

Network File System Version 4
Internet-Draft
Intended status: Informational
Expires: 25 July 2021

D. Noveck
NetApp
C. Lever, Ed.
Oracle
21 January 2021

Security Needs for the NFSv4 Protocols
draft-dnoveck-nfsv4-security-needs-02

Abstract

This document discusses the inadequate approach to security within the family of NFSv4 protocol specifications and proposes steps to correct the situation. Because the security architecture is similar for all NFSv4 minor versions, we recommend a single new standards-track document to encapsulate NFSv4 security fundamentals, and propose the introduction of several additional security-related documents.

Note

Discussion of this draft takes place on the NFSv4 working group mailing list (nfsv4@ietf.org), which is archived at <https://mailarchive.ietf.org/arch/browse/nfsv4/>. Working Group information can be found at <https://datatracker.ietf.org/wg/nfsv4/about/>.

This note is to be removed before publishing as an RFC.

The source for this draft is maintained in GitHub. Suggested changes should be submitted as pull requests at <https://github.com/chucklever/i-d-security-needs>. Instructions are on that page as well.

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 <https://datatracker.ietf.org/drafts/current/>.

Internet-Draft

NFSv4 Security Needs

January 2021

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 25 July 2021.

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the [Trust Legal Provisions](#) and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1.	Introduction	4
2.	Terminology	4
2.1.	Requirements Language	4
2.2.	Requirements Language as Used in This Document	4
2.3.	Glossary	5
3.	Use of this Document	6
3.1.	Current Use of this Document	6
3.2.	Future Use of this Document	7
4.	Situation to be Addressed	8
4.1.	NFSv4 Use Environments to be Addressed	8
4.2.	Emergence and Correction of Security Issues	9
5.	Major Problems to Address	12
5.1.	Problems with Security Presentation/Organization	12
5.1.1.	Problems with Presentation of Security Architecture	13
5.1.2.	Problems with Security Evaluation	15
5.2.	The Treatment of AUTH_SYS	15
5.2.1.	Current AUTH_SYS Security Policies	16
5.2.2.	Working Group Actions	18
5.3.	Problems with Confidentiality	20

5.4.	File Access Control	22
5.4.1.	File Content Integrity and Provenance	23
6.	Framework for Correcting Problems	23
6.1.	Correcting Problems with Regard to Threat Analyses . . .	24

6.2.	Correcting Problems with Regard to Use of Normative Terms	25
7.	Opportunities for Improvement Provided by Recent Work within the RPC Layer	26
7.1.	Opportunities for Improvement in Encryption	26
7.2.	Opportunities for Improvement in Authentication	27
7.3.	Opportunities for Improvement when Using RDMA Transports	27
8.	Issues that Need to be Addressed	28
8.1.	Threat Analysis Goals	28
8.1.1.	Background	28
8.1.2.	Issues to be Addressed	30
8.2.	NFSv4 Extension Policies	30
8.2.1.	Background	30
8.2.2.	Addressing Extension Issues	31
8.3.	TLS Encryption Policies	32
8.3.1.	Background	32
8.3.2.	Issues to Address	33
8.4.	Handling of AUTH_SYS	36
8.4.1.	Historical Background	36
8.4.2.	Background for Existing AUTH_SYS in NFSv4	37
8.4.3.	Core Issue to Resolve for AUTH_SYS	38
8.4.4.	Need to Better Document and Explain Issues with AUTH_SYS	38
8.4.5.	Issues to Address for Existing Use of AUTH_SYS . . .	40
8.4.6.	Issues to Resolve for Revised Approach to AUTH_SYS .	43
8.5.	Handling of State Protection	47
8.5.1.	Background	47
8.5.2.	Issues to be Addressed	48
9.	IANA Considerations	49
10.	Security Considerations	49
10.1.	Security Considerations Section for Eventual NFSv4-wide Standards-track Security Document	50
10.2.	Security Considerations Section for Eventual Rfc5661bis	50
10.3.	Security Considerations Section for Potential Revised RPC	

Specification Document	51
10.4. Security Considerations Section for Potential Standards-track Document Dealing with AUTH_SYS	52
11 . References	52
11.1 . Normative References	52
11.2 . Informative References	53
Acknowledgements	54
Authors' Addresses	54

[1](#). Introduction

This document proposes specification changes that modernize NFSv4 security. These changes are necessary because the original goal for the NFSv4 protocol, to enable secure file exchange on the open internet [[RFC2624](#)], has not been effectively realized. Moreover, existing approaches to NFSv4 security do not meet the security needs of many contemporary restricted-access environments.

Restricting physical access to the network on which a client and server interact is a common NFSv4 deployment technique. This restriction limits access to users associated with a particular organization but does not eliminate the need for protocol support that enables completely secure operation. [Section 4.1](#) contains a taxonomy of specific environments and their corresponding security needs.

As an example, the use of transport-layer security, including the encryption of network traffic and the use of client host authentication, can improve security when AUTH_SYS [[RFC5531](#)] is in use, replacing the security approaches specified in [[RFC7530](#)] and [[RFC8881](#)]. For a discussion of issues connected to shifting the security approach of mature protocols, see [Section 4.2](#).

The current document serves as a resource for the nfsv4 working group as it updates current standards-track documents and proposes new standards-track documents to address security issues within the NFSv4 protocol. For further details on the expected use of this document, see [Section 3](#).

[2.](#) Terminology

[2.1.](#) Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [BCP 14](#) [[RFC2119](#)] [[RFC8174](#)] when, and only when, they appear in all capitals, as shown here.

[2.2.](#) Requirements Language as Used in This Document

The current document is Informational. Therefore it cannot make normative statements using the keywords defined in [Section 2.1](#).

However, in discussing the treatment of security within the set of NFSv4 specifications, there are occasions in which the current document includes quotations from existing standards-track documents

that use these keywords. In such cases, the definitions within [BCP 14](#) [[RFC2119](#)] [[RFC8174](#)] are to be adhered to when the terms appear in all-capitals.

The current document also uses these keywords when exploring future standards-track documents. Although these terms do not have a normative effect in this setting, their meaning remains the same. These statements can differ from and may contradict existing normative statements for one or more of the following reasons:

- * Earlier standards-track documents did not handle a security issue appropriately.
- * The community did not recognize a security issue when a standards-track document was initially approved.
- * An actual or potential security feature was not available at the time standards-track documents were written and approved.

Changes in normative statements applying to implementations of a particular NFSv4 minor version require special care to avoid creating later difficulties. Problems might arise because:

- * Implementers might choose to leave unimplemented changes that are not mandatory-to-implement.
- * A client implementation assumes that a particular server behavior is foreclosed (e.g., by a "MUST NOT" or "SHOULD NOT"), but a future change allows that behavior.

[2.3.](#) Glossary

More complete definitions to be filled in later. The security definitions are based on [\[RFC3552\]](#) and [\[RFC4949\]](#).

Authentication Flavor

As defined in [Section 8.2 of \[RFC5531\]](#).

Confidentiality

Data is kept secret from unintended listeners. RPCSEC_GSS sometimes uses the term "privacy" to mean the same thing. As defined in [Section 2.1.1 of \[RFC3552\]](#).

Host authentication

The remote endpoint in the communication is the one we intended. Also referred to as "peer authentication". As defined in [Section 2.1.3 of \[RFC3552\]](#).

Identity

A local trusted label for an object or resource, such as a network address or a UID.

Integrity

Data that is received (or read) is the same data that was sent (or written). As defined in [Section 2.1.2 of \[RFC3552\]](#).

Non-repudiation

Demonstration to a third party (or the same parties at a later time) that both a sender's identity is certain and the data received from that sender is the same as the data that was sent. As defined in [Section 2.2 of \[RFC3552\]](#).

Trust

As defined in [Section 4 of \[RFC4949\]](#).

User authentication

Establish the truth that a user is who (s)he claims to be. As discussed in [Section 4.1 of \[RFC3552\]](#).

[3.](#) Use of this Document

[3.1.](#) Current Use of this Document

This document is a means to facilitate discussion of the specific inadequacies that need to be addressed within a new approach to NFSv4 security that we believe is required. For each specific inadequacy, as discussion proceeds, the treatment is expected evolve downward in the following list to enable the working group to formulate text for new standards-track documents to address the problem:

- * The statement that a specific inadequacy exists and needs to be dealt with.
- * A listing of potential ways to deal with a specific inadequacy.
- * A proposal by the authors that a specific adequacy is best dealt with a particular way, possibly accompanied by a list of other reasonable approaches.
- * A statement that the working is expected to address a specific inadequacy in a particular way.
- * A statement that the working is addressing a specific inadequacy in a particular way.

While it is expected that, if the working group concurs, this document will be adopted as a working group document, that transition is not expected to affect the document's pattern of use. When deciding on that issue the working group will not be considering the specific propoals to address inadequacies, which can be addressed later. The working group will be considering whether there are serous inadequacies requiring new documents and whether such a document will be helpful in addressing them.

The working group discussion anticipated, expected to be documented in future versions of this document, will guide the writing of a number of expected standards-track documents:

- * A new standards-track document discussing security for all NFSv4 minor versions, updating [\[RFC7530\]](#) and [\[RFC8881\]](#). It is intended that this document be referred to by rfc5661bis when it is ready for publication but it should be able to be published, on its own, before that, since it would update [\[RFC8881\]](#).
- * To address NFSv4.1-specific security issues, new text will be needed in an anticipated rfc5661bis, updating [\[RFC8881\]](#).

[draft-dnoveck-nfsv4-rfc5661bis](#) [\[I-D.dnoveck-nfsv4-rfc5661bis\]](#) is anticipated to be a precursor document whose drafting will be affected.

In addition, the working group discussion anticipated might guide the writing of a number of potential working group documents:

- * A new standards-track document discussing AUTH_SYS, including discussion of the potential use of client-host authentication.
- * A new standards-track document updating [\[RFC5531\]](#), focusing on a more complete and accurate treatment of AUTH_SYS.

[3.2.](#) Future Use of this Document

The pattern of use discussed above is expected to continue until the decisions to be made are finalized. While the documents whose drafting is to be guided, will be farly far into development, it is not necessary for the working group to wait for publication of those documents (or even WGLC) before scheduling WGLC for this one.

The question of possible publication of this document as an informational RFC remains open. It should probably be discussed shortly before WGLC. Given this uncertainty, if there is to be a milestone associated with this document, the target should be WGLC.

[4.](#) Situation to be Addressed

4.1. NFSv4 Use Environments to be Addressed

We will consider a range of network environments and how the type of environment affects the need for the features being introduced to provide for secure operation.

1. Within a data center local area network, there may well be sufficient administrative controls over the software running on machines with physical access to the network as to make additional security features within NFS unnecessary.

If such a network is physically isolated or has all access to it controlled using an appropriately configured firewall, it may be acceptable, from a security point of view, to use AUTH_SYS without client host authentication, and without encryption, assuming all attackers have been excluded from access.

Although such networks typically do have firewalls, they are generally configured to allow significant access, including NFSv4 access, to traffic originating outside the data center, although not outside the owning organization. In such cases, we effectively have a within-organization network, dealt with in item 2 below.

2. Within a network devoted to a single organization or a single site within an organization. Such networks often have associated firewalls configured to exclude access from outside the organization, and restrict it inside the organization.

It had until recently been assumed that, by excluding access to those outside the organization, it could be assumed that attackers would also be excluded. However, this approach ignores the possibility of insider threats. Since we feel these threats cannot be ignored, the security of NFSv4 needs to be upgraded to prevent attacks by insiders, making the security needs in this case more like those on the internet.

3. Use on the internet is the most challenging. While secure use on the internet was a goal of NFSv4 and steps were taken to achieve that goal, the approach required a number of steps, including a significant performance penalty and a significantly different approach to systems administration. Given that those administering such systems were unwilling to adapt in this way, we now need to meet the goal of secure use on the internet in a new way.

