國立交通大學

資訊科學與工程研究所

碩 士 論 文

以 SAML 及區域屬性為基礎的權限控制

Access control by location attribute base on SAML

研 究 生：朱伯昕

指導教授：葉義雄　教授

中 華 民 國　九 十 六　年 七 月

以 SAML 及區域屬性為基礎的權限控制

Access control by location attribute base on SAML

研 究 生：朱伯昕　　Student：Po-Hsin Chu

指導教授：葉義雄　　Advisor：Yi-Shiung Yeh

國 立 交 通 大 學

資 訊 科 學 與 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Institute of Computer Science and Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

June 2006

Hsinchu, Taiwan, Republic of China

中華民國九十五年六月

Access control by location attribute base on SAML

Student: Po-Hsin Chu Adviser: Prof. Yi-Shiung Yeh

Institute of Computer Science and Engineering
National Chiao Tung University

Abstract

There are many web services on Internet. In general used cases, users need to be authenticated and authorized by a server before a server providing services. There may be many levels of authorities for accounts in a system and they are necessary to provide a scheme to manage the authorities of accounts for management purposes. In this thesis, we propose a method to solve this problem by using SAML, which is announced and supported by W3C alliance, as an open standard base on XML. By using SAML, people can give account different authorities that are decided by the attributes from user's location (or other relative information). Also people would get benefits of SAML for Single sign on by building a loosely-coupled system.

致謝

能完成這份論文，首先謝謝我的父母，沒有他們的鼓勵與支持，我不會進入交通大學碩士班就讀。

另外也非常感謝我的指導老師葉義雄教授，在這短短的兩年內，除了給我知識及生活上許多寶貴的意見，也教導了我面對問題時所需要的正確態度。葉教授是個十分照顧學生的好老師，也十分熱衷於研究工作，能有這樣的好老師當我的指導教授，讓我獲益良多。老師於 2007 年 7 月 6 日，由於腦溢血離開了人世，也願他能在另一個世界好好的休息。

除了老師之外，也要感謝鎮宇學長、靜雯學姐與定宇學長在這段期間的幫助，在我遇到思考的瓶頸時，提供我許多不同的參考意見，讓我能夠順利的完成我的論文。

也要感謝實驗室的同學雅婷與長蓉，時時提醒著我關於學校的注意事項，讓我知道重要的系辦規定而不會漏東漏西。

最後，要感謝我的女朋友怡君，讓我在倍感壓力時讓我能有稍微喘息的空間休息。

謝謝你們

朱伯昕

2007 年 7 月 8 日

Catalog

Chapter 1: Introduction

1.1 Motivation and purpose

With the improvement of Internet, network offers many kinds of web services to people. To manage access right of accounts is an important issue. Due to the identity's unique characteristic, it is not possible for a user having the same account on every website. Without single sign on, there might be a lot of accounts for users to handle and manage. By using single sign on, this problem is solved.

Also, with single sign on system, it will increase the willingness for customers to consume the services if related services ally to a bigger group. In this thesis, we use SAML (Security Assertion Markup Language) as a standard. SAML is an XML-based framework for user communicating authentication and attributing information. It allows the service entities to define their own assertion (which includes identity, attributes) to other entities.

This thesis provides an idea to control the access right by the information provided by user, transferred by the standard of SAML assertion. With the standard of SAML, it could be easily extended when building the single sign on system.

1.2 Structure of the thesis

There are five chapters in this thesis. The contents of each chapter are described as following:
Chapter 1 Introduction:
Chapter 1 describes the motivation and purpose of thesis, also the structure of the thesis.

Chapter 2 Background:
Due to the SAML, it is an XML-based framework structure, and there is a lot of relative XML knowledge is used in SAML, this chapter introduces these knowledge briefly, the comparison between SAML ver.1.1, and ver.2.0.

Chapter 3 SAML 2.0 schema architecture
This chapter describes SAML principle and the SAML 2.0 schema definition from W3C standard, which includes SAML assertions, SAML binding and SAML protocols defined in the form of XML Schema.

Chapter 4 System design
This chapter describes the system structure and algorithm, decision tree of this system and the flow chart of how user accesses the resource. The demos of implementation are also shown in this chapter.

Chapter 5 System implementation
This chapter describes the study case and implementation by the previous chapter. Demos of implementation are also shown in this chapter.

Chapter 6 Conclusion
This chapter will make a conclusion on this thesis and describes relative work and further work.

Chapter 2: XML-related knowledge

2.1 XML

XML (E**X**tensible **M**arkup **L**anguage) is a general-purpose markup language recommendation by W3C (World Wide Web Consortium) and it is announced in Feb, 1998. XML is not a programming language, but a system is used to define other languages. As a simplified subset of Standard Generalized Markup Language, its primary purpose that is to facilitate the sharing of data across different systems which are connected on the Internet. In other words, XML was designed to describe data, and the usage of data depends on different application. There are many languages are based on XML, such as RSS, MathML, GraphML, XHTML and SAML.

Below is a simple example of XML document:

```
<?xml version=”1.0” encoding=”UTF-8”>
<book ISBN=”0131488940”>
<name>E-Learning</name>
<price>650</price>
<author>Johnny D</author>
</book>
```

Figure 2.1 XML document example

The first line in the document defines the XML version, and the character encoding is used in the document. The next line describes the root element of this document, which has the tag name “book”. For every XML document, there must be a root element, and all other elements must be within this element. The tag of XML document is not predefined, so the designers for their own tags must define it. The next three lines describe three child elements of the root element. XML elements can have attributes. For example, the book element has an attribute “ISBN”, and its value is “0131488940”. The attribute’s name is not predefined in a legal XML document, and the attribute values must be quoted. The last line defines the end of the root element.

XML provides a text-based way and a tree-type structure to build the information. At the base level, all information shows as text and uses tags to separate the information into a hierarchy of data.

2.1.1 XML Schema

XML Schema is an XML-based language used to describe the structure of an XML documents, its purpose is to define the legal building blocks of an XML file, just like DTD. It offers programmers to define their own XML structure to build up their system. In more detail, it defines elements, attributes that can appear in a document; also which elements are child elements, and element orders, numbers, data type of element.

The advantage of XML Schema compares with DTD is XML Schema uses XML syntax, and supports data types and namespaces. As these reason, XML Schema is more extensible and is easily to be parse by XML parser. XML Schema became a W3C recommendation at 2001. Below is an XML Schema mapping to previous XML document.

```
<?xml version=”1.0”?>
<xs:schema xmlns:xs=”http://nctu.edu.tw/cryptio”>
<xs:element name=”book”>
<xs:complexType>
<xs:sequence>
<xs:element name=”name” type=”xs:string”/>
<xs:element name=”price” type=”xs:string”/>
<xs:element name=”author” type=”xs:string”/>
</xs:sequence>
<xs:attribute name =”ISBN” type=”xs:string”>
</xs:complexType>
</xs:element>
</xs:schema>
```

Figure 2.2 XML schema Example

Figure 2.2 shows previous example’s element type that is defined by XML Schema. It is a complex type because it contains other elements. Child elements such as name, price and author are simple types because they do not have child elements.

Figure 2.3 is the list of all elements that are used in XML schema and Figure 2.4 is the list of all data types in XML schema. The reason why we use XML Schema but not use DTD is because XML Schema is supported by W3C as the standard of XML description language.

Element	Explanation
all	Specifies that the child elements can appear in any order. Each child element can occur 0 or 1 time
annotation	Specifies the top-level element for schema comments
any	Enables the author to extend the XML document with elements not specified by the schema
anyAttribute	Enables the author to extend the XML document with attributes not specified by the schema
appInfo	Specifies information to be used by the application (must go inside annotation)
attribute	Defines an attribute
attributeGroup	Defines an attribute group to be used in complex type definitions
choice	Allows only one of the elements contained in the <choice> declaration to be present within the containing element
complexContent	Defines extensions or restrictions on a complex type that contains mixed content or elements only
complexType	Defines a complex type element
documentation	Defines text comments in a schema (must go inside annotation)
element	Defines an element
extension	Extends an existing simpleType or complexType element
field	Specifies an XPath expression that specifies the value used to define an identity constraint
group	Defines a group of elements to be used in complex type definitions
import	Adds multiple schemas with different target namespace to a document
include	Adds multiple schemas with the same target namespace to a document
key	Specifies an attribute or element value as a key (unique, non-nullable, and always present) within the containing element in an instance document
keyref	Specifies that an attribute or element value correspond to those of the specified key or unique element
list	Defines a simple type element as a list of values
notation	Describes the format of non-XML data within an XML document
redefine	Redefines simple and complex types, groups, and attribute groups from an external schema
restriction	Defines restrictions on a simpleType, simpleContent, or a complexContent
schema	Defines the root element of a schema
selector	Specifies an XPath expression that selects a set of elements for an identity constraint
sequence	Specifies that the child elements must appear in a sequence. Each child element can occur from 0 to any number of times
simpleContent	Contains extensions or restrictions on a text-only complex type or on a simple type as content and contains no elements
simpleType	Defines a simple type and specifies the constraints and information about the values of attributes or text-only elements
union	Defines a simple type as a collection (union) of values from specified simple data types
unique	Defines that an element or an attribute value must be unique within the scope

Figure 2.3 XML schema elements

Constraint	Description
enumeration	Defines a list of acceptable values
fractionDigits	Specifies the maximum number of decimal places allowed. Must be equal to or greater than zero
length	Specifies the exact number of characters or list items allowed. Must be equal to or greater than zero
maxExclusive	Specifies the upper bounds for numeric values (the value must be less than this value)
maxInclusive	Specifies the upper bounds for numeric values (the value must be less than or equal to this value)
maxLength	Specifies the maximum number of characters or list items allowed. Must be equal to or greater than zero
minExclusive	Specifies the lower bounds for numeric values (the value must be greater than this value)
minInclusive	Specifies the lower bounds for numeric values (the value must be greater than or equal to this value)
minLength	Specifies the minimum number of characters or list items allowed. Must be equal to or greater than zero
pattern	Defines the exact sequence of characters that are acceptable
totalDigits	Specifies the exact number of digits allowed. Must be greater than zero
whiteSpace	Specifies how white space (line feeds, tabs, spaces, and carriage returns) is handled

Figure 2.4 XML schema data types

2.1.2 XML Encryption

XML Encryption provides end-to-end security for applications that needs the secure exchange of data in XML format. It provides the mechanism for the security requirements that are not covered by SSL:

- Encrypt part of the data being exchanged.
- Secure session between many parties.

