[MS-KKDCP]:
Kerberos Key Distribution Center (KDC) Proxy Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.

- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.

- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.

- **Patents.** Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft Open Specification Promise or the Community Promise. If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.

- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.

- **Fictitious Names.** The example companies, organizations, products, domain names, email addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

Reservation of Rights. All other rights are reserved, and this notice does not grant any rights other than specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications do not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments you are free to take advantage of them. Certain Open Specifications are intended for use in conjunction with publicly available standard specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.

Revision Summary

Date	Revision History	Revision Class	Comments
12/16/2011	1.0	New	Released new document.
03/30/2012	1.0	No change	No changes to the meaning, language, or formatting of the technical content.
07/12/2012	1.1	Minor	Clarified the meaning of the technical content.
10/25/2012	1.1	No change	No changes to the meaning, language, or formatting of the technical content.
01/31/2013	1.2	Minor	Clarified the meaning of the technical content.
08/08/2013	2.0	Major	Significantly changed the technical content.

Release: Monday, July 22, 2013

Contents

Release: Monday, July 22, 2013

1   Introduction

The **Kerberos Key Distribution Center (KDC)** Proxy Protocol (KKDCP) is used by an HTTP-based KKDCP server and KKDCP client to relay the Kerberos Network **Authentication Service** (V5) protocol [RFC4120] and Kerberos change password [RFC3244] messages between a Kerberos client and a KDC.

Sections 1.8, 2, and 3 of this specification are normative and can contain the terms MAY, SHOULD, MUST, MUST NOT, and SHOULD NOT as defined in RFC 2119. Sections 1.5 and 1.9 are also normative but cannot contain those terms. All other sections and examples in this specification are informative.

Note Throughout the remainder of this specification the Kerberos Network Authentication Service (V5) protocol will be referred to simply as Kerberos V5. Kerberos Network Authentication Service (V5) protocol [RFC4120] and Kerberos change password [RFC3244] messages will be referred to simply as Kerberos messages.

1.1   Glossary

The following terms are defined in [MS-GLOS]:

> **Authentication Service (AS)**
> **DC**
> **domain controller (DC)**
> **Hypertext Transfer Protocol (HTTP)**
> **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)**
> **Kerberos**
> **Key Distribution Center (KDC)**
> **realm**
> **ticket-granting ticket (TGT)**
> **TLS**
> **Transport Layer Security (TLS)**
> **URI**

The following terms are specific to this document:

> **MAY, SHOULD, MUST, SHOULD NOT, MUST NOT:** These terms (in all caps) are used as described in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

1.2   References

References to Microsoft Open Specifications documentation do not include a publishing year because links are to the latest version of the documents, which are updated frequently. References to other documents include a publishing year when one is available.

A reference marked "(Archived)" means that the reference document was either retired and is no longer being maintained or was replaced with a new document that provides current implementation details. We archive our documents online [Windows Protocol].

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information. Please check the archive site, http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624, as an additional source.

[MS-NRPC] Microsoft Corporation, "Netlogon Remote Protocol".

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, http://www.rfc-editor.org/rfc/rfc2119.txt

[RFC2616] Fielding, R., Gettys, J., Mogul, J., et al., "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999, http://www.ietf.org/rfc/rfc2616.txt

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000, http://www.ietf.org/rfc/rfc2818.txt

[RFC3244] Swift, M., Trostle, J., and Brezak, J., "Microsoft Windows 2000 Kerberos Change Password and Set Password Protocols", RFC 3244, February 2002, http://www.ietf.org/rfc/rfc3244.txt

[RFC4120] Neuman, C., Yu, T., Hartman, S., and Raeburn, K., "The Kerberos Network Authentication Service (V5)", RFC 4120, July 2005, http://www.ietf.org/rfc/rfc4120.txt

[RFC6113] Hartman, S., and Zhu, L., "A Generalized Framework for Kerberos Pre-Authentication", RFC 6113, April 2011, http://www.ietf.org/rfc/rfc6113.txt

[X680] ITU-T, "Abstract Syntax Notation One (ASN.1): Specification of Basic Notation", Recommendation X.680, July 2002, http://www.itu.int/rec/T-REC-X.680/en

Note There is a charge to download the specification.

