NAME

krb5.conf − Kerberos configuration file

DESCRIPTION

krb5.conf contains configuration information needed by the Kerberos V5 library. This includes information describing the default Kerberos realm, and the location of the Kerberos key distribution centers for known realms.

The *krb5.conf* file uses an INI-style format. Sections are delimited by square braces; within each section, there are relations where tags can be assigned to have specific values. Tags can also contain a subsection, which contains further relations or subsections. A tag can be assigned to multiple values. Here is an example of the INI-style format used by *krb5.conf*:

```
[section1]
        tag1 = value_a
        tag1 = value_b
        tag2 = value_c

[section 2]
        tag3 = {
                subtag1 = subtag_value_a
                subtag1 = subtag_value_b
                subtag2 = subtag_value_c
        }
        tag4 = {
                subtag1 = subtag_value_d
                subtag2 = subtag_value_e
        }
```

krb5.conf can include other files using the directives "include FILENAME" or "includedir DIRNAME", which must occur at the beginning of a line. FILENAME or DIRNAME should be an absolute path. The named file or directory must exist and be readable. Including a directory includes all files within the directory whose names consist solely of alphanumeric characters, dashes, or underscores. Included profile files are syntactically independent of their parents, so each included file must begin with a section header.

krb5.conf can cause configuration to be obtained from a loadable profile module by placing the directive "module MODULEPATH:RESIDUAL" at the beginning of a line before any section headers. MODULEPATH may be relative to the library path of the krb5 installation, or it may be an absolute path. RESIDUAL is provided to the module at initialization time. If *krb5.conf* uses a module directive, *kdc.conf* should also use one if it exists.

The following sections are currently used in the *krb5.conf* file:

[libdefaults]
Contains various default values used by the Kerberos V5 library.

[login] Contains default values used by the Kerberos V5 login program, *login.krb5*(8).

[appdefaults]
Contains default values that can be used by Kerberos V5 applications.

[realms]
Contains subsections keyed by Kerberos realm names which describe where to find the Kerberos servers for a particular realm, and other realm-specific information.

[domain_realm]
Contains relations which map subdomains and domain names to Kerberos realm names. This is used by programs to determine what realm a host should be in, given its fully qualified domain name.

[logging]
Contains relations which determine how Kerberos entities are to perform their logging.

[capaths]
Contains the authentication paths used with non-hierarchical cross-realm. Entries in the section are used by the client to determine the intermediate realms which may be used in cross-realm authentication. It is also used by the end-service when checking the transited field for trusted intermediate realms.

[dbdefaults]
Contains default values for database specific parameters.

[dbmodules]
Contains database specific parameters used by the database library.

[plugins]
Contains plugin module registration and filtering parameters.

Each of these sections will be covered in more details in the following sections.

LIBDEFAULTS SECTION

The following relations are defined in the [libdefaults] section:

default_keytab_name
This relation specifies the default keytab name to be used by application severs such as telnetd and rlogind. The default is "/etc/krb5.keytab". This formerly defaulted to "/etc/v5srvtab", but was changed to the current value.

default_realm
This relation identifies the default realm to be used in a client host's Kerberos activity.

default_tgs_enctypes
This relation identifies the supported list of session key encryption types that should be returned by the KDC. The list may be delimited with commas or whitespace.

default_tkt_enctypes
This relation identifies the supported list of session key encryption types that should be requested by the client, in the same format.

permitted_enctypes
This relation identifies the permitted list of session key encryption types.

allow_weak_crypto
: If this is set to 0 (for false), then weak encryption types will be filtered out of the previous three lists. The default value for this tag is false, which may cause authentication failures in existing Kerberos infrastructures that do not support strong crypto. Users in affected environments should set this tag to true until their infrastructure adopts stronger ciphers.

clockskew
: This relation sets the maximum allowable amount of clockskew in seconds that the library will tolerate before assuming that a Kerberos message is invalid. The default value is 300 seconds, or five minutes.

ignore_acceptor_hostname
: When accepting GSSAPI or krb5 security contexts for host-based service principals, ignore any hostname passed by the calling application and allow any service principal present in the keytab which matches the service name and realm name (if given). This option can improve the administrative flexibility of server applications on multi-homed hosts, but can compromise the security of virtual hosting environments. The default value is false.

k5login_authoritative
: If the value of this relation is true (the default), principals must be listed in a local user's k5login file to be granted login access, if a k5login file exists. If the value of this relation is false, a principal may still be granted login access through other mechanisms even if a k5login file exists but does not list the principal.

