Troubleshooting Kerberos Encryption Types

Author: Nathan Park (npark@pingidentity.com)
Last update: 2013.12.12

Summary: Troubleshooting Kerberos encryption types can be made easier with a few widely-available tools and troubleshooting methodologies.

The Kerberos 5 protocol supports multiple encryption types (etypes), and mixing some of these etypes can result in unexpected behavior or fallback to NTLM authentication. This extensible encryption type support means that more than one etype can be used in a single transaction. The default etype in Active Directory was RC4-HMAC, and was supported out of the box with Windows 2000/2003. With Windows 2008 and higher, Active Directory supports two new etypes, AES-128 and AES-256. There is also legacy support for DES, which can be used by down-level clients, or non-Windows services and clients that cannot support RC4 or AES. The algorithm that is selected for a given transaction will depend on the highest-strength algorithm supported by the KDC, client, and server.

The table below shows the supported etypes and their corresponding type codes. The codes will be referenced in the PingFederate server log as "keyType=<Code>":

Encryption Type	Code (dec, hex)	Comment
des-cbc-crc	1, 0x1	(legacy) Windows 2000+
des-cbc-md4	2, 0x2	not supported in Windows
des-cbc-md5	3, 0x3	(legacy) Windows 2000+
des3-cbc-sha1	5, 0x5	not supported in Windows
des3-cbc-sha1-kd	16, 0x10	not supported in Windows
aes-128-cts-hmac-sha1-96	17, 0x11	Windows Vista/2008+
aes-256-cts-hmac-sha1-96	18, 0x12	Windows Vista/2008+

rc4-hmac (arcfour-hmac)	23, 0x17	Windows 2000+
rc4-hmac-exp	24, 0x18	Windows 2000+

There are types of Kerberos messages that can use multiple different etypes:

Ticket - A service ticket will be encrypted with the server's key, with the etype used in the Ticket Granting Server (TGS) or Authentication Server (AS) reply. Because the service ticket will be decrypted by the service, it can only be encrypted with an etype supported by the server.

Reply - The etype used by the KDC in replies to the client refers to the section of the reply that will be decrypted by the client. This reply will be encrypted by the highest-strength algorithm supported by the client.

Session Key - The session key is a short-lived (typically 10 hours) secret shared between the client and the server (service), so the etype used for the session key will be the highest-strength algorithm supported by both the client and the server.

Additionally, the Kerberos client on PingFederate dynamically builds what is equivalent to a keytab in memory, which is generated from every available etype supported on the client. PingFederate uses the Domain/Realm Username and Domain/Realm Password configured in the Manage AD Domains/Kerberos Realms section of the PingFederate Admin UI. This user principal's password is passed as a parameter to a function called string2key which transforms the unencrypted password through a hash function into an encryption key suitable for use in the keytab.

Because of the new support for AES in Windows Vista/2008 and higher, the possibility for etype mismatches becomes more significant, especially in environments where there are different versions of Windows, non-Windows Kerberos clients, and IWA services. On top of that, if the environment is running a non-native AD functional level (i.e. domain controllers running different versions of Windows), there may be KDCs that don't support specific etypes, either because of the Windows version or the policy applied to the domain controllers. Support for DES etypes was deprecated with Windows 7/2008 R2, which further exacerbates troubleshooting encryption issues in mixed environments.

The table on the following page shows the exchange of AES tickets depending on support by the KDC, server, and client**

Usage of AES with different Windows operating systems

Client	Server	KDC	Ticket/Message encryption
Operating systems earlier than Windows Vista	Operating systems earlier than Windows Server 2008	Windows Server 2008	TGT might be encrypted with AES based on policy
Operating systems earlier than Windows Vista	Windows Server 2008	Windows Server 2008	Service ticket encrypted with AES
Windows Vista	Windows Server 2008	Windows Server 2008	All tickets and GSS encrypted with AES
Windows Vista	Windows Server 2008	Operating systems earlier than Windows Server 2008	GSS encrypted with AES
Windows Vista	Operating systems earlier than Windows Server 2008	Windows Server 2008	AS-REQ/REP and TGS-REQ/REP encrypted with AES
Operating systems earlier than Windows Vista	Windows Server 2008	Operating systems earlier than Windows Server 2008	No AES
Windows Vista	Operating systems earlier than Windows Server 2008	Operating systems earlier than Windows Server 2008	No AES
Operating systems earlier than Windows Vista	Operating systems earlier than Windows Server 2008	Operating systems earlier than Windows Server 2008	No AES

**Reference: http://technet.microsoft.com/en-us/library/cc749438(v=ws.10).aspx

Klist

Klist is a utility that allows a user to display and purge their Kerberos ticket cache and TGTs. There are a few useful parameters that can be used to display specific information, such as the *tickets*, *tgt*, and *purge* parameters. Klist is available out of the box with Windows Vista/2008 and higher, but for previous versions, you must install the Resource Kit Tools (Reskit) for your version of Windows.

