G-PASS: Security Infrastructure for Grid Travelers [⋆]

Tianchi Ma, Lin Chen, Cho-Li Wang, and Francis C. M. Lau

Department of Computer Science
The University of Hong Kong, Hong Kong
{tcma, lchen2, clwang, fcmlau}@cs.hku.hk

Abstract. Grid travelers are special mobile processes responsible for coordinating resources that are distributed across multiple virtual organizations (VOs). We propose a security infrastructure called G-PASS to provide security support for grid travelers during their trip and credential mapping when crossing VO boundaries. We demonstrate the power and feasibility of G-PASS with a bio-informatics application running on multiple VOs. We report and analyze the overheads incurred in migration decisions and the actual process migrations. G-PASS can be installed with GSI as the base, thus making it compatible with existing grid middleware.

1 Introduction

In grid computing, a virtual organization (VO) is a group of individuals or institutions who share some computing resources for a common goal. As grid technologies become more mature and widely adopted, more and more VOs are formed and various types of resources and modes of information sharing are supported. It then becomes inevitable that the future development of Grid applications will move towards large-scale deployment, and they need to access services and resources that are scattered among multiple VOs. To facilitate such deployment, several new features should be developed within the grid security infrastructure, which include: (1) support for VO crossing; (2) features to guarantee the security of the coordinating agents and their hosts; and (3) ways to maintain the trust relationship between multiple autonomic VOs.

In this paper, we propose a new type of mobile process, called *grid traveler*, which has the special ability to move across VO boundaries to coordinate the use of resources and access control under different protection domains. An accompanying security infrastructure named G-PASS is proposed for the credential management of grid travelers. G-PASS provides two useful functions: (1) G-PASS implements a new trust model for supporting simple credential verification and transfer, as well as the creation and atomicity of security transactions. The core

[⋆] This research is supported in part by HKU Foundation Seed Grant 28506002, the China 863 National Grid project, and a HKU grant for the HKU Grid Point.

of this trust model is the concept of “security instance”. A security instance includes a security transaction and its constraint specification. One can accomplish the delegation by binding his/her identity onto the security instance instead of binding onto some special host. (2) To support VO crossing, G-PASS provides an RBAC2 [2] qualified role-based privilege mapping with the granularity of security instance. Different from traditional role-based access control (RBAC), a gateway service called G-custom is imported to map the original credentials (recorded in a credential carrier called G-passport) to a locally recognized approval table. The local resource publisher can then work with the normal access control directly without having to install the role-mapping mechanism.

The rest of the paper is organized as follows. Section 2 gives the background of this research. In section 3, an overview of G-PASS and its features are given. The instance-oriented trust model and role-based privilege mapping are discussed in Section 4. Section 5 presents the performance test of G-PASS. Sections 6 and 7 discuss related work and conclude the paper.

2 Background

GSI is the generally used security system in current grids. It maintains basic trust relationships for resource sharing and job submission. However, GSI cannot satisfy the complex security requirements of grid travelers during VO crossings.

Firstly, GSI is too simple to deal with grid travelers. In general, each VO has its own security policy space. There is a set of identity bindings on access rights. A delegation issued from outside of the policy space will be regarded as an invalid request because the identity binding on it cannot be recognized by the local access control policy. Although GSI has implemented a simple role-based mapping mechanism, it is just a simple one-to-one mapping and can hardly deal with complex situations when crossing VOs. For example, for a traveler with multifarious identities, each of the identities provides only a partial approval that will be valid only under some specific constraints. The one-to-one mapping cannot support such complex role mapping relationships.

Secondly, GSI uses the X.509 delegation model and the delegation is bound to the target host. As the grid traveler travels over multiple hops, the delegation will be transferred by continuously issuing new delegation documents. The verifier at grid entry point needs to check all the signatures created in the delegation chain, thus introducing large overheads. The whole security system becomes less scalable.

