A Decentralized Authorization Mechanism for E-Business Applications *

Zoltán Miklós

Technical University of Vienna, Distributed Systems Group
Argentinier Straße 8/184-1
A-1040 Vienna, Austria
Z.Miklos@infosys.tuwien.ac.at

Abstract

E-business applications need robust and powerful mechanisms to authorize security-critical actions. This actions can be very complex, since they can be initiated not only by human users but also by applications or software agents.

Existing authorization mechanisms do not scale for large number of users if the trust relations are dynamic and fail to provide reliable authorization among strangers. Our mechanism uses authorization relevant attributes to define the policy. The attributes are assigned to principals in a decentralized manner.

We also present a method to reduce the financial losses which may arise if the authorization mechanism fails.

We conclude the paper with our plans for future research.

Keywords: trust management, authorization, assurance

1 Introduction

Most of the existing public key infrastructures can only be used to authenticate users. To separate the authorization process into two phases of authentication and access control is not appropriate for many e-business applications since it is very likely that service provider and the user are complete strangers in the physical world. On the other hand, e-business services have to authorize not only persons, but possible other applications or software agents which can initiate much more complex actions than human users.

New public key cryptography based authorization mechanisms [1, 4] bind the authorization information directly to keys. In this way the set of credentials can directly prove whether the requester is authorized to perform the action. Policies can directly authorize keys or delegate the responsibility to other credential issuers that it trusts to have both the required domain expertise and relation with the potential requesters.

In these systems policies authorize directly the principals simply by listing their public keys in the assertion. If a policy designer wishes to authorize large numbers of principals, he can have a large list of keys or he can delegate the right of designing policies. For each additional new service for which he wishes to authorize large numbers of principals, he again has to define a large list or initiate a large delegation network.

In real world situations the policies usually change less often than authorization relevant attributes, so this also indicates to separate the policy and attribute credential assertions.

This problems motivated us to design a public key based authorization mechanism which uses authorization relevant attributes to define the policy. With this method, defining policies for new services is simple and there is no need to redistribute the credentials, because the new policy can be defined using the existing attribute assertions. In our mechanism, which is called Nereus, the right to issue attribute credentials can be delegated, which makes possible to authorize actions of strangers.

1.1 Trust management approach to authorization

Trust management was first introduced in [3]. In trust management approach policies and trust relationships are expressed as programs. This makes it possible to define also complex trust relationships. A central part of trust management approach the delegation: any principal can issue credentials or delegate the responsibility to other parties he trusts to issue correct credentials.

The trust management approach separates the authorization decision from the application. The authorization question in trust management reads as follows. “Does the set

*This work was supported in part by the European Commission under contract IST-1999-10288, project OPELIX (Open Personalized Electronic Information Commerce System).

of credentials prove that the request compiles with the local security policy? ” The compliance checking procedure bases on a formal proof.

The remainder of the paper is structured as follows. In Section 2 we present our authorization mechanism. Section 3 describes a possible extension of our system using insurance techniques. Section 4 contains related work and section 5 provides conclusion and a look at future work.

2 The Nereus trust management system

2.1 Motivation

Binding the authorization information directly to public keys has apparent security advantages. The proof whether the action is allowed can be proved based on this credential. This approach also enables the delegation of issuing credentials. On the other hand if the delegation network is large, the original intent of the service provider might be lost. An another problem is, that cheating entities in the delegation network can defeat the whole authorization mechanism.

Authorization relevant attributes usually change more frequently than the policies so this motivated us to handle these statements separate.

Our approach shares the idea to completely separate the trust decision form applications with the existing trust management systems [2, 3]. We think nevertheless that for designing a good and expressive policies the designer needs to have a direct contact with the service provider. To find the policy designer, who has both the domain knowledge and direct contact with requesters might require long delegation chains, and the service provider has no guarantee, that the policy really corresponds with his original intent. Big delegation networks also involve the risk of having liable chain links.

2.2 Overview

The policy definitions of Nereus base on attributes. This makes policies highly flexible and human readable and also appropriate to define authorization of large number of users. The authorization attributes are assigned by credential issuers. Any principal may issue credentials, the role of delegation is to enable to find the right issuer, who has reliable information about the key owner. The right to issue credentials can be delegated. In this way we do not bind the authorization to keys, but the authorization decision can still be made relying on credentials and policies.

Nereus adopts the trust management approach, where the authorization is viewed as a proof-of-compliance problem. The Nereus trust management system has five basic components.