Example 1:

```
C:\Users\npark>klist tickets

Current LogonId is 0:0x15617

Cached Tickets: (2)

#0>     Client: npark @ TESTAD.PINGIDENTITY.COM
        Server: krbtgt/TESTAD.PINGIDENTITY.COM @ TESTAD.PINGIDENTITY.COM
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x40e00000 -> forwardable renewable initial pre_authent
        Start Time: 8/27/2013 9:04:53 (local)
        End Time:   8/27/2013 19:04:53 (local)
        Renew Time: 9/3/2013 9:04:53 (local)
        Session Key Type: RSADSI RC4-HMAC(NT)


#1>     Client: npark @ TESTAD.PINGIDENTITY.COM
        Server: ldap/dc1.testad.pingidentity.com/testad.pingidentity.com @
TESTAD.PINGIDENTITY.COM
        KerbTicket Encryption Type: RSADSI RC4-HMAC(NT)
        Ticket Flags 0x40a40000 -> forwardable renewable pre_authent ok_as_delegate
        Start Time: 8/27/2013 9:04:53 (local)
        End Time:   8/27/2013 19:04:53 (local)
        Renew Time: 9/3/2013 9:04:53 (local)
        Session Key Type: RSADSI RC4-HMAC(NT)
```

The first cached ticket is the user's TGT, which was generated by the AS response when the user logged in. Notice the ticket etype is AES-256, but the session key type is RC4. In the second cached ticket, the ticket and session key type are both RC4. What can be concluded from this example is that the client's highest-strength etype supported is RC4, and the KDC is capable of AES-256. The second ticket is encrypted with RC4, even though the KDC (dc1.testad.pingidentity.com) is capable of AES.

Example 2:

```
C:\Users\npark>klist tickets

Current LogonId is 0:0x201e402

Cached Tickets: (2)
```

```
#0>     Client: npark @ TESTAD.PINGIDENTITY.COM
        Server: krbtgt/TESTAD.PINGIDENTITY.COM @ TESTAD.PINGIDENTITY.COM
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x40e00000 -> forwardable renewable initial pre_authent
        Start Time: 8/27/2013 5:14:45 (local)
        End Time:   8/27/2013 15:14:45 (local)
        Renew Time: 9/2/2013 19:29:47 (local)
        Session Key Type: AES-256-CTS-HMAC-SHA1-96

#1>     Client: npark @ TESTAD.PINGIDENTITY.COM
        Server: ldap/dc1.testad.pingidentity.com/testad.pingidentity.com @
TESTAD.PINGIDENTITY.COM
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x40a40000 -> forwardable renewable pre_authent ok_as_delegate
        Start Time: 8/26/2013 19:04:53 (local)
        End Time:   8/26/2013 5:04:53 (local)
        Renew Time: 9/2/2013 19:04:53 (local)
        Session Key Type: AES-256-CTS-HMAC-SHA1-96

#2>     Client: npark @ TESTAD.PINGIDENTITY.COM
        Server: HTTP/pingfed.testad.pingidentity.com @ TESTAD.PINGIDENTITY.COM
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x40a00000 -> forwardable renewable pre_authent
        Start Time: 8/26/2013 19:29:47 (local)
        End Time:   8/27/2013 5:29:47 (local)
        Renew Time: 9/2/2013 19:29:47 (local)
        Session Key Type: AES-256-CTS-HMAC-SHA1-96
```

The first cached ticket is the user's TGT -- take note the session key is encrypted with AES-256. This client is running Windows 2008 R2, which is capable of this etype. The third cached ticket is for an instance of PingFederate running on hostname pingfed.testad.pingidentity.com. Notice this one is also using AES-256 for both the ticket and session key, which means that the KDC, client, and server (PingFederate in this case) all support AES.

An additional feature of klist -- purge -- allows a client or server to purge Kerberos ticket cache for troubleshooting any AS or TGS transactions. Typically, cache would be purged if changes are made to the client or the server's service principal names (SPNs).

Example 3:

```
C:\Users\npark>klist purge

Current LogonId is 0:0x198a5
        Deleting all tickets:
        Ticket(s) purged!

C:\Users\npark>klist tickets

Current LogonId is 0:0x198a5

Cached Tickets: (0)
```