Secure use on the internet requires not only protection of user data in-flight from unauthorized disclosure or modification, as in case 2 above, but also effort to deal with the possibility of extensive external monitoring and denial-of-service attacks.

4. Use of NFSv4 to access data located in the cloud poses many of issues discussed in item 3 above. However, in general, the cloud service provider is responsible for protecting multiple tenants from one another, often involving use of tenant-specific credentials. If the distribution of such credentials is quite tightly limited, we can have a situation like that in item 1. However, if those credentials are available to a larger group, then the situation become like that in item 2, with insider threats being a point of concern.
5. The use of NFSv4 within cloud-located data centers presents different issues. If access to the NFSv4 service is suitably restricted, the situation is quite close to that described in case 1, with the physical location of the data center irrelevant. However, if there means of external access that are not tightly restricted, the situation becomes like that of case 2 above, with the possibility of insider threats requiring a security model with many similarities to that needed for the environments described in item 3.

As will be discussed below, NFSv4 security, as currently defined, has significant security issues in many of the above environments, although not in all. No special provisions will be made for the more restricted environments, except where necessary to avoid incompatibilities when interacting with existing implementations.

By focusing on a single set of requirements for all of these environments, there will necessarily be occasions in which some security-related work is required that is not, strictly speaking, needed for the particular environment in which it is used. However, we have endeavored to keep the required effort small, eliminating requirements for major administrative changes or compute-intensive additions to the processing path.

[4.2.](#) Emergence and Correction of Security Issues

The fact that serious security issues exist in many of the environment discussed above has made it necessary to consider taking the drastic step of modifying security-related recommendations for existing protocols. This situation seems to call for an explanation and we will attempt to provide one.

One possible form this might take would be to explain why [[RFC7530](#)] and [[RFC5661](#)] made erroneous choices, with accompanying commentary explaining the choices that should have been made instead.

Such an explanation does not appear to be possible, since it would not have been possible for the authors and approvers of these documents to reach the conclusions we have adopted without an implausible level of foresight on their part. This is so even though we would not adopt the choices made in [[RFC7530](#)] and [[RFC5661](#)] today. Knowing what was known then, we might well have made the same choices, although it is hard not to fault the lack of attention, in the relevant documents, to the security consequences of the decisions made.

Although we will call attention in sections below to things which might have been done differently, this does indicate that we believe that the authors and approvers of these documents, could have arrived at conclusions similar to the ones we expect to choose now. Our choices are so different from the ones made previously because they take into account the needs of NFSv4 today and take advantage of security facilities not available earlier.

In order to better understand this divergence, we have to look at the history of NFS and take note of the context in which assumptions were established with regard to security or changes in security guidance occurred. In understanding this history and the choices of [RFC2119](#)-defined keywords, the constraints on the working group need to be understood, as described in [Section 6.2](#)

- * The basic assumptions about security for NFS were established over thirty years ago. At that time, it was generally taken for granted that a co-operating set of UNIX kernels would provide a uniform infrastructure to govern internode sharing within a network. Within this framework, security concerns focused on issues related to the appropriateness of granting or withholding of privileges by UNIX kernels and there was little concern for network security as understood today. As a result, NFS administrators came to take for granted the use of AUTH_SYS making it difficult to restrict its use, despite the security weaknesses

we perceive in it today. The assumptions underlying AUTH_SYS fit so well with the way UNIX systems were managed in general led to the use of AUTH_SYS becoming so well-entrenched that concrete steps to eliminate it or to substantially restrict its use were never seriously considered.

- * When NFSv4 was developed, around twenty years ago, serious steps were taken to improve the security situation by requiring the implementation of RPCSEC_GSS. This included support for

encryption, necessary to support use on the internet. AUTH_SYS was retained as an additional optional means of authentication, which would have allowed implementations to drop support at a later point, although, as things developed, that became less and less likely as use of AUTH_SYS remained quite common.

While it is possible to interpret the treatment of the matter in [[RFC3530](#)] as implying that these two (i.e. RPCSEC_GSS and AUTH_SYS) were equivalent and to find fault with that implication, it is not clear how different choices would have led to a better result. In particular, attempts to deprecate AUTH_SYS (e.g. by using the phrase "SHOULD NOT") would have been problematic in that they might have retarded adoption of NFSv4 rather than enhancing adoption of RPCSEC_GSS.

- * When NFSv4.1 was developed, there appeared to be no reason to change the security decisions made for NFSv4.0. Although it might have been possible to change the status of AUTH_SYS in NFSv4.1, there was no reason to tie the session architecture to changes in security that many implementers would have been unwilling to make.

The only significant security-related extension was the provision made for state protection, which required the server to protect clients from one another, apart from the users on whose behalf requests were being made.

- * In the ten years since [[RFC5661](#)] was published, there was no way to address new security needs without creating an essentially new protocol, on the order of NFSv4.1, and there was little interest in doing that. Within the framework established by [[RFC8178](#)], there was provision for the addition of OPTIONAL features, while addressing lingering security issues for existing minor versions

did not fit in that framework.

With the above history in mind, we will look at why NFSv4 security needs are different today and take advantage of some options that were not available on earlier occasions in which NFSv4 security decisions were made.

- * We have the option of providing better security for AUTH_SYS use by implementing client host authentication.
- * We have the option of providing confidentiality and integrity even when RPCSEC_GSS is not used.

- * We have the option of providing confidentiality and integrity universally, without special configuration effort or interfering with performance by requiring non-offloadable encryption/decryption on the client and server.
- * We have the option of providing state protection (to prevent clients interfering with one another) without the special implementation work specified in [[RFC8881](#)].

These options have been made available by the work at the RPC layer described in [Section 7](#) while the specifics of having NFSv4 take advantage of these options are described in [Section 8](#).

Providing a new security approach for multiple existing protocols is potentially disruptive and due care will need to be taken to avoid damaging implementation incompatibilities. However, as will be discussed below, the existing situation, in which serious security inadequacies need to be addressed, requires that significant changes be made. It is our expectation that the situation will be resolved by the eventual publication of a standards-track document updating [[RFC7530](#)] and [[RFC8881](#)] and much of this document will concern itself with determining how the needed changes can resolve existing security issues without undue disruption.

[5.](#) Major Problems to Address

The problems to be addressed concern the way that security information has been conveyed in earlier specifications (discussed in [Section 5.1](#)) as well as two sets of substantive security weaknesses discussed in Sections [5.2](#) and [5.3](#).

Although the inadequacies in the presentation of security issues have contributed to prolongation of the substantive weaknesses and will be discussed in the latter two sections, there is no reason to believe that correction of the presentation problems, will, by itself, improve the security situation, since the substantive problems still need to be addressed. Nevertheless, correction of the presentation issues is necessary so that the proposed solutions to the substantive security issues receive the proper attention and analysis, both now in connection with the changes to be proposed, and also going forward, as needs change.

[5.1](#). Problems with Security Presentation/Organization

While attention has previously focused on the deficiencies of the Security Considerations sections (e.g. Lack of a threat analysis), this is not the only presentation issue that needs to be addressed.

Problems with the overall presentation of NFSv4 security, particularly the discussion of the security architecture will be dealt with in [Section 5.1.1](#).

Problems with the evaluation of NFSv4 security including the lack of a threat analysis and the contents of the Security considerations sections will be dealt with in [Section 5.1.2](#).

[5.1.1](#). Problems with Presentation of Security Architecture

The presentation of the NFSv4 security architecture (for both minor versions) focuses on the work done to make RPCSEC_GSS usable in view of the basic architectural decisions made in going from NFSv3 to NFSv4. These include the use of NFS4ERR_WRONGSEC and SECINFO to allow the server namespace to be carved into regions with the use of particular security flavors and services (associated with particular quality-of- protection values) required in each one.

Unfortunately, features added later in the NFSv4.0 development cycle were not integrated into the description of the security architecture. A list of such noteworthy features follows.

Another consequence of the approach taken to writing Security Considerations sections has been that these sections may have not appropriately highlighted the security implications of new features being added to the protocols.

- * The addition of callbacks to NFS was not accompanied by an explanation of how security for callbacks was to be dealt with, It was never made clear whether the existing mandate to provide support for RPCSEC_GSS applied to reverse-direction operation. The fact that there is no reverse-direction complement to NFS4ERR_WRONGSEC and SECINFO means that there is no way that the client, as a responder, could force use of another flavor if it did not provide support for RPCSEC_GSS authentication, in the role of a responder.

The fact that callbacks are normally issued by the server itself and not on behalf a specific user, means that the typical function of RPCSEC_GSS authentication is not relevant to the application. However, if RPCSEC_GSS is used in the forward direction, then the mutual authentication of client and server should be adequate to provide authentication of the server to the client, although the possibility that an attacker might inject a spurious callback to the reverse-direction request stream remains a concern.

In any case, the absence of mention of reverse-direction authentication in a description of the NFSv4 security approach is troubling. The lack of mention in the Security Considerations section is noteworthy because this creates a large set of requests and responses for which there is no available facility to mandate encryption, and that lack could be exploited by monitoring attacks.

- * The addition of pNFS to NFSv4.1 [[RFC5661](#)] required that there be a means of providing that the proper quality of protection by provided by the a pNFS data server implementing the files mapping

type. Given that SECINFO is namespace-oriented, it was necessary to define SECINFO_NONAME, implementable by the data server, to provide this functionality.

This security-related change, while explained adequately in [[RFC8881](#)], is not mentioned in the description of the security architecture or the Security Considerations section. As this feature does create new threats, it should have been mentioned in the Security Considerations section. Also, this change while compatible with the existing NFSv4 security architecture, does represent a considerable change to it. It seems to call for special mention in a standards-track document devoted to NFSv4 security issues. It makes sense to include it in an introductory section providing an overview of the NFSv4 security architecture.

- * The implementation of sessions in NFSv4.1 [[RFC5661](#)] created a requirement to protect clients from one another, even when they were making requests on behalf of the same user. Although adequate means of dealing with this issue were described in [[RFC5661](#)], they were OPTIONAL features and implementation has been very limited.

In this case, the additional exposure makes the absence of mention of the issue in the Security Considerations section troublesome, although its role in the limited implementation of the corresponding security features is hard to evaluate at this point. In any case, this was a major architectural change to the security architecture which requires more attention. Over the years, there has been considerable discussion on the NFSv4 mailing list of the lack of authentication of the client, as opposed to the client users on whose behalf it is acting, although no work has been undertaken to make this a basic element of NFSv4 security. Despite the delay, the issue now seems on its way to effective resolution, using the work done in [[I-D.ietf-nfsv4-rpc-tls](#)], as described in [Section 8.5](#).

As we have seen, the description of the security architecture of NFSv4 is quite limited and a new security document will have some gaps to fill.

[5.1.2.](#) Problems with Security Evaluation

The principal difficulty in the overall approach to security taken in the Security Considerations section of [\[RFC7530\]](#) and [\[RFC8881\]](#) concerns the absence of a threat analysis within these documents. The absence of such an analysis has meant that:

- * There was no occasion to determine whether the security improvements made relative to earlier versions of NFS were adequate to realize any particular reasonable interpretation of the goal of "secure use on the internet", which goal was never clearly defined.
- * In the absence of a clear goal or a means of testing whether that goal had been met, a choice of security facilities was made based on their likely availability. The prevalent assumption that security against network-based attacks was not important in most NFS environments made it difficult to consider improvements, or to forthrightly discuss existing security weaknesses and make plans to address them.

[5.2.](#) The Treatment of AUTH_SYS

Sections [7](#) and [8.2](#) of [\[RFC5531\]](#) introduce the RPC auth_flavor enumerator and a pair of opaque fields (the credential and the verifier fields). These fields carry material in each RPC message that can identify and authenticate the RPC client and the user who initiated the RPC transaction. The enumerator field determines the structure and content of the credential and verifier.