XML Encryption is an encryption in XML format, which provides three types of encryption:

(1) Arbitrary data
(2) XML element
(3) XML element content

The result of Encryption is an XML Encryption <EncryptedData> element. The <EncryptedData> has the structure as Figure 2.5, where “?” denotes zero or one occurrence; “+” denotes one or more occurrences; “*” denotes zero or more occurrences; and the empty element tag means the element must be empty.

```
<EncryptedData Id? Type? MimeType? Encoding?>
<EncryptionMethod/>?
<ds:KeyInfo>
<EncryptedKey>?
<AgreementMethod>?
<ds:KeyName>?
<ds:RetrievalMethod>?
<ds:*>?
</ds:KeyInfo>?
<CipherData>
<CipherValue>?
<CipherReference URI?>?
</CipherData>
<EncryptionProperties>?
</EncryptedData>
```

Figure 2.5 <EncryptedData> structure

Below is an example of XML encryption, we could use this example tells the difference of encryption as arbitrary data, XML element or XML element content.

```
<?xml version=”1.0” ?>
<purchaseOrder>
<Order>
<Item>book</Item>
<ISBN>0131488940</ISBN>
<Quantity>2</Quantity>
</Order>
<Payment>
<CardName>Visa</CardName>
<CardID>4457-0124-5589-7123</CardID>
<ValidDate>08-08</ValidDate>
</Payment>
</purchaseOrder>
```

Figure 2.6 XML encryption example

(1)Encrypt as an arbitrary data:

In this way, we consider the XML file as an arbitrary data, the XML file will be transfer to the value in the <CipherValue> element. The MimeType attribute defines the data type, which is encrypted in <CipherValue> element.

```
<?xml version='1.0' ?>
<EncryptedData xmlns=”http://www.w3.org/2001/04/xmlenc#”
  MimeType=”text/xml”>
<CipherData>
<CipherValue>fqk65zaXM4bmpyfq7rGj1c/S1Lyw0<CipherValue>
</CipherData>
</EncryptedData>
```

Figure 2.7 XML encryption example – arbitrary data

(2)Encrypt an XML element:

This method encrypts parts of the original XML file, which is selected by user. The benefit of this method is to keep part of the data in secret and shows the information that could be recognized by the receiver. Below is an example to encrypt parts of the element in the example XML file, which encrypted the <Payment> element and it’s child element, and replaced by the <EncryptedData> element:

```
<?xml version=”1.0” ?>
<purchaseOrder>
<Order>
<Item>book</Item>
<ISBN>0131488940</ISBN>
<Quantity>2</Quantity>
</Order>
<EncryptedData
Type=”http://www.w3.org/2001/04/xmlenc#Element”
xmlns=”http://www.w3.org/2001/04/xmlenc#”>
<CipherData>
<CipherValue>3u1hui42ry1w4trwat3ug9i</CipherValue>
</CipherData>
</EncryptedData>
</purchaseOrder>
```

Figure 2.8 XML encryption example – XML element

(3)Encrypt XML element content:

The differences between encrypt XML element and XML element content is encrypt XML element content leaves the element tag and encrypt the content in the element.

2.1.3 XML Signature

XML Signature is a W3C recommendation, which defines the digital signature in XML syntax. The purpose of XML Signature is used to verify if the message is sent from current party. XML Signature can be applied to any digital content, including XML. An XML Signature may be applied to the content of one or more resources. Enveloped or enveloping signatures are over data with the same XML documents as the signature.

2.2 Single Sign On

Single-Sign-On (SSO) is a method of access control that enables a user to authenticate once then gain access of resources from multiple service providers. Before discussing the type of SSO, there are two roles that should be defined: Identity Provider and Service Provider. Identity Provider is a role which provides users to authenticate and also a Service Provider. Service Provider is a role which provides service to users. There are two message flows to start a Web Browser Single Sign On:

1. Identity Provider-initiated Single Sign On.
2. Service Provider-initiated Single Sign On.

For Identity Provider-initiated SSO, users visit Identity Provider to authenticate and links to Service Provider from Identity Provider. If the authentication is passed, Service Provider provides service to users. For Service Provider-initiated SSO, user access resource at Service Provider, Service Provider sends user to Identity Provider for authenticates. Below are two graphs which describe the steps of Identity Provider-initiated Single Sign On and Service Provider-initiated Single Sign On.

Figure 2.9 Service Provider initiated Single Sign On

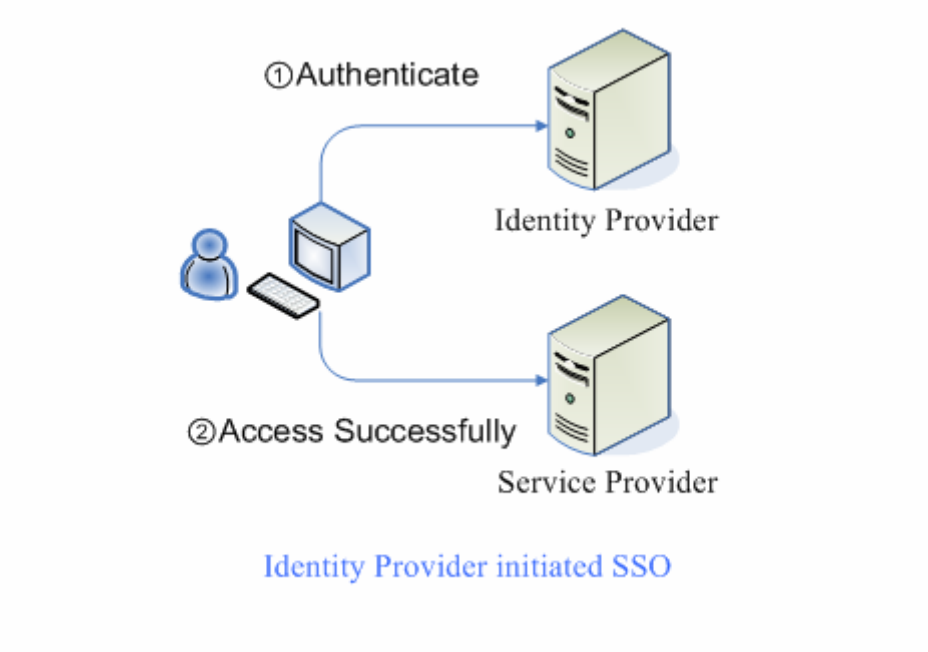


Figure 2.10 Identity Provider initiated Single Sign on

Chapter 3: SAML 2.0 schema

3.1 Introduction to SAML

The full name of SAML is Security Assertion Markup Language, it is a mechanism design by OASIS using for offering business parties having an XML structure to exchange authorization and authentication message.

SAML clearly defines many security assertions for different purposes: (1) Authentication Assertion (2) Attribute Assertion (3) Decision Assertion (4) Authorization Assertion, and using these assertions to improve the security of XML structure.

3.2 SAML Assertions

SAML assertions carry the information in XML statements about a principal that an asserting party claims to be true. Assertions are usually created by an asserting party which is based on a request of some sort from a relying party. The Assertion schema is defined below:

```
<complexType name="AssertionType">
        <sequence>
            <element ref="saml:Issuer"/>
            <element ref="ds:Signature" minOccurs="0"/>
            <element ref="saml:Subject" minOccurs="0"/>
            <element ref="saml:Conditions" minOccurs="0"/>
            <element ref="saml:Advice" minOccurs="0"/>
            <choice minOccurs="0" maxOccurs="unbounded">
                <element ref="saml:Statement"/>
                <element ref="saml:AuthnStatement"/>
                <element ref="saml:AuthzDecisionStatement"/>
                <element ref="saml:AttributeStatement"/>
            </choice>
        </sequence>
        <attribute name="Version" type="string" use="required"/>
        <attribute name="ID" type="ID" use="required"/>
        <attribute name="IssueInstant" type="dateTime" use="required"/>
</complexType>
```

Figure 3.1 Assertion schema

3.3 SAML Bindings

SAML Protocols are used to define how SAML Assertions are transported between participants. Binding is an instance of mapping protocol A to protocol B.

The protocols are defined within the SAML specification achieve the following actions:

- Returning one or more requested assertion.
- Performing authentication upon request and return the corresponding assertion.
- Registering a name identifier.
- Retrieving a message.
- Performing a single logout.

There are types of bindings. To classify them in usage from request message and response message would be:

Authentication Request message (Service Provider to Identity Provider):

1. HTTP Redirect Binding.
2. HTTP Host Binding.
3.HTTP Artifact Binding.

Response message (Identity Provider to Service Provider):

1. HTTP Hosting Bindings
2. HTTP Artifact Bindings.

3.4 SAML Protocols

The purpose of SAML Binding is to map the SAML protocols onto standard message and communication protocols. SAML protocol messages can be generated and exchanged using a variety of protocols. The protocols defined by SAML achieve the following actions:

- Returning one or more requested assertions. This can occur in response to either a direct request for specific assertions or a query for assertions that meet particular criteria.
- Performing authentication on request and returning the corresponding assertion.
- Registering a name identifier or termination a name registration on request.
- Retrieving a protocol message that has been requested by means of an artifact.
- Performing a near-simultaneous logout of a collection of related

session on request.

- Providing a name identifier mapping on request.

3.5 Identity Federation

Identity federation is the most different part between SAML 1.0 and SAML 2.0. Identity federation is a mechanism supported in SAML 2.0, which provides account linking between two sites with privacy protection

The principle of identity federation is using a pseudonym generated by Identity Provider to communicate between sites. The pseudonym will link user's identities in different site and communicate though the passing of pseudonym and the attachment of digital signature which proves the message is from Identity Provider.

