Open Software Foundation
Request For Comments: 8.1
October 1995

B. Blakley (IBM)

SECURITY REQUIREMENTS FOR DCE

1. INTRODUCTION

The DCE/DME Security Working Group, a joint subgroup of the DCE/MAN SIG, and the Security SIG, proposed and voted on a set of requirements for DCE security extensions. This RFC describes the items, the resultant voting process, and provides some analysis.

Initially a laundry list of requirements was proposed by individuals. All suggestions were accepted which met three criteria:

(a) The requirement must be understandable.

(b) The requirement must not duplicate another requirement on the list.

(c) The requirement must be within the scope of the OSF DCE effort.

The precise explanation of each requirement was worked out in committee. Finally, a vote was taken among the working group members, one member organization/one vote, for the priority of each requirement for the first post-DCE 1.1 major release (1.2 or 2.0) and the first post-DCE 1.1 minor release (1.1.1, etc.).

In the sections that form the main body of this document, the ''official'' statement of each requirement is given, as agreed to by the working group. The subsequent explanatory material is the author's ''unofficial'' elaboration, as an aid to the reader.

The identification code scheme (indicated by curly brackets), and the classification of items into groups, were done after the fact and are artificial; they are retained for internal reference purposes, but each item should be understood to stand alone.

> *NOTE: This edition of the RFC has been updated to include votes received at the August, 1994, meeting of the DCE/DME Security working group in Zurich, Switzerland, and via email thereafter. The votes can be found in section 16; a voting summary (annotated to indicate the top ten requirements) can be found in section 17.*
>
> *August, 1994, Note on Authentication Requirements: Requirements {E3}, {N1}, {E8}, and {PK1} all deal with authentication. They can be summarized as follows:*
>
> *(i) Requirement {E3} calls for ''non-password-based'' authentication technology support. This means mostly extensions to the registry and to the* `sec_login` *API to support the additional information and interaction required to allow integration of advanced authentication technologies.*
>
> *(ii) Requirement {N1} calls for support for traditional authentication clients other than Kerberos in a DCE client system. This includes support for integrated management of DCE and non-DCE credentials; for example, this could include extending the DCE login context to provide pointers to Lotus Notes ID files, Netware credentials, etc., in addition to DCE tickets.*
>
> *(iii) Requirement {PK1} calls for a particular additional authentication client, namely one based on public-key technology.*
>
> *(iv) Requirement {E8} calls for extensions to GSSAPI to allow multiple authentication mechanisms to be made available simultaneously under GSSAPI, and to allow new mechanisms to be installed and activated dynamically. Today, adding another authentication mechanism under GSSAPI in a DCE environment requires re-compiling the DCE client code.*

2. AUTHORIZATION

2.1. {A2} Authorization Based on Style of Authentication

REQT: Permit a server to base authorization on the method by which the user logged in without changing the authorization model.

''Authentication style'' refers to the use of various technologies such as passwords, smart cards, biometrics, etc. A vote for this requirement explicitly prohibits a scheme where the ACL writer can specify an authentication style on the ACL, as it was felt that the ACL writer would not generally have enough information to make a useful decision and any such scheme would be too complex to use. One proposed method for implementing this without impacting any ACL model is to assign the user to different groups depending on the authentication method used.

Impact if requirement not satisfied: Without the proposed support, principals who use strong authenticatione mechanisms can't be granted additional authority via the authentication mechanisms of DCE. This significantly limits the value of strong authentication.

2.2. {A3} Authorization Based on Capabilities

REQT: Permit a server to control access based on the client's presentation, and verifiable legitimate possession, of an unforgeable capability designating permitted rights to the information.

The capability model referred to here is one in which the client obtains, prior to accessing a server, a kind of ''ticket'' for some specific information (e.g., a specific file or set of them). This ticket could be in an extended PAC or separately transmitted. A server receiving and verifying such a ticket should not need to make further access control checks based on the client's identity or privileges (although it is permitted to). In principal, capabilities provide a great deal of flexibility for access control and may not be very difficult to implement within the DCE framework. However there is some controversy as to whether this flexibility can be managed in a practical manner.

Impact if requirement not satisfied: Authorization cannot be managed on a principal-by-principal basis without this support (in other words, the resource-oriented ACL model currently supported by DCE continues to be the only one available.)

2.3. {A4} Authorization Based on Location

REQT: Extend the authorization mechanism to permit restrictions based on the location of the user or delegates. The server must be able to verify the locations.

Location may be encoded as the name of the system into which the user logged in, but might more usefully be represented as the name of a group of systems or another orthogonal naming scheme. A vote for this requirement neither requires nor prohibits the location to be specified on ACLs.

Impact if requirement not satisfied: Without this support administrators can't discriminate among laptops and mainframes when stating authorization policies.

2.4. {A5} Anonymous Access

REQT: Permit an environment whereby a server can verify a client's right to access information without the server's gaining knowledge of the client's identity but with the possibility of maintaining individual accountability and auditability.

The term ''identity'' in this requirement refers to the ''individual identity'' that uniquely isolates the individual user, not group identities that pertain to a number of individuals. Authorization in such a case would be based on the client's other attributes in the PAC, rather than on the client's individual identity.

The accountability requirement implies an approach where the server receives the client's individual identity in a sealed form that the server cannot decipher, but which the server can write to an audit log for subsequent retrieval by some trusted authority.

Impact if requirement not satisfied: Without this support some European national privacy laws can't be obeyed by the administrators of a DCE system.

2.5. {A7} Authorization Based on Security Attributes

REQT: All security attributes of the user and delegates that the server can verify (obtained either from the client or by access to other information) should be useful as input to an authorization decision. The API should hide from the server application, as much as possible, the detailed syntax and semantics of this information. Authorization services should provide abstractions to track changes to security attributes of users in a way that is opaque to server applications.

In DCE today servers grant access based on the principal name and group memberships, which servers can obtain in a verified manner from a PAC that the server verifies. In the future, as other attributes are added to the PAC or verified by the server through another mechanism (e.g., by querying a third party), a vote for this requirement is a vote to enhance the authorization services to permit access to be based on these attributes.

There is also a strong statement that the authorization API must be such that applications should not necessarily have to worry about the nature of these attributes. For example, it should be possible to add in the future a new user attribute called ''clearance'' and an ACL tag of ''minimum_clearance,'' and existing applications that use the DCE ACL services should continue to work on objects with these new ACL entries.

The group deferred voting on this item as it was deemed too confusing and would require long discussion.

> *April 1994 Clarification: This requirement should be interpreted as follows: The DCE ''access decision function'' implemented under the DCE API verb* `sec_acl_is_authorized` *should be extended (or extensible by vendors or customers) to take into account information in extended PACs as described in DCE RFC 3. Existing DCE servers should be able to get the benefit of the EPAC information support without any code changes (i.e., the EPAC-based authorization functionality should all be implemented ''under,'' and hidden by, the* `sec_acl_is_authorized` *API verb.). A new administrative interface (for describing what information is to be considered by the access decision function, etc.) may be required.*

Impact if requirement not satisfied: Without this support you can't really use {A2}, {A3}, {A4} and {A8} functionality. Clearly groups aren't the only useful authorization information — though they are the only authorization information which can be used to make authorization decisions by the current DCE security code. Essentially this requirement wishes to have the DCE authorization checking algorithm USE the additional information which would be put into a PAC by mechanisms satisfying {A2}, {A3}, {A4}, and {A8}.

2.6. {A9} Mandatory Access Controls (Labeling)

REQT: Transmit labels associated with client principals and processes for optional enforcement by a server. Provide labels on information against which the client labels can be compared.

This is traditional mandatory labeling and label checking but without a requirement for NCSC or ITSEC evaluatability. Note the contrast with item {P7}.

The group decided to postpone a vote on this requirement.

Impact if requirement not satisfied: Without this support some government markets probably will not be able to deploy DCE.

3. DELEGATION

3.1. {D3} User-to-User Authentication

REQT: A server without access to a long term key should be able to authenticate itself.

For example, servers that do not have long term keys are those activated on diskless nodes by users who log into them. This requirement is one that permits a user's process to act as a server.

> *April 1994 Clarification: This requirement received four ''?'' votes indicating that it was poorly understood. Of those voting, only HP gave it a high priority for the long-term evolution of DCE.*

The goal of this requirement was to enable applications that do not have access to a principal's long term key to be the receivers of a protected RPC. Client applications already have this insulation from access to the user's long-term key. These applications only need access to the login context for the user. Servers, however, need current access to the key.

When a ''server'' is a named, long running system resource — like the name server or the security server — it seems natural that the server is running on a machine with access to the long term key to the identity of that server. This kind of server is not normally associated with a human user, but rather with a pseudo-user identity belonging to the server itself.

This model fits with the well with the type of core DCE services distributed by the OSF today. It will not, however, map well onto all application domains. In particular, some applications need a ''messaging'' model. While the model may be achieved by using RPCs — it is still necessary for the application's ''client'' code to play the role of a DCE ''server.'' This client code performs an RPC to the system service — and the output parameters to the RPC indicate that some activity has been queued. The client code then enters a wait state to get the real reply to the operation (a callback). This kind of messaging model can be found in many object-oriented environments as well as some existing communication protocols.

Using today's security model, there are two ways to accomplish this form of asynchronous activity: The application can force users to make their long term keys available (either in a keytab file which forces the key to be stored in the clear on the local machine, or via an application-provided key acquisition function which requires that the user type their key to an arbitrary application), or the application does not perform a protected RPC operation for the callback but simply uses an unprotected RPC to induce the client code to call the server using protected RPC to retrieve the data. Neither of these two solutions are desirable — the first exposes the user's long term key to arbitrary applications and potentially to clear-text storage on the machine, and the second contorts the application into an extra communication exchange.

Instead, server applications need to be provided with the same sort of insulation that is available for client applications. In particular it should be possible to direct a protected RPC to an application that has access to the login context for its principal rather than just access to the long-term key for that principal. This proposal is a bit more narrow than the way the requirement is currently worded — it would require that authentication be possible only if login context were available to the ''server'' code, while the original statement of the requirement would have admitted other solutions. This narrower scope, however, may be more understandable and covers the needs of many asynchronous or object-oriented environments.

It should also be noted that this requirement can be met using the existing Kerberos V5 protocol. Kerberos allows authentication using a ''user-to-user'' protocol that involves communicating with the TGS (Ticket Granting Service) with two TGTs (Ticket Granting Tickets). The primary change to DCE would be to detect that this protocol should be used rather than the conventional use of the TGS.

Public Key based protocols can also accommodate this requirement. All that is necessary is the generation of a temporary certificate signed by the principal. The user makes this certificate and corresponding keys available to the application rather than the certificate that identifies the user directly.

An example of an application which requires this function is the `talk` utility. Hoyt types `talk Belinda`, and is given the read end of one pipe and the write end of another pipe. Belinda, if she accepts the `talk` session, is given the other ends of both pipes. Now they are able to write messages to one another and read the answers. In order for Hoyt and Belinda to feel comfortable that their conversation is not overheard, it is necessary for them to authenticate one another; i.e., for Hoyt to authenticate Belinda and Belinda to authenticate Hoyt. Note that it's not good enough for Hoyt to authenticate Belinda's server and Belinda to authenticate Hoyt (which is what would happen if the talk daemon were implemented naively as a standard DCE server) — in this case Belinda is confident that Hoyt is at the other end of the conversation, but all Hoyt knows is that someone at Belinda's machine is talking to him.

