[MS-SFU]:
Kerberos Protocol Extensions:
Service for User and Constrained Delegation Protocol Specification

---

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's Open Specification Promise (available here: http://www.microsoft.com/interop/osp) or the Community Promise (available here: http://www.microsoft.com/interop/cp/default.mspx). 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.
- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail 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

Copyright © 2011 Microsoft Corporation.

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
10/22/2006	0.01		MCPP Milestone 1 Initial Availability
01/19/2007	1.0		MCPP Milestone 1
03/02/2007	1.1		Monthly release
04/03/2007	1.2		Monthly release
05/11/2007	1.3		Monthly release
06/01/2007	1.3.1	Editorial	Revised and edited the technical content.
07/03/2007	1.3.2	Editorial	Revised and edited the technical content.
07/20/2007	1.3.3	Editorial	Revised and edited the technical content.
08/10/2007	1.3.4	Editorial	Revised and edited the technical content.
09/28/2007	1.3.5	Editorial	Revised and edited the technical content.
10/23/2007	1.3.6	Editorial	Revised and edited the technical content.
11/30/2007	1.3.7	Editorial	Revised and edited the technical content.
01/25/2008	1.4	Minor	Updated the technical content.
03/14/2008	1.4.1	Editorial	Revised and edited the technical content.
05/16/2008	1.4.2	Editorial	Revised and edited the technical content.
06/20/2008	1.5	Minor	Updated the technical content.
07/25/2008	1.5.1	Editorial	Revised and edited the technical content.
08/29/2008	1.5.2	Editorial	Revised and edited the technical content.
10/24/2008	1.5.3	Editorial	Revised and edited the technical content.
12/05/2008	1.6	Minor	Updated the technical content.
01/16/2009	1.7	Minor	Updated the technical content.
02/27/2009	1.8	Minor	Updated the technical content.
04/10/2009	1.8.1	Editorial	Revised and edited the technical content.
05/22/2009	1.8.2	Editorial	Revised and edited the technical content.

Copyright © 2011 Microsoft Corporation.

Date	Revision History	Revision Class	Comments
07/02/2009	2.0	Major	Updated and revised the technical content.
08/14/2009	2.0.1	Editorial	Revised and edited the technical content.
09/25/2009	2.1	Minor	Updated the technical content.
11/06/2009	3.0	Major	Updated and revised the technical content.
12/18/2009	4.0	Major	Updated and revised the technical content.
01/29/2010	4.1	Minor	Updated the technical content.
03/12/2010	4.1.1	Editorial	Revised and edited the technical content.
04/23/2010	5.0	Major	Updated and revised the technical content.
06/04/2010	5.1	Minor	Updated the technical content.
07/16/2010	5.2	Minor	Clarified the meaning of the technical content.
08/27/2010	5.2	No change	No changes to the meaning, language, or formatting of the technical content.
10/08/2010	5.3	Minor	Clarified the meaning of the technical content.
11/19/2010	5.3	No change	No changes to the meaning, language, or formatting of the technical content.
01/07/2011	5.4	Minor	Clarified the meaning of the technical content.
02/11/2011	6.0	Major	Significantly changed the technical content.

Copyright © 2011 Microsoft Corporation.

Contents

Copyright © 2011 Microsoft Corporation.

Copyright © 2011 Microsoft Corporation.

1 Introduction

Service for User (S4U) specifies two extensions to the Kerberos Protocol. Collectively, these two extensions enable an application service to obtain a Kerberos service ticket on behalf of a user. The resulting service ticket can be used for:

- The requesting service's own information.
- Access control local to the service's machine, impersonating the user.
- Requests to some other service, impersonating the user.

There are two different S4U extensions. The first is the Service-for-User-to-Self (S4U2self) extension, which allows a service to obtain a Kerberos service ticket to itself on behalf of a user. This enables the service to obtain the user's authorization data that is then used in authorization decisions in the local service.

The second S4U extension is the Service-for-User-to-Proxy (S4U2proxy) extension. This Kerberos extension enables a service to obtain a service ticket on behalf of the user to a second, back end service. This allows back-end services to use Kerberos user credentials as if the user had obtained the service ticket and sent it to the back end service directly. Local policy at the ticket-granting service (TGS) can be used to limit the scope of the S4U2proxy extension.

1.1 Glossary

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

Active Directory
authorization
authorization data
constrained delegation
domain
domain controller (DC)
forwardable
Kerberos principal
key
Key Distribution Center (KDC)
KRB_AP_REQ/KRB_AP_REP
KRB_AS_REQ/KRB_AS_REP
KRB_TGS_REQ/KRB_TGS_REP
principal
privilege attribute certificate (PAC)
realm
service
Service for User (S4U)
Service for User to Proxy (S4U2proxy)
Service for User to Self (S4U2self)
service ticket
session key
ticket
ticket-granting service (TGS)
ticket-granting ticket (TGT)

The following terms are specific to this document:

Copyright © 2011 Microsoft Corporation.

pre-authentication: In the Kerberos protocol, the act of proving identity (and knowledge of a **key**) before the issuance of the initial **ticket-granting ticket (TGT)**, as specified in [RFC4120] sections 5.2.7 and 7.5.2.

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

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-ADA2] Microsoft Corporation, "Active Directory Schema Attributes M", June 2007.

[MS-PAC] Microsoft Corporation, "Privilege Attribute Certificate Data Structure", January 2007.

[Referrals] Raeburn, K., and Zhu, L., "Generating KDC Referrals to Locate Kerberos Realms", February 2008, http://tools.ietf.org/html/draft-ietf-krb-wg-kerberos-referrals-10

[RFC822] Crocker, D.H., "Standard for ARPA Internet Text Messages", STD 11, RFC 822, August 1982, http://www.ietf.org/rfc/rfc0822.txt

[RFC1964] Linn, J., "The Kerberos Version 5 GSS-API Mechanism", RFC 1964, June 1996, http://www.ietf.org/rfc/rfc1964.txt

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

[RFC3280] Housley, R., Polk, W., Ford, W., and Solo, D., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002, http://www.ietf.org/rfc/rfc3280.txt

[RFC3961] Raeburn, K., "Encryption and Checksum Specifications for Kerberos 5", RFC 3961, February 2005, http://www.ietf.org/rfc/rfc3961.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

[RFC4121] Zhu, L., Jaganathan, K., and Hartman, S., "The Kerberos Version 5 Generic Security Service Application Program Interface (GSS-API) Mechanism: Version 2", RFC 4121, July 2005, http://www.ietf.org/rfc/rfc4121.txt

[RFC4757] Jaganathan, K., Zhu, L., and Brezak, J., "The RC4-HMAC Kerberos Encryption Types Used by Microsoft Windows", RFC 4757, December 2006, http://www.ietf.org/rfc/rfc4757.txt

1.2.2 Informative References

[MS-GLOS] Microsoft Corporation, "Windows Protocols Master Glossary", March 2007.

[MS-KILE] Microsoft Corporation, "Kerberos Protocol Extensions", January 2007.

Copyright © 2011 Microsoft Corporation.

1.3 Overview

This protocol extends Kerberos by specifying **Service for User (S4U)** extensions in relation to [RFC4120] and [Referrals].

S4U supports two subprotocols: **Service for User to Self (S4U2self)** and **Service for User to Proxy (S4U2proxy)**. Both of these extensions allow a **service** to request a **ticket** from the **Key Distribution Center (KDC)** on behalf of a user. A ticket can be retrieved by the service to itself by using S4U2self or to another service via S4U2proxy. The client name, **realm**, and **authorization data** in the **service ticket** that uses these extensions are of the user, not of the service making the S4U request. This is in contrast to the Kerberos Protocol specified in [RFC4120] where any service tickets requested by a service will have the client name, realm, and authorization data of that requesting service.

1.3.1 S4U2self

The S4U2self extension allows a service to obtain a service ticket to itself on behalf of a user. The user is identified to the KDC using the user's name and realm. Alternatively, the user may be identified based on the user's certificate. The Kerberos **ticket-granting service (TGS)** request and response messages, **KRB_TGS_REQ** and KRB_TGS_REP, are used along with one of two new data structures. The new PA-FOR-USER data structure is used when the user is identified to the KDC by the user name and realm name. The other structure, PA-S4U-X509-USER, is used when the user certificate is presented to the KDC to obtain the **authorization** information. By obtaining a service ticket to itself on behalf of the user, the service receives the user's authorization data in the ticket.