An important part of this mechanism is the account linking while there's no pseudonym between two identities of the same user at different site. Figure 3.2 describes the steps of account linking between Identity Provider and Service Provider.

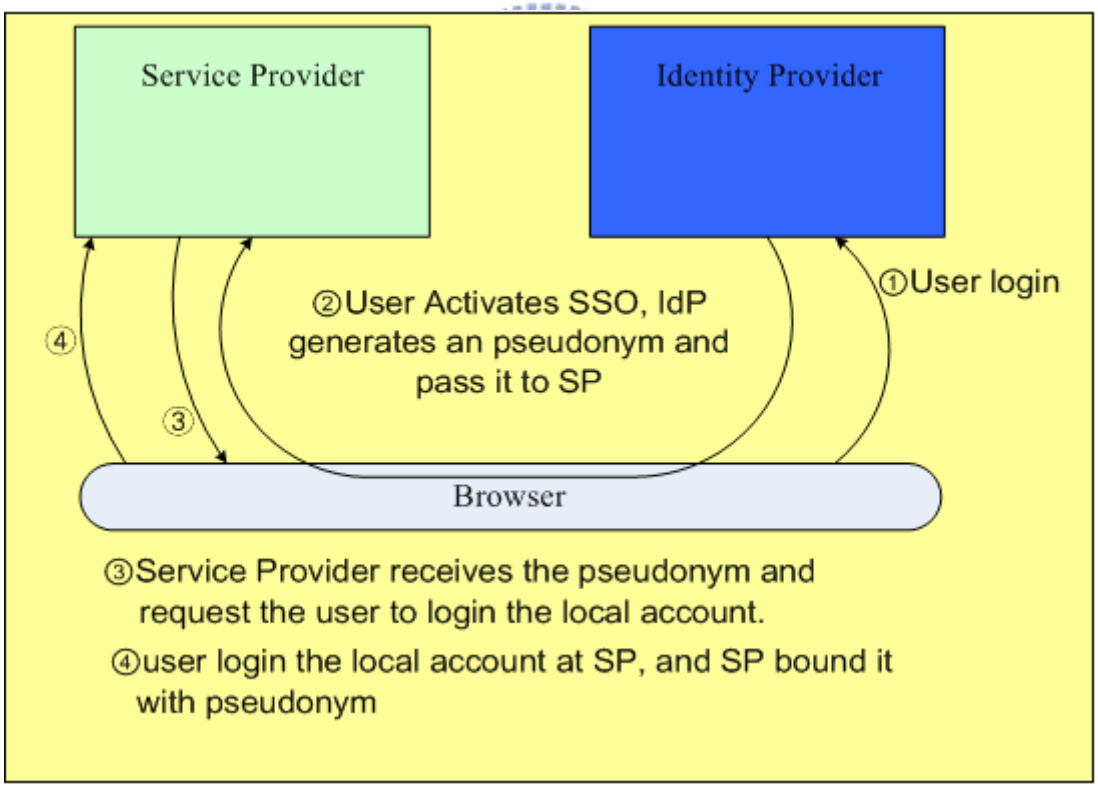


Figure 3.2 Account linking steps between IDP and SP

Chapter 4: System design

4.1 System introduction

As a web service system, the access control of accounts is very important. In different situation, it is necessary to limit the right of accounts due to some reason. The goal of the system design in this thesis is to design a system with access control on the accounts in different situation, due to the requirement of system. User login the account, and system decides the access rights of account due to the location that user offered.

The message passing through the component is base on SAML. With SAML, the system has the advantage what SAML has, for example Single Sign On. The access right of account is decided by the assertions pass to the service provider. When user found out there's an event corresponds to him, he will access the resource through the redirection from Event Manager to Service Provider. Event Manager will redirect the user with SAML assertion, while Service Provider decides the access rights of user from the assertion.

The assertion should include the pseudonym, location attribute from user, assertion available times and digital signature which shows the assertion is from Event Manager. The location attribute could be GPS coordinates, signals from sensor or IP address, due to the implementation.

4.2 Components in this system

4.2.1 Event Trigger

Event Trigger is the role who trigger event to the Event Manager. It could be humanity from home network, or a program receives the parameter from sensor to trigger the event. The task of Event Trigger is to parse the parameter from the environment we want to monitor and to decide whether the event should be trigger or not. If it does, then Event Trigger will send a message to Event Manager for further processing.

4.2.2 User

The role of user is to request the service from service provider. To be a user in the system implemented in this these, he has to login first, then polling the Event Manager Database, to check if there's any event occurs.

If there’s an event occur, user will be redirect to service provider with the assertion which includes the pseudonym from the database.

The polling could be replaced by other implementation, for example SMS system, and the platform could be replaced by hand-held system.

4.2.3 Event Manager

Event Manager is the main component in system. It is a service provider as well as an identity provider. It receives events from Event Trigger when event occurs and generates a pseudonym to databases. It handles the request from users and response user when event occurs.

When Event Manager receives the Event Occur message from Event Trigger, the Event Identity Provider in Event Manager will generate a pseudonym and store it to the corresponding entry in the database which decided by the parameter sent by Event Trigger.

After user known the event was occur and requesting the resource, Event Manager will redirect user to Service Provider with SAML assertion.

4.2.4 Service Provider

Service provider receives SAML assertion from Event Manager. The assertion includes the pseudonym and location attributes from user. Service Provider will decide whether the user have the right to access the resource or not by these parameter. User will be redirect back to Event Manager if the access checking failed.

4.3 Algorithms and decision flow

4.3.1 System architecture

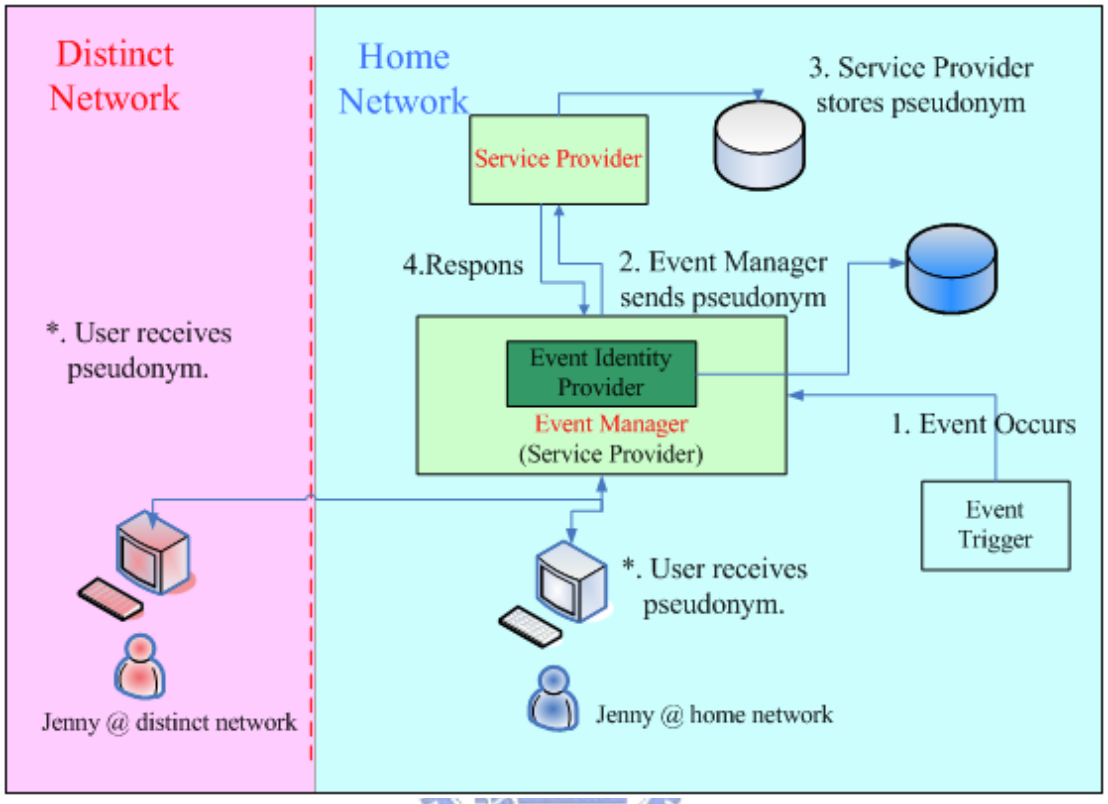


Figure 4.1 System architecture

Figure 4.1 is the system architecture. There are different areas in this system architecture. The one with the same area in Event Manager is called Home Network. The others we call them Distinct Network. Users from home network do not need a pseudonym to access the Service Provider since they are at the same area. Users from other network should need a pseudonym for service providers to access the resource they need.

The flow chart starts when Event Trigger actives the event. Event Trigger will send a message to Event Manager when event occurs. When Event Manager receives the event, the Event Identity Provider will generate a pseudonym and save it to the database in Event Manager. The pseudonym is used to link the event with the user info. At the same time, Event Manager will send a SAML assertion, which contains the pseudonym, trigger time, and other information to the Service Provider. Service provider receives this information, decides the assertion is available or not due to these information provided in the assertion. If it is

available, the service provider will add the pseudonym to the corresponding service information.

The user who uses the same account to login from different location will have different result. In home network, the user has the access right to access the service provided by service provider without providing a pseudonym. In distinct network, the access right is limited by the pseudonym, due to the location attribute provided by the Event Manager. The user from distinct network can only use the service corresponds to the pseudonym.

