Proactive Role Discovery in Mediator-Free Environments

Mohamed Shehab
UNC-Charlotte
Charlotte, NC, USA
mshehab@uncc.edu

Elisa Bertino
Purdue University
West Lafayette, IN, USA
bertino@cs.purdue.edu

Arif Ghafoor
Purdue University
West Lafayette, IN, USA
ghafoor@ecn.purdue.edu

Abstract

The rapid proliferation of Internet and related technologies has created tremendous possibilities for the interoperability between domains in distributed environments. Interoperability does not come easy at it opens the way for several security and privacy breaches. In this paper, we focus on the distributed authorization discovery problem that is crucial to enable secure interoperability. We present a distributed access path discovery framework that does not require a centralized mediator. We propose and verify a role routing protocol that propagates secure, minimal-length paths to reachable roles in other domains. Finally, we present experimental results of our role routing protocol based on a simulation implementation.

1. Introduction

Globalization has removed the barriers between markets, organizations, researchers and societies. In such a connected world, there are immense opportunities for collaboration in distributed environments. For example, enterprises are continuously splitting processes into several tasks across organizational boundaries to combine their efforts and become virtual enterprises [10, 20]. Furthermore, in recent years there has been an increasing demand to allow scientific institutions to collaborate in the management and analysis of the vast quantities of data generated by scientific experiments [12, 1].

With all the advantages that multi-domain collaboration is promising to offer it does not come easy as it opens the way for several security breaches. Security is hard to achieve in a centralized system [13, 14, 25], let alone in a dynamic distributed mediator-free environment. In a mediator-free collaboration environment there is no central trusted entity having a global view of the collaboration environment and handling security. Instead, domains have a limited view of the collaboration environment through their neighboring domains. In such an environment, domains collaborate to find resources and authorizations required to access resources across domain boundaries. For example, in distributed database environments interoperability enables users to access databases in different domains; but how can a domain acquire the required set of authorizations that enable access to the requested remote databases?

Discovering authorizations in mediator-free environments where none of the domains has a global view of the collaboration environment is a challenging task. In this paper, we present a role routing protocol (RRP) that efficiently propagates optimal access path information between domains. In RRP domains are not required to have a global view of the collaboration environment; instead domains only interact with their neighboring domains. RRP discovers secure access paths that have minimal length. The main contributions of the paper are summarized as follows:

- We propose a role routing protocol (RRP) that enables domains to propagate information about secure, minimal-length paths to destination roles. We show correctness and security of our role routing protocol.
- With a proof of concept implementation of role routing protocol we conducted experiments to compare our RRP with the shortest path and the flooding protocols.

The rest of the paper is organized as follows. Section 2 describes some preliminary concepts to facilitate background for the rest of the paper. Section 3 describes and analyzes the operation of our proposed role routing protocol. The experimental results are presented in Section 4. The related work is discussed in Section 5. Finally, we present our conclusions in Section 6.
2. Preliminaries

In our framework, we assume that all the domains adopt a role-based access control (RBAC) model [11, 7] to model their access control policies. The analysis presented in this paper can still be applied when other access control models are adopted. We have chosen RBAC because it is suitable for specifying the security requirements of a wide range of commercial, medical, government applications [23, 3, 2] and moreover it is being standardized by the National Institute of Standards [11]. A domain that does not use RBAC as its access control model can easily generate an export RBAC policy to join the collaboration. In RBAC, permissions are associated with roles, and users are granted permissions to join the collaboration. In RBAC, permissions are associated with roles, and users are granted permissions to join the collaboration. In RBAC, permissions are associated with roles, and users are granted permissions to join the collaboration. In RBAC, permissions are associated with roles, and users are granted permissions to join the collaboration. In this context, the access control policy can be expressed in a wide range of commercial, medical, government applications [23, 3, 2] and chose RBAC as its access control model.

2.1. Secure Interoperability

Collaboration among domains involves the interoperability of their access control policies. Domains typically achieve interoperability among their access control policies by introducing cross mappings between these policies [13, 14]. These mappings relate roles in different domains, and are represented by a set of cross domain arcs referred to as the set $F$. We will refer to such mappings as cross links. A cross link $(r_x, r_y)$, indicates that a user acquiring the role $r_x$ in domain $D_1$ can acquire role $r_y$ in domain $D_2$. Figure 1, shows the crosslinks as dotted edges connecting roles in neighboring domains. In the present work we assume that the cross domain mappings are selected by the administrators of the domains according to the interoperability requirements of each system. These links could be selected when the service level agreements (SLA) are negotiated [28, 8]. Furthermore, the domain administrators agree on a set of restricted cross links that are prohibited during the collaboration. These restricted access links are similar to negative authorizations adopted in several access control models [5, 11, 6, 7]. 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 in $R$ are prohibited during interoperating.

