QuARC: Expressive Security Mechanisms*

John D. Yesberg Mark S. Anderson

Information Technology Division,
Defence Science and Technology Organisation, Salisbury
PO Box 1500
Salisbury, South Australia, 5108.

Abstract

Security mechanisms in current distributed computer systems only allow a small range of security policies to be implemented. We present a system which uses some unusual mechanisms that allow it to implement a much wider variety of policies. This will allow computer security policies to be much better aligned with organisational policies.

The mechanisms discussed are quantitative authentication and vouching, rule-based roles with quantitative privileges, and committees. This paper provides an introduction to these mechanisms and shows how they are used in our system.

1 Introduction

The way that an organisation's computers implement security should reflect the security requirements (i.e. policy) of the organisation. Current computer security technology only allows quite simple policies to be enforced. This paper describes a research prototype system which allows implementation of more flexible network security policies which can better meet the needs of large organisations.

The novel mechanisms used to achieve this are *quantitative authentication*, *rule-based roles*, and *committees*. We call the system QuARC (Quantitative Authentication, Rule-based roles, and Committees).

Secure computer systems, especially those which are part of a network, require users to prove that they are who they say they are. This is called authentication, and is used to counter the threat of masquerade or impersonation. Our system is able to require very rigorous authentication for particularly sensitive operations, and allow simpler authentication for other operations. Multiple techniques (for example, passwords, tokens, and biometrics) can be used to achieve different authentication levels. We also present a novel scheme which allows already-authenticated users to vouch for other users' identities, thereby increasing the (latter) users' authentication levels.

In an organisation, a role is a set of rights and responsibilities. For example, in a bank, there may be teller, manager, and enquiries officer roles. Tellers may be permitted to perform deposits and withdrawals, enquiries officers may be permitted to open new accounts, and managers may be permitted to adjust interest rates. A role-based security system labels a group of privileges with a role name. Users are assigned (typically by the system administrator) to the appropriate role, and thereby have access to the appropriate privileges and operations.

Our system has a very flexible role mechanism, which allows users to enter a role if they can satisfy the *role entry rule*. The rule can specify conditions relating to time, place, prinicpal name, authentication level, privileges, and certificates (discussed in the next section). This allows the system to cater for a wide variety of circumstances.

The QuARC system provides a mechanism for decision-making by committees, reflecting a very important part of organisational policy. A committee may be given much more privilege than any one of its members, allowing it to authorise or perform particularly critical or sensitive operations. Some committees may be empowered to modify policy elements of the system, in real time.

In the next section, we outline the overall framework of the system, and introduce and briefly describe the novel mechanisms. We discuss some aspects of the QuARC system, and contrast it with three other distributed security systems in section 3. A status report and conclusions end the paper.

*The work reported in this paper has been funded in part by the Cooperative Research Centres Program through the Department of the Prime Minister and Cabinet of the Commonwealth Government of Australia.

0-8186-7318-4/95 $4.00

2 The System and Mechanisms

The QuARC system is described using the client server model, which is conventional for distibuted systems. Each user (client) has a *security agent* at his local workstation. The security agent may be a process running on the workstation, or a separate piece of hardware if this is required for assurance reasons. The security agent represents the user in security matters. The security agents communicate with various *security servers*, including an *authentication server*, a *role server*, a *vote server*, and a *certificate server*. The security agent may also communicate with other *resource servers*, such as a file server, a printer server, etc.

The system's overall security is entrusted to the security servers and the security agents. A flaw in these devices could potentially result in a catastrophic security breach. Each resource server, however, is responsible only for the resource that it protects. A failure in a printer server, for example, would not affect the security of resources managed by the file server.

The computer network itself is, for security analysis purposes, assumed to be hostile. That is, we assume that malicious entities may present various threats involving passive (e.g. eavesdropping) and active (e.g. masquerade) attacks.

Two types of information are used to make security decisions. These are policy information and certificate information. Policies are usually more general, and are intended to govern the general behaviour of servers, whereas certificates store information about specific instances in the system. For example, a statement that users need to authenticate themselves using a password would be called policy, whereas the statement that "John Yesberg" is assigned to the "Security Researcher" role would be encoded in a certificate.

