An Infrastructure for Service Authentication and Authorization Revocation in a Dynamic Aggregation of Networks

DAVID LAI & ZHONGWEI ZHANG
Department of Mathematics and Computing
University of Southern Queensland
Toowoomba, Queensland, 4350
AUSTRALIA
{lai,zhongwei}@usq.edu.au

Abstract: - When a user requests a service from a server (S_A), S_A will authenticate the user based on some stored authentication information. If the information is stored on another server or network which is not accessible to S_A or not in a compatible form of that required by S_A, the identity of the user cannot be established. Without a global authentication service, authentication of users from another autonomous network is a major security issue in service sharing.
In this paper, we extended Network Service Sharing Infrastructure (NSSI) by which many networks are linked together for service sharing. Within NSSI, individual networks authenticate and grant authorizations independent of each other by using their own authentication information repository (AIR). NSSI enables authentication and authorization results to be relayed to other linked networks to access a shared services while individual networks still maintain their own authentication scheme or authentication requirements. NSSI facilitates dynamic aggregation of networks for service sharing with minimum administrative overhead.

Key-Words: - sharing, service sharing, service authentication, service path, authentication propagation, authentication token.

1 Introduction

When a user request a service, the first step is to find out where to send the request. The next step is to submit the proper authentication information. If authentication is successful, the user can start the service session.

It is not easy to find out where a service is provided. But a service list of services provided by a network definitely helps. If a network expands its range of services with services provided by other networks, all local services provided by the network and the shared services should be included in the service list. As servers may stop providing services for some reason and service sharing relationships may change, it is important to keep the service list up to date.

To authenticate a user, a server may use its own authentication information repository (AIR). If corresponding authentication information for the user cannot be found, the server can either reject the service request or try to locate the right AIR and retrieve the appropriate authentication information in a suitable format.

The situation becomes more complicated as autonomous networks are linked together for sharing services. Servers may not be able to access AIRs in another autonomous network. To obtain a service, a user may have to register with many networks, and log on to different networks for individual service.

If the authorization of a user is changed or revoked at some stage during a service session, the server providing the service should know of the change or revocation as soon as possible.

Various methods such as the use of X.509 certificates [11], trust recommendations [5] [6], trust establishment [1] [2] [7] [8] [15] and Kerberos [9] have been proposed as possible so-

lutions to the problems. The major concerns about these solutions are the freshness of certificates, establishing a trusted common third party and static configuration of the networks in Kerberos.

It is desirable to have an infrastructure in which autonomous networks can link together for sharing services with minimum initial set up overheads and using local authentication for both local and shared services. Under this infrastructure, users can inquire about local services and shared services from an agent in the local network and servers will be notified of any change in user authorizations for login sessions.

In this paper, we further develop the Network Service Sharing Infrastructure (NSSI) which enables autonomous networks to use local authentication for shared services. Users can query information about shared services available within this infrastructure.

The rest of the paper is organized as follows. In section 2, the major issues of service authentication and related work are identified. In section 3, we outline NSSI along with the Service Network Graph and the Distributed Network Service Authentication protocol. In section 4, we will discuss how changes in authorization and revocation are propagated to servers which are providing services to the revoked users under the NSSI. We will conclude the paper in section 5 with a discussion on future work.

2 Problems of Service Authentication and Related Work

Service authentication is a process of establishing the identity of a user who requests service on a network. In the simplest case where there is only one server with its own AIR, service authentication is just a log-in session to the server. In such a login session, a user needs to provide a set of authentication information to the server and gets the appropriate authorization. What a user has to collect and maintain is a single set of authentication information.

Without a centralized authentication service, some servers within a network will have their own AIR while others will share an AIR as a group. The format and content of each record stored in an AIR may vary from one AIR to another for the same user.

The number of different authentication records for user in all those AIRs is the number of authentication information sets the user has to maintain. Even in the cases where all servers share the same AIR, or all AIRs have authentication records of the same format and content, the user still has to log in to each server independently to access services from individual servers.

When autonomous networks link together for service sharing, they form a graph of networks. We will refer to such a graph of networks as a Service Network Graph (SNG). Each node in an SNG represents an autonomous network participating in the sharing of services.

