Workgroup: OPSAWG WG
Internet-Draft:
draft-vishwakarma-opsawg-ssh-cert-radius-02
Published: 28 December 2021
Intended Status: Informational
Expires: 1 July 2022
Authors: Devendra Vishwakarma   Prakash Suthar
Cisco Systems   Google Inc.
Vivek Agarwal   Anil Jangam, Ed.
Cisco Systems   Cisco Systems

RADIUS Extension for Certificate-based SSH Authentication

Abstract

A scalable and centralized mechanism is required for a certificate-based administrative access to multitude of virtualized and physical network functions. While there are mechanisms that exist today to provide secure administrative command-line and API-based access, there are certain management and maintenance overheads as well as certain scalability challenges related to it. In this draft we discuss these challenges and propose a standardized, centralized server-based mechanism to authenticate a user over an SSH session using its client certificate.

Status of This Memo

Copyright Notice

Table of Contents

1. Conventions and Terminology

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

This document uses the terminology of [RFC3987] and [RFC3986] for RADIUS entities.

2. Introduction

With the pervasive use of virtualized infrastructure (e.g., microservices-based application designs), a high magnitude of individual and autonomous software application components are working together to realize a complete system functionality. With a large number of highly interacting agents, an authentication and authorization mechanism which is scalable and flexible is very critical for administrative access. A typical service authentication and authorization (AAA) infrastructure comprise of an identity management, verification and, validations functions.

In a typical day of an IT (Information Technology) enabled organization, IT engineers often connect to many different servers while carrying their tasks such as, change of configurations, write a software code, save a file, fetch an image, etc. There are mainly three different ways engineers can authenticate to the servers they are connecting to.

*Password based

*RSA key based

*OpenSSH certificate based local authentication

While these methods are currently being used, they suffer from the following limitations respectively.

*Password based: In this method, user needs to enter the username and password each time it tries to login to a server. With the increasing frequency and number of servers, the manual configuration becomes untenable. While script-based automation is an option, it is highly insecure as passwords are stored in cleartext. Also, a frequent password changes and adherence to complex password policies are required to prevent against any misuse. A 2-factor authentication mechanism provides some protection but it still involves a certain level of human interaction, which is difficult to automate.

*The RSA key-based authentication requires a frequent copy of files to each device, server, cloud-native network function (CNF), and virtual network functions (VNFs) and hence is not scalable. In addition, if a key gets compromised, the revocation of it from all servers and VNFs involves a lot of effort.

*OpenSSH certificate based local authentication requires root certification authority (CA) certificate to be copied to each individual device, server, and the VNF. If the developers are using client certificate from multiple certification authorities, all of the certification authority (CA) root certificate and intermediate certificate has to be installed on each of the servers that's being accessed. Also, a connectivity to the certificate revocation list (CRL) is required from all the devices, servers, and VNFs to check for revoked certificates. In a typical customer environment all the device, servers, and VNFs do not have access to a public CRL location. Also, any changes in the certificates (e.g. expiry or revocation) requires a manual or a script based cleanup of certificates from all the servers.

In order to address these limitations and move towards a password-less mechanism, we propose an approach that uses certificate based SSH authentication using RADIUS protocol. The centralized server-based system also helps solve all the above outlined limitations.

2.1. Centralized RADUIS Server based Solution

As shown in Figure 1, a method is devised to authenticate SSH sessions using client certificates, where the device, server, VNF, and CNF uses RADIUS protocol to validate the authenticity of the certificate from a centralized RADIUS server. The RADIUS server will get the username from the CN (common name) field in the client certificate.

The benefits of using certificates for SSH session authentication are as follows:

*It does not require a frequent client certificate replacement (e.g., a certificate is typically valid for a year), which solves the problem seen in the password-based authentications.

*It does not require client public keys to be copied to each device, server, VNF, or CNF that needs to be accessed.

*It doesn't need any secondary out-of-band authentication like OTP or a complex MFA (Multi-factor Authentication) solution.

*All the root certificates and intermediate certificates needs to be installed on the RADIUS server only. This makes it easy for many administrative tasks including initial setup, adding a new CA, retiring of an old CA, certificate revocation check, and the centralized access to the internet to download the revocation list.