k5login_directory
: If set, the library will look for a local user's k5login file within the named directory, with a filename corresponding to the local username. If not set, the library will look for k5login files in the user's home directory, with the filename .k5login. For security reasons, k5login files must be owned by the local user or by root.

kdc_timesync
: If the value of this relation is non-zero (the default), the library will compute the difference between the system clock and the time returned by the KDC and in order to correct for an inaccurate system clock. This corrective factor is only used by the Kerberos library.

kdc_req_checksum_type
: For compatibility with DCE security servers which do not support the default CKSUMTYPE_RSA_MD5 used by this version of Kerberos. Use a value of 2 to use the CKSUMTYPE_RSA_MD4 instead. This applies to DCE 1.1 and earlier. This value is only used for DES keys; other keys use the preferred checksum type for those keys.

ap_req_checksum_type
: If set this variable controls what ap-req checksum will be used in authenticators. This variable should be unset so the appropriate checksum for the encryption key in use will be used. This can be set if backward compatibility requires a specific checksum type.

safe_checksum_type
: This allows you to set the preferred keyed-checksum type for use in KRB_SAFE messages. The default value for this type is CKSUMTYPE_RSA_MD5_DES. For compatibility with applications linked against DCE version 1.1 or earlier Kerberos libraries, use a value of 3 to use the CKSUMTYPE_RSA_MD4_DES instead. This field is ignored when its value is incompatible

with the session key type.

preferred_preauth_types
: This allows you to set the preferred preauthentication types which the client will attempt before others which may be advertised by a KDC. The default value for this setting is "17, 16, 15, 14", which forces libkrb5 to attempt to use PKINIT if it is supported.

ccache_type
: User this parameter on systems which are DCE clients, to specify the type of cache to be created by kinit, or when forwarded tickets are received. DCE and Kerberos can share the cache, but some versions of DCE do not support the default cache as created by this version of Kerberos. Use a value of 1 on DCE 1.0.3a systems, and a value of 2 on DCE 1.1 systems.

dns_lookup_kdc
: Indicate whether DNS SRV records should be used to locate the KDCs and other servers for a realm, if they are not listed in the information for the realm. The default is to use these records.

dns_lookup_realm
: Indicate whether DNS TXT records should be used to determine the Kerberos realm of a host. The default is not to use these records.

dns_fallback
: General flag controlling the use of DNS for Kerberos information. If both of the preceding options are specified, this option has no effect.

realm_try_domains
: Indicate whether a host's domain components should be used to determine the Kerberos realm of the host. The value of this variable is an integer: −1 means not to search, 0 means to try the host's domain itself, 1 means to also try the domain's immediate parent, and so forth. The library's usual mechanism for locating Kerberos realms is used to determine whether a domain is a valid realm--which may involve consulting DNS if dns_lookup_kdc is set. The default is not to search domain components.

extra_addresses
: This allows a computer to use multiple local addresses, in order to allow Kerberos to work in a network that uses NATs. The addresses should be in a comma-separated list.

udp_preference_limit
: When sending a message to the KDC, the library will try using TCP before UDP if the size of the message is above "udp_preference_limit". If the message is smaller than "udp_preference_limit", then UDP will be tried before TCP. Regardless of the size, both protocols will be tried if the first attempt fails.

verify_ap_req_nofail
: If this flag is set, then an attempt to get initial credentials will fail if the client machine does not have a keytab. The default for the flag is false.

ticket_lifetime
: The value of this tag is the default lifetime for initial tickets. The default value for the tag is 1 day (1d).

renew_lifetime
The value of this tag is the default renewable lifetime for initial tickets. The default value for the tag is 0.

noaddresses
Setting this flag causes the initial Kerberos ticket to be addressless. The default for the flag is true.

forwardable
If this flag is set, initial tickets by default will be forwardable. The default value for this flag is false.

proxiable
If this flag is set, initial tickets by default will be proxiable. The default value for this flag is false.

rdns If set to false, prevent the use of reverse DNS resolution when translating hostnames into service principal names. Defaults to true. Setting this flag to false is more secure, but may force users to exclusively use fully qualified domain names when authenticating to services.

plugin_base_dir
If set, determines the base directory where krb5 plugins are located. The default value is the "krb5/plugins" subdirectory of the krb5 library directory.

APPDEFAULTS SECTION

Each tag in the [appdefaults] section names a Kerberos V5 application or an option that is used by some Kerberos V5 application[s]. The four ways that you can set values for options are as follows, in decreasing order of precedence:

```
#1)
        application = {
                realm1 = {
                        option = value
                }
                realm2 = {
                        option = value
                }
        }
#2)
        application = {
                option1 = value
                option2 = value
        }
#3)
        realm = {
                option = value
        }
#4)
        option = value
```

LOGIN SECTION

The [login] section is used to configure the behavior of the Kerberos V5 login program, *login.krb5*(8). Refer to the manual entry for *login.krb5* for a description of the relations allowed in this section.