The simpler auth_flavors (e.g., AUTH_SYS) merely identify the requesting user and sending host. These flavors do not provide any cryptographic proof of identity. Therefore they do not provide true authentication of the conveyed user and host identities.

The RPCSEC_GSS credential and verifier can both identify and authenticate the requesting user. They can additionally provide message integrity and confidentiality.

[\[RFC3530\]](#), [\[RFC7530\]](#), and [\[RFC8881\]](#) state that implementation of RPCSEC_GSS is REQUIRED. However, they do not also mandate its use, exposing an opportunity for attackers to issue unauthenticated requests targeting NFSv4 servers in which RPCSEC_GSS is not the only form of authentication available.

Among the RPC auth_flavors that can present unauthenticated requests to a server, AUTH_SYS is the most significant of these for many reasons:

- * Because the AUTH_SYS flavor is simple to implement and administer, existing NFSv4 servers and client implementations make it almost universally available.
- * Unlike the AUTH_NONE flavor, whose name gives a fair warning that requests are unauthenticated, [\[RFC5531\]](#) describes AUTH_SYS as a means of authentication without addressing the question of what is being authenticated and by whom.
- * The substance of server and client implementations is not described in standards-track documents, leaving much to implementers' choice.
- * There is no clear and complete description of the security consequences of deploying AUTH_SYS RPC services.
- * Even [\[RFC5531\]](#) designates AUTH_SYS as "insecure, all NFSv4 minor versions continue to allow its use.

The everyday use of AUTH_SYS arises from NFS's early history and development as part of the RPC authentication function and NFSv2 [\[RFC1094\]](#) and NFSv3 [\[RFC1813\]](#). That approach, in which security was made the responsibility of large multi-user clients trusted by servers, might have made sense when it was initially developed, but has become less relevant to NFSv4 security needs over time.

The designers of the NFSv4 protocol made RPCSEC_GSS mandatory-to-implement. AUTH_SYS was left in place as an alternative without considering the corresponding consequences for security. [\[RFC5531\]](#), [\[RFC7530\]](#), and [\[RFC8881\]](#) mention several common practices that servers have typically used to determine whether AUTH_SYS requests should be accepted, but without explicit discussion of their obvious security weaknesses.

[5.2.1](#). Current AUTH_SYS Security Policies

Current NFSv4 client and server implementations commonly provide the following security policies. These policies attempt to improve the security offered by AUTH_SYS. Because these policies are not specified or discussed within standards-track documents, they have not been subject to threat analysis. There is an opportunity to describe and analyze security policies for AUTH_SYS and other authentication flavors used with NFSv4 in an NFSv4-wide standards-

Network Address-based Access Control

A typical server policy, which first appeared with implementations of earlier versions of NFS, is to limit the acceptance of AUTH_SYS requests to requests from known clients, as determined by the client's network address.

The Security Considerations section of [\[RFC7530\]](#) does not address the possibility of address spoofing, although it does state that this is "certainly not a safe model", most likely alluding to this possibility. The text does not follow up to call into question the use of AUTH_SYS as an OPTIONAL security flavor (stated earlier in the form "MAY be implemented"). Instead, it is used as justification for the mandatory nature of RPCSEC_GSS implementation; the lack of security when only AUTH_SYS is in use is not mentioned further.

Network Port-based Access Control

Another frequently-used server-side security policy is to restrict the use of AUTH_SYS based on the associated client source port, assuming that this ensures that the client's kernel (or a privileged user space agent) has vetted the request. That assumption has, over time, become dubious.

[Appendix A of \[RFC5531\]](#) refers to this practice as follows:

"It should be noted that use of this flavor of authentication does not guarantee any security for the users or providers of a service, in itself. The authentication provided by this scheme can be considered legitimate only when applications using this scheme and the network can be secured externally, and privileged transport addresses are used for the communicating end-points (an example of this is the use of privileged TCP/UDP ports in UNIX systems -- note that not all systems enforce privileged transport address mechanisms)."

Identity Squashing

NFS servers can implement a security policy that executes all AUTH_SYS requests as a suitably unprivileged user ID, such as the anonymous user. An alternate policy treats only requests from a particular privileged user ID (e.g., root) this way.

Squashing does not prevent an attacker from masquerading as another user but only limits the range of user identities that can be assumed without difficulty.

[5.2.2.](#) Working Group Actions

The Security Considerations section of [\[RFC5531\]](#) states that AUTH_SYS is "known to be insecure" and further states "AUTH_SYS SHOULD NOT be used for services that permit clients to modify data." Nevertheless, AUTH_SYS has been commonly used by NFSv4 clients modifying data since NFSv4 was first implemented, while the security consequences have never been addressed.

Given the security difficulties that use of AUTH_SYS gives rise to, the NFS community faces a choice between a few alternatives:

- * Take steps to eliminate or otherwise minimize the use of AUTH_SYS as a valid authentication flavor, either for NFSv4 as a whole or for minor versions based on NFSv4.1, whose description was published as a Proposed Standard after the publication of [\[RFC5531\]](#), which stated that AUTH_SYS was "known to be insecure".

Given the many clients using AUTH_SYS, such steps might take the form of recommendations, with the difficulty of switching authentication models considered a valid reason to continue to use AUTH_SYS as long as one is aware of the security consequences.

Regardless of the working group's choice of normative terms, it is crucial that new NFSv4 standards clearly explain the security consequences of using AUTH_SYS.

- * Revise AUTH_SYS to enable secure implementation of an authentication model in which authenticated client hosts determine, within appropriate limits, the user IDs used for each request sent.

To do this, servers would authenticate client hosts issuing

AUTH_SYS requests, taking advantage of the authentication provided by [[I-D.ietf-nfsv4-rpc-tls](#)] as described in [Section 7.2](#). A more detailed discussion of issues to be addressed in establishing the rules for this form of authentication appears in [Section 8.4.6](#).

- * Continue to pursue alternative RPC security mechanisms that are simple to deploy and not costly to run. One such mechanism is RPC-over-TLS, which should be available in time to include in security documents published together with rfc5661bis. However, we can easily imagine mechanisms based on other open federated authentication standards that can be made available within NFSv4 via subsequent extensions.

The nfsv4 working group should consider all of these alternatives.

The first of these is unlikely to be feasible; otherwise, the NFS community might have already attempted such a step. The use of AUTH_SYS has continued even though alternatives are available, and there is no reason to expect that the use of the words "SHOULD NOT" or "MUST NOT" would prevent implementers from continuing to support it or administrators depending on such support, despite the negative effect on the security of NFSv4 implementations

Because of the current predominant role of AUTH_SYS in existing NFSv4 implementations, providing better security while remaining within the AUTH_SYS framework will also be difficult because of the issues discussed earlier in [Section 6.2](#). Nevertheless, we feel it needs to be attempted as a new security framework cannot address AUTH_SYS without clearly discussing its security weaknesses. Once that is done, we need to take advantage of strong client host authentication to provide a modicum of security while allowing administrators to delegate credential-checking to trusted clients, as they are now accustomed to doing.

This approach is far from ideal. It requires a trust relationship between clients and servers. Adopting it provides a reasonable approach to support NFSv4 in the near-term while allowing the development of a better alternative (e.g., a new authentication flavor) to proceed in parallel. Concerning our immediate need for better NFSv4 security, the obvious alternatives might not be realizable:

- * Given the weaknesses of AUTH_SYS and the need to present a threat analysis, we cannot commend AUTH_SYS without client host authentication as OPTIONAL to use, either explicitly or by implication.
- * Recommending that AUTH_SYS without client host authentication not be used, with RPCSEC_GSS as the only alternative, will not be effective and would limit adoption of a better, although far-from-ideal, alternative.

Therefore, the authors propose that upcoming NFSv4 standards-track documents mandate or recommend that, when AUTH_SYS is in use, a form of strong host authentication is always deployed along with it. Ultimately this is a matter for working group decision, however. For a further discussion of related issues, see [Section 8.4.3](#). Although the rest of this document assumes the authors' preference is selected, [Section 8.4.3](#) explains the changes to an eventual standards-track document required if the working group concludes that different choices are necessary.

[5.3](#). Problems with Confidentiality

Confidentiality is currently provided within NFSv4 by the use of RPCSEC_GSS, with the server charged with enforcing any administrator-specified needs to use these facilities on appropriate portions of the server namespace. Requirements for the use of integrity protection are handled similarly. This approach is capable of providing confidentiality when accessing certain directories or file systems, assuming that files that require such protection can be isolated in certain regions of the server namespace.

An important difficulty with regard to this approach to confidentiality is that there is significant non-encrypted data sent on NFSv4 connections which can allow extensive data to be gathered on the part of those engaged in monitoring attacks

- * Certain parts of RPC requests and response are not encrypted and can be the basis of traffic analysis. Fortunately, the structure of NFSv4 requests limits the data exposed since the requested

operation is always COMPOUND (or CB_COMPOUND). Nevertheless, the size of requests and information determinable from those allows patterns of reading and writing data by specific clients to be inferred.

- * Because the focus of this approach is on areas of the server namespace, there is no way to force use of encryption on requests used to set up connections and sessions.
- * Similarly, there is no provision for negotiating/enforcing the use of integrity or encryption on reverse-direction requests.

Despite the availability of encryption in NFSv4, it is very rarely used, which makes its formal sufficiency essentially irrelevant. In understanding why confidentiality is not more generally used, we need look at the issues below in order to understand how to address the problem going forward.

- * The cost is such that its use has a noticeable effect on performance, given that the design (by requiring different keys for different users) makes it difficult or impossible for the work of encryption to be offloaded.

It is likely that, given increasing network speeds, this factor is more important today than it was when this approach was settled upon.

- * In many environments it might not be possible to isolate files needing such protection in a way consistent with the way file protections are normally managed. As a result, encryption would be present for all files or for none, with none being chosen most often because of performance concerns.
- * Implementation of confidentiality requirements may face difficulties due to uncertainty about whether client encryption support is available. For example, [[RFC7530](#)] specifies that privacy support is required (although only for krb5), while [[RFC8881](#)] says only that clients "SHOULD" support privacy. The document give no guidance regarding what might be considered valid

reasons to not support privacy, leaving servers without any reliable way of demanding confidentiality on certain portions of the server namespace.

- * The concurrent use of AUTH_SYS might have a similar inhibiting effect. While it is possible, within the protocol, to force a transition to use of RPCSEC_GSS with privacy, RPCSEC GSS access might not be possible in many cases (e. g. the proper id mapping facilities might not have been configured).
- * The current Security Considerations sections for the NFSv4 protocols may have contributed to the continuation of the problem by not drawing sufficient attention to the security issues involved in allowing access without encryption.

Looking at the relevant Security Considerations sections, we find the following:

- * Although the cost of performing encryption is mentioned as a reason not to perform it, there is no discussion of the security consequences of not performing it.
- * Similarly, there is no discussion of the use of integrity services in preventing modification of user data in flight and the data corruption that attackers could cause when these services are not used.
- * There are a number of references to integrity protection for various sort of operations but none connected with the protection of user data, possibly suggesting to the reader that this issue is not important. Ironically, while the operations used to obtain security requirements for a given portion of the namespace are among these mentioned, an attacker could not use these to obtain transmission in the clear but would instead cause denial-of-service, as would happen the bogus security parameters were rejected.

All of the issues discussed above may have contributed to the lack of use of encryption with NFSv4, but the issues of poor performance due to the a non-offloadable nature of encryption appears to have had a critical role in creating an unsatisfactory situation for a protocol where high performance is often of critical importance.

Once the choice was made to limit encryption to specific file systems or directories, there was commonly a perceived need to avoid the cost of encryption and the difficulties that approach to the matter led to were exacerbated by the lack within the Security Considerations sections of information about the damage thus done. Because there was insufficient attention to these issues, effective action to address them was delayed, until the problem could no longer be ignored.