[X690] ITU-T, "Information Technology - ASN.1 Encoding Rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)", Recommendation X.690, July 2002, http://www.itu.int/rec/T-REC-X.690/en

Note There is a charge to download the specification.

1.2.2 Informative References

[MS-GLOS] Microsoft Corporation, "Windows Protocols Master Glossary".

1.3 Overview

Kerberos V5 [RFC4120] requires client connectivity to the Key Distribution Center (KDC) for authentication. Kerberos Key Distribution Center (KDC) Proxy Protocol (KKDCP) provides a mechanism for a client to use a KKDCP server to change passwords and securely obtain Kerberos service tickets. The KKDCP client sends Kerberos messages using HTTPS to the KKDCP server. The KKDCP server locates a KDC for the request and sends the request to the KDC on behalf of the Kerberos V5 client. Since the messages received by the KDC are Kerberos messages, the KDC does not have a role in KKDCP. Once the KKDCP server receives the response from the KDC it sends the Kerberos message using HTTPS to the KKDCP client.
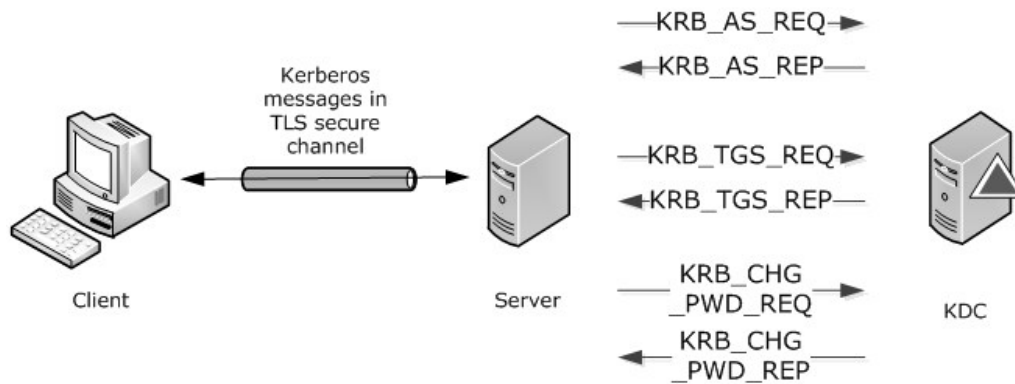
Figure 1: Messages between client, server, and KDC

1.4 Relationship to Other Protocols

KKDCP relies on either HTTP [RFC2616] or HTTPS [RFC2818] for network transport.

The KDC proxy server relies on **domain controller (DC)** location ([MS-NRPC] section 3.4.5.1.1) to find KDCs.

1.5 Prerequisites/Preconditions

KKDCP assumes the following:

- The KKDCP client is configured with the URL of the KKDCP server.

- The KKDCP client and server is configured for **Transport Layer Security (TLS)**.

1.6 Applicability Statement

KKDCP provides suitable Kerberos message proxying capability for Kerberos V5 clients where the client does not have connectivity to the KDC and a KKDCP server does.

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

2   Messages

2.1   Transport

Messages are transported by using **HTTP** POST as specified in [RFC2616]. These messages are sent via **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)** by default. The **URI** uses the virtual directory /KdcProxy unless otherwise configured. The body of the HTTP message contains the KDC_PROXY_MESSAGE (section 2.2.2).

KDC proxy messages are defined using Abstract Syntax Notation One (ASN.1), as specified in [X680], and encoded using Distinguished Encoding Rules (DER), as specified in [X690] section 10.

2.2   Message Syntax

KKDCP does not alter the syntax of any Kerberos messages.

