NFSv4 Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 14, 2011

W. Adamson
NetApp
K. Coffman
CITI, University of Michigan
N. Williams
Oracle

March 13, 2011

NFSv4 Multi-Domain Access
draft-adamson-nfsv4-multi-domain-access-04

Abstract

The Network File System, version 4 (NFSv4) uses a representation of identity that allows the use of users and groups from multiple, distinct administrative domains, and NFSv4 allows the use of security mechanisms that authenticate principals from multiple, distinct administrative domains. This document describes methods by which NFSv4 clients and servers can handle principals, users, groups from multiple administrative domains.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on September 14, 2011.

Table of Contents

1. Requirements notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Introduction

The NFS Version 4 [RFC3530] protocol enables the construction of a distributed file system which can join NFSv4.0 or NFSv4.1 [I-D.ietf-nfsv4-minorversion1] servers from multiple administrative domains, each potentially using separate name resolution services and separate security services, into a common multi-domain name space.

NFSv4 deals with two kinds of identities: authentication identities (referred to here as "principals") and authorization identities ("users" and "groups" of users). NFSv4 supports multiple authentication methods, each authenticating an "initiator principal" (typically representing a user) to an "acceptor principals" (always corresponding to the server). NFSv4 does not prescribe how to represent authorization identities on file systems. All file access decisions constitute "authorization" and are made by servers using information about client principals (such as username, group memberships, and so on) and file metadata related to authorization, such as a file's access control list (ACL).

Authentication in NFSv4 occurs at the the RPCSEC_GSS [RFC2203] layer where GSS-API mechanisms [RFC2743] are used to authenticate users on NFSv4 clients to NFSv4 servers, and to provide security services such as confidentiality and integrity protection for the protocol's messages. The NFSv4 protocol specifies no particular representation for authentication identities as these are entirely GSS-API mechanism-specific.

Authorization for file object access is done at the NFSv4 protocol layer (i.e., above the RPCSEC_GSS layer), on the server side, based on an authenticated client principal's authorization context and the authorization meta-data of the file system objects that the client wishes to access. File authorization meta-data is set and retrieved in the NFSv4 RPC [RFC1831] layer, specifically via the object's owner, owner_group and acl, dacl and sacl attributes (the last three being ACLs). ACLs are lists of ACL entries (ACEs). Each ACE has a "who" field identifying a subject to whom some permission is granted or denied. The owner and owner_group attributes and the who ACE field, all reference users and groups. On the wire, the protocol represents users and groups as strings of characters with this form: name@domain, where <name> is a user or group name, and <domain> is a the name of an administrative domain, more specifically a DNS [RFC1034] domainname.

NFSv4 server implementations usually do not, and really ought not, store authorization identities on disk in the same form as is used on the wire. The reason is that users' and groups' names change all too often, while searching for and updating file authorization meta-data

after a user/group name change is not trivial, particularly in a global namespace spanning multiple administrative domains.

NFSv4 servers therefore must perform two kinds of mappings:

1. Between the authentication identity and the authorization context (a principal's user ID, group memberships, etcetera)

2. Between the on-the-wire authorization identity representation and the on-disk authorization identity representation.

Many aspects of these mappings are entirely implementation-specific, but some require name resolution services, and in order to interoperate servers must use such services in compatible ways. Many implementations are limited to being able to represent users and groups from a single domain.

In this document we address both of those kind of mappings, describing possible implementation strategies, and specifying a name service for interoperation in a global namespace [I-D.ietf-nfsv4-federated-fs-reqts].

3. Background

NFSv4 uses a syntax of the form "name@domain" to represent, on the wire, the NFSv4 ACL name for users and groups. This design provides a level of indirection that allows a client and server with different internal representations of authorization identity to interoperate even when referring to authorization identities from different administrative domains.

Multi-domain capable sites need to meet certain requirements in order to ensure that clients and servers can map name@domain to internal representations reliably:

- The name portion of name@domain MUST be unique within the specified DNS domain.

- Every local representation of a user and of a group MUST have a canonical name@domain, and it must be possible to return the canonical name@domain for any identity stored on disk, at least when required infrastructure servers (such as name services) are online.

As described in [I-D.ietf-nfsv4-minorversion1] section 2.2.1.1 "RPC Security Flavors":

> NFSv4.1 clients and servers MUST implement RPCSEC_GSS. (This requirement to implement is not a requirement to use.) Other flavors, such as AUTH_NONE, and AUTH_SYS, MAY be implemented as well.

