1. Abstract

The Dynamic Host Configuration Protocol (DHCP) [1] includes an option that allows authentication of all DHCP messages, as specified in [2]. This document specifies a DHCP authentication mode based on Kerberos V tickets. This provides mutual authentication between a DHCP client and server, as well as authentication of all DHCP messages.

This document specifies Kerberos message exchanges between an uninitialized client and the KDC (Key Distribution Center) using an IAKERB proxy [7] so that the Kerberos key management phase is decoupled from, and precedes the address allocation and network configuration phase that uses the DHCP authentication option. In order to make use of the IAKERB proxy, this document specifies a transport mechanisms that work with an uninitialized client (i.e. a client without an assigned IP address).
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 RFC-2119.

3. Introduction

3.1 Terminology

- "DHCP client"
  A DHCP client is an Internet host using DHCP to obtain configuration parameters such as a network address.

- "DHCP server"
  A DHCP server is an Internet host that returns configuration parameters to DHCP clients.

- "Ticket"
  A Kerberos term for a record that helps a client authenticate itself to a server; it contains the client’s identity, a session key, a timestamp, and other information, all sealed using the server’s secret key. It only serves to authenticate a client when presented along with a fresh Authenticator.

- "Key Distribution Center"
  Key Distribution Center, a network service that supplies tickets and temporary session keys; or an instance of that service or the host on which it runs. The KDC services both initial ticket and Ticket-Granting Ticket (TGT) requests. The initial ticket portion is sometimes referred to as the Authentication Server (or service). The Ticket-Granting Ticket portion is sometimes referred to as the Ticket-Granting Server (or service).

- "Realm"
  A Kerberos administrative domain that represents a group of principals registered at a KDC. A single KDC may be responsible for one or more realms. A fully qualified principal name includes a realm name along with a principal name unique within that realm.

3.2 Protocol Overview

DHCP as defined in [1] defines the protocol exchanges for a client to obtain its IP address and network configuration information from a DHCP Server. Kerberos V5 as described in [6] defines the protocol and message exchanges to mutually authenticate two parties. It is
our goal to provide authentication support for DHCP using Kerberos. This implies that the Kerberos key management exchange has to take place before a client gets its IP address from the DHCP Server. Kerberos assumes that the client has a network address and can contact the Key Distribution Center to obtain its credentials for authenticated communication with an application server.

In this specification we utilize the key exchange using an IAKERB proxy described in [7]. This does not require any changes to either the IAKERB or the Kerberos V5 specification. This document also specifies a particular transport that allows an uninitialized client to contact an IAKERB proxy.

The Kerberos ticket returned from the key management exchange discussed in section 0 of this document is passed to the DHCP Server inside the DHCP authentication option with the new Kerberos authenticator type. This is described in Section 6 of this draft.

3.3 Related Work

A prior Internet Draft [3] outlined the use of Kerberos-based authentication for DHCP. The proposal tightly coupled the Kerberos client state machines and the DHCP client state machines. As a result, the Kerberos key management messages were carried in DHCP messages, along with the Kerberos authenticators.

We propose a protocol exchange where Kerberos key management is decoupled from and precedes authenticated DHCP exchanges. This implies that the Kerberos ticket returned in the initial key management exchange could be used to authenticate servers assigning addresses by non-DHCP address assignment mechanisms like RSIP [4] and for service specific parameter provisioning mechanisms using SLP [5].

S. Medvinsky, P. Lalwaney

Kerberos V Authentication Mode for DHCP

March 2000

4. System Architecture

Client
In this architecture, the DHCP client obtains a Kerberos ticket from the Key Distribution Center (KDC) using standard Kerberos messages, as specified in [6]. The client, however, contacts the KDC via a proxy server, according to the IAKERB mechanism, described in [7].