2.2.1   Namespaces

None.

2.2.2   KDC_PROXY_MESSAGE

This structure is a KDC proxy message that contains the Kerberos message to be proxied and optional information for DC location at the KKDCP server.

```
KDC-PROXY-MESSAGE::= SEQUENCE {
    kerb-message            [0] OCTET STRING,
    target-domain           [1] KERB-REALM OPTIONAL,
    dclocator-hint          [2] INTEGER OPTIONAL
}
```

kerb-message: A Kerberos message.

target-domain: An optional KerberosString that represents the **realm** to which the Kerberos message is sent. This value is not case-sensitive.

dclocator-hint: An optional **Flags** ([MS-NRPC] section 3.5.4.3.1) which contains additional data to be used to find a domain controller for the Kerberos message.

3   Protocol Details

3.1   Client Details

This section describes details of protocol processing that must be understood in order to implement a client that can correctly perform its role in the protocol message exchange.

3.1.1   Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

The KKDCP client has the following configuration setting:

KKDCPServerURL: A string containing the URL of the KKDCP server.

The following parameters are set when the Kerberos client calls ProxyMessage():

KerberosMessage: A temporary variable that contains a Kerberos message.

Error: A temporary variable that contains an error message or NULL. By default, it is set to NULL.

TargetDomain: The realm field of the Kerberos message ([RFC4120] section 5.4.1).

3.1.2   Timers

None.

3.1.3   Initialization

As stated in section 1.5, the KKDCP client MUST be configured with the URL of the KKDCP server.

3.1.4   Higher-Layer Triggered Events

The KKDCP client is triggered when the Kerberos client calls ProxyMessage() and when HTTPS returns an error or data.

3.1.5   Message Processing Events and Sequencing Rules

3.1.5.1   ProxyMessage() Call

Inputs:

- Input_kerb_message OCTET STRING

- Target_domain KERB-REALM - optional

- dclocator-hint INTEGER - optional

Outputs:

- Output_kerb_message OCTET STRING

The ProxyMessage() call enables Kerberos clients to pass Kerberos messages and realm data to the KKDCP client to proxy.

The KKDCP client SHOULD:

Establish an HTTPS connection using **KKDCPServerURL**.

Create a **KDC_PROXY_MESSAGE** (section 2.2.2) where:

kerb-message is set to **KerberosMessage** (section 3.1.1).

target-domain is set to the realm field of the Kerberos message ([RFC4120] section 5.4.1).

dclocator-hint: If the Kerberos client used only **Flags** G and H in **DsrGetDcNameEx2** ([MS-NRPC] section 3.5.4.3.1) when attempting to locate the domain controller, then this setting is not used. Otherwise, it is set to the **Flags** used.

Send the **KDC_PROXY_MESSAGE** using the HTTPS connection to the KKDCP server.

If the KKDCP client receives:

- A Kerberos message reply, the client SHOULD set *Output_kerb_message* to **KerberosMessage** (section 3.1.1) and return SUCCESS.

- Otherwise, the client SHOULD return Error, and SHOULD NOT return *Output_kerb_message*.

3.1.5.2   Receiving a KDC_PROXY_MESSAGE

When the KKDCP client receives the **KDC_PROXY_MESSAGE** (section 2.2.2), it SHOULD set **KerberosMessage** (section 3.1.1) to **KDC_PROXY_MESSAGE.kerb-message**.

3.1.5.3   Receiving a HTTP Error or Dropped Connection

When the KKDCP client receives an HTTP error or dropped connection:

- On HTTP 403 errors, the client SHOULD set **Error** (section 3.1.1) to STATUS_AUTHENTICATION_FIREWALL_FAILED.

- Otherwise, the client SHOULD set **Error** (section 3.1.1) to STATUS_NO_LOGON_SERVERS.

3.1.6   Timer Events

None.

3.1.7   Other Local Events

None.

3.2   Server Details

This section describes details of protocol processing that must be understood to implement a server that can correctly perform its role in the protocol message exchange.

