Open Software Foundation
Request For Comments: 91.0
January 1996

M. Burati (Hewlett Packard)
J. Pato (Hewlett Packard)

USER-TO-USER AUTHENTICATION — FUNCTIONAL SPECIFICATION

1. INTRODUCTION

The DCE 1.2.2 user-to-user authentication feature (see [RFC 63.1]) addresses the following requirement:[1]

> The goal of this requirement is to enable applications that do not have access to a principal's long term key to be the receivers of a protected RPC. Applications that are clients 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 the concept of ''server'' is associated with a 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. If for no other reason, the server is not normally associated with a human user, but rather with a pseudo-user corresponding to the system service.

This model fits well with the type of core DCE services distributed by the OSF today. This does 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 what is logically the application's ''client'' code to play the role of a ''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 the user to make their long term key available (either in a keytab file {forcing the key to be stored in the clear on the local machine} — or via an application provided key acquisition function {requiring that the user type their key to an arbitrary application}); 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.

Proposal: It should be possible to provide server applications 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 little more narrow than the way the requirement is currently worded — it would require that authentication be possible if the login context is 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 to be accomplished 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.

---

1. DCE 1.2 TPC item S7, referring to OSF-RFC 8.1 item D3.

2. TARGET

The target audience of this functionality is those servers (application or otherwise) that have a need to receive authenticated RPCs, but cannot for some reason have access to the server principal's long term key.

3. GOALS AND NON-GOALS

(a) It is a goal of this work to make use of the Kerberos v5 User-To-User mechanisms (`AP_OPTS_USE_SESSION_SKEY`) already existing in the base source code.

(b) It is a goal of this work to allow servers to register either using the existing KRB5 server-key ticket protocol, or the User-To-User protocol, but not both simultaneously.

(c) It is a goal/prerequisite to get the KRB5 based User-To-User protocol working with the DCE source base. It is not a goal to merge the Kerberos v5 base forward to a newer beta level. Bug fixes may be taken from a newer beta level on an as-needed basis.

(d) It is not a goal of this work to allow current DCE1.0.x/DCE1.1 based clients to make authenticated RPCs to servers registering their identity with this new functionality.

4. TERMINOLOGY

(a) **KDC** — Kerberos Distribution Center.

(b) **TKT** — Kerberos based Ticket.

(c) **keyproc** — Callback function provided by a server with a Kerberos specified signature, that retrieves the key to be used when decrypting tickets.

5. REQUIREMENTS

(a) This functionality must not break the existing DCE Auth RPC protocols.

(b) The default server authentication protocol for DCE core services must not change as a result of this work.

6. FUNCTIONAL DEFINITION

Servers without access to their long term key (e.g., keytable) will be able to register their identity based on a DCE login context, with the new RPC API routine `rpc_server_register_auth_ident()`.

New clients will be using a `libdce` that will detect when a server is using this new protocol. The client side runtime will receive the server's TGT and will pass this STGT along on the TKT request, using `AP_OPTS_USE_SESSION_SKEY` option. Then:

> ''The KDC will then decrypt the STGT using the long term key for the specified server and verify that it is a valid TGT. The session key from the STGT will then be used to encrypt the new ticket that is issued, instead of using the key of the server for which the new ticket will be issued.'' ([RFC 1510], section 3.3.3.)

The server side AUTH-RPC runtime upon receiving this auth information will call the Kerberos read request routine (`krb5_rd_req()`) with a keyproc that extracts the TGT session key from the server's credentials, instead of the current keytable based keyproc routine.

7. DATA STRUCTURES

The following flag will be added to the `sec_rgy_acct_auth_flags_t` definitions in `security/idl/rgynbase.idl`:

```
/*
 * allow and enforce server authentication via the
 * user-to-user protocol
 */
const unsigned32 sec_rgy_acct_auth_user_to_user = 0x40;
```

This flag must be set on a server principal account, in order for a server running as that identity, to successfully register auth info with the RPC runtime using `rpc_server_register_auth_ident()`. With this flag set on a server principal, that identity will no longer be able to use the keytable based server registration function `rpc_server_register_auth_info()`.

