NAME

gdm, gdm-binary, gdmchooser, gdmgreeter, gdmlogin − GNOME Display Manager

SYNOPSIS

gdm | **gdm-binary** [**--monte-carlo-sqrt2**] [**--no-console**] [**-nodaemon**] [**--preserve-ld-vars**] [**--version**] [**--wait-for-go**]

gdmlogin | **gdmgreeter** [*gnome-std-options*]

gdmchooser [**-clientaddress=***address*] [**-connectionType=***type*] [**-xdmaddress=***socket*] [*gnome-std-options*]

DESCRIPTION

GDM is the GNOME Display Manager, a program used for login session management. When no user is logged in on the console, GDM displays a graphical user interface that enables the user to enter their username and password. GDM supports XDMCP and supports flexible or on-demand servers via the **gdmflexiserver** command.

gdm is a wrapper script that launches **gdm-binary** and passes along any options. Before launching **gdm-binary**, the **gdm** wrapper script sources the **/etc/profile** file to set the standard system environment variables. To support internationalization, **gdm** also sets the LC_MESSAGES environment variable to LANG if neither LC_MESSAGES nor LC_ALL is set.

On startup, the GDM daemon parses its config file **gdm.conf**. For each local display, **gdm-binary** forks an Xserver and a slave process. The main **gdm-binary** process then listens to XDMCP requests from remote displays, if so configured, and monitors the local display sessions. The main daemon process also allows new local Xservers to start on demand using the **gdmflexiserver** command.

The GDM slave process opens the display and starts either the Graphical Greeter or the Standard Greeter. This choice is set by the "Greeter" parameter in **gdm.conf** for console login and the "RemoteGreeter" parameter for XDMCP logins. The parameter should be set to "gdmgreeter" to use the Graphical Greeter or "gdmlogin" to use the Standard Greeter. The Standard Greeter is lower-bandwidth, which tends to be more appropriate for remote logins. The GDM daemon communicates asynchronously with the slave process through a pipe.

From either the Graphical Greeter or the Standard Greeter, it is possible to launch the Chooser program **gdmchooser** to start remote XDMCP login sessions.

Although disabled by default, it is also possible to launch the Setup program **gdmsetup** to edit the configuration choices in **gdm.conf**. The root password must be entered to launch the Setup program. The ability to launch the Setup program is disabled by default as **gdmsetup** runs with root permissions and allows GDM to be configured in ways that affect security.

GDM relies on PAM, Pluggable Authentication Modules, for password authentication, but supports regular crypt() and shadow passwords on legacy systems. On Solaris, GDM uses **logindevperm** to set proper device permissions for the user on login.

All operations on user files are done with the effective user id of the user. If the sanity check fails on the user's **.Xauthority** file, a fallback cookie is created in **/tmp**.

OPTIONS

The following options are supported by **gdm** and **gdm-binary**:

--monte-carlo-sqrt2

--no-console Tell the daemon that it should not run anything on the console. This means that none of the local servers from the [servers] section of the **gdm.conf** configuration file are run, and the console is not used to communicate errors to the user. An empty [servers] section automatically implies this option.

-nodaemon If this option is specified, GDM does not fork into the background when run. You can use a single dash with this option to preserve compatibility with XDM.

--preserve-ld-vars When clearing the environment internally, preserve all variables starting with LD_. This is mostly for debugging purposes.

--version Print the version of the GDM daemon.

--wait-for-go If started with this option, GDM initiates, but only starts the first local display and then waits for a GO message in the fifo protocol. No greeter is shown until the GO message is sent. Also, flexiserver requests are denied and XDMCP is not started until GO is given. This is useful for initialization scripts that wish to start X early, but where you do not yet want the user to start logging in: the script sends the GO to the fifo when ready and GDM then continues.

The following options are supported by **gdmlogin** and **gdmgreeter**:

gnome-std-options Standard options available for use with most GNOME applications. See **gnome-std-options**(5) for more information.