3.2.1   Abstract Data Model

None.

3.2.2   Timers

None.

3.2.3   Initialization

Prior to receiving request messages, the server MUST open an HTTP/HTTPS endpoint, which will receive requests by clients with the URL for which they are configured.

3.2.4   Higher-Layer Triggered Events

None.

3.2.5   Message Processing Events and Sequencing Rules

3.2.5.1   Receiving a KDC_PROXY_MESSAGE

When the KKDCP server receives the **KDC_PROXY_MESSAGE** (section 2.2.2), it SHOULD:

1. Validate that the **KDC_PROXY_MESSAGE.kerb-message** is a well-formed Kerberos message. If not, then the KKDCP server SHOULD drop the connection and stop processing.

2. Before the KKDCP server can send a Kerberos message, it MUST discover the KDC to which the message will be sent. The KKDCP server SHOULD perform the equivalent of calling DsrGetDcNameEx2 ([MS-NRPC] section 3.5.4.3.1) where:

   - *AllowableAccountControlBits* has bits A, B, C, D, E, and F set.

   - *DomainName* is **TargetDomain**.

   - *Flags* is **KDC_PROXY_MESSAGE.dclocator-hint**. If there is no **dclocator-hint** in the message, *Flags* has bits G and H set.

      - If the Kerberos message is "FAST armored", then also set bit U.

   - All other fields are set to NULL.

3. Return the IP address of the DC in *DomainControllerInfo*.**DomainControllerAddress**.

4. Send the **KDC_PROXY_MESSAGE.kerb-message** to the KDC.

3.2.5.2   Receiving a Kerberos Message Response

When the KKDCP server receives the Kerberos message response, it SHOULD:

Create a **KDC_PROXY_MESSAGE** (section 2.2.2) where:

1. **kerb-message** is set to the Kerberos message response.

   - **target-domain** is not used.

   - **dclocator-hint** is not used.

2. Send the **KDC_PROXY_MESSAGE** using the HTTP connection to the KKDCP client.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

4   Protocol Examples

The following sections describe two common scenarios to illustrate the function of the KKDCP.