If a process or Windows service is running as LocalSystem, a tool called psexec.exe can be used to run a command shell as the LocalSystem, from which klist can be ran to display or purge tickets that are cached by the computer account:

```
C:\pstools>psexec -h -s cmd.exe

PsExec v1.98 - Execute processes remotely
Copyright (C) 2001-2010 Mark Russinovich
Sysinternals - www.sysinternals.com


Microsoft Windows [Version 6.0.6002]
Copyright (c) 2006 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>whoami
nt authority\system

C:\Windows\system32>klist tickets

Current LogonId is 0:0x3e7

Cached Tickets: (6)

#0>     Client: pingfed$ @ TESTAD.PINGIDENTITY.COM
        Server: krbtgt/TESTAD.PINGIDENTITY.COM @ TESTAD.PINGIDENTITY.COM
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x60a00000 -> forwardable forwarded renewable pre_authent
        Start Time: 9/9/2013 5:17:06 (local)
        End Time:   9/9/2013 14:53:53 (local)
        Renew Time: 9/12/2013 3:37:38 (local)
        Session Key Type: AES-256-CTS-HMAC-SHA1-96


#1>     Client: pingfed$ @ TESTAD.PINGIDENTITY.COM
        Server: krbtgt/TESTAD.PINGIDENTITY.COM @ TESTAD.PINGIDENTITY.COM
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x40e00000 -> forwardable renewable initial pre_authent
        Start Time: 9/9/2013 4:53:53 (local)
        End Time:   9/9/2013 14:53:53 (local)
        Renew Time: 9/12/2013 3:37:38 (local)
        Session Key Type: AES-256-CTS-HMAC-SHA1-96


#2>     Client: pingfed$ @ TESTAD.PINGIDENTITY.COM
        Server: cifs/DC1.TESTAD.PINGIDENTITY.COM @ TESTAD.PINGIDENTITY.COM
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x40a40000 -> forwardable renewable pre_authent ok_as_delegate
        Start Time: 9/9/2013 5:17:06 (local)
        End Time:   9/9/2013 14:53:53 (local)
        Renew Time: 9/12/2013 3:37:38 (local)
        Session Key Type: AES-256-CTS-HMAC-SHA1-96


#3>     Client: pingfed$ @ TESTAD.PINGIDENTITY.COM
```

```
        Server: pingfed$ @ TESTAD.PINGIDENTITY.COM
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x40a00000 -> forwardable renewable pre_authent
        Start Time: 9/9/2013 5:17:06 (local)
        End Time:   9/9/2013 14:53:53 (local)
        Renew Time: 9/12/2013 3:37:38 (local)
        Session Key Type: AES-256-CTS-HMAC-SHA1-96


#4>     Client: pingfed$ @ TESTAD.PINGIDENTITY.COM
        Server: LDAP/DC1.TESTAD.PINGIDENTITY.COM/TESTAD.PINGIDENTITY.COM @
TESTAD.PINGIDENTITY.COM
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x40a40000 -> forwardable renewable pre_authent ok_as_delegate
        Start Time: 9/9/2013 5:17:04 (local)
        End Time:   9/9/2013 14:53:53 (local)
        Renew Time: 9/12/2013 3:37:38 (local)
        Session Key Type: AES-256-CTS-HMAC-SHA1-96
```

The psexec.exe utility is contained within the PsTools package, and can be downloaded from the Windows Sysinternals Website: http://technet.microsoft.com/en-us/sysinternals/bb896649.aspx.

Debug Logging Kerberos in PingFederate

Degub logging for Kerberos can be enabled in PingFederate to allow an administrator to view the Kerberos protocol transactions coming from the IWA adapter. To enable debug logging for Kerberos, add the following parameter to the JAVA_OPTS in <PF install>\bin\run.bat:

```
-Dsun.security.krb5.debug=true
```

PingFederate must be restarted for the change to take effect. Kerberos transaction debug log entries will now appear in both the server log* and the console's STDOUT stream.

*For PingFederate 6.9 - 7.0.1, the entries only appear in the console's STDOUT stream.

