A Novel Access Control Mechanism based on Key-Chain-Web Model using Authorization Contexts

Vibhaj Rajan
Department of Computer Engineering
Institute of Technology, BHU, Varanasi

Subhash Chandra Patel
Department of Computer Engineering
Institute of Technology, BHU, Varanasi

Ravi Shankar Singh
Department of Computer Engineering
Institute of Technology, BHU, Varanasi

ABSTRACT
This paper proposes a number of useful improvements to the Key-Chain-Web access control mechanism which expands the usability of the mechanism in different scenarios. The improved services shall demonstrate the flexible and adaptive nature of the mechanism achieved through the use of relationships within co-ordination among resources in cloud and grid systems to provide access control. The proposed additions are very easy to implement and augments the fundamental principle of co-ordination based access control inherent in it. The proposed services are generic in nature to suit the access control needs of any distributed environment.

General Terms
Security, Internet, Cloud Computing, Enterprise Software, Service Oriented Architecture

Keywords
access control; key-chain-web; cloud grid systems; enterprise software; coordination

1. INTRODUCTION
The Key-Chain-Web [1] mechanism is an elegant model that captures relationships among co-ordinated resources to provide authorization delegation and access control. Challenging scenarios are found to exist for co-ordination amongst resources or objects with respect to specifications and realization of access control and co-ordination relationships. Key-Chain-Web model tried to simplify these issues without loss of flexibility in achievable co-ordination.

According to Ravi S Sandhu [2], the purpose of access control is to limit the actions or operations that a legitimate user of a system can perform. In a way, access control seeks to prevent activity that could lead to a breach of the system. Mandatory, discretionary and role-based [3] are the types of access control usually found in literature and implemented in systems. The co-ordination based model for access control was introduced by the innovative Key-Chain-Web mechanism which offered high flexibility and adaptability through use of operational semantics agnostic design, decentralized control and hierarchical access levels.

In this paper, we aim to add to the Key-Chain-Web mechanism features that were found to be very useful in access control during cloud project development and which enhances the flexibility and maintainability of the system. These features are plugged into the architecture by means of authorization and authentication contexts.

2. KEY-CHAIN-WEB MECHANISM
The Key-Chain-Web mechanism is based on three entities Users (keys), Resources (chains) and Relationships (webs). In addition, we have chain members representing relationship among users and resources, particularly in case of a collaborative environment where sharing may be achieved through members. Web relationships are used to automatically delegate rights for access among the users who are either owners or members of the resource chain.

Fundamental to this mechanism is the concept of authorization schemes which are of two types – resource scheme which operates on chains over keys and relationship scheme which operates on webs over chains providing all the runtime rights delegation support to the system. Every resource is identified to have some primitive operations like add, remove, edit, info and list, each belonging to one or more authorization schemes.

Operations are handled by names, thus their semantics are agnostic to the access control system. Moreover, the authorization requirements are implemented in a very fine-grained manner providing the flexibility in the usage. The decentralized control allows parent resources to define and manage authorization controls for its children making the administration very easy. Finally, the mechanism was designed to suit quickly to the hierarchical nature of relationship among entities in enterprise environments, thus enabling wide acceptance in enterprise softwares.

With this background, we discuss the important changes proposed to the mechanism as AccessGuard [4] services, which include the following:

- Authorization Contexts
  - ChildAuth
  - PublicAuth
  - GroupAuth
  - CustomAuth
- Authentication Contexts
  - SessionAuth
  - OpenIDAuth

3. AUTHORIZATION CONTEXTS
We define an authorization context to be the field or space in which the access control mechanism shall look for co-ordination based rights delegation at runtime. Thus an authorization context refers to a set of resources that satisfy certain conditions in coordination model so that they are searched for rights delegation during access control. By default, in Key-Chain-Web mechanism, the authorization

context consisted of a set of parent resources within certain level from the given resource. Thus this is what may be called ParentAuth context and is the default method earlier. Based on the conditions in coordination model, we came to identify four more types of authorization contexts which would be highly useful in access control systems for clouds and grids.

We elicit below the proposed four authorization contexts that comprise the AccessGuard service:

3.1 ChildAuth

As is obvious after identification of ParentAuth, the next idea would be to go in reverse direction. Such an authorization context made up of a set of child resources within certain level from the given resource is called ChildAuth. An interesting consequence of this context is the possible easy implementation of no-read-up/no-write-down security policy of Bell and La-Padula model and similar other read/write based policies for information security [5].

3.2 PublicAuth

Some resources demand no restrictions for their access, even public anonymous access [6] may be provided. Modification of Key-Chain-Web mechanism was challenging at the implementation level for achieving this which resulted into what is known as PublicAuth. This authorization context basically has no resources for rights delegation since it is allowed for all; it only has a means to specify public nature of authorization control and a way to represent anonymous users.

