SERAT : SEcure Role mApping Technique for Decentralized Secure Interoperability

Mohamed Shehab
School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN, USA
shehab@purdue.edu

Elisa Bertino
Department of Computer Sciences and CERIAS
Purdue University
West Lafayette, IN, USA
bertino@cs.purdue.edu

Arif Ghafoor
School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN, USA
ghafoor@purdue.edu

ABSTRACT

Multi-domain application environments where distributed domains interoperate with each other are becoming a reality in internet-based and web-services based enterprise applications. The secure interoperation in a multidomain environment is a challenging problem. In this paper, we propose a distributed secure interoperability protocol that ensures secure interoperation of the multiple collaborating domains without compromising the security of collaborating domains. We introduce the idea of access paths and access paths constraints. Furthermore, we device a path discovery algorithm that is capable of querying interoperating domains for the set of secure access paths between different domains.

Categories and Subject Descriptors

D.4.6 [**Security and Protection**]: Access controls; H.2.7 [**Database Administration**]: Security, integrity, and protection.

General Terms

Design, Security, Theory.

Keywords

decentralized secure interoperability, collaboration, access path, path discovery, Role based access control

1. INTRODUCTION

The rapid proliferation of Internet and related technologies has created tremendous possibilities for the interoperability between domains in distributed environments. Interoperability provides a means for domains to share resources and services, which enhances performance and resource utilization. For example, in peer-to-peer networks, peers collaborate with each other to provide efficient means for resource sharing. In distributed database environments interoperability enables users to access databases in different domains. Furthermore, interoperability is of great concern in several other areas such as in banking, medicine, government, military and in many other applications. Interoperability helps in bridging the gap between government to citizens, government to government, business to business and business to consumers.

However, interoperability does not come easy as it opens the way for several security and privacy breaches. Security is hard to achieve in a single domain let alone, secure interoperability in a multidomain environment [1]. Moreover, it is even harder to handle security in a dynamic interoperation environment where domains join and leave in an adhoc manner.

In order to address the security problems in an interoperation environment, we propose a distributed protocol that ensures the security requirements of such an environment. The challenge also lies in the fact that the collaborating domains could have heterogenous security constraints and this adds to the problem. In this paper we propose a protocol that provides a SEcure Role mApping Technique (SERAT) between domains. SERAT is a modular protocol that ensures secure, fair, dynamic and distributed interoperability between domains. SERAT assumes the user's access path is attached to user's access requests. SERAT provides path linking rules that ensure the secure path evaluation and updating. We also provide signature techniques that ensure the authenticity of the user's access path as it propagates between domains. Furthermore, SERAT is extensible in the sense that it enables additional domain specific constraints. We also provide an access path discovery technique that enables users to query the interoperation environment for the possible access paths. Moreover, SERAT also provides criteria for the selection from the set of possible access paths returned by the discovery technique. The selection criteria enables users to ensure higher levels of security in the selected paths according to their policies.

The rest of the paper is organized as follows. In Section 2 we discuss the requirements of secure interoperability, the maximal secure interoperability and the drawbacks of adopting the maximal secure interoperability solution. We introduce and define the access path in an interoperation in Section 3. The secure role mapping technique (SERAT), the different SERAT modules and the additional path constraints are discussed in Section 4. The access path discovery algorithm is presented in 5. Concluding remarks is added in Section 6.

SACMAT'05, June 1–3, 2005, Stockholm, Sweden.

2. SECURE INTEROPERABILITY

In this section we describe the general interoperability problem and present the solutions that are available in literature. To model the interoperability between different domains we assume that all the domains adopt a role-based access control (RBAC) model [4, 5]. RBAC is suitable for specifying the security requirements of commercial applications [3]. However, if a domain that does not use RBAC as its access control model wishes to join the interoperability session then it can easily provide an export RBAC policy. In RBAC, permissions are associated with roles, and users are granted membership in appropriate roles, thereby acquiring the roles permissions. The access control policy for domain i is modeled as a directed graph $G_i = \langle V_i, A_i \rangle$ where the vertex set V_i represents roles and the arcs set A_i represents the dominance relationship between roles. For example, if role r_1 dominates r_2 then $(r_1, r_2) \in A_i$, thus a user acquiring role r_1 can acquire permissions assigned to role r_2 by using the RBAC permission inheritance properties.

Given n secure systems, $G_i = \langle V_i, A_i \rangle$, $i = 1, 2, \ldots, n$, the interoperation between these systems is achieved by introducing cross domain pairwise mappings between n domains. Such mappings relate roles in different domains, and are represented by a set of cross-domain arcs referred to as the set F. The cross domain mappings are selected by the administrators of the domains according to the interoperability requirements of each system. Furthermore, the system administrators agree on a set of restricted accesses which is similar to negative authorizations adopted in several access control models. The restricted access is a binary relation R on $\bigcup_{i=1}^{n} V_i$ such that $\forall (u, v) \in R, u \in V_i, v \in V_j$, and $i \neq j$ where these edges are prohibited to exist during interoperation.