An example of a Kerberos debug log entry from an IWA authentication transaction:

```
17:09:51,606 DEBUG [IWAAuthenticationAdapter] Token Type = KERBEROS
17:09:51,621 DEBUG [IWAAuthenticationAdapter] Negotiate YIIG...FjN64B62DfQVGkFRYKw==
Debug is  true storeKey true useTicketCache false useKeyTab false doNotPrompt false
ticketCache is null isInitiator true
 KeyTab is null refreshKrb5Config is true principal is null tryFirstPass is false
useFirstPass is false storePass is fal
se clearPass is false
Refreshing Kerberos configuration
Config name:
C:\opt\pingfederate\pingfederate-7.0.1\pingfederate\bin\..\server\default\data\krb5.conf
>>> KdcAccessibility: reset
>>> KdcAccessibility: reset
                [Krb5LoginModule] user entered username:
pingfedtest@TESTAD.PINGIDENTITY.COM

Using builtin default etypes for default_tkt_enctypes
default etypes for default_tkt_enctypes: 18 17 16 23 1 3.
>>> KrbAsReq creating message
getKDCFromDNS using UDP
>>> KrbKdcReq send: kdc=dc1.TESTAD.PINGIDENTITY.COM. UDP:88, timeout=3000, number of
retries =3, #bytes=173
>>> KDCCommunication: kdc=dc1.TESTAD.PINGIDENTITY.COM. UDP:88, timeout=3000,Attempt =1,
#bytes=173
>>> KrbKdcReq send: #bytes read=261
>>>Pre-Authentication Data:
         PA-DATA type = 19
         PA-ETYPE-INFO2 etype = 18, salt = TESTAD.PINGIDENTITY.COMpingfedtest, s2kparams
= null
         PA-ETYPE-INFO2 etype = 23, salt = null, s2kparams = null
         PA-ETYPE-INFO2 etype = 3, salt = TESTAD.PINGIDENTITY.COMpingfedtest, s2kparams =
null

>>>Pre-Authentication Data:
         PA-DATA type = 2
         PA-ENC-TIMESTAMP
>>>Pre-Authentication Data:
         PA-DATA type = 16

>>>Pre-Authentication Data:
         PA-DATA type = 15

>>> KdcAccessibility: remove dc1.TESTAD.PINGIDENTITY.COM.
>>> KDCRep: init() encoding tag is 126 req type is 11
>>>KRBError:
         sTime is Wed Aug 28 17:09:51 EDT 2013 1377724191000
         suSec is 70541
         error code is 25
         error Message is Additional pre-authentication required
         realm is TESTAD.PINGIDENTITY.COM
         sname is krbtgt/TESTAD.PINGIDENTITY.COM
         eData provided.
         msgType is 30
>>>Pre-Authentication Data:
         PA-DATA type = 19
         PA-ETYPE-INFO2 etype = 18, salt = TESTAD.PINGIDENTITY.COMpingfedtest, s2kparams
= null
         PA-ETYPE-INFO2 etype = 23, salt = null, s2kparams = null
         PA-ETYPE-INFO2 etype = 3, salt = TESTAD.PINGIDENTITY.COMpingfedtest, s2kparams =
null

>>>Pre-Authentication Data:
         PA-DATA type = 2
         PA-ENC-TIMESTAMP
>>>Pre-Authentication Data:
         PA-DATA type = 16

>>>Pre-Authentication Data:
         PA-DATA type = 15
```

```
KrbAsReqBuilder: PREAUTH FAILED/REQ, re-send AS-REQ
Using builtin default etypes for default_tkt_enctypes
default etypes for default_tkt_enctypes: 18 17 16 23 1 3.
Using builtin default etypes for default_tkt_enctypes
default etypes for default_tkt_enctypes: 18 17 16 23 1 3.
>>> EType: sun.security.krb5.internal.crypto.Aes256CtsHmacSha1EType
>>> KrbAsReq creating message
getKDCFromDNS using UDP
>>> KrbKdcReq send: kdc=dc1.TESTAD.PINGIDENTITY.COM. UDP:88, timeout=3000, number of
retries =3, #bytes=255
>>> KDCCommunication: kdc=dc1.TESTAD.PINGIDENTITY.COM. UDP:88, timeout=3000,Attempt =1,
#bytes=255
>>> KrbKdcReq send: #bytes read=112
>>> KrbKdcReq send: kdc=dc1.TESTAD.PINGIDENTITY.COM. TCP:88, timeout=3000, number of
retries =3, #bytes=255
>>> KDCCommunication: kdc=dc1.TESTAD.PINGIDENTITY.COM. TCP:88, timeout=3000,Attempt =1,
#bytes=255
>>>DEBUG: TCPClient reading 1517 bytes
>>> KrbKdcReq send: #bytes read=1517
>>> KdcAccessibility: remove dc1.TESTAD.PINGIDENTITY.COM.
>>> EType: sun.security.krb5.internal.crypto.Aes256CtsHmacSha1EType
>>> KrbAsRep cons in KrbAsReq.getReply pingfedtest
Using builtin default etypes for default_tkt_enctypes
default etypes for default_tkt_enctypes: 18 17 16 23 1 3.
principal is pingfedtest@TESTAD.PINGIDENTITY.COM
EncryptionKey: keyType=18 keyBytes (hex dump)=0000: 12 5C 9C D0 94 2D 05 2A   28 F2 5F C4
