Identity Mapping
An Approach to Unravel Enterprise Security Management Policies

WOLFGANG ESSMAYR, EDGAR WEIPPL
Software Competence Center Hagenberg
Hauptstr. 99, A-4232 Hagenberg, Austria
E-mail: {wolfgang.essmayr, edgar.weippl}@scch.at

Key words: enterprise security management, authentication, authorization, access control

Abstract: Increasingly complex networks and distributed services entail new challenges concerning interoperability and integration of security mechanisms. The currently available solutions, e.g. directory services or distributed authentication systems have disadvantages that can be overcome by a new approach based on mapping identities. Identity mapping allows assigning the identity of one human to different users in various systems. The security features of every system can be fully used and no common denominator limits the power of a single system. This paper[i] describes the different types of mappings that are necessary to implement such a system. Mappings cannot occur only on a user-user basis but also roles and groups have to be considered to correctly represent modern security issues.

1. INTRODUCTION

During the last years electronic networks grew increasingly complex. In a modern enterprise there are many different IT systems that cooperate more or less integrated. Every system has its own security model, its own ways of authenticating and authorizing users and controlling their access.

The traditional, "chaotic" approach is that users, group- and role-hierarchies are set up for every system, individually. This leads to an enormous amount of redundancy as a single user, e.g. Jane Doe, will have an account on different operating systems, database management systems, workflow management systems, email-, Web-, and Ftp-servers. Individual applications may also require an account and so Jane Doe will end up having

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35515-3_53

10 or more usernames (JD, JaneDoe, Jane, Doe, JDoe, ...) and passwords. For a system administrator this will eventually become a nightmare, as she has to set up groups and roles, too.

Directory services (e.g. LDAP, compare Wahl et al., 1997) are an attempt to solve this problem. Accounts are managed hierarchically in a central directory that all applications access whenever they need information about a user. The disadvantages of this monolithic approach is that all applications have to support the system and that special security features of an application cannot be modelled correctly using a directory service like LDAP. Distributed authentication systems (e.g. Kerberos (see Kohl and Neuman, 1993), or SESAME (compare McMahon, 1996)) offer ways how an individual can identify herself and how she can be authenticated using a third party.

We propose an approach that combines the "chaotic" and the monolithic, directory-service-based solution. A central "identity server" stores the *identities* of users (simple identities) and of groups and roles (complex identities). Using "mediators" (compare Wiederhold, 1992), i.e. stubs that translate between the "identity server" and the application, the application does not have to offer any support for the "identity server".

Another important issue is the support of security features that an application offers. As our approach does not store access rights, but provides means for mapping identities, it fully supports all security mechanisms of any application. These two facts are a clear advantage over the directory-based solution.

2. SECURITY ASPECTS AND RELATED WORK

Information security is traditionally viewed as protecting information from unauthorized disclosure and modification regarding confidentiality, integrity, and availability of data items. The used mechanisms have been evolved within operating systems and database management systems and include *authentication* (establish the identity of one party to another), *access control* (determine what one party will allow another to do with respect to resources and objects mediated by the former), as well as *audit* (gather data about the activity in the system and analyse it to discover security violations or diagnose their cause), compare Sandhu and Samarati (1996). Well-known access control models are *Discretionary Access Controls* (DAC), *Role-Based Access Controls* (RBAC) and *Mandatory Access Controls* (MAC). For a comprehensive survey on these topics see, for instance, Castano et al. (1995) and Sandhu and Coyne (1996).

Within distributed environments information that is being transmitted among arbitrary communication partners must also be protected with respect to confidentiality, integrity and further requirements like anonymity, non-repudiation, etc. This kind of protection can be subsumed with the term *communication security* including authentication, key distribution, and secure communication protocols that are all based on cryptographic mechanisms like certificates, digital signatures, ciphers, hash functions, etc. For a detailed discussion of these issues see, for instance, Schneier (1996) or Oppliger (1998).

An enterprise security policy has to incorporate both aspects of security, namely, information and communication security combined with mechanisms that allow handling heterogeneity. An important link for incorporating all these aspects is the *identity* of the enterprise's resources and objects. We thus propose mapping mechanisms for identities that have the potential to enhance the integration of information and communication security mechanisms.