For example in Figure 1, if $(r_A, r_B) \in R$ then a user acquiring role $r_A$ is not permitted to acquire role $r_B$. Gong et al. [13, 14] proposed a solution to ensure secure interoperability. However, this solution is centralized, assumes a static environment, and is computationally inefficient.

2.2. Mediator-Free Secure Interoperability

In a mediator-free collaboration environment there is no central mediator or trusted party managing and ensuring secure interoperability among the collaborating domains. In such an environment, none of the collaborating domains has a global view of all the access control policies; instead the domains view the collaboration environment only through their established cross links. Domains have to collaborate in making access control decisions and to avoid possible violations. Shehab et al. [27, 26] presented a framework for secure collaboration in mediator-free environments. This framework is based on using the access history to dynamically make access control decisions. The access history is encoded in the access path, which is the sequence of roles acquired from the home domain to target domains in the collaboration environment. Using the access path to make access control decisions shares ideas with the Chinese Wall security policy [4]. The access path enables domains to make local access control decisions and to avoid violations without the need for the global view of the collaboration environment. A secure access path is defined as follows:

**Definition 1** Let $P = \{r_1, r_2, \ldots, r_n\}$ be an access path, where $i < j$ implies that role $r_i$ was acquired before $r_j$. Let $D(r_i)$ denote the domain of role $r_i$. The path $P$ is secure if it satisfies the following conditions:

C1. For all $i < j$ and $r_i, r_j \in P$, if $D(r_i) = D(r_j)$ then $r_j \not\leq r_i$.

C2. For all $r_i, r_{i+1} \in P$, if $D(r_i) \neq D(r_{i+1})$ then $(r_i, r_{i+1}) \in F$.

C3. For all $i < j$ and $r_i, r_j \in P$, $(r_i, r_j) \not\in R$.

Where $r_i \not\leq r_j$ means $r_i$ dominates $r_j$. Condition C1 ensures that roles acquired from the same domain are acquired according to the domain’s role hierarchy. This ensures that the access control policies of the domains included in the path are not violated. Conditions C2 and C3 ensure that sets $F$ and $R$ are honored.
3. Role Routing Protocol (RRP)

Domains in a mediator-free environment have a limited view of the collaboration environment through their neighboring domains with which they have established cross links. Domains also know about the restricted access relations that they are involved in. Collaboration with non-neighboring domains is made possible by building secure access paths through neighboring domains. In order to enable such multi-hop collaboration, a distributed algorithm is needed to discover and maintain role reachability information between the collaborating domains. In this section we present our distributed protocol that enables secure role routing.

3.1. Role Routing Problem Definition

Cross links are the main enablers of collaboration between domains. A cross link \((r_x, r_y)\) starts at an exit role \(r_x\) and ends at an entry role \(r_y\); we say role \(r_y\) is an out-neighbor of role \(r_x\), and that role \(r_x\) is an in-neighbor of role \(r_y\). An internal role is a role that is neither an exit nor an entry role. We define the path length of a path \(P\), referred to as \(l(P)\), as the number of cross links it contains. The role routing problem is defined as follows:

**Definition 2** Given \(n\) collaborating domains \(D_1, \ldots, D_n\), a set of cross links \(F = \{F(D_1), \ldots, F(D_n)\}\) and a set of restricted links \(R = \{R(D_1), \ldots, R(D_n)\}\) where \(F(D_i)\) and \(R(D_i)\) denotes the set of cross links and restricted links of domain \(D_i\) respectively, find for each domain \(D_i\) the optimal paths from its exit roles to entry roles of other domains, where an optimal path from role \(r_x\) to \(r_y\) is a secure path whose length is less than or equal to that of any other path from \(r_x\) to \(r_y\).