Given $G_i = \langle V_i, A_i \rangle, i = 1, \ldots, n$, a cross access relation F and a restricted access relation R, an interoperation $Q = \langle \cup_{i=1}^{n} V_i, A_Q \rangle$, where A_Q is the resulting arc set $A_Q \subseteq \{\cup_{i=1}^{n} A_i \cup F\}$, is secure if it satisfies all the following conditions [1]:

1. $A_Q \cap R = \emptyset$. the restricted access relation.
2. $\forall u, v \in V_i, (u, v)$ is legal in A_i if and only if (u, v) is legal in A_Q.

The conditions above ensure two important requirements of interoperability which are the *principles of autonomy and security.* The principle of autonomy requires that any access permitted within an individual domain must also be permitted under secure interoperation. The principle of security ensures that any access not permitted within an individual domain must also be denied under secure interoperation.

2.1 The Maximum Secure Interoperation (MSI)

Definition 1. *The maximum secure interoperation problem is defined as follows: For any positive integer $K \leq |F|$, determine whether a secure solution S exists such that $S \subseteq F$ and $|S| \geq K$.*

Simply, the MSI solution finds a maximal subset of the cross links set F such that the secure interoperability is ensured.Taking a closer look at the MSI solution we find that such a solution exhibits several drawbacks, which we discuss in this section.

- **NP-Completeness:** Li Gong et al.[1] shows a polynomial reduction of the Feedback Arc Set problem, which is a known NP-complete problem, to the MSI problem and thus proving that MSI is an NP-complete problem. Thus it is not practical to solve the MSI problem for a large number of collaborating domains. Moreover, any practical solution to this problem would be provided using heuristics and in such cases the generated solutions are approximate and are not the optimal solution(s).
- **Centralized Algorithm:** The MSI problem assumes full knowledge all domains $G_i = \langle V_i, A_i \rangle, i = 1, \ldots, n$, and the sets F and R. To solve the MSI problem a global view of the system is required and thus it has to be solved centrally. A trusted third party must exist that has the global view and computes the subset of F that satisfies the constraints of MSI. This solution is impractical in distributed environments with a large number of interacting parties.
- **Response to Changes:** A domain deciding to join or leave the interoperation after the MSI solution is computed would require the overall re-computation of the maximal solution. This is not a practical solution in a dynamic environment where domains are required to join and leave the interoperation environment transparently without the need for delays and revocations of current coalitions.
- **Fairness Issues:** The MSI solution violates fairness between the collaborating domains. The secure solution S is a subset of the set of cross links F, this means that to eliminate violations some cross links are discarded. However, in a violation several domains are involved and the removal of cross links may only affect a subset of these domains, especially in violations caused by cycles. The following example elaborates on the fairness issue. Consider Figure 1; a user in domain A having a role A_2 could access role A_3 by accessing roles B_2 and then B_1, finally accessing A_3 via B_1 this is clearly security violation. Furthermore, using a similar argument a user at B_1 could access role B_2. The MSI solution S would either include the edge (A_2, B_2) or (B_1, A_3). If the edge (A_2, B_2) is removed then users in domain A cannot access acquire any roles from domain B while on the other hand domain B users could still acquire roles from domain A; this is not a fair solution as it restricts accesses for users of domain A, whereas rights of users of domain B are left unmodified. Thus the MSI solution is optimal but does not guarantee fairness among all the collaborating parties in the interoperation.

Remark 1. *From the above discussion we conclude that the MSI solution is NP-Complete, it has to be solved centrally, static solution, and moreover it is not fair to all the participating domains. In the next sections we will propose a secure technique that is computationally simple, distributed, dynamic solution, and is ensure fairness to the participating domains.*

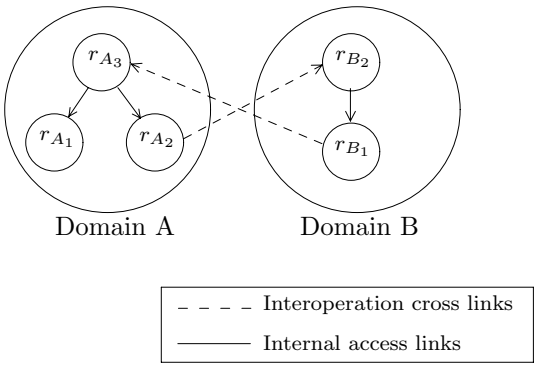


Figure 1: An example of two interoperating domains. The solid lines show the internal access links, while the dotted lines show the interoperation cross links F.

