Internet Engineering Task Force S. Sorce, Ed.
Internet-Draft Red Hat
Updates: 4120 (if approved) T. Yu, Ed.
Intended status: Standards Track T. Hardjono, Ed.
Expires: March 9, 2015 MIT Kerberos Consortium
September 5, 2014

Kerberos Authorization Data Container Authenticated by Multiple MACs
draft-ietf-krb-wg-cammac-09

Abstract

Abstract: This document specifies a Kerberos Authorization Data container that supersedes AD-KDC-ISSUED. It allows for multiple Message Authentication Codes (MACs) or signatures to authenticate the contained Authorization Data elements.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This document specifies a new Authorization Data container for Kerberos, called AD-CAMMAC (Container Authenticated by Multiple MACs), that supersedes AD-KDC-ISSUED. This new container allows both the receiving application service and the Key Distribution Center (KDC) itself to verify the authenticity of the contained authorization data. The AD-CAMMAC container can also include additional verifiers that "trusted services" can use to verify the contained authorization data.

2. Requirements Language

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

3. Motivations

The Kerberos protocol allows clients to submit arbitrary authorization data for a KDC to insert into a Kerberos ticket. These client-requested authorization data allow the client to express authorization restrictions that the application service will interpret. With few exceptions, the KDC can safely copy these client-requested authorization data to the issued ticket without necessarily inspecting, interpreting, or filtering their contents.

The AD-KDC-ISSUED authorization data container specified in RFC 4120 [RFC4120] is a means for KDCs to include positive or permissive

(rather than restrictive) authorization data in service tickets in a way that the service named in a ticket can verify that the KDC has issued the contained authorization data. This capability takes advantage of a shared symmetric key between the KDC and the service to assure the service that the KDC did not merely copy client-requested authorization data to the ticket without inspecting them.

The AD-KDC-ISSUED container works well for situations where the flow of authorization data is from the KDC to the service. However, protocol extensions such as Constrained Delegation (S4U2Proxy [MS-SFU]) require that a service present to the KDC a service ticket that the KDC previously issued, as evidence that the service is authorized to impersonate the client principal named in that ticket. In the S4U2Proxy extension, the KDC uses the evidence ticket as the basis for issuing a derivative ticket that the service can then use to impersonate the client. The authorization data contained within the evidence ticket constitute a flow of authorization data from the application service to the KDC. The properties of the AD-KDC-ISSUED container are insufficient for this use case because the service knows the symmetric key for the checksum in the AD-KDC-ISSUED container. Therefore, the KDC has no way to detect whether the service has tampered with the contents of the AD-KDC-ISSUED container within the evidence ticket.

The new AD-CAMMAC authorization data container specified in this document improves upon AD-KDC-ISSUED by including additional verifier elements. The svc-verifier element of the CAMMAC has the same functional and security properties as the ad-checksum element of AD-KDC-ISSUED; the svc-verifier allows the service to verify the integrity of the AD-CAMMAC contents as it already could with the AD-KDC-ISSUED container. The kdc-verifier and other-verifiers elements are new to AD-CAMMAC and provide its enhanced capabilities.

The kdc-verifier element of the AD-CAMMAC container allows a KDC to verify the integrity of authorization data that it previously inserted into a ticket, by using a key that only the KDC knows. The KDC thus avoids recomputing all of the authorization data for the issued ticket; this operation might not always be possible when that data includes ephemeral information such as the strength or type of authentication method used to obtain the original ticket.

The verifiers in the other-verifiers element of the AD-CAMMAC container are not required, but can be useful when a lesser-privileged service receives a ticket from a client and needs to extract the CAMMAC to demonstrate to a higher-privileged "trusted service" on the same host that it is legitimately acting on behalf of that client. The trusted service can use a verifier in the other-

verifiers element to validate the contents of the CAMMAC without further communication with the KDC.