3.2. {D3b} Delegation While Absent

REQT: It must be possible for a server to be authorized to acquire a user's credentials, using the server's password, and without the user being present to validate the credential acquisition in real time.

SPONSOR: Bob Blakley

Impact if requirement not satisfied: Today's DCE security model makes it quite difficult to deal with two situations:

(a) A user submits a request to a system service for an action to be performed by the service at a later time. At that later time (potentially much later — maybe a week, or a month, or even a year), the service dequeues

the user's request and executes it, possibly trying to access some resources on the user's behalf during the execution of the request. This is the classic ''batch processor'' scenario. Allowing the batch job executed by the batch processor to execute under the user's identity is very difficult with today's DCE architecture.

(b) A user logs on to her machine's operating system but not to DCE. Later, the user runs an application, which needs to issue a DCE call to get some work done. The user may or may not still be logged on: an example case in which the user will not be logged on is if she executed `at 1145 lpr -Pmyprt foo.txt`, where `at` is the UNIX command, and `foo.txt` is a file in the DCE DFS (which will of course have to be read in order to be printed). The operating system needs to acquire a DCE ticket in the identity of the user in order to execute the application and get the requested work done. But it may not have access to the user's DCE password — especially if the user is not logged on.

DCE should allow a user or system administrator to designate certain servers as ''trusted''; allow a ''trusted'' server to obtain TGTs for users, using the server's long-term key in place of the user's password.

3.3. {D4} Integrity Protection of Requests

REQT: Provide a service that permits a client to protect a request in a way that a server can detect if the request was modified by a delegate.

One implementation of this would sign (or encrypt) the client's request under a key specified in the PAC, thereby preventing a delegate from modifying the request in an undetectable manner. For the purposes of such ''locked'' requests, the delegate is performing mostly a routing function.

Impact if requirement not satisfied: Without this support, it may not be possible for a server to determine whether delegates have faithfully performed the operation requested by the initiator.

3.4. {D5} Integrity Protection of Responses

REQT: Provide a service that permits a server to protect a response in a way that a client can detect if the request was modified by a delegate.

4. EXTENSIONS TO OTHER APPLICATIONS AND MECHANISMS

4.1. {E1a} Secure Remote Login

REQT: Remote login should be secured using DCE mechanisms and offered as part of the DCE package.

Candidate applications that fall into this category include `telnet`, `rlogin` or other terminal emulation programs.

Impact if requirement not satisfied: See {E1b}.

4.2. {E1b} Secure Remote File Transfer

REQT: Remote file copy should be secured using DCE mechanisms and offered as part of the DCE package.

This includes `ftp` and `rcp`.

Impact if requirement not satisfied (applies to {E1a} and {E1b}): Users will need to log in separately to DCE and to other popular UNIX and Internet utilities. Lack of an enterprise-wide single signon capability is a significant source of customer dissatisfaction with all vendors' products.

4.3. {E3} Other User Authentication Methods

REQT: Provide interfaces that permit arbitrary site-selected user authentication methods, including smart cards and biometric devices.

4.4. {E4} More Options for Communications Security

REQT: Permit site or application-selectable options for protecting communications such as choice of encryption algorithms.

Note that DCE today already contains hooks that to identify alternative encryption algorithms in the protocols. This item refers to items such as the provision of additional algorithms, registration of multiple algorithms,

additional flexibility in the choice of algorithms by applications, and negotiation in the handling of multiple algorithms.

4.5. {E5} Application Security Services

REQT: Permit an application to gain access to enough of the underlying DCE security infrastructure to provide security services such as digital signatures, data confidentiality and nonrepudiation without the need for the application to implement another user and system registration facility and authentication mechanism.

Many application-level security mechanisms, for applications such as privacy enhanced mail and electronic document interchange, require facilities to permit registration of users, installation of user secrets (passwords or private keys), and check of user names in order to make authorization or authentication decisions. These facilities are conceptually similar to those already provided by DCE but differ in most details. This is a requirement to install services in DCE to access the existing DCE mechanisms so that applications will not need to implement their own. It may also require augmenting the DCE facilities to better service varying needs of applications. (See requirements {PK9b} and {PK9c} for specific applications to be supported.)

4.6. {E7} Algorithms

REQT: Permit independent selection of algorithms for confidentiality and integrity.

The requirement is that, in the case where messages are protected with both confidentiality and integrity, there be independent selection of the algorithms used for the two mechanisms.

Impact if requirement not satisfied: Without this requirement it may be difficult to export strong-integrity, weak-confidentiality DCE configurations.

4.7. {E8} GSSAPI Framework

This requirement proposes a set of enhancements to the DCE GSSAPI implementation related to incorporation and installation of alternate supporting GSSAPI security mechanisms, and to criteria to be employed in mechanism selection.

Sub-requirements:

{E8a} Service Provider Interface for GSSAPI.

The DCE implementation of GSSAPI shall support an explicit service provider interface to underlying mechanisms, enabling developers and integrators of DCE systems to incorporate separately-developed implementations of those mechanisms in a modular fashion.

{E8b} Dynamic Installation of Mechanisms.

In addition to {E8a}, the DCE implementation of GSSAPI shall enable administrators to install GSSAPI mechanism modules within an end system's installation, and thereby reconfigure the set of GSSAPI mechanisms available within that end system.

{E8c} Share Credentials Among Mechanisms.

It must be possible for several security mechanisms with different cryptographic technologies and different available combinations of message confidentiality and message integrity to share the same key distribution and credential distribution mechanisms. It must be possible in particular for several different mechanisms to have access to the same copy of a user's credentials (for example, a user's DCE PTGT). Note that satisfying this requirement will entail exposing DCE interfaces through which a DCE PTGT can be retrieved from the credential cache. A possible way to meet this requirement would be to implement a ''credential cache manager'' with a published interface, and modify the DCE security code to use this interface.

{E8d} Separate Cryptographic Functionality.

It must be possible to write mechanism code without having to implement crypto function; crypto function should be provided as a system service under an API. Crypto code should be separable from mechanism code so that crypto code can be replaced for exportable versions and so that hardware crypto support can be introduced. Replacement of crypto code and introduction of hardware crypto capability should both be possible without requiring changes to mechanism or application code. Note that X/Open is soliciting candidates for a standard crypto API;

this work should be examined for appropriateness.

{E8e} Mechanism Selection by QOP.

The DCE implementation of GSSAPI shall enable callers to determine the set of underlying mechanisms which satisfy a specified set of ''quality of protection'' parameters (e.g., caller-specified requirements for per-message confidentiality and integrity services).

4.8. {E9} Single Signon

NOTE: Most of this text previously appeared in the justification for requirements {E1a} and {E1b}.

There are two issues which must be addressed to achieve ''single signon'' in a heterogeneous distributed network:

(a) Integration of DCE login with local OS login, particularly in the case where the local OS is NOT a UNIX-type OS. This is the issue of adding extra attributes to the DCE user registry for operating systems that need extra information stored — for example, IBM's MVS. This is addressed by requirement {P10a} and is satisfied by the DCE 1.1 RFC 6.0 work. Note that additional work by DCE vendors is needed to exploit this support and produce an integrated OS/DCE login.

(b) Integrating DCE login with `telnet`, `rlogin`, `ftp`, `rsh`, etc. The issue here is that Kerberized commands would not necessarily use the same tickets as those used by DCE, particularly on nodes that don't otherwise support DCE. Morrie Gasser raised this issue in the SIG many months before the extra attributes issue was raised; it can be summarized as follows:

> If I am on a machine using DCE and I quote my password appropriately, so that I can use authenticated RPC, can I also use the tickets/credentials/whatever to do `telnet`/`rlogin`/`ftp`/`rsh`/etc. to other systems on the network that speak Kerberos, but do NOT speak DCE. If I have single signon, then I should only have to login once, and that login should work with both DCE and non-DCE Kerberos. The issue is one of human interface, not what particular flavor of tickets are used by what applications. I don't really care whether DCE uses vanilla tickets or chocolate tickets, as long as DCE and regular Kerberos can interoperate securely.

To properly support single signon, both issues have to be solved. A user expects single signon to be just that — login a single time and have that authentication used everywhere. This means that DCE has to be aware of non-UNIX operating system attributes, at least to some extent, that DCE Kerberos should interoperate with regular Kerberos, even with machines that don't support DCE itself, and finally that a local OS should support local login reliably, even if network links go down.

For single signon to really work with DCE and other Internet services, the following should be possible for a user who does NOT have access to source code:

(a) Login once to talk to other DCE applications on other machines.

(b) Login once (same login as above) to do `telnet`, etc., sorts of things to Internet machines that run DCE.

(c) Login once (same login above) to do `telnet`, etc., sorts of things to Internet machines that run Kerberos, but do not run DCE.

(d) Receive Kerberized `telnet`, etc., from Internet machines that run Kerberos, but do not run DCE.

(e) Integrate DCE login with non-UNIX operating system login, so that I can connect to or from non-UNIX operating systems and still have things all work right (i.e., the extra operating system-specific attributes stuff).

(f) Sufficient local login capability (perhaps by caching or whatever) so that I can login locally to my ''home'' system(s), even if the network links are broken.

(g) Login once to talk to local applications which have their own security subsystems.

(h) Login once to talk to both ''Kerberos-based'' (DCE and other Kerberos) remote servers, and remote servers which use other secure protocols (e.g., Novell NetWare, Lotus Notes, etc.).

Items (a) through (d) include the requirements called for in {E1a} and {E1b}; (e) is addressed by the DCE 1.1 RFC 6.0 work, and (f) is a vendor issue. Items (g) and (h) are not addressed by other requirements herein; some DCE enabling may be desirable for these capabilities.

5. IDENTITIES AND GROUPS

5.1. {I1} Global Groups

REQT: Permit the ability of a group to include members in any cell while retaining the shorthand, accountability and auditing capabilities of current DCE groups.

This requirement calls for four things:

(a) The registry administrator of a group's home cell should have control over group membership (i.e., should be able to add or delete members at will and should be able to prevent others from doing so).

(b) Any group, from any cell, should be able to be added to a DCE ACL.

(c) DCE cell's Privilege Server should seal a user's active DCE groups into that user's PAC when the user issues a DCE request from within the cell.

(d) DCE Privilege Servers receiving cross-cell ticket-granting requests should preserve the originating cell's group membership information but should be able to seal additional DCE groups (specified in the receiving cell's Registry Service) into the generated ticket.

Impact if requirement not satisfied: Without this support administration of DCE authorization based on groups is difficult in multi-cell environments.

5.2. {I2} Preserve Rights When Principal Changes Cell

REQT: It should be possible to set up an environment whereby a user registered in a cell does not necessarily lose access to everything when changing registration to another cell.

Given the facts that a user identity consists of a single cell UUID and user UUID, that ACLs contain the client's cell UUID, and that there is no easy way to find and then change all the ACLs in all remote cells on which a given user is listed, a user who moves to a different cell and authenticates himself under that new UUID will lose access to all files protected by DCE ACLs. Satisfying this requirement to retain access requires either the client's PAC to name all the client's old identities, or the server to support or interrogate a facility that maps old names on ACLs to new names.

Impact if requirement not satisfied: Without this support, organizational changes cause large security administration headaches.

5.3. {I3} Role Based Administration

REQT: DCE should provide the facilities to allow for ''role based'' security administration. This requirement is not meant to create another type of non-discretionary access control mechanism. Role based administration should be looked at as an ease-of-use issue for security administrators. The work involved in meeting this requirement must include determining the appropriate definition of ''roles''.

