bu = \{B_i \mid B_i \in B, B_ia = Eu\} \\ &Br = \{B_i \mid B_i \in B, Eu \in B_ia\} \\ &Bc = \{B_i \mid B_i \in B, B_ic \subseteq \text{Shared Contexts in } E\} \end{aligned} \quad (2)$$

If the observed dynamic behavior *E* is dissimilar to any category of the common behaviors, it is considered as an outlier behavior. The outlier degree of *E* is defined in (3):

$$outlier(B, E) = \max(outlier(Bu, E), outlier(Br, E), outlier(Bc, E)) \quad (3)$$

The outlier degree is defined based on the behavior similarity. Ideally, the dynamic behavior is expected to be exactly the same as one of the common behaviors. If the dynamic behavior *E* is quite similar to any common behavior in *B*, it is unlikely to be an outlier. Formula (4) presents the outlier degree of E, compared with each of the user's previous behavior. If the dynamic behavior is dissimilar to all of his previous behaviors, its outlier degree increases. The behavior similarity $sim(B_i, E)$ is normalized with values between 0 and 1. The outlier degree calculation method is the same for all common behavior categories, hence we can calculate $outlier(Br, E)$ and $outlier(Bc, E)$ using the same formula (4).

$$outlier(Bu, E) = 1 - \max_{B_i \in Bu}(sim(B_i, E)) \quad (4)$$

To compare dynamic behavior with each common behavior, a new behavior similarity metric is defined as (5):

$$sim(B_i, E) = \begin{cases} \dfrac{|LCS(B_is, E)|}{|B_is|} & \text{if } |B_ic|=0 \\ \max\limits_{\substack{\{E_{jk} | E_{jk} \subseteq E\} \\ \forall e_p \in E_{jk},\, e_{p|a} \supset B_ic}} \left(\dfrac{|LCS(B_is, E_{jk})|}{|B_is|}\right) & \text{if } |B_ic| \neq 0 \end{cases} \quad (5)$$

where the similarity between common behavior B_i and observed dynamic behavior *E* is determined by the Longest Common Subsequences (LCS) [16] length under certain context B_ic. There are two cases for common behavior context: i) no context defined in common behavior ($|B_ic|=0$): in this case behavior similarity is determined by LCS between dynamic behavior *E* and common behavior sequence B_is; and

ii) context is not empty in common behavior ($|B_ic|\neq 0$): in this case behavior similarity is determined by the maximum LCS between the subsequences of dynamic behavior $\{E_{jk} \mid E_{jk} \subseteq E\}$ and B_is; E_{jk} is a subsequence of E with each event e_p in E_{jk} shares the same context with common behavior B_ic ($e_{p|a}$ means comparing attributes of e_p with context of B_ic). E is considered as similar to common behavior B_i if they share longer subsequences under the same context. If the common behavior is defined under certain context, but the dynamic behavior does not occur at such context, comparing the similarity between them is unreasonable and meaningless.

The length of LCS is considered as a measure of the closeness of two sequences, which finds the maximum number of identical items in these two sequences (preserving the event order). Each element of the sequence may be an itemset, but the formula of LCS as (1) can only compare simple items rather than itemset. For example, a sequence of behavior about actions and accessed objects looks like *<<A-1, O-1> <A-2, O-1> <A-3, O-2>>*. The itemsets *<A-1, O-1>* and *<A-1, O-2>* are partially identical. We enhanced the LCS formula as (6), which allows comparing itemsets in sequence. Let $X=(x_1, x_2, \ldots, x_m)$ and $Y=(y_1, y_2, \ldots, y_n)$ be two sequences of lengths m and n, respectively. An element of the sequence, $x_i \in X$ and $y_j \in Y$, can be an itemset. Suppose the attribute values of an itemset (x_i and y_j) are ordered, such as all elements in sequence X and Y follows the order of *<Action, Resource, Location>*. For example, x_i = *<A-1, O-1, None>* and y_j = *<A-1, None, L-2>*. The problem of comparing two itemset x_i and y_j can be converted to the problem of $lcs(x_i, y_j)$. A common subsequence *cs* of x_i and y_j represented by $cs(x_i, y_j)$ is a subsequence that occurs in both sequences. $lcs\ (x_i, y_j)$ is a common subsequence of both sequences with maximum length. The length of $lcs(x_i, y_j)$ is denoted by $R(x_i, y_j)$. Solving $R(X, Y)$ is to determine the longest common subsequence for all possible prefix combinations of the two sequences X and Y. Let $r(i, j)$ be the length of the *lcs* of $(x_1, x_2, \ldots, x_i)$ and $(y_1, y_2, \ldots, y_j)$. Then $R(X, Y)$ can be defined recursively as following:

