Workgroup: Network File System Version 4
Internet-Draft:
draft-cel-nfsv4-rpc-tls-pseudoflavors-02
Published: 27 December 2021
Intended Status: Standards Track
Expires: 30 June 2022
Authors: C. Lever
Oracle

Pseudo-flavors for Remote Procedure Calls with Transport Layer Security

Abstract

Recent innovations in Remote Procedure Call (RPC) transport layer security enable broad deployment of encryption and mutual peer authentication when exchanging RPC messages. These security mechanisms can protect peers who continue to use the AUTH_SYS RPC auth flavor, which is not cryptographically secure, on open networks. This document introduces RPC auth pseudo-flavors that an RPC service can use to indicate transport layer security requirements for accessing that service, and a mechanism the service can use to enforce those requirements.

Note

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

Discussion of this draft occurs on the NFSv4 working group mailing list, archived at https://mailarchive.ietf.org/arch/browse/nfsv4/. Working Group information is available at https://datatracker.ietf.org/wg/nfsv4/about/.

Submit suggestions and changes as pull requests at https://github.com/chucklever/i-d-rpc-tls-pseudoflavors. Instructions are on that page.

Status of This Memo

This Internet-Draft will expire on 30 June 2022.

Table of Contents

1. Introduction

Each RPC transaction may be associated with a user and a set of groups. That transaction's RPC auth flavor determines how the user and groups are identified and whether they are authenticated. Peers

that host applications and RPC services may also be identified and authenicated in each RPC transaction, again depending on that transaction's RPC auth flavor [RFC5531].

Not all flavors provide peer and user identification and authentication. For example, the traditional RPC auth flavor AUTH_NONE identifies no user or group and provides no authentication of users or peers. The traditional RPC auth flavor AUTH_SYS provides identification of peers, users, and groups, but does not provide authentication of any of these.

Moreover, unlike some GSS security services, these RPC auth flavors provide no confidentiality or integrity checking services. Therefore AUTH_NONE and AUTH_SYS are considered insecure.

Mutual peer authentication and encryption provided at the transport layer can make the use of AUTH_NONE and AUTH_SYS more secure. An RPC service might want to indicate to its clients that it will not allow access via AUTH_NONE or AUTH_SYS unless transport layer security services are in place. To do that, this document specifies several pseudo-flavors that upper layers such as NFS [RFC8881] can use to enforce stronger security when unauthenticated RPC auth flavors are in use.

The author expects that, in addition to RPC-with-TLS [I-D.ietf-nfsv4-rpc-tls], other novel RPC transports will eventually appear that provide similar security features. These transports can benefit from the pseudo-flavors defined in this document, or this approach can be extended if new transport security features require it.

1.1. Terminology

This document adopts the terminology introduced in Section 3 of [RFC6973] and assumes a working knowledge of the Remote Procedure Call (RPC) version 2 protocol [RFC5531] and the Transport Layer Security (TLS) protocol [RFC8446].

This document adheres to the convention that a "client" is a network host that actively initiates an association, and a "server" is a network host that passively accepts an association request.

For the purposes of this document, an Upper-Layer Protocol is an RPC Program and Version tuple comprised of a set of procedure calls defining a single API. One example of a ULP is the Network File System Version 4.0 [RFC7530].

An "RPC auth flavor" is a set of protocol elements that can identify a network peer and a user and possibly authenticate either or both. Section 13.4.2 of [RFC5531] explains the differences between RPC auth flavors and pseudo-flavors.

RPC documentation historically refers to the authentication of a host as "machine authentication" or "host authentication". TLS documentation refers to the same as "peer authentication". The current document uses only "peer authentication".

The term "user authentication" in the current document refers specifically to the RPC caller's credential provided in the "cred" and "verf" fields in each RPC Call.

This document uses the term "insecure RPC auth flavor" (or "insecure flavor" for short) to refer to a class of RPC auth flavors which provide no user or peer authentication. Two prime examples of an insecure RPC auth flavor are AUTH_NONE and AUTH_SYS.

2. 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.