Impact if requirement not satisfied: Administration of user privileges will remain a complicated and time consuming task.

5.4. {I4} rgy_edit Sub-Administrator Support

SPONSOR: Charles Blauner

REQT: The `rgy_edit` program must be modified to support the concept of ''sub-administrators''. In a large enterprise, the job of userid administration is typically divided among many people. Each person is given a certain sub-set of all the users to administer. The `rgy_edit` program must be modified to allow the security administrator to define sub-groups of users that a ''sub-administrator'' can do basic userid administration for, while preventing that person from doing administrative functions on users in other sub-domains.

April 1994 Clarification: This requirement is an interface requirement. It is not necessary to change underlying administrative functionality as long as the user's view of administration is simplified.

Impact if requirement not satisfied: If this requirement is not met it means that the task of userid administration in a large enterprise will remain an overly complex task to do securely and efficiently.

5.5. {I5} Privileges

SPONSOR: Bob Blakley

REQT: DCE must implement a notion of ''privilege'' and a means of administering assignment of privileges to users through the Registry.

April 1994 Clarification: The notion of privilege here is not the one defined, for example, in the Compartmented Mode Workstation specification. Tnstead, what is meant here is that special administrative roles (examples Tight include ''GUEST, USER, OPERATOR, ADMINISTRATOR, AUDITOR, SUPERUSER, TNSTALLER'') should be defined and some mechanism defined for permitting users with certain privileges to perform some system operations regardless of DAC (ACL-based) authorization.

6. PERFORMANCE, ROBUSTNESS, EXTENSIBILITY, INTEGRATION, STANDARDS

6.1. {P4} Recovery from Security Compromise

REQT: After a detected security compromise, it must be possible to continue operating with an acceptable level of security and functionality. What is considered ''acceptable'' should be configurable by the security administration.

April 1994 Note: This requirement is already known to need clarification, but will be included in the August ballot anyway.

Important outstanding questions for those who feel this requirement is important:

(a) Who is the sponsor of this requirement?

(b) What behavior might satisfy this requirement?

(c) What policies for continuity of service should be supported?

(d) How serious must a compromise be before the system should not continue?

An example of one of the most serious compromises is disclosure of the secret key of a cell's privilege server. A less serious compromise is disclosure of one user's password. DCE must recognize that commercial requirements vary considerably — some organizations want to shut down completely when a compromise is detected, and others are willing to operate completely insecurely, if necessary to keep the business running. Therefore, a configurable range of emergency modes of operation should be provided. Note that this item does not place any requirement on the ability to detect a security compromise.

Impact if requirement not satisfied: Unknown.

6.2. {P5} Detection of Denial of Service

REQT: Maximize the ability to detect and diagnose denial of service attacks.

April 1994 Note: This requirement is known to require clarification, but will be included in the August ballot. A sponsor for this requirement is needed.

Impact if requirement not satisfied: Unknown.

6.3. {P6} Algorithms for Security Protocol Messages

REQT: Retain flexibility in the choice of algorithms and techniques for protecting messages in the security protocol.

April 1994 Note: This requirement is known to require clarification, but will be included in the August ballot. I believe that Denis Pinkas is the sponsor of this requirement; I need confirmation of this, or an alternative sponsor. I also need additional explanatory text.

In contrast to item {E7}, this item refers to the ability to select algorithms to protect the messages within the security protocols as well as algorithms to protect application messages.

6.4. {P10b} Kerberos Version 5 Interoperation

REQT: Interoperate with Kerberos version 5. In particular, the DCE security server should be tested to insure that it is a fully RFC 1510 compliant provider of Kerberos 5 services.

SPONSOR: Joe Pato

April 1994 Note: What other requirements exist for interoperability?

The old text stated a general requirement for interoperability with secure, non-DCE environments.

6.5. {P14} Defaults

REQT: Permit establishment of an environment whereby an application not otherwise interested in security can make maximum use of security features as configured by an administrator.

In designing APIs to support security, it should be possible for an application to choose (or automatically use) administrator-configured defaults wherever practical, as most applications, especially the client side of applications, have little basis on which to make a rational security-relevant decision. For example, one default configured by an administrator or user might be whether or not to delegate on outbound RPC calls. To enable this capability, the RPC delegation mechanism should allow an application to specify ''default'' as a delegation option.

Impact if requirement not satisfied: Without this support, the strong security features of DCE will typically remain unused! ''Unsophisticated'' users should get some benefit from all OSF's and the suppliers' hard work by default, out of the box.

7. INFRASTRUCTURE

7.1. {R1} Registry and Naming Convergence

REQT:

NOTE: Definition deferred until we find out the results of the naming working group.

7.2. {R7} Pull Model

REQT: Permit an environment where a server can obtain and verify all of the user's security attributes relevant to an authorization decision without action by or cooperation of any of the clients or delegates.

A vote for this item is a statement that there will be important cases where the client (more precisely, the user, the workstation or the user's privilege server) does not necessarily know, at the time a PAC is created, all the privileges (especially group memberships) that will be necessary to satisfy a subsequent access request. This requirement becomes much more relevant if DCE is extended to permit groups to name members outside their cell, as there is no way for any system to create a PAC naming all groups of which the user is a member. Since the server knows exactly what attributes are required to satisfy a request, a vote for this item permits an option whereby the server can verify the user's attributes even if the user does not supply them.

7.3. {R8} Client-Server Security Negotiations

SPONSOR: Charles Blauner

REQT: A DCE client must be able to determine the security characteristics of a DCE server before making an RPC call. The client should be able to determine whether or not the server supports the level of security that the client requires. If the server does not support the level of security the client requires, the client could then import the binding handle from a different server that provides the same service and check for the security provided by the new server. It would then be up to the client to find a server with adequate security or knowingly make the RPC call to a server with a lower level of security.

April 1994 Clarification: Servers should advertise the range of security levels they will support in their CDS entries; this information should be returned by CDS to clients in the binding.

Impact if requirement not satisfied: Currently, if a client makes an RPC call to a server without knowing the level of security supported by that server, there are three possible outcomes to that attempted RPC call:

(a) Client-Server mismatch; the RPC call fails and the application dies quietly or produces a runtime error depending on certain parameters.

(b) Client-Server mismatch; the RPC call succeeds, but the level of security provided is less than the client expected. In this case the client does not know that it is not getting the expected level of security.

(c) Client-Server match; the RPC call succeeds with the correct level of security being requested by the client being provided.

Of the three possible outcomes, the first two are not acceptable when maintaining a secure applications environment. Failure to meet the requirement means maintaining status quo in this unacceptable situation.

7.4. {R9} acl_edit Command Language Extension

REQT: The `acl_edit` command language needs to be extended to allow a user to do all the necessary processing for setting-up and maintaining ACLs without having to exit and re-execute it.

Impact if requirement not satisfied: Currently, a user must execute the `acl_edit` program three separate times using different options in order to complete the simple task of giving a user full access to a directory. This is not user-friendly or an acceptable overhead on administrators. If the requirement is not met, this problem will continue to exist.

7.5. {R10} Workstation Lockout Capability

REQT: The System Security Administrator (or Officer) needs the ability to lock a workstation out from DCE, so that no access to DCE services is possible from that machine.

> *April 1994 Notes: What level of ''no access'' is acceptable? Is ''unauthenticated'' access permissible from a locked workstation? How will machines be identified so that the administrator can specify them through the lockout interface, by machine principal name, IP address, or some other way? Is this capability intended for use in conjunction with some kind of intrusion detection? Are there operating system or other prerequisites for implementation of this requirement?*

7.6. {R11} RPC Security Level Negotiation

REQT: A server should be able to directly upgrade the security in a communication. Today the security, encryption and MAC, are set by the client's request. If the client requested security is inadequate, the server must deny a request even tho the requestor may be authorized. The client can then make another request, hoping to specify the required security. To avoid this, the server should be able to upgrade the security requested by the client and respond to the request with the appropriate security on the link.

> *April 1994 Clarification: An acceptable implementation of this requirement would be the following:*
>
> *(i) A return code from the server indicating failure due to impermissible security level along with an indication of what security level would be acceptable.*
>
> *(ii) A mechanism to permit the client code to retry the request with the specified security level — after passing a check to insure that the specified level is acceptable to the client — automatically.*

Voters should be aware that implementing this will be a large change to the DCE RPC AES.

8. AUDITING

8.1. {T2} Remote Auditing

REQT: Record security-relevant events on remote centralized audit systems.

8.2. {T4a} Separation of Roles

SPONSOR: Chii-Ren Tsai

REQT: An auditor principal should be created to solely perform audit system management to enforce ‘‘separation of roles.’’ Audit system management includes modification of the event selection list (ESL), invocation of audit daemon, audit trail analysis and audit trails backup and restore.

8.3. {T4b} Audit Policy Protection

SPONSOR: Chii-Ren Tsai

REQT: The audit event filters must not be modifiable by system administrators other than auditors.

8.4. {T5} Administrative Tool for Audit Events Selection

SPONSOR: Chii-Ren Tsai

REQT: It is necessary to provide an administration tool for audit events selection since to modify ESL in terms of large number of principals and groups is not an easy effort. Also, the tool should be able to provide information about principals, groups, realms and audit events, and detect any errors.

8.5. {T6} New Audit Event Class — Object Creation

SPONSOR: Chii-Ren Tsai

REQT: A finer level of granularity should be applied to ‘‘Object Creation’’, similar to ‘‘Object Deletion’’, rather than generalizing it under ‘‘Controlled Access’’ events.

8.6. {T7} Audit Events Allocation

SPONSOR: Chii-Ren Tsai

REQT: To ensure lack of collisions between audit records from different servers audit events definitions, namely set-id and event-id, are preassigned by major vendors with preagreement. This is a policy issue instead of a technical issue. If OSF can coordinate the definitions, it would be much easier for vendors and users.

8.7. {T8} Audit Events Expansion

SPONSOR: Chii-Ren Tsai

REQT: New audit events can be added to the audit system and be recognized by the audit daemon. The mechanism to expand audit events including how to define event specific information in audit records must be specified.

8.8. {T9} Audit Daemon Status Alert

SPONSOR: Chii-Ren Tsai

REQT: Whenever the audit daemon is killed, an alarm must be sent to the auditor.

8.9. {T10} Audit Trails Storage Full

SPONSOR: Chii-Ren Tsai

REQT: The following kinds of behavior were described in the RFC to address the audit log full occurrence:

(a) Default — Audit daemon stops writing audit records to the log.

(b) Override — Audit daemon discards old audit logs.

Neither of these behaviors are acceptable. To resolve this issue, two provisions need to be done:

(a) The auditor must be alarmed by the audit daemon whenever a configurable threshold for the audit trail filesystem is reached, e.g., the filesystem is 90% full. Optionally, the alarm can trigger a preconfigured program to backup audit trails automatically in order to preempt space.

(b) Several audit trail servers are configured, so that whenever a server is full, audit trails will be transferred to another server automatically. In the mean time, an alarm is sent to the auditor via a preconfigured method. The audit daemon must be aware of the configuration of audit trail servers. Consequently, audit trails distributed on different servers can be reconstructed to ensure that audit records are stored in order.

8.10. {T11} Audit Trails Protection

SPONSOR: Chii-Ren Tsai

REQT: Audit trails must be protected against unauthorized examination and modification. This protection should not rely upon security characteristics of the underlying operating system (this probably means that the audit trail will have to be encrypted to satisfy this requirement).