*Both the certificate validation and revocation check happen at a centralized AAA server.

```
                                  +----+   +----+
                                  |VNF1|   |VNF2|
                                  +--+-+   +--+-+
  +-----------+      ;-----;         |        |      +---------------+
  |           |     /       \     +--+--------+-+    | Radius Server |
  | Endpoint  |--->( Network )--->|     NFVI    |--->| (AAA Server)  |
  |           |     \       /     +-------------+    +-------+-------+
  +-----------+      `-----'                                 |
                                                             |
                          +--------------------+             |
                          |  Service Provider  |<------------+
                          | Cert Authority (CA)|
                          +--------------------+
```

Figure 1: Centralized RADIUS-based AAA System

3. Operational Details

Operation is identical to that defined in [RFC2865], [RFC5216], and [RFC4252].

3.1. Basic Setup

Analogous to 802.1X authentication where there is an EAP Supplicant (also known as EAP peer), pass-through authenticator, and a RADIUS server. This solution has an SSH client that is similar to EAP supplicant, an SSH server similar to pass-through authenticator, and a RADIUS server. The SSH transport protocol defined in RFC 4253 MUST be used as the lower layer of the EAP. The SSH transport protocol satisfies all the requirements of the EAP lower layer defined in the section 3.1 of the RFC 3748.

*Unreliable transport: Even though the EAP assumes that the lower layer can be a unreliable transport and has retransmission built into EAP itself. The SSH transport protocol is a reliable transport protocol and handles its own retransmission when used over TCP/IP. RFC 793 section 3.7 has the details of data communication and retransmission behavior of TCP.

*Lower layer error detection: Error detection must be provided by the EAP lower layer and SSH TP does that using the sequence numbers in the TCP packets. This is detailed in the section 3.3 of the RFC 793.

*Lower layer security: EAP does not require lower layers to provide security services, however SSH TP over TCP provides for

*Data integrity: as detailed in section 6.4 of RFC 4253 and section 9.3.2 of RFC 4251

*Replay protection: as detailed in section 9.3.3 of RFC 4251

*Authentication: as detailed in section 9.4 of RFC 4251

*Confidentiality: as detailed in section 6.3 of RFC 4253 and section 9.3.1

*Minimum MTU: EAP requires the lower layer to support 1020 bytes or higher MTU. SSH TP satisfies this requirement. In a typical TCP/IP network the MTU is by default set to 1500 bytes which satisfies the requirement for EAP.

*Possible duplication: while it is desirable that lower layers provide for non-duplication, this is not a requirement.

*Ordering guarantees: Lower layer transports for EAP MUST preserve ordering between a source and destination at a given priority level (the ordering guarantee provided by [IEEE-802]). SSH TP when used over TCP/IP guarantees ordered delivery of data from source to destination. Section 2.6 of RFC 793 details the use of sequence numbers in TCP.

The SSH client MUST support certificate-based authentication for the SSH session. The SSH client MUST also have a X.509 client certificate installed on the operating system. The client certificate MUST have "client authentication" as value in the enhanced key usage field of the certificate. This will ensure that the client is ready to complete SSH authentication using the installed X.509 client certificate.

The SSH server MUST be configured to send SSH authentication requests to a RADIUS server.

The RADIUS server MUST have an X.509 server certificate installed on the operating system. The server certificate MUST have ''server authentication" as value in the enhanced key usage field of the certificate. This will ensure that the RADIUS server is ready to authenticate SSH clients using certificates. The RADIUS server MUST also be configured to do EAP-TLS authentication as described in [RFC3748].

3.2. EAP TLS authentication

Although other inner methods of EAP could be supported for authentication here such as EAP-MSCHAP, EAP-MD5 or EAP-FAST etc, however they do not provide much benefit over the current password based authentication that exist. This draft only focuses on the EAP-TLS inner method as that gives the ability to allow certificate based authentication.

The SSH client will initiate an SSH connection to the SSH server. The SSH server drives the authentication by telling the SSH client which authentication methods can be used during the session.

The SSH client MUST choose a client certificate installed in the operating system as described in the section 2.1 Basic Setup. If there are multiple client certificates then the SSH client SHOULD choose a client certificate. If there is no certificate installed on the SSH client, then the client MAY choose another authentication methods defined in [RFC4251].

