NFSv4 Working Group
Internet-Draft
Intended status: Standards Track
Expires: March 26, 2015

W. Adamson
NetApp
N. Williams
Cryptonector

September 22, 2014

NFSv4 Multi-Domain FedFS Requirements
draft-adamson-nfsv4-multi-domain-federated-fs-reqs-05

Abstract

This document describes constraints to the NFSv4.0 and NFSv4.1 protocols as well as the use of multi-domain capable file systems, name resolution services, and security services required to fully enable a multi NFSv4 domain federated file system.

Requirements Language

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].

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 March 26, 2015.

Copyright Notice

Table of Contents

1. Introduction

An NFSv4 domain is defined as a set of users, groups and computers running NFSv4.0 [I-D.ietf-nfsv4-rfc3530bis] and NFSv4.1 [RFC5661] (hereafter referred to as NFSv4) protocols identified by an NFSv4 domain name.

The federated file system (FedFS) [RFC5716] describes the requirements and administrative tools to construct a uniform NFSv4 file server based namespace that is capable of spanning a whole enterprise and that is easy to manage.

The FedFS is the standardized method of constructing and administrating an enterprise wide NFSv4 filesystem, and so is referenced in this document. The issues with multi NFSv4 domain file systems described in this document apply to all such file systems, be they run as a FedFS or not.

Stand-alone NFSv4 domains can be run in many ways. While a FedFS can be run within all stand-alone NFSv4 domain configurations some of these configurations (Section 4) are not compatible with joining a multi NFSv4 domain FedFS namespace.

Multi NFSv4 domain file systems require support for global identities in name services, security services, and in the exporting of on-disk local identity representation. Many of the stand-alone NFSv4 domain deployments do not provide full support for global identities.

This document describes constraints to the NFSv4 protocols as well as the use of multi-domain capable file systems, name resolution services, and security services required to fully enable a multi NFSv4 domain file system, such as a multi NFSv4 domain FedFS.

2. Terminology

Name Service: provides the mapping between {NFSv4 domain, group or user name} and {NFSv4 domain, local ID}, as well as the mapping between {security principal} and {NFSv4 domain, local ID} via lookups. Can be applied to local or remote domains. Often provided by a Directory Service such as LDAP.

Domain: This term is used in multiple contexts where it has different meanings. Here we provide specific definitions used in this document.

DNS domain: a set of computers, services, or any internet resource identified by an DNS domain name [RFC1034].

Security realm or domain: a set of configured security providers, users, groups, security roles, and security policies running a single security protocol and administered by a single entity, for example a Kerberos realm.

NFSv4 domain: a set of users, groups, and computers running NFSv4 protocols identified by a unique NFSv4 domain name. See [RFC5661] Section 5.9 "Interpreting owner and owner_group".

Multi-domain: In this document this always refers to multiple NFSv4 domains.

FedFS domain: A file name space that can cross multiple shares on multiple file servers using file-access protocols such as NFSv4. A FedFS domain is typically a single administrative entity, and has a name that is similar to a DNS domain name. Also known as a Federation.

Administrative domain: a set of users, groups, computers, and services administered by a single entity. Can include multiple DNS domains, NFSv4 domains, security domains, and FedFS domains.

Local representation of identity: an object such as a uidNumber (UID) or gidNumber (GID) [RFC2307], a Windows Security Identifier (SID) [CIFS], or other such representation of a user or a group of users on-disk in a file system.

Global identity: An on-the-wire globally unique form of identity that can be mapped to a local representation. For example, the NFSv4 name@domain or the Kerberos principal@REALM.

Multi-domain capable filesystem: A local filesystem that uses a local ID form that can represent identities from both local and remote domains. For example, an SSID based local ID form where the SSID contains both a domain and a user or group component.

Principal: an RPCSEC_GSS authentication identity. Usually, but not always, a user; rarely, if ever, a group; sometimes a host or server.

Authorization Context: A collection of information about a principal such as username, userID, group membership, etcetera used in authorization decisions.

Stringified UID or GID: NFSv4 owner and group strings that consist of decimal numeric values with no leading zeros, and which do not contain an '@' sign. See Section 5.9 "Interpreting owner and owner_group" [RFC5661].

3. NFSv4 Server Identity Mapping

NFSv4 servers deal 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 principal" (always corresponding to the NFSv4 server). NFSv4 does not prescribe how to represent authorization identities on file systems. All file access decisions constitute "authorization" and are made by NFSv4 servers using authorization context information and file metadata related to authorization, such as a file's access control list (ACL).