8.11. {T12} Performance

SPONSOR: Chii-Ren Tsai

REQT: The performance impact of audit daemon on a system must be measured and published.

8.12. {T14} Audit Reduction Tool for Audit Trail Analysis

SPONSOR: Chii-Ren Tsai

REQT: It is desirable to provide a tool for audit trail analysis. The tool can be used to trace security or access violation, select audit records by a combination of attributes from the audit header, and generate audit reports.

8.13. {T16} Before/After events

SPONSOR: Chii-Ren Tsai

REQT: It should be possible to log the contents/value of a system resource before an auditable action is performed on that resource, and to record the modification requested and the success or failure of the modification. The idea here is to be able to reconstruct the original state of the resource from the information in the audit log.

8.14. {T18} Hierarchical Audit Events Filtering

SPONSOR: Chii-Ren Tsai

REQT: To support hierarchical audit event managements, the interpretation of an ESL must go through the whole list instead of first match. In addition, if there is an overlap in terms of a user and the user within a group or multiple groups, the union of actions should be taken by the audit daemon. For example, the user `micky@gftpd.citicorp.com` belongs to two groups, `tpd` and `odessa`, and the ESL is shown below:

```
princ:micky@gftpd.citicorp.com:Authentication:FAILED:LOG
group:tpd:Authentication:FAILED:ALARM
group:odessa:Authentication:FAILED_ACCESS:ALARM
realm:gftpd.citicorp.com:Authentication:ALL:LOG
```

In this example, FAILED and FAILED_ACCESS attempted by `micky` should be logged and alarmed, and SUCCESS by micky should by logged in the context of Authentication event class.

8.15. {T19} Default Configuration of Auditable Events

SPONSOR: Chii-Ren Tsai

REQT: If no filter is specified, the audit daemon should send an alarm to the auditor instead of assuming all events are logged which may generate too many audit records. So, it is good to have some defaults audit events specified in a default event list, which should not be overridden. However, if there is any filter specified in the ESL, the default list would be ignored.

9. MISCELLANEOUS

9.1. {X1} Loading and Booting Security

REQT: Provide a interfaces and an infrastructure whereby it is possible for a user to have increased confidence that he is interacting with legitimate software on his node, free from Trojan horses and viruses. This includes mechanisms permitting the hardware to verify the booted image on a diskless node, as well as the operating system being able to detect corruption of applications software. To the extent possible, this infrastructure should be integrated with the existing DCE registration infrastructure.

Providing secure booting of systems requires both hardware and software cooperation. To boot securely, a workstation (e.g., the boot ROM) must verify the integrity of a software image it receives before being willing to execute it. Similarly, for secure loading of applications (e.g., at the time a Unix process calls exec()), the operating system must be able to verify the integrity of the executable file. While major portions of the solution to this problem are outside the scope of DCE, DCE can provide an important part of the management infrastructure and support tools that allow vendors (or a subsequent OSF offering) to provide the full service in an interoperable way.

9.2. {X7} Maker/Checker for Security Administration

SPONSOR: Chii-Ren Tsai

REQT: It must be possible to control updates to the DCE security database using a separation of duties authorization model. A ''Maker/Checker'' model requires that two principals be involved in any policy related or authorization data change to the DCE security database. One principal with ''Maker'' privileges can enter the data and describe the changes to be made; a second principal with ''checker'' privileges can authorize or deny the changes described by the maker, but is not allowed to modify the data.

10. AUTHENTICATION

10.1. {N1} Alternate Authentication Mechanism Support

REQT:

> *August 1994 Note: This requirement now calls for the addition (or at least the support) of other (non-Kerberos) Authentication Services under GSSAPI in a DCE environment.*

10.2. {N2} dce_login Error Feedback

REQT: The `dce_login` program must be modified so that it does not provide any error feedback that could help someone that is attacking a system in a DCE cell. It should solicit a password whether or not a valid userid has been entered. The same error message should be returned for both invalid userids and invalid passwords.

Impact if requirement not satisfied: If the program is not modified, its use provides an attacker with additional information that can help attackers to penetrate a system.

10.3. {N3} dce_login Password Expiration Handling

REQT: The `dce_login` program must be modified to properly handle password expiration. The program currently allows users to login with an expired password. This means that DCE is not enforcing the password expiration policy set in the cell's registry. It must also provide a mechanism for changing a password when one has expired.

Impact if requirement not satisfied: Failure to meet this requirement means that the program will continue to fail to enforce the passwords expiration policy set by the cell administrator.

10.4. {N5} Login Policy Enhancements

REQT: DCE must give the administrator the ability to prevent a principal from logging on to a cell after N successive login failures. The value N needs to be specified by the Registry policy.

> *August 1994 Note: This is satisfied at the master registry but not at replicas in DCE 1.1.*

DCE must be able to notify a user at login time if he currently has a session active within the cell (this should be an option).

> *August 1994 Note: This calls for a modification to the* `sec_login` *API and the* `dce_login` *program to notify the user of the last login time and to tell the user if any of those logins is still active (i.e., credentials haven't yet expired).*

DCE must be able to notify user that his password will expire in N days. The valueN needs to be specified by the Registry policy.

> *August 1994 Note: This is a requirement on* `dce_login`.

DCE must allow administrators to prevent users from logging in during specified time periods.

> *August 1994 Note: This is satisfied when* `sec_login_policy_check` *is implemented (this function appears in the DCE 1.1 extended login spec but will not be implemented in DCE 1.1)*

10.5. {N6} Current PAC for Credential Refresh

SPONSOR: Jeannine Lonski

REQT: The DCE credential refresh process should obtain a ''truly updated'' PAC, whereas currently, it obtains an updated PAC which reflects only group deletions since last login, and not additional groups.

Impact if requirement not satisfied: Applications which wish to renew credentials via a refresh as opposed to a re-login, will not allow principals to gain the benefits of additional privileges due to new group memberships aquired after login. From a usability perspective, refreshing credentials is preferred over re-login because it allows the same credential file to be used. Thus, applications can offer a credential refresh process which allows periodic re-authentication without interrupting application work. Re-login on the other hand, requires users to exit out of an application and re-enter, as it changes the credential file, and UNIX restricts processes from changing the environment variables (credentials are pointed to by the KRB5CCNAME environment variable) of their siblings or parents.

10.6. {N7} Role-Based login

SPONSOR: Joe Wallin

REQT: DCE login information should include a username, password, and rolename. Access control should take role-based privileges into account, but accountability (auditing) should be done based on the username or corresponding id.

10.7. {N8} Log-Over

SPONSOR: Joe Wallin

A ''log-over'' function should be provided which requires username password, and rolename as input. This function should change active username and rolename as specified, and accountability and auditing should take place under the new identities. Execution of log-over should not require termination of any applications running under the original identity.

Impact if requirement not satisfied: Today's DCE makes shift-change operations difficult, since there is no way to change the active user identity associated with a console without shutting down the applications running at that console. Without a log-over function, this problem will continue to exist.

11. SYSTEM INTEGRITY

11.1. {S1} Better Support for Security Policy Data Update Integrity

REQT: This problem is best illustrated using the following example (of a problem which can arise during ACL administration).

Two users, `A` and `B`, both have administrative authority with respect to a resource `R`. User `A` accesses ACL on resource `R`, intending to add an entry for `Y` (a user or group). User `B` accesses same ACL, intending to update permissions for `X`. User `A` completes the addition. User `B` completes the change. The new entry for `Y` no longer exists; this results in two errors:

(a) User `B` deleted an entry (not `B`'s intent).

(b) User `A`'s completed (and confirmed) update is not in place.

DCE DACLs have no mechanism to prevent ACL update collisions like those illustrated above. This problem is particularly severe for ACLs with many entries, since granularity of DACL manipulation is the entire ACL (i.e., `rdacl_lookup` returns the entire ACL and `rdacl_replace` replaces the entire ACL).

Note that this problem also exists for Registry data; two users with admin authority with respect to the same registry object (say, an account) can make changes in such a way that one set of changes is inadvertently destroyed.

(e.g., if adminA and adminB's `rgy_edit` sessions both read the data for a particular account at the same time, then they each change something within that account data and do a replace, one of the two changes may be lost. The chances of it happening are minimal — even more so than with `acl_edit`, since only administrators run `rgy_edit` to make changes like this, but everybody uses `acl_edit`).

At the Paris SIG meeting I circulated a document which proposed some solutions to this problem, which the group discussed. The (quite correct!) conclusion was that my solutions were far too drastic. I think on further reflection that the two most reasonable solutions are:

(a) Add an interface like the Unix command line `chmod` interface, whereby an administrator can specify `+rw` or `-r` etc. This eliminates many of the worst side-effects without radical changes to code.

(b) A slightly more involved solution would be a ''test-and-update'' API verb which would commit a change only if the ACL matched an ACL specified in an input parameter. The typical use of the API set would then be:

```
read ACL'
make a copy (ACL'')
change the copy
call test-and-update (ACL', ACL'')
```

The changes found in ACL´´ would be committed only if the object's ACL ''still'' matched ACL´.

12. PUBLIC KEY

12.1. {PK1} Public-Key Authentication

SPONSOR: John Wray

REQT: Allow DCE security to operate without on-line trusted servers.

This is the requirement for public-key authentication in DCE. The rest of this section provides a justification for public-key (as opposed to the current private-key) authentication mechanisms.

Currently, the database maintained by `secd` contains much information that must be kept confidential (e.g., principal keys) as well as information that must be integrity-protected (e.g., principal privileges). This makes `secd` a valuable target for attack, and the presence of confidential data within it makes recovery from a compromise expensive (requiring every principal within the cell to be issued a new key).

This requirement is that such sensitive security data to be either removed completely from DCE, or where that is impossible, that it be moved into off-line agents, which can be better physically protected than the current on-line servers.

Impact if requirement not satisfied: Without this feature, system administrators must make a trade-off between availability of the security service (more security servers), and security (few, well-protected security servers). Recovery from a compromise of any security server will remain a very costly process.

12.2. {PK2} Persistent Non-Repudiation

SPONSOR: John Wray

REQT: The ability for a message reipient to prove at a (perhaps substantially) later date the identity of a message's originator.

This requirement assumes the initial co-operation of the message originator in attaching a signature to the message. The message recipient must be able, at a later date, to verify the signature, in such a way as to prove to a third party the message originator's identity. The ''perhaps substantially'' clause in the requirement statement implies that any keys necessary for verifying signatures must be retained in an integrity-protected repository. This requirement differs from {PK4} in that it would be acceptable here for message originator/content verification to require information that is known only to the message recipient.

Impact if requirement not satisfied: if this feature is not provided by the base DCE services, trusted messaging applications (e.g., expense approval) must implement this functionality themselves. Without a DCE framework for

storage and retrieval of signature keys, each such application is likely to solve the problem in its own, incompatible way, resulting in greatly increased management complexity.

12.3. {PK3} Establishment of Inter-Cell Trust

SPONSOR: Joe Pato

12.4. {PK4} Signatures for Broadcast Integrity/Identity.

SPONSOR: John Wray

REQT: users should be able to attach signatures to documents or name-space entries that can be verified at a (perhaps substantially) later date, providing proof of both integrity of document content and origin.

This is a variation on {PK2}, requiring non-targetted signatures (i.e., signatures that may be verified by any principal, rather than only by the principal to whom the message is addressed).