1.3.2 S4U2proxy

The Service-for-User-to-Proxy (S4U2proxy) extension provides a service that obtains a service ticket to another service on behalf of a user. On Microsoft Windows®, this feature is known as constrained delegation.<1> The Kerberos ticket-granting service (TGS) request and response messages, KRB_TGS_REQ and KRB_TGS_REP, are used along with the new CNAME-IN-ADDL-TKT and S4U_DELEGATION_INFO data structures. The second service is typically a proxy performing some work on behalf of the first service, and the proxy is doing that work under the authorization context of the user.

The S4U2proxy extension requires that the service ticket to the first service has the **forwardable** flag set (see Service 1 in the figure specifying Kerberos Ddelegation with forwarded TGT, section 1.3.3). This ticket may be obtained through an S4U2self protocol exchange.

This feature differs from the Kerberos forwarded-TGT delegation mechanism and the proxy-service-ticket delegation mechanism ([RFC4120] section 2.5) in the following ways:

- The service does not require the user to forward either the user's **ticket-granting ticket (TGT)** or the proxy ticket and the associated **session key**.
- The user does not need to authenticate through Kerberos (the S4U2self extension can be used instead, but this is not a requirement). In other words, the user does not need to have a TGT or a proxy service ticket.
- Local policy can be used to limit the services that can be delegated. This is contradictory to the forwarding-TGT delegation mechanism, as specified in [RFC4120] section 2.6, where a service can delegate to any other service. This is similar to the proxy ticket delegation, as specified in [RFC4120] section 2.5, except the client is not involved in making the delegation decision.
- The client has no control over whether a service can delegate on behalf of the user. The client does not request delegation nor does it pass a forwardable TGT to the service. The client cannot

Copyright © 2011 Microsoft Corporation.

detect that delegation will be, or has been, performed. If local policy allows the service to perform S4U2proxy delegation, this delegation is performed solely at the discretion of the service.

When using the S4U2proxy delegation and forwarded-TGT delegation mechanisms, the delegation is invoked when the server impersonates the client and performs operations on a remote server (such as ldap_bind() or RPC_bind()). The Kerberos Security Support Provider (SSP) will first detect whether the forwarded-TGT delegation mechanism is available (by checking whether there is a forwarded TGT in the local ticket cache); if no forwarded TGT is available, the Kerberos SSP will then try to perform the S4U2proxy delegation.

1.3.3 Protocol Overview

The following figure shows the message sequence for Kerberos delegation with a forwarded ticket-granting ticket (TGT). This is background information designed to show the workings of Kerberos delegation, as specified in [RFC4120] section 2.8. This mechanism is then compared to the Service for User (S4U) extensions.

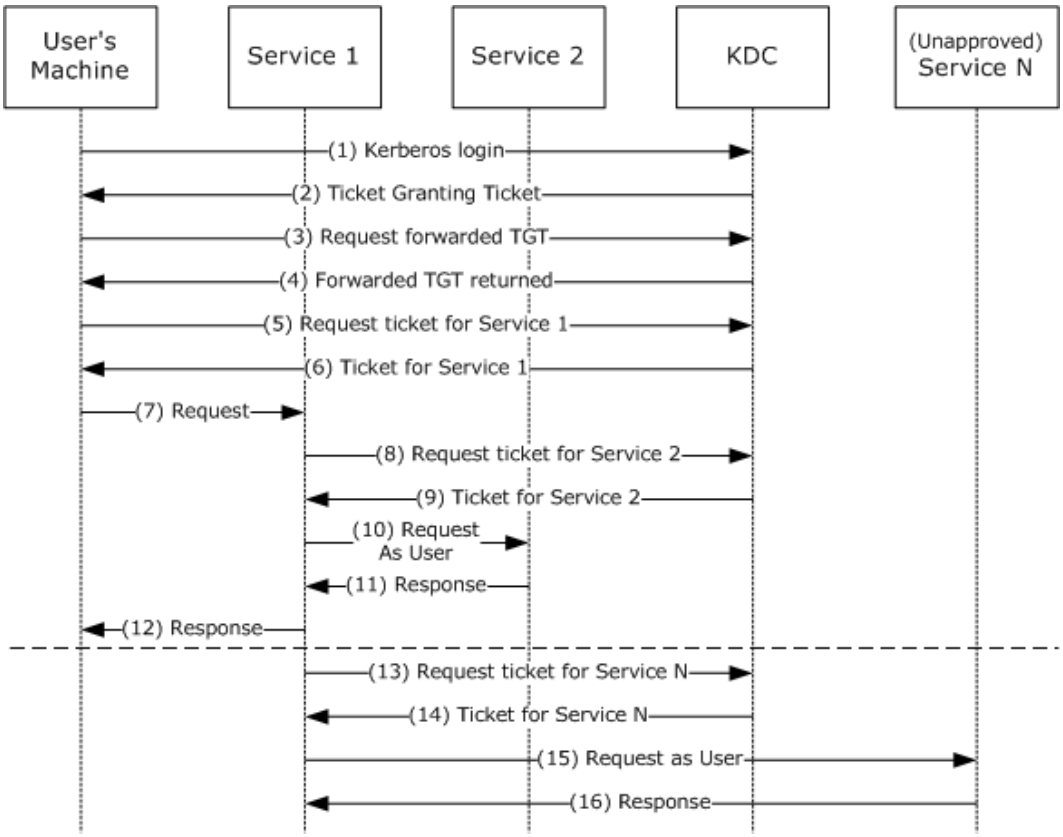

Figure 1: Kerberos Delegation with Forwarded TGT

The preceding depicts the following protocol steps:

1. The user authenticates to the Key Distribution Center (KDC) by sending a **KRB_AS_REQ** message and requests a forwardable TGT.

2. The KDC returns a forwardable TGT in the KRB_AS_REP message.

Copyright © 2011 Microsoft Corporation.

3. The user requests a forwarded TGT based on the forwardable TGT from step 2. This is done by the KRB_TGS_REQ message.

4. The KDC returns a forwarded TGT for the user in the KRB_TGS_REP message.

5. The user makes a request for a service ticket to Service 1 using the TGT returned in step 2. This is done by the KRB_TGS_REQ message.

6. The ticket-granting service (TGS) returns the service ticket in a KRB_TGS_REP message.

7. The user makes a request to Service 1 by sending a **KRB_AP_REQ** message, presenting the service ticket, the forwarded TGT, and the session key for the forwarded TGT.

8. To fulfill the user's request, Service 1 needs Service 2 to perform some action on behalf of the user. Service 1 uses the forwarded TGT of the user and sends that in a KRB_TGS_REQ message to the KDC, asking for a ticket for Service 2 in the name of the user.

9. The KDC returns a ticket for Service 2 to Service 1 in a KRB_TGS_REP message, along with a session key that Service 1 can use. The ticket identifies the client as the user, not as Service 1.

10. Service 1 makes a request to Service 2 by a KRB_AP_REQ, acting as the user.

11. Service 2 responds.

12. With that response, Service 1 can now respond to the user's request in step 7.

13. The TGT forwarding delegation mechanism as described here does not constrain Service 1's use of the forwarded TGT. Service 1 can ask the KDC for a ticket for any other service —in the name of the user.

14. The KDC will return the requested ticket.

15. Service 1 can then continue to impersonate the user with Service N. This can pose a risk if, for example, Service 1 is compromised. Service 1 can continue to masquerade as a legitimate user to other services.

16. Service N will respond to Service 1 as if it was the user's process.

The Server-for-User-to-Self (S4U2self) extension is intended to be used when the user authenticates to the service in some way other than by using Kerberos. For example, a user could authenticate to a Web server by some means private to the Web server. The Web server could then use S4U2self to get a ticket, with authorization data, just as if the user had used Kerberos originally. This simplifies the server's authorization decision by making all decision paths behave as though Kerberos was used. S4U2self primarily uses the KDC to get information about the user for the caller's own benefit. The Service-for-User-to-Proxy (S4U2proxy) extension allows the caller to contact some other service, acting on behalf of the user. The detailed overview is given in the following figure.