4.3.2 Interaction between user and system

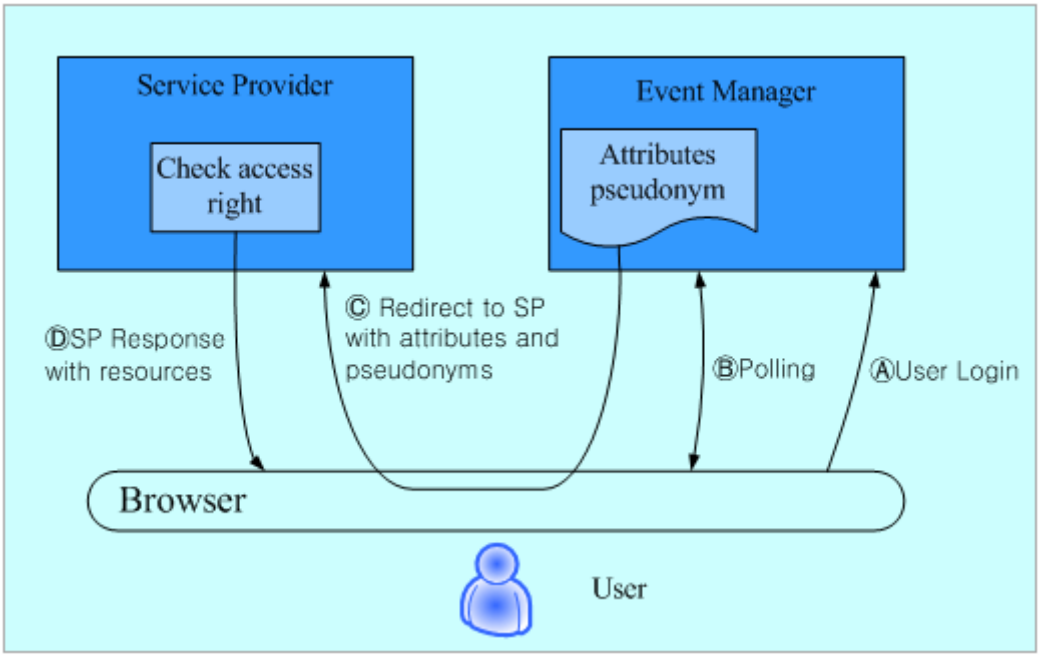


Figure 4.2 Interaction steps between SP and Event Manager

Figure 4.2 is the flow chart between users and Event Manager, Service Provider. Before accessing the service provider, user login from Event Manager. The polling action starts when user login Event Manager successfully. When the event occur, Event Manager will redirect user to service provider site with SAML assertion, and service provider will decide the access right for user due to the assertion provided by Event Manager. Below is the decision flow at Service Provider site.

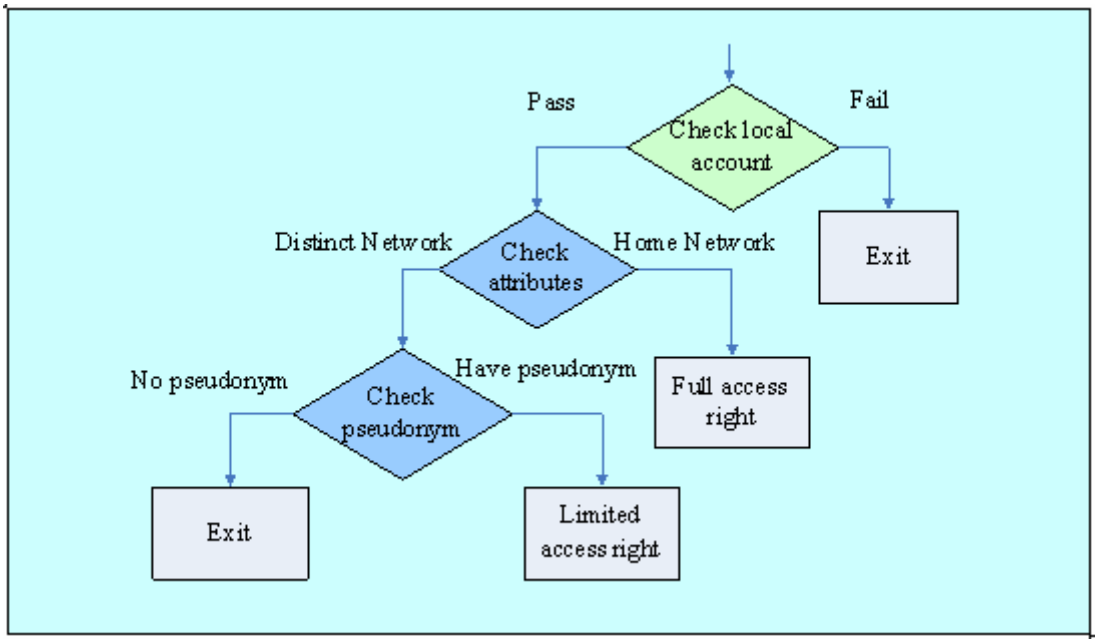


Figure 4.3 Decision Tree for access right

Figure 4.3 is the decision tree to decide whether service provider should provide the service to the user or not:

Step 1: The first step for user to do is to login local account at Event Manager. Event Manager will check whether the user has the right account and password or not. If he does, then there's a communication between user and Event Manager to inform the user whether the event is occur or not.

Step 2: This step starts when there is an event occurs, and user has been redirected to Service Provider site, starting access the resource. Service Provider will check the location attributes in SAML assertion from Event Manager to see whether the user is in home network or distinct network.

If the user is from home network, the user gets the full access right to get the information that service provider has.

Step 3: This step means for users is that they passed the account checking, but they are from distinct network. To limit their access rights, when there is a request from user, service providers have to check the assertions from Event Manager, whether there is an pseudonym in it or not. If there is a pseudonym exist corresponding the database in service provider, the user can only access the resource corresponds to the pseudonym but not all the resources in service provider.

Chapter 5: Apply LBAC to case history management system

5.1 Requirement Analysis of case history management system

In this chapter, we apply the location base access control system to a case history management system. Before the implementation, it's necessary to know the requirement of a case history management system. A case history management system stores the anamnesis of patients, and it should be able to accept the query and modification from doctors.

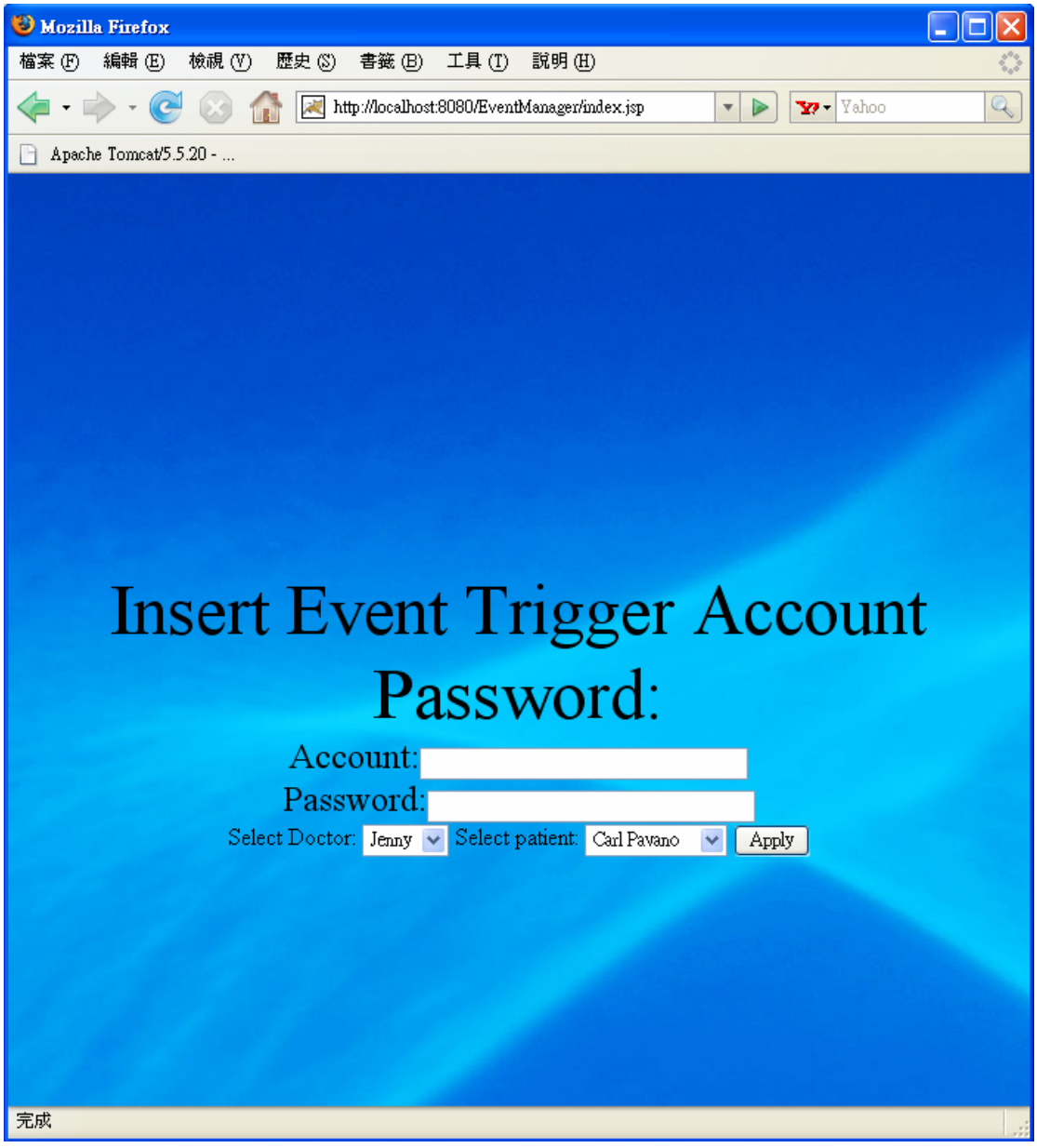
The reason why we add the access control to the user (doctor) is because we want to protect the privacy of patients. When the doctor is in hospital, it is necessary to have the full access rights to read the anamnesis of the patients. When doctor leaves hospital, we want to limit the doctor's right to access the patient's anamnesis due to the privacy reason. But it's necessary for a doctor to read the patient's case history when there's an emergency situation occurs, even the doctor is not in hospital. By this reason, when the emergency situation occurs, the Event Trigger (a nurse, in this case) will generate an event, and the doctor can access the specific patient's anamnesis through the pseudonym given by Event manager, even he's not in hospital.