The AUTH_NONE security flavor can be useful to the multi-domain NFSv4 or federated name space to grant universal access to public data without any credentials.

The AUTH_SYS security flavor uses a host-based authentication model where the [weakly-authenticated] client asserts the user's authorization identities using small integers as user and group identity representations. Because of the small integer authorization ID representation, AUTH_SYS can only be used in a name space where all clients and servers share a uidNumber and gidNumber translation service. A shared translation service is required because uidNumbers and gidNumbers are passed in the RPC credential; there is no negotiation of namespace in AUTH_SYS. Collisions can occur if multiple translation services are used. These and other issues are addressed in [I-D.williams-rpcsecgssv3] which describes a new version of RPCSEC_GSS that includes a modernized replacement for AUTH_SYS.

4. Terminology

Identity: a way to refer to a user or group.

Principal: an entity that is authenticated by RPCSEC_GSS (usually, but not always, a user; rarely, if ever, a group; sometimes a host).

Authorization context: the set of user and group IDs, privileges, labels, and other items relevant to authorization, corresponding to a subject (user or principal).

Domain-local ID: Most installations assign numeric, local identifiers to users and groups, using a namespace local to their domain. We call this a domain-local ID.

Local representation of identity: an item such as a POSIX user IDentifier (UID) or group ID (GID), or a Windows Security IDentifier (SID), or other such representation of a user or a group of users. These can be local to a single host.

Global representation of identity: a tuple consisting of a domain identifier (possibly the domain's name itself) and a domain-local user/group ID. We do not propose a standard global representation of identity, but the concept is useful. [NEEDSWORK: we refer to the global representation form in the RPCSEC_GSS PAC]

Group: a security entity representing zero, one or more users and, possibly, other groups. Can appear as the subject of an ACE.

Domain: a set of users, groups and computers administered by a single entity, and identified by a DNS domain name.

POSIX IDs: small non-negative integer (typically 0..2^31 or 0..2^32) identifiers. The namespace of user IDs (UIDs) is distinct from the namespace of group IDs (GIDs).

Windows SIDs: an identifier of security entities, including users and groups. The form of a SID is: S-1-<authority>-<RID_0>-<RID_1> -<RID_n> By convention some authority numbers denote security entities, identified by RID_n, local to a domain identified by RIDs 0..n-1. Domain RIDs are usually generated randomly within a "forest" of domains.

Name resolution: mapping from {domain, name} to {domain, ID}, and vice-versa via lookups. Can be applied to local or remote domains.

ID mapping: {remote domain, remote domain-local ID} to {local representation of ID} mappings.

5. Local Representations of Global Identity

Multi-domain support starts at the fileserver where local ID forms need to be able to represent global identities from both local and remote domains. Local representation of global identity also applies to clients, particularly clients with local filesystems. There's a range of local solutions to this multi-domain ID representation problem. In this section we describe several approaches to representing a <name>@<local or remote domain> on disk. None of these approaches are REQUIRED; all are INFORMATIVE. However, conventions relating to the use of name services are NORMATIVE.

5.1. Storing Name@Domainname

One simple approach to the multiple domain problem is to store the name@domain on disk.

This approach imposes a severe constraint on the administrators of these domains: user and group names must never be reused, as there is also no realistic way to keep the name@domain on disk representation up to date with user, group or domain renames and removals. Consider a remote domain's NFSv4 servers where real-time employee join/leave data may be (typically is!) considered privileged, and remote servers may not be sufficiently privileged to access it [NEEDSWORK].

5.2. Storing Remote-ID@Domainname

Most installations assign numeric, local identifiers to users and groups, using a namespace local to their domain. We call this a domain-local ID. We can then construct a global identity form consisting of a domain name and a domain-local user/group ID.

The user or group renaming issue can be addressed by using a global identity form where domain-local IDs are required. I.e., use name resolution to lookup name@domain to find the ID local to the specified domain, and join the ID with the specified domain name. This function still has a renaming problem with respect to DNS domain renames, but that is a more realistically manageable problem than the user/group renaming problem.

5.2.1. Storing Remote-ID@Domain-ID

The DNS domain renaming issue in the previous section can be addressed by assigning and publishing a unique ID to each DNS domain. I.e., use name resolution to lookup name@domain to find some ID local to the domain, lookup the domain ID and store <remote-ID,domain-ID>. The Windows Security Identifier (SID) is an example of this form.