Copyright © 2011 Microsoft Corporation.

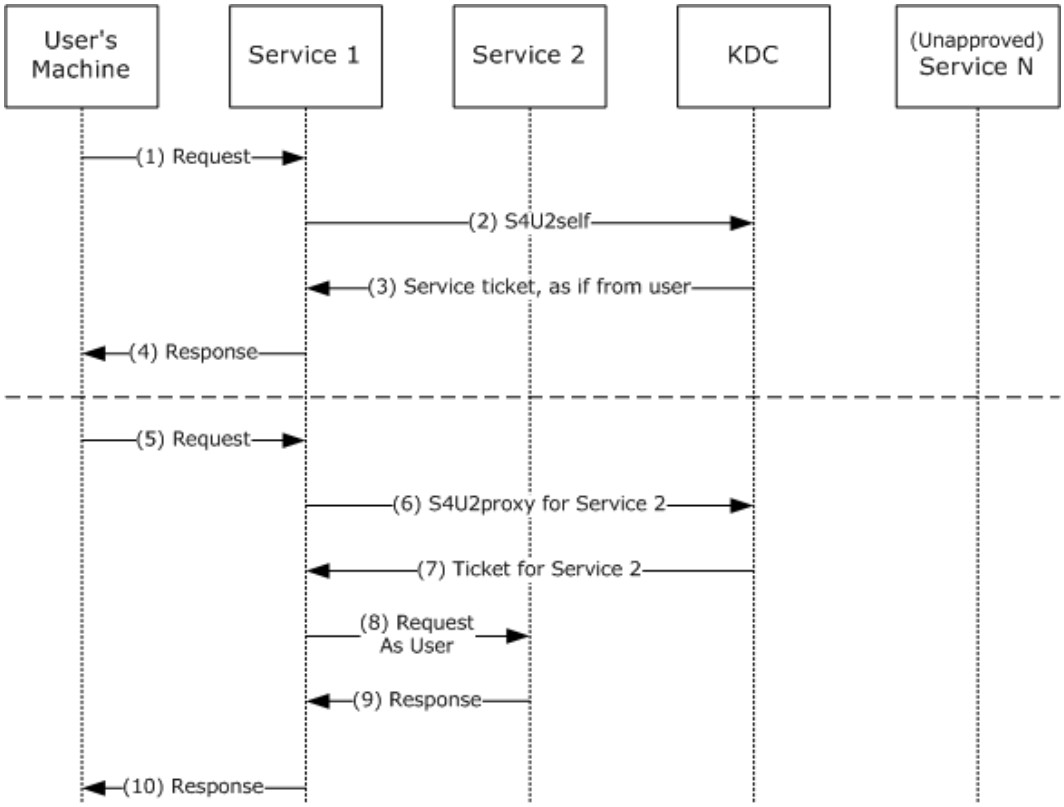

Figure 2: S4U2self and S4U2proxy

S4U2self is described in the top half of the preceding figure. Using this extension, the service receives a service ticket to the service itself (a ticket that cannot be used elsewhere).

The preceding figure depicts the following protocol steps:

1. The user's machine makes a request to Service 1. The user is authenticated, but Service 1 does not have the user's authorization data. Typically this is due to the authentication being performed by some means other than Kerberos.

2. Service 1, which has already authenticated with the KDC and has obtained its TGT, asks for a service ticket to itself on behalf of the named user by the S4U2self extension. The user is identified by the user name and the user's realm name in the S4U2self data (as specified in section 2.2.1). Alternatively, if Service 1 is in possession of the user's certificate, it may use the certificate to identify the user to the KDC using the PA-S4U-X509-USER structure.

3. The KDC returns a service ticket addressed to Service 1 as if it had been requested from the user with the user's own TGT. The service ticket may contain the authorization data of the user.

4. Service 1 can use the authorization data from the service ticket to fulfill the user's request. The service then responds to the user.

   Although S4U2self provides information about the user to Service 1, this extension does not allow Service 1 to make requests of other services on the user's behalf. That is the role of S4U2proxy.

5. The user's machine makes a request to Service 1. Service 1 needs to access resources on Service 2 as the user. However, Service 1 does not have a forwarded TGT from the user to perform

Copyright © 2011 Microsoft Corporation.

delegation by a forwarded TGT, as described in the figure specifying Kerberos delegation with forwarded TGT. Two preconditions apply to this step. First, Service 1 has already authenticated with the KDC and has a valid TGT. Second, Service 1 has a forwardable service ticket from the user to Service 1. This forwardable service ticket may have been obtained by a KRB_AP_REQ, as specified in [RFC4120] section 3.2, or by an S4U2self request.

6. Service 1 requests a service ticket to Service 2 on behalf of the named user. The user is identified by the client name and the client realm in the service ticket for Service 1. The authorization data in the ticket to be returned is also copied from the service ticket. Service 1 and Service 2 must be in the same realm. The user, however, can be in a different realm.

7. The KDC validates the **privilege attribute certificate (PAC)** by checking the signature data of the PAC structure, as specified in [MS-PAC] section 2.2.8. If valid, the KDC returns a service ticket for Service 2, but the client identity stored in the **cname** and **crealm** fields of the service ticket are that of the user, not Service 1.

8. Service 1 uses the service ticket to make a request to Service 2. Service 2 treats this request as coming from the user and assumes that the user was authenticated by the KDC.

9. Service 2 responds to the request.

10. Service 1 responds to the user's request of message 5.<2>

1.4 Relationship to Other Protocols

The S4U extensions are based on the Kerberos Protocol, as specified in [RFC4120]. [RFC4120] also details the dependence on lower-layer protocols such as TCP and UDP. Applications using other protocols may use S4U to create a common authorization path within the application.

The S4U2self extension is used to obtain a privilege attribute certificate (PAC), as specified in [MS-PAC], to determine the authorization capabilities of the user. In addition, the PAC is used in the S4U2proxy extension to validate that S4U2proxy service tickets have not been misused.

The referral mechanism, as specified in [Referrals], is used in the S4U2self protocol extension if the user's realm is different from that of the service trying to obtain an S4U2self service ticket.

1.5 Prerequisites/Preconditions

All Key Distribution Centers (KDCs) and Kerberos servers sending or receiving the Service for User (S4U) extensions in the KRB_TGS_REQ and KRB_TGS_REP messages must recognize the protocol extensions. Services can detect whether the KDC supports these extensions by checking the client name of the returned ticket. KDCs that do not understand these extensions will return the client name as the service that is making the request. KDCs that understand these extensions either return an error or return a service ticket that contains the client name as the user, not the service that is making the request.<3>

To support the lookup of users based on a supplied certificate, the KDC must have an accounts database available to it that supports looking up user accounts using one or more fields present in the certificate.

1.6 Applicability Statement

The Service for User to Proxy (S4U2proxy) extension supports delegation that is transparent to the client. Activities are performed under the user's identity in one or more services. Local policy may be used to limit this functionality and control which services can use this feature.

Copyright © 2011 Microsoft Corporation.

1.7 Versioning and Capability Negotiation

There is no version information in the Service for User (S4U) extensions. A service that uses these extensions will send the new options or data structures in the KRB_TGS_REQ and KRB_TGS_REP messages. Detecting whether a given Key Distribution Center (KDC) can support the extensions is specified in section 1.5.

1.8 Vendor-Extensible Fields

None.

1.9 Standards Assignments

None.

Copyright © 2011 Microsoft Corporation.

2 Messages

2.1 Transport

For more information on the Kerberos Protocol as well as dependencies on lower-level protocols, see [RFC4120], section 7.2.

2.2 Message Syntax

The Service for User (S4U) extensions use new structures conforming to the extensibility mechanisms provided in the Kerberos RFC, as specified in [RFC4120] section 1.5, and new values for options already specified by Kerberos in [RFC4120] section 1.1. The following sections describe these new structures and values.

2.2.1 PA-FOR-USER

In a KRB_TGS_REQ/KRB_TGS_REP subprotocol sequence, as specified in [RFC4120] section 3.3, a **Kerberos principal** uses its ticket-granting ticket (TGT) to request a service ticket to a service. The TGS uses the requesting **principal's** identity from the TGT passed in the KRB_TGS_REQ message to create the service ticket.