Given these weaknesses with regard to encryption, those needing to use NFSv4 outside local area networks often implemented NFSv4 over secure tunnels. As this work proceeded, it served as an inspiration for the work done in [[I-D.ietf-nfsv4-rpc-tls](#)] that provided the opportunity to provide encryption uniformly and potentially with high performance. An overview of steps necessary to address these issues appears in [Section 7.1](#).

[5.4](#). File Access Control

Originally, the NFS protocol provided only a basic form of access control in the form of permission bits (also known as mode bits).

NFSv4 introduced a rich access control list mechanism which supplements mode bit-based access control [[RFC7530](#)]. This is referred to as a Discretionary Access Control (DAC) mechanism, where access to the file's content is at the discretion of the file's owner and is based on identity of the requesting user. A more complete definition of DAC is available in [Section 4 of \[RFC4949\]](#).

The NFSv4.2 protocol provides a basis for supporting Mandatory Access Control (MAC) in the form of Security Labels, as described in [Section 9 of \[RFC7862\]](#). Mandatory Access Control is generally mandated from a central authority that constrains access to file content based on broad trust categories. Because this usage overloads the abbreviation MAC, which can also refer to Message Authentication Codes and Media Access Control, we avoid using the abbreviation unless the meaning is clear. A more detailed definition of MAC is available in [Section 4 of \[RFC4949\]](#).

Both of these file access authorization mechanisms need a threat analysis and a discussion of exposures due to missing access control facilities. However, without documentation of access semantics in the case of Security Labels, any threat analysis might not be complete.

[5.4.1.](#) File Content Integrity and Provenance

Ever since RPC has supported an integrity-protected mode of transport (via RPCSEC_GSS integrity pseudoflavors), NFS has been able to provide a strong guarantee that NFS messages in transit on a network are immutable.

Recent work in block storage has extended the immutability of file content from application to storage and back (for example, SCSI commands related to Protection Information, as defined by the T10 technical committee). This provides a guarantee that what an application reads at time $T+1$ is what an application wrote at time T . The NFS protocol can and should enable this stronger form of integrity guarantee.

Provenance is a deeper guarantee. It additionally identifies the author of a file's content, and guarantees that content is exactly what the author originally provided. An NFS client might use verifiable provenance to gate access to a critical executable or other resources contained in files stored on an NFS server.

Lastly, some systems provide a policy-based access control mechanism based on whether file data content passes various integrity checks.

These capabilities require the storage and transit of additional metadata associated with each file, which is currently not a part of the NFSv4 protocol. Security Considerations sections should explain how data is put at risk when these capabilities are not present.

[6.](#) Framework for Correcting Problems

This section addresses the fundamental issues of the organization and presentation of a new security framework. The complementary issues involved in making substantive improvements in NFSv4 security will be dealt with in [Section 7](#).

[6.1](#). Correcting Problems with Regard to Threat Analyses

Of prime importance is the inclusion of appropriate threat analyses, even apart from the requirement (in [BCP 72](#) [[RFC3552](#)]) that such analyses appear in Security Considerations sections. As we have already seen, without such an analysis, there is no way to determine whether any security changes adopted are adequate to address NFSv4's security difficulties. In the case of NFSv4, because of its complicated history and the conflict between the status of AUTH_SYS as specified by the NFSv4 specification documents ([[RFC7530](#)], [[RFC8881](#)]) and that in [[RFC5531](#)], there might be multiple analyses, located in different places:

- * Because the security needs of the various minor versions are so similar, the bulk of the analysis could appear in an NFSv4-wide standards-track document updating [[RFC7530](#)] and [[RFC8881](#)].
- * The case of a threat analysis for AUTH_SYS raises a special set of issues. Currently, AUTH_SYS is discussed and declared "insecure" within [[RFC5531](#)] with no threat analysis appearing within the document. A new threat analysis could appear in the NFSv4-wide security document referred to above but the fact that AUTH_SYS might be considered as part of RPC rather than of NFSv4, might dictate otherwise. Such an analysis might appear in a document updating [[RFC5531](#)], or in a new document devoted to that specific purpose. Whichever choice is made, there will be a need for a document updating [[RFC5531](#)].

The appropriate location of a threat analysis for AUTH_SYS as currently used depends on the question of whether to consider AUTH_SYS as part of RPC, as it formally is, or to transfer responsibility to NFSv4, since it was deprecated as insecure in [[RFC5531](#)], while at the same time embedded as a heavily used optional feature for all minor versions of NFSv4, including a new protocol, NFSv4.1 published as a Proposed Standard subsequently, in [[RFC5661](#)].

Also relevant to the question of the appropriate location for this threat analysis is the question of whether there might be a need for a revised treatment of AUTH_SYS including client host authentication. The possible existence of such a revised treatment, together with issues related to a threat analysis necessary for its inclusion is discussed in [Section 8.4.4](#).

[6.2.](#) Correcting Problems with Regard to Use of Normative Terms

Another important issue concerns the potential need for adjustment of inappropriate use of terms defined by [[RFC2119](#)], particularly any suggestion that use of AUTH_SYS may be validly used without important security consequences, as suggested by the current understanding that its use is optional.

However, as desirable as such clarifications might be, it is necessary that we be realistic about what such changes can accomplish, even if the particular issues cited in [Section 2.2](#) are properly attended to. Documents can be written to define certain implementations as non-conformant, but the ability of any such designation to affect implementer behavior is strongly affected by circumstances, particularly when the term is a hard one to pin down such as "SHOULD" or "SHOULD NOT".

The ability of such term choices to affect implementer behavior is at its maximum when a new protocol is defined. In this case, the specification document formalizes a contract between implementations, so that the consequence of not following these terms is a lack of interoperability rather than the fact that one's implementation may be, formally speaking, non-conformant. In the case of a protocol such as NFSv4, intended to build and extend the work of earlier NFS versions, the ability of the IETF to affect implementer choices is significantly reduced, as interoperability with the existing protocol and the existing infrastructure to support it will be of greater importance to implementers. Further, it needs to be recognized that when these terms are used to constrain security-related behavior rather than interoperability per se, their effect on implementer choice is likely to be similarly reduced.

Although [[RFC2119](#)] provides definitions of "SHOULD" and "SHOULD NOT", it is difficult to determine the proper interpretation unless it is clear what the "valid reasons" that might supersede the

recommendation or the expected consequences of doing so. For this reason, when these terms are used in proposed text, we will make such matters explicit, where they are not otherwise clear, as suggested by [Section 7 of \[RFC2119\]](#).

[7.](#) Opportunities for Improvement Provided by Recent Work within the RPC Layer

The work done to provide TLS support for RPC (in [\[I-D.ietf-nfsv4-rpc-tls\]](#)) presents an important opportunity to address the substantive security problems described in Sections [5.2](#) and [5.3](#). While there will be a need to specify appropriate policies for its use by NFSv4 as a ULP (see Sections [8.3.2](#) and [8.4.6](#) for details), the following opportunities present themselves and are likely to be taken advantage of:

- * The use of transport-level encryption is a good way of dealing with the inadequacies discussed in [Section 5.3](#). The use of a single key per connection makes offloaded implementations possible allowing use for all requests including those currently not encrypted. This is discussed in more detail in [Section 7.1](#)
- * Authentication of the client host, as provided by the authentication material presented at connection initialization might be used to authenticate the client host, to avoid acceptance of AUTH_SYS requests from unauthenticated clients. This is discussed in more detail in [Section 7.2](#)

Although [\[I-D.ietf-nfsv4-rpc-tls\]](#) does not provide support for RDMA transports, parallel services to provide encryption and authentication are expected to be available, as discussed in [Section 7.3](#).

7.1. Opportunities for Improvement in Encryption

[I-D.ietf-nfsv4-rpc-tls] provides the ability to encrypt all traffic on the connection used between an NFSv4 client and server. This requires that both the client and server provide support for RPC-over-TLS. Appropriate requirements/recommendations are discussed in [Section 8.3.2](#).

Because this facility is based on an opportunistic use of TLS, there is a need for ULP-defined policies to deal with situations in which the server rejected the request for a TLS connection. These matters are discussed in [Section 8.3.2](#) as well, as is the policy that servers are to adopt when dealing with a connection on which RPC-over-TLS is not requested.

Because this new form of encryption is being added to an existing protocol, all of these policies are more complicated than would be the case for a new protocol, since there might be a need to accommodate earlier implementations, which might not have had time to be enhanced, while providing the ability to take advantage of whatever confidentiality support that might be present for a given client and server.

7.2. Opportunities for Improvement in Authentication

When establishing a connection, as provided for by [\[I-D.ietf-nfsv4-rpc-tls\]](#), authentication material is provided by the client and could be used to authenticate the client host to the server.

By using this authentication material to authenticate the client host, it would be possible to correct some of the issues discussed in [Section 5.2](#). This would involve clearly specifying how that material is to be used, in contrast to the situation for the existing AUTH_SYS, for which current documentation (in [Appendix A of \[RFC5531\]](#)) is quite limited. Any such respecification would need to

be acceptable to those used to using AUTH_SYS while avoiding the security issues that make its further use, in the old form, inadvisable.

This could allow the following improvements:

- * This would avoid dependence on the use of source request IP address, which is insecure, given the possibility of address spoofing.
- * Authentication material could be separately defined for user and kernel-mode clients to avoid dependence on the outdated privileged port mechanism.
- * More options could be provided to allow limitation of the user ids upon behalf of which requests are made, to address the limitations of "root squashing".

The details of how the authentication material could be used are discussed [Section 8.4.6](#).

[7.3](#). Opportunities for Improvement when Using RDMA Transports

As noted above, [[I-D.ietf-nfsv4-rpc-tls](#)]) is not supported on RDMA transports, making it necessary to provide appropriate support for encryption and client host authentication in other ways.

- * Often RNICs used to implement RDMA transports will often implement encryption to secure inter-RNIC traffic. When, as is often the case, the inter-RNIC protocols are not standardized, the judgment as to the adequacy of the encryption is out-of-scope, from our point of view. However, the client and server could be configured to take advantage of this encryption, when it is judged appropriate to do so by those responsible. In addition, the transport characteristics mechanism defined in [[I-D.ietf-nfsv4-rpcrdma-version-two](#)] could allow those configuring the client and server to make independent judgments on encryption adequacy, with RPCSEC_GSS integrity and encryption only superseded, when both agree. (See [Section 8.3.2](#) for details).
- * Authentication material to support client host authentication can

be provided by using the transport properties mechanism provided for in [[I-D.ietf-nfsv4-rpcrdma-version-two](#)].

It should be noted that our need to normatively reference [[I-D.ietf-nfsv4-rpcrdma-version-two](#)] would affect the schedule of a standards-track document dealing with NFSv4 security. Given that the current milestone for requesting publication is December 2020, this not expected to pose an obstacle.

[8.](#) Issues that Need to be Addressed

These sections discuss the issues that need to be addressed by new standards-track documents including both issues of the presentation/evaluation of NFSv4 (discussed in [Section 5.1](#)) and the substantive security weaknesses discussed in [Sections 5.2](#) and [5.3](#). For each issue there is a section providing background followed by information suggesting how the issue would be addressed or issues that would need to be resolved before the issue could be addressed.

[8.1.](#) Threat Analysis Goals

[8.1.1.](#) Background

For reasons that remain unclear, none of the specification documents for the existing NFSv4 minor versions contain a threat analysis. As a result, for the reasons discussed in [Section 5.1](#), we need to provide an appropriate threat analysis for all NFSv4 minor versions. Because of the high degree of overlap between pairs of NFSv4 minor versions, most of the analysis will be common to all versions. However, there are some areas where features in later minor versions have significant security implications:

- * In NFSv4.1 [[RFC5661](#)], SECINFO_NO_NAME was added, in large part because it would be necessary to allow the data server, when the pNFS file mapping type is used, to communicate to the client that either integrity or encryption would be required when access to file data was provided through the data server (as opposed to the metadata server).