The following options are supported by **gdmchooser**:

-clientaddress=*address* Client address to return in response to xdm. This option is for running **gdmchooser** with xdm, and is not used within GDM.

-connectionType=*type* Connection type to return in response to xdm. This option is for running **gdmchooser** with xdm, and is not used within GDM.

-xdmaddress=*socket* Socket for xdm communication.

gnome-std-options Standard options available for use with most GNOME applications. See **gnome-std-options**(5) for more information.

EXTENDED DESCRIPTION

Standard Greeter

The Standard Greeter is the default graphical user interface that is presented to the user. The greeter contains a menu at the top, an optional face browser, an optional logo, and a text entry field. The Standard Greeter corresponds to the executable **gdmlogin**.

The text entry field is used to enter logins, passwords, passphrases, and so on. The field is controlled by the underlying daemon and is basically stateless. The daemon controls the greeter through a simple protocol where the daemon can ask the greeter for a text string with echo turned on or off. Similarly, the daemon can change the label above the text entry field to correspond to the value that the authentication system wants the user to enter.

The menu bar in the top of the greeter enables the user to select the requested session type or desktop environment, change the GTK+ theme (if enabled), select an appropriate locale or language, and optionally shutdown, reboot, or suspend the machine, configure GDM (if the user knows the root password), or start an XDMCP chooser.

Optionally, the greeter can provide a face browser that contains icons for all of the users on a system. The icons can be installed globally by the system administrator, or in the user home directories. If installed globally, the icons should be in the *share*/faces directory (though this can be configured with the GlobalFaceDir configuration option) and the filename should be the name of the user, optionally with ".png" appended.

Users can place their icons in a file called **~/.face**, and can use **gdmphotosetup** to graphically configure this. Face icons placed in the global face directory must be readable to the GDM user. However, the daemon proxies user pictures to the greeter. Therefore, those do not have to be readable by the GDM user, but must be readable by the root user.

Note that loading and scaling face icons located in user home directories can be a very time-consuming task, especially on large systems or systems running NIS. The browser feature is only intended for systems with relatively few users. Also, if home directories are on an on-demand mounted file system such as AFS, GDM might mount all of the home directories just to check for pictures if the face browser is on. However, GDM will try to give up after 5 seconds of activity, and only display the users whose pictures have been received so far.

To filter out unwanted user names in the browser, the "Exclude" parameter in **gdm.conf** can be set with a list of usernames separated by commas. The greeter automatically ignores the usernames listed, and excludes users whose UIDs are lower than the "MinimalUID" parameter, which is 100 by default.

When the browser is turned on, valid usernames on the machine are exposed to a potential intruder. This might be a bad idea if you do not know who has access to a login screen. This is especially true if you run XDMCP. Note that you should never run XDMCP on an open network.

The greeter can optionally display a logo in the login window. The image must be in a format readable to the **gdk-pixbuf** library (GIF, JPG, PNG, TIFF, XPM), and must be readable by the GDM user.

Graphical Greeter

The Graphical Greeter is a greeter interface that is displayed on the whole screen and is themable. The Graphical Greeter corresponds to the executable **gdmgreeter**.

Themes can be selected and new themes can be installed by running **gdmsetup**, or by setting the "GraphicalTheme" parameter in **gdm.conf**. The location of themes is specified by the "GraphicalThemeDir" parameter.

The look and feel of this greeter is controlled by the theme, so the user interface elements that are present might differ. The only item that must always be present is the text entry field, as described in the Standard Greeter section above. You can display a menu of available actions by pressing the F10 key. This can be useful if the theme does not provide certain buttons when you wish to perform a particular action.

Chooser

The Chooser displays a list of local machines that accept XDMCP connections. The user can also specify a machine by entering its name directly. Once a machine is selected, a remote XDMCP session can be started. The Chooser can be launched on the console directly from the Standard or Graphical Greeter. The chooser corresponds to the executable **gdmchooser**.

