Release 1.0, 1/13/98

Network Computer Login

Detailed Description

1. Overview

The Network Computer Login (NCL) facility consists of a login client (NCL client) running on a network computer and a login server (NCL server) running on a server used to support network computers. Together the login client and server authenticate a user and customize the client for that user. The login client and login server communicate by means of a remote authentication protocol (RAP). The remainder of this document describes the RAP protocol, the behavior and design of the client, and the behavior and design of the server.

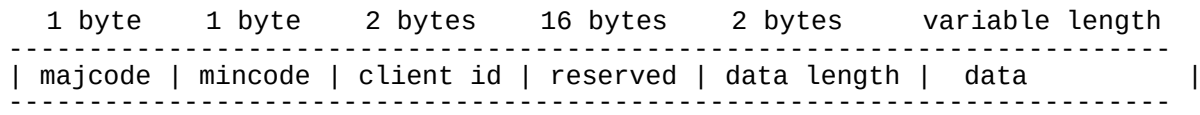
2. Remote Authentication Protocol (RAP)

An NCL client will initiate a TCP connection with an NCL server, using a well-known port for remote authentication protocol. The well-known TCP port for a RAP server is 256. A RAP session consists of the login client sending a single request packet to an authentication server, followed by the server sending one or more reply packets to the client. The end of a RAP session is marked by termination of the underlying TCP connection. Note that we refer to "packets" that are exchanged between the client and server for simplicity. Since the underlying TCP connection is stream-oriented, these packets may not correspond to the IP packets actually transmitted. This is merely a high-level view of the data exchange.

In this document, we use 8-bit bytes corresponding to octets of data transmitted over the network. All multi-byte integer data shall be transmitted in network byte order (i.e., high-order byte first). All string data is case-sensitive, unless otherwise specified. A particular application receiving string data may ignore case, but string data transmission must be presumed case-sensitive.

2.1. Requests:

A request packet has the structure shown in the figure below.



The majcode is a 1 byte unsigned integer value representing the request class. The mincode is a 1 byte unsigned integer value representing a particular request operation with the class specified by the major code.

In the initial implementation, the only defined major code is the value 1, indicating an authentication request (AUTH). The only defined minor code for AUTH requests is the value 1, indicating a simple authentication request for POSIX identification data (AUTH_SIMPLE).

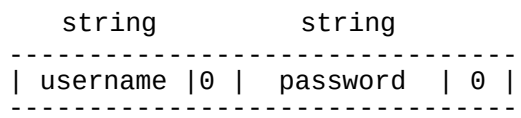
The client id is a 2 byte unsigned integer value representing a client workstation type. This value may be used by the server for server-specific purposes. In the initial implementation, the only defined client id is the value 1.

The data length is a 2 byte unsigned integer value indicating the length of the data contained in the data field that follows.

The content of the data field is dependent upon the major and minor codes specified.

2.1.1. AUTH_SIMPLE requests:

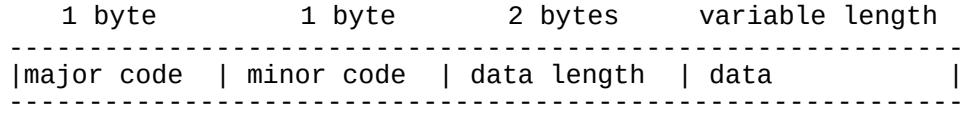
The data for an AUTH_SIMPLE authentication request consists of a null terminated username string, followed by a null-terminated password string, as shown in the figure below.



Both strings must be represented using ISO8859-1. The maximum total length of an AUTH_SIMPLE data field is 256 bytes, including null terminators.

2.2. Reply directives:

A reply packet has the structure of the figure below.



The major code is a 1 byte unsigned integer value representing a reply directive.