The role routing problem satisfies the Principle of Least Privilege [22], which requires that each principal be accorded the minimum access privileges needed to accomplish its task. In the context of secure collaboration, the principle of least privilege implies choosing access paths that minimize the path authorizations which are consequently the shortest secure paths. A major challenge in role routing is the presence of restricted access relations between roles in different domains. Figure 2, shows an example collaboration environment, where there is a restricted access relation \((r_1, r_3)\). This implies that roles \(r_1\) and \(r_3\) are prohibited to coexist in a secure path. The shortest path from role \(r_1\) to role \(r_4\) is path \(P_1 = \{r_1, r_2, r_3, r_4\}\), however this is not an optimal because it is not secure as it includes both \(r_1\) and \(r_3\). The path \(P_2 = \{r_1, r_2, r_5, r_6, r_7, r_8, r_4\}\) is optimal as it is the shortest secure path from \(r_1\) to \(r_4\). Another interesting observation is that the optimal path from \(r_2\) to \(r_4\) is \(P_3 = \{r_2, r_3, r_4\}\) which does not overlap with optimal path from \(r_1\) to \(r_4\). These observations imply that when designing a distributed role routing protocol in which domains advertise paths to their neighbors, they may have to advertise multiple paths to the same destination. This is unlike network routing protocols which require only the advertisement of the “best” path to a certain destination [24]. In Figure 2, role \(r_2\) is aware of two paths to \(r_4\). In the context of this example, both of these paths should be advertised to \(r_1\) to enable \(r_1\) to discover its optimal path \(P_2\).

![Figure 2. Optimal path, where \((r_1, r_3) \in R.\)](image)

3.2. Summary of Operation

The role routing protocol (RRP) is a distributed path-vector protocol [18, 21] whereby the basic unit of information stored at the domains and exchanged between them is in the form of an access path. In RRP, domains send path information updates to their neighboring domains, specifically their in-neighboring domains. For example in Figure 2, domain \(D(r_2)\) sends path updates to domain \(D(r_1)\), to describe roles reachable via the cross link \((r_1, r_2)\). The role routing protocol discovers optimal paths to roles in different domains. Domains running RRP maintain and store paths in three local path information tables (PIT)s; PIT-Out, PIT-In, and PIT-Loc. Paths that are advertised to neighboring domains are stored in PIT-Out, paths that are received from neighboring domains are stored in PIT-In, and optimal paths to entry and exit roles reachable from the local exit roles are stored in PIT-Loc. The PIT-Loc is a partial map of the collaboration environment representing the view with respect to the current domain. RRP does not require domains to retransmit the entire set of previously advertised paths, instead path advertisements are sent to neighboring domains incrementally and only in response to path updates. Upon receiving path advertisements, domains update their path information tables accordingly and propagate relevant changes to their neighboring domains. In response to path failures, domains
send path withdrawal messages to their neighboring domains. Domains ensure the alive status of neighboring domains by periodically sending Keep-Alive messages. Paths advertised by domains are loop free as RRP ensures that paths sent from domain \( D_i \) to domain \( D_j \) should not include roles from domain \( D_j \).

To enable domains to identify the set of potential optimal paths to advertise to their neighboring domains, RRT requires domains to mark roles in paths that have potential violations. The set of potential violators is defined as follows:

**Definition 3** The set of potential violators of a path \( P \), denoted by \( pv(P) \), is the set of roles \( r \in P \) such that \( r \) is involved in a restricted access relation of the form \( (r_x, r) \in R \) where \( r_x \) is any role not in the same domain as \( r \).

A domain updates the set of potential violators of a path before advertising it to its neighboring domains. For example in Figure 2, if path \( P = \{r_3, r_4\} \) is to be advertised by domain \( D(r_3) \) to domain \( D(r_2) \), then domain \( D(r_3) \) would mark role \( r_3 \) in path \( P \) to indicate that role \( r_3 \) might be involved in a violation, as \( (r_1, r_3) \in R \). Propagating the set of potential violators of a path enables domains to make smart decisions when choosing paths to forward to neighboring domains.

![Figure 3. Path information tables.](image)

**3.3. Handling Path Advertisements**

Received path advertisements contain a set of new paths reachable through the advertising domains. When a domain \( D_i \) receives a path advertisement \( AdvIn \) from domain \( D_j \) via cross link \( (r_i^X, r_j^E) \), the new paths in \( AdvIn \) are stored into PIT-In only if these paths do not violate the restricted access relations of domain \( D_i \). Figure 4 shows the \( InputFilter() \) algorithm that is executed by domain \( D_i \) upon receiving an advertisement. The \( InputFilter() \) algorithm generates a list of secure paths from \( AdvIn \), which we refer to as \( FilteredAdv \). Then the \( InputFilter() \) stores the paths in \( FilteredAdv \) into PIT-In. The updates in PIT-In should be propagated to neighboring domains and to PIT-Loc respectively.