5.2 LBAC case history management system implementation

This part displays the running of the program step by step with explanation. The system environment is Apache Tomcat version 5.5.20 with MySQL 5.0. Users access the system through the browser. The screen in blue background means it's at Event Manager, and the screen in red background means it's at Service Provider.

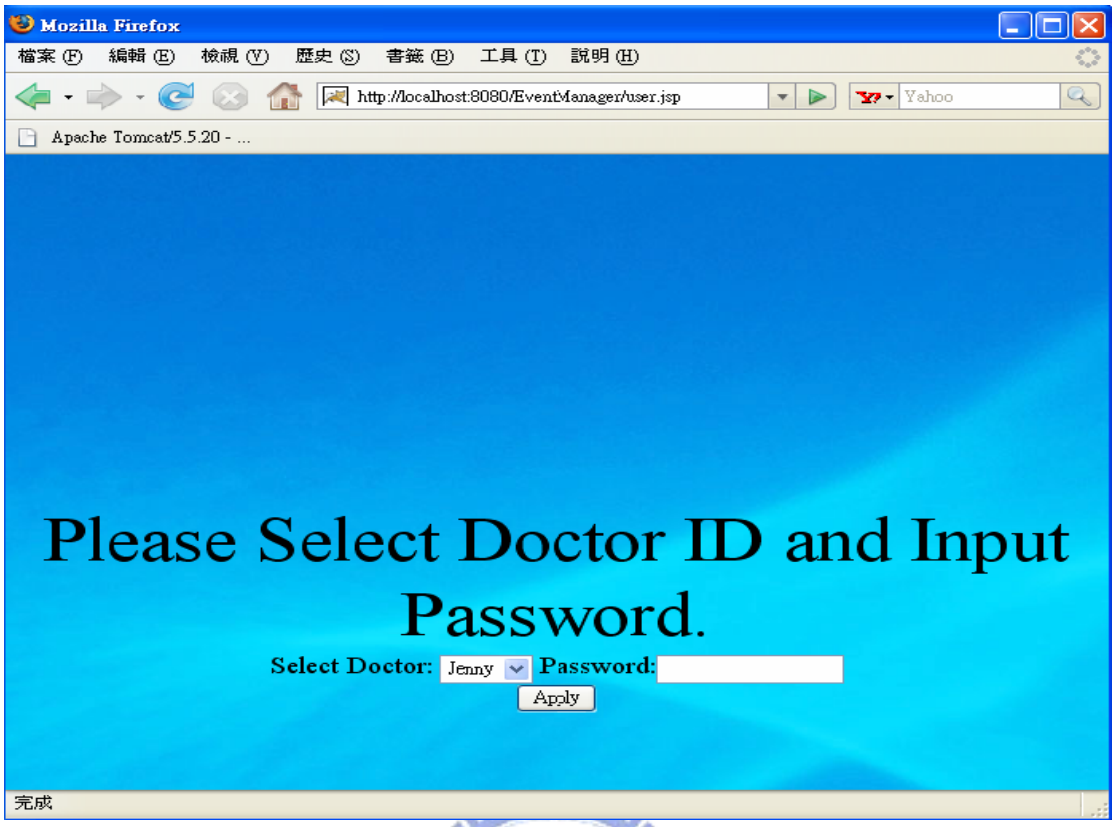
In this implementation, we use IP address as the attribute value of location information. Users connect from local host means the user from home network in Figure 8, and users from other IP address means the user from distinct network in Figure 8.

Below are screen shots from browser when system running.
Event Trigger Page: This page is for Event Trigger. The main function is to trigger an Event by selecting the doctor and patient.

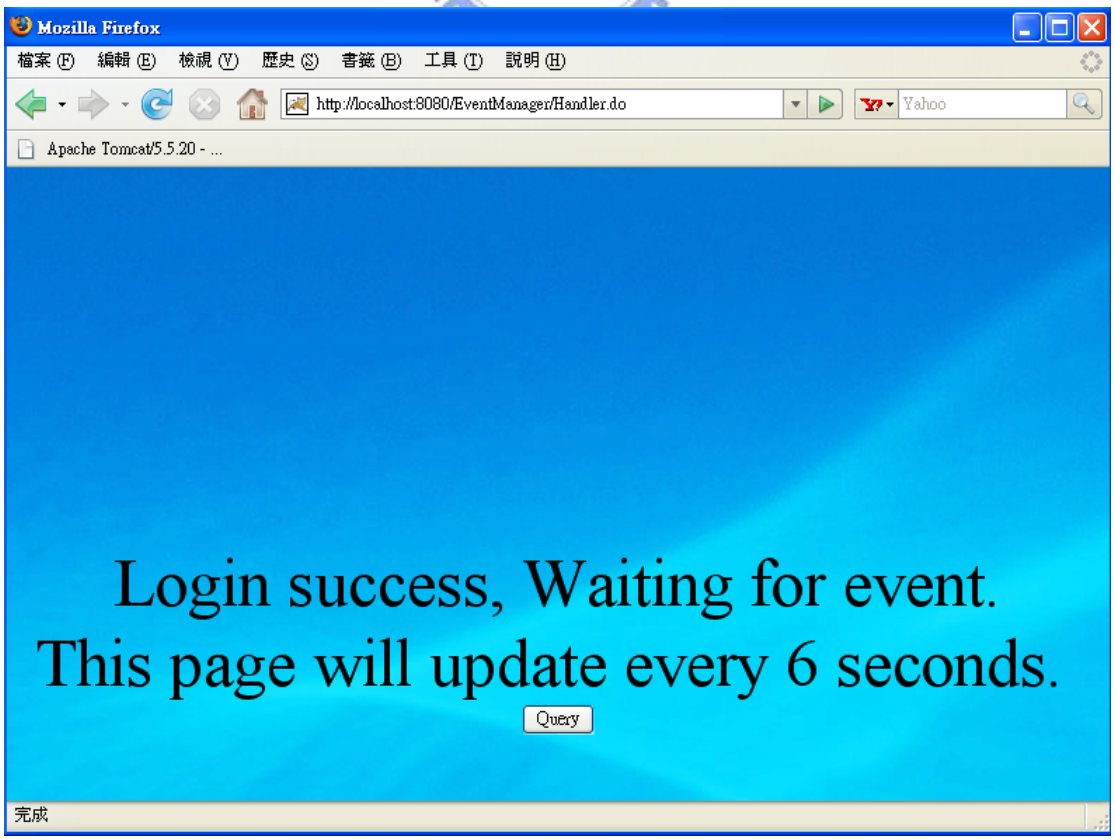


After insert the correct account and password of Event Trigger, selected the doctor name and patient name, the event will be send to Event Manager, which is not shown on browser. The page will be redirect back after the event is triggered.

User page-1: This is the page for user to login. User will be redirect to page 2 after enter the correct password.

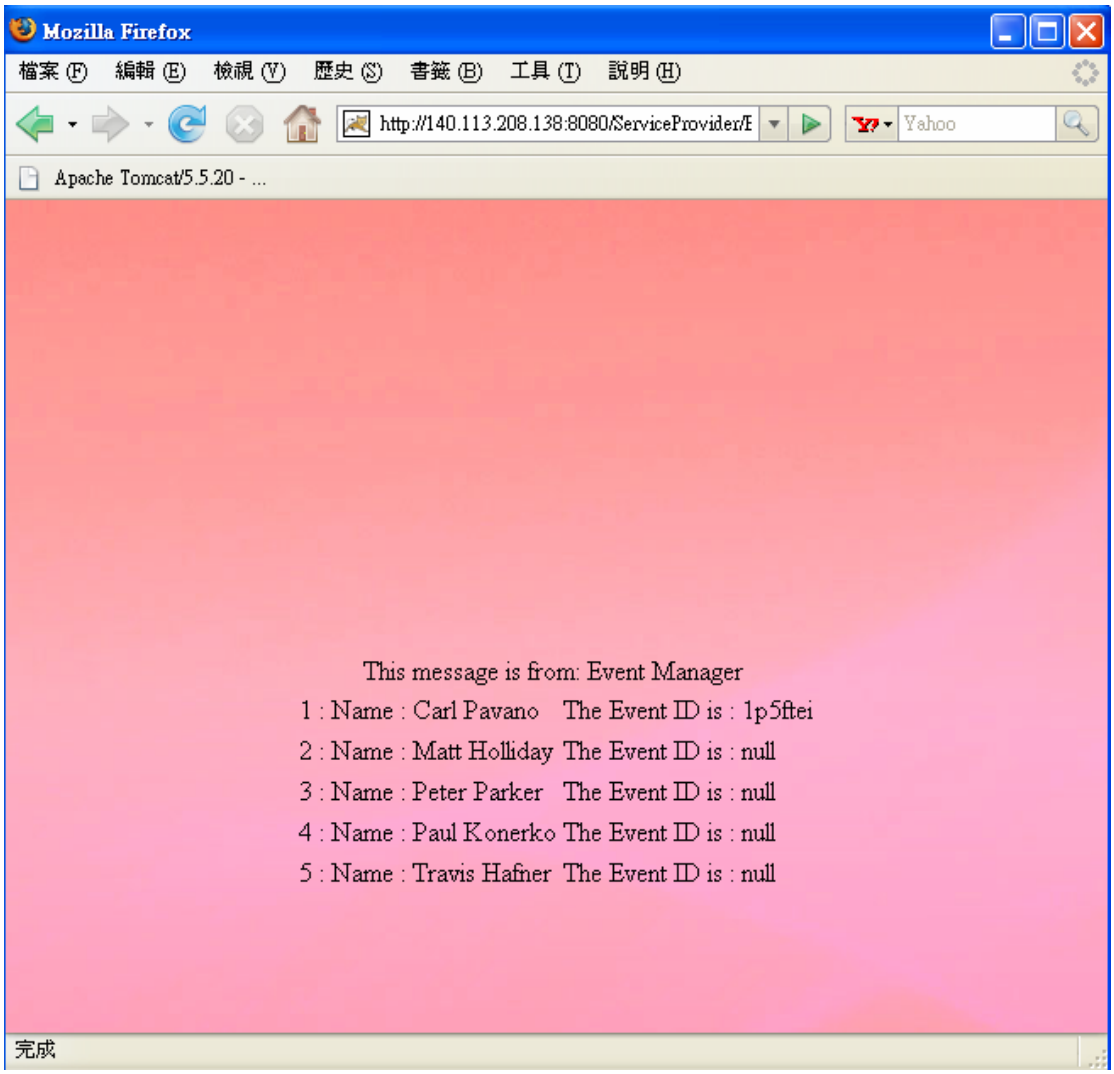


User Page 2: This page sends query to Event Manager every 6 seconds to check if there's any event corresponds to the user occurs or not. If event occurs, it will come out another button called "Get Event". The function Query button and "Get Event" is the same.

