NAME

krb5_425_conv_principal, **krb5_425_conv_principal_ext**, **krb5_524_conv_principal** **-** converts to and from version 4 principals

LIBRARY

Kerberos 5 Library (libkrb5, -lkrb5)

SYNOPSIS

#include <krb5.h>

krb5_error_code
krb5_425_conv_principal(*krb5_context context*, *const char *name*, *const char *instance*, *const char *realm*, *krb5_principal *principal*);

krb5_error_code
krb5_425_conv_principal_ext(*krb5_context context*, *const char *name*, *const char *instance*, *const char *realm*, *krb5_boolean (*func)(krb5_context, krb5_principal)*, *krb5_boolean resolve*, *krb5_principal *principal*);

krb5_error_code
krb5_524_conv_principal(*krb5_context context*, *const krb5_principal principal*, *char *name*, *char *instance*, *char *realm*);

DESCRIPTION

Converting between version 4 and version 5 principals can at best be described as a mess.

A version 4 principal consists of a name, an instance, and a realm. A version 5 principal consists of one or more components, and a realm. In some cases also the first component/name will differ between version 4 and version 5. Furthermore the second component of a host principal will be the fully qualified domain name of the host in question, while the instance of a version 4 principal will only contain the first part (short hostname). Because of these problems the conversion between principals will have to be site customized.

krb5_425_conv_principal_ext() will try to convert a version 4 principal, given by *name*, *instance*, and *realm*, to a version 5 principal. This can result in several possible principals, and if *func* is non-NULL, it will be called for each candidate principal. *func* should return true if the principal was "good". To accomplish this, **krb5_425_conv_principal_ext**() will look up the name in *krb5.conf*. It first looks in the v4_name_convert/host subsection, which should contain a list of version 4 names whose instance should be treated as a hostname. This list can be specified for each realm (in the realms section), or in the libdefaults section. If the name is found the resulting name of the principal will be the value of this binding. The instance is then first looked up in v4_instance_convert for the specified realm. If found the resulting value will be used as instance (this can be used for special cases), no further attempts will be made to find a conversion if this fails (with *func*). If the *resolve* parameter is true, the instance will be looked up with **gethostbyname**(). This can be a time consuming, error prone, and unsafe operation. Next a list of hostnames will be created from the instance and the v4_domains variable, which should contain a list of possible domains for the specific realm.

On the other hand, if the name is not found in a host section, it is looked up in a v4_name_convert/plain binding. If found here the name will be converted, but the instance will be untouched.

This list of default host-type conversions is compiled-in:

```
      v4_name_convert = {
              host = {
                      ftp = ftp
                      hprop = hprop
                      imap = imap
                      pop = pop
                      rcmd = host
                      smtp = smtp
              }
      }
```

It will only be used if there isn't an entry for these names in the config file, so you can override these defaults.

krb5_425_conv_principal() will call **krb5_425_conv_principal_ext**() with NULL as *func*, and the value of v4_instance_resolve (from the libdefaults section) as *resolve*.

krb5_524_conv_principal() basically does the opposite of **krb5_425_conv_principal**(), it just doesn't have to look up any names, but will instead truncate instances found to belong to a host principal. The *name*, *instance*, and *realm* should be at least 40 characters long.

EXAMPLES

Since this is confusing an example is in place.

Assume that we have the "foo.com", and "bar.com" domains that have shared a single version 4 realm, FOO.COM. The version 4 *krb.realms* file looked like:

```
      foo.com         FOO.COM
```

```
.foo.com  FOO.COM
.bar.com  FOO.COM
```

A *krb5.conf* file that covers this case might look like:

```
[libdefaults]
        v4_instance_resolve = yes
[realms]
        FOO.COM = {
                kdc = kerberos.foo.com
                v4_instance_convert = {
                        foo = foo.com
                }
                v4_domains = foo.com
        }
```

With this setup and the following host table:

```
foo.com
a-host.foo.com
b-host.bar.com
```

the following conversions will be made:

```
rcmd.a-host     -> host/a-host.foo.com
ftp.b-host -> ftp/b-host.bar.com
pop.foo         -> pop/foo.com
ftp.other  -> ftp/other.foo.com
other.a-host    -> other/a-host
```

The first three are what you expect. If you remove the "v4_domains", the fourth entry will result in an error (since the host "other" can’t be found). Even if "a-host" is a valid host name, the last entry will not be converted, since the "other" name is not known to represent a host-type principal. If you turn off "v4_instance_resolve" the second example will result in "ftp/b-host.foo.com" (because of the default domain). And all of this is of course only valid if you have working name resolving.

SEE ALSO

krb5_build_principal(3), krb5_free_principal(3), krb5_parse_name(3), krb5_sname_to_principal(3), krb5_unparse_name(3), krb5.conf(5)