7A A6 06 7B  .\...-.*(._.z...
0010: 27 2A 42 97 3A 1F D5 0E   63 C1 DF 63 E7 15 5C DF  '*B.:...c..c..\.

EncryptionKey: keyType=17 keyBytes (hex dump)=0000: EC BC FE 6B F7 EB 6A 20   E0 4D 05 57
FD 33 4E 05  ...k..j .M.W.3N.

EncryptionKey: keyType=16 keyBytes (hex dump)=0000: DF A4 8A FB 4C B6 34 B9   4F 2A 58 F1
4F 2A 7A 1C  ....L.4.O*X.O*z.
0010: 26 16 46 13 3B 9E A1 62                            &.F.;..b

EncryptionKey: keyType=23 keyBytes (hex dump)=0000: BE B0 65 85 38 B4 C2 6A   CC 04 91 9D
C7 06 7B EE  ..e.8..j........

EncryptionKey: keyType=1 keyBytes (hex dump)=0000: 23 4F 97 68 70 4F F1 EF
#O.hpO..

EncryptionKey: keyType=3 keyBytes (hex dump)=0000: 23 4F 97 68 70 4F F1 EF
#O.hpO..

Added server's keyKerberos Principal pingfedtest@TESTAD.PINGIDENTITY.COMKey Version 0key
EncryptionKey: keyType=18 keyBytes
 (hex dump)=
0000: 12 5C 9C D0 94 2D 05 2A   28 F2 5F C4 7A A6 06 7B  .\...-.*(._.z...
0010: 27 2A 42 97 3A 1F D5 0E   63 C1 DF 63 E7 15 5C DF  '*B.:...c..c..\.


                [Krb5LoginModule] added Krb5Principal
pingfedtest@TESTAD.PINGIDENTITY.COM to Subject
                             Added server's keyKerberos Principal
pingfedtest@TESTAD.PINGIDENTITY.COMKey Version 0key EncryptionKey: keyType=17 keyBytes
```

```
 (hex dump)=
0000: EC BC FE 6B F7 EB 6A 20   E0 4D 05 57 FD 33 4E 05  ...k..j .M.W.3N.


                [Krb5LoginModule] added Krb5Principal
pingfedtest@TESTAD.PINGIDENTITY.COM to Subject
Added server's keyKerberos Principal pingfedtest@TESTAD.PINGIDENTITY.COMKey Version 0key
EncryptionKey: keyType=16 keyBytes
 (hex dump)=
0000: DF A4 8A FB 4C B6 34 B9   4F 2A 58 F1 4F 2A 7A 1C  ....L.4.O*X.O*z.
0010: 26 16 46 13 3B 9E A1 62                            &.F.;..b


                [Krb5LoginModule] added Krb5Principal
pingfedtest@TESTAD.PINGIDENTITY.COM to Subject
Added server's keyKerberos Principal pingfedtest@TESTAD.PINGIDENTITY.COMKey Version 0key
EncryptionKey: keyType=23 keyBytes
 (hex dump)=
0000: BE B0 65 85 38 B4 C2 6A   CC 04 91 9D C7 06 7B EE  ..e.8..j........


                [Krb5LoginModule] added Krb5Principal
pingfedtest@TESTAD.PINGIDENTITY.COM to Subject
Added server's keyKerberos Principal pingfedtest@TESTAD.PINGIDENTITY.COMKey Version 0key
EncryptionKey: keyType=1 keyBytes
(hex dump)=
0000: 23 4F 97 68 70 4F F1 EF                            #O.hpO..


                [Krb5LoginModule] added Krb5Principal
pingfedtest@TESTAD.PINGIDENTITY.COM to Subject
Added server's keyKerberos Principal pingfedtest@TESTAD.PINGIDENTITY.COMKey Version 0key
EncryptionKey: keyType=3 keyBytes
(hex dump)=
0000: 23 4F 97 68 70 4F F1 EF                            #O.hpO..


                [Krb5LoginModule] added Krb5Principal
pingfedtest@TESTAD.PINGIDENTITY.COM to Subject
Commit Succeeded

17:09:52,012 DEBUG [KerberosContextManager] Adding Kerberos Context for Domain/Realm:
TESTAD.PINGIDENTITY.COM to cache.
Found KerberosKey for pingfedtest@TESTAD.PINGIDENTITY.COM
Found KerberosKey for pingfedtest@TESTAD.PINGIDENTITY.COM
Found KerberosKey for pingfedtest@TESTAD.PINGIDENTITY.COM
Found KerberosKey for pingfedtest@TESTAD.PINGIDENTITY.COM
Found KerberosKey for pingfedtest@TESTAD.PINGIDENTITY.COM
Found KerberosKey for pingfedtest@TESTAD.PINGIDENTITY.COM
Entered Krb5Context.acceptSecContext with state=STATE_NEW
>>> EType: sun.security.krb5.internal.crypto.ArcFourHmacEType
Using builtin default etypes for permitted_enctypes
default etypes for permitted_enctypes: 18 17 16 23 1 3.
>>> EType: sun.security.krb5.internal.crypto.ArcFourHmacEType
replay cache for npark@TESTAD.PINGIDENTITY.COM is null.
object 0: 1377724191004/4391
>>> KrbApReq: authenticate succeed.
```

```
Krb5Context setting peerSeqNumber to: 1861708578
>>> EType: sun.security.krb5.internal.crypto.ArcFourHmacEType
Krb5Context setting mySeqNumber to: 948288845
17:09:52,059 DEBUG [KerberosValidator] userNameWithDomain retrieved succesfully.
17:09:52,059 DEBUG [KerberosValidator] IWASubject created succesfully.
```

