Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 23, 2012

J. Howlett
JANET(UK)
S. Hartman
Painless Security

October 21, 2011

Key Negotiation Protocol (KNP)
draft-howlett-radsec-knp-02

Abstract

The Key Negotiation Protocol enables an untrusting RADIUS client and RADIUS server to derive a key by reference to a mutually trusted actor called the Introducer. This key may subsequently be used for one of two purposes. First, it can credential a TLS PSK ciphersuite applied to a RadSec connection between the RADIUS client and RADIUS server; or secondly, to establish a trust relationship between the RADIUS client and a second Introducer that is trusted by the first Introducer.

The composition of these capabilities enables a RADIUS client to establish a RadSec connection with any RADIUS server with whom it shares a direct or indirect trust relationship via one or more Introducers.

Status of This Memo

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

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

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

This Internet-Draft will expire on April 23, 2012.

Copyright Notice

Table of Contents

1. Introduction

TLS encryption for RADIUS (RadSec) [I-D.ietf-radext-radsec] provides a mechanism for securing the communication between a RADIUS [RFC2865] client and server on the transport layer by using TLS [RFC5246].

RadSec mandates the use of one of the [RFC5246] ciphersuites and recommends the use of two other ciphersuites specified in that document. However any ciphersuite, including the TLS Pre-Shared Key (PSK) ciphersuites [RFC4279], may be used providing that it supports encryption.

The Key Negotiation Protocol enables an untrusting RADIUS client and RADIUS server to derive a key by means of a mutually trusted actor called the Introducer. This key may subsequently for two purposes.

First, the key can be used to credential a TLS PSK ciphersuite when applied to a RadSec connection between the RADIUS client and RADIUS server, permitting a trusted exchange of RADIUS messages in the absence of a pre-existing relationship between the RADIUS client and RADIUS server. This is described as "Credentialing".

Secondly, the key can used as a credential by a RADIUS client to establish a trust relationship with a second Introducer that happens to be trusted by the first Introducer. This is described as "Introduction".

The composition of Credentialing and Introduction enables a RADIUS client to establish a RadSec connection with any RADIUS server with whom it shares an indirect trust relationship via one or more Introducers.

2. Motivation

The KNP is motivated by the following requirements:

- In the case of a non-federated RADIUS environment where a RADIUS client and RADIUS AS.server shares a direct trust relationship, a shared secret credential is used as the trust anchor between these systems. In transitioning to the use of RadSec, it may be more convenient if these systems are able to continue using the existing credential technology rather than introduce a new credential technology (such as X.509 certificates), as this may impose significant changes to operational practices (such as deploying a Public Key Infrastructure).

- In the case of a federated RADIUS environment where RADIUS clients and RADIUS servers are associated with different domains,

transitioning to the use of RadSec may impose a requirement to distribute and manage multiple trust anchors. It may be more convenient if the systems within these domains were able to use a single trust anchor for RADIUS systems in all other domains, in addition to those systems within its own domain. This may facilitate the scaling of large heterogeneous RADIUS environments where it may be difficult - for technical and/or administrative reasons - to impose support for even a small set of trust anchors.

- The use of multiple trust anchors within complex federated environments may impede essential trust management functions such as timely revocation. Reducing the number of trust anchors may therefore improve trust management within these environments, particularly if it can be reduced to a single trust anchor.

3. Overview

The Key Negotiation Protocol (KNP) enables a RADIUS client and RADIUS server that do not share a direct trust relationship to derive a shared key by virtue of both systems having a trust relationship with an EAP server called the Introducer. This key may be used for the following purposes:

1. Credentialing: the RADIUS client and RADIUS server can use the key to credential a TLS PSK ciphersuite applied to a RadSec connection.

2. Introduction: a credential can be derived from the key that can be used to authenticate the RADIUS client against a second Introducer that is trusted by the first Introducer.

The composition of these capabilities enables a RADIUS client to derive a key that can be used to credential a RadSec connection with any other RADIUS server with whom it shares a common Introducer and, through transitivity, any number of intermediate Introducers.

This transitivity of trust between a RADIUS client and RADIUS server across a chain of intermediate Introducers may appear very similar to the use of RADIUS proxies to establish a chain of trust between a RADIUS client and RADIUS server. There is however a very significant difference:

- In the case of RADIUS proxy, the RADIUS messages emitted by the RADIUS client and RADIUS server must transit through the intermediate RADIUS proxy(ies). There is no end-to-end relationship between the RADIUS client and RADIUS server, either in terms of connectivity or trust.

