NETWORK WORKING GROUP L. Zhu
Internet-Draft Microsoft Corporation
Updates: 4120 (if approved) J. Altman
Intended status: Standards Track Secure Endpoints
Expires: January 31, 2010 July 30, 2009

Initial and Pass Through Authentication Using Kerberos V5 and the GSS-API (IAKERB)
draft-ietf-krb-wg-iakerb-02

Status of this Memo

This Internet-Draft will expire on January 31, 2010.

Copyright Notice

Abstract

This document defines extensions to the Kerberos protocol and the

GSS-API Kerberos mechanism that enable a GSS-API Kerberos client to exchange messages with the KDC using the GSS-API acceptor as the proxy, by encapsulating the Kerberos messages inside GSS-API tokens. With these extensions a client can obtain Kerberos tickets for services where the KDC is not accessible to the client, but is accessible to the application server.

Table of Contents

1. Introduction

When authenticating using Kerberos V5, clients obtain tickets from a KDC and present them to services. This model of operation cannot work if the client does not have access to the KDC. For example, in remote access scenarios, the client must initially authenticate to an access point in order to gain full access to the network. Here the client may be unable to directly contact the KDC either because it does not have an IP address, or the access point packet filter does not allow the client to send packets to the Internet before it authenticates to the access point.

Recent advancements in extending Kerberos permit Kerberos authentication to complete with the assistance of a proxy. The Kerberos [RFC4120] pre-authentication framework [KRB-PAFW] prevents the exposure of weak client keys over the open network. The Kerberos support of anonymity [KRB-ANON] provides for privacy and further complicates traffic analysis. The kdc-referrals option defined in [KRB-PAFW] may reduce the number of messages exchanged while obtaining a ticket to exactly two even in cross-realm authentications.

Building upon these Kerberos extensions, this document extends [RFC4120] and [RFC4121] such that the client can communicate with the KDC using a Generic Security Service Application Program Interface (GSS-API) [RFC2743] acceptor as the proxy. The GSS-API acceptor relays the KDC request and reply messages between the client and the KDC. The GSS-API acceptor, when relaying the Kerberos messages, is called an IAKERB proxy. Consequently, IAKERB as defined in this document requires the use of GSS-API.

2. 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 [RFC2119].

3. GSS-API Encapsulation

The mechanism Objection Identifier (OID) for GSS-API IAKERB, in accordance with the mechanism proposed by [RFC4178] for negotiating protocol variations, is id-kerberos-iakerb:

```
   id-kerberos-iakerb ::=
     { iso(1) org(3) dod(6) internet(1) security(5) kerberosV5(2)
       iakerb(5) }
```

All context establishment token of IAKERB MUST have the generic token framing described in section 3.1 of [RFC2743] with the mechanism OID being id-kerberos-iakerb.

The client starts by constructing the ticket request, and if the ticket request is being made to the KDC, the client, instead of contacting the KDC directly, encapsulates the request message into the output token of the GSS_Init_security_context() call and returns GSS_S_CONTINUE_NEEDED [RFC2743] indicating that at least one more token is required in order to establish the context. The output token is then passed for use as the input token to the GSS_Accept_sec_context() call in accordance with GSS-API. The GSS-API acceptor extracts the Kerberos request in the input token, locates the target KDC, and sends the request on behalf of the client. After receiving the KDC reply, the GSS-API acceptor then encapsulates the reply message into the output token of GSS_Accept_sec_context(). The GSS-API acceptor returns GSS_S_CONTINUE_NEEDED [RFC2743] indicating that at least one more token is required in order to establish the context. The output token is passed to the initiator in accordance with GSS-API.

```
       Client <---------> IAKERB proxy <---------> KDC
```

The innerToken described in section 3.1 of [RFC2743] and subsequent GSS-API mechanism tokens have the following formats: it starts with a two-octet token-identifier (TOK_ID), followed by an IAKERB message or a Kerberos message.

Only one IAKERB specific message, namely the IAKERB_PROXY message, is defined in this document. The TOK_ID values for Kerberos messages are the same as defined in [RFC4121].

Token	TOK_ID Value in Hex
IAKERB_PROXY	05 01

The content of the IAKERB_PROXY message is defined as an IAKERB-HEADER structure immediately followed by a Kerberos message. The Kerberos message can be an AS-REQ, an AS-REP, a TGS-REQ, a TGS-REP, or a KRB-ERROR as defined in [RFC4120].

```
      IAKERB-HEADER ::= SEQUENCE {
          target-realm      [1] UTF8String,
             -- The name of the target realm.
          cookie            [2] OCTET STRING OPTIONAL,
             -- Opaque data, if sent by the server,
             -- MUST be copied by the client verbatim into
             -- the next IAKRB_PROXY message.
          ...
      }
```

The IAKERB-HEADER structure and all the Kerberos messages MUST be encoded using Abstract Syntax Notation One (ASN.1) Distinguished Encoding Rules (DER) [X680] [X690].