3. RPC Auth Pseudo-flavors for Transport Layer Security

Section 4 of [I-D.ietf-nfsv4-rpc-tls] introduces a special RPC auth flavor known as AUTH_TLS. This RPC auth flavor is used only in a NULL procedure that probes the presence of support for RPC-with-TLS, and acts as a STARTTLS barrier.

This auth flavor does not carry the identity of the peer or a user. RPC clients do not use this RPC auth flavor to authenticate users in RPC Calls for non-NULL RPC procedures.

Once transport layer security has been established between two RPC peers, an RPC client can use insecure flavors when forming RPC Calls with knowledge that the RPC server is known and trusted, and without concern that the communication can be altered or monitored.

In some cases an RPC service might want to restrict access to only clients that have authenticated, or perhaps only when encryption protects communication. The pseudo-flavors defined below enable RPC-based services to indicate and enforce access restrictions of this type.

3.1. Definitions of New Pseudo-flavors

This document specifies several pseudo-flavors that servers may advertise to clients via mechanisms not defined here. Using the RPC auth flavor registry instantiated in [RFC5531] gives us leeway to

introduce a narrow basic set of pseudoflavors in this document and then expand them, via additional documents, as needs arise.

RPC clients continue to use AUTH_NONE (0) or AUTH_SYS (1) in individual transactions while the network transport service provides cryptographically secure authentication or encryption, as follows:

- The new pseudo-flavor AUTH_NONE_MPA indicates that the client may use the AUTH_NONE RPC auth flavor only if both peers have mutually authenticated. Encryption of traffic between these peers is not required.

- The new pseudo-flavor AUTH_NONE_ENC indicates that the client may use the AUTH_NONE RPC auth flavor only if traffic between these peers is encrypted. Mutual peer authentication is not required.

- The new pseudo-flavor AUTH_NONE_MPA_ENC indicates that the client may use the AUTH_NONE RPC auth flavor only if both peers have mutually authenticated and traffic between these peers is encrypted.

- The new pseudo-flavor AUTH_SYS_MPA indicates that the client may use the AUTH_SYS RPC auth flavor only if both peers have mutually authenticated. Encryption of traffic between these peers is not required.

- The new pseudo-flavor AUTH_SYS_ENC indicates that the client may use the AUTH_SYS RPC auth flavor only if traffic between these peers is encrypted. Mutual peer authentication is not required.

- The new pseudo-flavor AUTH_SYS_MPA_ENC indicates that the client may use the AUTH_SYS RPC auth flavor only if both peers have mutually authenticated and traffic between these peers is encrypted.

Because the RPC layer is not aware of pseudo-flavors, the Upper-Layer Protocol is responsible for ensuring that appropriate transport layer security is in place when clients use AUTH_SYS or AUTH_NONE. The next section explains how server implementations enforce the use of transport layer security.

4. Channel Binding

Certain aspects of transport layer security are not new. A deployment might choose to run NFS on a virtual private network established via an ssh tunnel or over IPsec, for example. The Generic Security Service Application Program Interface (GSS-API) specification [RFC2743] recognized the use of security provided by transport services underlying GSS with the introduction of channel

binding. [RFC5056] further describes channel binding as a concept that...

> ...allows applications to establish that the two end-points of a secure channel at one network layer are the same as at a higher layer by binding authentication at the higher layer to the channel at the lower layer. This allows applications to delegate session protection to lower layers, which has various performance benefits.

We are particularly interested in ensuring that the mutual authentication done during a TLS handshake (most recently specified in [RFC8446]) on a transport service that handles RPC traffic can be recognized and used by Upper-Layer Protocols for securely authenticating the communicating RPC peers.

Section 7 of [RFC5929] identifies a set of API characteristics that RPC and its underlying transport provide to such protocols.

4.1. TLS Channel Binding

[RFC5929] defines several TLS channel binding types that Upper-Layer Protocol implementations can use to determine whether appropriate security is in place to protect RPC transactions that continue to use insecure RPC auth flavors such as AUTH_SYS.

When used with a Certificate handshake message, the 'tls-server-end-point' channel binding type as defined in Section 4 of [RFC5929] serves as authentication for securing pseudo-flavors that require mutual peer authentication.