XDMCP

GDM can be configured to enable XDMCP so that users can log in remotely and launch a graphical chooser that allows a remote login session to be started. See the [xdmcp] section of the **gdm.conf** file.

GDM grants access to the hosts specified in the GDM service section of your TCP Wrappers configuration file. GDM does not support remote display access control on systems without TCP Wrappers.

GDM includes several measures that make GDM more resistant to denial-of-service attacks on the XDMCP service. Several protocol parameters, handshaking timeouts, and so on can be fine-tuned. The default values should work for most systems, however. Do not change these values unless you know what you are doing.

By default, GDM listens to UDP port 177, although this can be configured. GDM responds to QUERY and BROADCAST_QUERY requests by sending a WILLING packet to the originator.

GDM can also be configured to honor INDIRECT queries and present a host chooser to the remote display. GDM remembers the user's choice and forwards subsequent requests to the chosen manager. GDM also supports an extension to the protocol which makes GDM forget the redirection once the user's connection succeeds. This extension is only supported if both daemons are GDM. This extension is transparent and is ignored by XDM or other daemons that implement XDMCP.

GDM only supports the MIT-MAGIC-COOKIE-1 authentication system. Because of this, the cookies are transmitted as clear text. Therefore, you should be careful about the network where you use this. That is, be careful about where your XDMCP connection is going. Note that if snooping is possible, an attacker could snoop your password as you log in, so a better XDMCP authentication would not help you much anyway. If snooping is possible and undesirable, you should use **ssh** for tunneling an X connection, rather then using GDM's XDMCP. Think of XDMCP as a sort of graphical telnet, with the same security issues.

Controlling GDM

You can control GDM behavior during runtime in several different ways. You can run certain commands, or you can talk to GDM using either a UNIX socket protocol, or a FIFO protocol.

You can control GDM behavior as follows:

- To stop GDM, you can either send the TERM signal to the main daemon, or run the **gdm-stop** command.
- To restart GDM, you can either send the HUP signal to the main daemon, or run the **gdm-restart** command.
- To restart GDM but only after all users have logged out, you can either send the USR1 signal to the main daemon, or run the **gdm-safe-restart** command.

The **gdm-stop**, **gdm-restart**, and **gdm-safe-restart** commands are in the **/sbin** directory.

The **gdmflexiserver** command can be used to communicate with the GDM daemon and to start new flexible (on demand) servers.

Configuration

The **gdm.conf** file contains comments that explain each configuration parameter.

Security

GDM is best used with a dedicated user id and group id that GDM uses for graphical interfaces such as **gdmgreeter**, **gdmlogin**, and **gdmchooser**. You can specify the name of this user and group in the [daemon] section of the **gdm.conf** file.

The GDM user and group, which are normally just GDM, should not be a user or group of any particular privilege. The reason for using the GDM user and group is to have the user interface run as a user without privileges, so that in the unlikely case that someone finds a weakness in the GUI, they cannot access root on the machine.

Note that the GDM user and group have some privileges that make them somewhat dangerous. They have access to the server authorization directory (specified by the ServAuthDir parameter in **gdm.conf**), which contains all of the X server authorization files and other private information. This means that someone who gains the GDM user/group privileges can then connect to any session. Do not, under any circumstances, make the GDM user/group a user/group that might be easy to get access to, such as the user **nobody**.

The server authorization directory (ServAuthDir) is used for a host of random internal data, in addition to the X server authorization files, and the naming is really a relic of history. The GDM daemon forces this directory to be owned by root:gdm with permissions of 1770. This means that only the root user and the GDM group have write access to this directory, but the GDM group cannot remove the root-owned files from this directory, such as the X server authorization files.

By default, GDM does not trust the server authorization directory and treats it in the same way as a temporary directory with respect to creating files. This means that someone breaking the GDM user cannot mount attacks by creating links in this directory. Similarly, the X server log directory is treated safely, but that directory should really be owned and writable only by the root user.

