Soft Enforcement of Access Control Policies in Distributed Environments

Vipul Goyal
Department of Computer Science & Engineering
Institute of Technology
Banaras Hindu University
Varanasi, India
vipulg@cpan.org

Abstract. We briefly consider the issue of access control in highly decentralized structures with one additional concept called entitlement. Entitlement to access a resource means not only that the access is permitted but also that the controller of the resource is obliged to grant the access when it is requested. Since there is no central system designer/administrator, there is no guarantee that policies will be properly implemented by all components of the system. The resource provider may refuse to grant access to a resource even when the seeker is entitled to it. We propose a technique to detect such policy violations by the resource providers and consequently take countermeasures against such practices.

1 Introduction

Recent years have seen an emergence of computing technologies with highly decentralized structures, such as Peer-to-Peer, Grid Computing and Ad-Hoc Networks. Since the various resources are not under the control of a single system designer/administrator, enforcing resource access control policies in such systems is a non-trivial task.

This issue was raised by Firozabadi et al in their recent paper [FS02b]. They argued that the traditional access control models are not sufficient in such environments and concluded that there is a need to devise mechanisms for the soft enforcement of access control policies in these systems. We provide a brief introduction to the problem described in [FS02b].

Existing access control models are originally designed for distributed applications operating on client-server architectures. A basic assumption for these models is that there is a centrally supervised management of the entire system such that access policies will be updated and enforced as they are prescribed. For example, when a new user is introduced then its identity and its access permissions will be added to the access control lists of the provided services. Given this assumption, the policy enforcement component is trusted always to comply with the prescribed policy (unless it develops faults). The question of what to do when a resource provider deliberately fails to comply with the system's policies does not arise.

In contrast, in a system of heterogeneous and independently designed subsystems this assumption no longer holds. In such systems, a number of individuals and/or institutions interact in a collaborative environment to create a *Virtual Community* (VC), shaped and organized according to a set of rules and policies that define how its resources can be shared among its members. A VC is also sometimes called a *Virtual Organisation*, as in [FKT01].

In a VC is there is no centrally controlled enforcement of the community policies. Consequently, there is no guarantee that community policies will be followed as they are prescribed: members of a VC may fail to, or choose not to, comply with the rules of the VC. If there is no way of practical (physical) enforcement of community policies then it would be useful to have a *normative control mechanism* for their *soft enforcement*. By soft enforcement we mean that even if VC members are practically able to avoid complying with the community policies, such behaviour can be detected and they can be subject to sanctioning and remedial action as the consequence of their behaviour.

As an example, consider a VC whose members have agreed to share resources among each other. Suppose a member has joined the community just for a 'free ride', i.e. it happily uses the resources of other community members but denies access to its own resources whenever requested. The resource seeker will of course come to know about the violation of community policies by this member when it is denied access to a resource to which it is entitled, but the question that remains is how that resource seeker can prove this policy violation to the community regulator. Clearly, for systems like collaborative grid computing to be possible, this is an urgent issue which needs to be addressed first. Since, all the subsystems in the VC are independent and autonomous, they cannot be prevented from taking any decision about access control; however a mechanism is needed to enable the resource seeker prove such policy violations by the subsystem to the community regulator. The community regulator would then take countermeasures such as termination/temporary suspension of membership of the violator or impose of fines etc.

2 The proposed solution

2.1 Analysis of the requirements

If the VC just requires the requests/replies to be digitally signed by each member, this may seem to be able to solve the problem. The resource seeker in this case receives a digitally signed reply denying access to the requested resource and hence a proof of policy violation. However, even this policy of digitally signing the replies cannot be enforced by the community regulator. At an extreme, the resource controller may just ignore the request and may not send any reply at all. In this scenario, we assert that no cryptographic technique can solve the problem. The resource seeker will have the capability to generate a request, however it will have to prove that it actually sent the request to the resource controller (and no reply/negative reply was received).

To achieve this, the only possible way seems to be ‘traffic monitoring’. In the next sub-section, we describe our technique. We assume support from routers for traffic monitoring.

2.2 The proposed scheme

Our scheme requires support from routers to cater to a traffic log request for a specified amount of time. The scheme proceeds as follows-

Step 1: The seeker request the resource but is unlawfully denied the access by the server; thus it detects a policy violation by the server.

Step 2: The seeker finds out the IP address of the last router in the path from itself to the server i.e. the router which connects directly to the resource server. This can be easily done by using the TTL (Time To Live) field in the IP packets and getting ICMP Time Exceeded messages, documented in RFC 792 (similar to the working of Traceroute software).