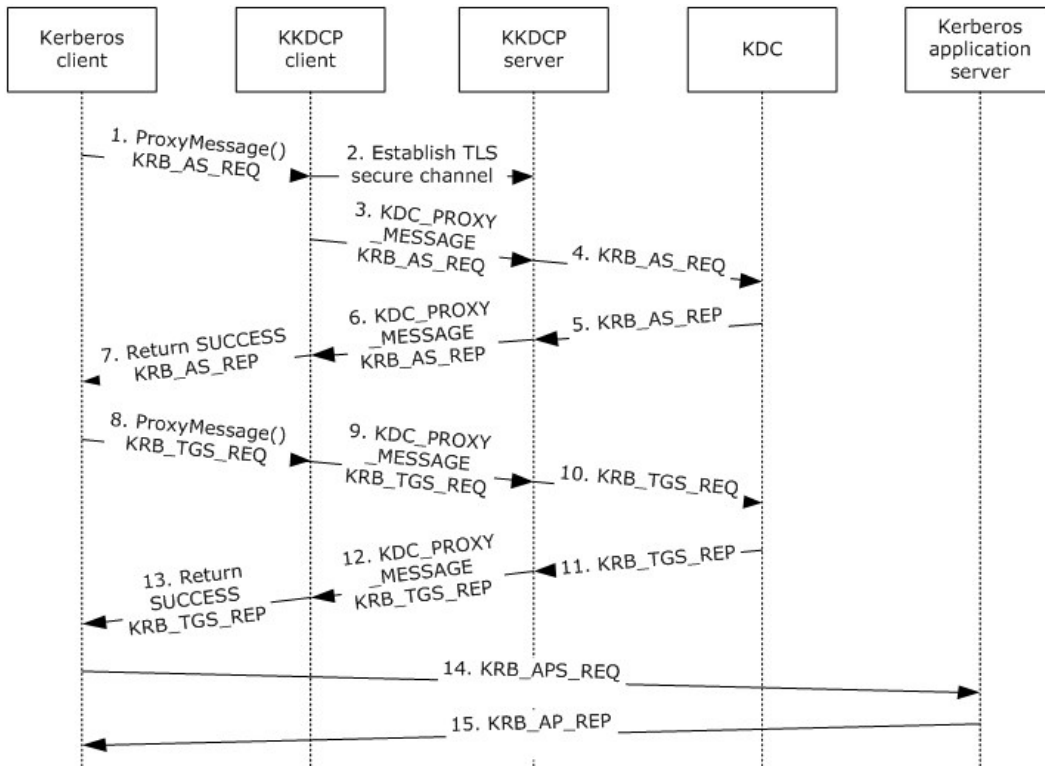## 4.1   Obtaining a Service Ticket

Figure 2: Obtaining a service ticket

When a Kerberos client wants to use Kerberos-based authentication and cannot locate a **DC** for the realm, it uses ProxyMessage() (section 3.1.5.1) to invoke the KKDCP client.

1. Because the Kerberos client does not have a **ticket-granting ticket (TGT)**, it calls ProxyMessage with a KRB_AS_REQ.