The IAKERB client fills out the IAKERB-HEADER structure as follows: the target-realm contains the realm name the ticket request is addressed to. In the initial message from the client, the cookie field is absent. The client MUST specify a target-realm. If the client does not know the realm of the client's true principal name [REFERALS], it MUST specify a realm it knows. This can be the realm of the client's host.

Upon receipt of the IAKERB_PROXY message, the GSS-API acceptor inspects the target-realm field in the IAKERB_HEADER, and locates a KDC of that realm, and sends the ticket request to that KDC.

The GSS-API server encapsulates the KDC reply message in the returned IAKERB message. It fills out the target realm using the realm sent by the client and the KDC reply message is included immediately following the IAKERB-HEADER header.

When the GSS-API acceptor is unable to obtain an IP address for a KDC in the client's realm, it sends a KRB_ERROR message with the code KRB_AP_ERR_IAKERB_KDC_NOT_FOUND to the client and the context fails to establish. There is no accompanying error data defined in this document for this error code.

```
      KRB_AP_ERR_IAKERB_KDC_NOT_FOUND      85
          -- The IAKERB proxy could not find a KDC.
```

When the GSS-API acceptor has an IP address for a KDC in the client realm, but does not receive a response from any KDC in the realm (including in response to retries), it sends a KRB_ERROR message with the code KRB_AP_ERR_IAKERB_KDC_NO_RESPONSE to the client and the context fails to establish. There is no accompanying error data defined in this document for this error code.

```
      KRB_AP_ERR_IAKERB_KDC_NO_RESPONSE    86
```

-- The KDC did not respond to the IAKERB proxy.

The IAKERB proxy can send opaque data in the cookie field of the IAKERB-HEADER structure in the server reply to the client, in order to, for example, minimize the amount of state information kept by the GSS-API acceptor. The content and the encoding of the cookie field is a local matter of the IAKERB proxy. The client MUST copy the cookie verbatim from the previous server response whenever the cookie is present into the subsequent tokens that contains an IAKERB_PROXY message.

The client and the server can repeat the sequence of sending and receiving the IAKERB messages as described above, in order to allow the client interact with the KDC through the IAKERB proxy, and to obtain Kerberos tickets as needed.

When obtaining the initial TGT, the client may start with an NT-ENTERPRISE name type and the client host does not have a Kerberos realm. To resolve the NT-ENTERPRISE name type, the client typically starts with the client host realm and then finds out the true realm of the client based on [REFERALS]. In this case the GSS-API client can retrieve the realm of the GSS-API server as follows: the client returns GSS_S_CONTINUE_NEEDED with the output token containing an IAKERB message with an empty target-realm in the IAKERB-HEADER and no Kerberos message following the IAKERB-HEADER structure. Upon receipt of the realm request, the GSS-API server fills out the target realm field using the realm of the server, and returns GSS_S_CONTINUE_NEEDED with the output token containing the IAKERB message with the server's realm and no Kerberos message following the IAKERB-HEADER header. The GSS-API client can then use the returned realm in subsequent IAKERB messages to resolve the NT-ENTERPRISE name type. Since the GSS-API server can act as a Kerberos acceptor, it always has a Kerberos realm in this case.

When the client obtained a service ticket, the client sends a KRB_AP_REQ message to the server, and performs the client-server application exchange as defined in [RFC4120] and [RFC4121].

For implementations conforming to this specification, both the authenticator subkey and the GSS_EXTS_FINISHED extension as defined in [PKU2U] MUST be present in the AP-REQ authenticator. This checksum provides integrity protection for the messages exchanged including the unauthenticated clear texts in the IAKERB-HEADER structure.

If the pre-authentication data is encrypted in the long-term password-based key of the principal, the risk of security exposures is significant. Implementations SHOULD provide the AS_REQ armoring

as defined in [KRB-PAFW] unless an alternative protection is deployed. In addition, the anonymous Kerberos FAST option is RECOMMENDED for the client to complicate traffic analysis.