- In the case of KNP, the RADIUS messages are able to transit directly between RADIUS client and RADIUS server. The path of transmissions between these systems is therefore entirely decoupled from the path of trust . There is an end-to-end relationship between the RADIUS client and RADIUS server, both in terms of connectivity and trust.

The use of RADIUS Proxies and Introducers are not mutually exclusive; deployers may choose to use both.

4. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

5. The KNP Actors

In the KNP, the RADIUS client and RADIUS server do not initially share a trust relationship. Instead, these actors share a trust relationship with a mutually trusted third party known as the "Introducer".

Figure 1 below depicts the trust relationships for a RADIUS client, RADIUS server and Introducer before the KNP has been invoked.

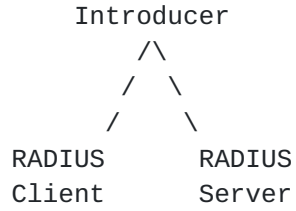

Figure 1

Figure 2 below depicts the new trust relationship between the RADIUS client, RADIUS server and Introducer after the KNP has been invoked.

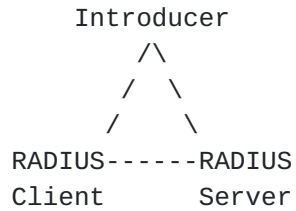

Figure 2

Figure 3 below depicts the flow of RADIUS packets from the RADIUS

client to the RADIUS server using the new trust relationship.

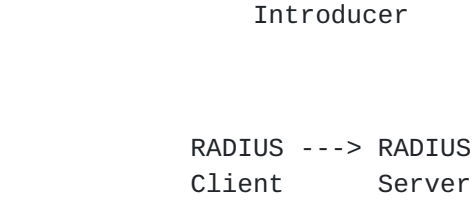

Figure 3

Note that the RADIUS messages are not routed by the Introducer, as they would in the case of a RADIUS Proxy. Instead, they flow directly from RADIUS client to RADIUS server.

6. Relationships With Other Protocols

The KNP builds on a variety of protocols. This section describes the relationship of KNP to these.

6.1. Relationship to EAP

In the KNP the RADIUS client assumes the role of an EAP peer. In this role, it attempts to authenticate against a RADIUS server that assumes the role of a pass-through EAP authenticator. An EAP server acts as the Introducer.

The KNP enables all three actors - RADIUS client (EAP peer), RADIUS server (EAP authenticator) and Introducer (EAP server) - to establish a common view of their mutual relationships as described by the EAP names and keys that the EAP exchange yields, using the norms established by the EAP Key Management Framework [RFC5247].

The RADIUS client must possess an EAP credential for the Introducer, allowing mutual authentication of both parties.

Figure 4 below depicts the relationships between these actors:

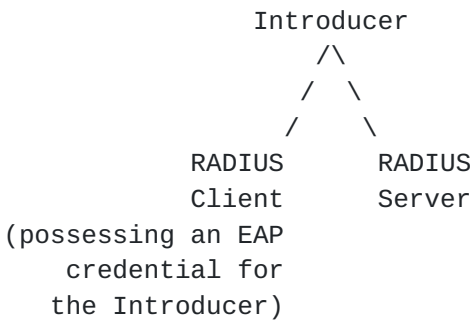

Figure 4

6.2. Relationship to RADIUS

The RADIUS server uses the RADIUS protocol to forward the EAP transaction to the Introducer.

The RADIUS server must share a RADIUS secret with the Introducer, allowing mutual authentication of both actors.

Figure 5 below depicts the relationships between these actors:

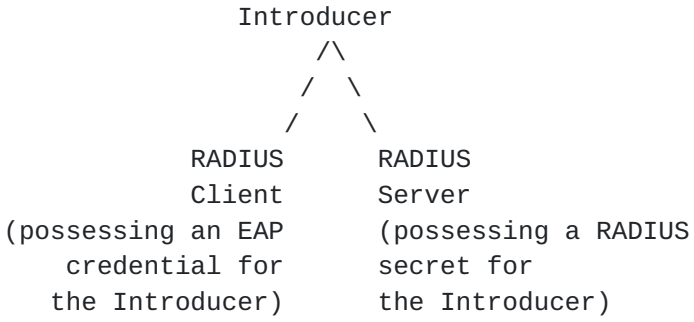

Figure 5

6.3. Relationship to the GSS API

The KNP builds on the GSS API [RFC2743] framework by using the GSS EAP mechanism [I-D.ietf-abfab-gss-eap] and EAP [RFC3748]. The GSS EAP tokens are transported between the RADIUS client and RADIUS server using the HTTP Negotiate authentication scheme [RFC4559].

