Anonymity Support for Kerberos

Abstract

   This document defines extensions to the Kerberos protocol to allow a
   Kerberos client to securely communicate with a Kerberos application
   service without revealing its identity, or without revealing more
   than its Kerberos realm. It also defines extensions that allow a
   Kerberos client to obtain anonymous credentials without revealing its
   identity to the Kerberos Key Distribution Center (KDC). This
   document updates RFCs 4120, 4121, and 4556. This document obsoletes
   RFC 6112 and reclassifies that document as Historic. RFC 6112
   contained errors, and the protocol described in that specification is
   not interoperable with any known implementation. This specification
   describes a protocol that interoperates with multiple
   implementations.

Status of This Memo

   This is an Internet Standards Track document.

   This document is a product of the Internet Engineering Task Force
   (IETF). It represents the consensus of the IETF community. It has
   received public review and has been approved for publication by the
   Internet Engineering Steering Group (IESG). Further information on
   Internet Standards is available in Section 2 of RFC 7841.

   Information about the current status of this document, any errata,
   and how to provide feedback on it may be obtained at
   http://www.rfc-editor.org/info/rfc8062.

Table of Contents

1. Introduction

   In certain situations, the Kerberos [RFC4120] client may wish to
   authenticate a server and/or protect communications without revealing
   the client's own identity. For example, consider an application that
   provides read access to a research database and that permits queries
   by arbitrary requesters. A client of such a service might wish to
   authenticate the service, to establish trust in the information
   received from it, but might not wish to disclose the client's
   identity to the service for privacy reasons.

   Extensions to Kerberos are specified in this document by which a
   client can authenticate the Key Distribution Center (KDC) and request
   an anonymous ticket. The client can use the anonymous ticket to
   authenticate the server and protect subsequent client-server
   communications.

   By using the extensions defined in this specification, the client can
   request an anonymous ticket where the client may reveal the client's
   identity to the client's own KDC, or the client can hide the client's
   identity completely by using anonymous Public Key Cryptography for
   Initial Authentication in Kerberos (PKINIT) as defined in
   Section 4.1. Using the returned anonymous ticket, the client remains

anonymous in subsequent Kerberos exchanges thereafter to KDCs on the
cross-realm authentication path and to the server with which it
communicates.

In this specification, the client realm in the anonymous ticket is
the anonymous realm name when anonymous PKINIT is used to obtain the
ticket. The client realm is the client's real realm name if the
client is authenticated using the client's long-term keys. Note that
a membership in a realm can imply a member of the community
represented by the realm.

The interaction with Generic Security Service Application Program
Interface (GSS-API) is described after the protocol description.

This specification replaces [RFC6112] to correct technical errors in
that specification. RFC 6112 is classified as Historic;
implementation of RFC 6112 is NOT RECOMMENDED. All known
implementations comply with this specification and not RFC 6112.

1.1. Changes since RFC 6112

In Section 7, the pepper2 string "KeyExchange" used in RFC 6112 is
corrected to appear in all capital letters to comply with the string
actually used by implementations.

The requirement for the anonymous option to be used when an anonymous
ticket is used in a Ticket-Granting Service (TGS) request is reduced
from a MUST to a SHOULD. At least one implementation does not
require this; it is not necessary that both the anonymous option and
anonymous ticket be used as an indicator of request type.

The authorization data type name "AD-INITIAL-VERIFIED-CAS" used in
RFC 6112 is corrected to appear as "AD_INITIAL_VERIFIED_CAS" in this
document.

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. Definitions

The anonymous Kerberos realm name is defined as a well-known realm
name based on [RFC6111], and the value of this well-known realm name
is the literal "WELLKNOWN:ANONYMOUS".

The anonymous Kerberos principal name is defined as a well-known
Kerberos principal name based on [RFC6111]. The value of the name-
type field is KRB_NT_WELLKNOWN [RFC6111], and the value of the name-
string field is a sequence of two KerberosString components:
"WELLKNOWN" and "ANONYMOUS".

The anonymous ticket flag is defined as bit 16 (with the first bit
being bit 0) in the TicketFlags:

            TicketFlags   ::= KerberosFlags
               -- anonymous(16)
               -- TicketFlags and KerberosFlags are defined in [RFC4120]

This is a new ticket flag that is used to indicate that a ticket is
an anonymous one.