In the S4U2self KRB_TGS_REQ/KRB_TGS_REP protocol extension, a service requests a service ticket to itself on behalf of a user. The user is identified to the KDC by the user's name and realm. Alternatively, the user may be identified using the user's certificate. The service uses its own TGT and adds a new type of padata. The padata type is specified in [RFC4120] section 5.2.7.

If the user, on whose behalf the service requests the service ticket, is identified using the user name and user realm, then the padata type, PA-FOR-USER (ID 129), is used. This padata type contains a unique identifier that indicates the user's identity. This unique identifier consists of the user name and user realm.

The PA-FOR-USER padata value is protected with the help of a keyed checksum, as defined below.

The following code defines the ASN.1 structure of the PA-FOR-USER padata type.

```
padata-type     ::= PA-FOR-USER
       -- value 129
padata-value    ::= EncryptedData
                      -- PA-FOR-USER-ENC

PA-FOR-USER-ENC ::= SEQUENCE {
   userName[0] PrincipalName,
   userRealm[1] Realm,
   cksum[2] Checksum,
   auth-package[3] KerberosString
}
```

userName: The PrincipalName type discussed in detail in [RFC4120] section 5.2.2. It consists of a name type and name string. The default value for name type is NT_UNKNOWN as specified in [RFC4120] section 6.2. The name string is a sequence of strings encoded as KerberosString, as specified in [RFC4120] section 5.2.1, that (together with the userRealm) represents a user principal.

userRealm: A KerberosString that represents the realm in which the user account is located. This value is not case-sensitive.

Copyright © 2011 Microsoft Corporation.

cksum: A checksum of *userName*, *userRealm*, and *auth-package*. This is calculated using the **KERB_CHECKSUM_HMAC_MD5** function ([RFC4757]). The value of the **userName.name-type** is first encoded as a 4-byte integer in little endian byte order, then these 4 bytes are concatenated with all string values in the sequence of strings contained in the **userName.name-string** field, then the string value of the **userRealm** field, and then the string value of **auth-package** field, in that order, to form a byte array which can be called S4UByteArray. Note that, in the computation of S4UByteArray, the null terminator is not included when concatenating the strings. Finally **cksum** is computed by calling the **KERB_CHECKSUM_HMAC_MD5** hash with the following three parameters: the session key of the TGT of the service performing the S4U2Self request, the message type value of 17, and the byte array S4UByteArray.

Note The term "message type" is used here as in [RFC4757]. This usage corresponds to the term, "Key Usage Number" used in [RFC4120].

auth-package: A string name of the authentication mechanism used to authenticate the user. This MUST be set to the string, "Kerberos". This value is not case-sensitive.

2.2.2 PA_S4U_X509_USER

If the service possesses the user certificate, it may obtain a service ticket to itself on that user's behalf using the S4U2self KRB_TGS_REQ/KRB_TGS_REP protocol extension, with a new padata type PA-S4U-X509-USER (ID 130).<4> This padata type contains a unique identifier that indicates the user's identity. This unique identifier consists of the user's certificate and, optionally, the user's name and realm.

The following code defines the structure of the PA-S4U-X509-USER padata type.

```
Message Type      padata-type    Contents of padata-value

AS-REQ            130            X509 certificate encoded per
                                 [RFC3280].

TGS-REQ/TGS-REP   130            PA-S4U-X509-USER ASN.1 structure


The corresponding data contains the DER encoded PA-S4U-X509-USER
structure.


PA-S4U-X509-USER::= SEQUENCE {
   user-id[0] S4UUserID,
   checksum[1] Checksum
}

S4UUserID ::= SEQUENCE {
   nonce [0] UInt32, -- the nonce in KDC-REQ-BODY
   cname [1] PrincipalName OPTIONAL,
   -- Certificate mapping hints
   crealm [2] Realm,
   subject-certificate [3] OCTET STRING OPTIONAL,
   options [4] BIT STRING OPTIONAL,
   ...
}
```


Copyright © 2011 Microsoft Corporation.

user-id: Contains the user identifiers. This can be either the user name and realm or the user's certificate.

checksum: This is the Kerberos checksum (as defined in [RFC3961]) computed over the DER encoding of the ASN.1 type S4UUserID contained in the user-id field that immediately precedes this field. The key used is the session key of the TGT used in the TGS request (note that the same key is used in the TGS request and reply when this padata is used in both the request and the reply); the checksum operation is the required checksum for the encryption type of that TGT session key per [RFC3961]; and the key usage is 26. Because there is no required checksum type defined for the encryption type RC4_HMAC_NT (23), if the key's encryption type is RC4_HMAC_NT (23) the checksum type is rsa-md4 (2) as defined in section 6.2.6 of [RFC3961]. If the encryption type is "not-newer" (note that the term "not-newer" is described in section 1 of [RFC4121]), a padata element of type 130 is included in the encrypted-pa-data field of the reply (note that the encrypted-pa-data field is described in appendix A of [Referrals]). The padata of type 130 in the encrypted-pa-data field contains the checksum value in the S4U request concatenated with the checksum value in the S4U reply. The checksum value of a Kerberos Checksum type here refers to the OCTET STRING of the Checksum field. The client when receiving this padata type in the encrypted-pa-data field MUST verify the checksum values match with the corresponding checksum values in the request and the reply.

nonce: This contains the identically named field in the KDC body of the containing request.

cname: The PrincipalName type discussed in detail in [RFC4120] section 5.2.2. It consists of a name type and name string. The default value for the name type is NT_UNKNOWN as specified in [RFC4120] section 6.2. The name string is a sequence of strings encoded as KerberosString, as specified in [RFC4120] section 5.2.1, that (together with the *crealm*) represents a user principal.

crealm: A KerberosString that represents the realm in which the user account is located. This value is not case-sensitive.

subject-certificate: This optional field contains the user's certificate that is encoded as specified in [RFC3280].

options: Specifies the additional options in the S4U request. Currently, only two options are defined.

Value	Meaning
0x40000000	This option causes the KDC to check logon hour restrictions for the user.
0x20000000	In a request, asks the KDC to sign the reply with key usage number 27. In a reply, indicates that it was signed with key usage number 27. This is the KERB_S4U_OPTIONS_use_reply_key_usage (0x20000000).<5> If this option is set in the request, and if the KDC understands this option, it will sign the reply with key usage number 27, and set the same option in the reply. Otherwise, it will sign the reply with key usage number 26 and not set the option in the reply.

In [MS-SFU], the client needs to be able to locate the KDC of the user's realm. If the S4U call is based on the certificate and no user name is supplied, the client uses a PA_S4U_X509_USER padata and the corresponding data contains the user's X509 certificate encoded as specified in [RFC3280].

2.2.3 CNAME-IN-ADDL-TKT

This is a new Key Distribution Center (KDC) option that MUST be set in a KRB_TGS_REQ message to request Service for User to Proxy (S4U2proxy) functionality. The kdc-options flags are specified in

Copyright © 2011 Microsoft Corporation.

[RFC4120] section 5.4.1, and the new cname-in-addl-tkt option is defined as the KDC option with bit position 14.

```
KDCOptions      ::= KerberosFlags
        -- cname-in-addl-tkt (14)
```

2.2.4 S4U_DELEGATION_INFO

The S4U_DELEGATION_INFO structure ([MS-PAC] section 2.9) lists the services that have been delegated by this client and subsequent services or servers. The list is meaningful as the Service-for-User-to-Proxy (S4U2proxy) feature could be used multiple times in succession from service to service. This is useful for auditing purposes.

Copyright © 2011 Microsoft Corporation.

3 Protocol Details

3.1 Service Details

This section defines the message processing for an application service (see Service 1 in the figure specifying entities involved in S4U protocols, section 3.1.5) using the Service for User (S4U) extensions<6>.

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

Before sending a KRB_TGS_REQ message with a Service for User (S4U) extension, the service MUST have already authenticated to the Key Distribution Center (KDC) and received a ticket-granting ticket (TGT).

3.1.4 Higher-Layer Triggered Events

This section contains the following information:

- S4U2self Triggered Events
- S4U2proxy Triggered Events

3.1.4.1 S4U2self Triggered Events

A service (see Service 1 in the figure specifying entities involved in S4U protocols, section 3.1.5) uses a KRB_TGS_REQ message with the S4U2self extension when the service is required to make a local access check for a user. This typically occurs when the user has sent some request to the service through a non-Kerberos protocol. The service uses the S4U2self KRB_TGS_REQ/KRB_TGS_REP protocol extension to obtain authorization data about the user from the Key Distribution Center (KDC).