NFSv4 servers therefore must perform two kinds of mappings:

1. Auth-to-authz: A mapping between the authentication identity and the authorization context information.

2. Wire-to-disk: A mapping between the on-the-wire authorization identity representation and the on-disk authorization identity representation.

A Name Service such as LDAP often provides these mappings.

Many aspects of these mappings are entirely implementation specific, but some require multi-domain capable name resolution and security services in order to interoperate in a multi NFSv4 domain file system.

NFSv4 servers use these mappings for:

1. File access: Both the auth-to-authz and the wire-to-disk mappings may be required for file access decisions.

2. Meta-data setting and listing: The auth-to-authz mapping is usually required to service file metadata setting or listing requests (such as ACL or unix permission setting or listing) as NFSv4 uses the name@domain on-the-wire identity representation which usually differs from the exported on-disk identity representation.

4. Stand-alone NFSv4 Domain Deployment Examples

In order to service as many environments as possible, the NFSv4 protocol is designed to allow administrators freedom to configure their NFSv4 domains as they please.

Stand-alone NFSv4 domains can be run in many ways. Here we list some stand-alone NFSv4 domain deployment examples focusing on the NFSv4 server's use of name service mappings (Section 3) and security services deployment to demonstrate the need for some multi-domain constraints to the NFSv4 protocol, name service configuration, and security service choices.

Because all on-disk identities participating in a stand-alone NFSv4 domain belong to the same NFSv4 domain, stand-alone NFSv4 domain deployments have no requirement for exporting multi-domain capable file systems.

These examples are for a NFSv4 server exporting a 32bit UID/GID based file system, a typical deployment. These examples are listed in the order of increasing NFSv4 administrative complexity.

4.1. AUTH_SYS with Stringified UID/GID

This example is the closest NFSv4 gets to being run as NFSv3.

File access: The AUTH_SYS RPC credential provides a UID as the authentication identity, and a list of GIDs as authorization context information. File access decisions require no name service interaction as the on-the-wire and on-disk representation are the same and the auth-to-authz UID and GID authorization context information is provided in the RPC credential.

Meta-data setting and listing: When the NFSv4 clients and servers implement a stringified UID/GID scheme, where a stringified UID or GID is used for the NFSv4 name@domain on-the-wire identity, then a name service is not required for file metadata listing as the UID or GID can be constructed from the stringified form on the fly by the server.

4.2. AUTH_SYS with name@domain

The next level of complexity is to not use a stringified UID/GID scheme for file metadata listing.

File access: This is the same as in Section 4.1.

Meta-data setting and listing: The NFSv4 server will need to use a name service for the wire-to-disk mappings to map between the on-the-wire name@domain syntax and the on-disk UID/GID representation. Often, the NFSv4 server will use the nsswitch interface for these mappings. A typical use of the nsswitch name service interface uses no domain component, just the uid attribute [RFC2307] (or login name) as the name component. This is no issue in a stand-alone NFSv4 domain deployment as the NFSv4 domain is known to the NFSv4 server and can be added after the return of the name service call.

4.3. RPCSEC_GSS with name@domain

This final example adds the complexity of RPCSEC_GSS with the Kerberos 5 GSS security mechanism.

File Access: The RPCSEC_GSS Kerberos credential provides a principal@REALM name as the authentication identity, and (as of this writing) no authorization context information. File access decisions therefore require a wire-to-disk mapping of the principal@REALM to a UID, and an auth-to-authz mapping to obtain the list of GIDs as the authorization context.

Deployments can use the nsswitch name service interface for the principal@REALM to UID mapping by stripping off the REALM portion. This requires that the principal portion of the principal@REALM matches the uid attribute [RFC2307] (or login name).

Meta-data setting and listing: This is the same as in Section 4.2.

5. Multi-domain Constraints to the NFSv4 Protocol

Joining NFSv4 domains under a single file namespace imposes slightly on the NFSv4 administration freedom. Here we describe the required constraints.

5.1. Name@domain Constraints

NFSv4 uses a syntax of the form "name@domain" as the on wire representation of the "who" field of an NFSv4 access control entry (ACE) for users and groups. This design provides a level of indirection that allows NFSv4 clients and servers with different internal representations of authorization identity to interoperate even when referring to authorization identities from different NFSv4 domains.

NFSv4 multi-domain capable sites need to meet the following requirements in order to ensure that NFSv4 clients and servers can map between name@domain and internal representations reliably. While some of these constraints are basic assumptions in NFSv4.0 [I-D.ietf-nfsv4-rfc3530bis] and NFSv4.1 [RFC5661], they need to be clearly stated for the NFSv4 mulit-domain case.