Thirdly, the atomicity of security transactions cannot be ensured in GSI. For example, a modification to a bank account will include a read and a write operation. There is no safe state between the two operations. If the two operations are approved by two identities, it is difficult to decide which operation should take the responsibility upon the occurrence of an exception during the modification - known as “separation of duty” problem. GSI allows only simple approvals from single identity. It can not deal with the separation of duty problem well.

3 System Architecture

Figure 1 shows the basic working mechanism in G-PASS. G-PASS consists of various security-related components to support grid travelers.

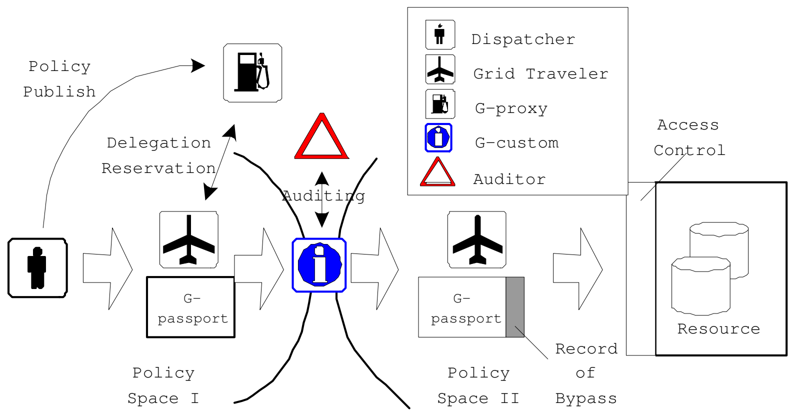

Fig. 1. G-PASS Architecture

The *G-passport* resembles a passport in real life with continuous passport pages. G-passport keeps several types of page content: (1) *G-dispatch* declares the delegation from the traveler's dispatcher. It records a privilege set asserted legal by original dispatcher. (2) *G-warrant* claims the intent of warranting a subset of the privileges declared in G-dispatch by a certain warrantor. At the same time, the warrantor promises to be responsible for the traveler's corresponding behavior. (3) *G-exception* records the security exceptions thrown by the hosts or resource access controllers. It is used for security monitoring. Systems can adjust their policy according to such records. (4) *G-event* records the user-defined security events.

With G-passport we design an instance-oriented trust model [3], which ensures the atomicity of security transactions. A chained signature technology [3] is adopted to ensure the integrity of the G-passport. Thus, the attacker cannot find a way to substitute, modify, or delete a page in the G-passport, nor to insert a page. Each page is defined as a contract, in which at least two identities are required to provide a digital signature, and to claim their responsibilities. This complies with Clark-Wilson's principle of separation of duty [4]. An auditor, if it exists, may be required to reserve a copy of the contract and to accuse any concerned identity who attempts to deny the approval or event.

G-custom is a border checker at the entry of a policy space. The role-based mapping is performed in G-custom. *G-proxy* grants new delegation in the name of the user. It enables single sign-on by the user and keeps his/her policy active even the user is currently offline. This procedure is called *delegation reservation*.

4 Trust Management

4.1 Instance-oriented Trust Model

The traditional trust model sits on top of the Public Key Infrastructure (PKI). Supposed user U holds the keypair $KP = (Pk, Sk)$, where Pk is the public key and Sk is the private key. A digital signature $S_{KP}(v)$ proves the correctness of statement v in U's name. We define a delegation

$$Del(U^{'}, p, U) = ((U^{'} \parallel p), S_{KP}(U^{'} \parallel p), Pk),\ p = \{r_i, c_i)|i = 1, \ldots, n\},\ n \geq 1$$

to claim that $U^{'}$ can have privileges in the set p in the name of U, where r_i represents the detailed privilege and c_i is the constraint of r_i.