![Figure 4. Algorithms executed by domain \( D_i \) upon receiving path advertisements.](image)
3.4. Handling Path Withdrawal

Path withdrawal messages are sent by a domain to notify neighboring domains that a set of previously advertised paths are no longer reachable. Upon receiving a path withdrawal message, domain \( D_i \) should remove all withdrawn paths from its PIT-In. Furthermore, domain \( D_i \) should inspect its PIT-Out to see if any of the withdrawn paths were previously advertised to its neighboring domains. If some of the withdrawn paths have been advertised previously, then domain \( D_i \) should perform the following steps:

S1. Domain \( D_i \) should send a path withdrawal message to domains it has previously advertised the withdrawn paths.

S2. The deleted set of withdrawn paths might affect the set of potential optimal paths. Domain \( D_i \) should run a procedure similar to the \( \text{OutputFilter}() \) algorithm to find a new set of replacement paths to the same destinations as the withdrawn paths. Then domain \( D_i \) should send path advertisements with replacement paths to its neighboring domains.

S3. If any of the withdrawn paths is included in PIT-Loc, then this path should also be removed from PIT-Loc. Then domain \( D_i \) should search for an optimal path to the same destination in PIT-Out, which is a procedure similar to the \( \text{LocalFilter}() \) algorithm.

Figure 5 shows the \( \text{FindAdvertised}() \) and \( \text{ReFilter}() \) algorithms that perform the steps described above.

3.5. Correctness Analysis

In this section, we prove the correctness of RRP. For a path \( P = \{r_n, \ldots, r_0\} \) we denote by \( P[r_k \ldots r_0] \) the corresponding sub-path \( \{r_k, r_{k-1}, \ldots, r_0\} \). The right sub-path of path \( P \) split at role \( r_k \) is denoted by \( P[r_k \ldots r_0] \).

**Lemma 1** Let \( P^* = \{r_s, \ldots, r_{k-1}, r_k, \ldots, r_d\} \) be an optimal path from role \( r_s \) to role \( r_d \). The right sub-path \( P^*[r_k \ldots r_d] \) of an optimal path \( P^* \) split at any role \( r_k \in P^* \) is shorter than or equal in length to any other secure path \( P_\alpha \) from role \( r_k \) to role \( r_d \) such that \( pv(P_\alpha) \subseteq pv(P^*[r_k \ldots r_d]) \).

**Proof.** Let \( P^* = \{r_s, \ldots, r_{k-1}, r_k, \ldots, r_d\} \) be an optimal path from role \( r_s \) to role \( r_d \). Assume path \( P^* \) is split such that \( P^* = P_1 \circ P_2 \) where \( P_1 = P^*[r_s, \ldots, r_{k-1}] \) and \( P_2 = P^*[r_k, \ldots, r_d] \). Assume that there is another path \( P_\alpha \) from role \( r_k \) to \( r_d \) such that:

1. \( pv(P_\alpha) \subseteq pv(P_r) \). AND
2. \( l(P_\alpha) < l(P_r) \).
Then, we can form a path \( P^* = P_l \circ P_a \). Since \( pv(P_a) \subseteq pv(P_l) \), \( P^* \) is secure. Also, \( l(P^*) < l(P^*) \) as shown below:

\[
l(P^*) = l(P_l) + l(P_a) \\
= l(P_l) + l(P_l) - l(P_r) + l(P_a) \\
= l(P^*) - l(P_r) + l(P_a)
\]

However this contradicts our initial assumption that \( P^* \) is an optimal path. Therefore, no such \( P_a \) can exist. \( \Box \)

**Theorem 1** Assume path \( P^* = \{r_n^X, r_{n-1}^E, r_n^X, \ldots, r_1^E, r_0^X \} \) is an optimal path from \( r_n^X \) to \( r_0^X \). For any pair of domains \( D(r_k), D(r_{k-1}) \), \( 0 < k \leq n \), domain \( D(r_k) \) receives \( P^*[r_{k-1}^E, \ldots, r_0^X] \) if it advertises path \( P^*[r_{k-1}^E, \ldots, r_0^X] \) to domain \( D(r_{k+1}) \).