2. The KKDCP client establishes a **TLS** secure channel with the KKDCP server.

3. The KKDCP client sends a KDC_PROXY_MESSAGE containing the KRB_AS_REQ to the KKDCP server.

4. The KKDCP server finds the KDC and sends the KRB_AS_REQ to the KDC.

5. The KDC returns a KRB_AS_REP to the KKDCP server.

6. The KKDCP server sends a KDC_PROXY_MESSAGE containing the KRB_AS_REP to the KKDCP client.

7. The KKDCP client returns the KRB_AS_REP and SUCCESS to the Kerberos client.

8. The Kerberos client processes the KRB_AS_REP and calls ProxyMessage with a KRB_TGS_REQ.

9. The KKDCP client sends a KDC_PROXY_MESSAGE containing the KRB_TGS_REQ to the KKDCP server.

10. The KKDCP server finds the KDC and sends the KRB_TGS_REQ to the KDC.

11. The KDC returns a KRB_TGS_REP to the KKDCP server.

12. The KKDCP server sends a KDC_PROXY_MESSAGE containing the KRB_TGS_REP to the KKDCP client.

13. The KKDCP client returns the KRB_TGS_REP and SUCCESS to the Kerberos client.

14. The Kerberos client processes the KRB_TGS_REP and sends a KRB_AP_REQ to the Kerberos application server.

15. The Kerberos application server processes the KRB_AP_REQ and sends a KRB_AP_REP to the Kerberos client.

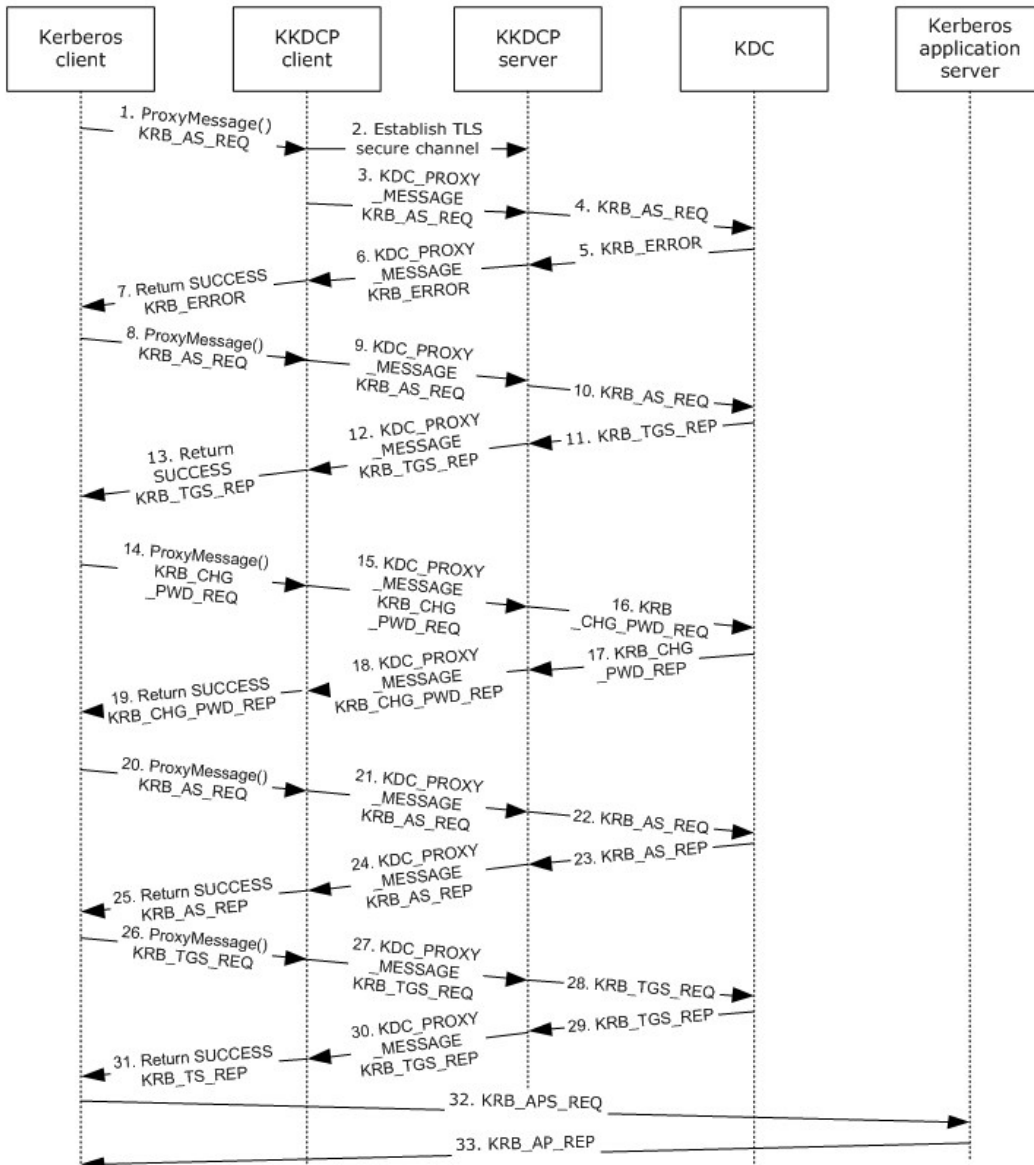## 4.2   Obtaining a Service Ticket with Password Change

Figure 3: Obtaining a service ticket with password change

When a Kerberos client wants to use Kerberos-based authentication and cannot locate a DC for the realm, it uses ProxyMessage() (section 3.1.5.1) to invoke the KKDCP client. If the logon requires the user to change the password prior to logon, applications can use KKDCP for Kerberos password change.

1. Since the Kerberos client does not have a TGT, it calls ProxyMessage with a KRB_AS_REQ.

2. The KKDCP client establishes a TLS secure channel with the KKDCP server.

3. The KKDCP client sends a KDC_PROXY_MESSAGE containing the KRB_AS_REQ to the KKDCP server.

4. The KKDCP server finds the KDC and sends the KRB_AS_REQ to the KDC.

5. The KDC returns KRB_ERROR for password change required before logon to the KKDCP server.

6. The KKDCP server sends a KDC_PROXY_MESSAGE containing the KRB_ERROR to the KKDCP client.

7. The KKDCP client returns the KRB_ERROR and SUCCESS to the Kerberos client.

8. The Kerberos client processes the KRB_ERROR and returns a password change required before logon error to the application. Since the application supports change password, it initiates a Kerberos change password. The Kerberos client calls ProxyMessage with a KRB_AS_REQ for kadmin/changepw.