When an SNG is formed, each autonomous network will have its own AIR and authentication information is not shared. Network administrators face the problem of authenticating users from other networks which have various authentication schemes and authentication information sets. It is obvious that enforcing a common authentication scheme is not feasible and involves substantial administrative overheads. For instance, when a network using an authentication scheme different from the common authentication scheme links to an SNG, it has to switch to the common authentication scheme. All users of the network have to collect and use a new set of authentication information.

When the network detaches from the SNG, it has to choose between reverting back to the original authentication scheme or stay with the common authentication scheme used by other SNGs. Obviously, if the initial adoption of the original authentication scheme by the network has its own reasons, and those reasons are still valid, the network is going to revert to the original authentication scheme. The administrative overhead and possible confusion and frustration among users are significant problems.

If individual networks do not share their authentication data, users must register themselves with each server or network they wish

to access. Maintaining a global set of authentication data is deemed to fail as some networks may be reluctant to disclose authentication data for security reasons. In addition, some networks may link to the graph or detached from the graph at any time. As a result, setting up a global authentication set is practically infeasible. A typical example is the X.500 [10] plan which has never succeeded in producing a global database of named entities.

Kerberos [9] represents a solution in which users authenticate with a central authentication server and the authentication status can be relayed to the required servers. With one set of authentication information and one log-in, users will be able to access services available from all servers within the same network. Service sharing is achieved by static links between individual realms. Unfortunately, it does not handle dynamic linking of networks efficiently.

Another suggested solution to this problem is the ISO X.509 [11] recommendation which was published in 1993. Authentication in X.509 is based on the secrecy of the private key and the binding of the public key to a user name by a Certificate Authority (CA). The crux of this authentication mechanism is trust in the Certificate Authority. Note that an administrator of an autonomous network may decide to set up a CA for the network or empower a third party to run the CA. However, when many autonomous networks form an SNG, they must agree on a common CA to issue all certificates or on certificate chaining mechanism. The workload increases with the number of networks and number of users involved.

Another approach is to establish a trust [1] [2] [7] [8]. Trust is the result of an assessment of an entity relative to a domain of action [4] by an observer. When an observer is authorized by a network administrator to give trust recommendations [5] [6], the observer becomes a trust agent. The trust is represented by a token and each trust token is signed by the trust agent.

It is reasonable for each autonomous network to have its own set of independent trust agents. A user will be asked to provide trust tokens from a few trust agents. By using the aggregated result [3] of the trust tokens, the server can determine the authentication and authorization status of the user for the requested service.

This works fine for individual networks. However, for an SNG, each autonomous network will have its own set of trust agents. Either all the networks adopt the same common set of trust agents or the user has to collect trust tokens from different sets of trust agents for services outsourced by different networks.

It is desirable to establish an infrastructure for service sharing which allows autonomous networks to link and detach from an SNG with minimum administrative overhead while retaining their own autonomy, independence and integrity. At the same time, forming an SNG should involve no extra input from users. In other words, users should not be involved in the service sharing process.

Our research aims at devising an authentication protocol and developing a service sharing infrastructure for an SNG which allows:

- Each node can have different authentication scheme of its own.
- Each node maintains its own AIR.
- service authentication can be performed locally at each node, but the authentication status will be relayed automatically to other nodes in the SNG.
- A current list of local and shared services is available to all users.
- Revocation of authorization is propagated to servers concerned.

3 Network Service Sharing Infrastructure

We outline the Network Service Sharing Infrastructure (NSSI) in an *ad hoc* SNG using the Distributed Networks Service Authentication Protocol (DNSA) in this section. The concept of an SNG is first reviewed and then followed by the DNSA protocol.

3.1 Service Network Graph

Service Sharing infrastructure is based on an SNG and service paths. An autonomous network is assumed to consist of the following entities:

- Authentication Server (AS) which authenticates local users;
- Server (S) which provides services;
- Service Locating Server (SLS) which stores information about local services and shared services;
- local user (U).

