NAME

xauth – X authority file utility

SYNOPSIS

xauth [**-f** *authfile*] [**-v**] [**-q**] [**-i**] [**-b**] [*command arg...*]

DESCRIPTION

The **xauth** program is used to edit and display the authorization information used in connecting to the X server. This program is usually to extract authorization records from one machine and merge them in on another (as is the case when using remote logins or to grant access to other users). Commands (described below) may be entered interactively, on the **xauth** command line, or in scripts. Note that this program does **not** contact the X server.

OPTIONS

The following options may be used with **xauth.** They may be given individually (e.g. –**q** –**i**) or may combined (e.g. **-qi**):

–**f** *authfile*
: This option specifies the name of the authority file to use. By default, xauth **will use the file specified by the XAUTHORITY environment variable or** *.Xauthority* **in the user's home directory.**

–**q**
: This option indicates that **xauth** should operate quietly and not print unsolicited status messages. This is the default if an **xauth** command is is given on the command line or if the standard output is not directed to a terminal.

–**v**
: This option indicates that **xauth** should operate verbosely and print status messages indicating the results of various operations (e.g. how many records have been read in or written out). This is the default if **xauth** is reading commands from its standard input and its standard output is directed to a terminal.

–**i**
: This option indicates that **xauth** should ignore any authority file locks. Normally, **xauth** will refuse to read or edit any authority files that have been locked by other programs (usually **xdm**(1) or another **xauth**).

–**b**
: This option indicates that **xauth** should attempt to break any authority file locks before proceeding and should only be used to clean up stale locks.

COMMANDS

The following commands may be used to manipulate authority files:

add *displayname protocolname hexkey*
: An authorization entry for the indicated display using the given protocol and key data is added to the authorization file. The data is specified as an even-lengthed string of hexadecimal digits, each pair representing one octet. The first digit of each pair gives the most significant 4 bits of the octet and the second digit of the pairgives the least significant 4 bits. For example, a 32 character hexkey would represent a 128-bit value. A protocol name consisting of just a single period is treated as an abbreviation for *MIT-MAGIC-COOKIE-1*.

add *displayname protocolname 'string'*
: An authorization entry for the indicated display using the given protocol and the protocolname is specified by 'string'. If the authorization name is UN_DE-1, then the data field contains an ASCII netname.

[n]extract *filename displayname...*
Authorization entries for each of the specified displays are written to the indicated file. If the *nextract* command is used, the entries are written in a numeric format suitable for non-binary transmission (such as secure electronic mail). The extracted entries can be read back in using the *merge* and *nmerge* commands. If the the filename consists of just a single dash, the entries will be written to the standard output.

[n]list [*displayname...]*
Authorization entries for each of the specified displays (or all if no displays are named) are printed on the standard output. If the *nlist* command is used, entries will be shown in the numeric format used by the *nextract* command; otherwise, they are shown in a textual format. Key data is always displayed in the hexadecimal format given in the description of the *add* command.

[n]merge [*filename...]*
Authorization entries are read from the specified files and are merged into the authorization database, superceding any matching existing entries. If the *nmerge* command is used, the numeric format given in the description of the *extract* command is used. If a filename consists of just a single dash, the standard input will be read if it hasn't been read before.

remove *displayname...*
Authorization entries matching the specified displays are removed from the authority file.

source *filename*
The specified file is treated as a script containing **xauth** commands to execute. Blank lines and lines beginning with a sharp sign (#) are ignored. A single dash may be used to indicate the standard input, if it hasn't already been read.

info Information describing the authorization file, whether or not any changes have been made, and from where **xauth** commands are being read is printed on the standard output.

exit If any modifications have been made, the authority file is written out (if allowed), and the program exits. An end of file is treated as an implicit *exit* command.

quit The program exits, ignoring any modifications. This may also be accomplished by pressing the interrupt character.

help [*string***]**
A description of all commands that begin with the given string (or all commands if no string is given) is printed on the standard output.

? A short list of the valid commands is printed on the standard output.

DISPLAY NAMES

Display names for the *add*, *[n]extract*, *[n]list*, *[n]merge*, and *remove* commands use the same format as the DISPLAY environment variable and the common *-display* command line argument. Display-specific information (such as the screen number) is unnecessary and will be ignored. Same-machine connections (such as local-host sockets, shared memory, and the Internet Protocol hostname *localhost*) are referred to as *hostname*/unix:*displaynumber* so that local entries for different machines may be stored in one authority file.

EXAMPLE

The most common use for **xauth** is to extract the entry for the current display, copy it to another machine, and merge it into the user's authority file on the remote machine:

```
example%  xauth extract - $DISPLAY | rsh other xauth merge -
```

ENVIRONMENT

This **xauth** program uses the following environment variables:

XAUTHORITY
: to get the name of the authority file to use if the *–f* option isn't used. If this variable is not set, **xauth** will use *.Xauthority* in the user's home directory.

HOME
: to get the user's home directory if XAUTHORITY isn't defined.

BUGS

Users that have unsecure networks should take care to use encrypted file transfer mechanisms to copy authorization entries between machines. Similarly, the *MIT-MAGIC-COOKIE-1* protocol is not very useful in unsecure environments. Sites that are interested in additional security may need to use encrypted authorization mechanisms such as Kerberos.

Spaces are currently not allowed in the protocol name. Quoting could be added for the truly perverse.

COPYRIGHT

AUTHOR

Jim Fulton, MIT X Consortium