4. Encoding

The Kerberos protocol is defined in [RFC4120] using Abstract Syntax Notation One (ASN.1) [X.680] and using the ASN.1 Distinguished Encoding Rules (DER) [X.690]. For consistency, this specification also uses ASN.1 for specifying the layout of AD-CAMMAC. The ad-data of the AD-CAMMAC authorization data element is the ASN.1 DER encoding of the AD-CAMMAC ASN.1 type specified below.

```
KerberosV5CAMMAC DEFINITIONS EXPLICIT TAGS ::= BEGIN

AD-CAMMAC                   ::= SEQUENCE {
      elements              [0] AuthorizationData,
      kdc-verifier          [1] Verifier-MAC OPTIONAL,
      svc-verifier          [2] Verifier-MAC OPTIONAL,
      other-verifiers       [3] SEQUENCE (SIZE (1..MAX))
                                OF Verifier OPTIONAL
}

Verifier             ::= CHOICE {
      mac            Verifier-MAC,
      ...
}

Verifier-MAC         ::= SEQUENCE {
      identifier     [0] PrincipalName OPTIONAL,
      kvno           [1] UInt32 OPTIONAL,
      enctype        [2] Int32 OPTIONAL,
      mac            [3] Checksum
}

END
```

elements:
   A sequence of authorization data elements issued by the KDC. These elements are the authorization data that the verifier fields authenticate.

Verifier:
   A CHOICE type that currently contains only one alternative: Verifier-MAC. Future extensions might add support for public-key signatures.

Verifier-MAC:

Contains a MAC computed over the ASN.1 DER encoding of the AuthorizationData value in the elements field of the AD-CAMMAC. The identifier, kvno, and enctype fields help the recipient locate the key required for verifying the MAC. For the kdc-verifier and the svc-verifier, the identifier, kvno and enctype fields are often obvious from context and MAY be omitted. For the kdc-verifier, the MAC is computed differently than for the svc-verifier and the other-verifiers, as described later. The key usage for computing the MAC (Checksum) is 64.

kdc-verifier:
A Verifier-MAC where the key is a long-term key of the local Ticket-Granting Service (TGS). The checksum type is the required checksum type for the enctype of the TGS key. In contrast to the other Verifier-MAC elements, the KDC computes the MAC in the kdc-verifier over the ASN.1 DER encoding of the EncTicketPart of the surrounding ticket, but where the AuthorizationData value in the EncTicketPart contains the AuthorizationData value contained in the CAMMAC instead of the AuthorizationData value that would otherwise be present in the ticket. This altered Verifier-MAC computation binds the kdc-verifier to the other contents of the ticket, assuring the KDC that a malicious service has not substituted a mismatched CAMMAC received from another ticket.

svc-verifier:
A Verifier-MAC where the key is the same long-term service key that the KDC uses to encrypt the surrounding ticket. The checksum type is the required checksum type for the enctype of the service key used to encrypt the ticket. This field MUST be present if the service principal of the ticket is not the local TGS, including when the ticket is a cross-realm TGT.

other-verifiers:
A sequence of additional verifiers. In each additional Verifier-MAC, the key is a long-term key of the principal name specified in the identifier field. The PrincipalName MUST be present and be a valid principal in the realm. KDCs MAY add one or more "trusted service" verifiers. Unless otherwise administratively configured, the KDC SHOULD determine the "trusted service" principal name by replacing the service identifier component of the sname of the surrounding ticket with "host". The checksum is computed using a long-term key of the identified principal, and the checksum type is the required checksum type for the enctype of that long-term key. The kvno and enctype SHOULD be specified to disambiguate which of the long-term keys of the trusted service is used.

5. Usage

Application servers and KDCs MAY ignore the AD-CAMMAC container and the authorization data elements it contains. For compatibility with older Kerberos implementations, a KDC issuing an AD-CAMMAC SHOULD enclose it in an AD-IF-RELEVANT container unless the KDC knows that the application server is likely to recognize it.

6. Assigned numbers

The ad-type number for AD-CAMMAC is 96.

The key usage number for the Verifier-MAC checksum is 64.

7. IANA Considerations

[RFC Editor: please remove this section prior to publication.]

There are no IANA considerations in this document. Any numbers assigned in this document are not in IANA-controlled number spaces.

8. Security Considerations

Although authorization data are generally conveyed within the encrypted part of a ticket and are thereby protected by the existing encryption scheme used for the surrounding ticket, some authorization data requires the additional protection provided by the CAMMAC.