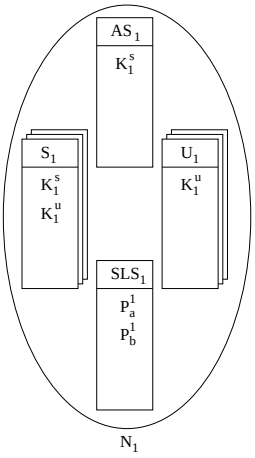

Figure 1: An autonomous network

We also assume that an encrypted channel authenticates statements transmitted via the channel [14]. All communications among autonomous networks and between hosts within the same network are assumed to be encrypted using symmetric encryption and each pair of communicating entities shared an unique symmetric key. For example, Server Key (K^s) is the encryption and decryption key shared between AS and S while Session Key (K^u) is the encryption and decryption key shared between S and U and generated for each nondiscriminatory log-in session.

Note that the network which provides the actual shared service is the *target network* and the network which initiated the service request is the *request network*.

We say that N_1 is *attached* to N_2 when N_2 delegates its authentication authority to N_1. In which case AS_2 generates and shares ATK^2 with AS^1. N_2 is the delegator network and N_1 is the delegatee network and N_2 provides services to N_1 as outsourced services of N_1. This is a one-way relationship and is represented by a single arrow as shown in Figure 2.

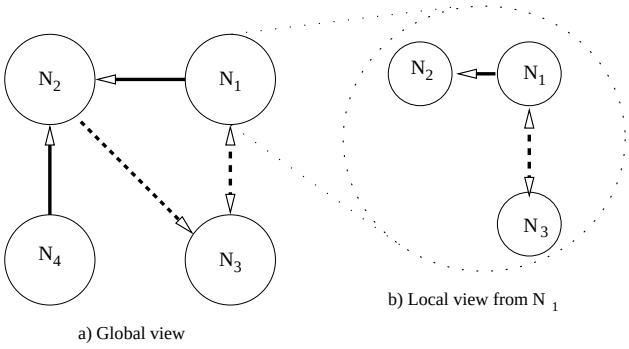

Figure 2: A Service Graph

We also say that N_1 and N_2 are *mutually linked* when N_1 is attached to N_2 and N_2 is also attached to N_1. Mutual linking is therefore a two-way relationship. We will use a double-end arrow to represent a mutual-link.

There are two levels of authentication authority delegation, *restricted* delegation and *free* delegation. If N_1 is granted *restricted* delegation by N_2, N_1 is not allowed to further delegate the authentication authority of N_2 to another network attached to it to form an indirect authentication authority delegation. On the other hand, if N_1 is granted *free* delegation by N_2, N_1 can further delegate the authentication authority of N_2 to another network attached to it. In other words, attaching to network N_1 also implies attaching to network N_2 if N_1 was granted *free* delegation of authentication authority by N_2. We will use solid arrows to indicate *free* delegation and dotted arrows to indicate *restricted* delegation.

A *Service Network Graph* is a set of networks attached to or mutually linked with each other. A global view of an SNG, as shown in Figure 2a, reveals all network attachment relationships.

As the attachment relationships can be one-way or two-way and the delegation can be *restricted* or *free*, not all networks can access

all other networks. A local view of the SNG shows only attachment relationships and networks which are accessible from a particular node.

For example, Figure 2b is a local view of Figure 2a from N_1. N_2 and N_3 are accessible from N_1 as both of them granted delegation to N_1. N_4 does not show up in the local view as N_4 has not delegated authentication authority to N_2. Note that the attachment relationship between N_2 and N_3 is missing in the local view as the delegation is a *restricted* delegation and cannot form an indirect delegation from N_3 to N_1.

With the Network Service Sharing Infrastructure in place, we can now proceed to look at the Distributed Networks Service Authentication Protocol.

3.2 Distributed Networks Service Authentication Protocol

The Distributed Networks Service Authentication Protocol has two distinct operation modes. One is the *Network Participation mode* (NP mode) in which a network links to another network in an SNG. Another is the *User Service mode* (US mode) in which a user access a local or shared service. We will discuss them in the following sections.

3.2.1 Protocol in NP Mode

Let us assume N_1 and N_2 are two separate autonomous networks as shown in Figure 3. Upon receiving the request from N_1 to attach to N_2, AS_2 will decide whether to reject the request, to approve a *restricted* delegation or to approve a *free* delegation.