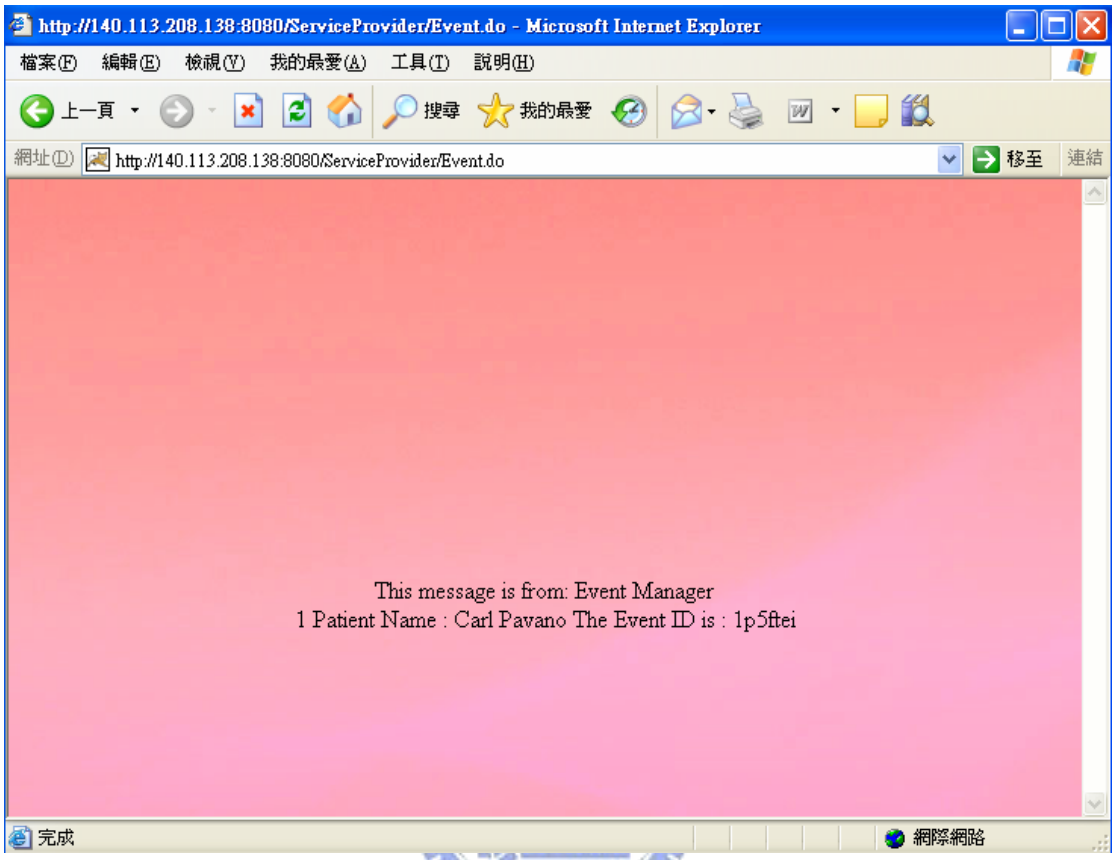


Service Provider Page: This page displays the result of Query.

The first figure is from home network. Though the user has only one pseudonym on one patient, due to the location attribute is from Local address, service provider lists all patients' name.



The second figure is from distinct network. Since the user has only one pseudonym, he can only get the patient's name which corresponding to the pseudonym.



Chapter 6: Conclusion and future work

Access control is an important issue in a web service system. The access control base on location attribute provides the system designer a flexible mechanism on account management and there's nothing different from user's view. In this thesis we use SAML assertion to transform the message. The benefit of using SAML in this system is with the standard of SAML, it's easy to transfer message through different site and easy to recognized through the digital signature in the assertion.

Another benefit of this system is when there's an integrating between two or more unions, it's easy to build the communication due to the currency of SAML which is base on XML standard.

In this thesis, we treat the location information (which could be a signal from wireless sensor or GPS) as a parameter due to the different usage. This information is provided from user's environment, and must not be modified by user. If it's a medical system described in chapter 5, it could be carry out by placing the sensor in hospital building. If the user's location is around the hospital and the handset device receives the signal from sensor, it's recognized as a user from home network.

As the future work, we would like to study about SAML usage on handheld device, and the integration of unions according to the attributes which contains location attribute by SAML assertion.

Appendix

This appendix includes the java code we used to implement the system in chapter 5.

A.1 Event Manager

A.1.1 Event Manager Page for Event Trigger: (index.jsp)

```
<%@ page import="java.io.*" %>
<%@ page import="java.sql.*"%>
<%@ page import="java.util.*"%>
<body BACKGROUND="bg.jpg">
<%for(int j=0;j<13;j++){out.println("<br>");}%>
<center>
<%String _username = request.getParameter("username");
    String _password = request.getParameter("pw");
    if((_username==null)&&(_password==null)){%>
    //Initial page, no user id and password insert
    <font size="20">Insert Event Trigger Account Password:</font>
    <br>
    <form action="index.jsp" method="post">
    <font size="5">Account:
    </font><input type="text" name="username" size="30">
    <br>
    <font size="5">Password:</font>
    <input type="password" name="pw" size="30"><br>
<%
      try
        {//Database Connection
          Class.forName("com.mysql.jdbc.Driver").newInstance();
          String dbuserName = "root";
          String dbpassword = "crypto123";
          String url="jdbc:mysql://localhost:3306/saml";
          //Database name and address
          Connection con=DriverManager.getConnection(url, dbuserName,
          dbpassword);
          System.out.println("Database connection establish");
          java.sql.Statement stmt = con.createStatement();
```

```
            ResultSet rs = stmt.executeQuery("select DoctorID,Name from
            DoctorInfo");
            out.println("Select Doctor:");
            out.println("<select NAME=doctor>");
            while(rs.next()){
                out.println("<option>");
                out.println(rs.getString("Name"));
                }
            out.println("</select>");
            rs = stmt.executeQuery("select Name from patientinfo");
            out.println("Select patient:");
            out.println("<select NAME=Patient>");
            while(rs.next()){
                        out.println("<option>");
                        out.println(rs.getString("Name"));
                            }
            out.println("</select>");
            }
             catch (Exception e)
                    {
                        System.err.println("cannot connect to SAML
                        database!");
                     }
                    %>

             <input name="Apply" value="Apply" type="submit"></form>
             <%
           }
else if(_username.equals("Trigger")&&(_password.equals("cx1346"))){
      //Event Trigger recognized by ID:Trigger Password:cx1346
      String pseudonym="";
      String LookupID=request.getParameter("doctor");
      String PatientID=request.getParameter("Patient");
      String str="abcdefghijklmnopqrstuvwxyz1234567890";
      for(int i=0;i<=6;i++){
            pseudonym+=str.charAt((int)(Math.random()*(36)));
               }//pseudonym generation
      try
```

```
{//database connection
   Class.forName("com.mysql.jdbc.Driver").newInstance();
   String dbuserName = "root";
 String dbpassword = "crypto123";
 String url="jdbc:mysql://localhost:3306/saml";
 Connection con = DriverManager.getConnection (url, dbuserName,
 dbpassword);
 System.out.println("Database connection establish");
 java.sql.Statement stmt = con.createStatement();
 ResultSet rs1 = stmt.executeQuery("select Name,EventID from
 DoctorInfo");
 while(rs1.next()){
      if(LookupID.equals(rs1.getString("Name")))
            {
              if(rs1.getString("EventID")==null){
              //If the doctor do not have Event
            String SQLcommand1 ="update DoctorInfo set EventID=
            '"+pseudonym+"' where Name='"+LookupID+"'";
            String SQLcommand2 ="update patientinfo set EventID=
            '"+pseudonym+"' where Name='"+PatientID+"'";
            stmt.executeUpdate(SQLcommand1);
            stmt.executeUpdate(SQLcommand2);
            out.println("Event ID:");
            out.println(pseudonym);
            out.println(" has been added into Database.<br>");
            out.println("Doctor :");
            out.println(LookupID);
            out.println("<br>");
            out.println("Patient :");
            out.println(PatientID);
                }//If the doctor does not have Event    “End”
                else{//The doctor already has event waiting for handle
                out.println("<br><font size=\"5\">
                The doctor has one Event waiting for handle</font>");
                out.println("<br><font size=\"5\">
                Please choose another doctor</font>");
                response.setHeader("Refresh","5;URL=index.jsp");
                }
```

```
                    //The doctor already has event waiting for handle    “END”
                   }
          }
        }//try end
        catch (Exception e)
        {
          System.err.println("cannot connect to SAML database!");
        }
        out.println("<font size=\"5\"><br>Redirect back in 5
        seconds...</font>");
        response.setHeader("Refresh","5;URL=index.jsp");
        }//Return to index
else{
      out.println("<font size=\"8\">Password or account not
      valid !</font>");
      response.setHeader("Refresh","3;URL=index.jsp");
      }
%>
</center>
</body>
```