9. The KKDCP client sends a KDC_PROXY_MESSAGE containing the KRB_AS_REQ to the KKDCP server.

10. The KKDCP server finds the KDC and sends the KRB_AS_REQ to the KDC.

11. The KDC returns a KRB_AS_REP to the KKDCP server.

12. The KKDCP server sends a KDC_PROXY_MESSAGE containing the KRB_AS_REP to the KKDCP client.

13. The KKDCP client returns the KRB_AS_REP and SUCCESS to the Kerberos client.

14. The Kerberos client processes the KRB_AS_REP and creates a Kerberos change password request (KRB_CHG_PWD_REQ) and calls ProxyMessage.

15. The KKDCP client sends a KDC_PROXY_MESSAGE containing the KRB_CHG_PWD_REQ to the KKDCP server.

16. The KKDCP server finds the KDC and sends the KRB_CHG_PWD_REQ to the KDC.

17. The KDC returns a Kerberos change password request (KRB_CHG_PWD_REP) to the KKDCP server.

18. The KKDCP server sends a KDC_PROXY_MESSAGE containing the KRB_CHG_PWD_REP to the KKDCP client.

19. The KKDCP client returns the KRB_CHG_PWD_REP and SUCCESS to the Kerberos client.

20. The Kerberos client processes the KRB_ERROR and returns a password change required before logon error to the application. The application initiates a logon with the new password. The Kerberos client calls ProxyMessage with a KRB_AS_REQ.

21. The KKDCP client sends a KDC_PROXY_MESSAGE containing the KRB_AS_REQ to the KKDCP server.

22. The KKDCP server finds the KDC and sends the KRB_AS_REQ to the KDC.

23. The KDC returns a KRB_AS_REP to the KKDCP server.

24. The KKDCP server sends a KDC_PROXY_MESSAGE containing the KRB_AS_REP to the KKDCP client.

25. The KKDCP client returns the KRB_AS_REP and SUCCESS to the Kerberos client.

26. The Kerberos client processes the KRB_AS_REP and calls ProxyMessage with a KRB_TGS_REQ.

27. The KKDCP client sends a KDC_PROXY_MESSAGE containing the KRB_TGS_REQ to the KKDCP server.

28. The KKDCP server finds the KDC and sends the KRB_TGS_REQ to the KDC.

29. The KDC returns a KRB_TGS_REP to the KKDCP server.

30. The KKDCP server sends a KDC_PROXY_MESSAGE containing the KRB_TGS_REP to the KKDCP client.

31. The KKDCP client returns the KRB_TGS_REP and SUCCESS to the Kerberos client.

32. The Kerberos client processes the KRB_TGS_REP and sends a KRB_AP_REQ to the Kerberos application server.

33. The Kerberos application server processes the KRB_AP_REQ and sends a KRB_AP_REP to the Kerberos client.

5   Security

5.1   Security Considerations for Implementers

Because KKDCP is typically used in the Internet, messages are only protected when HTTPS is used, and the KKDCP server's certificate is valid. When using HTTP, the KKDCP client is sending clear text Kerberos messages, which are vulnerable to attacks discussed in Kerberos V5 ([RFC4120] section 10), unless FAST [RFC6113] is used.

When the KKDCP server relays messages from Internet KKDCP clients to the KDC, it opens unauthenticated access to the KDC from the Internet, unless TLS client authentication is required. KKDCP servers can also provide some level of protection by only relaying valid Kerberos messages, and by throttling messages. KKDCP servers open KDCs to the Internet, exposing them to denial-of-service attacks (using Kerberos messages) that were previously only possible via other authentication protocols, such as NTLM.

5.2   Index of Security Parameters

None.