An anonymous ticket is a ticket that has all of the following
properties:

o The cname field contains the anonymous Kerberos principal name.

o The crealm field contains the client's realm name or the anonymous
   realm name.

o The anonymous ticket contains no information that can reveal the
   client's identity. However, the ticket may contain the client
   realm, intermediate realms on the client's authentication path,
   and authorization data that may provide information related to the
   client's identity. For example, an anonymous principal that is
   identifiable only as being in a particular group of users can be
   implemented using authorization data. Such authorization data, if
   included in the anonymous ticket, would disclose that the client
   is a member of the group observed.

o The anonymous ticket flag is set.

The anonymous KDC option is defined as bit 16 (with the first bit
being bit 0) in the KDCOptions:

            KDCOptions   ::= KerberosFlags
               -- anonymous(16)
               -- KDCOptions and KerberosFlags are defined in [RFC4120]

As described in Section 4, the anonymous KDC option is set to request
an anonymous ticket in an Authentication Service (AS) request or a
Ticket-Granting Service (TGS) request.

4. Protocol Description

   In order to request an anonymous ticket, the client sets the
   anonymous KDC option in an AS request or a TGS request.

   The rest of this section is organized as follows: it first describes
   protocol actions specific to AS exchanges, then it describes those of
   TGS exchanges. These are then followed by the description of
   protocol actions common to both AS and TGS and those in subsequent
   exchanges.

4.1. Anonymity Support in AS Exchange

   The client requests an anonymous ticket by setting the anonymous KDC
   option in an AS exchange.

   The Kerberos client can use the client's long-term keys, the client's
   X.509 certificates [RFC4556], or any other pre-authentication data to
   authenticate to the KDC and request an anonymous ticket in an AS
   exchange where the client's identity is known to the KDC.

   If the client in the AS request is anonymous, the anonymous KDC
   option MUST be set in the request. Otherwise, the KDC MUST return a
   KRB-ERROR message with the code KDC_ERR_BADOPTION.

   If the client is anonymous and the KDC does not have a key to encrypt
   the reply (this can happen when, for example, the KDC does not
   support PKINIT [RFC4556]), the KDC MUST return an error message with
   the code KDC_ERR_NULL_KEY [RFC4120].

   When policy allows, the KDC issues an anonymous ticket. If the
   client name in the request is the anonymous principal, the client
   realm (crealm) in the reply is the anonymous realm; otherwise, the
   client realm is the realm of the AS. As specified by [RFC4120], the
   client name and the client realm in the EncTicketPart of the reply
   MUST match with the corresponding client name and the client realm of
   the KDC reply; the client MUST use the client name and the client
   realm returned in the KDC-REP in subsequent message exchanges when
   using the obtained anonymous ticket.

   The KDC MUST NOT reveal the client's identity in the authorization
   data of the returned ticket when populating the authorization data in
   a returned anonymous ticket.

   The AD_INITIAL_VERIFIED_CAS authorization data, as defined in
   [RFC4556], contains the issuer name of the client certificate. This
   authorization is not applicable and MUST NOT be present in the
   returned anonymous ticket when anonymous PKINIT is used. When the

   client is authenticated (i.e., anonymous PKINIT is not used), if it
   is undesirable to disclose such information about the client's
   identity, the AD_INITIAL_VERIFIED_CAS authorization data SHOULD be
   removed from the returned anonymous ticket.

   The client can use the client's key to mutually authenticate with the
   KDC and request an anonymous Ticket-Granting Ticket (TGT) in the AS
   request. In that case, the reply key is selected as normal,
   according to Section 3.1.3 of [RFC4120].

4.1.1. Anonymous PKINIT

   This sub-section defines anonymous PKINIT.

   As described earlier in this section, the client can request an
   anonymous ticket by authenticating to the KDC using the client's
   identity; alternatively, without revealing the client's identity to
   the KDC, the Kerberos client can request an anonymous ticket as
   follows: the client sets the client name as the anonymous principal
   in the AS exchange and provides PA_PK_AS_REQ pre-authentication data
   [RFC4556] where the signerInfos field of the SignedData [RFC5652] of
   the PA_PK_AS_REQ is empty, and the certificates field is absent.
   Because the anonymous client does not have an associated asymmetric
   key pair, the client MUST choose the Diffie-Hellman key agreement
   method by filling in the Diffie-Hellman domain parameters in the
   clientPublicValue [RFC4556]. This use of the anonymous client name
   in conjunction with PKINIT is referred to as "anonymous PKINIT". If
   anonymous PKINIT is used, the realm name in the returned anonymous
   ticket MUST be the anonymous realm.