The use of TLS-based encryption, which we expect to be recommended (see [Section 8.3.2](#), may limit the need for this feature, but it will remain REQUIRED, when file-based pNFS is implemented.

- * NFSv4.1 [[RFC5661](#)] facilities were added to protect locking and session-related state from modification by other servers (i.e. SP4_MACHCRED and SP4_SSV).

These features have had very limited implementation work, so it would be desirable to provide alternative means to achieve the goal, as described in [Section 8.5](#).

While the bulk of the threat analysis would be minor-version-independent, differences between minor version will have to be noted, as will differences that reflect uses of the facilities discussed in [Section 7](#).

Beyond the lack of NFsv4 threat analyses, there is a further problem in that there is no threat analysis dealing with the AUTH_SYS case in the existing RPC specification [[RFC5531](#)]. In this case, the gap is easier to understand since RPC deals with security on a per-flavor basis and declares AUTH_SYS as "insecure". It may well be that the authors, the working group and the IESG thought this was adequate, and it would have been if this had resulted in its not being used subsequently. As things happened however, AUTH_SYS continued to be used extensively, including for NFsv4, so that there still remains a gap in this regard.

The following facts are relevant to the continued use of AUTH_SYS despite its security problems:

- * It was stated that AUTH_SYS "SHOULD NOT" be used for services capable of modifying data. Given that its further use was still anticipated, in some circumstances, a security analysis would still be required. In addition, use within the purview of the "SHOULD NOT" would make such analysis even more important, since that recommendation would allow use presumably based on balancing of possible benefits and corresponding problems. Without an understanding of the security problems (i.e. the substance behind the designation of AUTH_SYS as insecure), such balancing would not be possible.

- * Despite the recommendation that AUTH_SYS not be used for services, such as NFSv4, that are capable of modifying data, NFSv4 was specified allowing such use with very limited attention to the security issues that uses of AUTH_SYS gives rise to. As a result, there remains a needs for an appropriate threat analysis including for AUTH_SYS, since no matter what the working group decides to do in this area, use of AUTH_SYS, in connection with all NFS protocols (and possible other ONCRPC protocols) will continue for some time.
- * The reason stated for considering AUTH_SYS insecure, the lack of a verifier, while worthwhile to note, does not touch the substance of the problem, that unauthenticated requests are accepted and potentially acted upon. The difficulty is further compounded by the fact that the client validation checks made by actual implementations are not fully described within RFC (They are described above in [Section 5.2](#)).

[8.1.2](#). Issues to be Addressed

It is expected that a threat analysis dealing with all NFSv4 minor will be dealt with in new standards-track document dealing with NFSv4 security. Because this document will address security for all minor versions, it will update [[RFC7530](#)], [[RFC8881](#)], and [[RFC7862](#)].

There will, in addition, be a need for a threat analysis dealing with AUTH_SYS, which might or might not appear in the NFSv4-wide security document for reasons explained in [Section 6.1](#). The question of where this best be done is discussed in [Section 8.4.4](#).

[8.2](#). NFSv4 Extension Policies

[8.2.1](#). Background

The basic framework for the extension of NFSv4 was established by [[RFC8178](#)]. Given that the anticipated standards-track document dealing with NFSv4 security will, in a sense, extend NFSv4, there might be some uncertainty about whether the changes anticipated are in line with that extension framework and whether there might be a need for that document to update [[RFC8178](#)].

The following questions need to be addressed:

- * Whether the changes in policies anticipated here, with regard to encryption and use of AUTH_SYS, are consistent with [[RFC8178](#)].
- * How the use of new security-related facilities provided at the RPC level relates to the extension framework established by [[RFC8178](#)].

Internet-Draft

NFSv4 Security Needs

January 2021

- * How further extensions might relate to use of such new RPC-level facilities.
- * How to deal with and document extensions that provide new security-related features.

8.2.2. Addressing Extension Issues

We anticipate that the material would be in a section entitled, "NFSv4 Security Changes and the Existing NFSv4 Extension Model"

The following introductory material seems appropriate:

- * The security-related changes recommended in this document are outside the scope of the extension model presented in [\[RFC8178\]](#). However, they do not contradict it and there is no need for this document to update that one.
- * Normally, changes requiring coordinated work on the client and server are signaled using a new minor version. In that case, the new minor version is essentially treated as a new protocol and there is no need to update [\[RFC8178\]](#) or specifications for existing minor versions.

The following paragraphs discuss the core issues:

- * That approach is not possible in this case since the basic need is to change the handling of existing minor versions to improve security. This create the possibility of interoperability issues since there is neither a new minor version nor a means of distinguishing a consistent set of OPTIONAL features supported by the server and known to the client.
- * In order to limit the possibility of interoperability issues, the security changes are made in the form of recommendations, but, because there exist implementations built before these recommendations were in effect, clients and server have to be aware of the possibility that these recommendations might not be followed. As a result, the effect of these recommendations, made to limit security issues, is, from an interoperability point of view, similar to those that might arise from OPTIONAL features. With regard to the NFSv4 extension framework,

- The use of the new RPC-level facilities does not raise any extension issues, since the existing specification do not limit the transport used, except to require reliable and in-order delivery. For this reason, these facilities can be used without updating the existing minor version specifications.

Use of these facilities does not raise additional interoperability issues since [[I-D.ietf-nfsv4-rpc-tls](#)], by using the opportunistic TLS model, provided for interoperability between those that do and do not support these extensions.

- The policies to use these facilities, made the responsibility of the ULP by [[I-D.ietf-nfsv4-rpc-tls](#)] are significant additions to existing minor versions, requiring that the new standards-track document dealing with NFSv4 security update [[RFC7530](#)] and [[RFC8881](#)].

These changes do not raise interoperability issues since the policies apply to all uses of these transport-level facilities for NFSV4. That addresses the case in which client and server both provide support while other cases are dealt with as indicated in the previous bullet.

- The new recommendations regarding use of existing features (e.g. encryption, AUTH_SYS) often contradict existing guidance and so also require that new standards-track document dealing with NFSv4 security update [[RFC7530](#)] and [[RFC8881](#)].

These change do potentially raise interoperability issues which need to be dealt with by giving sufficient scope, within the framework of new recommendations, to accommodate existing behavior. As a result, from an interoperability standpoint, these changes are compatible with the previous designation of OPTIONAL use, even though the security consequences make it necessary that the use of these previously OPTIONAL facilities (e.g. transferring data in the clear, use of AUTH_SYS without client host authentication) be something to be warned against.

[8.3.](#) TLS Encryption Policies

[8.3.1.](#) Background

As discussed in [Section 5.3](#), NFSv4 as currently defined, has serious problems providing the appropriate level of confidentiality for its transmissions. Providing encryption at the transport layer, whether as described in [Section 7.1](#) or otherwise, can be expected to resolve this issue by providing encryption in a potentially offloadable way, making use of RPC-level privacy and integrity services unnecessary.

If we were defining a new protocol, such transport-level encryption could be REQUIRED without difficulty. However, given that we are upgrading the security for an existing protocol, it might not be possible to designate this encryption as "REQUIRED", since this might

give rise to extensive interoperability problems between new and old implementations. In any case, existing implementers of the existing protocol would not have the same interest in specification compliance as in the corresponding situation with a new protocol, leading to a situation in which one group of implementations essentially ignored the issue of specification-compliance with regard to the latest protocol specifications.

We need to be aware that,

- * In implementation of a new protocol, any requirements on the client the server such as we have been discussing act synergistically to marginalize non-compliant implementations. Since non-compliant server will not interoperate with a compliant client and vice versa.
- * The corresponding case of a requirement providing for a security upgrade, as in this case, is different. Implementers, and those charged with allocating resources to their work will naturally focus on improvement/maintenance of the working protocol and limit the effort devoted to being the first to do an implementation that would not be used until others do their part. In such cases, if there are parties who are not convinced of the necessity for the upgrade, it takes strong input from users/customers to ensure that the needed upgrade is given sufficient attention.
- * In some cases, a new protocol containing an important security upgrade can result in a situation more like the latter case than the former.

As we consider how to discuss the need for encryption in [Section 8.3.2](#), we will be mindful of the above. Since it not possible to mandate the use of encryption, we will make it RECOMMENDED. However, in discussing the valid reasons that encryption might not be provided, we will try to restrict them to the degree we can reasonably do so. Further, we will try to clearly state the security consequences which implementers and those configuring clients and server need to be aware of when choosing to not provide the RECOMMENDED encryption.

[8.3.2](#). Issues to Address

There is a need to recommend the implementation and use of appropriate encryption by the server and client, as is done, for example, by the following paragraphs.

- * Data sent over connections used to connect an NFSv4 client and server SHOULD be encrypted, whether this uses TLS encryption, as described in [[I-D.ietf-nfsv4-rpc-tls](#)] (to be used for TCP connections) or another form of encryption. For example, [[I-D.ietf-nfsv4-rpc-tls](#)] does not address encryption for RPC-over-RDMA ([[RFC8166](#)], [[I-D.ietf-nfsv4-rpcrdma-version-two](#)]) but RNICs used to implement RDMA transports will often implement encryption to secure inter-RNIC traffic.
- * Valid reasons for not implementing the encryption recommendation include insufficient time and resources to code, test, and deploy an appropriate implementation. Implementers and those deploying NFSv4 servers and clients and those deploying NFSv4 services need to remain aware of the consequences of not providing encryption. These consequences are of concern regardless of the validity of the reasons for not doing so, which may be expected to become more restricted as time goes on.
- * Given that it is necessary for both the client and the server to provide implementations capable of transport-level encryption, those deploying NFSv4 implementations need to be aware of possible steps to provide remediation, the difficulties involved in

providing it and the consequences of not doing do.

- * The lack of encryption for the connection as a whole can be addressed by encryption of individual requests using the facilities within RPCSEC_GSS specified in [[RFC7530](#)] and [[RFC8881](#)] as providing "privacy". When doing so, the following complicating issues need to be taken account of:
 - Such facilities are not likely to be offloadable and often will reduce performance dramatically.
 - Support for confidentiality via encryption is not required by [[RFC8881](#)] but only recommended, saying it "SHOULD be supported". Unfortunately, the document does not specify what might be valid reasons not to provide such support. Also, although [[RFC2119](#)] states the one choosing not to follow such a recommendation should be aware of the consequences, given the general reticence of NFSv4 specification Security Considerations sections about such matters, it is not clear that implementers have been able to properly consider the issue in the past. In any case, implementations without such support probably exist, making encryption unavailable for some existing client-server pairs.

- * When encryption is not provided, or, because of its effect on performance, not used, all of the user data read or written is transmitted in the clear, subject to interception or modification in flight. While this is clearly unacceptable for use on the internet, it should not be assumed that it is acceptable in more restricted environments.
- * Where this problem exists, it will often be necessary to restrict such traffic to specific network segments, and make it impossible, via administrative measures, to limit access to such network segments or monitor them extensively.

Although, as indicated in [[I-D.ietf-nfsv4-rpc-tls](#)], that there should be no need to negotiate security flavors to be used to provide integrity and confidentiality, the exact effect on existing clients

and servers needs to be made clearer.

The text below is a suggested way of doing this.