5.3. ID Mapping

Many file systems exported by NFS only store 32-bit user and group IDs which limit their ability to utilize the on disk representation described in Section 5.2. Such systems may need to use an additional service to map between <remote user ID, local user IDs> and <remote group IDs, local group IDs>. We call this an "ID mapping service".

The use of an ID mapping service is not strictly necessary if the system operates on IDs large enough and in a known format such that <user/group ID, domain ID> can be parsed and encoded into a native ID. However, a large class of operating systems, those which are Unix or Unix-like operating systems, such as Solaris and Linux, use 32-bit UIDs and GIDs in many interfaces and therefore need mapping for backwards compatibility reasons.

One example of such a service is to keep a local or distributed database for dynamically assigning a local 32 bit ID to every <ID>@<domain-ID>, or one could do that only for remote domains, reserving only a small part of the local 32-bit ID namespace for remote domains' users/groups.

The remote ID and remote domain are then used as inputs to a name resolution service which contacts the remote domain name service to resolve the remote name.

5.4. Use of Name Services

File systems often use a distributed directory service for resolving domain local 32 bit IDs to users and groups. The Network Information Service [NIS] and the Lightweight Directory Access Protocol [RFC4511] are the two broadly deployed distributed directory service protocols used for this purpose. LDAP is used instead of NIS in environments where scalabilty, security and/or extensibility are desired. Section 8.2 expands the LDAP protocol to include mappings between principals and local user and group IDs.

Support for LDAP [RFC4511] with the RFC2307 schema [RFC2307] is REQUIRED.

5.4.1. Using LDAP with RFC2307 Schema

Name resolution consists of searches with scope 'sub', a base DN corresponding to a domain (more on this below) and a filter of either of these forms, with matching on objectClass being optional:

- (objectClass=posixAccount)(uid="<username>)
- (objectClass=posixGroup)(cn="<groupname>)
- (objectClass=posixAccount)(uidNumber="<UID>)
- (objectClass=posixGroup)(gidNumber="<GID>)

The base DN SHOULD be formatted from a domain's DNS domainname as follows. First format the domainname as a string, then strip the trailing dot ('.'), if any, then replace all dots ('.') with ",DC=", then prepend "DC=" to the resulting string. For example, foo.bar.example becomes "DC=foo,DC=bar,DC=example". This convention is REQUIRED to be implemented. Domains with base DNs that do not match this convention MAY be used, but their domainname-to-base-DN mappings must be published where NFSv4 clients and servers may find them; we provide no conventions for publishing such mappings. We RECOMMEND that LDAP referrals be used to publish such mappings (e.g., the client does an LDAP search using "DC=foo,DC=example" as the base DN and gets a referral that includes the correct non-standard base DN for "foo.example").

Client and server implementations MUST support the use of LDAP referrals to find LDAP servers authoritative for any given base DN.

For example, to resolve a user named joe@foo.bar.example to a remote ID a system would do an LDAP search with DC=foo,DC=bar,DC=example as the base DN, scope='sub' and with a filter of (objectClass=posixAccount)(uid="<username>) looking for the uidNumber attribute.

5.4.2. Using Active Directory LDAP

[NEEDSWORK: Add text describing searches by which to resolve name@domain to SIDs and vice versa.]

5.4.3. Mapping Domain Names to Domain IDs

[NEEDSWORK: Add text on mapping domainnames to domain IDs. Note that Windows SID does this.]

We need to have a common way to map Domain Names to Domain IDs to enable mult-domain numeric IDs as described in Section 5.2.1. Currently we have two suggestions:

1. Just use SIDs, first asking MSFT to allocate a suitable authority for non-Windows domain SIDs.

2. - Store 96-bit numeric IDs which means we:

   * cast those to domain SIDs later

   * define a non-SID large ID format. This is a fine fallback should MSFT be unwilling to assign authority numbers for this purpose

6. Resolving Cross-Domain Authorization Information

In order to authorize client principal access to files, the NFS server must map the RPCSEC_GSS client principal name or the underlying GSS-API security context to authorization information including a local ID, a set of local group IDs and other local user privileges meaningful to the file system being exported.

In the cross-domain case where a client principal is seeking access to files on a server in a different NFSv4 domain, the NFS server needs to obtain, in a secure manner, the authorization information from an authoritative source: e.g. a directory service in the client principals NFS domain.