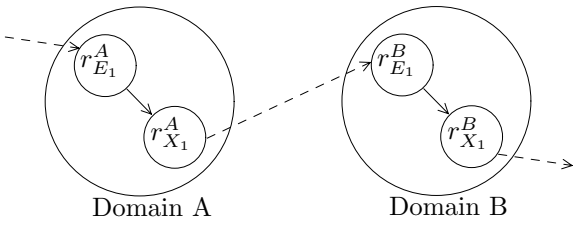


Figure 2: Shows two domains A and B. The solid lines show the internal access links, while the dotted lines show the interoperation cross links.

3. ACCESS PATHS IN AN INTEROPERATION ENVIRONMENT

In this section we introduce the idea of an access path in an interoperation environment and, explain what an access path is. We also discuss the types of cross links that are present in the interoperability environment.

DEFINITION 2. *The sequence in which a user*[1] *acquires roles in a multidomain environment during a certain session defines the user's access path.*

The access path starts from the user's *home domain* and ends at the user's *current domain*. The home domain is the domain to which a user originally belongs and at which the user's session starts. The domain where the user is currently present is referred to as the *current domain* and the domain the user is requesting access to is referred to as the *target domain*. When a user enters a domain he is assigned an *entry role* which is selected from the domain's role hierarchy according to the cross domain access agreements, which is referred to as the set F in previous sections. Similarly, an *exit role* describes the user's role in the current domain before accessing the target domain. Once a user acquires an entry role in the certain domain the user could acquire other roles in that domain according to the domain's role hierarchy, thus the entry and exit roles may differ. We use the following naming scheme to represent roles $r^{domain\_name}_{status}$ where $status \in \{(E)ntry, e(X)it\}$. To emphasize the definitions of entry and exit roles we provide the following example. Consider Figure 2; suppose that a user enters domain A and is assigned an entry role $r^A_{E_1}$. The user then connects to domain B, his exit role in A is $r^A_{X_1}$. The entry and exit roles describe the user's access path. For instance the user's access path would be described as follows $P = \{\ldots, r^A_{E_1}, r^A_{X_1}, r^B_{E_1}, r^B_{X_1}, \ldots\}$.

3.1 Insecure Paths

Insecure paths are paths that do not satisfy the security requirements of a secure interoperation mentioned in Section 2. The solution to the *maximum secure interoperation* problem ensures that all the paths present in the maximal set $S \subseteq F$ satisfy all the security requirements. However, the set S is generated by removing cross links to provide a maximal secure solution which significantly reduces interoperability. Clearly, the removal of cross links eliminates several other secure paths as well, which is a penalizing solution to several domains. Consider Figure 3 which shows an interoperation environment violating the security principle. A user in domain A having role r_{A_1} could follow the path $P_{violation} = \{r_{A_1}, r_{B_3}, r_{B_1}, r_{C_2}, r_{C_1}, r_{A_3}\}$, and clearly acquire role r_{A_3} which is at a higher level than his initial role r_{A_1}. The maximum secure interoperation solution is computed by removing one or more of the three cross links. Suppose that the cross link between r_{B_1} and r_{C_2} is removed; the removal of such link solves the security problem. However, this solution disconnects domains B and C; thus a legitimate user in domain B cannot connect to domain C which eliminates several other secure paths such as $\{r_{A_1}, r_{B_3}, r_{B_1}, r_{C_2}, r_{C_1}\}$ or any path that includes $\{r_{B_1}, r_{C_2}\}$. Thus the maximum secure interoperation problem unnecessarily affects involved domains by removing several secure paths in order to eliminate security violations. Undeniably, the solution involving the removal of links is not fair to all the domains.

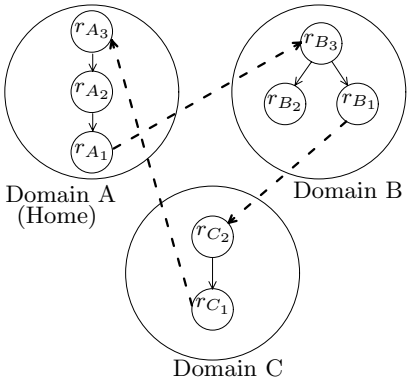


Figure 3: Example of a violation in a multidomain environment.

3.2 Types of cross links

Cross links are the main enabler of interoperability as they connect domains with each other. There are several types of cross links each type affects interoperability in different ways as will be discussed in the next sections. The main types of cross links are:

[1]A user could be a physical entity or an agent or even a web-service.

- **Explicit permitted access cross links *(P-Link)*:** These links describe the allowable interoperability links between different domains. The assignments are mutually decided by the domain administrators. The set F represents all such links.

- **Explicitly restricted access cross link *(R-Link)*:** These links describe the restricted interoperability links between domains. They are assigned by the domain administrators to avoid the collaboration between certain domains. The set R represents all such links.

- **Null cross links *(N-Link)*:** These links exist between domains that are not connected by any other types of links. Null links are used to indicate the permission of *open collaboration* between domains that can collaborate with one another via intermediate domains.