RPC-with-TLS requires the use of TLS session encryption [I-D.ietf-nfsv4-rpc-tls]. The presence of TLS under an RPC transport is enough to secure pseudo-flavors that require encryption. A peer can use channel binding to determine whether peer authentication has also occurred and whether that authentication was mutual or server-only.

Moreover, in the particular case of TLS, when a handshake fails, both peers are made aware of the failure reason via the Finished message. The failure reason can then be reported to the Upper-Layer Protocol so the local administrator can take specific corrective action.

For instance, an RPC server's local security policy might require that the RPC client's IP address or hostname match its certificates Subject Alt Name (SAN). This is not always possible if the client's IP address and hostname are assigned dynamically. When such a server causes a handshake failure, administrators can be made aware that the server's SAN policy restricted a client's access, and corrective action can then be taken.

4.2. SSHv2 Channel Binding

When RPC traverses an SSHv2 tunnel established between an RPC server and an RPC client, the 'tls-unique' channel binding type as defined in Section 3 of [RFC5929] can be used to authenticate peer endpoints and provide appropriate confidentiality.

4.3. Channel Binding for RDMA Transports

As of this writing, RPC-over-RDMA [RFC8166] does not provide a transport layer security service. However, Section 5 of [RFC5056] suggests a mechanism by which channel binding can protect RDDP [RFC5040], the protocol that handles remote direct data placement for the iWARP family of protocols. The transport layer underlying RDDP might use IPsec [RFC6071], TLS [RFC8446], or Encapsulating Security Payload (ESP) [RFC4303].

5. NFS Examples

This section presents examples of how a commonly-used Upper-Layer Protocol (NFS) can make use of these pseudo-flavors.

5.1. Network File System Versions 2 and 3

NFSv3 clients use the MNT procedure, defined in Appendix I of [RFC1813], to discover which RPC auth flavors may be used to access a particular shared NFSv3 filesystem.

To require NFSv3 clients to employ underlying transport security when using AUTH_NONE or AUTH_SYS, the NFS server includes one or more of the new pseudo-flavors defined in Section 8 in the auth_flavors list that is part of a MNT response.

When determining whether a filehandle-bearing operation is authorized, an NFSv3 server uses channel binding to ensure that appropriate transport layer security is in place before processing an incoming NFS request that uses an insecure RPC auth flavor. If that request is not authorized, the NFSv3 server can respond with an nfs_stat of NFS3ERR_STALE.

The usage of the MNT procedure as described in [RFC1094] is the same with the exception that an NFSv2 server responds with NFSERR_STALE instead of NFS3ERR_STALE.

5.2. Network File System Version 4

NFSv4 clients use the SECINFO or SECINFO_NO_NAME procedures, as defined in [RFC8881], to discover which RPC auth flavors may be used to access a particular shared NFSv4 filesystem.

To require NFSv4 clients to employ underlying transport security when using AUTH_NONE or AUTH_SYS, the NFS server includes one or more of the new pseudo-flavors defined in Section 8 in the SECINFO4resok list that is part of a SECINFO or SECINFO_NO_NAME response.

When determining whether a filehandle-bearing operation is authorized, an NFSv4 server uses channel binding to ensure that appropriate transport layer security is in place before processing an incoming NFSv4 COMPOUND that uses an insecure RPC auth flavor. If that request is not authorized, the NFSv4 server terminates the COMPOUND with a status code of NFS4ERR_WRONGSEC.

5.2.1. NFSv4 State Protection

Note: This section updates RFC 8881.

An alternate approach might place the updates described in this section in rfc5661bis.

Section 2.4.3 of [RFC8881] explains how an NFSv4 server determines when an NFSv4 client is authorized to create a new lease or replace a previous one. This mechanism prevents clients from maliciously or unintentionally wiping open and lock state for another client. Section 2.10.8.3 of that document further specifies how the server responds to unauthorized state changes.

When used with a Certificate handshake message, the 'tls-server-end-point' channel binding type as defined in Section 4 of [RFC5929] can provide protection similar to SP4_MACH_CRED.