3. IDENTITY MAPPING

In order to provide interoperability with respect to different identity types a mapping has to be established that allows synchronizing or simply mapping two or more identities of one or more identity types between different security domains. These mappings allow assigning a source identity, for instance, the user *Jane Doe*, to various corresponding target identities within different information systems. This concept has several advantages: First, human identities like users, for instance, are no longer forced to be schizophrenic by having multiple identities for different information systems. Furthermore, users are no longer forced to remember multiple passwords, i.e. *identifiers*. This fact dramatically increases the degree of security of particular environments. Second, the target information systems need not be changed for achieving interoperability with respect to security. Other approaches require information systems to be compliant to a particular standard or protocol (e.g. Kerberos or Sesame). Third, all the security mechanisms already offered by target information systems as, for instance, identification and authentication, authorization and access control, or auditing, can be fully utilized and need not be circumvented by creating virtual accounts with comprehensive privileges. Fourth, security information distributed across many information systems within an enterprise can be monitored and administrated centrally if the security policy of the enterprise specifies to do so. Finally, identity mappings form the basis for specifying and maintaining a globally defined, enterprise-wide security policy incorporating the security policies of any of the applied information systems. In order to realize these concepts an additional management component ("identity server") has to be globally established which has to maintain particular sensitive information of different information systems thus reducing the degree of autonomy of the engaged systems in favour of security, flexibility and interoperability. The concept can be used to realize the following features:

- *Single-sign-on* augments the user acceptance of the distributed environment.
- *Single point of administration* helps security administrators to maintain a globally defined enterprise-wide security policy.
- *Multiple points of access control* and *auditing* increases flexibility and expressiveness using the features implemented by individual information systems.
- *Single point of monitoring* alleviates security controllers to enforce the enterprise-wide security policy.

Within this work, the subject types *user*, *group*, and *role* are used as representative examples for identity mapping since they incorporate all properties that are relevant to specify concepts for identity mappings. Nevertheless, the proposed concepts are more generic and can be applied to further types of identity.

3.1 General Considerations

Heterogeneous systems may apply different kinds of identities including, for instance, numeric identities, textual identities (i.e. simple strings), an X.500 distinguished name, or an X.509 certificate. Additionally, naming constraints with respect to the applicable character set and the minimum and maximum length of the identity have to be considered. Identifiers allow to undoubtedly prove the subject's identity to the system (authentication). The most frequently used identifiers are of course *passwords*, but especially distributed and interoperable environments increasingly use *private keys* for authentication. Identifiers that base the authentication process on particular hardware like, for instance, smart-card readers, fingerprint scanners, etc., are not feasible for being mapped to software-based identifiers since a hardware-based authentication requires the physical presence of the mostly human subjects for authentication.

In general, identity mappings are defined from identities (e.g. users, groups, roles, etc.) located at a *source* domain (also denoted as source identities) to identities located at a *target* domain (denoted as target identities). Each of the mentioned domains represents a particular information system (or security sub-system of an information system) within an enterprise. Each domain may act as a source and as a target domain with respect to different mappings. However, source and target domain of one mapping must be distinct from each other since a mapping among identities of one single domain can rather be regarded as trivial. The source domain serves as the master or global domain that is able to control the mapping. The target domain, however, may preserve the autonomy with respect to accept or reject the establishment of particular mappings.

Identities may be of varying complexity with respect to their relationships to other identities (membership) or their reflexive structure (hierarchy). We distinguish the following identity types:

- *Simple identities*: e.g. users, processes, etc. that are not further structured.
- *Complex identities*: e.g. groups, roles, etc. that represent an arbitrary number of simple identities and may be further structured within hierarchies (e.g. role-hierarchy).

Complex identities have been designed in order to reduce the effort of administrating authorization issues. Authorizations logically propagate from complex identities to their associated simple identities (e.g. from roles to users) as well as along an identity hierarchy that represents the functional and organizational structure of an enterprise. Each complexity level of identities may occur at the source as well as at the target domain, which forms a totality of 4 mapping types, namely, *simple-to-simple*, *simple-to-complex*, *complex-to-simple*, and *complex-to-complex* mappings.