The term *certificate* is widely used in public key cryptography world to denote a mechanism for distributing public keys. In the QuARC system, we have a more general definition of a certificate: a statement made and signed by a principal. Certificates may refer to other principals' public keys, although this is only one of their many applications in our system.

Although the QuARC prototype system uses public key cryptography for both confidentiality and integrity of certificates and messages, the system is designed to be adapted to use whatever encryption mechanism is suitable for the environment. There is no further discussion of encryption in this paper.

2.1 Quantitative Authentication

Amongst the myriad of functions that staff perform for an organisation, some are more sensitive than others, and must be protected more carefully. Other operations need less protection. One of the threats from which protection is required is masquerade, in which one person impersonates another. Some operations, particularly in a military environment, will require a very high degree of authentication, possibly involving such techniques as retina scans, voice analysis, and use of a smart card. Other less sensitive operations, such as reading news, would require less demanding authentication procedures, e.g. a password.

In the QuARC system, a server will demand that a user attains a certain *level of authentication* before it will perform the required service. Users can supply as much authentication evidence as they need to obtain the required level.

This concept of quantitative authentication has applications in the real world. A commercial example is the Australian banking system. To open an account, one is required by law to provide "100 points" of identification (authentication). Valid authentication "tokens" include:

Token	Points
Passport or Birth Certificate	70
Driver's Licence	40
Credit card or ATM card	25
Existing account holder (1–3 yrs)	40
Existing account holder (> 3 yrs)	100
Telephone contact with employer	35
Telephone directory entry and contact	25
Electoral roll entry	25

Figure 1. Tokens accepted by banks.

By supplying enough tokens, a bank customer reaches the appropriate level of authentication required to open an account.

When a user operates an authentication mechanism, such as a biometric device or a password checker, if the result is successful, a certificate (signed by the device) is created, and stored by the *certificate server*. The user (actually the security agent on his behalf) is able to retrieve all such certificates, and send them with a request to the *authentication server*. The authentication server checks certificates in accordance with its policy, for example, that they were generated by a suitable device (i.e. one in the same geographical area as the security agent), and calculates an authentication level. The level is returned to the security agent.

The authentication server is able to calculate a level according to a number of different authentication policies. Policies may differ according to the relative strengths of particular authentication techniques,

geographic location, or time. The security agent maintains authentication levels for each policy, and supplies servers with the level according to whichever policy the servers require.

We have mentioned that the authentication server examines certificates created by various *electronic* sources, such as a password checker, a hand geometry scanner, a smart card authentication device, and so on. A novel aspect of our system is that *other users* may also make statements concerning a principal's authenticity. We call this operation *vouching*. The value of a vouch is determined by the authentication policy, and may depend on the vouching user's authentication levels, privileges, and other contextual information, such as the time of day.

Although a full discussion of vouching issues is outside the scope of this paper, we mention some of them briefly.

- Who may vouch for whom? How is this expressed?
- What are the relative strengths of different principals' vouches? What sort of arithmetic should be used when more than one principal vouches for a user?
- How can the vouch certificate be related to the vouchee? What stops another user from using it for impersonation?
- Could a vouch ever decrease an authentication level?

In some environments, an unattended (but "logged-in") workstation could be a vulnerability. A possible countermeasure involves authentication levels that change over time. A QuARC authentication policy may specify that a given authentication level decays exponentially, linearly, or simply runs out at a given time. Consider, in the military environment, that a user needs to perform a very critical operation, such as ordering an attack. A rigorous authentication would be demanded for this user. If the authentication decays exponentially, the high level will decrease quickly (at first), and further critical operations would need re-authentication. In command and control environments where users may move about, such a feature is desirable.

We do not have a formal definition for an authentication level, although it is related to the probability of impersonation. For this reason, we are unable to justify analytically the levels assigned for different styles of authentication. These choices are made empirically, according to how various styles are believed to counter the perceived threats. For the same reason, we are unable give a formal argument for exponential decay (or otherwise) of authentication level. We leave the choice of decay type and rate as a subjective security decision for the policy maker.