The are several reasons why a client has to go through this proxy in order to contact the KDC:

a) The client may not know the host address of the KDC and may be sending its first request message as a broadcast on a local network. The KDC may not be located on the local network, and even if it were - it will be unable to communicate with a client without an IP address. This document describes a specific mechanism that may be used by a client to communicate with the Kerberos proxy.

b) The client may not know its Kerberos realm name. The proxy is able to fill in the missing client realm name in an AS Request message, as specified in IAKERB. Note that in the case that PKINIT pre-authenticator is used [8], the realm name in the AS Request may be the KDC realm name and not the client's realm name.

c) The client may not know the realm name of the DHCP server. According to IAKERB, when the client sends a TGS Request with a missing server realm name, the proxy will return to the client an error message containing the missing realm name.

Note that in this case the proxy could return the client a wrong
realm name and the client could be fooled into obtaining a ticket for the wrong DHCP server (on the same local network). However, the wrong DHCP server must still be a registered principal in a KDC database. In some circumstances this may be an acceptable compromise. Also, see the security considerations section.

IAKERB describes the proxy as part of an application server - the DHCP server in this case. However, in this document we are not requiring the proxy to be integrated with the DHCP server. The same IAKERB mechanisms apply in the more general case, where the proxy is an independent application. This proxy, however, MUST be reachable by a client via a local network broadcast.

After a client has obtained a Kerberos ticket for the DHCP server, it will use it as part of an authentication option in the DHCP messages. The only extension to the DHCP protocol is the addition of a new authenticator type based on Kerberos tickets.

4.1 Cross-Realm Authentication

The above diagram shows a client communicating with a single KDC via a proxy. However, the DHCP client’s realm may be different from the DHCP server’s realm. In that case, the client may need to first contact the KDC in its local realm to obtain a cross-realm TGT. Then, the client would use the cross-realm TGT to contact the KDC in the DHCP server’s realm, as specified in [6].

In the following example a client doesn’t know its realm or the DHCP server’s realm, which happens to be different from the client’s realm. Here are the steps in obtaining the ticket for the DHCP server (based on [6] and [7]):

1) The client sends AS Request with NULL realm to the proxy.
2) The proxy fills in the realm and forwards the AS Request to the KDC in the client’s realm.
3) The KDC issues a TGT and sends back an AS Reply to the proxy.
4) The proxy forwards AS Reply to the client.
5) The client sends TGS Request for a principal name "dhcpsrvr" with NULL realm to the proxy.

6) The proxy returns KRB_AP_ERR_REALM_REQUIRED error with the DHCP server’s realm to the client.
7) The client sends another TGS Request for a cross-realm TGT to the proxy.
8) The proxy forwards the TGS Request to the KDC in the client’s realm.
9) The KDC issues a cross-realm TGT and sends back a TGS Reply to the proxy.
10) The proxy forwards TGS Reply to the client.
11) The client sends a TGS Request to the proxy for a principal ‘dhcpsrvr’ with the realm name filled in, using a cross-realm TGT.
12) The proxy forwards TGS Request to the KDC in the DHCP server’s realm.
13) The KDC issues a ticket for the DHCP server and sends TGS Reply back to the proxy.
14) The proxy forwards TGS Reply to the client.

In a most general case, the client may need to contact any number of KDCs in different realms before it can get a ticket for the DHCP server. In each case, the client would contact a KDC via the proxy server, as specified in Section 5 of this document.

4.2 Public Key Authentication

This specification also allows clients to perform public key authentication to the KDC, based on the PKINIT specification [8]. In this case, the size of an AS Request and AS Reply messages is likely to exceed the size of typical link MTU’s.

Here is an example, where PKINIT is used by a DHCP client that is not a registered principal in the KDC principal database:

1) The client sends AS Request with a PKINIT Request pre-authenticator to the proxy. This includes the client’s signature and X.509 certificate. The KDC realm field is left as NULL.
2) The proxy fills in the realm and forwards the AS Request to the KDC in the filled in realm. This is the realm of the DHCP server. Here, the client’s realm is the name of a Certification Authority - not the same as the KDC realm.
3) The KDC issues a TGT and sends back an AS Reply with a PKINIT Reply pre-authenticator to the proxy.
4) The proxy forwards the AS Reply to the client.
5) The client sends TGS Request for a principal name "dhcpsrvr" with the realm found in the TGT to the proxy.
6) The proxy forwards TGS Request to the KDC in the DHCP server’s realm.
7) The KDC issues a ticket for the DHCP server and sends TGS Reply back to the proxy.
8) The proxy forwards TGS Reply to the client.