REALMS SECTION

Each tag in the [realms] section of the file names a Kerberos realm. The value of the tag is a subsection where the relations in that subsection define the properties of that particular realm. For example:

```
[realms]
        ATHENA.MIT.EDU = {
                admin_server = KERBEROS.MIT.EDU
                default_domain = MIT.EDU
                database_module = ldapconf
                v4_instance_convert = {
                        mit = mit.edu
                        lithium = lithium.lcs.mit.edu
                }
                v4_realm = LCS.MIT.EDU
        }
```

For each realm, the following tags may be specified in the realm's subsection:

kdc The value of this relation is the name of a host running a KDC for that realm. An optional port number (preceded by a colon) may be appended to the hostname. This tag should generally be used only if the realm administrator has not made the information available through DNS.

admin_server
This relation identifies the host where the administration server is running. Typically this is the Master Kerberos server.

database_module
This relation indicates the name of the configuration section under dbmodules for database specific parameters used by the loadable database library.

default_domain
This relation identifies the default domain for which hosts in this realm are assumed to be in. This is needed for translating V4 principal names (which do not contain a domain name) to V5 principal names (which do).

v4_instance_convert
This subsection allows the administrator to configure exceptions to the default_domain mapping rule. It contains V4 instances (the tag name) which should be translated to some specific hostname (the tag value) as the second component in a Kerberos V5 principal name.

v4_realm
This relation is used by the krb524 library routines when converting a V5 principal name to a V4 principal name. It is used when V4 realm name and the V5 realm are not the same, but still share the same principal names and passwords. The tag value is the Kerberos V4 realm name.

auth_to_local_names
: This subsection allows you to set explicit mappings from principal names to local user names. The tag is the mapping name, and the value is the corresponding local user name.

auth_to_local
: This tag allows you to set a general rule for mapping principal names to local user names. It will be used if there is not an explicit mapping for the principal name that is being translated. The possible values are:

    DB:<filename>
    : The principal will be looked up in the database <filename>. Support for this is not currently compiled in by default.

    RULE:<exp>
    : The local name will be formulated from <exp>.

    DEFAULT
    : The principal name will be used as the local name. If the principal has more than one component or is not in the default realm, this rule is not applicable and the conversion will fail.

DOMAIN_REALM SECTION

The [domain_realm] section provides a translation from a hostname to the Kerberos realm name for the services provided by that host.

The tag name can be a hostname, or a domain name, where domain names are indicated by a prefix of a period ('.') character. The value of the relation is the Kerberos realm name for that particular host or domain. Host names and domain names should be in lower case.

If no translation entry applies, the host's realm is considered to be the hostname's domain portion converted to upper case. For example, the following [domain_realm] section:

```
[domain_realm]
    .mit.edu = ATHENA.MIT.EDU
    mit.edu = ATHENA.MIT.EDU
    dodo.mit.edu = SMS_TEST.MIT.EDU
    .ucsc.edu = CATS.UCSC.EDU
```

maps dodo.mit.edu into the SMS_TEST.MIT.EDU realm, all other hosts in the MIT.EDU domain to the ATHENA.MIT.EDU realm, and all hosts in the UCSC.EDU domain into the CATS.UCSC.EDU realm. ucbvax.berkeley.edu would be mapped by the default rules to the BERKELEY.EDU realm, while sage.lcs.mit.edu would be mapped to the LCS.MIT.EDU realm.

LOGGING SECTION

The [logging] section indicates how a particular entity is to perform its logging. The relations specified in this section assign one or more values to the entity name.

Currently, the following entities are used:

kdc These entries specify how the KDC is to perform its logging.

admin_server
: These entries specify how the administrative server is to perform its logging.

default These entries specify how to perform logging in the absence of explicit specifications otherwise.

Values are of the following forms:

FILE=<filename>

FILE:<filename>
: This value causes the entity's logging messages to go to the specified file. If the = form is used, then the file is overwritten. Otherwise, the file is appended to.

STDERR
: This value causes the entity's logging messages to go to its standard error stream.

CONSOLE
: This value causes the entity's logging messages to go to the console, if the system supports it.

DEVICE=<devicename>
: This causes the entity's logging messages to go to the specified device.

SYSLOG[:<severity>[:<facility>]]
: This causes the entity's logging messages to go to the system log.