From the above, we can see that the traditional delegation is identity-oriented. In the migration of a grid traveler, the target identity is the identity of the target host. So the delegation is also a host-oriented one. It has the following disadvantages for supporting grid travelers. Firstly, it fails to achieve separation of duty. The Clark-Wilson's principle regulated that the states before and after the transaction must be safe and verifiable, so that the responsibility is clear once any exception is raised. It is the issuer of the delegation who can define transactions; however, the privileges are defined by the resource provider. So new delegation mechanism should be developed to enforce the recording of privileges in the form of transactions. Secondly, it incurs a large overhead. Supposed a grid traveler gets a delegation of p_1 from identity U_1; after $k-1$ times of migration, it arrives at the host with identity U_k. A delegation chain is generated during the trip. The host will need to check at least $k - 1$ signatures to assert the validity of this delegation document.

To overcome the above drawbacks, we adopt an instance-oriented approach [3]. A security instance includes a security transaction and its validity specifications. Identities will be simply delegated onto the transaction instead of the privilege operations. This provides the atomicity of security transactions by which the separation of duty can be achieved.

Let $T(r_1, \ldots, r_k)$ be a transaction including a sequence of k operations $(o_1, \ldots, o_k)$, where the operation o_i is performed according to the defined privilege r_i, for $1 \leq i \leq k$. During the delegation granting, the issuer can specify the constraint set C for the transaction. Let $Ins(T, C) = (\{r_1 \ldots r_k\}, C)$ be a security instance of $T(r_1, \ldots, r_k)$ under C. Let $req(r, S)$ represent a request for operating p under the system state S. When it satisfies $r \in \{r_1 \ldots r_k\}$ and $S \in C$, the request is said to be covered by the instance $Ins(T, C)$.

When U wants to grant delegation to $Ins(T, C)$ with keypair $KP = (Pk, Sk)$, it can simply issue a capability, which is a signed document with permitted privileges to serve as the delegation of the instance; that is,

$$Del(Ins(T, C), U) = (Ins(T, C), S_{KP}(Ins(T, C)), Pk)$$

Note the target identity in the traditional host-oriented delegation has been removed from the new delegation document. Thus there needs not be a delegation chain to implement the one-by-one security guarantee, and the overhead in verifying the delegation can be greatly reduced from $k - 1$ to 1.

4.2 Role-based Privilege Mapping

To support VO crossing, the security credentials should be made effectively in the target VO's local policy space. G-PASS imports a role-based privilege mapping, which can proceed in two phrases: (1) role-based privilege mapping, and (2) normal access control. Figure 2 shows the main operations performed in phase (1). The credentials recorded in the G-passport are transformed into a privilege table that can be fully recognized in the target VO's local policy space. The privilege table is formed with an array of instances. Each instance is approved by several locally recognizable roles. As an assistance of the transformation, a role table is imported in which roles and their corresponding global identities are recorded. The role table can be published onto a G-custom service. In phase (2), because the role-based mapping has been done when the grid traveler entered the VO, the local resource provider need not perform RBAC again on the G-passport. It will firstly select an instance according to the given requests. Then it will check if there exists a role that is granted to use all the privileges recorded in the instance by the local access control policy.

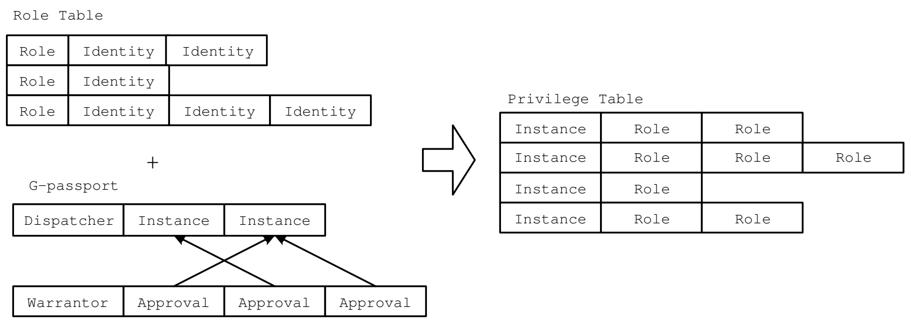

Fig. 2. Role-based privilege mapping

The advantage of this two-phase procedure is that the local resource provider needs not provide a role table themselves. This makes a policy adjustment easier because no synchronization and consistency problem need to be considered.