Impact if requirement not satisfied: Distribution of trusted (integrity-protected) information to multiple targets will require that each target be individually contacted so that the information may be transmitted using DCE session-oriented integrity protection mechanisms. If document signatures were provided, the information could be distributed simply by placing it in a widely-accessible place (e.g., the name service).

12.5. {PK5} Role-Based Persistent Non-Repudiation

SPONSOR: John Wray

REQT: An extension of {PK2}, explicitly requiring that the role in which the message signatory was operating be established as part of the signature verification process.

This extends {PK2} so that the current role(s) (under which the message originator is operating), become encoded in the signature in a non-forgeable way, and that subsequent verification of the signature reveals those roles (and not just the authorized roles available to the originator).

> *NOTE: This requirement is an enhancement of {PK2}; should there be a corresponding enhancement of {PK4}, or can we take this requirement as applying to both {PK2} & {PK4}?*

Impact if requirement not satisfied: If this feature is not provided by the base DCE services, trusted messaging applications (e.g., expense approval) may be forced to implement this functionality themselves. If roles are introduced at an application level, different applications' role models are likely to be incompatible with one another, resulting in increased management complexity.

12.6. {PK6} 2-Party Confidentiality

> *NOTE: Can Denis Pinkas sponsor this?*

REQT: Allow two parties to establish a shared session key without entrusting that key to the secrutiy server.

12.7. {PK7} Server-Less Verification of Foreign EPACs

SPONSOR: John Wray

REQT: The ability for an application server to directly verify an EPAC generated in a foreign cell, without requiring intermediate privilege servers to ''translate'' or otherwise approve the EPAC.

Application servers should be able to directly verify the signatures and checksums that protect EPACs, even when those signatures or checksums were applied by a foreign cell's privilege server. This implies that DCE should include a trusted means of obtaining any necessary verification keys for foreign privilege servers.

Impact if requirement not satisfied: not known.

12.8. {PK8} Asynchronous Messaging (Non-Trusted Store & Forward)

SPONSOR: Chii-Ren Tsai

Requirement: It enables a client to sign or seal a message that can be transferred to a peer/server via an arbitrary number of message forwarders or queues. The peer/server should be able to verify or unseal the message on receipt, so that end-to-end secure communication can be maintained.

For example, if Citibank in New York decides to send a financial message via the Internet to Citibank in Hong Kong to transfer funds. In this case, the message must be signed and sealed in New York since it will go through various untrusted queues of intermediate nodes before it arrives Citibank in Hong Kong. Thus, end-to-end security must be maintained. Otherwise, it could cause an unmeasurable loss. Currently, data encryption in DCE is based on DES, which is not strong enough to maintain data confidentiality of the message, especially the message is exposed to a number of untrusted nodes and is vulnerable to brute-force attacks because each node can hold the message for a certain amount of time which may be long enough to compromise the key used to encrypt the message.

Impact if requirement not satisfied: Under this circumstances, a proprietary solution must be seeked to enhance end-to-end security for asynchronous messaging; therefore, DCE security service cannot be used as an enterprised-wide security solution.

12.9. {PK9a} Secure E-mail

SPONSOR: Gary Gaskell

REQT: DCE should support secure email.

12.10. {PK9b} Privacy-Enhanced Mail Integration

REQT: Provide some integration between the DCE registry and the PEM infrastructure for registering users to minimize the additional work system managers and users have to do to participate in both environments, with the caveat that such integration should not be mandatory for those administrations choosing to keep them separate.

This item is similar to the generic requirement in {E5}, except it relates to PEM in particular. An example of integration might be the storage of PEM certificates and/or corresponding private keys in the registry, or the automatic initiation of a certificate creation process for PEM as a user is registered in DCE. Desirable points of integration are left to further study.

Impact if requirement not satisfied: Without this support, PEM users will have to log on twice to send and receive mail.

12.11. {PK9c} X.400 Security Integration

REQT: As {PK9b}, for X.400.

Impact if requirement not satisfied: Without this support, X.400 users will have to log on twice to send and receive mail.

12.12. {PK10} Direct Trust Relationships

SPONSOR: Tom Hunwick

REQT: The ability for a principal (registered in DCE cell-A) to directly establish a trust relationship(s) with a principal from a foreign cell (DCE cell-B), without involvement of the respective cell administrators.

The requirement of establishing direct trust does not necessarily imply that there was never a-priori security knowledge exchanged between the two organizations (intra or inter company). Interpretation of an organization's security policy will determine if it is acceptable for principals in these two organizations to use an established trust relationship. Security policy statements might include items like:

(a) A principal (other than an administrator) cannot establish a trust relationship for its organization.

(b) A principal can only establish a trust relationship with approved principals from approved cells.

Currently, the method of establishing trust between cells is to either allow everyone in the cell (the whole cell) to be trusted, or no-one in the cell is trusted.

This requirement is asking for the ability to establish a direct trust relationship at session connect time, given that the security policy allows for the session to be established.

A couple of assumptions can be stated:

(a) Every principal must know and trust at least one (their own) certification authority.

(b) Each principal can obtain a digitally signed list of other trusted certification authorities from its own.

Impact if requirement not satisfied: Cell administrators will have to continue managing expanding matrices of inter-cell keys.

12.13. {PK11} Non-Composable Key Change

SPONSOR: Bob Blakley

REQT: Compromise of a principal's key (including long-term key) should not compromise that principal's future keys, as long as the attacker in possession of the compromised key does not change the key before the legitimate owner.

It is possible to devise key-change protocols which:

(a) Use the principal's existing key to validate the change, and

(b) Securely distribute a new key to the principal, but

(c) Do not encrypt the new key under the old key, or do anything else which would allow an attacker in possession of the original key to learn the new key.

These key-change protocols have a property which is called ''perfect forward secrecy'' in the academic literature. This property should be exhibited by the DCE key-change and password-change protocols.

12.14. {PK12a} Long-Term Protection of Audit Data

SPONSOR: Ellen McDermott

REQT: The ability to determine the originator, verify the integrity and if required, maintain the confidentiality of audit data over a long period of time (perhaps many years).

This requirement assumes that the message originator must sign all audit data submitted to the audit repository. Authorized reviewers of audit data must be able to verify signatures in a manner that proves the identity of message originators to a third party, at any time after the submission of audit data. The requirement is for ''long-term'' protection which implies that keys required for verifying signatures must be kept in an integrity-protected cache, at a minimum. If the same key is used to sign and seal audit data, then the key must be confidentiality protected as well to ensure only authorize people may review audit data.

12.15. {PK12b} Disposable Signing Keys

SPONSOR: Ellen McDermott

This is a specific instance of {PK2}, with modifications for explicit long-term protection. If only symmetric (secret key) technology is used for long-term protection of data, then the keys used to apply integrity protection must be retained for the life of the data, increasing the potential for key disclosure and unauthorized data modification.

Using public-key technology, however, it would be possible for the signature-generating key to be deleted once the signature has been applied (retaining only the signature-verification key). Re-certification for long-term storage (longer than the projected life-time of a given key) may be accomplished without knowledge of the original signature key.

Impact if requirement not satisfied: Keys used to apply signatures to data must be retained for the life of the data, increasing the possibility of undetected modification of audit data.

12.16. {PK12c} Protect Audit Log Against Non-Auditor Examination

SPONSOR: Ellen McDermott

The use of public key technology would allow multiple sources of audit data to write to a single audit repository, in such a way that the resulting audit trail cannot be read by anyone other than a designated auditor. Even audit sources would be unable to read their own data.

Impact if requirement not satisfied: Audit data assmebled from a variety of sources requires an active audit data collection agent (in addition to the sources of audit data) so that audit sources may use DCE session-oriented security services to protect data in transit.

12.17. {PK13} Virus Detection/Software Certification

SPONSOR: Gary Gaskell

REQT: DCE should allow the attachment of signatures to executable files to allow detection of unauthorized modification of those files.

12.18. {PK14} Software License Management

SPONSOR: Bob Blakley

12.19. {PK15} Public-Key Support for Delegation through Untrusted Intermediates

SPONSOR: Joe Pato

REQT: An ''on-line'' version of {PK8}. A target server receiving a delegated request should be able to verify the original client's request even if it does not trust any intermediate in the delegation chain to properly represent the client's intentions. Similarly, the client should be able to verify the target server's response even if it trusts none of the intermediates.

12.20. {PK16} Notary Services

SPONSOR: John Wray

REQT: A DCE service whose function is to apply signatures to documents. These signatures bind the document content to the identity of the principal who presented the document for signature, as well as to the date of the signature.

This service differs from {PK2} in that, instead of verifying only document content and origin, the service validates:

(a) Content.

(b) Principal presenting the document.

(c) Date presented.

A Notary service has its own keys which are used to apply signatures to documents (possibly using services provided in response to {PK4}). A notarized signature (as opposed to a self-generated signature, which {PK2} and {PK4} address) is one whose date cannot be falsified by the originator. This means that a notarized document need not be disseminated by the originator until proof of the document's existence and date is required.

Impact if requirement not satisfied: Notary functions will be provided by applications. If DCE does not provide rules for notary signature formats, key usage, etc., these application-provided services will be incompatible with one another, resulting in increased management complexity.

12.21. {PK17} Workflow Computing

SPONSOR: Joe Pato

12.22. {PK18} Interoperability with Other PK Environments

REQT: Interoperate securely with PK-based distributed computing environments including OSI, Netware, ONC+, etc.

12.23. {PK19a} Revocation of the Private Key Component

SPONSOR: Ellen McDermott

REQT: The ability of a trusted authority to notify all parties within the computing environment (cell or multiple cells, if appropriate) that specified private keys are no longer valid as of a stated time.

A means for recovering from compromised keys must be available, before introducing public key technology into DCE. Currently in DCE when a key is compromised it is changed at the Security Server at soon as the compromise is recognized. Since the Security Server is an on-line trusted server which is involved in the initiation of any new sessions, recovery from the compromised key starts when sessions based on the compromised key time out. Public key based systems do not require an on-line trusted server thus making the current recovery model inappropriate for services based on public key technology.

Impact if requirement not satisfied: Compromised keys could be used indefinitely, in perpetrating masquerading/impersonation attacks and breaching confidentiality.

12.24. {PK19b} Revocation of the Identity Binding

SPONSOR: Ellen McDermott

REQT: The ability of a trusted authority to notify all parties within the computing environment (cell or multiple cells, if appropriate) that specified couplings/affiliations of identities and organizations and/or roles are no longer valid as of a stated time.

A means for notifying all parties within a cell (and communicating cells) that the association between a key, username/uuid and an organization is no longer valid must be available before introducing public key technology into DCE. Currently, the Security Server vouches for users' affiliations in a cell by allowing users to start new sessions under their present identity/affiliation. If a user's affiliation has changed, the Security Server will refuse to grant TGTs which are needed to create new sessions under the current identity coupling. Public key based systems do not require an on-line trusted server thus making the current means of revoking affiliations inappropriate for services based on public key technology.

Impact if requirement not satisfied: Invalid identity bindings could be used indefinitely to perpetrate fraud and breach confidentiality.

12.25. {PK20} Trust Hierarchy

SPONSOR: Ellen McDermott

REQT: A well defined structure of authority that can be recognized by users and applications.

Public key based services must be able to use the current hierarchical trust model. Public-key services are not obliged to incur the overhead of pairwise key exchanges between cell administrators. Any other trust model used by DCE must be published.