The SSH client initiates an SSH session to the SSH server. The SSH server upon receiving the connection request MUST initiate the EAP-TLS authentication by sending an EAP identity request to the SSH client.

The SSH client picks the common name from the user certificate and sends that as the EAP identity back to the SSH server.

3.3. RADIUS Access Request

The SSH server constructs a RADIUS authentication request and MUST set the service type = Cert Login. This service type will be an indication to the RADIUS server to use EAP-TLS authentication method for that SSH session.

The RADIUS server MUST use EAP-TLS authentication for this session. RADIUS server sends a response back to the SSH server as Radius Access Challenge(EAP-Message(Code=Request TYPE=TLS EAP(EAP-TLS)))

The SSH server strips the EAP message from the RADIUS packet received and forwards it to the SSH client. The message is (EAP-Message(Code=Request TYPE=TLS EAP(EAP-TLS)).

The SSH client starts the TLS handshake by sending the EAP-Message(TLS Client Hello) to the SSH server. The SSH server takes the EAP message received in the previous step and wraps it in the RADIUS access request and sends it to the RADIUS server. The message is Radius Access Request(EAP-Message(TLS Client Hello)).

Upon receipt of this message the RADIUS server processes the client hello message and sends a reply back to the SSH server with server hello, server certificate, server key exchange and certificate request. The message is Radius Access Challenge(EAP-Message(Server Hello, Server Certificate, Server Key exchange, Certificate Request).

The SSH server strips the EAP message from the RADIUS packet received in the previous step and forwards it to the SSH client. The message is EAP-Message(Server Hello, Server Certificate, Server Key exchange, Certificate Request). The SSH client validates the server root certificate installed. The SSH client moves ahead in the TLS handshake process and sends the client certificate in a message to the SSH server EAP-Message(TLS Client Certificate, TLS Client key exchange, TLS Certificate Verify)).

The SSH server takes the EAP message received in the previous step and wraps it in the RADIUS access request and sends it to the RADIUS server. The message is Radius Access Request(EAP-Message(TLS Client Certificate, TLS Client key exchange, TLS Certificate Verify)).

The RADIUS server validates the client certificate using the root CA certificate chain. RADIUS server sends a TLS finished message to the SSH server. The message is Radius Access Challenge(EAP-Message(TLS Finished)).

The SSH server strips the EAP message from the RADIUS packet received in the previous step and forwards it to the SSH client. The message is EAP-Message(TLS Finished). The SSH client moves ahead in the EAP phase and sends the next message. The message is EAP-Message(TYPE=EAP-TLS)).

The SSH server takes the EAP message received in the previous step and wraps it in the RADIUS access request and sends it to the RADIUS server. The message is Radius Access Request(EAP-Message(TYPE=EAP-TLS)).

RADIUS server processes the previous request and at this the EAP authentication is successful. The message sent back to the SSH server is Radius Accept(EAP-Message(EAP-Success)). The SSH server strips the EAP message from the RADIUS packet received in the previous step and forwards it to the SSH client. The message is EAP-Message(EAP-Success).

The SSH session is established at this point. A message to this effect is sent to the SSH client from the SSH server.