  The **severity** argument specifies the default severity of system log messages. This may be any of the following severities supported by the *syslog(3)* call minus the LOG_ prefix: LOG_EMERG, LOG_ALERT, LOG_CRIT, LOG_ERR, LOG_WARNING, LOG_NOTICE, LOG_INFO, and LOG_DEBUG. For example, to specify LOG_CRIT severity, one would use CRIT for **severity.**

  The **facility** argument specifies the facility under which the messages are logged. This may be any of the following facilities supported by the *syslog(3)* call minus the LOG_ prefix: LOG_KERN, LOG_USER, LOG_MAIL, LOG_DAEMON, LOG_AUTH, LOG_LPR, LOG_NEWS, LOG_UUCP, LOG_CRON, and LOG_LOCAL0 through LOG_LOCAL7.

  If no **severity** is specified, the default is ERR, and if no **facility** is specified, the default is AUTH.

In the following example, the logging messages from the KDC will go to the console and to the system log under the facility LOG_DAEMON with default severity of LOG_INFO; and the logging messages from the administrative server will be appended to the file /var/adm/kadmin.log and sent to the device /dev/tty04.

```
[logging]
        kdc = CONSOLE
        kdc = SYSLOG:INFO:DAEMON
        admin_server = FILE:/var/adm/kadmin.log
        admin_server = DEVICE=/dev/tty04
```

CAPATHS SECTION

Cross-realm authentication is typically organized hierarchically. This hierarchy is based on the name of the realm, which thus imposes restrictions on the choice of realm names, and on who may participate in a cross-realm authentication. A non hierarchical organization may be used, but requires a database to construct the authentication paths between the realms. This section defines that database.

A client will use this section to find the authentication path between its realm and the realm of the server. The server will use this section to verify the authentication path used be the client, by checking the transited field of the received ticket.

There is a tag name for each participating realm, and each tag has subtags for each of the realms. The value of the subtags is an intermediate realm which may participate in the cross-realm authentication. The subtags may be repeated if there is more then one intermediate realm. A value of "." means that the two realms share keys directly, and no intermediate realms should be allowed to participate.

There are n**2 possible entries in this table, but only those entries which will be needed on the client or the server need to be present. The client needs a tag for its local realm, with subtags for all the realms of servers it will need to authenticate with. A server needs a tag for each realm of the clients it will serve.

For example, ANL.GOV, PNL.GOV, and NERSC.GOV all wish to use the ES.NET realm as an intermediate realm. ANL has a sub realm of TEST.ANL.GOV which will authenticate with NERSC.GOV but not PNL.GOV. The [capath] section for ANL.GOV systems would look like this:

```
[capaths]
        ANL.GOV = {
                TEST.ANL.GOV = .
                PNL.GOV = ES.NET
                NERSC.GOV = ES.NET
                ES.NET = .
        }
        TEST.ANL.GOV = {
                ANL.GOV = .
        }
        PNL.GOV = {
                ANL.GOV = ES.NET
        }
        NERSC.GOV = {
                ANL.GOV = ES.NET
        }
        ES.NET = {
                ANL.GOV = .
        }
```

The [capath] section of the configuration file used on NERSC.GOV systems would look like this:

```
[capaths]
        NERSC.GOV = {
                ANL.GOV = ES.NET
                TEST.ANL.GOV = ES.NET
                TEST.ANL.GOV = ANL.GOV
                PNL.GOV = ES.NET
                ES.NET = .
        }
        ANL.GOV = {
                NERSC.GOV = ES.NET
        }
        PNL.GOV = {
                NERSC.GOV = ES.NET
        }
        ES.NET = {
                NERSC.GOV = .
        }
        TEST.ANL.GOV = {
                NERSC.GOV = ANL.GOV
                NERSC.GOV = ES.NET
        }
```

In the above examples, the ordering is not important, except when the same subtag name is used more then once. The client will use this to determine the path. (It is not important to the server, since the transited field is not sorted.)

If this section is not present, or if the client or server cannot find a client/server path, then normal hierarchical organization is assumed.

This feature is not currently supported by DCE. DCE security servers can be used with Kerberized clients

and servers, but versions prior to DCE 1.1 did not fill in the transited field, and should be used with caution.

DATABASE DEFAULT SECTION

The [dbdefaults] section indicates default values for the database specific parameters. It can also specify the configuration section under dbmodules for database specific parameters used by the loadable database library.