Some protocol extensions such as S4U2Proxy allow the KDC to issue a new ticket based on an evidence ticket provided by the service. If the evidence ticket contains authorization data that needs to be preserved in the new ticket, then the KDC MUST revalidate it.

Extracting a CAMMAC from a ticket for use as a credential removes it from the context of the ticket. In the general case, this could turn it into a bearer token, with all of the associated security implications. Also, the CAMMAC does not itself necessarily contain sufficient information to identify the client principal. Therefore, application protocols that rely on extracted CAMMACs might need to duplicate a substantial portion of the ticket contents and include that duplicated information in the authorization data contained within the CAMMAC. The extent of this duplication would depend on the security properties required by the application protocol.

The method for computing the kdc-verifier does not bind it to any authorization data within the ticket but outside of the CAMMAC. At least one (non-standard) authorization data type, AD-SIGNEDPATH,

attempts to bind to other authorization data in a ticket, and it is very difficult for two such authorization data types to coexist.

To minimize ticket size when embedding CAMMACs in Kerberos tickets, a KDC MAY omit the kdc-verifier from the CAMMAC when it is not needed. In this situation, the KDC cannot always determine whether the CAMAMC contents are intact. The KDC MUST NOT create a new CAMMAC from an existing one unless the existing CAMMAC has a valid kdc-verifier, with two exceptions.

Only KDCs for the local realm have knowledge of the local TGS key, so the outer encryption of a local TGT is sufficient to protect the CAMMAC of a local TGT from tampering, assuming that all of the KDCs in the local realm consistently filter out CAMMAC authorization data submitted by clients. The KDC MAY create a new CAMMAC from an existing CAMMAC lacking a kdc-verifier if that CAMMAC is contained within a local TGT and all of the local realm KDCs are configured to filter out CAMMAC authorization data submitted by clients.

An application service might be use the S4U2Proxy extension, or the realm policy might disallow the use of S4U2Proxy by that service. In this situation, the application service could modify the CAMMAC contents, but such modifications would have no effect on other services. The KDC MAY create a new CAMMAC from an existing CAMMAC lacking a kdc-verifier if it is inserting the new CAMMAC into a service ticket for the same service principal as the ticket that contained the existing CAMMAC, and if all of the realm's KDCs are configured to reject S4U2Proxy requests made by that service principal.

9. Acknowledgements

Shawn Emery, Sam Hartman, Greg Hudson, Ben Kaduk, Zhanna Tsitkov, and Kai Zheng provided helpful technical and editorial feedback on earlier versions of this document.

10. References

10.1. Normative References

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

[RFC3961] Raeburn, K., "Encryption and Checksum Specifications for Kerberos 5", RFC 3961, February 2005.

[RFC4120] Neuman, C., Yu, T., Hartman, S., and K. Raeburn, "The Kerberos Network Authentication Service (V5)", RFC 4120, July 2005.

[X.680] ISO, , "Information technology -- Abstract Syntax Notation One (ASN.1): Specification of basic notation -- ITU-T Recommendation X.680 (ISO/IEC International Standard 8824-1:2008)", 2008.

[X.690] ISO, , "Information technology -- ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER) -- ITU-T Recommendation X.690 (ISO/IEC International Standard 8825-1:2008)", 1997.

10.2. Informative References

[MIT-Athena] Steiner, J., Neuman, B., and J. Schiller, "Kerberos: An Authentication Service for Open Network Systems. In Proceedings of the Winter 1988 Usenix Conference. February.", 1988.

[MS-SFU] Microsoft, "[MS-SFU]: Kerberos Protocol Extensions: Service for User and Constrained Delegation Protocol", January 2013, <http://msdn.microsoft.com/en-us/library/cc246071.aspx>.

[RFC1510] Kohl, J. and B. Neuman, "The Kerberos Network Authentication Service (V5)", RFC 1510, September 1993.

Authors' Addresses

Simo Sorce (editor)
Red Hat

Email: ssorce@redhat.com

Tom Yu (editor)
MIT Kerberos Consortium

Email: tlyu@mit.edu

Thomas Hardjono (editor)
MIT Kerberos Consortium

Email: hardjono@mit.edu