$$r(i,j)=\begin{cases} 0 & \text{if } i=0 \text{ or } j=0 \\ r(i-1,j-1)+\dfrac{R(x_i,y_j)}{\max(|x_i|,|y_j|)} & \text{if } R(x_i,y_j)>0 \\ \max\{r(i-1,j), r(i,j-1)\} & \text{if } R(x_i,y_j)=0 \end{cases} \quad (6)$$

Finally the outlier will be detected by comparing the outlier degree $outlier(B, E)$ in formula (3) with an outlier degree threshold δ. If the outlier degree is greater than a threshold δ, E is identified as an outlier and will be notified to system administrators. The system administrator makes the final decision to grant or deny the outlier behavior. Based on intensive training, Security Agent may acquire enough trust from the system administrators about the outlier detection, and then Security Agent can be configured to make the final decision without manual involvement.

IV. CASE STUDY

In this section, we present an end-to-end case study to examine our proposed approach.

A. Implementation

We developed a prototype implementation of the proposed approach and applied on a simulated legacy PACS system and DI-r. ClearCanvas [17] is an open source implementation of a PACS viewer. A User Agent is deployed on the workstation to assist the ClearCanvas viewer to render the authentication flow. Health information exchange open source (HIEOS) [18] is an open source implementation that is used to simulate a set of DI-r web service interfaces to retrieve images. A generic Security Agent is deployed in front of HIEOS to perform authorization flow. Security middleware and DI-r make an agreement about applicable authorization policies, authorization model, and event filtering criteria. Security Agent is trained based on the security middleware generated training knowledge to perform its tasks.

B. Online security services

Let us consider a scenario where a user intends to use a PACS viewer application to display a patient's diagnostic report that is stored at the DI-r. One applicable authorization policy in this case is "*Only physicians are allowed to view and change a patient's diagnostic reports; other healthcare staffs only have the privilege of viewing the patient's diagnostic reports.*" Identity Provider issues an assertion including the statement of user's role "physician" after authenticating the end user. Resource Provider supplies the resource type as "diagnostic report" and the resource owner as "patient". Authorization engine grants this access request after evaluating the applicable policies with attribute values.

C. Post security services

The system kept running over one month and the Behaviour Monitor component totally collected 3000 user access events from the DI-r. These events are parsed and converted into attributed events. Each event is described by the following attributes: "User(U), Role(R), Location(L), Operation(O), Resource owner(W), Resource(E), Date(D), Time(T)". Each attribute value is represented by a quantitative value (e.g., L-1 means location "Oshawa"; L-2 means location "Toronto"; R-1 means role "physician"; R-2 means role "nurse").

The Apriori algorithm [11] is applied on the attributed events for discovering highly associated groups of events, where all events in one group share the same set of attribute values. We refer to the group of events as *basketset* and the shared set of attribute values as *itemset*. We define an association-based similarity metric between two events, which encode both the size of basketset and the length of itemset. Figure 3 is a visualization of the relationship among events. This graph is generated by Gephi [19], an open source network analysis and visualization software package. The undirected graph edges illustrate the associations between events according to our defined similarity metric. Each node represents an event, and each weighted edge represents the similarity value between two events. The events are grouped into a few of clusters. Our approach allows an event being assigned to multiple clusters.