**Proof.** Let path \( P^* = \{r_n^X, r_{n-1}^E, r_n^X, \ldots, r_0^X \} \) be an optimal path from \( r_n^X \) to \( r_0^X \). Let \( P_{k-1}^* = P^*[r_{k-1}^E, \ldots, r_0^X] \). By the assumption that \( P^* \) is an optimal path from role \( r_n^X \) to role \( r_0^X \), \( P_{k-1}^* \) is a potential optimal path from role \( r_k^E \) to role \( r_{k-1}^X \), and \( P_{k-1}^* \) will be advertised from domain \( D(r_{k-1}) \) to domain \( D(r_k) \). What we need to show is that the path \( \{r_k^E \circ P_{k-1}^* \} \) will be advertised from domain \( D(r_k) \) to domain \( D(r_{k+1}) \). This means we need to show that:\n
**F1.** Let \( P_{k-1}^* = \{r_k^E \circ P_{k-1}^* \} \). If path \( P_{k-1}^* \) \( \in \) \( P^* \) before the execution of \( \text{InputFilter}(\cdot) \), then path \( P_{k-1}^* \) \( \in \) \( \text{FilteredAdv} \) after the execution of \( \text{InputFilter}(\cdot) \).

**F2.** Let \( P_k^* = \{r_k^E \circ P_k^* \} \). If path \( P_k^* \) \( \in \) \( \text{FilteredAdv} \) then after the execution of \( \text{OutputFilter}(\cdot) \) path \( P_k^* \) \( \in \) \( \text{AdvToBeSent} \).

(Proof for (F1)). By statement (2a) of \( \text{InputFilter}(\cdot) \), for all paths \( P \in \text{InAdv} \), a path \( r_k^E \circ P \) will be added to \( \text{FilteredAdv} \) if for all roles \( r \in P, (r_k^E, r) \not\in R(D(r_k)) \). As \( P^* \) is secure and \( P_{k-1}^* \subseteq P^* \), \( P_k^* \) is also secure. Therefore, for all roles \( r_i, r_j \in P_k^*, i \geq j \), \( (r_i, r_j) \not\in \bigcup_{m=0}^{k-1} R(D(r_m)) \). Therefore path \( P_k^* \) \( \in \) \( \text{FilteredAdv} \) after the execution of \( \text{InputFilter}(\cdot) \).

(Proof for (F2)). After statement (2) of \( \text{OutputFilter}(\cdot) \), path \( P_{k-1}^* \) \( \in \) \( \text{Adv} \). Let \( P_k^* = P^*[r_k^E, \ldots, r_0^X] \). As \( P^* \) is secure and \( P_k^* \subseteq P^* \), \( P_k^* \) is also secure. Therefore, for all roles \( r_i, r_j \in P_k^*, i \geq j \), \( (r_i, r_j) \not\in \bigcup_{m=0}^{k} R(D(r_m)) \)

\[
D(r_k) \not\in D(P_k^*)
\]

We also observe that since \( P^* \) is secure, each domain involved in the path appears at exactly one position. Therefore,\n
\[
D(r_k) \not\in D(P_k^*)
\]

By (1) and (2), \( P_{k-1}^* \) is not removed from \( \text{Adv} \) by statement (4) of \( \text{OutputFilter}(\cdot) \). Consider the iteration of statement (5) for \( r = r_0^X \). After the execution of statement (5a), \( P_{k-1}^* \) \( \in \) \( \text{PtoR} \). Now consider the iteration of the nested loop in statement (5b) for \( P = P_{k-1}^* \). By Lemma 1, path \( P_k^* \) \( \in \) \( \text{AdvToBeSent} \) after the execution of statement (5). \( \Box \)

Theorem 1 implies that domains forward all optimal paths, thus proving the correctness of the algorithm. The proof for the correctness in the case of path withdrawals as not included as it follows the same guidelines as the above proof. This is due to the fact that the new optimal paths are already stored in \( \text{PIT-In} \), and they will be advertised after a withdrawal message is received, as indicated in Figure 5.
3.6. Security Analysis

In this section we show that that the generated access paths do not contain any security violations and respect the access control policies of the involved domains.