- * Although the pseudo-flavors created to indicate use of integrity or encryption continue to be a part of NFSv4, when transport-level encryption is provided, their use is unnecessary, although server responses to SECINFO and SECINFO_NONAME may continue to include them.
- * When TLS is used to provide encryption, as specified in [[I-D.ietf-nfsv4-rpc-tls](#)], the client and server will, as indicated by that document, be aware that encryption, is present. In the case of RPC-over-RDMA, when the RNICs provide encryption transparently, the server and client have the opportunity to each make their own judgment about the adequacy of the encryption provided and communicate this to their peer as a transport property as provided for by [[I-D.ietf-nfsv4-rpcrdma-version-two](#)]). In either case, the client and server will only act, as described below, on knowledge about the existence of encryption shared by both parties:
 - The server treats all requests as being made requesting encryption, even if a different pseudo-flavor is used. As a result, the server will never return, NFS4ERR_WRONGSEC because of a quality-of-protection issue, despite a possible mismatch between the pseudo-flavor specified in SECINFO response and the one actually used.

The server will still return NFS4ERR_WRONGSEC when an inappropriate security flavor or oid is used and clients need to be prepared to use SECINFO to determine the security to be used.

- The client is free to ignore the quality-of-protection issues in a SECINFO response, making use of authentication-only variant of RPCSEC_GSS most efficient.
- Although use of integrity or privacy pseudo-flavors is unnecessary on an encrypted channel, the server still needs to accept such requests.

- Within NFSv4.1 [[RFC8881](#)], support for SECINFO_NONAME is still REQUIRED when pNFS is supported, even if the server only accepts connection on which encryption is used, making generation of NFS4ERR_WRONGSEC on a connection to the metadata server impossible.

[8.4.](#) Handling of AUTH_SYS

[8.4.1.](#) Historical Background

During the development of NFSv2 [[RFC1094](#)], little attention was directed to network security and no attention was directed to the possibility that a machine on the network might represent itself as a client, with the ability to make requests on behalf of non-existent client processes it identified by user id. During this time, NFS clients were multi-user machines and it made sense to many for the client kernel to have the same responsibilities to verify user credentials and keep track of process user ids as it had successfully been doing with local file systems. This was the origin of AUTH_UNIX (later renamed AUTH_SYS) which was based on a model of co-operating UNIX kernels and provided no protection against an attacker with kernel privileges. Despite the later name change, actual implementations were strongly focused on UNIX, deriving the necessary implementation requirements from shared code and from informal inter-developer communications, rather than standards documents. As a result, as discussed in [Section 5.2](#), when it was necessary to discuss the security of AUTH_SYS in connection with RPC (in [[RFC5531](#)]), there was insufficient information on which to base a threat analysis, although it was correctly concluded that the result of use was a lack of security.

With regard to the following aspects of AUTH_SYS, this UNIX orientation was significant. In many cases as the computing environment changed, it was not possible to change the details to keep up with evolving needs.

- * As the assumption of kernel trustworthiness became increasingly unviable as a basis for security, no action was taken to address the issue. While the check for a privileged port retained some residual value, it was used to exclude requests issued by non-

kernel clients as inherently untrustworthy. The more significant

and difficult problem of distinguishing trustworthy and untrustworthy kernel-level access was never addressed, leaving AUTH_SYS of little value as a means of authentication, despite its continuing use.

- * AUTH_SYS was focused on the use of the non-hierarchical 32-bit user id space typical of UNIX. Although later attempts were made to generalize this within NFSv4, the fact that this approach is embedded in many server filesystems meant that this aspect of AUTH_SYS, does not seriously obstruct access to such servers.
- * The particular user id of zero (denoting "root" in UNIX) was often treated specially, by treating the request as issued by the predefined user id "nobody". While this was undoubtedly helpful when first defined, it is now the case that other non-root users might have significant privileges and that the ability of an attacker to assume any chosen user id allows much important information to be obtained and corrupted.

Later, with the development of NFSv3 [[RFC1813](#)], some of these assumptions started to seem less justifiable and alternate authentication models were developed. However, because AUTH_SYS was so suited to the Unix environment, making NFS administration so similar to local file system administration, its use remained predominant despite its obvious weaknesses from the security standpoint.

[8.4.2](#). Background for Existing AUTH_SYS in NFSv4

When NFSv4 was first defined (in [[RFC3530](#)]), RPCSEC_GSS was added as a mandatory-to-implement means of authentication which was optional to use, presumably under the assumption that other optional-to-implement authentication flavors, such as AUTH_SYS would be used by some clients

The continued use of AUTH_SYS is troubling and hard to explain if one is focused on network security, but the following possibilities are worthy of note:

- * It might well have been felt that the problems with AUTH_SYS, although troubling, did not really apply to the most common use case for NFS, on local area networks.

- * The familiarity of AUTH_SYS to Unix administrators, and general consistency with existing UNIX methods of system administration may have made it difficult to contemplate its abrupt removal in NFSv4.0, with the expectation that doing so might have substantially impeded NFSv4 adoption.

Regardless of why this happened, the important fact is that it did happen, despite the serious security problems that it gave rise to. Later subsections of [Section 8.4](#) will discuss ways to address the resulting situation.

[8.4.3](#). Core Issue to Resolve for AUTH_SYS

Given the serious security weaknesses described in [Section 5.2](#), there is clearly a need to discourage its further use in its existing form, since the difficulties with it will not be eliminated by the adoption of encryption of NFSv4 connections alone

There are two basic forms that such an effort might take:

- * Seek to discourage the use of AUTH_SYS in the expectation that it will be eventually replaced by RPCSEC_GSS or a new security flavor.
- * Seek to eliminate or mitigate its security problems and to discourage the use AUTH_SYS variants that have not addressed the underlying problem, in favor of a revised approach to AUTH_SYS with better security, based on client host authentication.

While the authors are strongly of the opinion that the second choice is more likely to be successful, and this document reflects that view, there is no intention to foreclose the first option, should the working group choose it. If that option is chosen, then the material in [Section 8.4.6](#) becomes irrelevant and can be ignored while much of that in [Section 8.4.5](#) can be simply modified to expand its scope, as described below in that section.

[8.4.4](#). Need to Better Document and Explain Issues with AUTH_SYS

As described in [Section 8.1](#) there is a need for an appropriate threat analysis for NFSv4. For a number of reasons, issues with AUTH_SYS that would make it difficult to refer to in an NFSv4-wide security document without some additional work preparatory work:

- * Unlike the case of RPCSEC_GSS, there is no discussion of possible security threats.

- * There is no complete documentation how AUTH_SYS is to be used.

- * The reason for deciding that AUTH_SYS is "insecure", while probably valid, adds considerable confusion since it makes it harder to address the issue complained of.

Under other circumstances, it might have been possible to dispense with further analysis of AUTH_SYS because of the following facts:

- * AUTH_SYS is declared "insecure", which might be considered as making a further security analysis beside the point.
- * There are a number of statements with [\[RFC5531\]](#) limiting the use of AUTH_SYS, which might lead a reader to conclude that it was, in essence, a historical artifact.

Despite the above, we will need a more complete of treatment of AUTH_SYS security for a number of reasons.

- * Despite the recognition of its insecurity and restrictions on its use, AUTH_SYS has been at least on a par with RPCSEC_GSS in terms of NFSv4 use, for over a decade following.
- * Given that the ongoing security issues with AUTH_SYS might give rise to efforts to address them, an accurate understanding of these issues is even more important, regardless of how the working group chooses to resolve the issue discussed in [Section 8.4.3](#).

If the decision is made to improve the security characteristics of AUTH_SYS, it is important to understand existing security issues and how they might be best addressed.

If the decision is made to eliminate the use of AUTH_SYS, an adequate understanding of the security issues that made that decision necessary would seem to be required. In that context, the current statement in [\[RFC5531\]](#), regarding the insecurity of AUTH_SYS needs to be clarified.

- * It is likely that attempts to eliminate or restrict use of AUTH_SYS will take the form of a recommendation that it not be used or only be used in certain ways or under certain

circumstances. Such recommendations normally require that the user be made aware of the consequences of doing something other than what is recommended. In this case that would be either to use AUTH_SYS or to use it without client host authentication.

An important question is where such a treatment should appear. In deciding this question, the important question is whether, at this point, it is best addressed as a feature associated with NFSv4 or with RPC.

- * While structurally, AUTH_SYS, presented as one of security supported security flavors associated with RPC would seem to require treatment within the framework of RPC, its recent treatment raises questions about continuing to deal with it solely within this framework. This is not due solely because of the attempt (in [[RFC5531](#)]) to discourage its use, as was done for AUTH_DH. In that case it remains appropriate to discuss it as part of RPC, without a threat analysis, since there is no continuing use justifying another course.
- * The continued use of AUTH_SYS by all NFSv4 minor versions might well be considered to justify a transfer of responsibility to NFSv4, with the authentication flavor being mentioned, in RPC specifications, only as a historical artifact.

The potential continued use of AUTH_SYS with client-host authentication strengthens the NFSv4 connection. Since [[I-D.ietf-nfsv4-rpc-tls](#)] assigns the specification of policies related to authentication to the ULP, it seems that AUTH_SYS needs to be discussed by NFSv4 documents in a manner more substantial than simply listed as another authentication flavor.

- * AUTH_SYS, both with and without client host authentication could be discussed in its own document, including an appropriate threat analysis.

That document could be referenced by a new document updating or obsoleting [[RFC5531](#)]. This would provide an opportunity to address existing text which was essentially ignored by NFSv4.

That document could also be referenced by a new standards-track document dealing with NFSv4 security. The threat analysis would

provide a basis for the whatever approach the working group decides to take with regard to the discussion of AUTH_SYS with client-host authentication. See [Section 8.4.6](#) for further details.

The last of these is most likely to be adopted regardless of how the working group decides the issue described in [Section 8.4.3](#). However, if the working group decides to strongly discourage the use of AUTH_SYS even with client-host authentication, the first is a possibility that should be considered.

[8.4.5](#). Issues to Address for Existing Use of AUTH_SYS

This section presents material that might be included in a standards-track document in an effort to suitably discourage use of AUTH_SYS in its current (insecure) form.

While it is sometimes the case that such a task can be accomplished by substituting one of the terms defined in [\[RFC2119\]](#) by another, that is not the case here. While the designation of AUTH_SYS as optional (to implement) with [\[RFC7530\]](#) and [\[RFC8881\]](#) using the language "MAY be implemented", is wrong and probably needs to be fixed, the situation is more complex than generally recognized. It should be kept in mind that these terms are, in many cases, ignored or followed while ignoring the underlying intent. As a result, such changes, without supporting explanations might not be effective in changing implementer behavior.

To summarize the confusing situation with AUTH_SYS and NFS and the role of the RPC specification, [\[RFC5531\]](#).

- * At the time of publication of [\[RFC5531\]](#), AUTH_SYS was extensively used by NFSv2 [\[RFC1094\]](#), NFSv3 [\[RFC1813\]](#), and NFSv4.0 [\[RFC3530\]](#).
- * The Security Considerations Section of [\[RFC5531\]](#), states "AUTH_SYS SHOULD NOT be used for services that permit clients to modify data", which all versions of NFS clearly are.
- * [Section 10 of \[RFC5531\]](#) states "Implementors MAY include AUTH_SYS in their implementations to support existing applications." Although this statement was probably intended to authorize continued use of AUTH_SYS, the contradiction between this

statement and the Security Consideration section is not noted so there is no way to determine how this contradiction is to be resolved or precisely what "an application" is.

- * The case of NFSv4.1, later defined in [\[RFC5661\]](#), is different since, being a separate protocol, with additional vulnerabilities when AUTH_SYS is used, it would presumably not be an "existing application".
- * Turning from [\[RFC5531\]](#) to [\[RFC8881\]](#) and [\[RFC7530\]](#), we find AUTH_SYS described as an OPTIONAL means of authentication, with its security weaknesses not being discussed or treated as a significant concern.

While there may well be sufficient loopholes within [\[RFC5531\]](#) to avoid an outright contradiction of the rules established there, it is certainly troubling that the security implications of implementing and using AUTH_SYS are not properly recommended against.