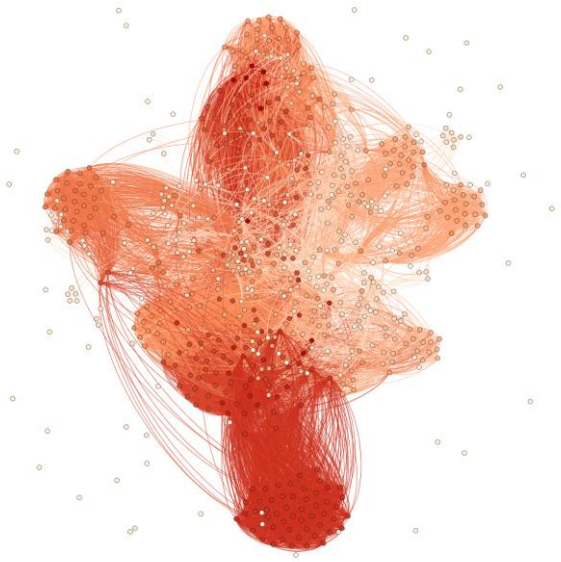

Fig. 3 Visualization of association between events

Sequential pattern mining algorithm CloSpan [20] is employed to discover user's daily behaviour in each cluster. First, we convert the event database into sequence dataset where each sequence is a set of ordered events performed by the same user within one day. Therefore, the discovered frequent sequence patterns can be viewed as the user's daily behaviour. In a post-analysis phase, we investigate the characteristics of the discovered sequence patterns in each cluster. For example: What is common among the users who accessed the system around the rush hour? What is the frequent behaviour pattern of a specific user in the system? Through analysing the common attribute values in each item of sequence patterns, context attributes are extracted to describe the circumstances of the complete sequence. The followings are some discovered behaviour patterns in the experiment:

- 50% of users have access requests at most 6 times during rush hour "10:00am".
- 80% of access requests from user "U-22" at location "L-6" are at time "1:00pm".

We can see the busiest time of user "U-22" is different from other users: "U-22" has more access request at 1:00pm but the normal rush hour is 10:00am. The system administrators may limit the maximum access request number during rush hour with differentiated policies. For example, an observed dynamic behaviour of user "U-22" is considered as outlier behavior if most access requests of user "U-22" is around "10:00am", because the dynamic behavior is changed from his pervious behaviour. In contrast, an observed dynamic behavior of user "U-23" is considered as outlier if most access requests of user "U-23" is around "3:00pm", because the dynamic behavior of user "U-23" is quite different from other users.

V. CONCLUSION

This paper contributes to the security and access control literature by proposing a common method for secure sharing medical images among legacy PACS systems and DI-r's. We have proposed a novel security middleware that replaces the existing trusted model for cross-PACS domains integration. Customizable and trainable software agents are deployed at the legacy systems to fulfil the authentication flow, to make authorization decisions as well as to collect user activities. In addition to the online security services, the security middleware provides post security services to recover user's access behavior patterns. We introduced a behavior model to represent behavior patterns. A variety of data mining techniques (i.e., association mining, sequence mining, and clustering) are applied to explore the user's common behavior. Furthermore, this research work proposed a new behavior similarity metric to measure the closeness between observed dynamic behavior and common user behaviors, and an outlier degree measurement to determine whether an observed dynamic behavior is outlier or not.

We plan to extend our work to provide step-by-step guidance throughout the whole policy enhancement process such as: i) investigating the characteristics of the extracted behavior patterns and committing recommendations to identify common behavior and abnormal behavior; and ii) detecting system security policy vulnerabilities and providing reasonable advice on policy consolidation.

REFERENCES

[1] Dossier Santé du Québec, "Diagnostic imaging group, Status of Diagnostic Imaging Repository (DI-r) projects across Canada". Available: http://www.camrt.ca/

[2] IHE IT Infrastructure Technical Framework Integration Profiles Volume 1. Available: http://www.ihe.net/, 2012

[3] IHE IT Infrastructure White Paper for Access Control, Available: http://www.ihe.net/, 2009

[4] Integration the Healthcare Enterprise website. Available: http://www.ihe.net

[5] N. Mehran, and K. Sartipi, "Modeling service representatives in enterprise systems using generic agents," *Service Oriented Computing and Applications (SOCA)*, vol. 5, pp. 245-264, Dec. 2011.