6.4. Relationship to the HTTP

The KNP uses HTTP to transport its request and response protocol messages between the RADIUS Client and RADIUS server.

7. Key Negotiation Protocol

As described previously, the KNP provides two operations: Credentialing and Introduction.

The KNP provides these operations using a common protocol pattern. This pattern is applied against two types of Target actor, depending on the operation in question:

- In the case of Credentialing, the Target actor is a RADIUS server. If Credentialing is successful, the RADIUS client and RADIUS server will derive a common PSK that can be applied with a TLS-PSK

ciphersuite and RadSec.

o In the case of Introduction, the Target actor is an Introducer. If an Introduction is successful, the RADIUS client and Introducer will derive an EAP credential that can subsequently be used for subsequent Credentialing or Introduction operations.

For both operations it is essential that all three actors - RADIUS Client, Introducer and Target (whether a RADIUS server, in the case of Credentialing, or another Introducer, in the case of Introduction) - are able to authorise the claims that the other actors make about their respective names. These claims are validated using different processes for each relationship; these are summarised in Figure 6 below.

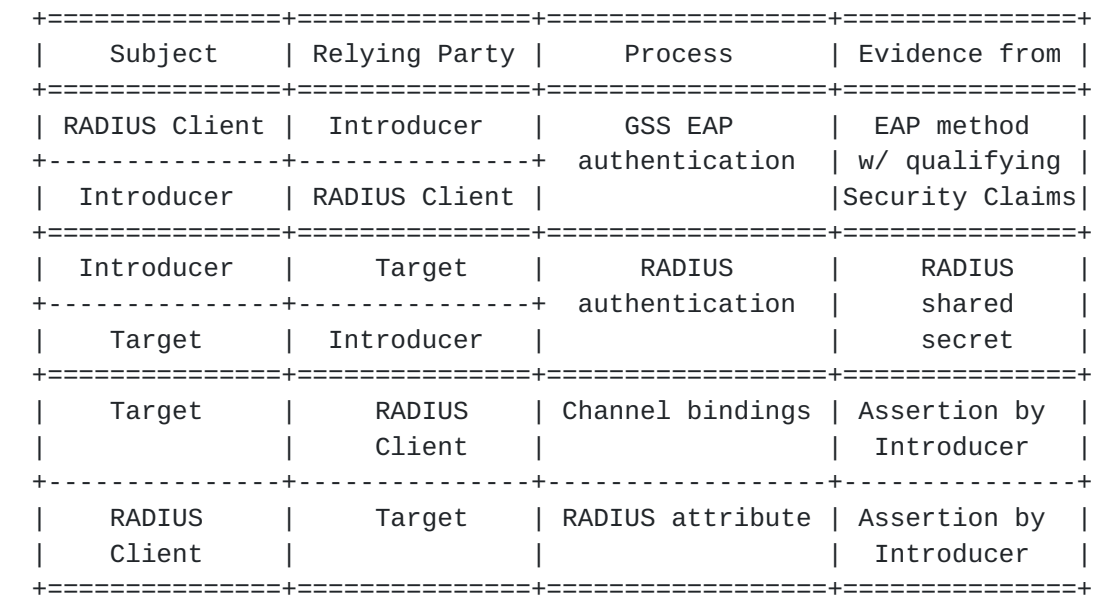

Figure 6

7.1. Operation Independent Flow

The RADIUS Client invokes the KNP by establishing an HTTP connection with the Target's KNP end-point.

The RADIUS Client MUST use the GSS EAP mechanism [I-D.ietf-abfab-gss-eap] to authenticate to the Introducer, requesting mutual authentication from the GSS layer.

The RADIUS Client, Target and Introducer MUST support EAP channel bindings [I-D.ietf-emu-chbind]. The Introducer MUST use validate the EAP channel bindings [I-D.ietf-emu-chbind] provided by the RADIUS Client. If the channel binding verification fails, the Introducer MUST reject the authentication.

The completion of the EAP method exchange results in the derivation of an EAP MSK known only to the RADIUS Client and Introducer and Peer-Id(s) and Server-Id(s) identifying these respectively. The Introducer MUST replicate the keying material and Server-Id to the Target.

The RADIUS Client and Target, in possession of the EAP MSK, create a GSS-API security context as described in section 6 of [I-D.ietf-abfab-gss-eap].

The RADIUS Client POSTs a key negotiation request, encoded as an HTML form dataset, to the Target. This request contains a set of operation-specific controls that specifies key negotiation parameters. A key negotiation request MUST contain the following controls:

- Version: the version of the KNP.
- Request-Identifier: a unique alphanumeric identifier for the request.
- Requestor-Name: the requestor's GSS EAP initiator name.
- Operation-Type: the type of operation.
- Authenticator-Type: message authentication algorithm.
- Authenticator-Value: message authenticator value.