   Upon receiving the anonymous PKINIT request from the client, the KDC
   processes the request, according to Section 3.1.2 of [RFC4120]. The
   KDC skips the checks for the client's signature and the client's
   public key (such as the verification of the binding between the
   client's public key and the client name) but performs otherwise
   applicable checks and proceeds as normal, according to [RFC4556].
   For example, the AS MUST check if the client's Diffie-Hellman domain
   parameters are acceptable. The Diffie-Hellman key agreement method
   MUST be used and the reply key is derived according to
   Section 3.2.3.1 of [RFC4556]. If the clientPublicValue is not
   present in the request, the KDC MUST return a KRB-ERROR with the code
   KDC_ERR_PUBLIC_KEY_ENCRYPTION_NOT_SUPPORTED [RFC4556]. If all goes
   well, an anonymous ticket is generated, according to Section 3.1.3 of
   [RFC4120], and PA_PK_AS_REP [RFC4556] pre-authentication data is
   included in the KDC reply, according to [RFC4556]. If the KDC does
   not have an asymmetric key pair, it MAY reply anonymously or reject
   the authentication attempt. If the KDC replies anonymously, the

signerInfos field of the SignedData [RFC5652] of PA_PK_AS_REP in the
reply is empty, and the certificates field is absent. The server
name in the anonymous KDC reply contains the name of the TGS.

Upon receipt of the KDC reply that contains an anonymous ticket and
PA_PK_AS_REP [RFC4556] pre-authentication data, the client can then
authenticate the KDC based on the KDC's signature in the
PA_PK_AS_REP. If the KDC's signature is missing in the KDC reply
(the reply is anonymous), the client MUST reject the returned ticket
if it cannot authenticate the KDC otherwise.

A KDC that supports anonymous PKINIT MUST indicate the support of
PKINIT, according to Section 3.4 of [RFC4556]. In addition, such a
KDC MUST indicate support for anonymous PKINIT by including a padata
element of padata-type PA_PKINIT_KX and empty padata-value when
including PA-PK-AS-REQ in an error reply.

When included in a KDC error, PA_PKINIT_KX indicates support for
anonymous PKINIT. As discussed in Section 7, when included in an
AS-REP, PA_PKINIT_KX proves that the KDC and client both contributed
to the session key for any use of Diffie-Hellman key agreement with
PKINIT.

Note that in order to obtain an anonymous ticket with the anonymous
realm name, the client MUST set the client name as the anonymous
principal in the request when requesting an anonymous ticket in an AS
exchange. Anonymous PKINIT is the only way via which an anonymous
ticket with the anonymous realm as the client realm can be generated
in this specification.

4.2. Anonymity Support in TGS Exchange

The client requests an anonymous ticket by setting the anonymous KDC
option in a TGS exchange, and in that request, the client can use a
normal Ticket-Granting Ticket (TGT) with the client's identity, an
anonymous TGT, or an anonymous cross-realm TGT. If the client uses a
normal TGT, the client's identity is known to the TGS.

Note that the client can completely hide the client's identity in an
AS exchange using anonymous PKINIT, as described in the previous
section.

If the ticket in the PA-TGS-REQ of the TGS request is an anonymous
one, the anonymous KDC option SHOULD be set in the request.

When policy allows, the KDC issues an anonymous ticket. If the
ticket in the TGS request is an anonymous one, the client name and
the client realm are copied from that ticket; otherwise, the ticket
in the TGS request is a normal ticket, the returned anonymous ticket
contains the client name as the anonymous principal and the client
realm as the true realm of the client. In all cases, according to
[RFC4120], the client name and the client realm in the EncTicketPart
of the reply MUST match with the corresponding client name and the
client realm of the anonymous ticket in the reply; the client MUST
use the client name and the client realm returned in the KDC-REP in
subsequent message exchanges when using the obtained anonymous
ticket.