[6] K. Sartipi, K. Kuriakose, and W. Ma, "An Infrastructure for Secure Sharing of Medical Images between PACS and EHR Systems," *International Conference on Computer Science and Software Engineering (CASCON)*, pp. 245-259, 2013

[7] W. Ma, & K. Sartipi. Cloud-based Identity and Access Control for Diagnostic Imaging Systems. *In Proceedings of the International Conference on Security and Management (SAM) (p. 320). The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp)*, 2015

[8] W. Ma, K. Sartipi, H. Sharghi, D. Koff, and P. Bak. "OpenID connect as a security service in Cloud-based diagnostic imaging systems." *In SPIE Medical Imaging,. International Society for Optics and Photonics*, pp. 94180J-94180J, 2015.

[9] W. Ma, and K. Sartipi, "An Agent-Based Infrastructure for Secure Medical Imaging System Integration," *in Computer-Based Medical Systems (CBMS), 2014 IEEE 27th International Symposium on*, pp. 72-77. IEEE, 2014.

[10] M. H. Yarmand, and K. Sartipi, and D. G. Down, "Behavior-based access control for distributed healthcare systems," *Journal of Computer Security*, 21.1, pp. 1-39, 2013

[11] A. Rakesh, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," *in ACM SIGMOD Record*, vol. 22, no. 2, pp. 207-216. ACM, 1993.

[12] A. Rakesh, and R. Srikant, "Mining sequential patterns," *in Proceedings of the Eleventh International Conference on. IEEE*, pp. 3-14. IEEE, 1995.

[13] Aggarwal, C. C., and Reddy, C. K. (Eds.). *Data clustering: algorithms and applications*. CRC Press, 2013.

[14] Xu, R., and Wunsch, D. 2005. Survey of clustering algorithms. *Neural Networks, IEEE Transactions on*, 16(3), 645-678.

[15] Chandola, V., Banerjee, A., & Kumar, V. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3), 15, 2009

[16] Bergroth, L., Hakonen, H., and Raita, T. 2000. A survey of longest common subsequence algorithms. *In String Processing and*

Information Retrieval, 2000. SPIRE 2000. Proceedings. Seventh International Symposium on (pp. 39-48). IEEE.

[17] Open Source ClearCanvas PACS Website. [online]. Available: http://www.clearcanvas.ca/

[18] Open Source HIEOS Website. [Online]. Available: http://sourceforge.net/projects/hieos/

[19] Gephi - The Open Graph Viz Platform Website. [Online]. Available: http://gephi.github.io/

[20] X. Yuan, J. Han, and R. Afshar, "CloSpan: Mining closed sequential patterns in large datasets,"i*In Proceedings of SIAM International Conferen*Figures

Weina Ma was born in Baoding/China, in 1982. She obtained her B.Sc and M.Sc both in Computer and Software Engineering from Northwestern Polytechnic University in Xi'an/China, in 2005 and 2008, respectively. She started Ph.D study in Software Engineering in University of Ontario Institute of Technology in Oshwa/Canada from 2013. Her major research interests are knowledge engineering and data mining, eHealth services, and high performance computing and cloud computing.

Karmran Sartipi received B.Sc and M.Sc in Electrical Engineering from University of Tehran, and MMath and Ph.D in Computer Science (Software Engineering) from University of Waterloo. Dr. Sartipi has over 70 publications in Computer Science with focus on Information Security, Software and Knowledge Engineering, Data Analytics, and Medical Informatics. He has supervised more than 30 graduate students in inter-disciplinary fields, and developed several software tools in different scientific areas. He has collaborated with researchers in computer science, health science, business, and entrepreneurship fields for several years.