The Target extracts the key negotiation parameters and assesses their compliance to the Target's key negotiation policies. The Target MUST return an operation-specific document providing information about the resulting key negotiation context.

- Version: the version of the KNP.
- Request-Identifier: the identifier for the request that this is a reponse to.
- Responder-Name: the requestor's GSS EAP acceptor name.
- Operation-Type: the type of operation.
- Status-Code: a status code.
- Expires-After: a timestamp indicating the time of expiration.

- Authenticator-Type: message authentication algorithm.

- Authenticator-Value: message authenticator value.

TODO: consider use of SAML authentication assertion instead?

The RADIUS server and client SHOULD cache the GSS context until expiry of the GSS context. However, either party MAY delete a GSS context at any time. If a GSS context is deleted, any operation-specific derived materials SHOULD also be deleted, although such materials MAY be retained for auditing purposes.

7.2. The Credentialing Operation

This section describes the Credentialing operation-specific extensions to the general KNP flow.

The RADIUS Client MUST specify the following control values within the key negotiation request:

- Operation-Type: Credentialing

The PSK identity and value shall be outputs of GSS_Pseudo_random() [RFC4401] using the Pseudo-Random Function defined for the GSS EAP mechanism [I-D.ietf-abfab-gss-eap].

For the PSK identity, the prf_in input string MUST be prepended with the string "tls-psk-knp-identity"; desired_out_len MUST be set to 128 octets.

For the PSK value, the prf_in input string MUST be prepended with the string "tls-psk-knp-value"; desired_out_len MUST be set to 64 octets.

Note: these output values should use base64 encoding

7.3. The Introduction Operation

This section describes the Introduction operation-specific extensions to the general KNP flow.

The RADIUS Client MUST specify the following control values within the key negotiation request:

- Operation-Type: Introduction"

The EAP identity and credential shall be outputs of GSS_Pseudo_random() [RFC4401] using the Pseudo-Random Function defined for the GSS EAP mechanism [I-D.ietf-abfab-gss-eap].

For the EAP identity, the prf_in input string MUST be prepended with the string "tls-psk-eap-identity"; desired_out_len MUST be set to 128 octets. The output string MUST be appended with the realm of the Introducer to form an NAI.

For the EAP credential, the prf_in input string MUST be prepended with the string "tls-psk-eap-psk"; desired_out_len MUST be set to 64 octets.

Note: these output values should use base64 encoding.

8. Security Considerations

TODO

9. IANA Considerations

TODO

10. Acknowledgements

TODO

11. Normative References

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

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

[RFC2865] Rigney, C., Willens, S., Rubens, A., and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, June 2000.

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, "Extensible Authentication Protocol (EAP)", RFC 3748, June 2004.

[RFC4279] Eronen, P. and H. Tschofenig, "Pre-Shared Key Ciphersuites for Transport Layer Security (TLS)", RFC 4279, December 2005.

[RFC4401] Williams, N., "A Pseudo-Random Function

(PRF) API Extension for the Generic Security Service Application Program Interface (GSS-API)", RFC 4401, February 2006.

[RFC4559] Jaganathan, K., Zhu, L., and J. Brezak, "SPNEGO-based Kerberos and NTLM HTTP Authentication in Microsoft Windows", RFC 4559, June 2006.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[RFC5247] Aboba, B., Simon, D., and P. Eronen, "Extensible Authentication Protocol (EAP) Key Management Framework", RFC 5247, August 2008.

[I-D.ietf-abfab-gss-eap] Hartman, S. and J. Howlett, "A GSS-API Mechanism for the Extensible Authentication Protocol", draft-ietf-abfab-gss-eap-03 (work in progress), October 2011.

[I-D.ietf-radext-radsec] Winter, S., McCauley, M., Venaas, S., and K. Wierenga, "TLS encryption for RADIUS", draft-ietf-radext-radsec-09 (work in progress), July 2011.

[I-D.ietf-emu-chbind] Hartman, S., Clancy, T., and K. Hoeper, "Channel Binding Support for EAP Methods", draft-ietf-emu-chbind-10 (work in progress), October 2011.

Authors' Addresses

Josh Howlett
JANET(UK)
Lumen House, Library Avenue, Harwell
Oxford OX11 0SG
UK

Phone: +44 1235 822363
EMail: Josh.Howlett@ja.net

Sam Hartman
Painless Security

EMail: hartmans-ietf@mit.edu