3.4. Radius Accounting Request

```
Endpoint                  VNF/CNF                 Radius Server
(SSH Client)           (SSH/EAP Server)           (AAA Server)
+---+------+           +--------+-----+           +-------+---+
    |                           |                         |
    |   SSH Request to VM       |                         |
    |-------------------------->|                         |
    |                           |                         |
    |   EAP Identity Request    |                         |
    |<--------------------------|                         |
    |                           |                         |
    |   EAP Identity Response   |                         |
    |    (CN from User Cert)    |                         |
    |-------------------------->|                         |
    |                           | Radius Access Request   |
    |                           | (ST: Cert Login)        |
    |                           | EAP Msg: Identity       |
    |                           | (ID: CN from User Cert) |
    |                           |------------------------>|
    |                           |                         |
    |                           | Radius Access Challenge |
    |                           | (EAP Message            |
    |                           |  RT: TLS EAP(EAP-TLS)   |
    |   (EAP Message            |<------------------------|
    |    RT: TLS EAP(EAP-TLS)   |                         |
    |<--------------------------|                         |
    |                           |                         |
    |    EAP Message            |                         |
    |    (TLS Client Hello)     |                         |
    |-------------------------->|                         |
    |                           | Radius Access Request   |
    |                           | EAP-Message:            |
    |                           | (TLS client hello)      |
    |                           |------------------------>|
    |                           |                         |
    |                           |             Cert Login  |--+
    |                           |             supported   |  |
    |                           |                         |<-+
    |                           |                         |
    |                           | Radius Access Challenge |
    |                           | (EAP Message:           |
    |                           |  Server Hello,          |
    |   (EAP Message:           |  Server Cert,           |
    |    Server Hello,          |  Key Exchange,          |
    |    Server Cert,           |  Cert Request)          |
    |    Key Exchange,          |<------------------------|
    |    Cert Request)          |                         |
    |<--------------------------|                         |
    |                           |                         |
    |--+ Server Cert            |                         |
    |  | Validation             |                         |
    |<-+                        |                         |
    |                           |                         |
    |  EAP message              |                         |
    |  (TLS Client Certificate  |                         |
    |   TLS Client Key Xchange  |                         |
    |   TLS Cert Verify)        | Radius Access Request   |
    |--------------------------->| (EAP message            |
    |                           | (TLS Client Certificate |
    |                           | (TLS Client Key Xchange |
    |                           |  TLS Cert Verify)       |
    |                           |-------------------------|
    |                           |                         |
    |                           |             Client Cert |--+
    |                           |             Validated   |  |
    |                           |                         |<-+
    |                           |                         |
    |                           |                         |
    |                           | Radius Access Challenge |
    |                           | (EAP Message:           |
    |                           |  TLS Finished)          |
    |                           |<------------------------|
    |       TLS Finished)       |                         |
    |<--------------------------|                         |
    |                           |                         |
    |      EAP Message          |                         |
    |     (Type: EAP-TLS)       |                         |
    |--------------------------->|                         |
    |                           | Radius Access Request   |
    |                           | EAP Msg(Type: EAP-TLS)  |
    |                           |------------------------>|
    |                           |                         |
    |                           | Radius Access Accept    |
    |                           | (EAP Message:           |
    |                           |  EAP-Success)           |
    |                           |<------------------------|
    |       EAP Success         |                         |
    |<--------------------------|                         |
    |                           |                         |
    |    Session Established    |                         |
    |<--------------------------|                         |
    |                           |                         |

    Legends: CN: Common Name ST: Service Type RT: Request Type
```

Figure 2: Radius Accounting Request Flow

3.5. Radius Access Accept

As shown in Figure 2, when the Radius server supports the new service-type, it sends a Radius Access Accept message. In case where server does not support the certificate-based login type, it responds with Radius Access Reject response indicating the new login not supported. The corresponding call flow is shown in Figure 3.

```
     Endpoint                  VNF/CNF               Radius Server
     (SSH Client)           (SSH/EAP Server)           (AAA Server)
     +---+------+           +--------+-----+           +-------+---+
         |                           |                         |
         |   SSH Request to VM       |                         |
         |-------------------------->|                         |
         |                           |                         |
         |   EAP Identity Request    |                         |
         |<--------------------------|                         |
         |                           |                         |
         |   EAP Identity Response   |                         |
         |    (CN from User Cert)    |                         |
         |-------------------------->|                         |
         |                           | Radius Access Request   |
         |                           | (ST: Cert Login)        |
         |                           | EAP Msg: Identity       |
         |                           | (ID: CN from User Cert) |
         |                           |------------------------>|
         |                           |                         |
         |                           | Radius Access Reject    |
         |                           | (Login Not Supported*)  |
         | (EAP Failure              |<------------------------|
         | Cert login Not Supported) |                         |
         |<--------------------------|                         |
         |                           |                         |

         Legends: CN: Common Name ST: Service Type RT: Request Type
```

Figure 3: AAA Server Does not Support Service Type

4. Protocol Details

Operation is identical to that defined in [RFC2865], [RFC5216], and [RFC4252].

4.1. Packet Format

4.2. Packet Types

The RADIUS Packet type is determined by the 'Code' field in the first octet of the packet.