5. Key Management Exchange That Precedes Network Address Allocation

An uninitialized host (e.g. on power-on and reset) does not have a network address. It does have a link layer address or hardware address. At this time, the client may not have any information on its realm or the realm of the address allocation server (DHCP Server).

In the Kerberos key management exchange, a client gets its ticket granting ticket (TGT) by contacting the Authentication Server in the KDC using the AS_Request / Reply messages (shown as messages 1 and 2 in Figure 1). The client then contacts the Ticket Granting Server in the KDC to get the DHCP server ticket (to be used for mutual authentication with the DHCP server) using the TGS_REQ / TGS_REP messages (shown as messages 3 and 4 in the above figure). It is also possible for the client to obtain a DHCP server ticket directly with the AS Request / Reply exchange, without the use of the TGT.
In the use of Kerberos for DHCP authentication, the client (a) does not have an IP/network address (b) does not know he KDC’s IP address (c) the KDC may not be on the local network and (d) the client may not know the DHCP Server’s IP address and realm. We therefore require a Kerberos proxy on the local network to accept broadcast Kerberos request messages (AS_REQ and TGS_REQ) from uninitialized clients and relay them to the appropriate KDC.

The uninitialized client formulates a broadcast AS_REQ or TGS_REQ as follows:

The request payload contains the client hardware address in addresses field with a negative value for the address type. Kerberos v5 [6] allows for the usage of negative address types for "local" use. Note that IAKERB [7] discourages the use of the addresses field as network addresses may not be known or may change in situation where proxies are used. In this draft we incorporate the negative values permitted in the Kerberos transport in the address type field of both the AS_REQ and TGS_REQ messages. The negative value MUST be the negative number of the hardware address type "htype" value (from assigned numbers RFC) used in RFC 2131. The address field of the message contains the clients hardware address.

The request payload is UDP encapsulated and addressed to port 88 on the server/proxy. The UDP source port is selected by the client. The source and destination network addresses are the all-zero’s address and the broadcast address, respectively. For IPv4, the source IP address is set to 0.0.0.0 and the destination IP address is set to 255.255.255.255. The data link layer header source address corresponds to the link layer/hardware address of the client. The destination link layer address is the broadcast address at the link layer (e.g. for Ethernet the address is ffffffff).

In the case where AS_REQ message contains a PKINIT pre-authenticator for public key-based client authentication (based on [8]), the message will probably not fit into a single UDP packet given typical link MTU’s.

It is assumed that the proxy server on a network is configured with a list of KDC’s, their realms and their IP addresses. The proxy server will act as a client to the KDC and forward standard Kerberos messages to/from the KDC using unicast UDP or TCP transport mechanisms, according to [6].

Upon receiving a broadcast request from a client, the proxy MUST record the client’s hardware address that appears as the source address on the frame as well as in the addresses field of the request message. Based on the realm of the KDC specified in the request, the proxy determines the KDC to which this message is relayed as a unicast message from the proxy to the KDC. In the case that the client left the KDC realm name as NULL, it is up to the proxy to first determine the correct realm name and fill it in the request (according to [7]).
On receiving a request, the KDC formulates a response (AS_REP or TGS_REP). It includes the client’s addresses field in the encrypted part of the ticket (according to [6]). This response is unicast to the proxy.

Upon receiving the reply, the proxy MUST first determine the previously saved hardware address of the client. The proxy broadcasts the reply on its local network. This is a network layer broadcast. At the link level, it uses the hardware address obtained from the addresses field of the request.

The client on receiving the response (link layer destination address as its hardware address, network layer address is the broadcast address) must verify that the hardware address in the ticket corresponds to its link layer address.

Upon receiving a TGS_REP (or an AS_REP with the application server ticket) from the proxy, the client will have enough information to securely communicate with the application server (the DHCP Server in this case), as specified in the following section.