4. Addresses in Tickets

In IAKERB, the machine sending requests to the KDC is the GSS-API acceptor and not the client. As a result, the client should not include its addresses in any KDC requests for two reasons. First, the KDC may reject the forwarded request as being from the wrong client. Second, in the case of initial authentication for a dial-up client, the client machine may not yet possess a network address. Hence, as allowed by [RFC4120], the addresses field of the AS-REQ and TGS-REQ requests SHOULD be blank and the caddr field of the ticket SHOULD similarly be left blank.

5. Security Considerations

A typical IAKERB client sends the AS_REQ with pre-authentication data encrypted in the long-term keys of the user before the server is authenticated. This enables offline attacks by un-trusted servers. To mitigate this threat, the client SHOULD use Kerberos FAST[KRB-PAFW] and require KDC authentication to protect the user's credentials.

The client name is in clear text in the authentication exchange messages and ticket granting service exchanges according to [RFC4120] whereas the client name is encrypted in client- server application exchange messages. By using the IAKERB proxy to relay the ticket requests and responses, the client's identity could be revealed in the client-server traffic where the same identity could have been concealed if IAKERB were not used. Hence, to complicate traffic analysis and provide privacy for the IAKERB client, the IAKERB client SHOULD request the anonymous Kerberos FAST option [KRB-PAFW].

Similar to other network access protocols, IAKERB allows an unauthenticated client (possibly outside the security perimeter of an organization) to send messages that are proxied to interior servers. To reduce attack surface, firewall filters can be applied to allow from which hosts the client requests can be proxied and the proxy can further restrict the set of realms to which the requests can be proxied.

In a scenario where DNS SRV RR's are being used to locate the KDC, IAKERB is being used, and an external attacker can modify DNS responses to the IAKERB proxy, there are several countermeasures to

prevent arbitrary messages from being sent to internal servers:

1. KDC port numbers can be statically configured on the IAKERB proxy. In this case, the messages will always be sent to KDC's. For an organization that runs KDC's on a static port (usually port 88) and does not run any other servers on the same port, this countermeasure would be easy to administer and should be effective.

2. The proxy can do application level sanity checking and filtering. This countermeasure should eliminate many of the above attacks.

3. DNS security can be deployed. This countermeasure is probably overkill for this particular problem, but if an organization has already deployed DNS security for other reasons, then it might make sense to leverage it here. Note that Kerberos could be used to protect the DNS exchanges. The initial DNS SRV KDC lookup by the proxy will be unprotected, but an attack here is at most a denial of service (the initial lookup will be for the proxy's KDC to facilitate Kerberos protection of subsequent DNS exchanges between itself and the DNS server).

6. Acknowledgements

Jonathan Trostle, Michael Swift, Bernard Aboba and Glen Zorn wrote earlier revision of this document.

The hallway conversations between Larry Zhu and Nicolas Williams formed the basis of this document.

7. IANA Considerations

There is no IANA action required for this document.

8. References

8.1. Normative References

[GSS-EXTS]
Emery, S., "Kerberos Version 5 GSS-API Channel Binding Hash Agility", draft-ietf-krb-wg-gss-cb-hash-agility-03.txt (work in progress), 2007.

[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.

[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.

[RFC4121] Zhu, L., Jaganathan, K., and S. Hartman, "The Kerberos Version 5 Generic Security Service Application Program Interface (GSS-API) Mechanism: Version 2", RFC 4121, July 2005.

[RFC4178] Zhu, L., Leach, P., Jaganathan, K., and W. Ingersoll, "The Simple and Protected Generic Security Service Application Program Interface (GSS-API) Negotiation Mechanism", RFC 4178, October 2005.

8.2. Informative references

[KRB-ANON] Zhu, L. and P. Leach, "Kerberos Anonymity Support", draft-ietf-krb-wg-anon-04.txt (work in progress), 2007.

[KRB-PAFW] Zhu, L. and S. Hartman, "A Generalized Framework for Kerberos Pre-Authentication", draft-ietf-krb-wg-preauth-framework-06.txt (work in progress), 2007.

[PKU2U] Zhu, L. and J. Altman, "Public Key Cryptography Based User-to-User Authentication - (PKU2U)", draft-zhu-pku2u (work in progress), 2007.

[REFERALS] Raeburn, K. and L. Zhu, "Kerberos Principal Name Canonicalization and KDC-Generated Cross-Realm Referrals", draft-ietf-krb-wg-kerberos-referral (work in progress), 2009.

Authors' Addresses

Larry Zhu
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
US

Email: larry.zhu@microsoft.com

Jeffery Altman
Secure Endpoints
255 W 94th St
New York, NY 10025
US

Email: jaltman@secure-endpoints.com