4.2.1. Access Request

The Access-Request packet type is same as defined in [RFC2865]. Here is a summary of the Access-Request packet format as shown in Figure 4. The fields are transmitted from left to right.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Code      |  Identifier   |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                     Request Authenticator                     |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Attributes ...
+-+-+-+-+-+-+-+-+-+-+-+-+-
```

Figure 4: Access Request Packet Format

Within the same framework, this draft adds a new service-type attribute value that will be sent in the Access-Request packet.

4.3. Attributes

4.3.1. Service Type

The 'Type' attribute value will have the same format as the service-type field defined in [RFC2865] and is shown in Figure 5. The fields are transmitted from left to right.

```
  0                   1                   2                   3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Length     |             Value             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Value (cont.)                 |
|-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5: Service Type Encoding

Type

*6 for Service-Type

Length

*6 bytes

Value: the current types supported are as follows.

*Login

*Framed

*Callback Login

*Callback Framed

*Outbound

*Administrative

*NAS Prompt

*Authenticate Only

*Callback NAS Prompt

*Call Check

*Callback Administrative

This draft introduces a new service-type value as below.

*Cert Login

The Cert Login value shall be used by AAA Client in an Access-Request packet to indicate to the RADIUS server that EAP-TLS authentication needs to be used for this session. It is recommended that the endpoint shall have a client certificate installed and ready to be used during the authentication.

5. IANA Considerations

This section provides guidance to the Internet Assigned Numbers Authority (IANA) regarding registration of values related to the RADIUS protocol, in accordance with [RFC8126].

A new attribute is proposed to be added to the RADIUS Radius Access Request in the Service-Type Enum.

6. Security Considerations

The user certificate used by the clients must be stored in a non-shared location of the operating system. This will ensure that the users on the same system are not able to use each other certificates.

All the security considerations apply from the RFC 2865 as well.

7. Summary

A scalable, centralized, and standard-based method for management of user login authentication is described. The proposal comprise of an enhancement to the RADIUS protocol message and certain server side enhancements shall be required to support the new functionality. Once implemented, the enhanced server shall provide an improved user experience involving a high frequency and a high scale of user authentication across a range of interconnected agents (e.g. client and servers). The enhancement provides an improved configuration and management of the authentication infrastructure and reduces the overhead related to deployment of root and intermediate certificates at individual network nodes. This enhancement not only makes the initial setup easier, but also revocation check easier due to the centralized design. Addition and retirement of root and intermediate certificates are among the most time saving aspects of the proposed enhancement.

8. Acknowledgments

We are grateful for the input from IESG members and from a number of individual members of the IETF community who share our concern for doing the right thing.

9. References

9.1. Normative References

[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>.

[RFC2865] Rigney, C., Willens, S., Rubens, A., and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, DOI 10.17487/RFC2865, June 2000, <https://www.rfc-editor.org/info/rfc2865>.

[RFC3748]
Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, Ed., "Extensible Authentication Protocol (EAP)", RFC 3748, DOI 10.17487/RFC3748, June 2004, <https://www.rfc-editor.org/info/rfc3748>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", January 2005, <https://tools.ietf.org/html/rfc3986#section-3.3>.

[RFC4251] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Protocol Architecture", RFC 4251, DOI 10.17487/RFC4251, January 2006, <https://www.rfc-editor.org/info/rfc4251>.

[RFC4252] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Authentication Protocol", RFC 4252, DOI 10.17487/RFC4252, January 2006, <https://www.rfc-editor.org/info/rfc4252>.

[RFC5216] Simon, D., Aboba, B., and R. Hurst, "The EAP-TLS Authentication Protocol", RFC 5216, DOI 10.17487/RFC5216, March 2008, <https://www.rfc-editor.org/info/rfc5216>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

9.2. Informative References

[RFC3987] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", January 2005, <https://tools.ietf.org/html/rfc3986#section-3.3>.

Authors' Addresses

Devendra Vishwakarma
Cisco Systems
Apex, North Carolina 27523
United States of America

Email: dvishwak@cisco.com

Prakash Suthar
Google Inc.
Mountain View, California 94043
United States of America

Email: psuthar@google.com

Vivek Agarwal

Cisco Systems
Apex, North Carolina 27523
United States of America

Email: vivagarw@cisco.com

Anil Jangam (editor)
Cisco Systems
San Jose, California 95134
United States of America

Email: anjangam@cisco.com