3.1.4.2 S4U2proxy Triggered Events

A service uses a KRB_TGS_REQ message with the Service for User to Proxy (S4U2proxy) extension when the service determines that it needs to contact another service on behalf of a user for which it has a service ticket. S4U2proxy is used when the request to the second service must use the user's credentials, not the credentials of the first service. The service sends a KRB_TGS_REQ with the S4U2proxy information to obtain a service ticket to another service.

Copyright © 2011 Microsoft Corporation.

3.1.5 Message Processing and Sequencing Rules

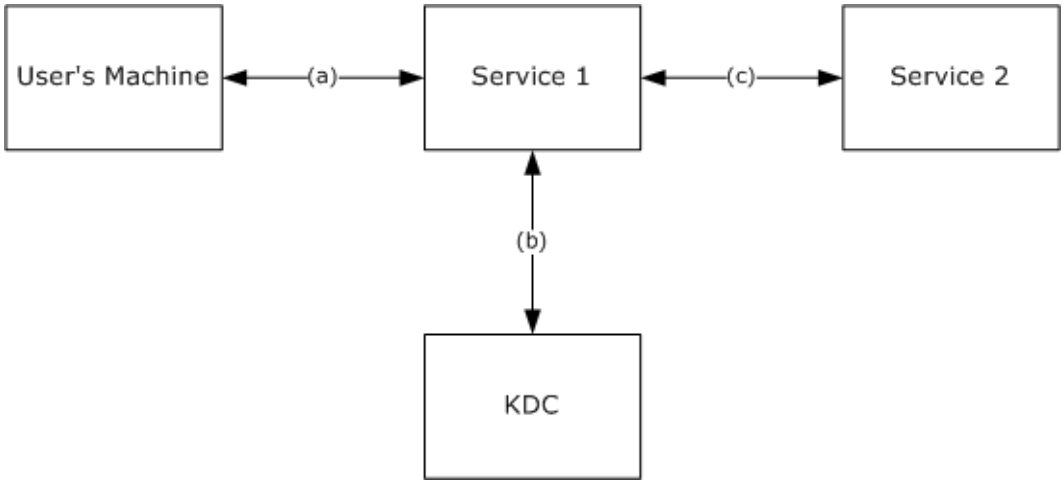

Figure 3: Entities Involved in Service for User (S4U) Protocols

The previous figure shows the entities involved in S4U protocols and the principal communications between them. In the following discussions of processing the S4U messages, it is assumed that Service 1 has started up and has already authenticated itself to its own Key Distribution Center (KDC) via the standard KRB_AS_REQ/KRB_AS_REP exchange (b). In addition, the user has contacted the service and authenticated through some mechanism (a) other than using the KDC. Service 1 authenticates to Service 2 via the application protocol using the standard KRB_AP_REQ/KRB_AP_REP exchange (c).

3.1.5.1 Service Sends S4U2self KRB_TGS_REQ

The Service for User to Self (S4U2self) extension allows Service 1 to use the service's ticket-granting ticket (TGT) in a Kerberos KRB_TGS_REQ message to retrieve a service ticket to the service itself, as if the ticket was originally requested by the user.

In the S4U2self request, the user is identified by the user realm and the user name or alternatively using the user's certificate if the service has it as specified in sections 3.1.5.1.1 and 3.1.5.1.2. The user identification for these cases is carried in a PA-FOR-USER padata or a PA-S4U-X509-USER padata, respectively.

3.1.5.1.1 Using the User's Realm and User Name to Identify the User

Service 1 uses the name and realm of the user to locate the appropriate **domain controller (DC)** to provide the authorization information for the user. The user's realm may be found by local policy, or, if the user name is a User Principal Name, by using KRB_AS_REQ and KRB-ERROR messages as follows. Service 1 sends a KRB_AS_REQ without any **pre-authentication** to Service 1's Key Distribution Center (KDC). If this KDC holds the user's account, then it MUST return KDC_ERR_PREAUTH_REQUIRED, and the user's realm is handled by the KDC. Otherwise, the KDC can refer Service 1 to another realm that may contain the user account or that may have better information about the realm of the user account, as specified in [Referrals] section 4. The KDC does this by returning a KDC_ERR_WRONG_REALM error (as specified in [RFC4120] section 7.5.9) in the KRB_ERROR message and setting the **crealm** field to the next realm to try. Service 1 then sends a KRB_AS_REQ to the next realm, repeating the process until it reaches a KDC in the user's realm or receives some other error.

Copyright © 2011 Microsoft Corporation.

After the realm with the user's account is identified, Service 1 begins the protocol to retrieve the service ticket on behalf of the user. The first step is for the service to retrieve a TGT to the ticket-granting service (TGS) in the user's realm.

If the user's realm is the same as Service 1's realm, the service already has the TGT that it needs. If the user's account is in a different realm, the service constructs a KRB_TGS_REQ with the name of the TGS of the user's realm as the sname field in the request. The cname and crealm fields are set to the name and realm of Service 1. See [RFC4120] section 5.3 for the use of sname and cname. If there is not a direct trust relationship with an inter-realm **key** between Service 1's realm and the user's realm, the service's TGS MUST return a TGT to a realm closer to the user's realm. This process is repeated until Service 1 obtains a TGT to a TGS in the user's realm.

Using the TGT to the TGS in the user's realm, Service 1 requests a service ticket to itself. The S4U2self information in the KRB_TGS_REQ consists of: padata-type = PA-FOR-USER (ID 129), which consists of four fields: **userName**, **userRealm**, **cksum**, and **auth-package**. Service 1 sets these fields as follows: The userName is a structure consisting of a name type and a sequence of a name string (as specified in [RFC4120] section 6.2). The name type and name string fields are set to indicate the name of the user. The default name-type is NT_UNKNOWN. The userRealm is the realm of the user account. If the user 's realm name is unknown, Service 1 SHOULD use its own realm name. The auth-package field MUST be set to the string, "Kerberos". The auth-package field is not case-sensitive.

Multiple intermediate realms may need to be transited. Service 1 MUST send a KRB_TGS_REQ with the S4U2self data in the PA-FOR-USER structure to each TGS in turn along the referral path (as specified in [Referrals]).

The service MUST request a forwardable ticket if it wants to use the returned service ticket as the input for a later Service-for-User-to-Proxy (S4U2proxy) request.

3.1.5.1.2 Using the User's Certificate to Identify the User

If Service 1 has the user certificate, it SHOULD present the certificate to the domain controller (DC) to identify the user.<7> To locate the user account object if the user's name is not available, Service 1 MUST send a KRB_AS_REQ message to its KDC with a PA_S4U_X509_USER (ID 130) padata that contains the client's X509 certificate encoded in ASN.1, as specified in [RFC3280].

3.1.5.2 Service Receives S4U2self KRB_TGS_REP

Services can detect whether the KDC supports S4U by checking the cname of the returned ticket. KDCs that do not support S4U ignore the S4U2self and S4U2proxy data and return a service ticket with the cname containing the name of the service that made the request ([RFC4120] section 3.3.3). In service tickets from KDCs that support S4U, the cname contains the name of the user.

3.1.5.3 Service Sends S4U2proxy KRB_TGS_REQ

As shown in the figure specifying entities involved in Service for User (S4U) protocols in section 3.1.5, Service 1 uses the Service-for-User-to-Proxy (S4U2proxy) extension to request a service ticket to Service 2. By using the S4U2proxy extension, the cname and crealm fields in the resulting service ticket and any authorization data is that of the user, not Service 1. To use this extension, Service 1 and Service 2 MUST belong to the same realm.

Before using the S4U2proxy extension, Service 1 MUST have already authenticated to the Key Distribution Center (KDC) and have a ticket-granting ticket (TGT). Service 1 must also have a service ticket to itself from the user for whom the S4U2proxy request is about to be made. Service 1 requests a service ticket to Service 2 by sending a KRB_TGS_REQ with the S4U2proxy extensions to

Copyright © 2011 Microsoft Corporation.

the TGS. This request MUST include the new cname-in-addl-tkt options flag in the **kdc-options** field. The user's service ticket to Service 1 MUST be placed in the **additional-tickets** field of the request. The **sname** and **realm** fields shall be set to the name and realm of Service 2. The service ticket being passed in the **additional-tickets** field MUST have the forwardable flag set.