4. DISTRIBUTED DYNAMIC AND SECURE INTEROPERATION

In this section we present a distributed, dynamic solution for the secure interoperation problem. Our scheme utilizes the user's current access path to dynamically grant or refuse future access requests. The proposed solution uses ideas from the Chinese Wall [2], as the user's access history controls his future accesses. The user's view of the possible future paths is dependent on his current path; thus a more dynamic form of interoperation between the collaborating domains is achieved. Furthermore, our approach allows the complete set F to exist even if it contains insecure paths. However, our path evaluation technique ensures that insecure paths are not taken even though they are allowed to exist. The proposed technique performs secure role mapping between interoperating domains.

4.1 Overview of SERAT

Each of the interoperating domains runs SERAT to ensure secure interoperability. SERAT uses the user's access path to evaluate the access requests. SERAT is represented by a layered model, each layer provides clearly defined functions that are required by the overall protocol. Figure 4 shows the layers involved in the SERAT protocol. Understanding the function of each layer is instrumental in understanding the operation of SERAT in an interoperability environment.

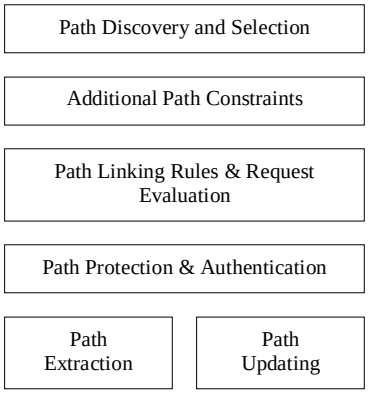


Figure 4: SERAT layered model.

When a user wishes to acquire a role in a target domain, the user sends a request to the current domain administrator. This domain administrator updates the user's path, signs the user's path, appends the user's signed path to the request and forwards the request to the target domain. Upon the reception of the request the target domain administrator extracts the user's path, checks the path's authenticity, and evaluates the user's request. Figure 5 shows the interaction between the different SEART modules when evaluating the user's request. The details of each module are discussed in the following sections. The domain administrators are responsible for request evaluation, path verification, evaluation and updates, thus our system model assumes system administrators to be trusted entities. It is also assumed that each domain is required to perform access control decisions for its resources only, which is a reasonable assumption.

A distributed database system is a practical example of our multi-domain interoperability. Multiple databases exist in different domains, and are linked via network connections. In a distributed database system, although a transaction will usually perform all of its actions at the home domain, it may also perform actions (or actions may be performed on behalf of it) at other domains. If this happens, an agent is created at the remote site to represent the transaction at that site. This agent becomes part of the original transaction for concurrency control and recovery purposes. A given transaction originates at one site but may migrate to one or more sites to execute its work. An agent represents the transaction at each site to which it migrates. Each site contains a SERAT module that is part of the site's access control system. This module evaluates requests and makes sure the interoperability security constraints are not violated.

4.2 Path Protection and Authentication

The user's access path is carried by the user's request as it migrates between domains. Figure 6 show an example of such user's request, which includes the user's access path and the path's signature. A path protection mechanism that is based on an authentication scheme is required to avoid the tampering of the user's access path and to enable the target domain to verify the authenticity of the presented user's access path. The proposed technique provides a signature that is generated by all the domains visited in the path. The signature preserves both the path entries and the path sequence. Every domain i has a public d_i and a private keys e_i. When a user decides to leave his current domain to connect to a target domain the path signature is generated by the current domain. The current domain concatenates the entry and exit roles with the current path and hashes them. The resulting hash is signed using the current domain's private key $e_{current}$. The generated hash is appended to the user's path. Equation 1 shows how the path signature (PS) is generated, where $||$ is the concatenation operator, $\oplus$ is the bitwise XOR operator, $HASH$ is a secure hash function, and $SIGN$ is the signature function. The signature could be done using modular exponentiation similar to techniques used in RSA signatures [8].

$$PS_{new} = SIGN_{e_{current}}(PS_{old} \oplus HASH(r_{entry}^{current} || r_{exit}^{current})) \quad (1)$$

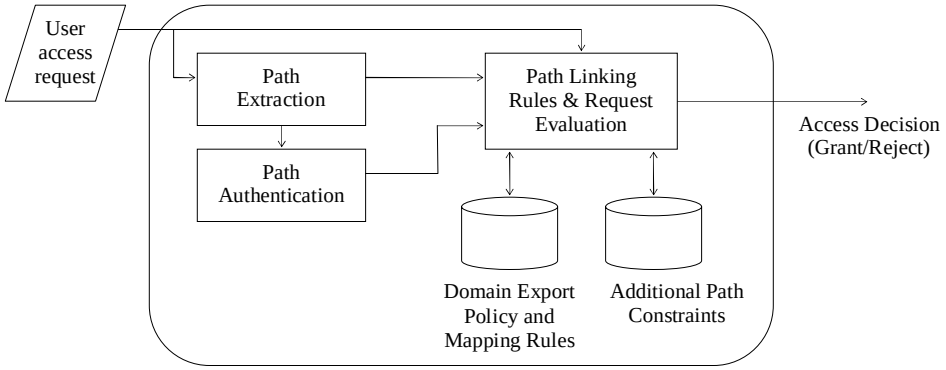


Figure 5: SERAT's access request evaluation module.