While we may need to adjust the normative language, the effect of doing so is limited for reasons that have already been discussed. As a result, we need to put substantial emphasis on clearly stating the consequences of doing what "SHOULD NOT" (or "should not") be done, as is suggested by [\[RFC2119\]](#), at least for the former case.

The following text is one way to present the situation:

- * The use of AUTH_SYS, as described (incompletely) in [Appendix A of \[RFC5531\]](#) is quite detrimental to security and so SHOULD NOT be used.
- * This is so because it involves accepting requests from clients on behalf of specific users without authentication of the clients themselves. In essence, the task of authentication is delegated to NFSv4 client implementations, with the server accepting such authentication decisions. Normally, such an arrangement would require the establishment of a trust relationship between client

and server.

- * Typically, when using AUTH_SYS, use of a privileged port represents the client kernel's judgment (by granting root access) that the client proper is to be trusted but there is no good reason to trust the client kernels in this regard.
- * Although [[RFC8881](#)] and [[RFC7530](#)] previously treated implementation of AUTH_SYS as optional, it appears that this treatment was in error in that it focused on the relevant interoperability constraints without attention to the security consequences of use (discussed in [Section 7 of \[RFC2119\]](#)). While use of AUTH_SYS was not characterized using any of the keywords defined in [[RFC2119](#)], it was reasonable for implementers to assume this was valid given the lack of attention to network security issues and the fact that there would be no point in implementing support for AUTH_SYS in servers if it could not be used.
- * Although, as a result of these previous approaches to AUTH_SYS without client host authentication, it might be used extensively despite the security issues that cause us to specify that it "SHOULD NOT" be used are sufficient to justify this shift. However, due to the difficulty of making such a shift, the need to maintain continuity of support is a valid reason to continue to use it in this way, as long as this decision is made with awareness of the security consequences:

- That it is possible for an attacker to impersonate a trusted client by using its IP address in source IP address of packet sent, enabling to send its own requests to read or modify data, as any particular user.
- That compromising any client whose AUTH_SYS requests are accepted allows it to create a connection from which its unauthenticated requests will be executed by the server.
- That it is possible for attackers to work around "root squashing" since it can assume any identity it intends to,

other than that of root.

8.4.6. Issues to Resolve for Revised Approach to AUTH_SYS

The availability of client-host authentication makes possible a revised approach to AUTH_SYS addressing some of its security vulnerabilities. A revised approach to AUTH_SYS might provide:

1. The ability to prevent, using client host-authentication, another machine masquerading as the expected client.

The authentication features described in [\[I-D.ietf-nfsv4-rpc-tls\]](#), would be assure that you could identify trusted client machines, eliminating a major security issue with AUTH_SYS. On the other hand, it is not clear whether servers would be wise to trust clients to this degree, especially if the items below (#2 and #3) are not satisfactorily dealt with.

2. It would be possible to include, as part of the client authentication material, information known only to the client and the server. The secret value would be chosen by the client, and saved so it is usable by later client instances. If this were done, it would prevent any process with root privileges on a trusted machine from impersonating the trusted client. If this were not done, then any compromise of a single trusted client would compromise any server which trusted it to present AUTH_SYS requests.

To make this approach viable, servers would have to maintain, in persistent storage, a record of the secret value assigned to each authenticated client. When an authenticated client connected with a different value it would indicate that a root process on the client-host was attempting to impersonate the NFSv4 client.

3. The power granted to the authenticated client-host, to make a request on behalf of user or group has troubling consequences. Although most implementations have the ability to exclude

requests from the root user, that may not, as has been noted previously, be sufficient to satisfactorily deal with the problem, since many deployments contain security-critical files that can be accessed and modified but users other than root.

It would be desirable to exclude a wider range of users such as all users within a given group (or alternatively include only users within a given group). However, neither of these is viable since servers receiving AUTH_SYS requests typically have no knowledge of what users belong to what groups, relying on the AUTH_SYS client to provide this information within the request.

Given this lack of knowledge on the part of the servers, any restrictions on the ability of AUTH_SYS requests to access or modify files designate a set of protected files in some way. Since designating a list of files is complicated and hard to maintain, it might be possible to describe the files whose access and/or modification by means of an ACL. When a request is made to access a file, that request is suppressed (i.e. treated as a request by "nobody") iff it would be allowed to access by the designated ACL and similarly in the case of modifying requests.

How to address the question of the potential use of AUTH_SYS with client-host authentication would depend on the working group's judgement about how likely such an arrangement would be likely to be in providing security when AUTH_SYS is used. This in turn would depend on the working group's judgment as to whether solutions for #2 and #3 could be defined and effectively deployed, as well as the results of the threat analysis

We assume that AUTH_SYS with client-host authentication needs to be considered inferior to RPCSEC_GSS but sufficiently superior to AUTH_SYS without client authentication to make it desirable to encourage its use, given that it is impossible to eliminate use of AUTH_SYS, even with a "MUST NOT". A number of potential introductions to the topic, of varying tone are explored below:

- * If the working group wants to focus on the improvement relative to AUTH_SYS without client-host authentication, the following paragraphs might be an appropriate introduction:

When AUTH_SYS is used with client-host authentication, then the server, by allowing use of AUTH_SYS for that specific client, is essentially delegating the user authentication task to that client who is trusted by the server to perform that task locally, just as typically done when authenticating users for the purpose of local file access, even though the scope of damage that could be enabled by any false authentication is likely to be much wider in this case.

By accepting AUTH_SYS requests from a particular client, a server is relying on the security of the client for the security of all data stored on the server. This is in contrast to RPCSEC_GSS, in which authentication is provided for each request issued. Because of this, the use of AUTH_SYS should only be allowed from client hosts that can reasonably be trusted.

- * If the working group wants to adopt a more cautionary tone regarding use of AUTH_SYS with client-host authentication, the following paragraphs might be an appropriate introduction:

By accepting AUTH_SYS requests from a particular client, a server is relying on the security of the client for the security of all data stored on the server. This is in contrast to RPCSEC_GSS, in which authentication is provided for each request issued. Because of this, the use of AUTH_SYS should only be allowed from client hosts that can reasonably be trusted and whose internal security is sufficiently robust that the additional risk to server data can be considered reasonable.

- * If the working group wants to focus more on encouraging use of RPCSEC_GSS, then on improving the security of those choosing to use AUTH_SYS, a paragraph like the following might be appropriate.

While the use of AUTH_SYS with client-host authentication eliminates some of the major security issues that arise from use of AUTH_SYS, it is still inferior, from a security point of view, to RPCSEC_GSS. As a result, its use should be limited to situations in which, use of RPCSEC_GSS is not a practical possibility.

The following paragraph might be added if item #2 is not successfully addressed:

- * It should be noted that any security weaknesses in the client host kernel which might allow an attacker to execute in privileged mode, are likely to allow that attacker to present AUTH_SYS requests to the server and have them executed without any authentication beyond verifying that they were issued by a trusted host.

The following paragraph might be added if item #2 is successfully addressed:

- * In order to assure that it is not possible for a process running as root to successfully masquerade as a previously active kernel-based client on that machine, user-level clients even privileged ones, will present distinct authentication material from a kernel-based client and from each other. Servers SHOULD maintain, persistently, sufficient information so to be able to detect a situation in which process with root privileges on a trusted server masquerades as a client previously recognized on that and judged secure.

The following paragraph might be added if item #3 is not successfully addressed:

- * There is no standardized way the server can effectively limit the range of client user identities and group memberships assumed by AUTH_SYS requests except for the commonly implemented practice of "root squashing".

The following paragraph might be added if item #3 is successfully addressed:

- * Server SHOULD provide a means by which specific sets of sensitive files are made inaccessible or unmodifiable by use of AUTH_SYS requests, in order to limit the damage that could occur in the event that a client compromise allows unauthenticated requests to be generated and acted upon. One desirable way of providing such a facility would be to allow an ACL to be associated with each authenticated client host (e.g. by designating a file whose ACL is to be used). If this is done any request whose user together with a group set would allow it to be accepted by the specified ACL

would be rejected.

The following closing paragraphs are proposed for an eventual introduction to AUTH_SYS, as revised. In these paragraphs, material to be added only if a particular item above is addressed or not addressed will appear in brackets with the prefix "if-#x:" or "if-not-#x:".

- * The ability to make AUTH_SYS requests of a server SHOULD be limited to those cases in which the client host making those requests can be trusted to appropriately verify a user's credentials (e.g. passwords) before issuing RPC requests on that user's behalf. [if-#3: This limitation should be applied even when the set of users or owning groups is restricted to specially protect files deemed sensitive.] [if-not-#3: When not so limited, requests purportedly from any user except possibly root, can be synthesized and acted upon, without effective authentication.]
- * For that reason, accepting AUTH_SYS requests from a large set of client hosts is a practice to be avoided, with individual hosts included where their security has been verified. The only possible exception is where a set of policies is in place which effectively limits the kernel software allowed to run on those hosts to ensure that those hosts appropriately providing user authentication [if-not-#2: and capable of excluding untrusted code from masquerading as an NFSv4 server by assuming root privileges.]

[8.5.](#) Handling of State Protection

[8.5.1.](#) Background

The inclusion of state management within NFSv4 created new security issues in which the objects requiring protection were not files associated with particular users but locks, opens, and delegations, associated with particular clients.

In NFSv4.0, some protection could be provided when RPCSEC_GSS was used by limiting modification of state objects to users having access to the associated file. Nevertheless, the fundamental issue remained unresolved.

When sessions were introduced into NFSv4 (in NFSv4.1) there were additional vulnerabilities that needed to be protected against. There needed to be protection to prevent a session established by one client being bound to a connection established by another client, leading to the following possibilities:

- * The existing session could be destroyed.
- * The associated clientid could be destroyed.
- * The existing session slots could be used resulting in the legitimate owner of the session having his requests rejected because his slot sequence values had been rendered invalid because of the interloper's requests using the same slots.

Facilities were defined in [[RFC5661](#)] to allow such client-based protection to be implemented: SP4_MACHCRED and SP4_SSV but implementation has been quite limited. Client-host authentication makes it possible to apply such protection more generally, as described below.

[8.5.2](#). Issues to be Addressed

While transport-level encryption would make it harder for attackers to mount attacks based on these vulnerabilities, the authentication material provided when a TLS connection is established could enable the vulnerabilities discussed above to be eliminated without the work of implementing SP4_MACHCRED or SP4_SSV, and irrespective of whether the client is using AUTH_SYS or RPCSEC_GSS.

If one has a TLS connection including suitable authentication material, then the server is in a position to reject attempts to bind to sessions established under this connection by other connections that are not TLS connections with client-host authentication or are not established by the same client, unless SP4_MACHCRED or SP4_SSV is in effect. This would make state protection available to those using SP4_NONE, i.e. the vast majority, whether they used RPCSEC_GSS or AUTH_SYS with client-host authentication.

The fact that this is an NFSv4.1-only security feature would make it difficult to fully address solely in the NFSv4-wide security

document which will be written. The following proposal is intended to allow the security document to be published before proceeding to publish the anticipated rfc5661bis:

- * In the anticipated standards-track NFSv4 security document, it can be stated that when SP4_NONE is specified and the client owning the session is authenticated, then the session is protected from being bound-to or destroyed by an unauthenticated client or a different authenticated client, then the effect is the same as if SP4_MACHCRED or SP4_SSV were used by the owning client and a non-matching client was found. Since the new standards-track document would update [[RFC8881](#)], that provides adequate notice of this change

- * Later, when rfc5661bis is ready to be published, there would an opportunity to provide introductory material explaining the various ways clients can be authenticated for the purpose of state protection, including use of SP4_MACHCRED or SP4_SSV and authentication as part of use of RPC-TLS. This would allow defining the circumstances under which one connection can access/modify state owned by an another client in terms of whether the connections involved are part of the same client. References to such abstract relationships would replace most current mentions of SP4_NONE, SP4_MACHCRED, and SP4_SSV in [[RFC8881](#)].