2.2 Rule-based Roles

Many authors have described role-based security systems [1,2,3]. In essence, a role is a named collection of privileges. The primary advantage of role-based systems is that the indirection provided by role names eases system administration.

In the QuARC system, the *role server* manages the database of roles. When a user needs to *enter* a role, the security agent sends a request to the role server. The role server decides whether that user is allowed to enter the role, and if so, it returns a list of privileges to the security agent.

The novelty of our mechanism is that users are not explicitly assigned to roles. Each role has an entry rule, which describes what conditions need to be satisfied by a user in order to enter the role and acquire its privileges. A typical rule for entering a role may require a certain authentication level (according to some named authentication policy), and a certificate signed by a particular authority, stating that a user may enter the role. For example, the "Security Researcher" role requires a user to have an authentication level of at least 80 points, and a certificate signed by the Chief of the Division allocating him to that role.

The reason for this approach is flexibility. Creators of roles can specify different requirements for different roles. Rules can specify conditions relating to time, place, principal, authentication level, privileges held, and, most generally, certificates. The appropriate authorities can assign users to different roles by creating certificates. For example, whereas the Chief of the Division assigns research roles, a Security Officer would be responsible for creating a certificate indicating a clearance level, which is needed to enter roles whose privileges give access to classified information.

Existing systems often have an administrator user (or role or account). This administrator typically has extraordinary power on the system, and may perform operations which are not permitted by the security policy. One of our aims is for the QuARC system to mirror the organisational policy. Thus a person who is responsible for some aspect of the organisation should be responsible for performing the relevant operations on the computer. A good correspondence between organisational policy and computer security policy increases the organisations confidence that the computer's policy is suitable.

The system uses quantitative privileges [4,5]. That is, each privilege has a quantity associated with it.

This quantity may refer to the strength of the privilege, the number of times it can be used, or any other use defined by the server concerned. Some privileges will be consumed by servers. This allows economic mechanisms to be used to control access to resources. Other privileges may remain at a constant level for a given role.

One of the interesting possibilities with this scheme is that if a principal is *only just* able to enter a role, he may not acquire as much of each privilege as another principal who easily satisfies the role entry rules.

Privileges are generally used to give principals access to certain operations by servers. For example, a sales clerk role may have a privilege which gives access to the printer server operation to print on the "invoice" forms. Although most of the privileges for day-to-day roles would be used by resource servers, the security servers make use of privileges too. Special privileges are required to add, modify, and delete roles, for example. Also, principals may need a privilege in order to use a certain authentication device, and a voting privilege in order to cast a vote for a committee.

Just as QuARC authentication levels can have an expiry time, so can roles. A role server may apply an expiry time to a role, so that if the user wants to remain in the role after that time, the role entry rules will be re-evaluated, to make sure that the user is still permitted to be in the role. If the user is only admitted to a role because of certificated information which has an expiry time, it is sensible to set the role expiry time to the certificate expiry time.

2.3 Committees

Committees play an important part in the running of large organisations, and it is advantageous for a computer security system to reflect this. In some sense, a committee moderates the actions of an individual. We could model a committee as a filter which only allows *reasonable* actions through. The filtering can be used as a basis for giving a committee higher authority than the individual. For example, the House of Parliament has higher authority than the Cabinet of Ministers, which itself has higher authority than a single minister.

The term separation of duty has been used in the literature [6,7,8] to describe committee-like moderation. (An analysis of the correspondence between typical separation-of-duty mechanisms and a committee is left for another publication.) One benefit of implementing a committee system is that there is an implicit mapping from the organisational policy to the computer policy.

In the QuARC system, a committee is a principal, like human users. A committee is able to enter roles and request various actions by different servers. However, a committee is not required to authenticate itself using typical human authentication techniques, although cryptographic secret sharing schemes [9] could be used for this purpose. Whereas human users are represented by a security agent at their workstation, a committee is represented by the *vote server*. When members (either human or sub-committees) of a committee vote on a motion, their votes are communicated to the vote server. The vote server checks and counts votes in accordance with the committee's policy and the result is evaluated. The vote server then acts for the committee, entering the relevant roles, and carrying out the operations specified in the motion. A typical motion might be that a committee issue (i.e. sign) a certain certificate. The members of the committee would vote, and if the motion is passed, the certificate would be signed. For example, an interview panel committee may sign a certificate which is an offer of a job to the interviewee.