The target domain receives the path and its signature with the user's request. The target domain can easily verify the authenticity of the path by using the signature, the public keys and the path information. The target domain computes the hash of the entry and exit roles from the path information and recursively applies equation 2, and thus verifying the authenticity of the provided access path.

$$PS_{old} = SIGN_{d_{current}}(PS_{new}) \oplus HASH(r_{entry}^{current} \,||\, r_{exit}^{current}) \quad (2)$$

A path signature generation and verification example is shown in Figure 7. This design ensures that a path entries are only added by visited domains, because of signature using the private keys. This makes it very hard for an attacker to insert entries in the path as knowledge of the corresponding private keys is required. Thus this scheme ensures that the path is generate only by authorized domains.

4.3 Path Linking Rules

After verifying the authenticity of the user's path the target domain checks the user's request to acquire a role. The path linking rules are a set of rules used in deciding whether a request is to be permitted or rejected. In this section we present two types of path linking rules the *flexible* and the *strict* interoperability path linking rules. The three main domain types referred to are namely the *home domain* (H), *current domain* (C), and the *target domain* (T). Figure 8 shows the three main types of domains and the paths and links between them.

DEFINITION 3. *The secure path linking problem is defined as follows: Given a secure path $Path_i$ and a link $Link_i$, does the addition of $Link_i$ to $Path_i$ generate a secure path that does not violate the restricted set R and the principles of autonomy and security.*

4.3.1 Flexible Path Linking Rules

These rules allow null cross links to exist and are used as a methodology for *open interoperation*. That is, if a null cross link exists between two domains, then such domains are willing to interoperate with one another through other domains as long as a secure path between them can be established. This approach does not limit the interoperability between

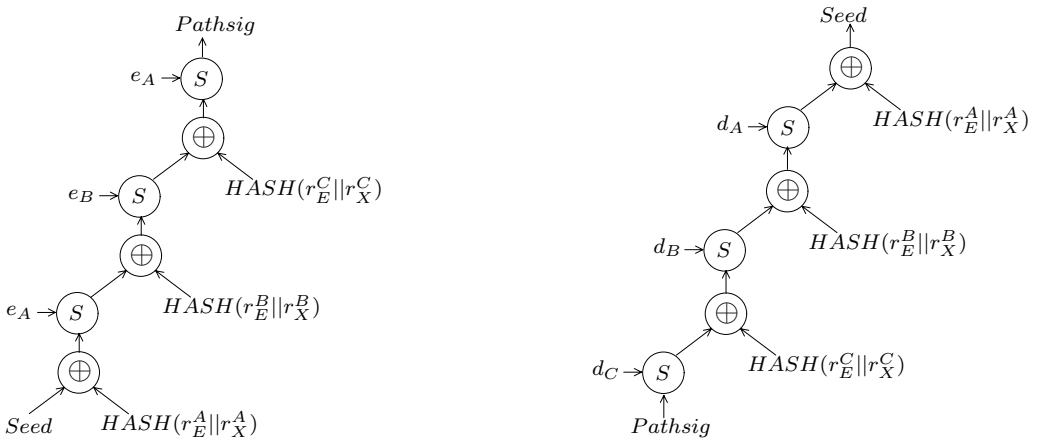
```
<UserRequest>
   <RequestedRole>
      roleA3
   </RequestedRole>
   <Path>
      <Domain name = "A" index = "1">
         <EntryRole>
            roleA1
         </EntryRole>
         <ExitRole>
            roleA1
         </ExitRole>
      </Domain>
      <Domain name = "B" index = "2">
         <EntryRole>
            roleB3
         </EntryRole>
         <ExitRole>
            roleB1
         </ExitRole>
      </Domain>
      <Domain, name = "C" index = "3">
         <EntryRole>
            roleC2
         </EntryRole>
         <ExitRole>
            roleC1
         </ExitRole>
      </Domain>
      <PathSignature>
         !A@GXYZ190FPH
      </PathSignature>
   </Path>
</UserRequest>
```

Figure 6: A request submitted by a user having the access path of user in Figure 3 and is trying to connect to role r_{A_3}

domains. Moreover, these rules ensure the generation of secure access paths. We now formally state the flexible path linking rules.