3.3 GroupAuth

GroupAuth is a smooth side effect to improvements in ChildAuth. With growing enterprises, it was found that certain resources are shared among users related to a particular set of resources. The co-ordination model easily captures them by using GroupAuth authorization scheme which essentially defines ChildAuth over another root resource specified by the resource itself. Such an authorization can be extensively used to bound the access to any resource to a set of users with certain similar relationship with one another.

3.4 CustomAuth

This final authorization scheme goes to implementation level and gives an option of defining the necessary property for any set of resources whose members are allowed access. This property is evaluated on access control and depending on its value, the mechanism succeeds or fails. CustomAuth are particularly useful in scenario where the property which bounds the set of resources is not explicitly defined by the coordination relationships.

4. AUTHENTICATION CONTEXTS

We define authentication context to be alternative methods for identifying the same key. Many methods exists including username/password tokens or certificates like X.509 [7] or Kerberos [8]. By default, the Key-Chain-Web mechanism uses username/password token.

We elicit below the proposed two authentication contexts that comprise the AccessGuard service:

4.1 SessionAuth

This corresponds to saving some state on the system in the form of sessions. The sessions may be saved using cookies [9] on client side, so that the user may be identified on every request. Thus SessionAuth finds utilization in enterprise software-as-a-service models for authenticated service interaction from user interface.

4.2 OpenIDAuth

OpenIDAuth is just another method to identify the user key based on predecided OpenID [10] identities set by the user. The OpenID framework is emerging as a viable solution for Internet-scale user-centric identity infrastructure. The OpenID identity selects the key for the user which may be combined with SessionAuth for stateful interaction. The basic use of this is in the user convenience to use multiple identities all referring to same key in the Key-Chain-Web model.

5. IMPLEMENTATION DETAILS

The Key-Chain-Web mechanism with AccessGuard services is currently implemented over relational database system using MySQL. The CirrusBolt [11] project implements the chain.authorize service which is the core service in AccessGuard. This service have been modified to implement the proposed additions and the result tested by using the services in TPR Executive [12] project which is a service oriented enterprise application built using principles of software-as-a-service.

We give below the implementation details of each of the proposed contexts:

5.1 Authorization Contexts

These are implemented by modification to the structure and interpretation of design of chains and webs.

5.1.1 ChildAuth

This is implemented very easily by extending level value to include negative numbers in chains. When negative, this shall imply opposite direction to parent which suits to refer children. Some important code changes are:

```
$level = $memory['level'];
$moveup = $level > -1;
if(!$moveup) $level = -1 * $level;
$memory['level'] = $memory['level'] > -1 ?
$memory['level'] + 1 : $memory['level'] - 1;
```

5.1.2 PublicAuth

This is implemented by using prefix for operation specification. In our case, we used ‘pb’ prefix on operations to indicate PublicAuth. The lines that deal with this context are:

```
if(($memory['init'] || $memory['self']) &&
($memory['keyid'] == $memory['masterkey'] ||
strpos($memory['authorize'],'pb'.$memory['actio
n']) !== false ||
(strpos($memory['authorize'],$memory['action'])
=== false && $memory['keyid'] > -1)))
        return $memory;
```

5.1.3 GroupAuth

This is accomplished by adding two fields to chains:

5.1.3.1 grroot

this field gives ID of the root used for group check.

5.1.3.2 grlevel

this indicates the maximum levels into the web hierarchy to look for GroupAuth

The relevant code snippets are as follows:

```
if(strpos($memory['authorize'],'gr'.$memory['ac
tion']) !== false){
  $level = $memory['grlevel'];
  $moveup = $level > -1;
  if(!$moveup) $level = -1 * $level;
  $memory['chainid'] = $memory['grroot'];
}
```

5.1.4 CustomAuth

This simply involves executing custom services if needed and using the result as affirmations for access control. The code changes are as follows:

```
if($memory['custom']){
  $memory = Snowblozm::execute(
    $memory['custom'], $memory);
  if($memory['valid'])
    return $memory;
  $memory['valid'] = true;
}
```

5.2 Authentication Contexts

These are implemented by addition of new schemas and services as elaborated below.

5.2.1 SessionAuth

We added a schema table called sessions to save the sessionid, keyid and expiry values. The sessionid is sent over cookie and is checked on every request to get the session user. The functional parts of the service are:

```
if(isset($_COOKIE[COOKIEKEY])){
  $memory = Snowblozm::run(array(
   'service' => 'cbcore.session.info.workflow',
   'sessionid' => $_COOKIE[COOKIEKEY]
  ), $memory);
}
```

5.2.2 OpenIDAuth

For implementing this, we added a schema table called openids to save OpenID emails and corresponding keyids. This table is queried for getting the keyid after authenticating with OpenID email. Later on we may add OpenID URL instead of email. Some code excerpts are as follows:

```
if(!$openid->mode &&
$memory['openid_identifier']){
$openid->identity=$memory['openid_identifier'];
  $openid->required = array('contact/email',
        'namePerson/first', 'namePerson/last');
  header('Location: ' . $openid->authUrl());
  exit;
}
elseif($openid->mode == 'cancel'){
  // Cancel
}
elseif($openid->validate()) {
  $attr = $openid->getAttributes();
  // Use Attributes and select keyid
}
```

6. CASE STUDY

We give the example of TPR Executive which is the Training and Placement portal for IIT BHU Varanasi.