There are several methods the cross-domain authoritative authorization information can be obtained:

1. A mechanism specific GSS-API authorization payload containing credential authorization data such as a "privilege attribute certificate" or PAC.

2. An NFS server local domain directory query when there is a security agreement between the two cross-domain directory services plus regular update data feeds so that the NFS server local domain directory service is authoritative for the client prinicpal domain.

3. A direct query from the NFS server to the client principal authoritative directory service.

The authorization data information SHOULD be obtained via the GSS-API name attribute interface [I-D.ietf-kitten-gssapi-naming-exts] either via a single attribute for the credential authorization data or via discrete GSS-API name attributes corresponding to the authorization data elements described in Section 6.1. Details for those attributes are TBD.

Note that the retrieval of attribute values used by the GSS-API name attribute interface implementation could utilize any of the above mentioned methods of obtaining the authorization information.

If the named attribute interface is not available, or the attributes are not available, other means of determining a principal’s authorization data SHOULD be used, such as those described in Section 6.2 and Section 6.3.

6.1. Credential Authorization Data

Here we list in more detail the authorization information that an NFSv4 server needs in order to make a file access decision. The credential authorization data contains the user and group IDs corresponding to the client principal, in global representation of identity form. Note that the server may need to map the global IDs to local IDs as described in Section 5.3.

The ability to map IDs to the name@domain form is required for the NFSv4 server to be able to respond to file authorization meta-data (ACL) set and retrieve requests.

Credential authorization data consists of:

- UserID: This field contains the principal's global ID and/or local ID mapping thereof, and the name@domain form thereof.
- PrimaryGroupID: This field contains the global ID and/or local ID mapping thereof for the principal's primary group, and the name@domain form thereof.
- Groups: This field contains an array of group IDs for the groups that the user is a member of, in global ID form and/or local ID mappings thereof, as well as in name@domain forms.
- Optional field(s) for privileges and authorizations granted to the principal, if any.
- Optional field(s) for other privilege information such as the multi-level security label range/set of the principal.
- Optional implementation-specific items relevant to authorization.

6.2. Using Credential Authorization Data

Authorization context information can sometimes be obtained from the credentials authenticating a principal; the GSS-API represents such information as attributes of the initiator prinicpal name. For example: Kerberos 5 [RFC4120] has a method for conveying "authorization data", both client-asserted as well as KDC-authenticated authorization data, and one KDC implementation uses this feature to convey a "privilege attribute certificate" (PAC) listing the principal's user and group "security identifiers" (SIDs). Another example is the Kerberos General PAC [I-D.sorce-krbwg-general-pac] which lists the principal's user and group "universal user identifiers" (UUIDs) as well as their string representations and DNS domains. PKIX [RFC5280] certificates allow

for extensions that could be used similarly.

6.2.1. Using a PAC

The Windows operating system uses an authorization context called a "PAC" [PAC], which contains a user Security IDentifier (SID) and a list of group SIDs. Some Kerberos Key Distribution Centers (KDCs), notably Windows KDCs, issue Kerberos Tickets with PACs as Kerberos authorization data.

Some KDCs (will) issue Kerberos Tickets with the General PAC [I-D.sorce-krbwg-general-pac] as authorization data. The General PAC authorization data MUST be authenticated in the sense that its contents must come from an authenticated, trusted source, such as a directory server or the issuer of the client principal's credential.

When a client principal is authenticated using such a ticket, the server SHOULD extract the PAC from the client's ticket and map, if need be, the SIDs or UUIDs in the PAC to local ID representations.

The authorization context information in a PAC can be considered a single, authenticated, discrete GSS-API name attribute, in which case the server must parse it into its individual elements.

6.3. Using Directory Services

If suitable and sufficient authenticated GSS-API name attributes for the client principal are not available, then the server may try to map the client principal name to a local notion of user account, and then lookup that user account's authorization context information through authenticated name service lookups.

6.3.1. Mapping Principal Names to Username

One simple method for Kerberos principal-to-username mapping is to first apply an algorithmic or table-based Kerberos client principal realm name to domain name mapping, then a client principal name to username mapping. Finally, the server can look up the user's authorization context using the user's domain's name services.

A trivial Kerberos realm-name-to-domainname mapping consists of using the realm name as the domainname. [NEEDSWORK: Add notes about internationalization.] Servers SHOULD implement this mapping as an option, possibly as a default option.