3.1.5.4 Service Receives S4U2proxy KRB_TGS_REP

Services can detect whether the KDC supports S4U by checking the cname of the returned ticket. KDCs that do not support S4U ignore the S4U2self and S4U2proxy data and return a service ticket with the cname containing the name of the service that made the request ([RFC4120] section 3.3.3). In service tickets from KDCs that support S4U, the cname contains the name of the user.

Service 1 now has a service ticket to Service 2 with the cname and crealm of the user and authorization data of the user, just as if the user had requested the service ticket. Note, however, that the session key for authenticating to that ticket is owned by Service 1.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 KDC Details

This section defines the message processing for KDCs responding to S4U requests<8>.

3.2.1 Abstract Data Model

To support all functionality of SFU, the account database MUST be extended to support the following additional information for each principal:

- ServicesAllowedToSendForwardedTicketsTo: A list of services to which a service can forward tickets to support constrained delegation. SFU implementations that use an **Active Directory** for the configuration database SHOULD use the **msDS-AllowedToDelegateTo** attribute ([MS-ADA2] section 2.182).
- TrustedToAuthenticationForDelegation: A Boolean setting to control when to set the FORWARDABLE ticket flag ([RFC4120] section 2.6) in S4USelf ([MS-SFU]) service tickets for the principal. SFU implementations that use an Active Directory for the account database SHOULD use the **userAccountControl** attribute ([MS-ADTS] section 2.2.15) TA flag. The default is FALSE.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

Copyright © 2011 Microsoft Corporation.

3.2.5 Message Processing and Sequencing Rules

If an implementation supports the SFU extensions, then the TGS-REQ processing rules in the following sections extend the rules in specified related sections of [RFC4120] and [Referrals].

The SFU KDC MUST copy the populated fields from the PAC in the TGT to the newly created PAC and, after processing all fields it supports, the SFU KDC MUST generate a new Server Signature (section 3.2.5.1) and KDC Signature (section 3.2.5.2) which replace the existing signature fields in the PAC, as discussed in the sections that follow.

3.2.5.1 Server Signature

The KDC creates a keyed hash ([RFC4757]) of the entire PAC message, with the **Signature** fields of both PAC_SIGNATURE_DATA structures set to zero, using the server account key with the strongest cryptography that the domain supports. It also populates the returned PAC_SIGNATURE_DATA structure ([MS-PAC] section 2.8) fields as follows:

- The **SignatureType** field SHOULD be the value ([MS-PAC] section 2.8) corresponding to the cryptographic system used to calculate the checksum.
- The **Signature** field SHOULD be the keyed hash ([RFC4757]) of the entire PAC message, with the **Signature** fields of both PAC_SIGNATURE_DATA structures set to zero.

3.2.5.2 KDC Signatures

The KDC creates a keyed hash ([RFC4757]) of the **Server Signature** field using the strongest "krbtgt" account key, and populates the returned PAC_SIGNATURE_DATA structure ([MS-PAC] section 2.8) fields as follows:

- The **SignatureType** field SHOULD be the value ([MS-PAC] section 2.8) corresponding to the cryptographic system used to calculate the checksum.
- The **Signature** field SHOULD be the keyed hash ([RFC4757]) of the **Server Signature** field in the PAC message.

3.2.5.3 KDC Receives S4U2self KRB_TGS_REQ

When a KDC processes a TGS-REQ ([RFC4120], section 3.3.2) and it is an S4U2self KRB_TGS_REQ, the KDC MUST verify the client name as follows:

- If a referral TGT is received and a PAC is provided, the **Name** field in the **PAC_CLIENT_INFO** structure MUST have the form of "client name@client realm".
- If PA-S4U-X509-USER was sent in KRB_TGS_REQ, the client name and client realm MUST match cname and crealm in the **user_id** field in PA-S4U-X509-USER.
- Otherwise, the client name and client realm MUST match *userName* and *userRealm* in PA-FOR-USER sent in KRB_TGS_REQ.

If any of these verifications fails, the KDC MUST return KDC_ERR_POLICY.

3.2.5.3.1 KDC Replies with Referral TGT

When a KDC determines that a referral TGT is required ([Referrals] section 8), then if Service 1 is not in the KDC's realm, the KDC SHOULD reply with referral TGT ([Referrals]) where:

Copyright © 2011 Microsoft Corporation.

- KRB_TGS_REP cname contains the name of Service 1.
- KRB_TGS_REP crealm contains the realm of Service 1.
- If a PAC is provided, the referral TGT **Name** field in the **PAC_CLIENT_INFO** structure of the PAC contains username@userRealm. This format is the syntax of the single-string representation ([RFC1964] section 2.1.1) using the **username** and **userRealm** fields from the PA-FOR-USER pre-authentication data.

3.2.5.3.2 KDC Replies with Service Ticket

When a KDC processes a TGS-REQ ([RFC4120] section 3.3.2) and if the Service 1 account is in the KDC's realm, the KDC MUST reply with the service ticket, where:

- sname contains the name of Service 1.
- realm contains the realm of Service 1.
- cname contains the **userName** field of the PA-FOR-USER data.
- crealm contains the **userRealm** fields of the PA-FOR-USER data.

If the *TrustedToAuthenticationForDelegation* parameter on the principal is set to:

- **true**: the KDC MUST set the FORWARDABLE ticket flag ([RFC4120] section 2.6) in the S4U2self service ticket.
- **false**: the KDC MUST NOT set the FORWARDABLE ticket flag ([RFC4120] section 2.6) in the S4U2self service ticket.

If the KRB_TGS_REQ contains a PA-S4U-X509-USER padata type, the KDC MUST include the PA-S4U-X509-USER padata type in the KRB_TGS_REP.

3.2.5.4 KDC Receives S4U2proxy KRB_TGS_REQ

When a KDC processes a TGS-REQ ([RFC4120] section 3.3.2) and it is an S4U2proxy KRB_TGS_REQ, the KDC will perform the following steps.

The KDC MUST return KRB-ERR-BADOPTION, if:

- Service 1 and Service 2 do not belong to this realm.
- The SPN for Service 2, identified in the **sname** and **srealm** fields of the KRB_TGS_REQ, is not in the Service 1 account's **ServicesAllowedToSendForwardedTicketsTo** parameter.
- The service ticket in the **additional-tickets** field is not set to **forwardable**.

If the signature of the PAC ([MS-PAC] section 2.8) was not created in this realm, the KDC MUST return KRB-AP-ERR-MODIFIED.

The KDC MUST reply with the service ticket where:

- The **sname** field contains the name of Service 2.
- The **realm** field contains the realm of Service 2.
- The **cname** field contains cname from the service ticket in the **additional-tickets** field.

Copyright © 2011 Microsoft Corporation.

- The **crealm** field contains crealm from the service ticket in the **additional-tickets** field.
- The **FORWARDABLE** ticket flag is set.
- The S4U_DELEGATION_INFO structure is in the new PAC.

The TGS returns the new service ticket in the KRB_TGS_REP message to Service 1.

If the PAC of the service ticket in the **additional-tickets** field does not have an S4U_DELEGATION_INFO structure ([MS-PAC] section 2.9), the KDC MUST add an S4U_DELEGATION_INFO structure to the new PAC where:

- S4U2proxyTarget contains the name of Service 2.
- TransitedListSize is set to 1.

Otherwise, if a PAC was provided, the KDC MUST copy the existing S4U_DELEGATION_INFO structure into the new PAC and increment TransitedListSize by 1.

The KDC MUST also add the name of Service 1 to the S4UTransitedServices list in the structure.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

Copyright © 2011 Microsoft Corporation.

4 Protocol Examples

4.1 S4U2self Single Realm Example

The following figure depicts the S4U2self KRB_TGS_REQ message being processed from the service to the Kerberos TGS. In this case, the user's account belongs to the same realm as the service.