In the initial implementation, the following major codes are defined:

major code	directive
1	termination directive (DONE)
2	error directive (ERROR)
3	identification directive (ID)
4	file system mount directive (MOUNT)
5	environment variable directive (ENV)
6	information display directive (INFO)

The minor code is a 1 byte unsigned integer value representing the reply subtype. Defined minor codes are documented below in the sections documenting defined reply packet types.

The data length is a 2 byte unsigned integer value specifying the length of the data field that follows.

The content of the data field is dependent upon the directive type specified by the major and minor codes.

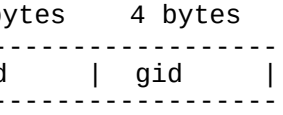
2.2.1. ID directives:

The ID directive provides the client with identification information for the authenticated user.

In the initial implementation, the only defined minor code for ID directives is the value 1, indicating standard POSIX identification data (ID_POSIX).

2.2.1.1. ID_POSIX data:

The data field of an ID_POSIX directive consists of a uid and gid as shown in the figure below.

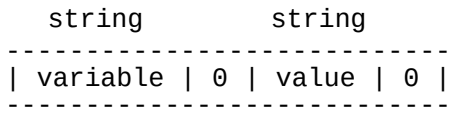


2.2.2. ENV directives:

An ENV reply directs the client to modify a local environment variable. This may be used to pass a variety of system-specific information, such as the path to a user's home directory or where preferences are stored. The client may or may not actually modify a local environment variable, in response.

In the initial implementation, the only defined minor code for ENV directive is the value 1, indicating an environment variable definition directive (ENV_SET).

The data for all ENV reply packets, regardless of minor code, shall consist of a null terminated variable name string, followed by a null-terminated value string, as shown in the figure below.



Both strings must be represented using ISO8859-1. The login server may refer to the value of a previously set environment variable, in a subsequent ENV directive. This is done by prepending a "$" character to the variable name. (For example, if HOME was previously defined via an ENV directive, "$HOME" may be used to represent the value of that variable in a subsequent directive.)

2.2.2. MOUNT directives:

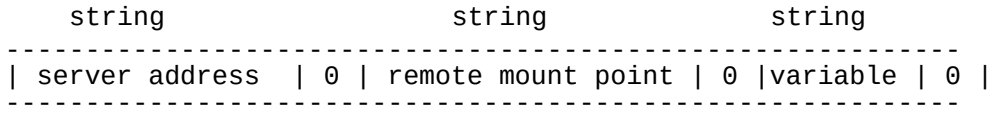
The MOUNT reply directs the client to mount a specified file system.

In the initial implementation, the following minor codes for MOUNT directives are defined:

minor code	Mount type
1	NFS-accessed file system (MOUNT_NFS)
2	TFTP-accessed file system (MOUNT_TFTP)

2.2.3.1. MOUNT_NFS data:

The data field of a MOUNT_NFS directive consists of a null-terminated server address, a null-terminated remote mount point, and a null terminated environment variable name, as shown in the figure below:



The server address may be represented as either a domain name or as a dotted-decimal address (represented as a string). A null string may be used to specify use of the login server as the file server.

The remote mount point must consist of an absolute path to the mountpoint. The pathname

must use a forward slash ("/") as a directory delimiter.

The variable field consists of an environment variable to be associated with the mounted file system. The login client must associate the local mount point for the file system with the specified variable name, as with ENV directives. A null string may be used to specify that no environment variable is to be associated with the local mount point.

All three strings must be represented using ISO8859-1.

2.2.3.2. MOUNT_TFTP data:

The data field of a MOUNT_TFTP directive is identical to that of the MOUNT_NFS directive.

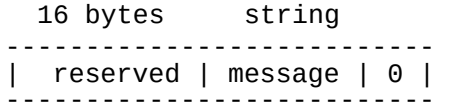
2.2.4. INFO directives:

An INFO reply directs the client to display information to the user. This may be used to convey important information to the user, such as account status or a system's message of the day.

In the initial implementation, the only defined minor code for INFO directives is the value 1, indicating a string display directive (INFO_STRING).