3.2 Identity Mapping Types

We use the following terms throughout this work: (1) *propagation* is an (implicit) transfer of authorizations from one identity to another, (2) *mapping* is an association between a source and a target identity, (3) *derived propagation* is a propagation originated by a mapping, and *implicit mapping* is a mapping originated by another mapping.

A mapping from a simple identity at the source domain (e.g. $user_1$) to a simple identity at the target domain (e.g. $user_2$) is called *simple-to-simple* mapping. The particular semantic of this mapping is that $user_1$ at the source domain should become $user_2$ when requesting services from the target domain. Mappings in general may be established by an administrator (*mandatory* establishment) or by a source domain identity itself (*discretionary* establishment), if the source domain identity knows how to authenticate the particular target identity. In some cases an *automatic* establishment of mappings is feasible, for instance, mapping identities with equal names between the source and target domains.

A *simple-to-complex* mapping from a simple identity at the source domain to a complex identity at the target domain can be regarded as *remote* *membership.* As illustrated in the example of *Figure 1*, *$user_1$* at the source domain is mapped to *$role_1$* and *$role_2$* at the target domain, thus being a remote member of the particular roles. Furthermore, *$user_2$* at the source domain is mapped to *$role_2$* at the target domain, thus being a remote member of *$role_2$*. In order to establish the adequate remote membership definitions, *$newuser_1$* and *$newuser_2$* have to be created at the target domain and have to be made members of the particular roles, there. This kind of membership can be seen as a *derived propagation* of authorizations, since it is originated by the simple-to-complex mapping. As a consequence, an *implicit mapping* has been established between *$user_1$* and *$newuser_1$* respectively *$user_2$* and *$newuser_2$* as shown in *Figure 1*.

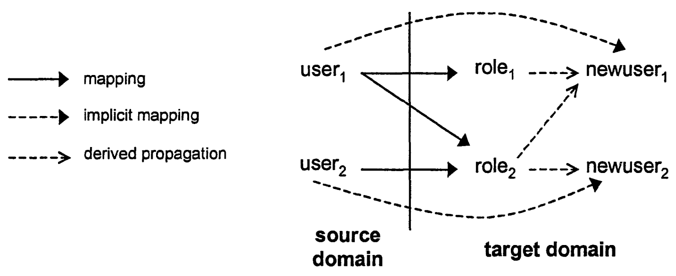

Figure 1. Simple-to-complex mapping

In some cases the relevant identities that should be implicitly mapped to identities of the source domain already exist at the target domain. For instance, imagine the presence of *$user_1$* at the target domain. By defining a condition that compares source to target identities for equality, *$user_1$* at the source domain could be automatically mapped to *$user_1$* at the target domain. Thus, *$newuser_1$* does not need to be created at the target domain neither needs to be deleted, if the mapping is removed. Moreover, conditions that are more sophisticated than evaluating equality (e.g. evaluating several attributes of the particular identities within regular expressions) can be utilized for retrieving the suitable target identities.

If identities at the source domain that participate in particular mappings respectively mappings themselves get removed, the relevant implicit mappings will be removed, too. Thus, the derived propagations as well as the newly created identities at the target domain must be deleted.

As mentioned above, complex identities alleviate the task of administrating authorization issues. Authorizations may be assigned to a complex identity thus propagating to all of the members of the particular complex identity. Furthermore, complex identities may be structured within a hierarchy providing further propagation (or inheritance) of authorizations. A *complex-to-simple* mapping from a complex identity at the source domain to a simple identity at the target domain specifies that all simple identities that are direct or indirect members of the complex identity at the source domain be implicitly mapped to the particular simple identity at the target domain.