Impact if requirement not satisfied: Multiple trust models could develop between cells. These models may have flaws or used in conjunction with other trust models may introduce flaws. Flaws in trust models result in: inappropriate or misplaced trust, and lack of containment in cases of compromise.

12.26. {PK21} Performance Profile

SPONSOR: Ellen McDermott

REQT: The performance of base services using public key technology must be at least as fast as base services using the current implementation of DES.

The current DCE security services are based on the Data Encryption Standard (DES), which is a notably faster mechanism (~100 times faster) for confidentiality than well known public key algorithms, such as RSA and ElGamal. Public key based services must be able to provide confidentiality and integrity with the same or better efficiency as that found in DCE 1.x.

Impact if requirement not satisfied: Implementations of public key based services that are significantly slower than the current version of DCE will result in commercially unacceptable products.

12.27. {PK22} X.509 Certificate Format

SPONSOR: Ellen McDermott

REQT: Public key certificates used within DCE will be X.509 compliant.

The majority of currently available public key products use X.509 compliant certificates. X.509 is well defined and is the only widely accepted public key certificate standard. As DCE integrates public key services they must be X.509 compliant to allow interoperability between non-DCE public key services and DCE public key services at the application level.

X.509 is used by both PEM and X.500 (and the latter is used in X.400), so this item can be seen as an enabler for the extension of DCE to support of those services. An X.509 capability in DCE, even without PEM or X.400 support, may be useful in that vendors could provide the added value in a manner that is interoperable with other implementations.

DCE should permit acquisition of X.509 certificates by DCE users. This may, but does not necessarily, include the ability to use X.509 for user authentication.

Similarly, DCE should permit servers to be registered with X.500 names and should permit servers to get X.509 certificates.

Impact if requirement not satisfied: Implementations of public key services using non-compliant X.509 certificates will not be able to interoperate with services using X.509 certificates or other non-compliant certificate formats.

Without this support, users and servers will have to log on twice to get strong authentication to X.500 directories.

12.28. {PK23} Interoperability with Recognized Standards

SPONSOR: Ellen McDermott

REQT: Public key services must be interoperable with relevant standards in service area.

For example, if the service area is secure electronic mail, the public key electronic mail system must be compliant with at least one of the following mail systems: X.400, PEM, PGP. This implies compatible certificate and revocation list formats and processing procedures. OSF will have to decide on the minimal set of standards a service area must meet in order to be accepted as a part of an OSF offering.

Impact if requirement not satisfied: Implementations of multiple versions of service based on public key technology will emerge making interoperability improbable and possibly weakening the level of security of the system. Implementations of public key services that do not meet widely accepted standards have had less exposure, which means that they may contain weakness that could have been found due to the rigorous scrutiny of the standards process.

12.29. {PK24} Support for Multiple Cryptographic Algorithms

SPONSOR: Ellen McDermott

REQT: Services based on public key cryptography must be algorithm independent.

Services based on public key cryptography must be able to be implemented using a set of public key cryptographic algorithms. The goal of this requirement is to allow the consumer of the product to choose an algorithm(s) based on their needs and their government's restrictions.

Impact if requirement not satisfied: Products will be built that cannot be exported out of the country where they are developed and/or imported to consuming countries. Multiple implementations of the same service may result from customers' demand for services with different algorithms, which will lead to complexity in key management.

12.30. {PK25} Standardized Formats for Signed Data

SPONSOR: Ellen McDermott

REQT: Signed data will adhere to an agreed upon format.

OSF must endorse a format for signed data. This will enable DCE signature services to be interoperable in verifying signed data.

Impact if requirement not satisfied: Multiple implementations of signature services using different formats may result, making verification of signatures and data integrity difficult at best.

12.31. {PK27} Role/Identity/Key Binding

SPONSOR: Ellen McDermott

REQT: Before public key services are integrated into DCE a statement of role/identity/key binding and alignment with secret key services in DCE must be made.

OSF must make an explicit statement of role/identity/key binding and alignment for both public key and secret key services. This statement should highlight the relationship between services of different cryptographic bases, and their binding definitions.

Impact if requirement not satisfied: DCE could end up with redundant or insufficient service infrastructure in an attempt to enable interoperability between cells using different bindings for different cryptographic systems. This scenario would result in increased complexity in cell management.

12.32. {PK28} No Mandatory Dependence on Public Key

SPONSOR: Ellen McDermott

REQT: None of the core DCE services may be provided solely by a public key service.

DCE security services have been developed using DES, a widely available secret key system, which is accepted in many countries. Public key cryptography is not efficient for the core security services found in DCE. Encryption (for confidentiality) using public key systems is on the order of 100 times slower than DES. The overhead of signatures for data integrity can be overkill in many circumstances. Efficiency of authentication systems based on public key vs secret key systems is arguable depending on the desired qualities.

Impact if requirement not satisfied: Versions of DCE based solely public key security services will not gain wide acceptance in all of countries in which DCE is present.

12.33. {PK29} Definition of Key Escrow in DCE

SPONSOR: Ellen McDermott

REQT: A key escrow scheme must be defined and accepted as the OSF escrow standard.

Public key products, such as secure mail and secure electronic documents, have signed and/or encrypted data that can have a long life. A means of validating signatures or decrypting messages at a period of time well into the future must exist. Therefore, a way of securely storing the keys required for validation and decryption must be available.

Impact if requirement not satisfied: Multiple, possibly non-secure, key storage sites may be created, which will lead to increased management burden and possible security holes.

12.34. {PK30} Signature API

SPONSOR: John Wray

REQT: An API enabling applications to create and verify unforgeable digital signatures.

This requirement includes the provision of a trusted public-key storage mechanism, which the signature verification API can use to retrieve verification keys. Additionally, explicit application-provided key management should be supported.

The API should employ a standard format for signatures, and support common mechanisms (e.g., the PKCS series of standards).

Impact if requirement not satisfied: Applications requiring public key technology will have to implement the keystore and user private-key acquisition in incompatible ways. Public-key applications will also have to implement and ship include their own cryptographic code, resulting in export issues for these applications.

12.35. {PK31} Key Establishment

SPONSOR: Bob Blakley

REQT: It should be possible to distribute long-term keys to DCE servers over the network in a secure fashion without requiring the receiving server to have a secret key in its posession.

Impact if requirement not satisfied: It will continue to be necessary to have a trusted administrator physically present at each server's machine to install that server's initial long-term key.

12.36. {PK32} Multiple Signatures

SPONSOR: Gary Gaskell

12.37. {PK33} Smartcard Support

SPONSOR: Roger Merckling

13. LOGOFF

13.1. {F1} DCE Logoff

SPONSOR: Bob Blakley

REQT: DCE should implement a ''logoff'' function which will invalidate the logged-off user's credentials and generate and audit log entry.

13.2. {F2} Detection of Active Connection

SPONSOR: Chii-Ren Tsai

REQT: It should be possible to configure the system so that any given principal must keep an active connection with a watchdog process and that process would attempt to cause the user to be terminated if the active connection is lost.

> *August 1994 Note: This requirement is intended to provide automated termination of idle client-server associations established using DCE credentials.*

13.3. {F3} Idle Session Timeout

SPONSOR: Chii-Ren Tsai

REQT: DCE login sessions should have idle session timeouts. Timeout of the session should cause a complete termination of client security data.

> *August 1994 Note: This requirement is intended to provide automated invalidation of the DCE credentials associated with idle terminal sessions on the user's DCE client machine.*

13.4. {F4} Complete Termination on Disconnect

SPONSOR: Chii-Ren Tsai

REQT: Termination of a login session should cause all security data on a client machine to be destroyed. No data should be retained that would allow a serial user of the machine to impersonate a user that had previously used the machine.

> *August 1994 Note: This requirement is intended to provide automated invalidation of the DCE credentials associated with idle terminal sessions on the user's DCE client machine.*

14. PROPOSED REQUIREMENTS REMOVED FROM LIST

The following requirements were initially proposed by at least one individual from the working group, but were since unanimously deleted for the reasons given.

14.1. {A6} Authority Identity (DELETED)

REQT: Provide functions that permit authorization to be determined based on the identity of the user's authority or authorities (trusted registry) instead of the user's identity.

A vote for this requirement is a vote for a scheme whereby the server has the option of looking only at the authority's identity, plus the attributes that the authority claims for the client, in order to evaluate access. It is

similar to {A5} in that the client's individual identity is not relevant to the access control decision, but unlike {A5} does not require the client's identity to be hidden from the server.

The group decided not to vote on this as a separate item, as it is viewed to be subsumed by item {R2}. That is, DCE already provides the ability for an ACL to list ''any principal in a cell'', and if the only types of authorities supported are those for the entire cell, then trusting any principal in a cell is semantically equivalent to trusting the authority for the cell. Therefore, the only new requirement this imposes is a way of deciding who the trusted authority is for the cell.

14.2. {E6} Interoperability with Selected Non-DCE Systems (DELETED)

REQT: Consider interoperability with popular non-DCE environments such as PC LANs.

This proposed requirement was considered out of scope for DCE.

14.3. {P1} Performance (DELETED)

REQT: Authentication and authorization should not significantly degrade access. Performance degradation due to communication security should be minimized.

This item was deleted because good performance is a generally accepted principal for any software engineering effort and need not be spelled out explicitly.

14.4. {P2} Robustness to Prevent Denial-of-Service (DELETED)

REQT: Enabling of security should not result in a significantly increased risk of denial of service, either due to bugs, random environmental events, or malicious attacks.

This item was deleted for reasons similar to {P1}.

14.5. {P3} Interoperation with Operating System Security (DELETED)

This item is subsumed in {P10a}.

14.6. {P11} Interoperability Among All DCE Variants (DELETED)

Deleted because this requirement is not unique to security.

14.7. {P12} Extensibility Mechanisms for Future DCE Versions (DELETED)

Deleted because this requirement is not unique to security, and in any case it is good software engineering.

14.8. {P14} Standards (DELETED)

REQT: Accommodate security standards, or draft standards, where appropriate. Candidates include those from ISO, ECMA, IETF and POSIX.

Deleted because this requirement is not unique to security and is a given OSF goal.

14.9. {P7} B1 Portability (DELETED)

14.10. {P8} C2 Portability (DELETED)

14.11. {P9} ITSEC Evaluation (DELETED)

> *April 1994 Note: As agreed at the April meeting, these items have all been withdrawn. If functional requirements which move the DCE code toward these goals are formulated, they will be accepted.*

14.12. {T14} Internationalization (DELETED)

SPONSOR: Chii-Ren Tsai

REQT: Internationalization of audit records should be supported, because event specific information may contain different languages originated from geographically dispersed regions within Citicorp.

> *August 1994 Note: We agreed to remove this requirement because it is viewed as outside the mission of the Security SIG, and within the mission of the I18N SIG, to assert this requirement.*

14.13. {T17} Supress Sensitive Data in Audit Logs (DELETED)

SPONSOR: Chii-Ren Tsai

REQT: It should be possible to designate data ''sensitive'' so that the audit subsystem will not write it into the audit log.

August 1994 Note: This requirement was withdrawn with the permission of the sponsor.

14.14. {N4} Site Definable Password Complexity (DELETED)

REQT: The password complexity policies supported by `rgy_edit` and `dce_login` must be extended. The extensions should allow for the following two additional policy choices:

(a) Complex passwords that require a mix of alpha, numeric and special characters.

(b) A site definable policy.

This policy would require securely branching to site provided code that defines specific local password complexity rules. Since some sites will not want to reveal those rules, the ability to securely execute their own password checking code is a must.