Flexible Path Linking Rules: Let $Path_i$ be a secure path and let $Link_i = (R_{X_a}^C, R_{E_b}^T)$ be a link. The concatenation of $Path_i$ and $Link_i$ is denoted as $Path_i||Link_i$, must verify the following conditions:

1. $Link_i \in F$.
2. If $T \in Path_i$, where $T.Path_i$ is the set of roles associated with domain T in path $Path_i$, then all roles $S \in T.Path_i$ must satisfy the condition that $R_{E_i}^T \leq S$.
3. The link should not introduce any violation to the re-



(a) Generating Path Signature (b) Signature Verification

Figure 7: The signature generation and verification for $Path = \{r_E^A, r_X^A, r_E^B, r_X^B, r_E^C, r_X^C\}$ and S represents the signature function used.

stricted set R, thus link $Link_i \notin R$ and for all roles $S \in Path_i$ must satisfy $(S, R_{E_b}^T) \notin R$.

Next Theorem proves that the flexible path linking rules assure the correctness of the path computed with respect to the security and autonomy principles.

THEOREM 1. *Let $Path_i$ a path which is correct with respect to the principles of security and autonomy. Let $Path_{i+1} = Path_i||Link_i$ be a concatenation of $Path_i$ and $Link_i$ that verify the flexible path linking rules. Then $Path_{i+1}$ is correct with respect to the principles of security and autonomy.*

We now prove that if $Link_i$ is added to $Path_i$ after satisfying all the above rules then the new path $Path_{i+1} = Path_i||Link_i$ does not violate the principles of security and autonomy.

PROOF. The initial path $Path_i = (r_1^E, r_1^X, \ldots, r_n^E)$ is secure. We proceed using the proof by contradiction. Assume to the contrary that the new path $Path_{i+1}$ is not secure after satisfying all the above rules. If this is the case, then this means there exists a violation in path $Path_{i+1} = Path_i||\{r_n^X, r_{n+1}^E\}$. This violation can be due to $Path_i$ or (r_n^X, r_{n+1}^E) or (s, r_n^X) or (s, r_{n+1}^E), where $s \in Path_i$. Since $Path_i$ is the initial path and it is assumed to be secure then it cannot contain a violation. Rules 1 and 3 check for $(r_n^X, r_{n+1}^E) \in F$ and $(r_n^X, r_{n+1}^E) \notin R$ respectively thus this link cannot be the cause of the violation. We are now left with only two links namely (s, r_n^X) and (s, r_{n+1}^E), however rule 2 checks the integrity of adding such links and insures the it does not violate the ordering among the roles in the domain's internal roles hierarchy, thus these links cannot be the cause of any security or autonomy violations. In this case as all the possible links that could lead to a violation have been proven to be secure after passing the above rules which contradicts our assumption and thus path $Path_{i+1}$ can only be a secure path. □

4.3.2 Strict Path Linking Rules.

The strict path linking rules forbid the presence of null cross links. That is, two domains can interoperate only if F contains cross links between them. This requirement is very strict and hinders interoperability between the domains. However, this requirement allows system administrators to clearly specify and limit the interoperability between domains. The strict path linking rules presented in what follows ensure that the *strict path property* holds.

DEFINITION 4. *The strict path property states that the privileges gained by taking any path from the home domain to the target domain through intermediate domains are a subset of the privileges gained through a direct path from the home domain to the target domain.*

Strict Path Linking Rules: Let $Path_i$ be a secure path and let $Link_i = (R_{X_a}^C, R_{E_b}^T)$ be a link. The Path $Path_i||Link_i$, must verify the following conditions:

1. $Path_i$ and $Link_i$ must verify the Flexible Path Linking Rules.
2. A direct link must exist from the home domain to the target domain $Link_{direct} = (R_{X_i}^H, R_{E_i}^T)$, where $R_{E_b}^T \leq R_{E_i}^T$ and all roles $S \in H.Path_i$ must satisfy the condition that $R_{X_i}^H \leq S$. Where $H.Path_i$ is the set of roles associated with domain H in path $Path_i$.

REMARK 2. *Rule 4 does not allow the null cross links to exist and restricts interoperability. The direct path is an upper bound on the interoperation between domains.*

Next Theorem proves that the strict path linking rules assure the correctness of the path computed with respect to the security and autonomy principles.

THEOREM 2. *Let $Path_i$ a path which is correct with respect to the principles of security and autonomy. Let $Path_{i+1} = Path_i||Link_i$ be a concatenation of $Path_i$ and $Link_i$ that verify the strict path linking rules. Then $Path_{i+1}$ is correct with respect to the principles of security and autonomy. Furthermore, $Path_{i+1}$ satisfies the strict path property.*

We now prove that if $Link_i$ is added to $Path_i$ after satisfying all the above rules then the new path $Path_{i+1} =$

$Path_i||Link_i$ does not violate the strict path property and principles of security and autonomy.