- The NFSv4 domain portion of name@domain MUST be unique within the NFSv4 multi-domain namespace. See [RFC5661] section 5.9 "Interpreting owner and owner_group" for a discussion on NFSv4 domain configuration.

- The name portion of name@domain MUST be unique within the specified NFSv4 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.

Due to UID and GID collisions, stringified UID/GIDs MUST not be used in a multi NFSv4 domain file system.

Note that for stand-alone NFSv4 domains it does not matter if the choice of the NFSv4 domain name is replicated by another stand-alone NFSv4 domain deployment. Indeed, if a stringified UID/GID scheme is used, or just UNIX mode bits are used (NFSv4 ACLs are not set or listed) and the simple nsswitch interface that strips the @domain and the @REALM is used, then the domain portion of name@domain can be ignored, and even be different for each client and server in the domain.

5.1.1. NFSv4 Domain and DNS Services

Here we address the relationship between NFSv4 domains and DNS domains in an multi NFSv4 domain deployment.

The definition of an NFSv4 domain name needs clarification to work in a multi-domain file system name space. Section 5.9 [RFC5661] loosely defines the NFSv4 domain name as a DNS domain name. This loose definition for the NFSv4 domain is a good one, as DNS domain names are globally unique. As noted above Section 5.1 , pretty much any choice of NFSv4 domain name can work within a stand-alone NFSv4 domain deployment whereas the NFSv4 domain is required to be unique in a multi NFSv4 domain deployment.

A typical configuration is that there is a single NFSv4 domain that is served by a single DNS domain. In this case the NFSv4 domain name can be the same as the DNS domain name.

An NFSv4 domain can span multiple DNS domains. In this case, one of the DNS domain names can be chosen as the NFSv4 domain name.

Multiple NFSv4 domains can also share a DNS domain. In this case, only one of the NFSv4 domains can use the DNS domain name, the other NFSv4 domains must choose another unique NFSv4 domain name.

5.1.2. NFSv4 Domain, Name Service, and Domain Aware File Systems

As noted above Section 5.1, each name@domain is unique across the multi NFSv4 domain namespace, and maps to a local representation of ID in each NFSv4 domain. This means that each NFSv4 domain has a single name resolution service exporting the NFSv4 domain local ID name space.

An NFSv4 domain administrator that wants to give NFSv4 local file access to a remote user from a different NFSv4 domain needs to create a local ID for the remote user which can then be assigned on-disk and used for local access decisions. Since the local ID for the remote user must be able to be mapped to a name@remote-domain, only multi-

domain capable file systems can be exported in a multi NFSv4 domain FedFS.

We note that many file systems exported by NFSv4 use 32 bit POSIX UID and GIDs as a local ID form and as this local ID form has no domain component, these file systems are not domain aware and can not participate in a multi NFSv4 domain FedFS. There are ways to overcome this deficiency, but these practices are beyond the scope of this document.

5.2. RPC Security Constraints

As described in [RFC5661] 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 underlying RPCSEC_GSS security mechanism used in a multi NFSv4 domain FedFS is REQUIRED to employ a method of cross NFSv4 domain trust so that a principal from a security service in one NFSv4 domain can be authenticated in another NFSv4 domain that uses a security service with the same security mechanism. Kerberos, and PKU2U [I-D.zhu-pku2u] are examples of such security services.

The AUTH_NONE security flavor can be useful in a multi NFSv4 domain FedFS 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 host (the NFSv4 client) asserts the user's authorization identities using small integers, uidNumber, and gidNumber [RFC2307], as user and group identity representations. Because this authorization ID representation has no DNS domain component, AUTH_SYS can only be used in a name space where all NFSv4 clients and servers share an [RFC2307] name service. A shared name 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 name services are used, so AUTH_SYS MUST not be used in a multi NFSv4 domain FedFS.

5.2.1. NFSv4 Domain and Security Services

As noted above in Section 5.2, caveat AUTH_NULL, multi NFSv4 domain security services are RPCSEC_GSS based with the Kerberos 5 security mechanism being the most commonly (and as of this writing, the only) deployed service.

A single Kerberos 5 security service per NFSv4 domain with the upper case NFSv4 domain name as the Kerberos 5 REALM name is a common deployment.

Multiple security services per NFSv4 domain is allowed, and brings the issue of mapping multiple Kerberos 5 principal@REALMs to the same local ID. Methods of achieving this are beyond the scope of this document.

6. Resolving Multi-domain Authorization Information

When an RPCSEC_GSS principal is seeking access to files on an NFSv4 server, after authenticating the principal, the server must obtain in a secure manner the principal's authorization context information from an authoritative source such as the name service in the principal's NFSv4 domain.