The TGS MUST NOT reveal the client's identity in the authorization
data of the returned ticket. When propagating authorization data in
the ticket or in the enc-authorization-data field of the request, the
TGS MUST ensure that the client confidentiality is not violated in
the returned anonymous ticket. The TGS MUST process the
authorization data recursively, according to Section 5.2.6 of
[RFC4120], beyond the container levels such that all embedded
authorization elements are interpreted. The TGS SHOULD NOT populate
identity-based authorization data into an anonymous ticket in that
such authorization data typically reveals the client's identity. The
specification of a new authorization data type MUST specify the
processing rules of the authorization data when an anonymous ticket
is returned. If there is no processing rule defined for an
authorization data element or the authorization data element is
unknown, the TGS MUST process it when an anonymous ticket is returned
as follows:

o If the authorization data element may reveal the client's
   identity, it MUST be removed unless otherwise specified.

o If the authorization data element that could reveal the client's
   identity is intended to restrict the use of the ticket or limit
   the rights otherwise conveyed in the ticket, it cannot be removed
   in order to hide the client's identity. In this case, the
   authentication attempt MUST be rejected, and the TGS MUST return
   an error message with the code KDC_ERR_POLICY. Note this is
   applicable to both critical and optional authorization data.

o If the authorization data element is unknown, the TGS MAY remove
   it, or transfer it into the returned anonymous ticket, or reject
   the authentication attempt, based on local policy for that
   authorization data type unless otherwise specified. If there is
   no policy defined for a given unknown authorization data type, the
   authentication MUST be rejected. The error code is KDC_ERR_POLICY
   when the authentication is rejected.

The AD_INITIAL_VERIFIED_CAS authorization data, as defined in
[RFC4556], contains the issuer name of the client certificate. If it
is undesirable to disclose such information about the client's
identity, the AD_INITIAL_VERIFIED_CAS authorization data SHOULD be
removed from an anonymous ticket.

The TGS encodes the name of the previous realm into the transited
field, according to Section 3.3.3.2 of [RFC4120]. Based on local
policy, the TGS MAY omit the previous realm, if the cross-realm TGT
is an anonymous one, in order to hide the authentication path of the
client. The unordered set of realms in the transited field, if
present, can reveal which realm may potentially be the realm of the
client or the realm that issued the anonymous TGT. The anonymous
Kerberos realm name MUST NOT be present in the transited field of a
ticket. The true name of the realm that issued the anonymous ticket
MAY be present in the transited field of a ticket.

4.3. Subsequent Exchanges and Protocol Actions Common to AS and TGS for
      Anonymity Support

In both AS and TGS exchanges, the realm field in the KDC request is
always the realm of the target KDC, not the anonymous realm when the
client requests an anonymous ticket.

Absent other information, the KDC MUST NOT include any identifier in
the returned anonymous ticket that could reveal the client's identity
to the server.

Unless anonymous PKINIT is used, if a client requires anonymous
communication, then the client MUST check to make sure that the
ticket in the reply is actually anonymous by checking the presence of
the anonymous ticket flag in the flags field of the EncKDCRepPart.
This is because KDCs ignore unknown KDC options. A KDC that does not
understand the anonymous KDC option will not return an error but will
instead return a normal ticket.

The subsequent client and server communications then proceed as
described in [RFC4120].

Note that the anonymous principal name and realm are only applicable
to the client in Kerberos messages, and the server cannot be
anonymous in any Kerberos message per this specification.

A server accepting an anonymous service ticket may assume that
subsequent requests using the same ticket originate from the same
client. Requests with different tickets are likely to originate from
different clients.

Upon receipt of an anonymous ticket, the transited policy check is
performed in the same way as that of a normal ticket if the client's
realm is not the anonymous realm; if the client realm is the
anonymous realm, absent other information, any realm in the
authentication path is allowed by the cross-realm policy check.

5. Interoperability Requirements

Conforming implementations MUST support the anonymous principal with
a non-anonymous realm, and they MAY support the anonymous principal
with the anonymous realm using anonymous PKINIT.

6. GSS-API Implementation Notes

GSS-API defines the name_type GSS_C_NT_ANONYMOUS [RFC2743] to
represent the anonymous identity. In addition, Section 2.1.1 of
[RFC1964] defines the single string representation of a Kerberos
principal name with the name_type GSS_KRB5_NT_PRINCIPAL_NAME. The
anonymous principal with the anonymous realm corresponds to the
GSS-API anonymous principal. A principal with the anonymous
principal name and a non-anonymous realm is an authenticated
principal; hence, such a principal does not correspond to the
anonymous principal in GSS-API with the GSS_C_NT_ANONYMOUS name type.
The [RFC1964] name syntax for GSS_KRB5_NT_PRINCIPAL_NAME MUST be used
for importing the anonymous principal name with a non-anonymous realm
name and for displaying and exporting these names. In addition, this
syntax must be used along with the name type GSS_C_NT_ANONYMOUS for
displaying and exporting the anonymous principal with the anonymous
realm.