A.1.2 Event Manager Page for User: (user.jsp)

```
//This page receives user’s input and check the password. The
information will be send to Servlet auth.class
<%@ page import="java.io.*" %>
<%@ page import="java.sql.*"%>
<%@ page import="java.util.*"%>
<html>
<body BACKGROUND="bg1.jpg">
<%for(int j=0;j<13;j++){out.println("<br>");}%>
<center>
<font size="20">
Please Select Doctor ID and Input Password.
</font>
<form action="Handler.do" method="post">
<%//Handler.do could be found at web.xml in WEB-INF folder, it maps
    //handler.do to auth.class
```

```
 session.invalidate();
 try
    {//Query database
     Class.forName("com.mysql.jdbc.Driver").newInstance();
     String dbuserName = "root";
     String dbpassword = "crypto123";
     String url="jdbc:mysql://localhost:3306/saml";
     Connection con = DriverManager.getConnection (url, dbuserName,
     dbpassword);
     System.out.println("Database connection establish");
     java.sql.Statement stmt = con.createStatement();
     ResultSet rs = stmt.executeQuery("select DoctorID,Name from
     DoctorInfo");
     out.println("<font size=\"4\">Select Doctor:</font>");
     out.println("<select NAME=doctor>");
     while(rs.next()){
              out.println("<option>");
              out.println(rs.getString("Name"));
                   }
     out.println("</select>");
    }//Try End
     catch (Exception e)
          {
           System.err.println("cannot connect to SAML database!");
          }
%>
<font size="4">Password:</font>
<input type="password" name="pw" size="15">
<br>
<input name="Apply" value="Apply" type="submit"></form>
</center>
</body>
</html>
```

A.1.3 Event Manager Servlet-1: (Auth.java)

```
//This servlet receives information from user.jsp and pass to polling.jsp
package com.SAML;
```

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.sql.*;

public class auth extends HttpServlet {
public void doPost(HttpServletRequest request,HttpServletResponse
response) throws IOException, ServletException{
        boolean auth = false;
        String doctorid=null;
        String password=null;
        HttpSession session = request.getSession();
        doctorid = request.getParameter("doctor");
        password = request.getParameter("pw");
        try
        {//Database connection
         Class.forName("com.mysql.jdbc.Driver").newInstance();
         String dbuserName = "root";
         String dbpassword = "crypto123";
         String url="jdbc:mysql://localhost:3306/saml";
         Connection con = DriverManager.getConnection
         (url, dbuserName,dbpassword);
          System.out.println("Database connection establish by auth");
        java.sql.Statement stmt = con.createStatement();
        //connect database done
        String query ="select name,password from doctorinfo";
        //SQL command
         ResultSet rs = stmt.executeQuery(query);
         while(rs.next()){
           if((doctorid.equals(rs.getString("name"))
             &&(password.equals(rs.getString("password"))))
               {
                 synchronized(session){
                   session.setAttribute("LoginResult","YES");
                   session.setAttribute("doctorid",doctorid);
                   session.setAttribute("password",password);
                   }
                 auth = true;
```

```
            }
          }//While done
        if(!auth){
                  session.setAttribute("LoginResult","NO");
                }
        rs.close();
        con.close();
       }
         catch (Exception e)
              {
               System.err.println("cannot connect to SAML database!");
              }

      RequestDispatcher view =
      request.getRequestDispatcher("polling.jsp");
      //post to polling.jsp
      view.forward(request,response);
      }


public void doGet(HttpServletRequest request, HttpServletResponse
response)throws IOException, ServletException{
      doPost(request,response);
      }
}
```

A.1.4 Event Manager Page for User: (polling.jsp)

```
//This page is for user polling the Event Manager and will update every
//6 seconds
 <html>
 <HEAD>
 <META HTTP-EQUIV="refresh" CONTENT="6">
 //Refresh Timer
 <BODY BACKGROUND="bg1.jpg">
 <%@ page import="java.io.*" %>
 <%@ page import="java.sql.*"%>
 <%@ page import="java.util.*"%>
 <%for(int j=0;j<13;j++){out.println("<br>");}%>
```

```
<%
   String doctorid = null;
   String password = null;

   String LoginResult =(String)session.getAttribute("LoginResult");
   if((LoginResult!=null)&&(LoginResult.equals("YES"))){
   doctorid =(String)session.getAttribute("doctorid");
   password =(String)session.getAttribute("password");
   %>

   <center><font size="8">Login success, Waiting for event.<br>
   This page will update every 6 seconds.
   </font><br>
   <form action="Event.do" method="post">
   <input name="query" value="Query" type="submit">
   </form>
   //This form will pass request to Event.do, which could be found at
   //web.xml, mapping to the Servlet "AssertionPass.java"
   <%try
           {
            Class.forName("com.mysql.jdbc.Driver").newInstance();
            String dbuserName = "root";
            String dbpassword = "crypto123";
            String url="jdbc:mysql://localhost:3306/saml";
            Connection con = DriverManager.getConnection
           (url, dbuserName, dbpassword);
           System.out.println("Database connection establish");
           java.sql.Statement stmt = con.createStatement();
           ResultSet rs = stmt.executeQuery("select Name,EventID from
           DoctorInfo");
           while(rs.next()){
               if(doctorid.equals(rs.getString("Name"))
              &&!(rs.getString("EventID")==null)){
               synchronized(session){
               session.setAttribute("pseudonym",rs.getString("EventID"));
               }
    %> //Executes when theres an pseudonym mapping to doctor id
        <form action="Event.do" method="post">
```

```
            <input name="query" value="Get Event" type="submit">
            </form>
      <%
                }
             }
            }
            catch (Exception e)
                  {
                        System.err.println("cannot connect to SAML
                        database!");
                   }

    }
  else{
     out.println("Login Fail");
     out.println("<br>");
     out.println("Password Error,Access Deny.");
     response.setHeader("Refresh","5;URL=user.jsp");
  }
  %>
  </center>
  </body>
  </html>
```

A.1.5 Event Manager Servlet-2 :(AssertionPass.java)

```
//this servlet is to accept the user's input from polling.jsp and save it to an
//XML format file for transform
  package com.SAML;
  import javax.servlet.*;
  import javax.servlet.http.*;
  import java.io.*;
  import java.net.*;
  import java.sql.*;
  import com.SAML.SamlAssertion;
  public class AssertionPass extends HttpServlet {
  public void doPost(HttpServletRequest request,HttpServletResponse
  response)throws IOException, ServletException{
  HttpSession session = request.getSession();
```

```
String pseudonym = (String)session.getAttribute("pseudonym");
String RemoteAddr = request.getRemoteAddr();
//call assertion generator
com.SAML.SamlAssertion Assertion =
new com.SAML.SamlAssertion();
String Ass =
Assertion.getSamlAssertion("Event Manager",pseudonym,RemoteAddr);
          //System.out.println(Ass);
   try {//Writing into XML file
          FileWriter Writer = new FileWriter("Assertion.xml");
          Writer.write(Ass);
          Writer.close();
        }catch(IOException ex){
                    ex.printStackTrace();
              }
session.invalidate();

response.sendRedirect("http://140.113.208.138:8080/ServiceProvider/Eve
nt.do");
//redirect user to Service Provider
}
public void doGet(HttpServletRequest request, HttpServletResponse
response)throws IOException, ServletException{
          doPost(request,response);
        }
}
```

A.1.6 Event Manager java class:(SamlAssertion.java)

```
//This class generates assertion in xml format
package com.SAML;
import java.io.*;
import org.apache.xml.security.utils.Base64;
import java.util.Date;

public class SamlAssertion {
//XML namespace content
private String samlLink="\"urn:oasis:names:tc:SAML:2.0:assertion\"";
private String samlpLink="urn:oasis:names:tc:SAML:2.0:protocol";
```

```
//three requirement attribute in Assertion
// " " is include in the parameter Content
private String Version="\"2.0\"";
//requirement elements

public SamlAssertion(){}
public String getSamlAssertion(String Issuer, String pseudonym, String
locationinfo){
//for Event Manager to SP use
String Assertion="<?xml version=\"1.0\" encoding=\"utf-8\"?>";
Date tmp=new Date();
String ID="\"";
String str="abcdefghijklmnopqrstuvwxyz1234567890";
    for(int i=0;i<=30;i++){
    ID+=str.charAt((int)(Math.random()*(36)));
    }
    ID+="\"";
//assertion attributes setting
    Assertion+="<saml:Assertion xmlns:saml="+samlLink;
    Assertion+=" ";

    Assertion+="Version="+Version;   //設定 version attribute

    Assertion+=" ";

    Assertion+="ID="+ID;//設定 Id attribute

    Assertion+=" ";
    Assertion+="IssueInstant=\""+tmp.toString()+"\">";
// assertion attribute end

//building Issuer Element;
    Assertion+=Issuer(Issuer);
//building Subject Element;
    Assertion+=Subject(pseudonym,1);
//building Condition Element, user the method without parameter
    Assertion+=Conditions();
    Assertion+=AttributeStatement("Location",locationinfo);
    Assertion+="</saml:Assertion>";
//building assertion end
```

```
            tmp=null;
            return Assertion;
        }
//this method is for generating the issuer element
public String Issuer(String Issuer){
        String Element="<saml:Issuer>";
        Element+=Issuer;
        Element+="</saml:Issuer>";
        return Element;
}

//this method is for generating subject element
//if this function not called, then there wont be any subject
//IDType 1,2,3 means the ID is BaseID, NameID, or EncryptedID
//IDType 0 means no ID selected

public String Subject(String Content, int IDType){
            String Element="<saml:Subject>";
//if this function is called, then string no longer empty;
//SubjectContent+="<saml:Subject>";

        if (IDType==1)
        {
            Element+="<saml:BaseID>";
            Element+=Content;
            Element+="</saml:BaseID>";
        }
        else if(IDType==2)
        {
            Element+="<saml:NameID>";
            Element+=Content;
            Element+="</saml:NameID>";
        }
        else if(IDType==3)
        {
                Element+="<saml:EncryptedID>";
            Element+=Content;
            Element+="</saml:EncryptedID>";
```