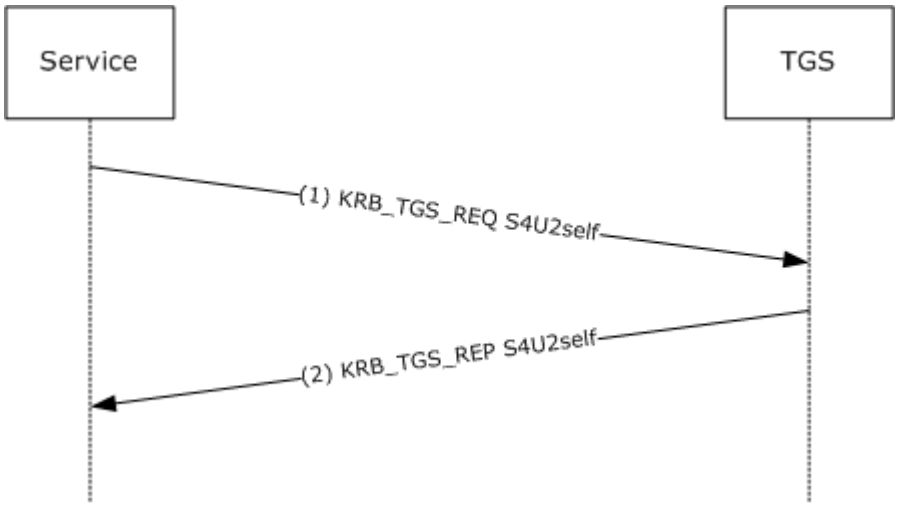

Figure 4: S4U2self KRB_TGS_REQ

The precondition to the previous figure is that the service has already authenticated to the KDC and has a TGT.

In step 1, the service uses the S4U2self extension to retrieve a service ticket to itself on behalf of the user. The service fills out the PA_FOR_USER data structure and sends the KRB_TGS_REQ to the TGS.

Assuming that the TGS supports the PA_FOR_USER extension, the TGS returns the service ticket for the user in the KRB_TGS_REP message in step 2. The privilege attribute certificate (PAC) returned in the service ticket contains the authorization data, as specified in [MS-PAC] section 3. If the service requested the forwardable option and the local policy of the TGS allows it, the TGS shall set the **ticket-flag** field to forwardable.

4.2 S4U2self Multiple Realm Example

The multiple-realm scenario requires extra KRB_TGS_REQ/KRB_TGS_REP exchanges. The service must retrieve a S4U2self service ticket for the user from the service's KDC. To do so, the service must retrieve a TGT for the user to the service's TGS. This is accomplished by first obtaining a TGT to the user's TGS and then following referrals from the user's TGS to the service's TGS.

There are two preconditions to the following figure. The first is that the service has already authenticated to its KDC and has a TGT to its TGS. The second precondition is that the service has identified the realm for the user account. One approach for finding the user's realm is through the use of KRB_AS_REQ messages and using the information returned from the KDC, as specified in [Referrals].

In the following figure, TGS_A represents the TGS in the service's realm. TGS_B represents the TGS in the user's realm.

Copyright © 2011 Microsoft Corporation.

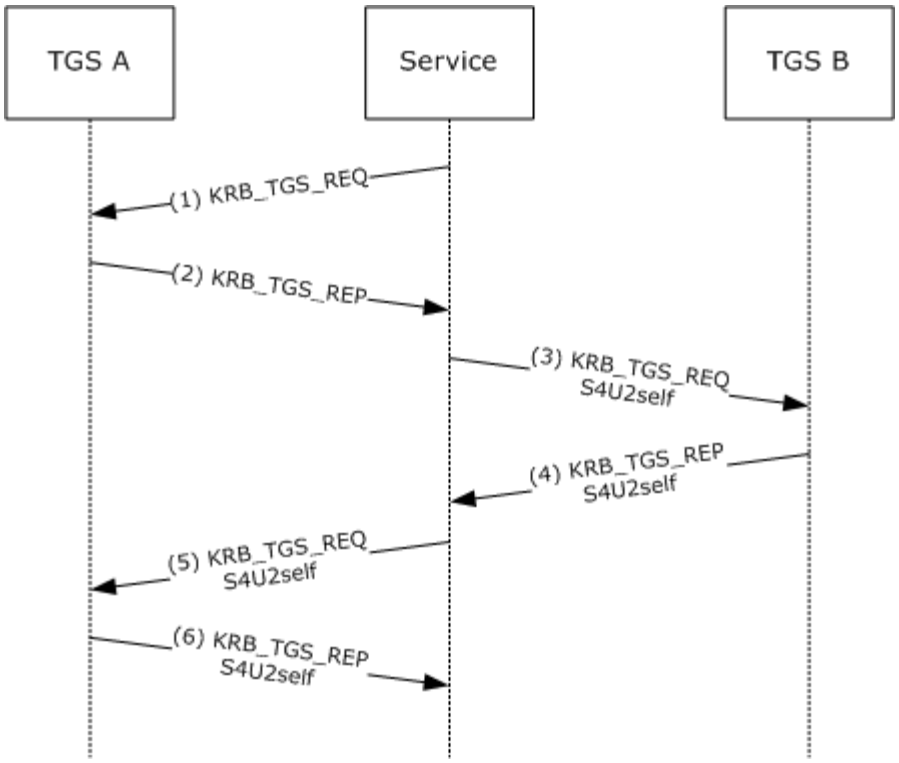

Figure 5: S4U2self Multiple Realm Example

In step 1, the service sends a request to its TGS, TGS_A, for a TGT to TGS_B. No S4U2self information is included in this request. TGS_A responds with the cross-realm TGT to TGS_B in step 2. If TGS_B was not the user's realm but was instead just a realm closer, then the service would send a KRB_TGS_REQ message to TGS_B to get a TGT to the next realm.

The service now uses the TGT to TBS_B to make the S4U2self request in the KRB_TGS_REQ in step 3. The service uses the PA-FOR-USER padata-type field in the request to indicate the user information in the S4U2self request.

TGS_B creates a PAC with the user's authorization information (as specified in [MS-PAC] section 3) and returns it in a TGT referral in the KRB_TGS_REP message in step 4. TGS_B cannot create the service ticket. TGS_B does not possess the service's account information, because the service is part of the realm served by TGS_A.

If there are more TGSs involved in the referral chain, steps 3 and 4 will be repeated to follow the chain.

In step 5, the server uses the TGT from the referral from step 4 and uses the PA-FOR-USER padata-type to request the service ticket to itself on behalf of the user. TGS_A creates the service ticket for the user to the service and returns it in step 6. The PAC returned in this step will contain the appropriate combination of authorization data placed in the PAC by TGS_B in step 4 and the data from TGS_A in step 6, as specified in [MS-PAC] section 4.2.1.

4.3 S4U2proxy Example

The following figure depicts a service obtaining a service ticket on behalf of a client to another service, a proxy service. The ticket-granting service (TGS) is the TGS for both the service and the

Copyright © 2011 Microsoft Corporation.

proxy service. It is also assumed that service has already authenticated to the Key Distribution Center (KDC) and has obtained a ticket-granting ticket (TGT) to the TGS.

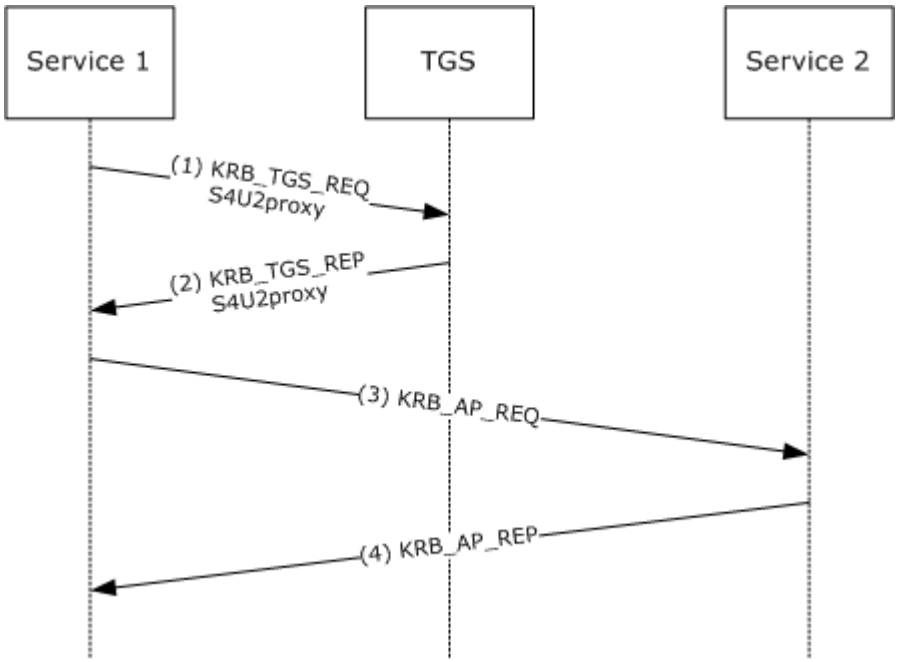

Figure 6: S4U2proxy Example

In step 1, Service 1 is attempting to obtain a service ticket to Service 2 on behalf of the user. Service 1 sends the KRB_TGS_REQ with the user's service ticket to Service 1 as an additional ticket in the request. Service 1 also sets the CNAME-IN-ADDL-TKT flag in the kdc-options in the request.