Accessibility

GDM supports "Accessible Login" to allow users to log in to their desktop session even if they cannot easily use the screen, mouse, or keyboard in the usual way. This feature enables the user to launch assistive technologies at login time by means of special "gestures" from the standard keyboard and from a keyboard, pointing device, or switch device attached to the USB or PS/2 mouse port. This also enables the user to change the visual appearance of the login UI before logging in, for example to use a higher-contrast color scheme for better visibility. GDM only supports accessibility with the Standard Greeter, so the "Greeter" parameter in **gdm.conf** must be set to the Standard Greeter "gdmlogin".

To enable Accessible Login, the system administrator must modify the default login configuration by manually modifying three human-readable configuration files, stored in gdm.conf, AccessKeyMouseEvents, and AccessDwellMouseEvents.

To allow users to change the color and contrast scheme of the login dialog, set the "AllowThemeChange" parameter in **gdm.conf** to "true".

To restrict user changes of the visual appearance to a subset of available themes, the "GtkThemesToAllow" parameter in **gdm.conf** can be set to a list of acceptable themes separated by commas. For example:

GtkThemesToAllow=blueprint,HighContrast,HighContrastInverse

To enable the use of assistive technologies such as the Onscreen Keyboard, Screen Reader, or Magnifier, the "AddGtkModules" parameter in **gdm.conf** must be uncommented and set to "true". Also, the "GtkModulesList" parameter must be uncommented and set to "gail:atk-bridge:dwellmouselistener:keymouselistener".

System administrators might wish to load only the minimum subset of these modules that is required to support their user base. Depending on the end-user needs, it might not be necessary to load all of the GtkModules:

•

If a user needs the integrated Screen Reader and Magnifier, you must include "gail" and "atk-bridge".

- If a user needs a pointing device without buttons or switches, include "dwellmouselistener".
- If a user needs a pointing device with switches, alternative physical keyboard, or switch/button device, include "keymouselistener".

Including all four modules is suitable for most system configurations. The Onscreen Keyboard can operate without gail and atk-bridge, but with a reduced feature set. For optimum accessibility, we recommend including gail and atk-bridge.

When "keymouselistener" or "dwellmouselistener" have been added to the GtkModules loaded by GDM, you can assign user actions to the launching of specific assistive technologies. These gesture associations are contained in the files AccessKeyMouseEvents and AccessDwellMouseEvents, respectively. The gesture format is described in the two files.

The AccessKeyMouseEvents file controls the keymouselistener Gesture Listener and is used to define key-press, mouse button, or XInput device sequences that can be used to launch programs needed for accessibility. To reduce the likelihood of unintentional launch, these ’gestures’ may be associated with multiple switch presses and/or minimum durations.

The DwellKeyMouseEvents file controls the dwellmouselistener and supports gestures that involve only motion of a pointing device such as the system mouse. Motion of an alternative pointing device such as a head pointer or trackball can also be defined. All gestures are specified by the same syntax, there is no distinction between a ’core mouse’ gesture and motion from an alternate input device.

Motion gestures are defined as "crossing events" into and out of the login dialog window. If the ’dwellmouselistener’ GtkModule is loaded, alternative pointing devices are temporarily "latched" to the core pointer, such that motion from alternative devices results in movement of the onscreen pointer.

To use text-to-speech services at login time (for instance, when using the Screen Reader in speech mode) on some operating systems, the gdm user must be a member of the "audio" group.

Logging

GDM uses syslog to log errors or status. GDM can also log debugging information, if enabled in the **gdm.conf** file.

Output from the various X servers is stored in the GDM log directory, which is configurable but is usually *var*/log/gdm. The output from the session can be found in a file called *display*.log. Four older versions of this file are also stored, by appending 1 through 4 to the filename. These files are rotated, as new sessions on that display are started. You can use these logs to view what the X server said when it started up.