If the request is approved, AS_2 generates and sends ATK_a^2 to AS_1. SLS_2 also sends information about services available for sharing to SLS_1 in the form of service paths (P). The information is encrypted with ATK_a^2 and is sent to AS_1 which will decrypt and forward the information to SLS_1.

A Service Path is a service locater similar to a URL and is represented by a string of network path and costing metrics:

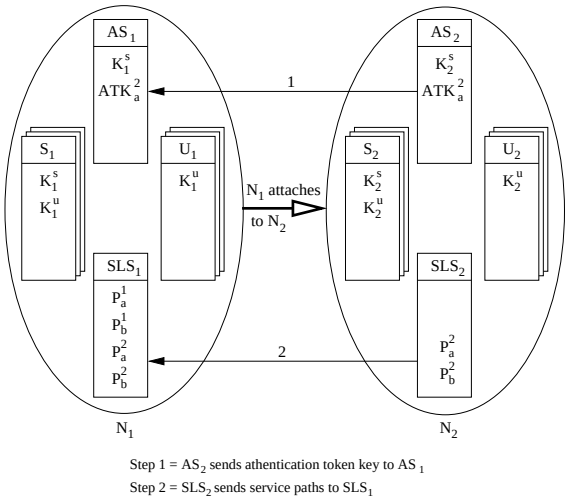

Figure 3: Attaching one network to another network

- Delegation field;
- NetworkPath field;
- TargetNetwork field;
- Server field;
- Service field;
- CostMetrics field.

A letter 'F' in the delegation field indicates the service path can be fowarded in case of authentication delegation while a letter 'R' indicates that the service path is not to be forwarded to other networks.

Services available for sharing include local services in N_2 and those outsourced by N_2. Service paths for services outsourced via a *restricted* delegation are not forwarded to SLS_1. In other words, only service paths with prefix 'F' will be forwarded to SLS_1.

Both AS_1 and SLS_1 will acknowledge the receipt of information from AS_2 and SLS_2 respectively. Note that N_2 can also request to attach to N_1 and form a mutual link with N_1.

From the information of service paths received, SLS_1 can work out its own set of service paths. All newly acquired service paths should have:

- a letter 'F' in the delegation field;.

- the delegation field changed to 'R' if the authentication delegation is *restricted*;.
- the delegator network added to the NetworkPath field;
- CostMetrics field adjusted according to the extra cost required to pass a shared service request to N_2.

Local services will have 'F' as the delegation field and '.' as the NetworkPath field.

Suppose SLS_2 of N_2 has three service paths:

- <F:./$Server_2$/$Service_2A$ >:<8>
- <F:N_4/N_7/$Server_7$/$Service_7A$ >:<22>
- <R:N_5/N_8/$Server_8$/$Service_8A$ >:<13>

The first service path is a local service offered by $Server_2$ in N_2. The second service path is a shared service offered by $Server_7$ in N_7 via N_4. The third service is a shared service offered by $Server_8$ in N_8 via N_5. When N_1 attaches to N_2 and is given a *restricted* delegation, only the first two service paths that has prefix 'F' are forwarded to SLS_1 from SLS_2.

If N_1 offers two local services $Service_1A$ and $Service_1B$ by $Server_1$, the service paths for SLS_1 will be:

- <R:N_2/$Server_2$/$Service_2A$ >:<9>
- <R:N_2/N_4/N_7/$Server_7$/$Service_7A$ >:<23>
- <F:./$Server_1$/$Service_1A$ >:<5>
- <F:./$Server_1$/$Service_1B$ >:<5>

The two service paths from SLS_2 now have a prefix of 'R' because of the *restricted* delegation.

Assuming it costs 1 extra unit to pass a service request to N_2, the CostMetrics of the shared services are increased from 8 to 9 and from 22 to 23 respectively.

Now if N_3 attaches to N_1 and is granted *free* delegation, SLS_1 will forward only the two local service paths to SLS_3:

- <F:./$Server_1$/$Service_1A$ >:<5>
- <F:./$Server_1$/$Service_1B$ >:<5>

The service paths of the shared services will not be forwarded as they are acquired through a *restricted* delegation.