At the GSS-API [RFC2743] level, an initiator/client requests the use
of an anonymous principal with the anonymous realm by asserting the
"anonymous" flag when calling GSS_Init_Sec_Context(). The GSS-API
implementation MAY provide implementation-specific means for
requesting the use of an anonymous principal with a non-anonymous
realm.

GSS-API does not know or define "anonymous credentials", so the
(printable) name of the anonymous principal will rarely be used by or
relevant for the initiator/client. The printable name is relevant
for the acceptor/server when performing an authorization decision
based on the initiator name that is returned from the acceptor side
upon the successful security context establishment.

   A GSS-API initiator MUST carefully check the resulting context
   attributes from the initial call to GSS_Init_Sec_Context() when
   requesting anonymity, because (as in the GSS-API tradition and for
   backwards compatibility) anonymity is just another optional context
   attribute. It could be that the mechanism doesn't recognize the
   attribute at all or that anonymity is not available for some other
   reasons -- and in that case, the initiator MUST NOT send the initial
   security context token to the acceptor, because it will likely reveal
   the initiator's identity to the acceptor, something that can rarely
   be "undone".

   Portable initiators are RECOMMENDED to use default credentials
   whenever possible and request anonymity only through the input
   anon_req_flag [RFC2743] to GSS_Init_Sec_Context().

7. PKINIT Client Contribution to the Ticket Session Key

   The definition in this section was motivated by protocol analysis of
   anonymous PKINIT (defined in this document) in building secure
   channels [RFC6113] and subsequent channel bindings [RFC5056]. In
   order to enable applications of anonymous PKINIT to form secure
   channels, all implementations of anonymous PKINIT need to meet the
   requirements of this section. There is otherwise no connection to
   the rest of this document.

   PKINIT is useful for constructing secure channels. To ensure that an
   active attacker cannot create separate channels to the client and KDC
   with the same known key, it is desirable that neither the KDC nor the
   client unilaterally determine the ticket session key. The specific
   reason why the ticket session key is derived jointly is discussed at
   the end of this section. To achieve that end, a KDC conforming to
   this definition MUST encrypt a randomly generated key, called the
   "KDC contribution key", in the PA_PKINIT_KX padata (defined next in
   this section). The KDC contribution key is then combined with the
   reply key to form the ticket session key of the returned ticket.
   These two keys are combined using the KRB-FX-CF2 operation defined in
   Section 7.1, where K1 is the KDC contribution key, K2 is the reply
   key, the input pepper1 is US-ASCII [ANSI.X3-4] string "PKINIT", and
   the input pepper2 is US-ASCII string "KEYEXCHANGE".

   PA_PKINIT_KX   147
     -- padata for PKINIT that contains an encrypted
     -- KDC contribution key.

   PA-PKINIT-KX ::= EncryptedData -- EncryptionKey
     -- Contains an encrypted key randomly
     -- generated by the KDC (known as the KDC contribution key).
     -- Both EncryptedData and EncryptionKey are defined in [RFC4120]

The PA_PKINIT_KX padata MUST be included in the KDC reply when
anonymous PKINIT is used; it SHOULD be included if PKINIT is used
with the Diffie-Hellman key exchange but the client is not anonymous;
it MUST NOT be included otherwise (e.g., when PKINIT is used with the
public key encryption as the key exchange).

The padata-value field of the PA-PKINIT-KX type padata contains the
DER [X.680] [X.690] encoding of the Abstract Syntax Notation One
(ASN.1) type PA-PKINIT-KX. The PA-PKINIT-KX structure is an
EncryptedData. The cleartext data being encrypted is the DER-encoded
KDC contribution key randomly generated by the KDC. The encryption
key is the reply key, and the key usage number is
KEY_USAGE_PA_PKINIT_KX (44).

The client then decrypts the KDC contribution key and verifies that
the ticket session key in the returned ticket is the combined key of
the KDC contribution key and the reply key as described above. A
conforming client MUST reject anonymous PKINIT authentication if the
PA_PKINIT_KX padata is not present in the KDC reply or if the ticket
session key of the returned ticket is not the combined key of the KDC
contribution key and the reply key when PA-PKINIT-KX is present in
the KDC reply.