The output from the user session is redirected to **~/.xsession-errors** before even the PreSession script is started, so it is not necessary to redirect this again in the session setup script. If the user session lasted less then 10 seconds, GDM assumes that the session crashed and allows the user to view this file in a dialog before returning to the login screen. This enables the user to view the session errors from the last session and correct the problem.

You can suppress the 10-second warning by returning code 66 from the Xsessionscript or from your session binary (the default Xsession script propagates those codes back). This is useful if you have special logins for which it is not an error to return less than 10 seconds later, or if you already set up the session to display an error message and the GDM message would be confusing and redundant.

The session output is piped through the GDM daemon, so the **~/.xsession-errors** file is capped by GDM at about 200 kilobytes, to prevent a possible denial-of-service attack on the session. An application could, on reading some wrong data, print out warnings or errors on stderr or stdout. This could fill up the user’s home directory, the user would then have to log out and log back in to clear this. This could

be especially nasty if quotas are set. GDM also correctly traps the XFSZ signal and stops writing the file, which would lead to killed sessions if the file was redirected in the old-fashioned way from the script.

Note that some distributors seem to override the **~/.xsession-errors** redirection and redirect in their own Xsession script (set by the BaseXsession configuration key), which means that GDM cannot trap the output and cap this file. You also lose output from the PreSession script which can make debugging more difficult, as perhaps useful output of what is wrong is not printed out. See the description of the BaseXsession configuration key for more information, especially on how to handle multiple display managers using the same script.

Note that if the session is a failsafe session, or if GDM cannot open this file for some reason, a fallback file is created named **/tmp/xses-***user*.XXXXXX, where XXXXXX are random characters.

If you run a system with quotas set, use the PostSession script to delete the **~/.xsession-errors** file, so that this log file is not stored unnecessarily.

EXIT STATUS

The following exit values are returned:

0 Application exited successfully

>**0** Application exited with failure

FILES

The following files are used by this application:

/usr/bin/gdm Wrapper script that launches GNOME Display Manager

/usr/bin/gdm-binary Executable for GNOME Display Manager

/usr/bin/gdmchooser Executable for GDM Chooser

/usr/bin/gdmgreeter Executable for GDM Graphical Greeter

/usr/bin/gdmlogin Executable for GDM Standard Greeter

The system administrator can specify, in the **gdm.conf** file, the maximum file size that GDM should accept. If the face browser is enabled, a tunable maximum icon size is also enforced. On large systems, the face browser should be turned off for performance reasons. Looking up icons in home directories, scaling, and rendering face icons can take quite a long time.

In general, GDM is very reluctant to read or write user files. For instance, GDM refuses to touch anything but regular files. Links, sockets, and devices are ignored. The value of the "RelaxPermissions" parameter in the **gdm.conf** file determines whether GDM accepts files that are writable by the user's group or others. These are ignored by default.

Note that normally it is assumed that the home directory is only readable by the user. However, NFS traffic can be snooped. For setups with NFS directories, set the "UserAuthDir" parameter in the **gdm.conf** file to a local directory such as **/tmp**. GDM tries to open the normal authorization file for reading as root. If this fails, GDM concludes that it is on an NFS mount and automatically uses UserAuthFBDir (usually **/tmp**), as defined in the **gdm.conf** file. This can be changed by setting the "NeverPlaceCookiesOnNFS" parameter in the [security] section of the **gdm.conf** file to "false".