In the above example, a Kerberos token is received by the IWA adapter. PingFederate goes through a Kerberos authentication in the following steps:

1. There is no ticket cache on PingFederate, so PF must go to the KDC's Authentication Server (AS) and request a Ticket Granting Ticket (TGT). This is referred to as the Authentication Server Request (AS-REQ). Note that the default etypes used by PF are set to 18 (AES-256), 17 (AES-128), 16 (TripleDES), 23 (RC4), 1, and 3 (single DES). The etypes are preferred in this order, with AES-256 being the highest priority and single DES being the lowest. PF then generates the AS-REQ noted in this log entry:
```
Using builtin default etypes for default_tkt_enctypes
default etypes for default_tkt_enctypes: 18 17 16 23 1 3.
>>> KrbAsReq creating message
getKDCFromDNS using UDP
```

2. When the AS-REQ is sent, a KRBError is immediately returned by the AS, with the preferred pre-authentication data types and supported etypes.

   Note that the encryption types supported by the AS are 18 (AES-256), 23 (RC4), and 3 (DES). The preferred data types at the AS are 2 (encrypted timestamp), 16 (pkinit), and 15 (pkinit - deprecated), in that order. This means that data type 2 and AES-256 (type 18) will be used for pre-authentication:
```
>>>KRBError:
         sTime is Wed Aug 28 17:09:51 EDT 2013 1377724191000
         suSec is 70541
         error code is 25
         error Message is Additional pre-authentication required
         realm is TESTAD.PINGIDENTITY.COM
         sname is krbtgt/TESTAD.PINGIDENTITY.COM
         eData provided.
         msgType is 30
>>>Pre-Authentication Data:
         PA-DATA type = 19
         PA-ETYPE-INFO2 etype = 18, salt = TESTAD.PINGIDENTITY.COMpingfedtest,
s2kparams = null
         PA-ETYPE-INFO2 etype = 23, salt = null, s2kparams = null
         PA-ETYPE-INFO2 etype = 3, salt = TESTAD.PINGIDENTITY.COMpingfedtest,
s2kparams = null

>>>Pre-Authentication Data:
         PA-DATA type = 2
         PA-ENC-TIMESTAMP
>>>Pre-Authentication Data:
         PA-DATA type = 16

>>>Pre-Authentication Data:
        PA-DATA type = 15
```

PF then generates the pre-authentication data by encrypting a timestamp with PF's long-term key (the service account's password) with AES-256 using the Kerberos string2key function and performs another AS-REQ. The following log entries show the AS-REQ reattempted with fresh pre-authentication data.

```
KrbAsReqBuilder: PREAUTH FAILED/REQ, re-send AS-REQ
Using builtin default etypes for default_tkt_enctypes
default etypes for default_tkt_enctypes: 18 17 16 23 1 3.
Using builtin default etypes for default_tkt_enctypes
default etypes for default_tkt_enctypes: 18 17 16 23 1 3.
>>> EType: sun.security.krb5.internal.crypto.Aes256CtsHmacSha1EType
>>> KrbAsReq creating message
getKDCFromDNS using UDP
```

3. Once the AS validates the pre-authentication message, the AS replies with a TGT, which contains copies of PF's sessions keys that can be used in any TGS exchanges later. Notice that session keys (referred to as "`KerberosKey`" in the logs) have been created in all supported algorithms. Once this step is complete PingFederate will not need to re-authenticate to the AS for another 10 hours. This is the default lifetime of a TGT within an AD Kerberos environment.

4. From this point on, PF can begin servicing Authentication Protocol Requests (AP-REQ) beginning with this following log entry:

```
Entered Krb5Context.acceptSecContext with state=STATE_NEW
>>> EType: sun.security.krb5.internal.crypto.ArcFourHmacEType
```

   PingFederate now takes the user's service ticket and decrypts it with PF's RC4-HMAC long-term keys which were originally generated during the keytab creation at the top of the log (etype is referred to as "`ArcFourHmacEType`" in the log entries).

5. Once decrypted, the ticket data is validated (timestamp is compared, client principal is compared to principal in AP-REQ, ticket lifetime is verified, and client IP address is verified). If all of these criteria are met, the user is authenticated to PF.

```
>>> EType: sun.security.krb5.internal.crypto.ArcFourHmacEType
replay cache for npark@TESTAD.PINGIDENTITY.COM is null.
object 0: 1377724191004/4391
>>> KrbApReq: authenticate succeed.
```

6. PF then extract's the user's principal name from the decrypted service ticket (`npark@TESTAD.PINGIDENTITY.COM`) and creates an IWASubject that will be used to populate the IWA Adapter Attributes, `${Username}` and `${Domain/Realm Name}`.

Troubleshooting

Some typical etype-related issues and troubleshooting:

1. **GSSException: Failure unspecified at GSS-API level (Mechanism level: Encryption type AES256 CTS mode with HMAC SHA1-96 is not supported/enabled).**

   This is usually refers to a mismatch between the etype used to encrypt the service ticket and the supported etypes used by PingFederate. The above example happens most frequently in mixed Windows environments where some Domain Controllers may support higher-strength algorithms than other DCs in the domain. The latest versions of the IWA Kit (3.0+) support all of the etypes available in Active Directory 2008.