We will explain how to handle a service request in section 3.2.2.

3.2.2 Protocol in US Mode

When a local user U_1 in N_1 requests a service, it will first query SLS_1 whether it is available or not. SLS_1 then returns a message containing a valid service path (P_a^1 or P_a^2 for example) plus its cost metrics or "Service not available" to U_1. If U_1 is comfortable with the cost metrics, the user will authenticate itself to AS_1 and pass along the service path and cost metrics which AS_1 will use to determine the path to reach the target server.

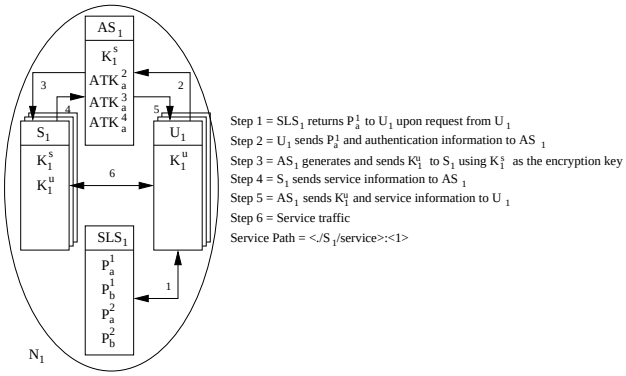

Figure 4: User requesting a local service

If the authentication is successful, AS_1 will generate a session key K_1^u. If the service is available on a local server S_1 as indicated by the Service Path, AS_1 will encrypt K_1^u using encryption key K_1^s of server S_1 and send it along with user authorization information to S_1. S_1 acknowledges the session key K_1^u and returns AS_1 all service information for the request. AS_1 relays the service information and K_1^u to U_1 as shown in Figure 4.

AS_1 will keep a record of the user request and the service path.

If the service is available in N_2 instead of in N_1 as shown in Figure 6, AS_1 retrieves the authentication token key of AS_2, ATK_a^2 and uses it to encrypt the session key K_1^u instead of using a server encryption key K_1^s. The encrypted

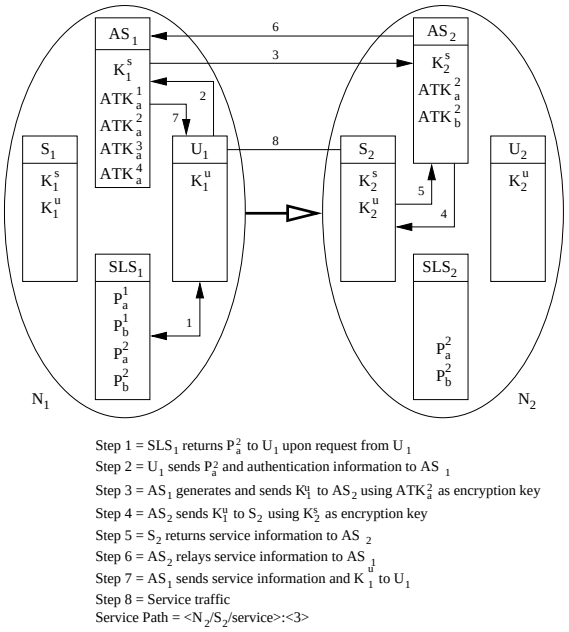

Figure 5: User requesting a shared service

session key K_1^u together with the service path and user authorization information forms an *authentication token*.

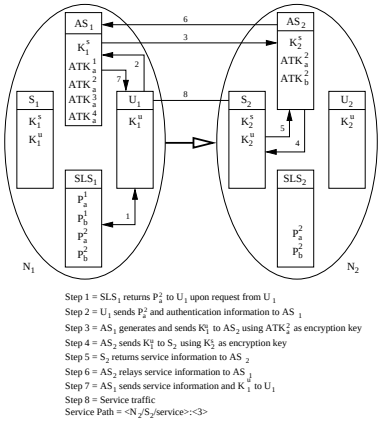

Figure 6: User requesting a shared service

On receiving the authentication token from AS_1, the authentication server AS_2 in N_2 extracts the K_1^u from the authentication token. The service path embedded in the authentication token indicates that the service is offered by S_2. So AS_2 encrypts the authentication token with K_2^s and sends it to S_2 as explained before.