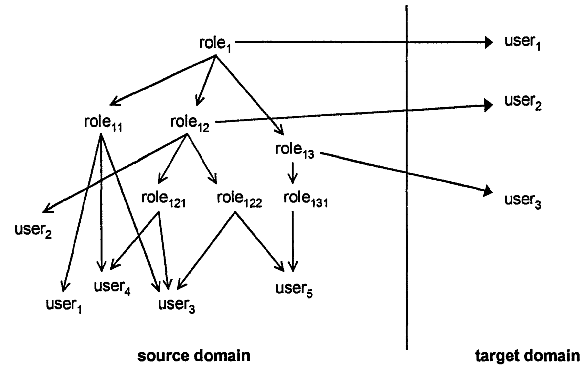

Figure 2. Complex-to-simple mapping

Figure 2 gives an example: the role-hierarchy at the source domain consists of a root-role *role$_1$* with three direct sub-roles *role$_{11}$*, *role$_{12}$* and *role$_{13}$*; *role$_{12}$* has two sub-roles again, namely, *role$_{121}$* and *role$_{122}$*, and *role$_{13}$* has one further sub-role, namely, *role$_{131}$*. There are five users at the source domain; *user$_1$* being a member of *role$_{11}$*, *user$_2$* being a member of *role$_{12}$*, *user$_3$* being a member of *role$_{11}$*, *role$_{121}$* and *role$_{122}$*, *user$_4$* being a member of *role$_{11}$* and *role$_{121}$*, and finally *user$_5$* being a member of *role$_{121}$* and *role$_{131}$*.

The example specifies three complex-to-simple mappings: *role$_1$* at the source domain is mapped to *user$_1$* at the target domain, *role$_{12}$* at the source domain is mapped to *user$_2$* at the target domain, and *role$_{13}$* at the source domain is mapped to *user$_3$* at the target domain. Because of the first mapping, all users at the source domain will be implicitly mapped to *user$_1$* at the target domain since they are indirect members of *role$_1$*.

Obviously, many kinds of conflicts may occur if it is ambiguous to which target identity a particular source identity should be effectively mapped. For instance, *user$_2$* is a direct member of *role$_{12}$* and simultaneously an indirect member of *role$_1$*, which results in a conflict, since *role$_1$* and *role$_{12}$* are mapped to different users at the target domain. The conflict can be resolved by computing the *level of indirection* (counting edges) from the simple source identity to the mapped target identity. The level of indirection from *user$_2$* at the source domain to *user$_2$* at the target domain is lower (=2) than to *user$_1$* at the target domain (=3) promoting *user$_2$* at the source domain to be mapped to *user$_1$* at the target domain.

However, this conflict cannot be resolved in the same way in case of *$user_3$* at the source domain who is indirectly mapped to *$user_1$* and *$user_2$* at the target domain with the same level of indirection. This kind of conflict can still be automatically resolved by computing the *strength of indirection* in counting the number of ways the target identity can be reached. In case of *$user_3$* at the source domain: the target identity *$user_2$* can be reached over *$role_{121}$* or *$role_{122}$* (two times) whereas *$user_1$* at the target domain can additionally be reached over *$role_{11}$* (three times). Thus, the strength of indirection is higher for *$user_1$* than for *$user_2$* leading to a mapping of *$user_3$* at the source domain to *$user_1$* at the target domain.

Some conflicts can still not be solved automatically as, for instance, the case of *$user_4$* having the same level of indirection as well as strength of indirection for *$user_1$* respectively *$user_2$* at the target domain. A further policy for resolving the conflict could be to choose the *least powerful role* (if a common root-role exists), which is *$role_1$* for *$user_1$* at the source domain. If none of the proposed policies can solve the conflict the definition of the particular mapping can either be prevented at administration-time, or handled at run-time by prompting the affected identity for its preference.

Since complex identities do not necessarily support hierarchies (e.g. flat groups) a mapping of that kind can be regarded as a trivial case of a complex-to-simple mapping.

A mapping from a complex identity at the source domain to a complex identity at the target domain is called *complex-to-complex* mapping. It can be resolved into various simple-to-complex mappings that are automatically maintained for any simple identity that is a direct or indirect member of the particular complex identity at the source domain.

3.3 Further Considerations

Complex identities represent an arbitrary number of simple identities (due to membership) that receive the authorizations associated to the particular complex identity. Thus, the authorizations propagate from the complex identity to its members. This propagation may either be static (e.g. in case of groups) or dynamic (e.g. in case of roles). For dynamic membership, the members have to explicitly activate the particular complex identity (e.g. a role) in order to receive its authorizations. Consequently, if a mapping is specified for a complex identity with dynamic membership, the mapping has to be established dynamically (i.e. at run-time), too. The currently activated set of complex identities can be a further means for resolving conflicts as mentioned above.