A trivial Kerberos principal name to username mapping for 1-component principal names is to use the principal name, unmodified, as the username. Servers SHOULD implement this mapping as an option,

possibly as a default option.

6.3.2. Using a Name Service to Map Principal Names to User Accounts

Name services such as the Solaris gsscred database where the local identity is looked up in a database keyed by the GSS exported name token, or LDAP with the extension described in Section 8.2, can be used to map principal names to user accounts.

7. Multi Domain User Group Membership Determination

User group membership is easy to determine for users in a system's local domain: the operating system will already know how to do that. For users in remote domains, the authentication service may provide group membership information. If not, we need methods for group memebership determination.

[NEEDSWORK:

1. provide a[n obviously limited] mode of operation that depends only on RFC2307 and therefore does not support group nesting;

2. provide a more full-fledged mode of operation that depends RFC2307bis;

3. provide a more full-fledged mode of operation that depends AD's schema.]

User group membership in remote domain's groups, and/or for remote users, may be determined using LDAP with the RFC2307 schema. The RFC2307 schema does not define the values of the 'memberUid' attribute, but in practice it seems that those are expected to be the names of users as found in the 'uid' attribute of 'posixAccount' entries. There is work in progress to update RFC2307 [I-D.howard-rfc2307bis] to allow the use of DNs in the member attribute. Group nesting is also enabled.

Assuming the ability to store DNs in the member attribute, then, group membership determination can be done as follows. Given a user ID whose DN has been determined:

1. Search the user's domain for groups that the user is a member of in the user's home domain. Since we cannot assume a domain is authoritative for another domains group membership, filter out groups that are not local to the user's home domain.

2. Search the server's domain for groups that the user is a member of in the server's home domain.

3. Search the server's domain for groups that the user's group memberships determined in steps 1 and 2 are members of.

4. Continue searching for nested group memberships given the list of groups from steps 2 and 3 while being careful to detect or prevent loops.

However, the above procedure has the same user/group name renaming

issue. By skipping step 2 we can get down to just a group renaming issue. To fully address the rename issue we need either a new attribute or value type for memberUid, storing user/group IDs in some global ID representation.

[NEEDSWORK: Add text defining such a new attribute/value type.]

8. LDAP and Multi-Domain NFSv4

Each of the three methods of retrieving cross-domain authorization information described in section Section 6 can require a directory service query. Cross-domain querys are not only inefficient, but also implies knowledge of multiple systems where two different domains rely on completely different infrastructures for user information.

[NEEDSWORK: Describe why LDAP is REQUIRED]

8.1. LDAP Service Discovery

[NEEDSWORK: this is just an idea place holder.]

Two potential methods:

1. Use local methods (configuration, DNS SRV RR lookups, ...) to discover local domain's servers, then depend on LDAP referrals for discovering all other domains servers.

2. Use DNS SRV RRs much the way AD does

NICO: I would prefer that we have one REQUIRED to implement service discovery mechanism as follows:

- specify local DS discovery using DNS SRV RR lookups much like AD does (i.e., have a label to indicate the purpose of the LDAP service, not just _ldap). Make this general enough that clients could discover DSes of remote domains on their own.

- use LDAP referrals (and DNS resolution of the host parts of the referrals) to discover DSes of other domains.

The main benefit of this mechanism is that we can leave the work of finding topologically-close caches and/or authoritative servers to the clients' local DSes, thus avoiding the need to deal directly with topology in our spec.

8.2. LDAP Attribute for Principal Name to Local ID Translation

The gSSPrincipal objectclass allows for the use of the gSSAuthName attribute described in the following section.

```
objectclass ( 1.3.6.1.4.1.250.10.7
NAME 'gSSPrincipal'
DESC 'GSS Principal Name'
SUP  posixAccount
```

```
                        MAY ( gSSAuthName ) )
```

The gSSAuthName attribute provides a method for the translations between a posixAccount and (multiple) GSS-API security principals, used as described in Section 6.3.1.

The gSSAuthName attribute stores a user's GSS-API principal name in exported name token form (see [RFC2743]).

```
                        attributetype ( 1.3.6.1.4.1.250.10.6
                        NAME ( 'gSSAuthName')
                        DESC 'GSS-API exported principal name
                        exported token'
                        EQUALITY bitStringMatch
                        SYNTAX 1.3.6.1.4.1.1466.115.121.1.6)
```

8.3. Name Resolution and LDAP Caching

As noted in Section 5.2, most local representations require a name service to perform ID to name translations. Implementations are REQUIRED to support the use of LDAP as a name service, relying on LDAP referrals for federated namespace construction.