Impact if requirement not satisfied: DCE will continue to allow trivial passwords that do not meet the standards and policies of many of the organizations that want to use DCE.

14.15. {PK26a} Statement of OSF's Role with Regard to Licensing Issues (DELETED)

SPONSOR: Ellen McDermott

REQT: OSF must state its role with regard to cryptographic algorithm licensing.

OSF must decide whether it will: negotiate with the patent holders of public key algorithms and broker the use of these algorithms for use with DCE; or have vendors negotiate their own licensing agreements.

Impact if requirement not satisfied: Confusion over the proper way to proceed when building public key services for DCE will result, causing a delay in the release of products.

14.16. {PK26b} Statement of OSF's Role with Regard to Export Issues (DELETED)

SPONSOR: Ellen McDermott

REQT: OSF must state its role with regard to exporting cryptographic material outside of the US.

OSF must decide whether it will: submit applications for export on behalf of vendors adhering to approved implementation of a public key service; or have vendors submit applications directly with the State Department.

Impact if requirement not satisfied: Confusion over the proper way to proceed when building and shipping public key services for DCE will result, causing a delay in release of products.

14.17. {PK26c} Statement of OSF's Role with Regard to Certifying Authorities (DELETED)

SPONSOR: Ellen McDermott

REQT: OSF must state its role with regard to setting up and participating in a certifying authorities' hierarchy.

OSF must decide whether it will: build, administer and participate in a hierarchy of certifying authorities; or have vendors and end-users find other means of provision of such a hierarchy.

Impact if requirement not satisfied: If OSF does not provide an approved certification authority hierarchy, disjoint and untrustworthy certification authority hierarchies may result.

15. REQUIREMENTS SATISFIED BY SECURITY FUNCTIONALITY OF DCE 1.1

The following requirements were voted on for RFC 8.0 and are considered by a consensus of the working group to have been satisfied by the security mechanisms provided in DCE 1.1.

15.1. {A1} Authorization Based on Delegation (SATISFIED IN DCE 1.1)

REQT: Given a delegation model that permits the server to know which principals were involved in a delegated request, build an authorization model that permits the server to enforce an authorization decision based on those identities. Extend the authorization mechanism to allow an administrator (e.g., the ACL writer) to specify permitted delegates in a practical way.

The idea is fundamentally to permit the delegates (when or if delegation is supported in DCE) to be named in the ACL.

NOTE: Satisfied by RFC 3.0 work in DCE 1.1.

15.2. {A8} Other Security Attributes (SATISFIED IN DCE 1.1)

REQT: Through an extensible mechanism, permit the server to obtain from the client verifiable security attributes of the user other than principal or group memberships.

This requirement is a prerequisite to {A7}. Examples of new attributes that might be supported are clearance and role.

NOTE: Satisfied by RFC 3.0, RFC 6.0 work in DCE 1.1.

15.3. {D1} Full Delegation (SATISFIED IN DCE 1.1)

REQT: Permit a client to explicitly specify that its current access rights should be transferred to a server, so that the server can access remote resources using the client's rights. Delegation should be available through all authentication mechanisms and APIs, including RPC. It must be possible for the user to control whether or not delegation takes place.

In this scheme, the delegate is permitted to fully impersonate the user (for some period of time), and this delegation should be transferable through RPC and any other authentication protocol. any delegation scheme, there is a strong requirement that the user should be able to specify that delegation must not happen.

NOTE: Satisfied by RFC 3.0 work in DCE 1.1.

15.4. {D2} Delegation with Restrictions (SATISFIED IN DCE 1.1)

REQT: Where delegation is specified by a client, permit the client to restrict delegated rights in some useful fashion to be a subset of those rights currently possessed by the client.

This requirement does not necessarily imply extensions to the delegation mechanism. An acceptable implementation is one where the client can restrict its current rights (as in DCE today), and then delegate all of its current rights.

NOTE: Satisfied by RFC 3.0 work in DCE 1.1.

15.5. {E2} Security API Through Protocols Other Than RPC (SATISFIED IN DCE 1.1)

REQT: Applications that choose not to use RPC for communications should have the ability to make use of all the DCE authentication and communications security mechanisms through a portable API.

This is the GSS API requirement.

NOTE: Satisfied by RFC 5.0 work in DCE 1.1.

15.6. {P10a} OS Integration (SATISFIED IN DCE 1.1)

REQT: Integrate DCE with certification/registration environments of other operating systems. This includes a generic ability to centralize registry databases for multiple systems, to permit flexibility in storing OS-specific user attributes in the registry, and to permit adjusting the frequency of registry updates based on local system needs.

This is an extension to DCE to make it more portable to environments supporting other than Unix-based hosts.

NOTE: Satisfied by RFC 6.0 work in DCE 1.1.

15.7. {R2} Trust Relationships (SATISFIED IN DCE 1.1)

REQT: Implement some kind of trust relationship between cells so that it is not necessary for all potentially interacting cells to directly communicate in advance. There should be at least one default implied relationship (e.g., a hierarchy based on names) throughout DCE to minimize the number of explicit trust relationships that a manager of a new cell must specify. In addition, a security manager should have the ability to establish additional explicit trust relationships within a cell that override or augment the default relationships.

This requirement was created mostly for management ease in an environment of many cells. It is unlikely that a security manager of a cell can make a rational decision as to which authorities to trust for all possible remote cells with which the users and servers in his cell need to communicate. It is unlikely that security manager will even know, in advance, what those cells are. Therefore, some well-defined (and of course, secure) default relationship needs to be consistently supported throughout the environment.

On the other hand, there is also the requirement that security managers cognizant of trust relationships be able to override the default where they have direct pairwise agreements with each other.

NOTE: Satisfied by RFC 7.0 work in DCE 1.1.

15.8. {T1} Local Auditing (SATISFIED IN DCE 1.1)

REQT: Record security-relevant DCE events on the local system.

NOTE: Satisfied by RFC 28.0, RFC 29.0 work in DCE 1.1.

15.9. {X2} Limit Password Dictionary Attacks (SATISFIED IN DCE 1.1)

REQT: A site or cell should have the option to configure a mechanism that prevents dictionary attacks on poorly chosen passwords by wiretappers or nodes masquerading as legitimate clients. A tool to avoid picking bad passwords is not by itself an adequate solution.

The Kerberos protocol used in DCE returns recognizable messages (e.g., the TGT) encrypted only by the user's password, which is subject to capture and analysis. Satisfying this requirement requires a change to the Kerberos protocol or use of an intermediary to communicate with the Kerberos server that is trusted not to guess passwords. Public key techniques are useful in simplifying the solution to this problem.

NOTE: Satisfied by RFC 26.0 work in DCE 1.1.

15.10. {X3} GSSAPI Changes for Exportability (SATISFIED IN DCE 1.1)

SPONSOR: Ivan Milman

REQT: The DCE 1.1 version of GSSAPI doesn't come in exportable and non-exportable versions (i.e., `gss_seal` is not split out into two routines: one that works, and one that doesn't). OSF should provide exportable and non-exportable versions of `gss_seal`.

April 1994 Clarification: The requirement here is to provide a version of the `gss_seal` *verb which just turns cleartext into cleartext, so that vendors can produce an exportable package in which the API is present but not functional.*

Examination of the DCE 1.1 code indicates that this has been done, so the requirement has been withdrawn.

15.11. {T3} Application Access to Audit Functions

REQT: Extend the auditing functionality introduced in DCE 1.1 to allow applications to use the audit APIs and associated mechanisms. The extensions should allow application audit data to be sent to the same centralized audit repository used internally by DCE. Additionally, any audit based alarm mechanism or reporting/reduction tools that may be provided by DCE should be usable with application based auditing.

Impact if requirement not satisfied: Applications will have to develop their own auditing mechanism. This implies that a security administrator will have to look at many different audit files of possibly different formats to get an accurate picture of what is happening in the system. The potential for different audit record formats also means that it will be very difficult to develop a single audit reduction/analysis tool. In summary the lack of a centralized audit for both DCE and applications will result in an increase in the complexity of the security administrator's job.

16. OSF MEMBER VOTES

This section presents the votes received from OSF member companies on the requirements described in this document. The votes are arranged by item category.

Key to non-obvious company abbreviations:

Bel Bellcore Cit Citibank Hug Hughes Phl Phillips Trc Transarc SWP Sweden Post DST Distributed Systems Technology Center, Queensland University of Technology Shl Shell Bar Barclay's Bank

	Bel	IBM	Cit	EDS	Hug	Phl	Trc	SWP	DST	HP	Shl	Bar
A2	8	0	0	0	10	5	0	0	8	0	0	9
A3	0	0	0	0	0	0	0	0	2	0	0	0
A4	7	4	5	0	10	10	30	0	12	10	0	20
A5	0	0	0	0	0	0	0	0	0	0	7	0
A7	0	0	5	0	0	10	0	0	2	0	7	0
A9	-2	0	0	0	0	0	0	0	2	0	7	0
D3	0	4	0	0	0	0	10	0	3	30	7	0
D3b	5	14	0	0	0	0	0	0	3	10	0	0
D4	0	4	0	0	0	5	0	0	0	0	4	0
D5	0	4	0	0	0	5	0	0	0	0	4	0
E1a	5	14	0	15	0	10	0	30	8	30	10	0
E1b	5	14	0	15	0	10	0	30	0	30	10	0
E3	15	0	0	0	0	20	0	0	10	10	7	0
E4	0	0	0	0	15	4	30	0	3	20	4	0
E5	5	5	0	0	15	5	30	30	9	0	7	20
E7	0	0	0	0	15	5	0	0	8	0	7	0
E8a	0	5	0	0	7	15	10	0	4	0	0	0
E8b	0	5	0	0	0	0	10	0	0	0	0	15
E8c	0	5	0	0	0	10	0	0	0	0	0	15
E8d	0	0	0	0	10	0	10	0	0	0	0	20
E8e	0	0	0	0	0	0	1	0	0	0	0	0
E9	20	15	20	10	7	35	1	30	7	30	7	9
I1	0	5	0	10	0	10	5	0	7	0	7	9
I2	5	0	0	0	0	5	0	15	-2	0	4	0
I3	5	4	5	0	15	10	0	0	12	25	10	0
I4	10	0	0	10	0	5	0	0	-1	0	7	15
I5	0	4	0	0	0	0	0	0	0	-1	4	0
P4	0	0	0	0	0	0	0	0	0	0	0	0
P5	0	0	0	0	0	0	0	0	0	0	0	0
P6	0	0	0	0	15	0	0	0	0	0	0	0
P10b	5	2	0	0	0	0	30	0	3	30	7	0
P14	0	15	0	0	5	0	0	0	1	10	4	0
R1	0	-8	0	0	0	0	0	0	0	0	0	0
R7	0	0	0	0	0	0	0	0	-6	0	0	0
R8	10	5	0	0	0	5	30	0	2	0	4	0
R9	5	4	0	0	0	0	0	0	-3	0	7	0
R10	0	0	0	0	0	0	1	0	0	0	0	0
R11	10	4	2	0	0	5	1	0	3	-1	7	0
T2	5	5	5	0	0	5	30	0	2	0	0	20