**Theorem 2** The access paths stored and advertised by RRP are secure access paths.

**Proof.** For a path \(P_n = \{r^X_n, r^E_{n-1}, r^X_{n-1}, \ldots, r^E_0, r^X_0\}\) to be secure it must satisfy conditions C1, C2, and C3 discussed in Section 2.2. Condition C1 is satisfied because RRP advertises a path only through the entry roles that dominate exit roles on which the path was received, that is \(r^E_i \preceq r^X_i\) for all domains \(D_i \in D(P_n)\). Condition C2 is satisfied because RRP advertises paths only through the established cross links between domains. We prove that condition C3 holds by induction. The inductive hypothesis is that path \(P_k\) is stored by \(D_k\) only if it does not violate the restricted access set \(\bigcup_{i=0}^{k-1} R(D_i)\), that is for all \(i < j\) and \(r_i, r_j \in P_k, (r_i, r_j) \notin \bigcup_{i=0}^{k-1} R(D_i)\). (Base Case), \(P_0 = \{r^X_0\}\). \(P_0\) contains a single role and thus does not violate any restricted access sets. (Inductive Step) Assume \(P_k = \{r^X_k, r^E_{k-1}, r^X_{k-1}, \ldots, r^E_0, r^X_0\}\) for some \(k \geq 0\) does not violate the restricted access set \(\bigcup_{i=0}^{k} R(D_i)\) and is stored by domain \(D_k\). Note that path \(P_{k+1}\) is obtained by prepending roles \(r^X_{k+1}\) and \(r^E_k\) to \(P_k\). By Definition 2, all restricted accesses that involve \(r^X_{k+1}\) and \(r^E_k\) are listed in \(R(D_{k+1})\) and \(R(D_k)\) respectively. Domain \(D_k\) checks that path \(r^X_k \circ P_k\) does not violate \(R(D_k)\) before sending an advertisement to domain \(D_{k+1}\), as indicated in OutputFilter() Step 4. Domain \(D_{k+1}\) checks that the path \(r^X_{k+1} \circ r^E_k \circ P_k\) does not violate \(R(D_{k+1})\) before storing the received advertisement, as indicated in InputFilter() Step 2. Therefore, the path \(P_{k+1}\) is stored by domain \(D_{k+1}\) only if it does not violate the restricted access set \(\bigcup_{i=0}^{k} R(D_i)\). \(\Box\)

![Figure 6. Generation of a sample secure path](image)

Path authentication can easily be maintained by using an onion signing technique proposed by Shehab et al. [31, 26] which ensures that paths are not tapered with during their transition between domains.

4. Experimental Results

This section describes the experimental evaluation of RRP based on our proof of concept implementation that simulates the collaboration environment. All the experiments were performed on Intel Pentium IV CPU 3.2GHz with 512MB RAM and running Linux. Java J2SE v5.0 and the Psim-J simulation library [17] were used as the implementation platform. Each domain is simulated as a process with an single message queue. The queuing policy is FIFO. To forward path advertisements domains insert messages in the receiver’s message queue. Each domain has an access role hierarchy which is implemented as a binary tree. Domains select their neighboring domains subject to a probability \(p\), then the crosslinks are generated randomly between each of the neighboring domains. The generated crosslinks are introduced into the simulated collaboration environment using a crosslink scheduler process.

Each domain maintains a PIT-In and PIT-Out database. Several evaluation metrics are collected, which include the number of discovered roles and the size of PIT-In and PIT-Out. The collected metrics were averaged over all the domains in the collaboration network and over several repeated simulation runs. The metrics collected for RRP were compared with metrics collected for the shortest path protocol and the flooding protocol. The shortest path protocol forwards only paths of minimal length without using the potential violators set. The flooding protocol forwards advertisements about every discovered secure path. The presented experiments will investigate the effect of varying several parameters such as the neighborhood probability \(p\) and the maximum path length \(P_{max}\) on the role routing protocols.

4.1. Varying Neighborhood Probability \(p\)

Domains select their neighboring domains randomly with probability \(p\). Increasing the value of \(p\) would connect more domains and would result in the routing protocol discovering more roles in other domains. In this experiment the number of domains was fixed to 100 domains, and the maximum path length \(P_{max}\) was set to 15. The value of \(p\) was varied from 0.1 to 0.9. Figure 7(a) shows the number of discovered roles by RRP, shortest path protocol (SPP), and the flooding protocol. The number of discovered roles is monotonically increasing with respect to \(p\). Note that, the number of roles discovered by RRP and flooding protocol are equal. The flooding protocol forwards all possible secure paths between domains, which implies that the flooding protocol computes a cover of all possible secure paths. Therefore, RRP discovering equal
Figure 7. Neighborhood Probability $p$.