The vote server performs actions for each committee according to the committee's policy. Such policies specify which privileges allow members to move motions, vote, have casting votes, and so on. Policies also specify parameters such as the quorum of the committee, whether proxy voting is allowed, whether ballots are secret, whether the committee members may vote while they are temporally or geographically remote from the meeting (if there is one), and what conditions apply to changing the policy. Each motion that is submitted may have policy parameters included in it to override the standard committee policy parameters.

We have mentioned that committees can potentially be very powerful principals. Some of the most sensitive operations in the QuARC system would be the modification of the system policies themselves. The committee policies, role rules, privileges assigned to roles, and authentication policies can all be modified by principals with the appropriate privileges. This means that major changes to the system can be made from *within* the framework of the security policy, rather than by a system administrator, who would need to have total jurisdiction, outside any policy. A committee-based system models large organisations much more closely than one with an all-powerful system administrator.

Although we have suggested that privileges allowing modification of policy should be given to committees rather than individuals, this is not a technical

requirement. Any principal, including an individual user, may obtain any privileges if the appropriate role rules can be satisfied. For example, a user with a certain set of privileges may be permitted to create a subrole, which represents a subset of those privileges. This is an effective way to delegate privileges to another user.

3 Discussion and Contrasts

3.1 Confidentiality

One aspect of computer security which we have not yet mentioned is the confinement of information, in which users or processes are prevented from communicating sensitive (e.g. classified) information to inappropriate destinations. Neither QuARC nor any of the three other security systems (contrasted later) claim to solve this problem.

It is possible, however, for the QuARC to give some limited support for multilevel operations. The security agent is a trusted process. If it is a software process on the workstation, then the entire operating system must be trusted, and, to some degree, the problem disappears. If the security agent is an attached piece of hardware, then this piece of hardware must be trusted. If the operations to be performed on the sensitive information are simple enough to be carried out by the security agent, without divulging its contents to the workstation, then confinement is possible.

Technology exists [10] to allow classified system-high networks to be connected to lower level networks. This means that the confinement protection is applied to the network rather than an individual node. Such a system would be appropriate in a QuARC environment.

3.2 Single Sign-On

Many organisations today are encountering the problem of users requiring multiple login names and passwords to use different systems. The term *single sign-on* refers to solutions which allow a user to authenticate themselves only once, and thereafter have access to all available resources, regardless of whether the resource is on a local or remote machine.

While the QuARC system's authentication level can be used by any servers, it does not conform strictly to the standard definition of single sign-on. A user who has authenticated at a low level (e.g. with only a password) will be able to access any services that do not require a more stringent authentication. In order to access other more sensitive services, further authentication evidence needs to be provided, perhaps in the form of a fingerprint scan, or by the use of a cryptographic token.

In some multi-domain situations, a server in one domain may not trust some of the authentication methods used by another domain, and may demand other varieties. Requiring every server in the world to trust every authentication device is not realistic. Thus, the single sign-on concept is only useful in a limited scope, such as within a single administrative domain.

3.3 Delegation and Revocation

The QuARC role rule flexibility allows for a relatively simple delegation mechanism. One of the possibilities for entering a role could be holding a delegation certificate signed by another principal who can enter the role. This is similar to the delegation mechanism of DSSA [11]. (DSSA is discussed later).

In systems such as QuARC and DSSA which rely on certificates for the transfer of security information, revocation is an important issue. There is currently no solution for guaranteed immediate revocation in a system which assumes a hostile network. The general approach is to compromise performance and security by adjusting certificate validity times, as discussed by 11.

3.4 Policy Modification

We have noted earlier that some principals may be given the rights to make changes to the system policy. The system policy is encoded in server parameters, such as role entry rules and privileges, committee policies, and authentication policies. Allowing a user to create, modify, and destroy roles, for example, is effectively allowing modification of the system policy.

Systems like QuARC derive their flexibility from their complexity — the large number of configurable parameters. However, such complexity can have undesireable side-effects, such as subtle loop-holes in the security. Just as a system's security needs to be analysed and acredited before its initial use, policy modifications need to be made carefully, and possibly analysed for potential unintended side-effects.