The Kerberos KDC also requires that the flag `sec_rgy_acct_auth_dup_skey` be set on a server principal to which user-to-user type tickets will be granted. Setting this flag is sufficient to allow Kerberos only clients/servers to make use of user-to-user functionality. Both this flag and the above newly defined flag must be set on a principal to allow the KDC to generate the tickets and for DCE to allow/enforce usage of user-to-user functionality by this server principal.

8. USER INTERFACES

The registry administration tool (`dcecp`) will need to be updated to include setting/viewing of the new flag described in section 7.

New option for `dcecp account` commands:

> `−usertouser` — Server may only use User-to-User authentication.

This will be just another yes/no flag for display purposes, and will default to no/unset for account creation.

There is no plan or commitment to provide functionality allowing the setting/viewing of this new flag with `rgy_edit`.

9. API'S

The RPC interface (`rpc.idl`) currently defines `rpc_server_register_auth_info()` for server registration with the runtime. This interface will be enhanced with the new User-To-User server registration routine:

```
rpc_server_register_auth_ident(
    [in]    unsigned_char_p_t           server_princ_name,
    [in]    unsigned32                  auth_svc,
    [in]    rpc_auth_identity_handle_t  identity,
    [out]   unsigned32                  *st
);
```

10. REMOTE INTERFACES

The following addition to `src/rpc/sys_idl/mgmt.idl` will allow the client side to fetch both the server name and TGT data, for a UtoU registered server, in one round trip.

```
/*
 * R P C _ _ M G M T _ I N Q _ S V R _ N A M E _ T G T
 */

[idempotent]
void rpc__mgmt_inq_svr_name_tgt
(
    [in]        handle_t                binding_handle,
    [in]        unsigned32              authn_proto,
    [in]        unsigned32              princ_name_size,
    [in]        unsigned32              tkt_size_avail,
    [out, string, size_is(princ_name_size)]
                char                    princ_name[],
    [out]       unsigned32              *tkt_length,
    [out, size_is(tkt_size_avail), length_is(*tkt_length)]
                char                    *data,
```

```
        [out]       error_status_t          *status
    );
```

11. MANAGEMENT INTERFACES

N/A.

12. RESTRICTIONS AND LIMITATIONS

(a) The functionality delivered as part of this work will not allow a server to register using both the old protocol and the new (user2user) protocol, even if registering multiple server names/identities. It should not prevent such functionality in the future, but makes no claims to support it at this time.

(b) This functionality does not provide the capability to register more than one server-name/identity pair from a single server.

13. OTHER DEPENDENCIES

None known at this time.

14. COMPATIBILITY

We have not yet finalized the best way to inform older clients that this server cannot handle the authentication mechanism they support. At this time, the best proposal we can make is to just return the already defined Kerberos error code `KRB5KRB_AP_ERR_METHOD` (‘‘Alternative authentication method required’’). By using an existing error code, older clients will be able to translate that code into an error message using existing message catalogs.

15. STANDARDS

Kerberos Version 5, [RFC 1510]. See also [TM-424].

16. OPEN ISSUES

There are still issues being investigated, regarding credential refresh and its impact on this functionality.

REFERENCES

[RFC 63.1] R. Salz, ‘‘DCE1.2 Contents Overview’’, May 1995.

[RFC 1510] J. Kohl, C. Neuman, ‘‘The Kerberos Network Authentication Service (V5)’’, September 1993.

[TM-424] D. Davis, R. Swick, ‘‘Workstation Services and Kerberos Authentication at Project Athena’’, Technical Memorandum TM-424, MIT LCS, February 1990.

AUTHORS’ ADDRESSES

Michael Burati
Hewlett Packard
300 Apollo Drive
Chelmsford, MA 01824
USA
Internet email: burati@apollo.hp.com
Telephone: +1-508-436-4351

Joe Pato
Hewlett Packard
300 Apollo Drive
Chelmsford, MA 01824
USA
Internet email: pato@apollo.hp.com
Telephone: +1-508-436-4350