T4a	0	0	5	0	7	0	0	0	0	0	0	0
T4b	0	0	5	0	7	0	0	0	0	0	0	0
T5	0	0	5	0	0	0	10	0	5	0	0	0
T6	0	0	2	0	0	0	0	0	8	0	0	0
T7	0	0	2	0	0	0	0	0	0	0	0	0
T8	0	0	5	0	0	0	0	0	0	0	0	0
T9	0	0	5	0	0	0	0	0	0	0	0	0
T10	0	0	5	0	0	0	0	0	2	0	0	0
T11	0	0	5	0	10	0	0	0	5	0	0	0
T12	0	0	2	0	0	0	0	0	0	25	0	0
T13	0	0	5	0	0	0	0	0	0	0	0	0
T14	10	-8	20	0	0	0	0	0	9	0	0	0
T16	0	0	2	0	0	0	0	0	0	0	0	0
T18	0	-8	2	0	0	0	0	0	0	0	0	0
T19	0	0	2	0	0	0	0	0	0	0	0	0
X1	0	0	0	0	10	0	0	0	0	-1	-4	0
X7	0	0	20	0	0	0	0	0	0	0	-4	0
N1	0	5	5	0	8	30	0	0	6	0	-4	20
N2	5	4	0	10	0	5	0	0	0	0	4	20
N3	5	4	0	10	0	5	0	0	0	0	4	20
N5	5	0	0	15	5	5	0	0	0	0	4	20
N6	0	0	0	0	0	0	0	0	4	0	4	0
N7	0	0	0	0	15	0	0	0	9	0	7	0
N8	0	5	2	0	15	0	0	0	4	0	7	0
S1	0	14	0	0	0	0	0	0	0	0	0	0
F1	15	0	0	0	0	0	0	0	0	0	0	0
F2	0	0	0	0	0	0	0	0	0	0	0	0
F3	0	0	0	0	0	0	0	0	0	0	0	0
F4	0	0	0	0	0	0	0	0	0	0	0	0
PK1	25	14	20	60	10	0	0	0	9	0	10	15
PK2	0	0	5	0	5	15	0	0	4	0	4	0
PK3	0	0	5	0	0	0	0	0	5	0	4	0
PK4	15	14	20	40	5	5	0	0	4	0	4	0
PK5	0	0	0	0	0	5	0	0	2	0	4	0
PK6	0	0	0	0	0	10	0	0	5	0	4	0
PK7	0	0	0	0	0	0	0	0	2	0	0	0
PK8	0	0	5	0	0	5	0	0	0	0	0	0
PK9	10	0	0	40	0	10	0	0	7	0	10	0
PK9a	0	0	0	0	0	0	0	0	0	0	0	0
PK9b	5	0	0	0	0	0	0	0	0	0	10	0
PK10	0	0	0	0	15	10	0	0	2	0	7	0
PK11	0	0	0	0	0	0	0	0	2	25	0	0
PK12a	0	0	2	0	9	0	0	0	4	0	4	0
PK12b	0	0	0	0	0	0	0	0	2	0	0	0
PK12c	0	0	0	0	0	0	0	0	2	0	4	0
PK13	0	-8	0	0	0	5	0	0	8	0	0	0
PK14	0	-8	0	0	0	0	0	0	0	0	0	0
PK15	0	0	0	0	0	5	0	0	5	0	0	0
PK16	15	5	20	5	0	5	0	30	1	0	7	0
PK17	0	0	0	0	0	0	0	0	0	0	0	0

PK18	0	4	2	0	0	10	0	9	1	0	10	0
PK19a	0	4	0	0	12	0	0	0	5	0	0	15
PK19b	0	0	0	0	0	0	0	0	1	0	0	15
PK20	0	0	2	0	10	5	0	30	-3	0	8	0
PK21	0	0	0	0	0	0	0	0	0	0	0	0
PK22	10	14	5	40	0	5	0	30	10	0	10	9
PK23	0	0	0	0	10	0	0	0	3	0	0	0
PK24	0	0	0	0	15	5	30	0	9	0	10	0
PK25	10	0	0	0	0	0	0	0	2	0	7	0
PK27	0	0	0	0	0	0	0	0	2	0	10	0
PK28	0	14	0	0	0	0	30	0	4	0	0	0
PK29	0	-8	0	0	0	0	0	30	6	0	4	0
PK30	10	14	5	0	0	0	10	0	4	0	4	9
PK31	10	5	2	0	0	0	0	15	0	0	4	9
PK32	0	0	0	0	0	0	0	0	4	0	4	0
PK33	10	0	5	0	0	0	0	0	4	0	0	0
Total	300	309	239	280	292	339	309	279	297	315	334	304
Max	25	15	20	60	15	35	30	30	12	30	10	20

NOTE: The Phillips vote was illegal (too many total votes cast); I have allowed it since the error was caused by use of an out-of-date ballot which included more items than the current ballot.

17. VOTE SUMMARY

This section presents the voting summary statistics, organized by item category.

Item	Total	Mean	S.D.	Rank
A2	40	3.33	4.09	30
A3	2	0.20	0.60	83
A4	108	9.82	8.39	7
A5	7	0.70	2.10	76
A7	24	2.40	3.47	48
A9	-6	-0.60	5.24	101
D3	54	4.91	8.58	20
D3b	32	3.20	4.77	39
D4	13	1.30	2.00	63
D5	13	1.30	2.00	63
E1a	122	11.09	10.26	5
E1b	114	10.36	10.72	6
E3	62	5.64	6.91	17
E4	76	6.91	9.73	13
E5	126	11.45	10.39	4
E7	35	3.50	4.90	34
E8a	41	4.56	5.04	29
E8b	30	3.00	5.10	43
E8c	30	3.00	5.10	43
E8d	40	4.00	6.63	30
E8e	1	0.11	0.31	89
E9	191	15.92	10.56	1
I1	53	4.82	3.97	22
I2	27	2.70	4.71	45

I3	86	7.82	7.31	10
I4	46	4.18	5.29	27
I5	-2	-0.18	3.46	99
P4	0	0.00	0.00	91
P5	0	0.00	0.00	91
P6	15	1.50	4.50	58
P10b	77	7.00	11.07	11
P14	35	3.18	4.82	34
R1	-8	-0.89	2.51	104
R7	-6	-0.67	1.89	101
R8	56	5.60	8.70	18
R9	13	1.30	2.87	63
R10	1	0.11	0.31	89
R11	22	2.00	4.86	49
T2	72	7.20	9.41	14
T4a	12	1.33	2.54	67
T4b	12	1.33	2.54	67
T5	20	2.22	3.42	50
T6	10	1.11	2.51	71
T7	2	0.22	0.63	83
T8	5	0.56	1.57	79
T9	5	0.56	1.57	79
T10	7	0.78	1.62	76
T11	20	2.22	3.42	50
T12	27	2.70	7.46	45
T13	5	0.56	1.57	79
T14	31	3.44	7.73	41
T16	2	0.22	0.63	83
T18	-6	-0.67	2.67	101
T19	2	0.22	0.63	83
X1	-4	-0.36	4.42	100
X7	16	1.60	6.25	56
N1	70	6.36	9.63	15
N2	48	4.36	5.82	24
N3	48	4.36	5.82	24
N5	54	4.91	6.40	20
N6	8	0.80	1.60	74
N7	31	3.10	5.09	41
N8	33	3.30	4.58	37
S1	14	1.56	4.40	61
F1	15	1.67	4.71	58
F2	0	0.00	0.00	91
F3	0	0.00	0.00	91
F4	0	0.00	0.00	91
PK1	163	14.82	16.27	2
PK2	33	3.30	4.45	37
PK3	14	1.40	2.15	61

PK4	107	10.70	11.65	8
PK5	11	1.10	1.81	70
PK6	19	1.90	3.24	52
PK7	2	0.22	0.63	83
PK8	10	1.11	2.08	71
PK9	77	7.70	11.65	11
PK9a	0	0.00	0.00	91
PK9b	15	1.50	3.20	58
PK10	34	3.40	5.12	36
PK11	27	2.70	7.46	45
PK12a	19	1.90	2.84	52
PK12b	2	0.22	0.63	83
PK12c	6	0.60	1.28	78
PK13	5	0.56	4.09	79
PK14	-8	-0.89	2.51	104
PK15	10	1.11	2.08	71
PK16	88	8.80	9.36	9
PK17	0	0.00	0.00	91
PK18	36	3.60	4.15	32
PK19a	36	3.27	5.15	32
PK19b	16	1.60	4.48	56
PK20	52	5.20	9.12	23
PK21	0	0.00	0.00	91
PK22	133	12.09	11.75	3
PK23	13	1.30	3.03	63
PK24	69	6.90	9.25	16
PK25	19	1.90	3.42	52
PK27	12	1.20	2.99	67
PK28	48	5.33	9.75	24
PK29	32	3.20	9.56	39
PK30	56	5.09	4.76	18
PK31	45	4.09	4.93	28
PK32	8	0.80	1.60	74
PK33	19	2.11	3.35	52

18. ITEM RANKING

This section presents the items listed in rank order. The top 16 items are identified by title for convenience.

Item	Total	Rank	Title
E9	191	1	Single Signon
PK1	163	2	Public-Key Authentication
PK22	133	3	X.509 Certificate Format
E5	126	4	Application Security Services
E1a	122	5	Secure Remote Login
E1b	114	6	Secure File Transfer
A4	108	7	Authorization Based onLocation
PK4	107	8	Digital Signatures
PK16	88	9	Notary Services
I3	86	10	Role-based Authorization
PK9	77	11	Secure Electronic Mail
P10b	77	11	Kerberos Version 5 Interoperation
E4	76	13	More Options for Communications Security
T2	72	14	Remote Auditing
N1	70	15	Alternate Authentication Mechanism Support
PK24	69	16	Support for Multiple Crypto Algorithms

E3	62	17
R8	56	18
PK30	56	18
N5	54	20
D3	54	20
I1	53	22
PK20	52	23
PK28	48	24
N2	48	24
N3	48	24
I4	46	27
PK31	45	28
E8a	41	29
E8d	40	30
A2	40	30
PK19a	36	32
PK18	36	32
E7	35	34
P14	35	34
PK10	34	36
N8	33	37
PK2	33	37
PK29	32	39
D3b	32	39
N7	31	41
T14	31	41
E8b	30	43
E8c	30	43
T12	27	45
PK11	27	45
I2	27	45
A7	24	48
R11	22	49
T5	20	50
T11	20	50
PK12a	19	52
PK33	19	52
PK25	19	52
PK6	19	52
X7	16	56
PK19b	16	56
PK9b	15	58
F1	15	58
P6	15	58
S1	14	61
PK3	14	61
PK23	13	63
R9	13	63
D5	13	63
D4	13	63
T4b	12	67
T4a	12	67
PK27	12	67
PK5	11	70
PK15	10	71

PK8	10	71
T6	10	71
PK32	8	74
N6	8	74
T10	7	76
A5	7	76
PK12c	6	78
PK13	5	79
T13	5	79
T9	5	79
T8	5	79
PK7	2	83
T16	2	83
A3	2	83
T7	2	83
T19	2	83
PK12b	2	83
R10	1	89
E8e	1	89
PK21	0	91
P4	0	91
P5	0	91
F4	0	91
PK17	0	91
F2	0	91
F3	0	91
PK9a	0	91
I5	-2	99
X1	-4	100
R7	-6	101
A9	-6	101
T18	-6	101
PK14	-8	104
R1	-8	104

AUTHOR'S ADDRESS

Bob Blakley
IBM
11400 Burnet Road, Mail Stop 9356
Austin, TX 78750
USA

Internet email: blakley@vnet.ibm.com
Telephone: +1-512-838-8133