In the case when N_1 is not directly attached to N_2 as shown in Figure **??** the service path would indicate that the target network is only reachable via N_4. In this case K_1^u is passed on from AS_1 to AS_2 via authentication server AS_4 in N_4. AS_1 will encrypt K_1^u with ATK_a^4 while AS_4 will encrypt K_1^u with ATK_a^2. Service information returned from S_2 will follow a similar path but in the reverse order.

To end a service session gracefully, a server will return an *End of Session* message to the AS of the network from which the service request originates. The AS will then delete the service path and user request record from its database.

3.3 Service List

As discussed in Section 3.2.1, when network N_2 delegates its authentication to another network N_1, SLS_2 sends all its service paths to AS_1 encrypted with ATK_2. AS_1 will then decrypt and forward all the service paths to SLS_1. From the service paths received, SLS_1 can work out a local view of the SNG. This local view is used to optimize the service paths received. Together with the service paths for local service, these service paths formed a complete service list available to users in N_1.

3.3.1 Local View of SNG

The information needed to build or rebuild a local view of an SNG can be extracted from the NetworkPath field of service paths acquired by a network when it attaches to another network in the SNG.

For example, if N_1 attaches to N_2 and receives the following service paths:

- <F:./$Server_2$/$Service_2A$ >:<8>
- <F:N_4/$Server_4$/$Service_4A$ >:<12>
- <F:N_4/N_7/$Server_7$/$Service_7A$ >:<20>
- <F:N_5/$Server_5$/$Service_5A$ >:<16>

SLS_1 can derive the conclusions:

- N_1 is attached to N_2;
- N_2 is attached to N_4;
- N_4 is attached to N_7.

- N_2 is attached to N_5;

A local view can be constructed as shown in Figure 7

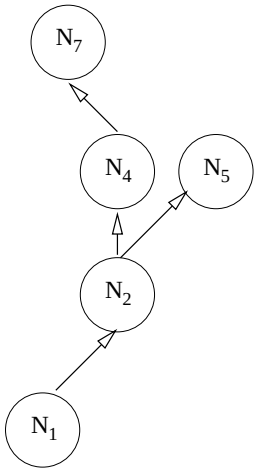

Figure 7: Local view of SNG from N_1

Later on, N_1 attaches to N_3 and acquires three service paths:

- <F:./$Server_3$/$Service_3A$ >:<4>
- <F:N_7/$Server_7$/$Service_7A$ >:<12>
- <F:N_8/$Server_8$/$Service_8A$ >:<6>

N_1 rebuilds the local view using the extra information from the newly acquired service paths as shown in Figure 8 All network related information such as inter-network traffic capacity, congestion control along the path and popularity of the server, are all recorded in the local view.

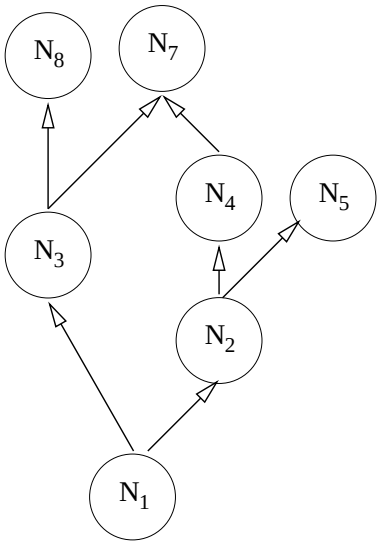

Figure 8: A new local view of SNG from N_1

We shall use the local view to optimize the service paths before dispatching them to users as a service list.