- A language, describing the security critical actions.
- A mechanism identifying the principals, who can be authorized to perform an action.
- A policy language, which enables to describe the authorized actions
- A credential definition language, which enables to define the authorization relevant attributes, and supports the delegation of credential issuing.
- A general-purpose and application independent compliance checker, which determines whether an action is authorized based on the set of credentials and policies.

In Nereus principals are identified by their public keys, actions are defined in the current version as a set of name - value pairs.

To reduce the risk of false attribute certification, we suggest an insurance mechanism, which is described in section 3.

2.3 Credentials and policies in Nereus

There are two types of attribute credentials in Nereus. *Binding credentials* bind specific attributes to public keys and *delegation credentials*, which allow to delegate the right to bind a specific attribute to keys.

Example of an attribute credential: (We use very short keys only to keep the credentials readable.)

```
Nereus-Version:  1
Comment:  A simple attribute credential
Type:  binding
Issuer:  "RSA:abcd1234"
Licensee:  "RSA:1234abcf"
Attribute name:  position
Attribute value:  manager
Insurance:  120
NotBefore:01-01-2001
NotAfter:  31-12-2002
Signature:  "RSA-SHA1:f102bca"
```

This credential states that the owner of the key RSA:1234abcf is in position manager. This is certified by RSA:abcd1234. The credential may also contain more attributes.

To delegate the right to more than one entity is also possible. In this case the keys should be separated with ‘OR’.

Example of a delegation credential:

```
Nereus-Version:  1
Comment:  A simple attribute credential
Type:  delegation
Issuer:  "RSA:abcd1234"
```

```
Licensee:  "RSA:1234abcf"
Attribute name:  position
Attribute value:  manager
Insurance:  120
NotBefore:01-01-2001
NotAfter:  31-12-2002
Signature:  "RSA-SHA1:f102bca"
```

The difference to the previous example is the credential type, which is here 'delegation'. The owner of the key RSA:abcd1234 issues this credential, if he wants to delegate the responsibility of certifying the attribute 'manager' to key RSA:1234abcf.

There are also two types of policy credentials. Binding credentials assign authorization to attributes, and delegation credentials, which allow the delegation of the right to define policies.

Example of a policy:

```
Nereus-Version:  1
Comment:  A simple policy
Type:  binding
Issuer:  "RSA:1357acd"
Licensees:  @(position)=manager AS-
SERTED BY "RSA:abcd1234" &&
@(entitled_to_sign)=yes ASSERTED BY
"RSA:1432dbca"
Conditions:  (app_domain == "SPEND") &&
(@(dollars) < 1000 )
Minimum insurance value:  110
```

This policy allows for owners of keys, who have been certified as manager and entitled to sign to spend money less than 1000 dollars. The policy designer can specify, who can certify these attributes. The holder of the key RSA:abcd1234 can issue an attribute credential or delegate the responsibility.

There is a special policy with the text 'POLICY' in issuer field, which represents the local policy and serves as "trust root" in compliance checking. The following example shows a policy which delegates all responsibilities to define policies to key RSA:1234fff.

```
Nereus-Version:  1
Type:  delegation
Issuer:  "POLICY"
Licensees:  "RSA:1234fff"
```

2.4 The authorization mechanism of Nereus

Nereus uses very similar authorization mechanisms as the existing trust management systems. The requester sends all relevant credentials with his request. The authorizing application passes the user request, the user credentials and the policy credentials to the compliance checker, which decides whether the action conforms the security policy with the presented set of credentials. In our mechanism, like in KeyNote trust management system the proof of compliance is a general purpose and application-independent algorithm. As argued in [2], this approach has several advantages, because the compliance can be formalized, proved and implemented in a standard package and so we can gain a more reliable implementation and a clear definition.

Informally, the compliance checker of the KeyNote trust management searches one path in the users credentials graph from the local POLICY assertion to the requester key. Our system on the other hand checks whether all required attributes are present.

We investigate the formal proof of our compliance checking mechanism.

2.5 Comparison with KeyNote trust management system

Our system adopts the idea of compliance checking and delegation of responsibilities, but Nereus is different in many points from KeyNote [1].

Use of attributes The use of attributes makes the system more flexible if authorizations for large numbers of users have to be defined. In Nereus the attribute and policy assertions are separated and have different syntax.

Limited delegation The attribute binding credentials cannot be further delegated. The delegation credentials cannot be used to request a service.

Compliance checking Nereus uses a different compliance checking method.

3 Trust insurance economics

In the real world insurance companies help us to alleviate the financial risks we face in different situations. Lai et al. [8] analyzed first the use of insurance techniques in distributed systems security.