   When authenticating to PingFederate via the IWA adapter, if the client gets a WWW-Authenticate in the HTTP request/response headers with both Negotiate and NTLM, but gets prompted for NTLM authentication; AND they do a `klist -tickets` and have a service ticket for `HTTP/<PF hostname>`:

   This can usually mean one or both of the following is true:
   1) The Windows service account being used for Kerberos authentication in PingFederate is forced to use AES-128 or AES-256 (check boxes on AD account “This account supports Kerberos AES xxx bit encryption”).
   2) The Java Cryptography Extensions (JCE) have not been installed in the JDK that PingFederate uses. These JAR files are required for Java to use higher-strength cryptographic algorithms such as AES-256. They can be found here:

   JDK6 -
   http://www.oracle.com/technetwork/java/javase/downloads/jce-6-download-429243.html

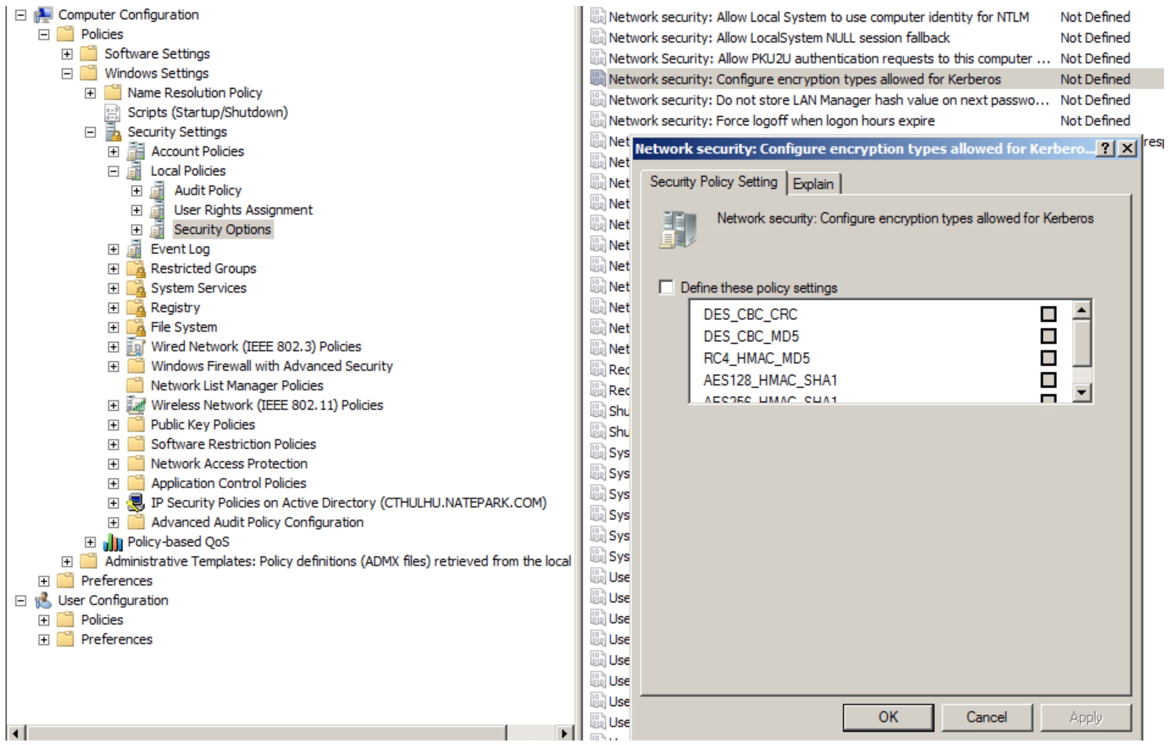
   JDK7 -
   http://www.oracle.com/technetwork/java/javase/downloads/jce-7-download-432124.html

   If another MIT Kerberos 5 implementation other than Active Directory is being used to authenticate to PingFederate, please refer to the implementation’s documentation. Typically, the etypes must be defined explicitly via the default_tgs_enctypes, default_tkt_enctypes, and permitted_enctypes settings in the [libdefaults] stanza of the krb5.conf file such as:

   ```
   [libdefaults]
   default_tkt_enctypes = des-cbc-md5 des-cbc-crc des3-cbc-sha1
   default_tgs_enctypes = des-cbc-md5 des-cbc-crc des3-cbc-sha1
   ```

```
permitted_enctypes = des-cbc-md5 des-cbc-crc des3-cbc-sha1
```

In Active Directory environments, the Kerberos etypes can be configured in Group Policy in the following location: Computer Configuration\Policies\Windows Settings\Security Settings\Local Policies\Security Options\Network security: Configure encryption types allowed for Kerberos. If left undefined, which is the default setting, Active Directory will use the highest-strength algorithm. Also note that the DES encryption types have been deprecated in Server 2008 and higher. If legacy Kerberos systems require these, you must define a policy and link it to your domain.