This protocol provides a binding between the party that generated the
session key and the Diffie-Hellman exchange used to generate the
reply key. Hypothetically, if the KDC did not use PA-PKINIT-KX, the
client and KDC would perform a Diffie-Hellman key exchange to
determine a shared key, and that key would be used as a reply key.
The KDC would then generate a ticket with a session key encrypting
the reply with the Diffie-Helman agreement. A man-in-the-middle
(MITM) attacker would just decrypt the session key and ticket using
the Diffie-Hellman key from the attacker-KDC Diffie-Hellman exchange
and re-encrypt it using the key from the attacker-client Diffie-
Hellman exchange, while keeping a copy of the session key and ticket.
This protocol binds the ticket to the Diffie-Hellman exchange and
prevents the MITM attack by requiring the session key to be created
in a way that can be verified by the client.

7.1. Combining Two Protocol Keys

   KRB-FX-CF2() combines two protocol keys based on the pseudo-random()
   function defined in [RFC3961].

   Given two input keys, K1 and K2, where K1 and K2 can be of two
   different enctypes, the output key of KRB-FX-CF2(), K3, is derived as
   follows:

    KRB-FX-CF2(protocol key, protocol key, octet string,
            octet string) -> (protocol key)

    PRF+(K1, pepper1) -> octet-string-1
    PRF+(K2, pepper2) -> octet-string-2
    KRB-FX-CF2(K1, K2, pepper1, pepper2) ->
          random-to-key(octet-string-1 ^ octet-string-2)

   Where ^ denotes the exclusive-OR operation. PRF+() is defined as
   follows:

   PRF+(protocol key, octet string) -> (octet string)

   PRF+(key, shared-info) -> pseudo-random(key, 1 || shared-info) ||
               pseudo-random(key, 2 || shared-info) ||
               pseudo-random(key, 3 || shared-info) || ...

   Here the counter value 1, 2, 3, and so on are encoded as a one-octet
   integer. The pseudo-random() operation is specified by the enctype
   of the protocol key. PRF+() uses the counter to generate enough bits
   as needed by the random-to-key() [RFC3961] function for the
   encryption type specified for the resulting key; unneeded bits are
   removed from the tail.

8. Security Considerations

   Since KDCs ignore unknown options, a client requiring anonymous
   communication needs to make sure that the returned ticket is actually
   anonymous. This is because a KDC that does not understand the
   anonymous option would not return an anonymous ticket.

   By using the mechanism defined in this specification, the client does
   not reveal the client's identity to the server, but the client's
   identity may be revealed to the KDC of the server principal (when the
   server principal is in a different realm than that of the client) and
   any KDC on the cross-realm authentication path. The Kerberos client
   MUST verify the ticket being used is indeed anonymous before
   communicating with the server, otherwise, the client's identity may
   be revealed unintentionally.

   In cases where specific server principals must not have access to the
   client's identity (for example, an anonymous poll service), the KDC
   can define the server-principal-specific policy that ensures any
   normal service ticket can NEVER be issued to any of these server
   principals.

   If the KDC that issued an anonymous ticket were to maintain records
   of the association of identities to an anonymous ticket, then someone
   obtaining such records could breach the anonymity. Additionally, the
   implementations of most (for now all) KDCs respond to requests at the
   time that they are received. Traffic analysis on the connection to
   the KDC will allow an attacker to match client identities to
   anonymous tickets issued. Because there are plaintext parts of the
   tickets that are exposed on the wire, such matching by a third-party
   observer is relatively straightforward. A service that is
   authenticated by the anonymous principals may be able to infer the
   identity of the client by examining and linking quasi-static protocol
   information such as the IP address from which a request is received
   or by linking multiple uses of the same anonymous ticket.

   Two mechanisms, the FAST facility with the hide-client-names option
   in [RFC6113] and the Kerberos5 starttls option [RFC6251], protect the
   client identity so that an attacker would never be able to observe
   the client identity sent to the KDC. Transport- or network-layer
   security between the client and the server will help prevent tracking
   of a particular ticket to link a ticket to a user. In addition,
   clients can limit how often a ticket is reused to minimize ticket
   linking.

   The client's real identity is not revealed when the client is
   authenticated as the anonymous principal. Application servers MAY
   reject the authentication in order to, for example, prevent
   information disclosure or as part of Denial-of-Service (DoS)
   prevention. Application servers MUST avoid accepting anonymous
   credentials in situations where they must record the client's
   identity, for example, when there must be an audit trail.