This document modifies the text of the first bullet in Section 2.4.3 of [RFC8881] to include the use of transport layer security as follows:

* The principal that created the client ID for the client owner is the same as the principal that is sending the EXCHANGE_ID operation. Note that if the client ID was created with SP4_MACH_CRED state protection (Section 18.35), either:

  - The principal **MUST** be based on RPCSEC_GSS authentication, the RPCSEC_GSS service used **MUST** be integrity or privacy, and the same GSS mechanism and principal **MUST** be used as that used when the client ID was created. Or,

  - The principal **MUST** be based on AUTH_SYS, and the server **MUST** use channel binding to verify the identity of the client peer when performing any of the operations specified in the spa_mach_ops bitmaps. Or,

- The principal **MUST** be based on AUTH_NONE, and the server **MUST** use channel binding to verify the identity of the client peer when performing any of the operations specified in the spa_mach_ops bitmaps.

Subsequent discussion of SP4_MACH_CRED in [RFC8881] in Sections 2.10.5.1, 2.10.8.3, and 2.10.11.3 would need similar adjustments.

Further, NFSv4 server implementations may implement a security policy that restricts the set of clients or security flavors that can establish a lease via SETCLIENTID or EXCHANGE_ID. However, [RFC8881] does not allow EXCHANGE_ID or CREATE_SESSION to return NFS4ERR_WRONGSEC, and [RFC7530] does not allow SETCLIENTID to return NFS4ERR_WRONGSEC.

NFSv4.1-based protocols might be updated to allow EXCHANGE_ID or CREATE_SESSION to return NFS4ERR_WRONG_CRED. However, that solution would be challenging for NFSv4.0, which does not have a definition for NFS4ERR_WRONG_CRED.

More discussion is necessary to determine the exact mechanism to handle this case in both protocols and to determine which documents need to specify that mechanism.

6. Implementation Status

This section is to be removed before publishing this document as an RFC.

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC7942]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs.

Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

There are currently no known implementations of the new RPC pseudo-flavors requested by this document.

7. Security Considerations

Discussion of shortcomings peculiar to the AUTH_SYS RPC auth flavor appears in the final paragraph of Appendix A of [RFC5531] and in Appendix A of [I-D.ietf-nfsv4-rpc-tls].

When implementing or deploying transport layer security to protect an upper-level RPC protocol:

* RPC clients that support transport layer security **SHOULD** use it whenever possible. Typically the only reason not to is when performance is important and reasonable security can be provided in some other way.

* RPC clients that support transport layer security and have the ability to authenticate **SHOULD** do so. The only reason not to authenticate is when authentication and encryption can only be enabled together, performance is paramount, and there are other available mechanisms that can provide peer authentication securely.

The pseudo-flavors defined in this document enable RPC servers to indicate required levels of security so that RPC clients can make informed and autonomous decisions that balance performance and scalability against security needs.

Important security considerations specific to the use of channel binding are discussed throughout [RFC5056] and in Section 10 of [RFC5929].

8. IANA Considerations

RFC Editor: In the following subsections, please replace RFC-TBD with the RFC number assigned to this document. Furthermore, please remove this Editor's Note before this document is published.

8.1. New RPC Auth Flavors

Following Appendix B of [RFC5531], this document requests several new entries in the RPC Authentication Flavor Numbers registry. The purpose of these new flavors is to indicate the use of transport layer encryption or mutual peer authentication with insecure RPC auth flavors. All new flavors described in the sections below are pseudo-flavors.

8.2. Pseudo-flavors for Secure AUTH_NONE

The fields in the new entries are assigned as follows:

Identifier String	Flavor Name	Value	Description	Reference
AUTH_NONE_MPA	NONE_MPA	TBD	AUTH_NONE with mutual peer authentication	RFC_TBD
AUTH_NONE_ENC	NONE_ENC	TBD	AUTH_NONE with transport layer encryption	RFC_TBD
AUTH_NONE_MPA_ENC	NONE_MPA_ENC	TBD	AUTH_NONE with peer authentication and encryption	RFC_TBD

Table 1

Please allocate the numeric values from the range 400000-409999.

