

Electronic Credential based Security Management in Decentralized Computing Environment

Chenxi Huang*, Jie Xu**, Keith Bennett*

*Department of Computer Science, University of Durham
Durham DH1 3LE, England
chenxi.huang@durham.ac.uk

**School of Computing, University of Leeds
Leeds LS2 9JT, England

Abstract

Role Based Access Control (RBAC) and Access Control List (ACL) are the most commonly adopted access control mechanisms in traditional centralized computing environment. Nowadays people frequently work in a highly dynamic and distributed computing environment, in which two or more heterogeneous systems do not share the same security domain. Traditional access control mechanisms that require pre-registration become obsolete. Electronic credential based distributed access control turns out to be the best solution to the current situation. We adopt three kinds of certificates in the solution: attribute certificate, capability certificate and delegation certificate. A language is required to delineate a unified certificate format and to enable administrators to define the trust relationships between e-strangers. We present a policy language, Z-Trust Language, to meet this end. Administrators can finely manage privileges assigned to individual users with the Language. A java program has also been developed as the compliance checker.

1. Introduction

In cyber space, people meet on websites, communicate via email and do business online. Unlike traditional face-to-face trading, they do not know the actual appearance and/or behaviour of each other. How can they trust an unknown cyber business partner before the real transaction starts? Fortunately, modern people hold passports of the cyber space, electronic credentials, to prove their trustworthiness. Electronic credential is the counterpart of paper credential of the real world. For example, X.509 is one of the most popular formats. E-strangers are those who have no previous knowledge of each other but prove their authenticity and trustworthiness by disclosing electronic credentials to establish trust. In a highly distributed computing environment like the

Internet, the trust relationship between e-strangers is often dynamic and transient, subjects and objects do not share the same security domain as in traditional centralized computing environment.

Traditional access control mechanisms, such as Access Control List (ACL), require pre-registration with the server before users could be granted access right. The limitations of ACL in a distributed environment had been discussed in [6].

Here is a typical situation that often occurs in distributed environment. IBM has a collaborative project with another software company DurSoft. The two companies need to exchange staff to successfully carry out the project. Alice is an employee of IBM and she needs to work temporarily in DurSoft for one

month. She should be able to access all the local resources of the project during her stay in DurSoft. But she does not have an account in DurSoft, nor does anyone in DurSoft know her before. How can she prove her validity to access the local resources? On the other hand, the regulation of DurSoft allows the technicians of the project from IBM to legally access local resources, how can the security administrator of DurSoft describe and manage the trust relationship among IBM, DurSoft and the project?

Alice needs to show a certificate issued by IBM to certify her validity to access the resources. The administrator of DurSoft needs to specify a security policy that accepts certificates issued by IBM and correctly interprets the content of the certificate. We present a policy language both to define the certificate format and to set up the security policy.

2. Related Credential Based Access Control Approaches

PolicyMaker and KeyNote [1] first drew the concept of trust management, which they define as “a unified approach to specifying and interpreting security policies, credentials, and relationships; it allows direct authorization of security-critical actions”. Five basic components were introduced. They are actions, principals, policies, credentials, and compliance checker. A policy language had been developed for both the credentials and the policies. The principal perform actions on the resources. A principal is able to delegate actions to another user through assertions. The drawback of PolicyMaker is that it was designed to be a programming language and too complicated for those non-programmers. We also feel that an action is too simple to describe the privileges assigned to users. The design of KeyNote is unable to meet the new requirements of some kind of situations, such as the requirements of patient-centred healthcare.

PERMIS has defined a Privilege Management Infrastructure (PMI) that makes authorization based on the Attribute Certificate (AC). The PMI architecture converts AC to RBAC by mapping the outside users to local roles based on their attributes. Much research has been done trying to apply RBAC into Health domain, but failed with the conclusion that RBAC itself is too inflexible to meet the requirements of patient-centred healthcare [4]. PMI is an extension of RBAC thus it inherits the limitations of RBAC, it cannot finely control the privileges assigned to the individual users.

Extensible Access Control Markup Language (XACML) is an OASIS standard specification that defines the XML schema for extensible access-control policy language. It provides a method for basing authorization decision on attributes of both the subject and the resource. A set of standard logical and mathematical operators has been defined. Furthermore, it could define a series of actions that must be performed in conjunction with policy enforcement, i.e., obligations. The deficiency of XACML is that it does not support privilege delegation.

3. Certificates

Three kinds of certificates are involved in the certificate-based solution. They are attribute certificate, capability certificate, and delegation certificate. Attribute certificate contains attributes of the certificate holder. Attributes are name-value pairs such as “name=Alice” or “company=IBM”. Attributes describe the common properties of a certain set of people. Capability certificate contains the capabilities that the certificate holder has been granted. Capability is what a user can do in the designated system. For example, “Alice can read the documents of project A” is a capability. Capabilities relate individual users with arbitrary privileges therefore complement the deficiency of attribute certificate. Delegation certificate

delegates capabilities from one principal to another. Delegation in our context is the ability to issue a certain kind of certificate based on the upper level's authorization. For example, Alice says "Bob is able to issue capability certificate to other people to access my resources" is a kind of delegation. Delegation could be infinitely transferred unless delegator inhibits. Delegation facilitates separation of power and enhances security.