Identity mappings may also be specified at the identity type level (e.g. *User* or *Role*), saying that implicit mappings should be maintained for any identity within the extent of the particular identity type. The problem can be compared to *complex-to-simple* respectively *complex-to-complex* mappings as specified above. An identity type mapping additionally may define constraints concerning the *conditionality* and *cardinality* of the mapping. The former determines if a mapping *must* or *may* be established, the latter specifies if *one* or *more* instances can be mapped.

Some identity types require authentication (e.g. in case of human related identities) in order to prove the authenticity of the identity. Authentication mechanisms typically use an *identifier* (e.g. password, private key), which is an optional attribute that can be uniquely assigned to a determined identity. Consequently, mappings may be also specified for identifiers. For instance, an identity type mapping of users from a source to a target domain may generate random identifiers at the target domain in order to eliminate direct authentication of source identities at the target domain. In general, one of the following options is feasible for mapping identifiers: (1) *keep*: do not change the target identifier, (2) *equal*: set the target identifier equal to the source identifier, (3) *random*: create a random identifier at the target domain, (4) *default*: create a random identifier at the target domain, having the constraint to be changed by the identity as soon as first utilized, and (5) *one-time*: create random identifiers at the target domain that are automatically changed by the system as soon as being utilized.

In case of options 3 to 5 the source identities loose their autonomy within the target domain, since they can no longer directly authenticate as target identity.

Table 1. Options for identity mapping

source/target	**none**	**password**	**private key**
none	-	-	-
password	default, random, one-time	keep, equal, default, random, one-time	default, random, one-time
private key	random, one-time	random, one-time	keep, equal, random, one-time

Table 1 illustrates the possible combinations of identifier types together with their mapping options regarding three kinds of heterogeneity with respect to source and target identifiers, namely *none* (no identifier), *password*, and *private key*.

4. CONCLUSIONS

We have shown that identity mapping can be the basis of a new kind of security system that allows every application to use its built-in security features in contrast to existing solutions like directory services (e.g. LDAP) or distributed security systems (e.g. Kerberos, Sesame). Despite its multiple points of access control and auditing identity mapping can still provide a global interface for administration so that key requirements like single-sign-on, single-point-of-administration, and single-point-of-monitoring can be implemented. Building upon the theoretical concepts of this paper, the next steps will be to implement the "identity server" and its "mediators" for major information systems so that they can be integrated into an enterprise-wide security system. Another important issue is the design of a user interface to effectively manage the complex tasks of administering a security policy in a distributed system environment.

5. REFERENCES

Castano S., Fugini M., Mertella G., Samarati P. (1995) *Database Security*, Addison-Wesley.

Kohl J., Neuman C. (1993) The Kerberos Network Authentication Service, *RFC 1510.*

McMahon P.V. (1994) SESAME V2 Public Key and Authorization Extensions to Kerberos, *ISOC Symposium.*

Oppliger R. (1998) *Internet and Intranet Security.* Artech Haus Publishers, Norwood, MA, 1998. ISBN 0-89006-829-1.

Sandhu R.S., Coyne E.J (1996) Role-Based Access Control Models, *IEEE Computer*, Vol. 29, No. 2, Feb. 1996.

Sandhu R.S., Samarati P. (1996) Authentication, Access Control, and Audit. *ACM Computing Surveys*, Vol. 28, No. 1, March 1996.

Schneier B. (1996) *Applied Cryptography*, 2nd Ed., John Wiley & Sons, 1996. ISBN 3-89319-854-7.

Steiner J.G., Neuman C., Schiller J.I. (1988) Kerberos: An Authentication Service for Open Network Systems. Proc. Winter 1988 Usenix Conference, 1988.

Wahl M., Howes T., Kille S. (1997) Lightweight Directory Access Protocol (v3), *RFC 2251*, 1997.

Wiederhold G. (1992) Mediators in the Architecture of Future Information Systems. *IEEE Computer*, Vol. 25, No. 3, pp. 38-49, March 1992.

X.500 (1997) Information technology – Open systems Interconnection – The Directory: Overview of concepts, models and services, *International Telecommunication Union (ITU)*, 1997.

---

[i] This work has been funded by the Kplus program of the Austrian government, the province of Upper-Austria, and the Chamber of Commerce of Upper-Austria