It should be noted that this use of client-host authentication provides applicable protection even when AUTH_SYS is used. This is so however the issues discussed in [Section 8.4.6](#) are addressed, and independent of the decision made regarding the choice discussed in [Section 8.4.3](#) unless it is decided that AUTH_SYS MUST NOT be used. Issues regarding the impersonation of a large set of users are not relevant because user ids are not referenced. Furthermore, the possibility of a privileged user-level client implementing a denial-

of-service attach (if issue #2 in [Section 8.4.6](#) is not addressed), is not of concern since such a privileged process would find it far easier to deny service directly on the client host.

[9.](#) IANA Considerations

The current document does not require any actions by IANA.

[10.](#) Security Considerations

The convention of requiring a Security Considerations Section within an I-D encounters difficulties when applied to a document dealing solely with security, making it most unclear what such a section might contain for such documents.

In addition, the nature of this particular document poses further issues regarding the possible content of a Security Consideration Section:

- * This document does not define a protocol or protocol feature that can be implemented, making it unclear how a threat analysis could be performed whose security considerations would be described.
- * Despite this document's informational nature, it does attempt to address security issues for an existing set of protocols

In light of the above, this section will summarize the security-related message of earlier sections and will, in subsections, discuss the appropriate contents of Security Considerations Sections for a eventual standards-track documents to be written

The message of this document with regard to security issues can be summarized as follows

- * The current security approach for the NFSv4 protocols has some significant weaknesses, requiring a major reworking of the security approach.

- * This reworking can be accomplished without change to the NFS protocols per se, by taking advantage of recent security improvements defined within the RPC layer.
- * A threat analysis, not provided in previous NFSv4 specification documents, will be the provided within a new standards-track document addressing security for all minor versions of NFSv4.
- * Because there is no threat analysis of AUTH_SYS in existence, one needs to be provided, preferably in a new standard-track document dealing with AUTH_SYS. This is despite the designation of AUTH_SYS (in [\[RFC5531\]](#) as "insecure", which was never, at least in the case of NFSv4, acted upon. Such a document is needed to clarify the security weaknesses of AUTH_SYS and the degree such weakness could be rectified by the implementation of TLS-based encryption and/or client host authentication.

[10.1.](#) Security Considerations Section for Eventual NFSv4-wide Standards-track Security Document

Although it is clear that such a document would need to include a threat analysis, as provided for by [\[RFC3552\]](#). However, the length of such an analysis might well be so large that it cannot be reasonably contained within a Security Considerations Section. In that case, the Security Considerations Section will summarize the results of the threat analysis, referring to those parts that appear in other sections of the document.

[10.2.](#) Security Considerations Section for Eventual Rfc5661bis

This section would have to refer to the Security Considerations section of the new NFSv4-wide security document. In addition, to deal with NFSv4.1-specific security issues (i.e. pNFS and state protection), there would be additional material, most likely within the Security Considerations section or a subsection thereof.

[10.3.](#) Security Considerations Section for Potential Revised RPC Specification Document

The general form of this section in [\[RFC5531\]](#) in which references are made to documents for each security flavor, seem appropriate.

However, the following changes are likely to be required:

- * The treatment of AUTH_SYS might need to reference a new document describing AUTH_SYS, rather than the incomplete description in [Appendix A](#), which needs to be removed.

In addition, the last two sentences of the initial paragraph of [section 14](#) need to be reworked to respond to the fact that while, there was no clear violation of them, this insecure authentication flavor was used by NFSv4 for over a decade subsequently. Although the following attempt at loophole removal might not survive the editing and review process, the issues it raises are worth considering.

- As discussed in the referenced document, the use of AUTH_SYS in the clear without client host-authentication introduces significant security vulnerabilities in that it allows unauthenticated requests to be created by an unauthenticated client. This make its use highly inappropriate for any RPC service, particularly those that allow clients to modify data.
- Similarly, in specifications of standards-track RPC services, it is inappropriate to make such use REQUIRED or RECOMMENDED. Where such use is allowed at all (whether by "MAY" or "SHOULD NOT" is used), viable non-disruptive alternatives need to be provided and the associated Security Consideration sections need to clearly explain the security consequences of use.
- * The case of AUTH_DH is similar to that of AUTH_SYS in that there is a simple declaration of insecurity. However, there is a document referenced and there is no indication that the loopholes in the treatment were bypassed in this case, as they were for AUTH_SYS. Still, it might be worth revising the final two sentences as were done with AUTH_SYS.
- * Given that the final paragraph of the Security Considerations section was drafted without consideration of reverse direction operation and without consideration the possibility that either TLs encryption or client host authentication might be available (and potentially REQUIRED), redrafting it to be more appropriate for the environment made possible by [\[I-D.ietf-nfsv4-rpc-tls\]](#) is likely.

The following replacement text is suggested to deal with these issues:

- Standards-track RPC services need to mandate server and client support for RPCSEC_GSS when used in the forward direction. The treatment of authentication for reverse-direction operation needs to provide for authentication of the identity of the requester (i.e. on the server) to the responder (i.e. on the client). When reverse direction requests are not made on behalf of a specific user, the mutual authentication provided by RPC-TLS can be relied on but authentication of users on reverse-direction requests will need to use RPCSEC_GSS.
- For either direction of operation, when user identities must be authenticated, such services need to mandate support for an authentication pseudo-flavor. In situations in which it is possible/allowed for RPC services to be used without TLS encryption there is also a need to mandate pseudo-flavors with additional appropriate levels of security, depending on the need for authentication only, integrity (a.k.a. non-repudiation), or encryption to provide data confidentiality.

10.4. Security Considerations Section for Potential Standards-track Document Dealing with AUTH_SYS

As mentioned above, a prime goal of a general threat analysis for AUTH_SYS would be to clarify the degree to which the security weaknesses of AUTH_SYS can be successfully addressed using transport-level security facilities. Those pieces of the threat analysis might well appear outside the Security Consideration section proper.

The Security Considerations would need to summarize these and provide a basis for ULPs that allow AUTH_SYS to specify the appropriate conditions for use and the consequences of not enforcing these.

11. References

11.1. Normative References

[I-D.ietf-nfsv4-rpc-tls]

Myklebust, T. and C. Lever, "Towards Remote Procedure Call Encryption By Default", Work in Progress, Internet-Draft, [draft-ietf-nfsv4-rpc-tls-11](https://tools.ietf.org/html/draft-ietf-nfsv4-rpc-tls-11), 23 November 2020, <<https://tools.ietf.org/html/draft-ietf-nfsv4-rpc-tls-11>>.

[I-D.ietf-nfsv4-rpcrdma-version-two]

Lever, C. and D. Noveck, "RPC-over-RDMA Version 2 Protocol", Work in Progress, Internet-Draft, [draft-ietf-](#)

nfsv4-rpcrdma-version-two-03, 10 August 2020,
<<https://tools.ietf.org/html/draft-ietf-nfsv4-rpcrdma-version-two-03>>.

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", [BCP 72](#), [RFC 3552](#), DOI 10.17487/RFC3552, July 2003, <<https://www.rfc-editor.org/info/rfc3552>>.
- [RFC5531] Thurlow, R., "RPC: Remote Procedure Call Protocol Specification Version 2", [RFC 5531](#), DOI 10.17487/RFC5531, May 2009, <<https://www.rfc-editor.org/info/rfc5531>>.
- [RFC7530] Haynes, T., Ed. and D. Noveck, Ed., "Network File System (NFS) Version 4 Protocol", [RFC 7530](#), DOI 10.17487/RFC7530, March 2015, <<https://www.rfc-editor.org/info/rfc7530>>.
- [RFC7862] Haynes, T., "Network File System (NFS) Version 4 Minor Version 2 Protocol", [RFC 7862](#), DOI 10.17487/RFC7862, November 2016, <<https://www.rfc-editor.org/info/rfc7862>>.
- [RFC8166] Lever, C., Ed., Simpson, W., and T. Talpey, "Remote Direct Memory Access Transport for Remote Procedure Call Version 1", [RFC 8166](#), DOI 10.17487/RFC8166, June 2017, <<https://www.rfc-editor.org/info/rfc8166>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in [RFC 2119](#) Key Words", [BCP 14](#), [RFC 8174](#), DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8881] Noveck, D., Ed. and C. Lever, "Network File System (NFS) Version 4 Minor Version 1 Protocol", [RFC 8881](#), DOI 10.17487/RFC8881, August 2020, <<https://www.rfc-editor.org/info/rfc8881>>.

[11.2](#). Informative References

[I-D.dnoveck-nfsv4-rfc5661bis]

Noveck, D., "Network File System (NFS) Version 4 Minor Version 1 Protocol", Work in Progress, Internet-Draft, [draft-dnoveck-nfsv4-rfc5661bis-00](https://tools.ietf.org/html/draft-dnoveck-nfsv4-rfc5661bis-00), 31 December 2020, <<https://tools.ietf.org/html/draft-dnoveck-nfsv4-rfc5661bis-00>>.

Noveck & Lever

Expires 25 July 2021

[Page 53]

Internet-Draft

NFSv4 Security Needs

January 2021

- [RFC1094] Nowicki, B., "NFS: Network File System Protocol specification", [RFC 1094](https://www.rfc-editor.org/info/rfc1094), DOI 10.17487/RFC1094, March 1989, <<https://www.rfc-editor.org/info/rfc1094>>.
- [RFC1813] Callaghan, B., Pawlowski, B., and P. Staubach, "NFS Version 3 Protocol Specification", [RFC 1813](https://www.rfc-editor.org/info/rfc1813), DOI 10.17487/RFC1813, June 1995, <<https://www.rfc-editor.org/info/rfc1813>>.
- [RFC2624] Shepler, S., "NFS Version 4 Design Considerations", [RFC 2624](https://www.rfc-editor.org/info/rfc2624), DOI 10.17487/RFC2624, June 1999, <<https://www.rfc-editor.org/info/rfc2624>>.
- [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](https://www.rfc-editor.org/info/rfc3530), DOI 10.17487/RFC3530, April 2003, <<https://www.rfc-editor.org/info/rfc3530>>.
- [RFC4949] Shirey, R., "Internet Security Glossary, Version 2", FYI 36, [RFC 4949](https://www.rfc-editor.org/info/rfc4949), DOI 10.17487/RFC4949, August 2007, <<https://www.rfc-editor.org/info/rfc4949>>.
- [RFC5661] Shepler, S., Ed., Eisler, M., Ed., and D. Noveck, Ed., "Network File System (NFS) Version 4 Minor Version 1 Protocol", [RFC 5661](https://www.rfc-editor.org/info/rfc5661), DOI 10.17487/RFC5661, January 2010, <<https://www.rfc-editor.org/info/rfc5661>>.
- [RFC8178] Noveck, D., "Rules for NFSv4 Extensions and Minor Versions", [RFC 8178](https://www.rfc-editor.org/info/rfc8178), DOI 10.17487/RFC8178, July 2017, <<https://www.rfc-editor.org/info/rfc8178>>.

Acknowledgements

The authors would like to thank all who contributed to [\[I-D.ietf-nfsv4-rpc-tls\]](#) for their important work providing security

at the RPC layer, enabling us to break the NFSv4 security logjam.

The authors would like to thank Tom Haynes (of Hammerspace) for introducing the idea of dealing with security for all minor versions in a single document.

Authors' Addresses

Noveck & Lever	Expires 25 July 2021	[Page 54]
----------------	----------------------	-----------

Internet-Draft	NFSv4 Security Needs	January 2021
----------------	----------------------	--------------

David Noveck
NetApp
1601 Trapelo Road
Waltham, MA 02451
United States of America

Phone: +1 781 572 8038
Email: davenoveck@gmail.com

Charles Lever (editor)
Oracle Corporation
United States of America

Email: chuck.lever@oracle.com