Computer Configuration
- Policies
  - Software Settings
  - Windows Settings
    - Name Resolution Policy
    - Scripts (Startup/Shutdown)
    - Security Settings
      - Account Policies
      - Local Policies
        - Audit Policy
        - User Rights Assignment
        - Security Options
      - Event Log
      - Restricted Groups
      - System Services
      - Registry
      - File System
      - Wired Network (IEEE 802.3) Policies
      - Windows Firewall with Advanced Security
      - Network List Manager Policies
      - Wireless Network (IEEE 802.11) Policies
      - Public Key Policies
      - Software Restriction Policies
      - Network Access Protection
      - Application Control Policies
      - IP Security Policies on Active Directory (CTHULHU.NATEPARK.COM)
      - Advanced Audit Policy Configuration
    - Policy-based QoS
  - Administrative Templates: Policy definitions (ADMX files) retrieved from the local
- Preferences

User Configuration
- Policies
- Preferences

Network security: Allow Local System to use computer identity for NTLM	Not Defined
Network security: Allow LocalSystem NULL session fallback	Not Defined
Network Security: Allow PKU2U authentication requests to this computer ...	Not Defined
Network security: Configure encryption types allowed for Kerberos	Not Defined
Network security: Do not store LAN Manager hash value on next passwo...	Not Defined
Network security: Force logoff when logon hours expire	Not Defined

Network security: Configure encryption types allowed for Kerbero...

Security Policy Setting | Explain

Network security: Configure encryption types allowed for Kerberos

☐ Define these policy settings

- ☐ DES_CBC_CRC
- ☐ DES_CBC_MD5
- ☐ RC4_HMAC_MD5
- ☐ AES128_HMAC_SHA1
- ☐ AES256_HMAC_SHA1

OK | Cancel | Apply

In Active Directory, also check the user's settings in the Account tab for the following values:
- Use Kerberos DES encryption types for this account
- This account supports Kerberos AES 128 bit encryption.
- This account supports Kerberos AES 256 bit encryption.
If in a mixed environment with Domain Controllers running Windows 2000/2003 and DCs running 2008+, enabling AES on the account may cause issues if a user's TGT session keys are generated with AES and the user attempts to use these session keys for Service Tickets generated on Windows 2000/2003 DCs. Additionally, DES encryption types were deprecated with Windows 2008, so if account principals are configured to use DES, you must either: 1) enable DES via Group Policy as described above; or 2) disable DES on the account options.

2. **javax.security.auth.login.LoginException: KrbException: KDC has no support for encryption type (14) - KDC has no support for encryption type**

   This error means that the KDC does not support the request encryption type. For environments that enforce a minimum-strength encryption algorithm defined by policy or compliance, check to make sure the krb5.conf [libdefaults] contain the correct entries.

   This exception can also be thrown when using native ticket cache on some Windows platforms. Microsoft has added a new feature in which they no longer export the session keys for Ticket-Granting Tickets (TGTs). As a result, the native TGT obtained on Windows has an "empty" session key and null EType. The effected platforms include: Windows Server 2003, Windows 2000 Server Service Pack 4 (SP4) and Windows XP SP2. To remedy:

   Update the Windows registry to disable this new feature. The registry key allowtgtsessionkey should be added--and set correctly--to allow session keys to be sent in the Kerberos Ticket-Granting Ticket. On the Windows Server 2003 and Windows 2000 SP4, here is the required registry setting:
   HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Lsa\Kerberos\Parameters
   Value Name: allowtgtsessionkey
   Value Type: REG_DWORD Value: 0x01 (default is 0)
   By default, the value is 0; setting it to "0x01" allows a session key to be included in the TGT. Here is the location of the registry setting on Windows XP SP2:
   HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Lsa\Kerberos\
   Value Name: allowtgtsessionkey
   Value Type: REG_DWORD Value: 0x01

3. **INFO [STDOUT] [Krb5LoginModule] authentication failed Clock skew too great (37)'.**

   Make sure the PingFederate server's clock is synchronized with the Active Directory domain or forest. Remember that the pre-authentication data type used by Active Directory is always an encrypted timestamp, so if the time on the PingFederate server is +/- five minutes, pre-authentication will fail, and no TGT will be generated.

4. **Preauthentication failed while getting initial credentials**

   In Active Directory Kerberos, pre-authentication is generated by an encrypted timestamp. The user's password acts as the long-term key used to encrypt the timestamp. If the user types the wrong password, a preauthentication failure will occur and be recorded in the DC's System log.

It can also be caused by the KDC not finding a suitable encryption key for the user based on either misconfigured policy (see Group Policy above).

This error may also occur if invalid data is contained within the [libdefaults] stanza of the krb5.conf file in non-Windows Kerberos implementations.

References:
http://web.mit.edu/kerberos/krb5-devel/doc/admin/etypes.html
http://technet.microsoft.com/en-us/library/cc749438(v=ws.10).aspx
http://blogs.technet.com/b/askds/archive/2010/10/19/hunting-down-des-in-order-to-securely-deploy-kerberos.aspx
http://www.kerberos.org/software/tutorial.html
Garman, Jason. *Kerberos: The Definitive Guide.* O'Reilly & Associates, 2003

Disclaimer