We suggest an insurance mechanism for authorization, which is a possible extension of our Nereus mechanism. If a malicious user manages to perform an action, for which he is not authorized, the service provider, who offers the service for money, has less incomes. Our idea is to use an insurance technique to reduce the risk of this kind of losses. We are considering the use of surety bonding technique, which was applied by Reiter and Stubblebine [13] to design a meaningful authentication metrics.

The main idea of the method is that if k_1 issues a credential for k_2 then the credential includes an insurance value,

for which the owner of k_1 insures the assertion. If the assertion turns out to be incorrect, the owner of k_1 is liable for the stated amount.

Lets assume first, that a policy requires one attribute to be present. We can build a graph of credentials as follows. The nodes of the graph are the keys, and the edges represent the delegation relationship or binding the attribute to the key. The source node represents the policy. The target node is the attribute credential. In this graph there exists one or more path from the source node to the target node. Obtaining a false attribute-key binding means that every path form the source to the target node has some liable edge. Reiter and Stubblebine [13] argues, that the minimum insured amount serves a good metrics for the attribute - key binding. This amount can be calculated with the Ford-Fulkerson graph algorithm for determining the graph's minimum capacity cut. For a more detailed description of the insurance technique we refer to [13].

The Ford-Fulkerson algorithm works originally with one source and one target node, as applied in [13]. It is easy to prove, that it can be also applied in our situation, with one source and multiple target nodes. This case can be simple reduced to the one target node case, if we imagine a super-target node, which is insured with an infinite value by all of our target nodes.

Our idea is to include the required minimum insured amount into the policy and ask the compliance checker to calculate the minimum insurance value of the attribute-key binding using the Ford-Fulkerson algorithm. The policy contains the minimum insurance value expected by the service provider. The compliance checker delivers positive answer only if the minimum insurance value is higher, than the required amount.

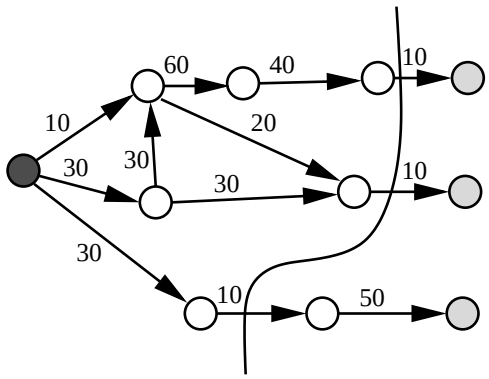

Figure 1. Minimum insurance value

Figure 1 shows a minimum cut which yields 30 dollars insurance for the specified policy. The dark grey point represents the policy (which is in this case identical with the local POLICY), the light grey points represent the required credentials and the white points are the delegation credentials. The number on the edges denotes the insurance value.

Providing insurance for credential can be seen as a good business opportunity. Insurance providers might build up a reinsurance network to further reduce their own risks.

3.1 Discussion

The use of insurance in credentials introduces new problems, like identifying the false delegation in the chain, payment of insurance premiums, recovering funds. Attribute certifiers do not know for what purposes the credentials they issue will be used for. Insurance providers not very likely insure "To whom it may concern" type of statements, so a superset of possible actions should also be included in the credential.

We have to find a solution for these problems if we would like to apply these technique. We belive that is worth to studying these field, even if the problems seem not to easy to solve, because they can lead to a reliable authorization mechanism and so it can stimulate e-business on a global level.

An important aspect of using insurance techniques, that the trust relations are built not on personal relations or trusted authorities, but on the basis of financial interests, which might be preferable for the e-business community. The trust insurance economy needs to be further studied to determine, whether the self regulating economy can provide the right solution for companies. A service provider probably will not accept a solution, where users very often gain access to the service using bogus credentials, even if he can recover the losses from insurance providers. In this case we might need -possible decentralized- endorsement and licensing services to regulate the development of economical processes.

4 Related work

Our work is closely related to KeyNote [1]. A comparison of our system to KeyNote can be found in Section 2.

Herzberg et al. [7] presents a trust policy language (TPL) to define the mapping of strangers into roles. Certificate issuers are either known in advance or have to present sufficient certificates to be considered as a trusted certificate issuer. Their system supports also negative certificates, while in Nereus is assertion monotone. In TPL language the policy designer can limit the length of certificate chains and specify for important roles, how many different certificate issuer has to assign. This technique increases the reliability of the system, but as argued in [13] the insurance based method provides more meaningful results. Their system automatically collects the missing certificates.

Li [9] designed a logic-based language to represent policies, credentials and requests in distributed authorization in his PhD thesis. This language as Nereus enables to delegate the attribute authorities to entities having certain attributes. In his later works [10] Li addresses the credential chain discovery and proposes a role-based trust-management language.