The TGS makes sure the forwardable flag is set in the additional-ticket and uses its local policy to determine if Service 1 is allowed to obtain a service ticket on behalf of a user to Service 2. If these conditions are met, the TGS crafts the KRB_TGS_REP message to return a service ticket. This response will contain the cname field of the user that is taken from the additional-ticket, instead of using the cname of Service 1. The forwardable flag will be set in the service ticket. The authorization data in the service ticket will be copied from the service ticket passed to the TGS in the additional-tickets field.

In step 3, Service 1 uses the service ticket from step 2 to contact Service 2. The service ticket will contain the user's name as the **cname** field. Step 4 shows the KRB_AP_REP from Service 2 to Service 1 in response to KRB_AP_REQ, as described in step 3.<9>

Copyright © 2011 Microsoft Corporation.

5 Security

5.1 Security Considerations for Implementers

The S4U2self extension allows a service to obtain a service ticket to itself on behalf of a user. This extension is used to obtain authorization data for the user to allow the service to make access control decisions on the local system. As such, the service should be certain to adequately authenticate the user before obtaining the service ticket.

The S4U2proxy extension allows a service to obtain a service ticket to a second service on behalf of a user. When combined with S4U2self, this allows the first service to impersonate any user principal while accessing the second service. This gives any service allowed access to the S4U2proxy extension a degree of power similar to that of the KDC itself. This implies that each of the services allowed to invoke this extension should be protected nearly as strongly as the KDC and should be limited to those that the implementer knows to have correct behavior.

A service can confirm that the service ticket did not originate from the client by the **S4UTransitedServices** field in the S4U_DELEGATION_INFO structure (see [MS-PAC] section 2.9).

5.2 Index of Security Parameters

None.

Copyright © 2011 Microsoft Corporation.

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:

- Microsoft Windows® 2000 operating system
- Windows® XP operating system
- Windows Server® 2003 operating system
- Windows Vista® operating system
- Windows Server® 2008 operating system
- Windows® 7 operating system
- Windows Server® 2008 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.

<1> Section 1.3.2: Constrained delegation is supported on Windows Server 2003, Windows Server 2008 and Windows Server 2008 R2.

<2> Section 1.3.3: The S4U protocol extensions are only supported on Windows Server 2003, Windows Vista, Windows Server 2008, Windows 7 and Windows Server 2008 R2.

<3> Section 1.5: Windows 2000 Server and Windows XP will ignore the S4U_DELEGATION_INFO PAC buffer if it is present. Windows 2000 Server and Windows XP can process the PAC_CLIENT_INFO buffer. For more information, see section 3.1.5.1.

<4> Section 2.2.2: Windows Vista, Windows Server 2008, Windows 7, and Windows Server 2008 R2 will send the PA-S4U-X509-USER padata type alone if the user's certificate is available. If the user's certificate is not available, it will send both the PA-S4U-X509-USER padata type and the PA-FOR-USER padata type. When the PA-S4U-X509-USER padata type is used without the user's certificate, the certificate field is not present.

Windows **domain** controllers starting with Windows Server 2008, will first look for the information in the PA-S4U-X509-USER padata type if present; if it is not present Windows Server 2008 will look at the PA-FOR-USER padata type.

In Windows 2000 Server, Windows Server 2003, and Windows Server 2008 with SP2, KDCs do not add the PA-S4U-X509-USER padata type in the encrypted-pa-data field in TGS-REP.

<5> Section 2.2.2: As of Windows 7 and Windows Server 2008 R2, Windows S4U clients always set this option. If the KDC is Windows Server 2008 R2, it will reply with the same option bit in the reply.

<6> Section 3.1: Windows 2000 and Windows XP do not support S4U.

Copyright © 2011 Microsoft Corporation.

<7> Section 3.1.5.1.2: S4U2self requests using the user's certificate are supported only on Windows Vista, Windows Server 2008, Windows 7, and Windows Server 2008 R2. Windows Vista, Windows Server 2008, Windows 7, and Windows Server 2008 R2 support a number of mapping options:

- Based on the user principal name (UPN) contained in the **SubjectAltName** (SAN) field of the certificate
- Based on the issuer name and subject name combination
- Based on the subject name alone
- Based on subject name and serial number in the certificate
- Based on the subject key identifier
- Based on the SHA1 hash of the public key
- Based on the user's e-mail name as defined in [RFC822]

The algorithm used to locate the user account is as follows:

- If the certificate contains an SAN/UPN extension, KDC will use that to map the client. If the certificate contains an SAN/UPN extension and no user object is found based on the UPN, the authentication fails.
- If there is no UPN in the certificate, the KDC constructs the string "X509:<I><S>" (where "I" is the value from the **Issuer** field and "S" is the value from the **Subject** field in the certificate) to look up.
- If there is no UPN in the certificate and no user object is located in the previous steps, the client account is looked up based on the distinguished name (DN) of the subject; the KDC constructs the "X509:<S>" string (where "S" is the value from the **Subject** field in the certificate) to look up.
- If there is no UPN in the certificate and no user object is located in the previous steps, the KDC uses the subject and serial number to construct the "X509:<S><SR>" string (where "S" is the subject name and "SR" is the serial number from the certificate) to look up.
- If there is no UPN in the certificate and no user object is located, and the client certificate contains a subject key identifier, the KDC constructs the "X509:<SKI>" string (where "SKI" is the subject key identifier) to look up.
- If there is no UPN in the certificate and no user object is located in the previous steps, the KDC constructs the "X509:<SHA1-PUKEY>" string to look up.
- If there is no UPN in the certificate and no user object is located in the previous steps, the client account is looked up based on the SAN/822name, and the KDC constructs the "X509:<RFC822>" string to look up.

<8> Section 3.2: Windows 2000 KDCs do not support S4U.

<9> Section 4.3: The TGS checks the service's account in Active Directory for the Allowed-to-Authenticate-for-Delegation setting. The UserAccountControl flag for this feature is 0x1000000.

The following behavior is applicable to Windows 7 and Windows Server 2008 R2 only: The padata type PA-S4U-X509-USER (ID 130) is used in the encrypted pa-data and the

Copyright © 2011 Microsoft Corporation.

KERB_S4U_OPTIONS_use_reply_key_usage option bit is set (KERB_S4U_OPTIONS_use_reply_key_usage is described in section 2.2.2).

Copyright © 2011 Microsoft Corporation.

7 Change Tracking

This section identifies changes that were made to the [MS-SFU] protocol document between the January 2011 and February 2011 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.

Copyright © 2011 Microsoft Corporation.

- 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
3.1.5 Message Processing and Sequencing Rules	61155 Clarified the description of part (c) fo thefigure.	N	Content updated.
3.2.5 Message Processing and Sequencing Rules	45445 Added details about how the SFU creates a new PAC.	N	Content updated.
3.2.5.1 Server Signature	45445 Added section.	N	New content added.
3.2.5.2 KDC Signatures	45445 Added section.	Y	New content added.

Copyright © 2011 Microsoft Corporation.

8 Index

Copyright © 2011 Microsoft Corporation.

S

T

U

V

Copyright © 2011 Microsoft Corporation.