In the stand-alone NFSv4 domain case where the principal is seeking access to files on an NFSv4 server in the principal's home NFSv4 domain, the server administrator has knowledge of the local policies and methods for obtaining the principal's authorization information and the mappings to local representation of identity from an authoritative source. E.g., the administrator can configure secure access to the local NFSv4 domain name service.

In the multi-domain case where a principal is seeking access to files on an NFSv4 server not in the principal's home NFSv4 domain, the server is REQUIRED to obtain the principals' authorization context information from an authoritative source. In this case there is no assumption of:

- Remote name service configuration knowledge
- The syntax of the remote authorization context information presented to the NFSv4 server by the remote name service for mapping to a local representation.

There are several methods the NFSv4 server can use to obtain the NFSv4 domain authoritative authorization information for a remote principal from an authoritative source. While any detail is beyond the scope of this document, some general methods are listed here.

1. A mechanism specific GSS-API authorization payload containing credential authorization data such as a "privilege attribute certificate" (PAC) [PAC] or a "general PAD" (PAD) [I-D.sorce-krbwg-general-pac]. This is the preferred method as the payload is delivered as part of GSS-API authentication,

avoids requiring any knowledge of the remote authoritative service configuration, and its syntax is well known.

2. When there is a security agreement between the local and remote NFSv4 domain name services plus regular update data feeds, the NFSv4 server local NFSv4 domain name service can be authoritative for principal's in the remote NFSv4 domain. In this case, the NFSv4 server makes a query to it's local NFSv4 domain name service just as it does when servicing a local domain principal. While this requires detailed knowledge of the remote NFSv4 domains name service, the authorization context information presented to the NFSv4 server is in the same form as a query for a local principal.

3. An authenticated direct query from the NFSv4 server to the principal's NFSv4 domain authoritative name service. This requires the NFSv4 server to have detailed knowledge of the remote NFSv4 domain's authoritative name service and detailed knowledge of the syntax of the resultant authorization context information.

7. Stand-alone Examples and Multi NFSv4 Domain FedFS

Revisiting the stand-alone (Section 4) NFSv4 domain deployment examples, we note that due to the use of AUTH_SYS, neither Section 4.1 nor Section 4.2 configurations are suitable for multi NFSv4 domain deployment.

The Section 4.3 configuration example can participate in a multi NFSv4 domain FedFS deployment if:

- The NFSv4 domain name is unique across the FedFS.
- All exported file systems are multi-domain capable.
- A secure method is used to resolve remote NFSv4 domain principals authorization information from an authoritative source.

8. Security Considerations

There are no security considerations introduced by this document beyond those described in NFSv4.0 [I-D.ietf-nfsv4-rfc3530bis] and NFSv4.1 [RFC5661].

9. Normative References

[CIFS] Microsoft Corporation, "[MS-CIFS] -- v20130118 Common Internet File System (CIFS) Protocol", January 2013.

[I-D.ietf-nfsv4-rfc3530bis] Haynes, T. and D. Noveck, "Network File System (NFS) version 4 Protocol", draft-ietf-nfsv4-rfc3530bis-25 (Work In Progress), February 2013.

[I-D.sorce-krbwg-general-pac] Sorce, S., Yu, T., and T. Hardjono, "A Generalized PAC for Kerberos V5", draft-ietf-krb-wg-general-pac-02 (Work In Progress awaiting merge with other document), June 2011.

[I-D.zhu-pku2u] Zhu, L., Altman, J., and N. Williams, "Public Key Cryptography Based User-to-User Authentication - (PKU2U)", draft-zhu-pku2u-09 (Work In Progress), November 2008.

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

[RFC1034] Mockapetris, P., "DOMAIN NAMES - CONCEPTS AND FACILITIES", RFC 1034, November 1987.

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

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

[RFC5661] Shepler, S., Eisler, M., and D. Noveck, "Network File System (NFS) Version 4 Minor Version 1 Protocol", RFC 5661, January 2010.

[RFC5716] Lentini, J., Everhart, C., Ellard, D., Tewari, R., and M. Naik, "Requirements for Federated File Systems", RFC 5716, January 2010.

Appendix A. Acknowledgments

Andy Adamson would like to thank NetApp, Inc. for its funding of his time on this project.

We thank Chuck Lever, Tom Haynes, Brian Reitz, and Bruce Fields for their review.

Authors' Addresses

William A. (Andy) Adamson
NetApp

Email: andros@netapp.com

Nicolas Williams
Cryptonector

Email: nico@cryptonector.com