6. Authenticated Message Exchange Between the DHCP Client and the DHCP Server

The ticket returned in the TGS response is used by the DHCP client for two purposes: to establish a shared session key with the DHCP server, and is also included as part of a Kerberos authenticator in the DHCP request.

If the size of the authenticator is greater than 255 bytes, the DHCP authentication option is repeated multiple times. When the values of all the authentication options are concatenated together, they will make up the complete authenticator. This method of handling long authenticators is currently not specified in the DHCP authentication draft [2] but is expected be added in the next version of the draft.

Once the session key is established, the following messages sent by both the DHCP client and the DHCP server will use an authenticator defined as protocol 1 in [2].

The Kerberos authenticator for a DHCP request message is specified below:

```
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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Code      |    Length     |    Protocol   |0 0| Algorithm |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+              Replay Detection (64 bits)                       +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
The format of this authenticator is in accordance with [2]. The code for the authentication option is TBD, and the length field contains the length of the remainder of the option, starting with the protocol field.

The value of the protocol field for this authenticator is 2.

The algorithm field can take the following values:

1 - HMAC-MD5
2 - HMAC-SHA-1

Replay protection field is a monotonically increasing counter field. Inside the Kerberos authenticator, it may be set to any value. This field must be monotonically increasing in the following messages sent by the client (inside an authenticator using protocol 1).

Authenticator token consists of type-length-value attributes:

```
| +---------------------------------+        |
| +---------------------------------+        |
| +---------------------------------+  +---+ |
| +---------------------------------+  |     |
| +---------------------------------+  |     |
| +---------------------------------+  +---+ |
| +---------------------------------+  +---+ |
| +---------------------------------+  +---+ |
| +---------------------------------+  +---+ |
| +---------------------------------+  +---+ |
| +---------------------------------+  +---+ |
```

<table>
<thead>
<tr>
<th>Type</th>
<th>Reserved</th>
<th>Payload Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>attribute</td>
<td>value...</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The following attributes must be included in the authenticator token:

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Message Integrity Code</td>
<td>Depends on the value of the algorithm field. Its length is 16 bytes for HMAC-MD5 [9, 10] and 20 bytes for HMAC-SHA-1 [11, 10]. The Kerberos session key found in the Kerberos ticket is used directly as an HMAC key (see security considerations).</td>
</tr>
</tbody>
</table>
1            AP_REQ                  ASN.1 encoding of a Kerberos
            AP_REQ message, as specified in
            [6].

AP_REQ contains the Kerberos ticket for the DHCP server and also
contains information needed by the DHCP server to authenticate the
client. After verifying the AP_REQ and decrypting the Kerberos
ticket, the DHCP server is able to extract a session key which it
now shares with the DHCP client.

The Kerberos authenticator token contains its own replay protection
mechanism inside the AP_REQ structure. The AP_REQ contains a
timestamp that must be within an agreed upon time window at the DHCP
server. However, this does not require the DHCP clients to maintain
an accurate clock between reboots. Kerberos allows clients to
synchronize their clock with the KDC with the help of Kerberos
KRB_AP_ERR_SKEW error message, as specified in [6].

The DHCP server must save both the session key and its associated
expiration time found in the Kerberos ticket. Up until the
expiration time, the server must accept client requests with the
authenticator using protocol 1, with the Kerberos session key as the
MD5 HMAC key.

S. Medvinsky, P. Lalwaney                                         -10-
Kerberos V Authentication Mode for DHCP        March 2000

All DHCP server responses must also contain an authenticator using
protocol 1, with the same Kerberos session key as the MD5 HMAC key.

When the session key expires, it is up to the client to obtain a new
ticket from the KDC and to attach a Kerberos authenticator to the
next DHCP request message (instead of the authenticator using
protocol 1).

7. Detailed message flows for Kerberos and DHCP message Exchanges

The following flow depicts the kerberos exchange in which a AS REQ
message is used to directly request the DHCP Server ticket. There
are no changes to transport mechanisms below when the additional
phase of using TGS requests/responses with TGT’s is used.

Client                         IAKERB Proxy             KDC

KB-client

-------- AS_REQ -------->

AS_REQ Address type = - (htype)
AS_REQ Address= hw address
src UDP port = senders port
destination UDP port = 88
src IP = 0.0.0.0
destination IP = 255.255.255.255
src link layer address =
client’s HW/link address [e.g. Ethernet address]

destination link layer address =
link broadcast address [e.g. fffffff for Ethernet]

----------------------------------------------
(unicast to UDP port 88)

----------------------------------------------
(unicast AS REP)
Encrypted portion of ticket
Includes clients HW address

<--------------------------AS_REP -------------------------->

Ticket includes client’s hardware address

S. Medvinsky, P. Lalwaney
Kerberos V Authentication Mode for DHCP  March 2000

src UDP port = 88
destination UDP port = copied from src port in AS_REQ

src IP = Proxy’s IP address
destination IP = 255.255.255.255

src link layer address = Proxy’s HW/link address
destination link layer address =
Client’s link layer address from AS_REQ

The client uses the ticket received from the KDC in the DHCP
Authentication option as described in Section 6.

Client
DHCP-client DHCP Server

--------DHCPDISCOVER ---->
(Auth Protocol = 2 [Kerberos] )

| AP REQ | HMAC |
| Ticket| KB Authenticator |

1. Server decrypts ticket with
service key
2. Server decrypts
authenticator and checks
content and checksum to validate the client.
3. Recompute HMAC with session key and compare.

<---DHCPOFFER---------
  (Auth Protocol = 1 )

-------DHCPREQUEST-->
  (Auth Protocol = 1 )

<---DHCPACK---------
  (Auth Protocol = 1 )

8. Security Considerations

DHCP clients that do not know the DHCP server’s realm name will get it from the proxy, as specified in IAKERB [7]. Since the proxy is not authenticated, a DHCP client can be fooled into obtaining a ticket for the wrong DHCP server in the wrong realm.

This could happen when the client leaves out the server realm name in a TGS Request message to the proxy. It is also possible, however, for a client to directly request a DHCP server ticket with an AS Request message. In those cases, the same situation occurs when the client leaves out the realm name in an AS Request.

This wrong DHCP server is still registered as a valid principal in a database of a KDC that can be trusted by the client. In some circumstances a client may assume that a DHCP server that is a Kerberos principal registered with a trusted KDC will not attempt to deliberately misconfigure a client.

This specification provides a tradeoff between:

1) The DHCP clients knowing DHCP server’s realm ahead of time, which provides for full 2-way authentication at the cost of an additional configuration parameter.
2) The DHCP clients not requiring any additional configuration information, besides a password or a key (and a public key certificate if PKINIT is used). This is at the cost of not being able to fully authenticate the identity of the DHCP server.

In this specification the Kerberos session key is used directly as an HMAC key to authenticate DHCP message. Standard security practice, however, is to use different keys for different purposes.
The Kerberos session key is already used to encrypt a part of an AP_REQ message.

There is a near term plan to update the Kerberos V specification with the procedure for deriving a separate authentication key from the Kerberos session key. After the Kerberos V specification is updated, this draft should also be updated, to use the same key derivation procedure for the DHCP message authentication key.

9. References


10. Author’s Addresses

Sasha Medvinsky  
Motorola  
San Diego, CA 92121  
Email: smedvinsky@gi.com

Poornima Lalwaney

Motorola  
San Diego, CA 92121  
Email: plalwaney@gi.com

11. Expiration

This memo is filed as <draft-smedvinsky-dhc-kerbauth-00.txt>, and expires September 30, 2000.

12. Intellectual Property Notices

This section contains two notices as required by [12] for standards track documents. Per [12], section 10.4(A):

The IETF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on the IETF’s procedures with respect to rights in standards-track and standards-related documentation can be found in BCP-11. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification can be obtained from the IETF Secretariat.

Per [12] section 10.4(D):

The IETF has been notified of intellectual property rights claimed in regard to some or all of the specification contained in this document. For more information consult the online list of claimed rights.

13. Full Copyright Statement