4. The Z-Trust Policy Language

4.1 The need for the policy language

The benefits of devising a new policy language to define the certificate format and specify the security policy over an existing programming language had been partially discussed in [1]. First of all, the language is application independent so that the same certificate/policy could be understood by all the applications that share the same standard language. There is no need to develop a particular authentication and authorization system for each application thereby simplifying the security design of the application. Secondly, security policies are dynamic and subject to frequent change to meet newly surfaced security requirements. It is much easier to rewrite the policy than to rewrite and redeploy the application.

4.2 Language definition

Here is the definition of the policy language. The server has a security policy Policies = {{Precondition+, Privilege+}*}. Precondition = {Rule*}, Rule = {{Auth}+ || {Cert}+}, Cert = {Auth?, Type, Prop*}, Prop = {{Attr || Capa || Dele}+}. Privilege = {{Type, Prop*}+}.

“+” denotes one or more occurrence. “*” denotes zero or more occurrence. “?” denotes zero or one occurrence. “||” denotes logical operator OR. Auth denotes the unique identity of the trusted certificate authority. Type denotes a certain format of the certificate. Prop denotes the properties contained in the certificate. Attr

denotes attributes, which is one kind of properties. Capa denotes capabilities, which is one kind of properties. Dele denotes delegation capability, which is one kind of properties.

Trust in our context is completely based on the certificates that the user produces. It is *certificate trust* or *issuer trust*. If the issuer of the certificate could be found in the server's trust list and the content is within allowed set of *Privilege*, then trust is established and the user will be given access right. Under this condition the issuer is called *source of trust*. If the identity of the issuer is not recognized, then a valid certificate chain must be found, which traces back to a *source of trust*, to support the user's request. The certificate chain could be comprised of a series of attribute, capability and/or delegation certificates. If none of the above could be satisfied, then trust establishment process fails.

4.3 Example

Here is the security rule written in plain language to be enforced. "The owners of the resources are able to grant/delegate access rights to their own resources to anyone."

The following is the same rule written in our Z-Trust policy language.

```
[01] <Policy>
[02] <!-- "owners-decide-everything-principle"
the owners of the resources are able to issue
capability certificate, delegation certificate of
their own resources to anyone. -->
[03] <Precondition>
[04] <Issuers>
[05] <AnyIssuer/>
[06] </Issuers>
[07] </Precondition>
[08] <Privilege>
[09] <CapabilityCert>
[10] <Function id="equal">
[11] <AttributeValue>
[12] <Source>certificate</Source>
[13] <Name>issuer</Name>
```

```

[14] </AttributeValue>
[15] <AttributeValue>
[16] <Source>resource</Source>
[17] <Path>
[18] <AttributeValue>
[19] <Source>certificate</Source>
[20] <Name>target</Name>
[21] </AttributeValue>
[22] </Path>
[23] <Name>owner</Name>
[24] </AttributeValue>
[25] </Function>
[26] <Actions>
[27] <AnyAction/>
[28] </Actions>
[29] </CapabilityCert>
[30] </Privilege>
[31] </Policy>

```

[03]-[07] the precondition the issuer must satisfy. Here the element *AnyIssuer* means everyone could issue such a certificate. The format of the certificate will be given in the following sections

[09]-[29] element *CapabilityCert* describes the content of the capability certificate that could be issued

[10] The element *Function* indicates an internal function. *equal* is the name of the function. The function takes two input parameters and returns the compared result

[11]-[14] the first input parameter of the function is the issuer of the certificate

[15]-[24] the second input parameter of the function is the owner of the resource

[17]-[22] the *Path* element indicates the exact location of the resource

[26]-[28] any action understood by the application could be granted

5. Conclusion

We draw an improved certificate based solution to solve the access control problems in the distributed environment. We have presented a policy language, Z-Trust Language, to help the

security administrators describe the diverse trust relationships between E-Strangers. The semantic of the language is simple, easy to understand. It is server-centric and supports three kinds of certificates. Trust establishment process is based on both sides' attributes. It enables administrators to finely grant privileges to the individual users, hence complements the deficiencies of other approaches and RBAC.

References

- [1] M. Blaze, J. Feigenbaum, J. Ioannidis, A. Keromytis, "The Keynote Trust-Management System", RFC 2704, September 1999, version 2
- [2] D. W. Chadwick, O. Otenko, "The PERMIS X.509 Role Based Privilege Management Infrastructure", SACMAT 2002
- [3] S. Godik, T. Moses, A. Anderson, B. Parducci, C. Adams, D. Flinn, G. Brose, H. Lockhart, K. Beznosov, M. Kudo, P. Humenn, S. Andersen, S. Crocker, "eXtensible Access Control Markup Language (XACML)", OASIS standard, 2003, version 1
- [4] M. Turner, F. Zhu, I. Kotsiopoulos, M. Russell, D. Budgen, K. Bennett, P. Brereton, J. Keane, P. Layzell, and M. Rigby, "Using Web Service Technologies to create and Information Broker", ICSE, 2004
- [5] "The medical records confidentiality act of 1995", http://www.cdt.org/privacy/medical/950000mrca_summary.shtm
- [6] M. Blaze, J. Feigenbaum, J. Ioannidis, and A. D. Keromytis, "The Role of trust management in distributed systems security", in Proceedings of Fourth International Workshop on Mobile Object Systems: Secure Internet Mobile Computations (MOS'98, Brussels, Belgium), no. 1603 in Lecture Notes in Computer Science, (Heidelberg, Germany), pp. 185-210, Springer-Verlag, July 1999