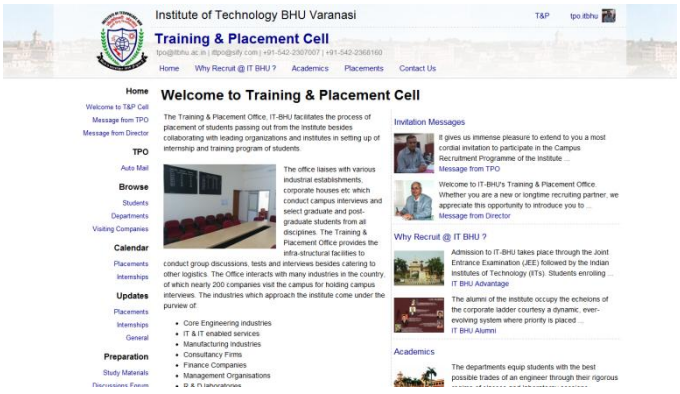

Figure 1. TPR Executive Training & Placement Portal

In this project we mention the modules where we used the different improvements below.

6.1 Willingness

This module requires that the willingness of students of same department are visible to each other. We achieved this using CustomAuth for checking the department value.

6.2 Company

This module required that the documents are shared among the students. For this we used GroupAuth over all students.

6.3 Preparation

The preparation portal requires that it be managed by students of the institute. So we used ChildAuth since there is no difference between read and write access.

6.4 Notes

The notes module gives every user a space for posting articles and links and give comments. In this we used PublicAuth when the post is made publically visible.

6.5 Gmail Sign-in

The portal uses Gmail Sign-in using OpenIDAuth and SessionAuth together since we require stateful interation from client end.

7. FURTHER RELATED WORK

We point to further related work in this section that is related to the approach in this paper.

7.1 Coordination Models

Coordination models are very useful tools for managing objects and resources for large distributed systems including clouds and grids. The Key-Chain-Web model can be extended to include a very versatile model for specification and use of coordination relationships. This is being pursued in the development of Coordination Graphs in which the current work provides the foundation concepts.

7.2 Service Oriented Environment

Service oriented environments are very popular nowadays due to the agility obtained for adapting to changing requirements of enterprises. A different kind of service oriented

environment is being developed on which services can run and compose very easily and this paper provides the insights for this project.

7.3 Cloud Security Architecture

Cloud security is an emerging issue and many solutions are being proposed. One of our projects also concentrates on development of secured data architecture for clouds and this leverages the AccessGuard framework.

8. CONCLUSIONS

We have presented a set of improvements to the innovative Key-Chain-Web mechanism for access control using coordination based models. All the improvements were explained with implementation details and case study. The set of improvements are hoped to be consistent and cohesive to the initial design and provides completeness to the mechanism. It has increased the flexibility and adaptability of the access control system. The mechanism is built to be extensible enough suit to changes yet to be proposed.

9. ACKNOWLEDGMENTS

The guidance and support given by Shri Ravi Shankar Singh is gratefully acknowledged.

10. REFERENCES

[1] V. Rajan and R. S. Singh, May 2012, “A mechanism for flexible access control during co-ordinated resource sharing in enterprise grids”, IEEE Explore, pp.341-345.

[2] R. S. Sandhu and P. Samarati, September 1994, Access Control Principles and Practice, IEEE Communications Magazine, pp.40-48.

[3] R. S. Sindhu et. al., February 1996, “Role based access control models”, IEEE Computer, pp38-47

[4] AccessGuard Framework, 2012, https://github.com/tr4n2uil/cirrusbolt/tree/master/php/accessguard

[5] Chiara Bodei et. al., 1999, “Static analysis of process for no red-up and no write-down”, LNCS 1578, pp.120-134, Springer

[6] M. Backes, J. Camenisch and D. Sommer, November 2005, “Anonymous yet accountable access control”, ACM WPES ’05

[7] X.509 Certificates, ITU-T Recommendation, ISO/IEC 9594-8http://www.itu.int/rec/T-REC-X.509/en

[8] B. Clifford Neuman and Theodore Ts'o, September 1994, "Kerberos: An Authentication Service for Computer Networks," IEEE Communications, vol. 32 (9), pp. 33–38.

[9] D. M. Kristol, November 2001, “HTTP Cookies: Standards, privacy, and politics”, ACM Transactions on Internet Technology, pp.151-198

[10] D. Recordon and D. Reed, 2006, “OpenID 2.0: a platform for user-centric identity management”, Proceedings of the second ACM workshop on Digital identity management, pp.11-16

[11] CirrusBolt, 2011, Service Computing Platform Foundation Engine, https://github.com/tr4n2uil/cirrusbolt

[12] TPR Executive, 2012, Training and Placement Portal, https://github.com/tr4n2uil/tprexecutive