3.3.2 Optimization of Service Path

From the local view shown in Figure 8, we can see that if user issue a service request for $Service_7A$, there are two paths to access $Service_7A$ from N_1:

- <F:N_2/N_4/N_7/$Server_7$/$Service_7A$ >:<21>
- <F:N_3/N_7/$Server_7$/$Service_7A$ >:<13>

assuming it costs 1 extra unit of resources to pass a shared service request to N_2 and N_3. If the CostMetrics is the only determining factor for choosing which service path to use, we may safely leave out the service path with higher cost. We tag those unwanted service paths with a prefix 'D' in the delegation field indicating these service paths are not preferred service paths and there are other alternate service paths for the same service. The unwanted service paths are tagged and not actually removed so that the information in these service paths are not lost. The service path may be untagged at a later time if the CostMetrics for various service paths change or the alternate service paths are no longer valid because networks are detached from the SNG.

Assuming N_1 offers $Service_1A$ with a cost of 7 units, it is simple to work out the Service List of N_1:

- <F:./$Server_1$/$Service_1A$ >:<7>
- <F:N_2/$Server_2$/$Service_2A$ >:<9>
- <F:N_2/N_4/$Server_4$/$Service_4A$ >:<13>
- <DF:N_4/N_7/$Server_7$/$Service_7A$ >:<21>
- <F:N_5/$Server_5$/$Service_5A$ >:<17>
- <F:./$Server_3$/$Service_3A$ >:<5>
- <F:N_7/$Server_7$/$Service_7A$ >:<13>

- <F:$N_8/Server_8/Service_8 A$ >:<23>

If network related information are used when selecting a service path, we may retrieve the related information from the local view of the SNG. Again, we tag those non-preferred service paths with a prefix 'D' without physically deleting them.

3.3.3 Updating of Service Path

A service path becomes invalid if the server providing the service stops providing the service for some reason. To ensure all service paths within a service list are valid, we require that:

- Within a network, the SLS probes all servers within the network periodically to confirm that they are working properly.
- When a server shuts down, it will notify the SLS within the same network.
- When a SLS detects some local service disruption or resumption, it will broadcast the name of the discontinued services to SLSs of attached networks.
- When a SLS receives a service disruption or resumption broadcast, it will update its service paths accordingly; the broadcast will then be relayed to all attached networks.
- For a service disruption, the corresponding service path is tagged with a prefix 'D'.
- For a service resumption, the prefix 'D' is removed from the corresponding service path, making it valid again.

Networks which are part of the NetworkPath field in a service path may detach from an SNG. This will also make service paths invalid. To handle this situation, we require that an AS broadcast to all networks attached before it detaches from the SNG. An AS upon receiving such a message will notify the SLS in the same network and the message is relayed to other AS of attached networks. The SLS will then rebuild the local view of the SNG, delete the service paths that involves the detached network as part of the NetworkPath, and optimize the service paths again.

4 Changes and Revocations in Authorization

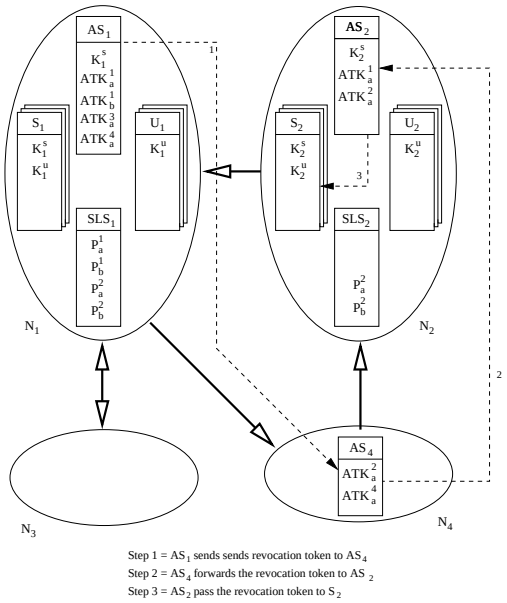

Figure 9: Changes in user authorization is pushed to server

Changes in authorization may change a user's right for services. A total revocation of authorization for a user is equivalent to unregistering the user and deny the user from any service the network may offer. In this section, we will see how changes in user authorization are propagated to servers concerned.

Let us assume that the authorization of user U_r registered with network N_1 is revoked and AS_1 is the only agent that has full access to the authentication and authorization information in the AIR used. Any changes in authorization for U_r must involve AS_1 since it is the only agent which can alter the content of the AIR.