8.3. Pseudo-flavors for Secure AUTH_SYS

The fields in the new entries are assigned as follows:

Identifier String	Flavor Name	Value	Description	Reference
AUTH_SYS_MPA	SYS_MPA	TBD	AUTH_SYS with mutual peer authentication	RFC_TBD
AUTH_SYS_ENC	SYS_ENC	TBD	AUTH_SYS with transport layer encryption	RFC_TBD
AUTH_SYS_MPA_ENC	SYS_MPA_ENC	TBD	AUTH_SYS with peer authentication and encryption	RFC_TBD

Table 2

Please allocate the numeric values from the range 410000-419999.

9. References

9.1. 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, 23 November 2020, <https://datatracker.ietf.org/doc/html/draft-ietf-nfsv4-rpc-tls-11>.

[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/rfc/rfc2119>.

[RFC5531] Thurlow, R., "RPC: Remote Procedure Call Protocol Specification Version 2", RFC 5531, DOI 10.17487/RFC5531, May 2009, <https://www.rfc-editor.org/rfc/rfc5531>.

[RFC5929] Altman, J., Williams, N., and L. Zhu, "Channel Bindings for TLS", RFC 5929, DOI 10.17487/RFC5929, July 2010, <https://www.rfc-editor.org/rfc/rfc5929>.

[RFC7942] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", BCP 205, RFC 7942, DOI 10.17487/RFC7942, July 2016, <https://www.rfc-editor.org/rfc/rfc7942>.

[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/rfc/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/rfc/rfc8881>.

9.2. Informative References

[RFC1094] Nowicki, B., "NFS: Network File System Protocol specification", RFC 1094, DOI 10.17487/RFC1094, March 1989, <https://www.rfc-editor.org/rfc/rfc1094>.

[RFC1813] Callaghan, B., Pawlowski, B., and P. Staubach, "NFS Version 3 Protocol Specification", RFC 1813, DOI 10.17487/RFC1813, June 1995, <https://www.rfc-editor.org/rfc/rfc1813>.

[RFC2743] Linn, J., "Generic Security Service Application Program Interface Version 2, Update 1", RFC 2743, DOI 10.17487/RFC2743, January 2000, <https://www.rfc-editor.org/rfc/rfc2743>.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, DOI 10.17487/RFC4303, December 2005, <https://www.rfc-editor.org/rfc/rfc4303>.

[RFC5040] Recio, R., Metzler, B., Culley, P., Hilland, J., and D. Garcia, "A Remote Direct Memory Access Protocol Specification", RFC 5040, DOI 10.17487/RFC5040, October 2007, <https://www.rfc-editor.org/rfc/rfc5040>.

[RFC5056] Williams, N., "On the Use of Channel Bindings to Secure Channels", RFC 5056, DOI 10.17487/RFC5056, November 2007, <https://www.rfc-editor.org/rfc/rfc5056>.

[RFC6071] Frankel, S. and S. Krishnan, "IP Security (IPsec) and Internet Key Exchange (IKE) Document Roadmap", RFC 6071, DOI 10.17487/RFC6071, February 2011, <https://www.rfc-editor.org/rfc/rfc6071>.

[RFC6973] Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., Morris, J., Hansen, M., and R. Smith, "Privacy Considerations for Internet Protocols", RFC 6973, DOI 10.17487/RFC6973, July 2013, <https://www.rfc-editor.org/rfc/rfc6973>.

[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/rfc/rfc7530>.

[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/rfc/rfc8166>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/rfc/rfc8446>.

Acknowledgments

David Noveck is responsible for the basic architecture of this proposal. The author is also grateful to Bill Baker, Rick Macklem, Greg Marsden, and Martin Thomson for their input and support.

Special thanks to Transport Area Directors Martin Duke and Zaheduzzaman Sarker, NFSV4 Working Group Chairs David Noveck and Brian Pawlowski, and NFSV4 Working Group Secretary Thomas Haynes for their guidance and oversight.

Author's Address

Charles Lever
Oracle Corporation
United States of America

Email: chuck.lever@oracle.com