Step 3: The seeker contacts that router and submits a traffic log request. A traffic log request consist of the following components-

1) The destination IP address

2) Time t1 and t2

On receiving this request, the router will log to a file all the traffic between the destination IP address supplied and the sender of the traffic log request starting from time t1 and ending at time t2.

The seeker may supply t1 as the current time (actually little less than current time to take care of possible time difference between itself and the server) and t2 as the expected maximum time for a request/response (e.g. an hour more than t2).

Step 4: The resource seeker sends resource request to the resource server and if it violates the community policy, the seeker may report to the community regulator which could then request the appropriate log file from the router to verify the suspected violation.

While the traffic log request may seem to place some extra burden on the routers, we argue that the routers will have to cater to a log request only when there is a policy violation by a subsystem.

Note that all routers are not required to support the log requests. Support is needed only from the routers directly connected to the resource server. We choose the end router since the routing path of the IP packets between the two systems may change and hence an intermediate router in the path may not be able to log the complete traffic.

2.3 The LAN Scenario

If the server and the seeker are on the same LAN and are connected directly, presumably, there are no routers in the path and hence no traffic monitoring can be done through routers. In this case the above scheme fails.

However a similar scheme can be designed in this case taking the advantage of packet sniffing. For this, there should be a trusted system on the same LAN. The log requests could then be sent to that trusted system and the system would log the traffic using a packet sniffer.

2.4 The Privacy of the Communication

Finally, we ask a question about the privacy of the communication "Does the possibility of the traffic logging by a router threaten the privacy of the communication between the two parties?". The answer seems to be "no". Since logging is requested by one of the two parties involved in the communication only, logging cannot disclose any private information. The requesting party may itself log all the communication if it likes. However, consider a slightly different scenario, when the community regulator is interested in the communication between the two parties. In today's networks, IP address forging is anything but impossible. Hence, if the community regulator forges the IP address of one of the parties and submits the log request for the other party to the appropriate router, it could later obtain the complete log file recording all the communication between the two parties.

In our opinion, this threat and its possibility is small enough to neglect in usual practice rather than introducing expensive cryptographic protocols. However in environments where privacy is a prime concern and all the communication needs to be protected from the community regulator, each subsystem could be assigned a certificate binding its IP address to a public key. The log request to the router could then be digitally signed to avoid any fake requests.

3 Conclusion

Soft enforcement of policies in highly decentralized environments like peer to peer, grid computing and ad-hoc networks is a recent issue. The sub-systems should not be able to make false claims about following the community policy and any policy violation should be detectable. We first identify that any scheme for policy violation detection in such environments would require some kind of traffic monitoring. We then propose a simple and practical solution based on the concept of traffic log requests. Minimal burden on the end router on the path is placed since it is required to cater to the traffic log request only when there is a policy violation by the resource server directly connected to it.

References

[BDF02] Olav Bandmann, Mads Dam, and B. Sadighi Firozabadi. Constrained Delegations. In Proceedings of the IEEE Symposium on Security and Privacy, pages 131–140, 2002.

[FKNT02] I. Foster, C. Kesselman, J. Nick, and S. Tuecke. The physiology of the grid: An open grid services architecture for distributed systems integration. http://www.globus.org/research/papers/ogsa.pdf, January 2002.

[FKT01] Ian Foster, Carl Kesselman, and Steven Tuecke. The Anatomy of the Grid – Enabling Scalable Virtual Organisations. International Journal of Super-computer Applications, 15(3), 2001.

[FS02a] B. Sadighi Firozabadi and Marek Sergot. Revocation Schemes for Delegated Authorities. In Proceedings of the Third International Workshop on Policies for Distributed Systems and Networks, pages 210–213, Monterey, California, USA, June 2002. IEEE.

[FS02b] B. Sadighi Firozabadi and M. Sergot. Contractual Access Control. In proceedings of Security Protocols, 10th International Workshop, Cambridge, UK, 2002

[FSB01] B. Sadighi Firozabadi, M. Sergot, and O. Bandmann. Using Authority Certificates to Create Management Structures. In Proceedings of the 9th International Workshop on Security Protocols, Cambridge, UK, 2001..

[PWFK02] L. Pearlman, V. Welch, I. Foster, and C. Kesselman. A Community Au-thorisation Service for Group Collaboration. In Proceedings of the Third International Workshop on Policies for Distributed Systems and Networks, pages 50–59, Monterey, California, USA, June 2002. IEEE.