9. IANA Considerations

   This document defines an 'anonymous' Kerberos well-known name and an
   'anonymous' Kerberos well-known realm based on [RFC6111]. IANA has
   updated these two entries in the "Well-Known Kerberos Principal
   Names" and "Well-Known Kerberos Realm Names" registries,
   respectively, to refer to this document.

   In addition, IANA has updated the reference for PA_PKINIT_KX (147) in
   the "Pre-authentication and Typed Data" registry to refer to this
   document.

10. References

10.1. Normative References

   [ANSI.X3-4]
             American National Standards Institute, "Coded Character
             Set - 7-bit American Standard Code for Information
             Interchange", ANSI X3-4, 1986.

   [RFC1964] Linn, J., "The Kerberos Version 5 GSS-API Mechanism",
             RFC 1964, DOI 10.17487/RFC1964, June 1996,
             <http://www.rfc-editor.org/info/rfc1964>.

   [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
             Requirement Levels", BCP 14, RFC 2119,
             DOI 10.17487/RFC2119, March 1997,
             <http://www.rfc-editor.org/info/rfc2119>.

   [RFC2743] Linn, J., "Generic Security Service Application Program
             Interface Version 2, Update 1", RFC 2743,
             DOI 10.17487/RFC2743, January 2000,
             <http://www.rfc-editor.org/info/rfc2743>.

   [RFC3961] Raeburn, K., "Encryption and Checksum Specifications for
             Kerberos 5", RFC 3961, DOI 10.17487/RFC3961, February
             2005, <http://www.rfc-editor.org/info/rfc3961>.

   [RFC4120] Neuman, C., Yu, T., Hartman, S., and K. Raeburn, "The
             Kerberos Network Authentication Service (V5)", RFC 4120,
             DOI 10.17487/RFC4120, July 2005,
             <http://www.rfc-editor.org/info/rfc4120>.

   [RFC4556] Zhu, L. and B. Tung, "Public Key Cryptography for Initial
             Authentication in Kerberos (PKINIT)", RFC 4556,
             DOI 10.17487/RFC4556, June 2006,
             <http://www.rfc-editor.org/info/rfc4556>.

   [RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70,
             RFC 5652, DOI 10.17487/RFC5652, September 2009,
             <http://www.rfc-editor.org/info/rfc5652>.

   [RFC6111] Zhu, L., "Additional Kerberos Naming Constraints",
             RFC 6111, DOI 10.17487/RFC6111, April 2011,
             <http://www.rfc-editor.org/info/rfc6111>.

   [RFC6112] Zhu, L., Leach, P., and S. Hartman, "Anonymity Support for
             Kerberos", RFC 6112, DOI 10.17487/RFC6112, April 2011,
             <http://www.rfc-editor.org/info/rfc6112>.

   [X.680]   International Telecommunications Union, "Information
               technology - Abstract Syntax Notation One (ASN.1):
               Specification of Basic Notation", ITU-T Recommendation
               X.680, ISO/IEC International Standard 8824-1:1998, 1997.

   [X.690]   International Telecommunications Union, "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 8825-1:1998, 1997.

10.2. Informative References

   [RFC5056]   Williams, N., "On the Use of Channel Bindings to Secure
               Channels", RFC 5056, DOI 10.17487/RFC5056, November 2007,
               <http://www.rfc-editor.org/info/rfc5056>.

   [RFC6113]   Hartman, S. and L. Zhu, "A Generalized Framework for
               Kerberos Pre-Authentication", RFC 6113,
               DOI 10.17487/RFC6113, April 2011,
               <http://www.rfc-editor.org/info/rfc6113>.

   [RFC6251]   Josefsson, S., "Using Kerberos Version 5 over the
               Transport Layer Security (TLS) Protocol", RFC 6251,
               DOI 10.17487/RFC6251, May 2011,
               <http://www.rfc-editor.org/info/rfc6251>.

Authors' Addresses

   Larry Zhu
   Microsoft Corporation
   One Microsoft Way
   Redmond, WA 98052
   United States of America

   Email: larry.zhu@microsoft.com

   Paul Leach
   Microsoft Corporation
   One Microsoft Way
   Redmond, WA 98052
   United States of America

   Email: pauljleach@msn.com

   Sam Hartman
   Hadron Industries

   Email: hartmans-ietf@mit.edu

   Shawn Emery (editor)
   Oracle

   Email: shawn.emery@gmail.com