Note that in a topographically widely separated set of domains the need to do name service lookups in various domains' name services may prove brittle, resulting in non-deterministic server behavior (e.g., sometimes a user can access share, sometimes they cannot; sometimes they appear to be members of some group, sometimes they do not). To avoid this, site administrators may wish to maintain local caches of remote domains' name services such that LDAP searches for users/ groups in remote domains can be satisfied locally for some set of key attributes (such as naming and ID attributes), with referrals used in all other cases.

Domains in a federated namespace may provide each other with LDAP LDIF delta feeds by which to maintain cached LDAP contents up to date.The LDAP DN hierarchy described in Section 5.4.1 has the advantage of aiding delta feeds from remote domains where each domain's information is in its own DN subtree.

9. Security Considerations

Caching of remote domains' LDAP search results persents some security considerations. For example, some attributes' values may not be visible unless a user's credentials are used. Some attributes' values may not be intended to be visible to users, but to hosts. Caching servers MUST be capable of issuing referrals as needed for attributes whose values they may not read. Some domain federations will want to have their domains trust each others' caching servers.

More considerations to come

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.

[RFC3530] Shepler, S., Callaghan, B., Robinson, D., Thurlow, R., Beame, C., Eisler, M., and D. Noveck, "Network File System (NFS) version 4 Protocol", RFC 3530, April 2003.

[RFC1831] Srinivasan, R., "RPC: Remote Procedure Call Protocol Specification Version 2", RFC 1831, August 1995.

[RFC2203] Eisler, M., Chiu, A., and L. Ling, "RPCSEC_GSS Protocol Specification", RFC 2203, September 1997.

[RFC2743] Linn, J., "Generic Security Service Application Program Interface Version 2, Update 1", RFC 2743, January 2000.

[RFC4120] Neuman, C., Yu, T., Hartman, S., and K. Raeburn, "The Kerberos Network Authentication Service (V5)", RFC 4120, July 2005.

[RFC4511] Sermersheim, J., "Lightweight Directory Access Protocol (LDAP): The Protocol", RFC 4511, June 2006.

[RFC2307] Howard, L., "An Approach for Using LDAP as a Network Information Service", RFC 2307, March 1998.

[I-D.howard-rfc2307bis] Howard, L. and H. Chu, "An Approach for Using LDAP as a Network Information Service", draft-howard-rfc2307bis-02 (work in progress), August 2009.

[I-D.sorce-krbwg-general-pac] Sorce, S., Yu, T., and T. Hardjono, "A Generalized PAC for Kerberos V5", draft-sorce-krbwg-general-pac-01 (work in progress), December 2010.

[I-D.ietf-nfsv4-federated-fs-reqts] Lentini, J., Everhart, C., Ellard, D., Tewari, R., and M. Naik, "Requirements for Federated File Systems", draft-ietf-nfsv4-federated-fs-reqts-06 (work in progress), October 2009.

[I-D.ietf-kitten-gssapi-naming-exts]
Williams, N. and L. Johansson, "GSS-API Naming Extensions", draft-ietf-kitten-gssapi-naming-exts-09 (work in progress), February 2011.

[I-D.ietf-nfsv4-minorversion1]
Shepler, S., Eisler, M., and D. Noveck, "NFS Version 4 Minor Version 1", draft-ietf-nfsv4-minorversion1-29 (work in progress), December 2008.

[PAC] Brezak, J., "Utilizing the Windows 2000 Authorization Data in Kerberos Tickets for Access Control to Resources", October 2002.

10.2. Informative References

[I-D.williams-rpcsecgssv3]
Haynes, T. and N. Williams, "Remote Procedure Call (RPC) Security Version 3", draft-williams-rpcsecgssv3-01 (work in progress), March 2011.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.

[NIS] Sun Microsystems, "System and Network Administration", March 1990.

Authors' Addresses

William A. (Andy) Adamson
NetApp

Email: andros@netapp.com

Kevin Coffman
CITI, University of Michigan

Email: kwc@umich.edu

Nicolas Williams
Oracle

Email: nicolas.williams@oracle.com