PROOF. The $Path_{i+1}$ satisfies the flexible path linking rules and thus it is guaranteed to be correct with respect to the principles of security and autonomy, the proof follows from the proof of flexible path linking rules. Rule 2 requires the presence of a direct link, and requires that $R^T_{E_b} \leq R^T_{E_i}$ this ensures that role gained via the direct link in the target domain dominates role gained through $Link_i$. The second part of rule 2 ensures hierarchical consistency in the home domain by requiring all the roles $S \in H.Path_i$ to satisfy the condition $R^H_{X_i} \leq S$. This ensures that the direct link exit role is dominated by all the roles associated with domain H in path $Path_i$. According to the requirements of rule 2 the strict path property is satisfied. □

4.4 Additional Types of Constraints on Paths

Our path-based approach naturally lends itself to the specifications of several interesting classes of constraints, that we briefly discuss in what follows. These constraints can be easily added to the path evaluation to enhance the security requirements of the path.

- ***Separation of Duty (SoD) Constraints*:** Separation of Duty has been investigated extensively in Role-Based Access Control (RBAC) [4, 5]. RBAC uses mutual exclusion constraints to implement SoD policies. The SoD constraints could be divided in two types, namely the *static* and *dynamic* mutually exclusive roles (SMER) and (DMER) respectively [7]. SMER constraints specify that two or more mutual exclusive roles must never be assigned to the same subject simultaneously. An example of a SMER constraint is that "no user is allowed to be a member of $r^a_{D_1}$ and $r^b_{D_2}$ simultaneously". DMER constraints specify that two or more mutual exclusive roles must never be, activated by the same subject simultaneously. This means that two dynamically mutual exclusive roles may be assigned to the same subject. However, the subject is only allowed to activate at most one of its dynamically mutual exclusive roles (permissions) at the same time. A more general type of DMER requires that no user be a member of t or more roles in a set of m roles $\{r_1, r_2, \ldots, r_m\}$ in a given session. In this paper we are more concerned with the dynamic constraints; thus we discuss only the DMER constraints. These constraints can be resolved dynamically by checking the access path of the user to ensure that the DMER constraints are not violated before a domain grants access to a certain role. The SoD constraints could be applied on roles in the same and/or in different domains. These could be easily added as an extra check by the target domain as the access path is also provided with the user request.
- ***Bound on Number of Domains*:** Given n collaborating domains, this constraint imposes a bound m on the number of allowed domains in a user's access path, where $m \leq n$. Furthermore, this constraint helps in limiting the reach of users in the interoperability environment. For example if the limit is set to $m = 2$, then only a pairwise interoperation between domains is possible and the user cannot migrate multiple times. This constraint is also useful if the administrators want to avoid security breaches if domains are collude against other domains.
- ***Path Ordering Constraints*:** Before allowing $Link_i$ to be connected path $Path_i$, this path should contain a certain sequence of accesses. This is a useful constraint in workflow environments where the user is required to perform tasks in a certain order and sequence. Furthermore, this constraint could also accommodate the negative approach where if the path contains a certain sequence of roles or domains, then access is to be denied.

The SERAT protocol provides path linking rules that ensure that the permitted paths for a user in a session are all secure and satisfy the principles of security and autonomy. We have proved that path security is guaranteed if the path linking rules are performed by each domain. Moreover, the paths are also guaranteed to satisfy the additional path constraints which are extensible domain specific constraints. Note that, all the proposed path evaluation rules and constraints only depend on the user's path and the access constraints at the target domain. Consequently, the target domain has all the required information to evaluate the user's request. Thus, the path verification and request evaluation is inherently a distributed algorithm. SERAT also allows all the cross links to exist and thus it does not unfairly remove any cross links as in the case of MSI solution [1]. Thus SERAT provides a fair solution by avoiding the isolation of domains which is caused by the removal of cross links.

5. PATH DISCOVERY

A user residing in a certain domain would like to examine the set of possible paths from his current domain to a certain role in a target domain. *Path discovery* allows a domain in an interoperation environment to dynamically discover a path to any other role in another domain, whether directly reachable or indirectly reachable through one or more intermediate domains. The path discovery algorithm is a distributed algorithm that returns the set of secure paths (if any exists) from the source domain to the target domain. Furthermore, we add to the complexity of the path discovery by allowing the user to provide a list of domains that should be included in the path discovery. We also provide optimality metrics that are used to choose paths from the set of paths returned by the discover algorithm. As will be noted this approach is similar to on-demand routing but it is more complicated as it includes several other constraints such as the path linking rules and constraints. We now discuss the main steps involved in the path discovery protocol.