6   Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs:

- Windows 8 operating system

- Windows Server 2012 operating system

- Windows 8.1 operating system

- Windows Server 2012 R2 operating system

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that the product does not follow the prescription.

7 Change Tracking

This section identifies changes that were made to the [MS-KKDCP] protocol document between the January 2013 and August 2013 releases. Changes are classified as New, Major, Minor, Editorial, or No change.

The revision class **New** means that a new document is being released.

The revision class **Major** means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements or functionality.

- An extensive rewrite, addition, or deletion of major portions of content.

- The removal of a document from the documentation set.

- Changes made for template compliance.

The revision class **Minor** means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class **Editorial** means that the language and formatting in the technical content was changed. Editorial changes apply to grammatical, formatting, and style issues.

The revision class **No change** means that no new technical or language changes were introduced. The technical content of the document is identical to the last released version, but minor editorial and formatting changes, as well as updates to the header and footer information, and to the revision summary, may have been made.

Major and minor changes can be described further using the following change types:

- New content added.

- Content updated.

- Content removed.

- New product behavior note added.

- Product behavior note updated.

- Product behavior note removed.

- New protocol syntax added.

- Protocol syntax updated.

- Protocol syntax removed.

- New content added due to protocol revision.

- Content updated due to protocol revision.

- Content removed due to protocol revision.

- New protocol syntax added due to protocol revision.

- Protocol syntax updated due to protocol revision.

- Protocol syntax removed due to protocol revision.

- New content added for template compliance.

- Content updated for template compliance.

- Content removed for template compliance.

- Obsolete document removed.

Editorial changes are always classified with the change type **Editorially updated.**

Some important terms used in the change type descriptions are defined as follows:

- **Protocol syntax** refers to data elements (such as packets, structures, enumerations, and methods) as well as interfaces.

- **Protocol revision** refers to changes made to a protocol that affect the bits that are sent over the wire.

The changes made to this document are listed in the following table. For more information, please contact protocol@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
2.1 Transport	67160 Added information about how to define KDC proxy messages.	Y	Content updated.
6 Appendix A: Product Behavior	Modified this section to include references to Windows 8.1 and Windows Server 2012 R2.	Y	Content updated.

8 Index

A

Abstract data model
   client 9
   server 10
Applicability 7

C

Capability negotiation 7
Change tracking 20
Client
   abstract data model 9
   higher-layer triggered events 9
   initialization 9
   message processing
     ProxyMessage call 9
     receiving KDC_PROXY_MESSAGE (section
       3.1.5.2 10, section 3.1.5.3 10)
   other local events 10
   overview 9
   sequencing rules
     ProxyMessage call 9
     receiving KDC_PROXY_MESSAGE (section
       3.1.5.2 10, section 3.1.5.3 10)
   timer events 10
   timers 9

D

Data model - abstract
   client 9
   server 10

E

Examples
   obtaining service ticket 13
   obtaining service ticket with password change 15

F

Fields - vendor-extensible 7

G

Glossary 5

H

Higher-layer triggered events
   client 9
   server 11

I

Implementer - security considerations 18
Index of security parameters 18

Informative references 6
Initialization
   client 9
   server 11
Introduction 5

K

KDC_PROXY_MESSAGE message 8

M

Message processing
   client
     ProxyMessage call 9
     receiving KDC_PROXY_MESSAGE (section
       3.1.5.2 10, section 3.1.5.3 10)
   server
     receiving KDC_PROXY_MESSAGE 11
     receiving Kerberos message response 11
Messages
   KDC_PROXY_MESSAGE message 8
   Namespaces message 8
   transport 8

N

Namespaces message 8
Normative references 6

O

Obtaining service ticket example 13
Obtaining service ticket with password change
   example 15
Other local events
   client 10
   server 12
Overview (synopsis) 6

P

Parameters - security index 18
Preconditions 7
Prerequisites 7
Product behavior 19

R

References
   informative 6
   normative 6
Relationship to other protocols 7

S

Security
   implementer considerations 18