Feigenbaum [6] analyzes the infrastructural needs for authorization which enables electronic commerce. She formulates basic principles, what we also try to apply, namely use of expressive credentials and policies, authorization based on compliance checking, competent credential issuers.

In SPKI/SDSI [4] certificates can be also used directly for authorization. In delegation is similar to our delegation, SDSI certificates contain a boolean value specifying whether the owner of the public key is permitted to delegate the authorization. In SPKI also the delegation certificates can be used as credentials opposite to Nereus.

The X.509v3 certificates may have an 'Extension' field, which can be used to provide some attributes or privileges of the owner (for example 'manager' or 'right to sign bills'). The problem with this approach, that the life time of this privileges usually much shorter than the life time of the certificate. Attribute certificates [5] try to overcome these problems. AC is a separate structure from the identity certificate, usually short lived and binds attributes of the owner to his public key certificate. A person may have multiple ACs connected to his public key certificate. Nereus addresses the analogous problems in trust management. Attribute certificates are used by Oppliger et al. [12] to implement role-based authorization and access controls.

5 Conclusion and future work

We presented the design of a trust management system which is scalable for large number of users. In our system the policies grant access rights on the basis of authorization relevant attributes. Policies only grant positive access rights. The attributes are assigned to principals in a decentralized manner.

We also presented an insurance technique, which can enable very robust authorization mechanisms and so can be applied to authorize also previously unknown entities.

In the current version of Nereus we support simple name value pairs as actions, which is appropriate for a large number of applications. We are working on more sophisticated area specific definitions of the security-critical actions for publish/subscribe middleware [11]. The application independent authorization procedure makes possible to apply the mechanism for middleware services.

We are also working on a formal proof of correctness of our compliance checker.

We made the implicit assumption, that all relevant credentials and policies are avaliable at the time of compliance checking as other trust management systems also do. It is not a trivial task to collect all required credentials, it needs further investigations.

Credential revocation also raises important questions.

References

[1] M. Blaze, J. Feigenbaum, J. Ioannidis, and A. Keromytis. *The KeyNote Trust Management System Version 2*. IETF. `http://www.ietf.org/rfc/rfc2704.txt`.

[2] M. Blaze, J. Feigenbaum, and A. D. Keromytis. The role of trust management in distributed systems security. In *Secure Internet Programming*, pages 185–210, 1999.

[3] M. Blaze, J. Feigenbaum, and J. Lacy. Decentralized trust management. In *Proceedings 1996 IEEE Symposium on Security and Privacy*, pages 164–173, May 1996.

[4] C. M. Ellison, B. Frantz, B. Lampson, R. Rivest, B. M. Thomas, and T. Ylonen. *SPKI Certificate Theory*. IETF. `http://www.ietf.org/rfc/rfc2693.txt`.

[5] S. Farrell. *An Internet AttributeCertificate Profile for Authorization, Internet Draft*. IETF. `<draft-ietf-pkix-ac509prof-*.txt>`.

[6] J. Feigenbaum. Towards an infrastucture for authorization (position paper). In *1998 USENIX E-Commerce Conference – Invited Talks Supplement*, pages 15–19, 1998.

[7] A. Herzberg, Y. Mass, J. Mihaeli, D. Naor, and Y. Ravid. Access control meets public key infrastructure, or: Assigning roles to strangers. In *IEEE Symposium on Security and Privacy*, pages 2–14, 2000.

[8] C. Lai, G. Medvinsky, and C. Neuman. Endorsements, licensing and insurance for distributed system services. In *Proceedings of the Second ACM Conference on Computer and Communications Security*, Nov. 1994.

[9] N. Li. *Delegation Logic: A Logic-based Approach to Distributed Authorization.* PhD thesis, New York University, New York, Sept. 2000.

[10] N. Li, W. H. Winsborough, and J. C. Mitchell. Distributed credential chain discovery in trust management. In *Proceedings of the 8th ACM Conference on Computer and Communications Security*, Nov. 2001.

[11] Z. Miklós. Towards an access control mechanism for wide-area publish/subscribe systems. In *Proceedings of the International Workshop on Distributed Event-based Systems*, July 2002.

[12] R. Oppliger, G. Pernul, and C. Strauss. Using attribute certificates to implement role-based authorization and access controls. In *Proceedings of the 4. Fachtagung Sicherheit in Informationssystemen (SIS 2000)*, pages 169–184, 2000.

[13] M. K. Reiter and S. G. Stubblebine. Authentication metric analysis and design. *ACM Transactions on Information and System Security*, 2(2):138–158, 1999.