5.1 Neighborhood maintenance

Neighborhood information is very important in a dynamic interoperation environment where domains dynamically join and leave the system. These neighborhood updates provide the connectivity information between roles in different domains, which is basically information about the set F. This information is critical for path and request evaluation. To maintain this neighborhood information, every domain is required to periodically send out a "Hello" message to all the domains that share cross links with it. The Hello message announces the existence of the domain and informs

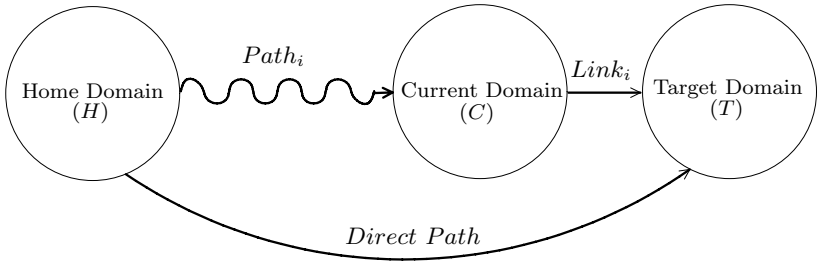


Figure 8: Types of domains and $Path_i$ and the new link $Link_i$.

the neighbors about the cross links associated with this domain. Domains receiving the the Hello messages from its neighbors maintain a neighbors list which includes all its neighbors with their corresponding cross links. The time-to-live (TTL) of such messages is set to 1 as it is only sent to neighboring domains.

5.2 Path Discovery

The proposed approach allows the path to be discovered on-demand by propagating the path request and the route reply messages between the source and the destination. Our path discovery algorithm is implemented by route exploration from source to destination. The path establishment is done on-demand using limited flooding. When a domain (source) needs to establish a path with a target (destination) which is not in the source's neighborhood list, the source broadcasts a path request message to all its neighbor domains. The path request message includes the destination domain and the requested role. Upon receiving the path request message, the receiving domain performs a path evaluation. If the path is accepted then the domain updates the path and rebroadcasts the request through all the cross links that are linked to the domain excluding the domains that are already included in the path. This constraint avoids possible loops during path exploration and reduces the number of message broadcasts.

5.3 Path Selection

The source waits for a timeout period of T_{max}, and if no reply arrives from the destination domain then this means there are no secure paths from source to the destination. On the other hand, if a reply is received from the destination before the timeout period then it will contain the set of all possible secure paths from the source to the destination. We refer to such set as $S_{s,d}$. However, the source selects one path according to certain selection criteria. The selection criteria is based on the path properties which include:

- ***Path length***: The source could select the path having the shortest length in terms of the number of visited domains.
- ***Path visited domains***: Select the path a path that contains certain set of domains, or certain sequence of domains.
- ***Path selection based on composite domain reputation***: Domains could be given reputation metrics and the path reputation is computed using the domains included in the path, and the path having the highest reputation be selected.

6. CONCLUSIONS

In this paper, we have presented a distributed, dynamic protocol that provides a solution for secure interoperability between domains. The protocol assumes the user's access path information propagates with user access requests. Path linking rules have been defined to ensure that path linking is executed securely. We also provided a path signing scheme that ensures that the path is tamper proof as it propagates between domains.

Furthermore, we have presented additional path constraints that could be adopted by different domains to regulate the degree of interoperability and thus making SERAT an extensible protocol. We also provided a path discovery algorithm that enables users to enumerate paths to a certain role in a target domain. We also provided path selection criteria that could be used when selecting a path from the set of possible paths.

7. ACKNOWLEDGMENTS

Portions of this work have been supported by the sponsors of the Center for Education and Research in Information Assurance and Security (CERIAS) at Purdue University, and the National Science Foundation under NSF Grants IIS-0209111 and IIS-0242419. This research of Elisa Bertino was supported in part by the NSF under the Project "The Design and Use of Digital Identities", by an IBM Fellowship, and by the sponsors of CERIAS at Purdue University.

8. REFERENCES

[1] L. Gong and X. Qian, "Computational Issues in Secure Interoperation," *IEEE Transaction on Software and Engineering.,* Vol. 22, No. 1, January 1996.

[2] D.F.C. Brewer and M.J. Nash, "The Chinese Wall Security Policy," *In Proceedings of IEEE Symposium on Security and Privacy.,* pages 206-214, 1989.

[3] E. Bertino, E. Ferrari, and V. Atluri, "The Specification and Enforcement of Authorization Constraints in Workflow Management Systems," *ACM Transactions on Information and System Security.,* 2(1):65-104, February, 1999.

[4] D. F. Ferraiolo, R. S. Sandhu, S. Gavrila, D. R. Kuhn, and R. Chandramouli. "Proposed NIST Standard for Role-Based Access Control," *ACM Transactions on Information and Systems Security.,* 4(3):224274, Aug. 2001.

[5] D. F. Ferraiolo, D. R. Kuhn, and R. Chandramouli. "Role-Based Access Control," *Artech House*, Apr. 2003.

[6] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E.

Youman. "Role-based access control models," *IEEE Computer,* 29(2):3847, February 1996.

[7] Ninghui Li, Ziad Bizri, Mahesh V. Tripunitara, "On Mutually Exclusive Roles and Separation of Duty," *ACM Conference on Computer and Communications Security*, October 2004.

[8] R. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Communications of the ACM*, 21 (2), pp. 120-126, February 1978