5 Sample Application and Performance Evaluation

We evaluate the efficiency of G-PASS using a distributed bio-informatics application, BLAST (Basic Local Alignment Search Tool [5]), written in G-JavaMPI [8]. G-JavaMPI is a new MPI middleware which supports parallel and distributed Java computing in a grid environment. A special feature of G-JavaMPI is its support of transparent Java process migration which can facilitate dynamic load sharing and resource-driven task migrations. In G-PASS, the migratable

Table 1. Performance of JavaMPI BLAST Program with/without Migration

Workload	Balanced Workload	Imbalanced Workload, Without Migration		Imbalanced Workload, With Migration		Speedup With Migration Support
	Execution Time (μs)	Execution Time (μs)	Slowdown	Execution Time (μs)	Slowdown	
97%	310490	636577	105%	323070	4%	49.25%
85%	309628	373868	21%	310688	0.34%	16.9%
77%	309773	349056	13%	311226	0.47%	10.84%

JavaMPI process is regarded as a kind of grid traveler which needs security protection when it migrates across VOs.

We set up a simulated grid environment consisting of four grid points representing four universities. At each grid point there are some Linux machines and a shared NCBI protein database of size 4GB. The four grid points are A (machine 1), B (machine 2), C (machine 3), and D (machine 4, machine 5) respectively. All the machines are connected by a campus network with 100Mbps connection. The simulation program consists of four processes, each running on one of the grid points. The four grid points are grouped to form two VOs, where $VO1 = (A, B, C)$ and $VO2 = (C, D)$. The members of each VO share resources with each other based on original GSI support. Each VO provides public services to users from other VOs based on G-PASS. G-customs of the VOs are established in the domain of each site as daemons. They communicate, approve each other and duplicate their policies at runtime.

We artificially create a computation-intensive background task. By tuning their sleep times and calculation precisions, we can generate CPU load (CPU usage) at 97%, 85%, and 77% respectively. During execution, the simulation starts to produce the background workload on site A to increase its CPU workload. According to our migration decisions, the JavaMPI process in site A will be instructed to migrate to an idle machine, either machine 4 or 5, in grid point D. As grid points A and D belong to different VOs, G-PASS is invoked.

The table 1 presents the execution times of the JavaMPI BLAST program under three conditions: (1) execution time under balanced workload, (2) imbalanced workload (with background process running in A) without migration, and (3) imbalanced workload with migration enabled. Comparing to normal execution (case (1)), the slowdown percentages of case (2) (i.e., without migration) are 105%, 21%, 13% under the three workload status , while the slowdowns can be reduced to 4%, 0.34%, 0.47% when migration is enabled (case (3)). Under the imbalanced workload case, the program with migration support can improve the execution time about 49.25%, 16.8% and 10.84%, as compared with the one without migration (case (2)).

Table 2 presents the amounts of G-PASS-related overhead incurred during migration-in and migration-out phases. In the "Migration-Out" phase, the G-PASS operations include generating G-dispatch and granting initiated G-warrants to record the migration event in the contractual history of the process's

Table 2. G-PASS Overhead In Migration-In and Migration-Out Phases

Workload	Migration-Out Time (μs)			Migration-In Time (μs)		
	G-PASS-related	Others	G-PASS Percentage	G-PASS-related	Others	G-PASS Percentage
97%	157941	584065	21%	1282016	134704	90.49%
85%	58725	307901	16%	1304660	134855	90.63%
77%	44561	285671	13%	1308782	135253	90.63%

G-passport. Their overhead ranges from 13% to 21%. The other operations include some pre-migration operations (such as closing files or sockets, etc.), extracting the process state of about 2MBbytes and recording it in a dump file. In the "Migration-In" phase, the G-PASS-related operations include performing G-passport verification, generating new G-evidence for recording the VO crossing and appending it onto the G-passport. They count for more than 90% of the total time. This is mainly because of the large overhead caused by the operation of generating G-evidence.