When AS_1 is alerted to any authorization changes for U_r, it will make changes in the AIR. AS_1 will then check the user request records. If U_r is engaged in a service session, AS_1 will push the authorization changes to the server concerned as shown in the service path using a

revocation token. The content of a revocation token is similar to that of an authentication token. Only the user authorization is replaced with the changed authorization or with "authorization revoked" to indicate a total revocation of authorization. The mechanism is exactly the same as the transfer of authentication token for a service session as shown in Figure 9. On receiving such an authorization change message, the server will determine the action according to the change and predefined access control policy.

5 Conclusion

A user interested in a service available in a distributed network environment has to establish a trust relationship with a local network first. Each time a service is requested, the network has to authenticate the user before granting the user access to any local or outsourced service. A service authentication protocol which relays authentication status from a local network to a target network is required for shared services. In this paper, we extended Network Service Sharing Infrastructure based on Service Network Graphs, Service Paths and incorporated the Distributed Network Service Authentication protocol. With NSSI, a user register with an autonomous network within an SNG can log on by using the authentication server of the network and access the shared services of other networks within the SNG. A current service list with optimized network cost is available to all users. Revocation of authorization is pushed from the AS which initiated the revocation process to the server in the service path of the user is engaged in service sharing.

We shall focus on service path, authorization and security issues in the future.

References

[1] T. Beth and M. Borcherding and B. Klein, Valuation of Trust in Open Networks. *Proceedings of the Conference on Computer Security 1994*, 1994.

[2] M. Reiter and S. Stubblebine, Authentication Metric Analysis and Design. *ACM Transactions on Information and System Security*, Vol. 2, No.2, 1999.

[3] A. Abdul-Rahman and S. Halles, A Distributed Trust Model. *Proceedings of New Security Paradigms Workshops*, 1997.

[4] D. Denning, A new paradigm for trusted systems. *Proceedings of 1992-1993 ACM SIGSAC New Security Paradigms Workshop*, 1993.

[5] M. Montaner, B. Lopez and J. L. Rosa, Developing Trust in Recommender Agents. *Proceedings of the first international joint conference on Autonomous agents and multi-agent systems*, 2002

[6] S. Robles, J. Borrell, J. Bigham, L. Tokarchuk and L. Cuthbert, Design of a Trust Model for a Secure Multi-Agent Marketplace. *Proceedings of the fifth international conference on Autonomous agents*, 2001

[7] A. Abdul-Rahman and S. Hailes, Using Recommendations for Managing Trust in Distributed Systems. *Proceedings of IEEE Malaysia International Conference on Communication '97 (MICC'97), Kuala Lumpur, Malaysia*, 1997

[8] A. Abdul-Rahman and S. Hailes, Supporting Trust in Virtual Communities. *Hawaii Int. Conference on System Sciences 33 , Maui, Hawaii*, January 2000.

[9] The Kerberos Network Authentication Service (V5). *Internet Engineering Task Force (IETF) and the Internet Engineering Steering Group (IESG) Porpoised Standard, RFC1510.*

[10] X.500 (02/01). *International Telecommunication Union ITU-T Recommendations X series.* http://www.itu.int/rec/recommendation.asp

[11] X.509 (03/00). *International Telecommunication Union ITU-T Recommendations X series.* http://www.itu.int/rec/recommendation.asp

[12] M. Naor and K. Nissim, Certificate Revocation and Certificate Update. *Proceedings 7th USENIX Security Symposium (San Antonio, Texas)*, Jan 1998.

[13] S. G. Stubblebine, Recent-secure authentication: Enforcing revocation in distributed systems. *IEEE Computer Society Symposium on Security and Privacy, Oakland, California*, May 1995.

[14] B. Lampson, M. Abadi, M. Burrows and E. Wobber, Authentication in Distributed Systems: Theory and Practice. *ACM Transactions on Computer Systems*, vol. 10, no. 4, 1992.

[15] R. Au, M. Looi and P. Ashley, Automated cross-organisational trust establishment on extranets, *Proceedings of the workshop on Information technology for virtual enterprises*, 2001, page 3 - 11.

Acknowledgment: Special thanks to our colleagues Walter Spunde and David Mason for their constructive suggestions.