The following tags are used in this section:

database_module
: This relation indicates the name of the configuration section under dbmodules for database specific parameters used by the loadable database library.

ldap_kerberos_container_dn
: This LDAP specific tag indicates the DN of the container object where the realm objects will be located. This value is used if no object DN is mentioned in the configuration section under dbmodules.

ldap_kdc_dn
: This LDAP specific tag indicates the default bind DN for the KDC server. The KDC server does a login to the directory as this object. This value is used if no object DN is mentioned in the configuration section under dbmodules.

ldap_kadmind_dn
: This LDAP specific tag indicates the default bind DN for the Administration server. The Administration server does a login to the directory as this object. This value is used if no object DN is mentioned in the configuration section under dbmodules.

ldap_service_password_file
: This LDAP specific tag indicates the file containing the stashed passwords for the objects used for starting the Kerberos servers. This value is used if no service password file is mentioned in the configuration section under dbmodules.

ldap_servers
: This LDAP specific tag indicates the list of LDAP servers. The list of LDAP servers is whitespace-separated. The LDAP server is specified by a LDAP URI. This value is used if no LDAP servers are mentioned in the configuration section under dbmodules.

ldap_conns_per_server
: This LDAP specific tag indicates the number of connections to be maintained per LDAP server. This value is used if the number of connections per LDAP server are not mentioned in the configuration section under dbmodules. The default value is 5.

DATABASE MODULE SECTION

Each tag in the [dbmodules] section of the file names a configuration section for database specific parameters that can be referred to by a realm. The value of the tag is a subsection where the relations in that subsection define the database specific parameters.

For each section, the following tags may be specified in the subsection:

database_name
: This DB2-specific tag indicates the location of the database in the filesystem.

db_library
: This tag indicates the name of the loadable database library. The value should be db2 for db2 database and kldap for LDAP database.

disable_last_success
: If set to true, suppresses KDC updates to the "Last successful authentication" field of principal entries requiring preauthentication. Setting this flag may improve performance. (Principal entries which do not require preauthentication never update the "Last successful authentication" field.)

disable_lockout
: If set to true, suppresses KDC updates to the "Last failed authentication" and "Failed password attempts" fields of principal entries requiring preauthentication. Setting this flag may improve performance, but also disables account lockout.

ldap_kerberos_container_dn
: This LDAP specific tag indicates the DN of the container object where the realm objects will be located.

ldap_kdc_dn
: This LDAP specific tag indicates the bind DN for the KDC server. The KDC does a login to the directory as this object.

ldap_kadmind_dn
: This LDAP specific tag indicates the bind DN for the Administration server. The Administration server does a login to the directory as this object.

ldap_service_password_file
: This LDAP specific tag indicates the file containing the stashed passwords for the objects used for starting the Kerberos servers.

ldap_servers
: This LDAP specific tag indicates the list of LDAP servers. The list of LDAP servers is whitespace-separated. The LDAP server is specified by a LDAP URI.

ldap_conns_per_server
: This LDAP specific tag indicates the number of connections to be maintained per LDAP server.

PLUGINS SECTION

Tags in the [plugins] section can be used to register dynamic plugin modules and to turn modules on and off. Not every krb5 pluggable interface uses the [plugins] section; the ones that do are documented here.

Each pluggable interface corresponds to a subsection of [plugins]. All subsections support the same tags:

module This tag may have multiple values. Each value is a string of the form "modulename:pathname", which causes the shared object located at pathname to be registered as a dynamic module named modulename for the pluggable interface. If pathname is not an absolute path, it will be treated as relative to the plugin base directory.

enable_only
: This tag may have multiple values. If there are values for this tag, then only the named modules will be enabled for the pluggable interface.

disable
: This tag may have multiple values. If there are values for this tag, then the named modules will be disabled for the pluggable interface.

The following subsections are currently supported within the [plugins] section:

pwqual interface

The pwqual subsection controls modules for the password quality interface, which is used to reject weak passwords when passwords are changed. In addition to any registered dynamic modules, the following built-in modules exist (and may be disabled with the disable tag):

dict
: Checks against the realm dictionary file

empty
: Rejects empty passwords

hesiod
: Checks against user information stored in Hesiod (only if Kerberos was built with Hesiod support)

princ
: Checks against components of the principal name

kadm5_hook interface

The kadm5_hook interface provides plugins with information on principal creation, modification, password changes and deletion. This interface can be used to write a plugin to synchronize MIT Kerberos with another database such as Active Directory. No plugins are built in for this interface.

clpreauth and kdcpreauth interfaces

The clpreauth and kdcpreauth interfaces allow plugin modules to provide client and KDC preauthentication mechanisms. The following built-in modules exist for these interfaces:

pkinit
: This module implements the PKINIT preauthentication mechanism.

encrypted_challenge
: This module implements the encrypted challenge FAST factor.

encrypted_timestamp
: This module implements the encrypted timestamp mechanism.

FILES

/etc/krb5.conf

SEE ALSO

syslog(3)