The other operations performed during this phase include restoring the process state and post-migration operations to re-open files or sockets. The total overhead of process migration is the sum of migration-in cost, migration-out cost and dump-transferring cost. The cost of dump-transferring depends on the network link performance and hard disk performance in local host. In our test, the costs are $3.545s$, $0.889s$, and $0.789s$ under the three background workloads. The percentage of the overhead caused by the G-PASS security protection out of the total migration cost is about 50% of the total overhead. Although G-PASS operations contribute almost 50% of the overhead in our experiment scenario, it can be amortized in those applications with larger problem sizes, which are very common in many grid applications, especially for those data-intensive ones.

6 Related Work

In [6], a Community Authorization Server (CAS) is proposed to issue delegation capabilities. However, CAS is a centralized server that is pre-authorized by the resource provider. In G-PASS, the capabilities can be issued by the user, which do not have to be recognized by the resource provider. Indeed, it is the role-based mapping mechanism that makes this more efficient access control possible. Warrantors are allowed to bid their approvals on part of the capabilities in a distributed manner. Therefore G-PASS is more flexible than CAS and is more suitable for grid travelers.

In [7], an extensible delegation profile is proposed with support for host tracing and privilege shrinking. Host tracing can also be implemented in G-passport by defining a handover event and enforcing the hosts to record it. Privilege shrinking can be implemented as another event in which the host declares that the delegation on some events is invalid from that moment on. However, extensible delegation cannot provide atomicity support on authorization, while such a support is provided by the G-passport.

7 Conclusion

In this paper, a security infrastructure called G-PASS is proposed. The goal is to support VO crossing and information gathering for grid travelers. G-PASS works as an infrastructure, providing protocols and documents (G-passport) as well as primary establishments (G-custom). It is compatible with the GSI in terms of key preparation and GRAM plug-in. Therefore, G-PASS can be used together with existing grid middleware, especially the Globus Toolkit. We envision that with more large-scale applications taking advantage of the grid environment, mobile travelers will be more common and demand more capabilities, and thus mobility support and role-based privilege mapping will be under the limelight in the grid security field.

References

1. I. Foster, C. Kesselman, G. Tsudik, and S. Tuecke. “A Security Architecture for Computational Grids,” Proc. 5th ACM Conference on Computer and Communications Security Conference, pp. 83-92, 1998.
2. R. S. Sandhu, E.J. Coyne, H.L. Feinstein, C.E. Youman, “Role-Based Access Control Models,” IEEE Computer 29(2): 38-47, IEEE Press, 1996.
3. T. Ma, S. Li, “An Instance-Oriented Security Mechanism in Grid-based Mobile Agent System,” IEEE International Conference on Cluster Computing, pp. 492-495, 2003.
4. D.D. Clark. and D.R. Wilson, “Non Discretionary Controls Commercial Applications,” IEEE Symposium on Security and Privacy, pp. 184-194, April 1997.
5. S. F. Altschul, W. Gish, W. Miller, E.W. Myers, and D.J. Lipman. “Basic local alignment search tool,” J. Mol. Biol., 215:403-410, 1990.
6. L. Pearlman, V. Welch, I. Foster, C. Kesselman, S. Tuecke. “A Community Authorization Service for Group Collaboration,” IEEE 3rd International Workshop on Policies for Distributed Systems and Networks, 2002.
7. S. Tuecke, et.al. “Internet X.509 Public Key Infrastructure Proxy Certificate Profile,” IETF, 2003.
8. L. Chen, C.L. Wang, and F.C.M. Lau. “A Grid Middleware for Distributed Java Computing with MPI Binding and Process Migration Supports,” Journal of Computer Science and Technology, Vol. 18, No. 4, July 2003, pp. 505-514.