3.5 Role Exit Rules

Each role in the QuARC system has an entry rule. Roles may also have role *exit* rules, which have to be satisfied before a principal may leave a role. Such rules can be used to encode part of the *responsibilities* associated with a role in the organisation. For example, a duty officer may not be permitted to leave that role until a replacement officer is available to take over. While this may not be considered a security feature, it does allow the computer system to implement further aspects of organisational policy.

There is potential for conflict of policies in the system if a role expires, yet the role exit conditions are not satisfied. A "metapolicy" is required to adjudicate

in this situation. Our work in this area is only at a very early stage.

3.6 Contrasts with Other Systems

The three most well-known security systems for computer networks are OSF's DCE (Distributed Computing Environment), ECMA's SESAME (Secure European System for Applications in a Multivendor Environment), and DEC's DSSA (Distributed System Security Architecture). Of these, DCE has the largest installed base.

DCE [12] is not aimed specifically at security, but it does include several security functions. DCE uses the well-known Kerberos authentication protocol [13], with secret keys. Servers and clients can authenticate each other, and provide confidentiality and integrity protection for their communications.

DCE's file servers and registries use an access control list (ACL) mechanism, which may be tailored for use by other object servers. Access can be specified in terms of groups, which provide some of the same administration advantages as roles. By judicious settings of groups' access control lists, users can be added to groups by people other than the cell administrator. However, the flexibility inherent in our role rule language for specifying, say, temporal or geographic conditions is not available in DCE. Although an access control list can support a variety of access types, this does not really allow the flexibility associated with quantitative, consumable privileges.

DCE does not provide any mechanisms for supporting committee constructs. The concept of quantitative authentication is not meaningful within DCE, although an object's ACL may allow different accesses for authenticated and unauthenticated users.

SESAME [14] is the closest of the three systems to QuARC. It has equivalent components to our security agent ("Subject Sponsor"), authentication server, and role server ("Privilege Attribute Server"). All user privileges are stored and managed by the privilege attribute server. Although a variety of privilege attribute types are supported, (e.g. identity, group membership, role, clearance, and capability) we are unable to determine the ability of non-administrator users to affect a user's privileges in SESAME. Although we have no direct evidence that SESAME could support quantitative privileges, the mechanisms do appear to be flexible enough to support this. Consumption of privileges could, however, be beyond its capabilities.

SESAME is reported to have an authentication protocol which is structured to support "future authentication methods". This does not indicate to us that it will support quantitative authentication. There is no support for committees in SESAME.

There are two types of delegation in SESAME. In the first type, a user delegates some authority to a server, which allows the server to use further services on behalf of the original user. The second type of delegation is quite unusual. SESAME allows a user to have several identities: different ones for different purposes, such as logon, audit, charging, access to objects, non-repudiation, and acquiring privileges. We quote an example from [14]:

> One simple example is where Tom is to be able to use Jerry's identity to access the system for a period (say Jerry is on holiday). He is permitted to do this with respect to accesses to all objects accessable by Jerry's "Access" identity. ... Tom would not be able to use the identity Jerry for [acquiring privileges].

This use of multiple identities does not appear to be as elegant as the QuARC and DSSA style, in which a delegator signs a certificate giving a delegate various accesses.

DSSA [11] uses a directory of certificates to store security attributes, like QuARC. However, the main security attributes stored are public keys and access control lists. The DSSA is able to use public key or "secret key with trusted server" style authentication techniques. Smartcards play a significant part in DSSA, and perform some of the security agent functions. There is, however, no indication of support for quantitative authentication levels. The DSSA does not use committee mechanisms.

The DSSA uses roles in a very simplified way. When a user logs in, he has access to all privileges assigned to him. To operate in accordance with the least privilege principal, a user needs to generate a subrole, which has only a subset of the available privileges, and enter that subrole. This is the only concept of role used by the DSSA, and it does not really offer the main administrative advantages of privilege management which are normally associated with role-based systems.