Figure 8. Accumulated Metrics and Simulation Steps ($p=0.3$).

Figure 9. Accumulated Metrics for RRP.

Figure 10. Varying the value of $P_{\text{max}}$. 

8
number of roles as the flooding protocol is an indication to the correctness of RRP as proven analytically in Section 3.5. Also note, that RRP consistently discovers more roles than SPP. This is because SPP forwards path advertisements for only the shortest paths and does not use the potential violator set to make forwarding decisions. Furthermore, this supports our argument that using the shortest path criteria alone fail to compute optimal access paths, as discussed in Section 3.1.

Figures 7(b)&7(c) show the size of PIT-In and PIT-Out respectively. The sizes of the PIT databases of RRP and SPP are orders of magnitude lower than the databases maintained by the flooding protocol. This is because the flooding protocol forwards advertisements about all the possible secure paths. Note, that RRP is able to discover all roles discovered by the flooding protocol without the need to maintain large PIT-In and PIT-Out databases.

In the next experiment p was set to 0.3 and the metrics were accumulated at each simulation step. Figure 8(a) shows that the RRP and the flooding protocol converge to the same number of discovered roles. Figures 8(b)&8(c) show that the PIT databases maintained by RRP and SPP are smaller than the PIT databases maintained by the flooding algorithm throughout the simulation period. Furthermore, the results in Figure 8 show that the RRP algorithm converges as indicated by the flat region in the curves. Figure 9 shows more detailed results for RRP metrics plotted with respect to p and simulation step.

4.2. Varying Maximum Path Length $P_{max}$

The value of $P_{max}$ controls the maximum path length of forwarded paths. In this section the value of $P_{max}$ was varied and metrics were collected for collaboration environments consisting of 100 domains. The neighborhood probability p was set to 0.5. Figure 10, shows metrics generated for different $P_{max}$ values. Note that beyond a certain $P_{max}$ all the collected metrics reach a plateau, for example in Figure 10(a) the number of discovered roles for RRP stays at 27 roles for values of $P_{max}$ ≥ 8. The value of $P_{max}$ could be used to control the behavior of the routing algorithm, for example if $P_{max}$ is set to 5 all three protocols tend to discover the same number of roles, as can be seen in Figure 10(a).

Note that, for all values of $P_{max}$ the number of roles discovered by RRP is equal to that of the flooding algorithm. Furthermore, RRP is able to discover the same number of roles while maintaining smaller PIT databases as indicated in Figures 10(b)&10(c).

5. Related Work

The problem of secure mediator-free interoperability in a multi-domain environment has been addressed in [26, 27]. Where they proposed an on-demand access path discovery protocol which is a flooding protocol. Such an on-demand technique provides no performance guarantees and does not target optimal paths. As we have shown in Section 4 that RRP is able to discover all the roles discovered by the flooding algorithm. Additionally, RRP requires a very low network overhead when compared to the flooding algorithm. The problem of secure interoperation in multi-domain environment has been addressed in [13, 14, 25]. In all such approaches a trusted third party that has a global view of the collaboration environment is required to perform the security policy composition and integration. Dawson et al. [9] presented a mediator based approach to provide secure interoperability for heterogeneous databases. In this approach all access requests go through the central mediator which has a global view of the collaboration environment. Other approaches related to centralized database collaboration have been proposed in [19, 15, 29, 30].

Another area of related research is in path vector network protocols such as the Border Gateway Protocol (BGP) [24, 21]. Although not specified in the BGP standard, most vendor implementations ultimately default to best path selection based on path length [16]. RRP’s goal is to discover optimal paths which are secure and are of minimal length. Furthermore, in BGP there is no notion of restricted access links R.

6. Conclusions

In this paper, we have shown how to achieve decentralized, mediator-free role routing protocol based on the collaboration of neighboring domains. We first presented a role routing mechanism that facilitates the storage and propagation of secure, minimal-length paths to reachable roles in other domains. We showed the correctness and security of the routing mechanism. Finally, we presented experimental results that provided a comparison of the RRP with the flooding and the shortest path protocols.

References