2.2.4.1. INFO_STRING data:

The data field of an INFO_STRING directive consists of a reserved field and a null-terminated message string, as shown in the figure below:



The reserved field is a 16 byte field, and should be represented as a null string in ISO8859-1. The message string must be null-terminated. End of line (EOL) must be represented as a carriage return followed by a line feed (CR/LF).

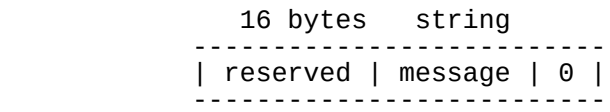
2.2.5. ERROR directives:

An ERROR directive indicates an unrecoverable error, and signals termination (on failure) of the RAP session. The client should display an error message to the user and terminate the login process upon receiving an ERROR directive.

In the initial implementation, the following minor codes for ERROR directives are defined:

minor code	error type
1	system-specific login error (ERR_SYS)
2	unsupported major code (ERR_MAJCODE)
3	unsupported minor code (ERR_MINCODE)
4	unsupported client (ERR_CLIENT)
5	malformed request (ERR_REQUEST)
6	incorrect login (ERR_LOGIN)
7	unknown user (ERR_USER)
8	incorrect password (ERR_PASSWD)

All ERROR directives use the data format shown in the following figure, regardless of the minor code.



The format of ERROR directive data is the same as that described in Section 2.2.4.1 for INFO_STRING directive data.

For standard errors (i.e., those that are not system specific, minor codes 2 through 8), the server may optionally return a null message string. For standard errors, the client may also opt to display a standard, locally-defined error message in place of any transmitted error message.

2.2.6. DONE directives

A DONE directive signals the successful completion of a RAP session.

A DONE directive never contains any data. Correspondingly, the data length field must always be 0. The minor code field is reserved, and should be set to 0.

3. NCL server

The NCL server will act as a RAP authentication server. The NCL server will accept and then process incoming TCP connections on the RAP well-known port.

3.1. Successful-case behavior

For a successful authentication request, the NCL server will do the following:

1) Receive an AUTH_SIMPLE request.

2) Authenticate the username and password combination, retrieving POSIX identification data, information necessary to mount the user's home directory and the user's preferences directory, as well as a user-specific account status string, if any.

3) Send the POSIX identification data to the client as an ID_POSIX directive

4) If normalization of the username is required, send an ENV directive to the client for the USER environment variable.

5) Send one or more MOUNT directives to the client, containing mount data to account for user home directory and possible seperate preference directory.

6) Send the user-specific account status string, if any, to the client as an INFO_STRING directive.

7) Send a DONE directive to the client, terminating the RAP session.

3.2. Error-case behavior

In case an authentication or other error is detected, the NCL server will log the error and send the client an appropriate ERROR directive.

4. Client

The client will initiate a RAP session with an NCL server, in order to authenticate a user and obtain information necessary to customize the NC environment for that user.

4.1. Basic client behavior

During a successful login process, the client will do the following:
1) Lock the screen with a username/password entry screen. In addition to the username and password, this screen may allow the user to optionally specify a login server.

2) Upon receiving user input, the client will establish a TCP connection with the specified NCL server, at the RAP well-known port. The client will send the server an AUTH_SIMPLE authentication request including the specified username and password. The client may

provide a "Cancel" button to allow the user to terminate an authentication attempt.

3) Enter a loop to process all additional reply packets received from the NCL server.

4) Unlock the screen.

4.2. Reply packet processing

Reply directives from the NCL server will be processed as follows:

directive	response
ID_POSIX	Set the uid, and gid for the login process.
MOUNT_NFS	Set up NFS mount information.
MOUNT_TFTP	Set up TFTP mount information.
SET_ENV	Set the specified environment variable as directed.
INFO_STRING	Display the specified message string to the user.
ERROR	Display the specified message string to the user and terminate the login process.

4.3. Termination on failure

Termination of login on failure shall result in the client presenting a new login screen.