Although the DSSA and QuARC delegation techniques are very similar, it is much more important in the DSSA:

> When a user authenticate himself to a workstation, the user at the same time *delegates* to the workstation the right to speak on behalf of the user. This delegation is expressed in a certificate signed by the user's smartcard at login.

4 Project Status

The QuARC prototype system is implemented on Sun Sparcstations. The client-server communications are implemented as DCE remote procedure calls (although DCE security services are not used). Authentication server, role server, certificate server, and security agent software is operational, and a preliminary vote server has been constructed, although only a single parameter (quorum) of committee policy is used at this stage. We have a graphical user interface for the security agent. A hand geometry scanner and password checking program are integrated into the system as authentication devices, and work is proceeding on integrating an RF passive identification system and a smart card interface.

Other work planned includes:

- development of a mathematical model for analysing the system and its policies;
- construction of an arithmetic for vouching; and
- further support for role responsibilities (along the lines of role exit rules).

5 Conclusions

We have described a system which uses some new security mechanisms to allow implementation of flexible security policies.

Quantitative authentication allows the system to demand different levels of authentication according to the sorts of operations that are to be performed. A variety of common authentication techniques can be used to provide authentication evidence. Other people may also vouch for a user's authenticity.

Roles are used to group privileges for ease of administration. Entry rules are specified for each role, to allow a role server to decide whether a user can enter the role and receive the relevant privileges. This allows many management operations to take place within the framework of the security policy, rather than being performed by a superuser with ultimate power.

Committees allow groups of people to make joint decisions. By providing a committee structure for computer security, a clear mapping can be made between organisational policy and computer policy. The moderating tendencies of a committee can justify allowing the committee to enter very powerful roles. Such roles may be powerful enough to change aspects of the system policy itself.

If a real-world organisation uses a computer security policy which is too simple, then there will be frequent occasions on which the security has to be circumvented by a system administrator. As computer security systems are able to implement more realistic (i.e. complex) policies, such occasions will arise less frequently.

6 References

[1] T. C. Ting, "A user-role based data security approach," Database Security: Status and Prospects, 1988.

[2] R. S. Sandhu, E. J. Coyne, H. F. Feinstein and C. E. Youman, "Role-based access control: a multi-dimensional view," Proc. Computer Security Applications Conf., 1994.

[3] D. Ferraiolo and R. Kuhn, "Role-based access controls," National Computer Security Conference, 1992.

[4] S. J. Mullender and A. S. Tanenbaum, "Protection and Resource Control in Distributed Operating Systems," Vrije Universiteit Amsterdam, IR-79, March, 1983.

[5] K. C. Sevcik and D. C. Tsichritzis, "Authorisation and access control within overall system design," Proc. Intl. Workshop on Protection in Operating Systems, 1974.

[6] D. D. Clark and D. R. Wilson, "A comparison of commercial and military computer security policies," Proc IEEE Symp. Security and Privacy, 1987.

[7] R. S. Sandhu, "Transaction control expressions for separation of duties," 4th Aerospace Computer Security Applications Conference, 1988.

[8] P. Terry and S. Wiseman, "A 'New' Security Policy Model," Proc IEEE Symp. Security and Privacy, 1989.

[9] E. Dawson and D. Donovan, "The breadth of Shamir's secret-sharing scheme," *Computers and Security*, 13, no. 1, pp. 69–78, February, 1994.

[10] M. Anderson, K. Hayman, D. Marriott, J. Yesberg, L. Nayda and B. Beahan, "P1 prototype stubs: an overview," Defence Science and Technology Organisation, Australia, ERL-0668-RR, 1992.

[11] M. Gasser, A. Goldstein, C. Kaufman and B. Lampson, "The Digital Distributed System Security Architecture," National Computer Security Conference, 1989.

[12] Open Software Foundation, *Introduction to OSF DCE*. Englewood Cliffs, NJ, Prentice-Hall, 1992.

[13] J. Kohl and B. C. Neuman, Internet-Draft: The Kerberos Network Authentication Service (V5), September, 1992.

[14] P. Kaijser, T. Parker and D. Pinkas, "SESAME: The solution to security for open distributed systems," *Computer Communications*, 17, no. 7, pp. 501–518, July, 1994.