```
        }

        Element+="</saml:Subject>";
                return Element;
}

//saml:Condition
//ConditionType 1,2,3,4 means Condition,AudienceRestriction,
//OneTimeUse,ProxyRestriction
//ConditionType 0 means no Condition Element
public String Conditions(String Condition, int ConditionType){
//set condition available time attribute
//20 minutes
        Date NotBefore = new Date();
        Date NotOnOrAfter = new Date(NotBefore.getTime()+1200000);

        String Element="<saml:Conditions ";
                Element+="NotBefore="+"\""+NotBefore.toString()+"\""+"
";

Element+="NotOnOrAfter="+"\""+NotOnOrAfter.toString()+"\""+">";
        if(ConditionType==1)
        {
                Element+="<saml:Condition>";
                Element+=Condition;
                Element+="</saml:Condition>";
        }
        else if(ConditionType==2)
        {
                Element+="<saml:AudienceRestriction>";
                Element+=Condition;
                Element+="</saml:AudienceRestriction>";
        }
        else if(ConditionType==3)
        {
                Element+="<saml:OneTimeUse>";
                Element+=Condition;
                Element+="</saml:OneTimeUse>";
```

```
        }
        else if(ConditionType==4)
        {
            Element+="<saml:ProxyRestriction>";
            Element+=Condition;
            Element+="</saml:ProxyRestriction>";
        }
        Element+="</saml:Conditions>";
        return Element;
}

public String Conditions(){
//set condition available time attribute
//20 minutes
        Date NotBefore = new Date();
        Date NotOnOrAfter = new Date(NotBefore.getTime()+1200000);
        String Element="<saml:Conditions ";
        Element+="NotBefore="+"\""+NotBefore.toString()+"\""+"            ";
        Element+="NotOnOrAfter="+"\""+NotOnOrAfter.toString()+"\""+">
        ";
        Element+="</saml:Conditions>";
        return Element;
        }
public String AttributeStatement(String AttributeName, String
AttributeValue){
        String Element="<saml:AttributeStatement>";
                Element+="<saml:Attribute Name=\"";
                Element+=AttributeName;
                Element+="\"";
                Element+=">";
                Element+="<saml:AttributeValue>";
                Element+=AttributeValue;
                Element+="</saml:AttributeValue>";
                Element+="</saml:Attribute>";
                Element+="</saml:AttributeStatement>";
        return Element;
        }
}
```

A.1.7 Event Manager assertion server: (Server.java)

```
//This java program receives the request from service provider and pass
//assertion to service provider
import java.io.*;
import java.net.*;
class Server{
public static void main(String[] args){
   int counter = 0;
   try{
        ServerSocket serverSock = new ServerSocket(5568);
        while(true)
        {
          String input="";
          System.out.print("Waiting...");
          Socket sock = serverSock.accept();
          System.out.println("Get request:"+counter);
          File f = new File("Assertion.xml");
          FileReader FReader = new FileReader(f);
          BufferedReader BReader = new BufferedReader(FReader);
          String line ="";
          while((line = BReader.readLine())!=null)
          {      input=input+line;                    }
          BReader.close();
          PrintWriter PWriter = new PrintWriter(sock.getOutputStream());
          PWriter.println(input);
          PWriter.close();
          f.delete();
        }
       }catch(IOException ex){ex.printStackTrace();}
   }
}
```

A.2 Service Provider

A.2.1 Service Provider Servlet : (EventHandler.java)

```
package com.SAML;
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.net.*;
import java.sql.*;
import org.w3c.dom.*;
import javax.xml.parsers.*;

//this servlet is at Service Provider site and open an socket to connect the
//SocketServer and Receives the xml file from socket

public class EventHandler extends HttpServlet {
public void doPost(HttpServletRequest request,HttpServletResponse
response)throws IOException, ServletException{
        String Assertion = null;
        String Issuer = null;
        String Loc = null;
        String sEventID= "";
        try { //retrieve file
                Socket s = new Socket("140.113.208.138",5568);
                InputStreamReader streamReader =
                    new InputStreamReader(s.getInputStream());
                BufferedReader reader =
                    new BufferedReader(streamReader);
                Assertion = reader.readLine();
                FileWriter Writer = new FileWriter("SPAssertion.xml");
                Writer.write(Assertion);
                Writer.close();
                DocumentBuilderFactory factory =
                    DocumentBuilderFactory.newInstance();
                DocumentBuilder DocBuilder =
                    factory.newDocumentBuilder();
                File f = new File("SPAssertion.xml");
```

```
            //parsing xml file
            Document AssertionDoc = DocBuilder.parse(f);
            Node node = AssertionDoc.getDocumentElement();
            NodeList nodelist = node.getChildNodes();
            Issuer = nodelist.item(0).getFirstChild().getNodeValue();
            sEventID =
nodelist.item(1).getFirstChild().getFirstChild().getNodeValue();
            Loc =
nodelist.item(3).getFirstChild().getFirstChild().getFirstChild().getNodeVa
lue();
            HttpSession session = request.getSession();
            session.setAttribute("sIssuer",Issuer);
            session.setAttribute("sLoc",Loc);
            session.setAttribute("sEventID",sEventID);
            f.delete();
            }catch(Exception ex){
               ex.printStackTrace();
            }
        System.out.println("EventHandler Start");
        System.out.println(Issuer);
        System.out.println(Loc);
        System.out.println(sEventID);
        System.out.println("EventHandler Done");
        //print in system log for further checking
        RequestDispatcher view =
        request.getRequestDispatcher("handler.jsp");
        view.forward(request,response);
        //pass to handler.jsp
}

public void doGet(HttpServletRequest request, HttpServletResponse
response)throws IOException, ServletException{
        doPost(request,response);

        }
}
```

A.2.2 Service Provider Page : (handler.jsp)

this page is at service provider site , to display the information handled by Event.do

```
<body BACKGROUND="bg2.jpg">
<%@ page import="java.io.*" %>
<%@ page import="java.net.*"%>
<%@ page import="java.sql.*"%>
<%@ page import="java.util.*"%>
<%for(int j=0;j<13;j++){out.println("<br>");}%>
<center>
This message is from:
<%
   String Issuer = (String)session.getAttribute("sIssuer");
   String Attribute = (String)session.getAttribute("sLoc");
   String EventID = (String)session.getAttribute("sEventID");
   out.println(Issuer);
   %>
   <br>
   <%
   boolean isLocal = false;
   boolean noEvent = true;
   if(Attribute.equals("127.0.0.1")){ isLocal = true; }
   if(isLocal)
   {
      try{//this section executes when user from homenetwork
     Class.forName("com.mysql.jdbc.Driver").newInstance();
     String dbuserName = "root";
     String dbpassword = "crypto123";
     String url="jdbc:mysql://localhost:3306/saml";
    Connection con = DriverManager.getConnection (url, dbuserName,
    dbpassword);
    System.out.println("Database connection establish");
    System.out.println("Database connection establish by Service
    Provider");
    java.sql.Statement stmt = con.createStatement();
    String query ="select PatientID,Name,EventID from PatientInfo";
```

```
      ResultSet rs = stmt.executeQuery(query);
       out.println("<table>");
       out.println("<font size=\"5\">");
              while(rs.next())
                        {//print the result
                          out.println("<tr>");
                          out.println("<td>");
                          out.println(rs.getString("PatientID"));
                          out.println("</td>");
                          out.println("<td>");
                          out.println(": Name :");
                          out.println(rs.getString("Name"));
                          out.println("</td>");
                          out.println("<td>");
                          out.println("    The Event ID is :");
                          out.println(rs.getString("EventID"));
                          out.println("</td>");
                          out.println("</tr>");
                          }
                          out.println("</table>");
                          out.println("</font>");
              }
              catch (Exception e)
                {
                  System.err.println("Error Generate by isLocal TRUE part");
                 }
    }
    else{ //lookup by pseudonym
try{
       Class.forName("com.mysql.jdbc.Driver").newInstance();
       String dbuserName = "root";
       String dbpassword = "crypto123";
       String url="jdbc:mysql://localhost:3306/saml";
       Connection con = DriverManager.getConnection (url, dbuserName,
       dbpassword);
       System.out.println("Database connection establish");
       System.out.println("Database connection establish by Service
       Provider");
```

```
        java.sql.Statement stmt = con.createStatement();
        String query ="select PatientID,Name,EventID from PatientInfo";
        ResultSet rs = stmt.executeQuery(query);
        while(rs.next())
              {
                if(EventID.equals(rs.getString("EventID")))
                    { //get Event
                       noEvent = false;
                       out.println(rs.getString("PatientID"));
                       out.println("Patient Name :");
                       out.println(rs.getString("Name"));
                       out.println("    The Event ID is :");
                       out.println(rs.getString("EventID"));
                     }
               }
      }
        catch (Exception e)
              {
               System.err.println("Error Generate by isLocal FALSE part");
              }

  if(noEvent){//no access right
      out.println("You do not have Event ID, also not in Home Network. ");
      out.println("System Will redirect you to user login page.");
response.setHeader("Refresh","5;URL=http://140.113.208.138:8080/Eve
ntManager/user.jsp");
}

    }
    %>
</center>
</body>
```

Reference:

[1]Extensible Markup Language (XML) 1.0 (Fourth Edition)
http://www.w3.org/TR/xml/
[2]XML Encryption Syntax and Processing
http://www.w3.org/TR/xmlenc-core/
[3]”Debunking SAML myths and misunderstandings” by Frank Cohen
[4]OASIS, Authentication context for the OASIS Security Assertion Markup Language V 2.0, 2005
[5]OASIS, Conformance requirements for the OASIS Security Assertion Markup Language V2.0, 2005
[6]OASIS, Glossary for the OASIS Security Assertion Markup Language V2.0, 2005
[7]OASIS, Profiles for the OASIS Security Assertion Markup Language V2.0, 2005
[8]OASIS, Binding for the OASIS Security Assertion Markup Language V2.0, 2005
[9]OASIS, Assertion and Protocols for the OASIS Security Assertion Markup Language V2.0, 2005
[10]OASIS, Metadata for the OASIS Security Assertion Markup Language V2.0, 2005
[11]OASIS, Security Assertion Markup Language V2.0 Technical Overview, 2005