GDM Login Scripts and Session Files

The following GDM login scripts are discussed below:

- **/etc/X11/gdm/Init/***hostname*
- **/etc/X11/gdm/Init/XDMCP**
- **/etc/X11/gdm/Init/Default**
- **/etc/X11/gdm/PostLogin/***hostname*
- **/etc/X11/gdm/PostLogin/XDMCP**
- **/etc/X11/gdm/PostLogin/Default**
- **/etc/X11/gdm/PreSession/***hostname*
- **/etc/X11/gdm/PreSession/XDMCP**
- **/etc/X11/gdm/PreSession/Default**
- **/etc/X11/gdm/Xsession**
- **/etc/X11/gdm/PostSession/***hostname*
- **/etc/X11/gdm/PostSession/XDMCP**
- **/etc/X11/gdm/PostSession/Default**

The following session files are discussed below:

- **/usr/share/xsessions/∗.desktop**
- **~/.dmrc** (default user session)

When the X server has been successfully started, GDM tries to run the Init/*displayname* script. For example, **Init/:0** for the first local display. If this file is not found, GDM attempts to run Init/*hostname*. For example, **Init/somehost**. If this file is also not found, GDM tries **Init/XDMCP** for all XDMCP logins or **Init/Flexi** for all on-demand flexible servers. If none of the above are found, GDM runs **Init/Default**. The script runs with root privileges and GDM blocks until the script terminates. Use the **Init/**∗ script for programs that are supposed to run alongside the GDM login window, for example **xconsole**. Commands to set the background and so on should go in this file too.

The system administrator decides whether clients started by the **Init** script should be killed before starting the user session. This is controlled by the "KillInitClients" parameter in **gdm.conf**.

When the user has been successfully authenticated, GDM tries the scripts in the **PostLogin** directory in the same manner as for the **Init** directory. This is done before any session setup is done, so this is the script where you might set up the home directory if you need to (though you should use the pam_mount module for this, if you can). You have the USER and DISPLAY environment variables set for this script, and again it is run with root privileges. The script should return 0 on success as otherwise the user is not logged in. This is not true for failsafe session however.

After the user session has been set up from the GDM perspective, GDM runs the scripts in the **PreSession** directory, again in the same manner as the **Init** directory. Use this script for local session management or accounting. The USER environment variable contains the login of the authenticated user and DISPLAY is set to the current display. The script should return 0 on success. Any other value causes GDM to terminate the current login process. This is not true for failsafe sessions however. Also, the X_SERVERS environment variable is set and this points to a fake generated X servers file for use with the **sessreg** accounting program.

After this, the user's session is started. The available session executables are taken from the Exec= line in the **.desktop** files in the path specified by SessionDesktopDir. The user chooses from these sessions at login time and GDM reads the file **~/.dmrc** for the user's default. The default GNOME session uses the Xsession script. The script is run as the user, and this is the user session. This script should load the user's profile and generally do all that is needed to launch a session. As many systems reset the language selections done by GDM, GDM also sets the GDM_LANG variable to the selected language. You can use this to reset the language environment variables after you run the user's profile. If the user elected to use the system language, then GDM_LANG is not set.

When the user terminates the session, the **PostSession** scripts are run, similar to **Init**, **PostLogin**, and **PreSession**. Again, the script is run with root privileges, the slave daemon blocks, the USER environment variable contains the name of the user who just logged out, and DISPLAY is set to the display the user used. Note, however, that the X server for this display might already be dead so you should not try to access it. Also, the X_SERVERS environment variable is set and points to a fake generated X servers file for use with the **sessreg** accounting program.

Note that the **PostSession** script runs even when the display fails to respond due to an I/O error or similar. Thus, there is no guarantee that X applications will work during script execution.

Except for the **Xsession** script, all of these scripts also have the environment variable RUNNING_UNDER_GDM set to yes, so that you can use similar scripts for different display managers. The **Xsession** always has GDMSESSION set to the basename of the session that the user chose to run, without the **.desktop** extension. In addition, DESKTOP_SESSION is also set to the same value.

None of the **Init**, **PostLogin**, **PreSession**, or **PostSession** scripts are necessary and they can be omitted. However, the **Xsession** script is required, as is at least one session **.desktop** file.

Configuration Files

/etc/X11/gdm/gdm.conf Contains GDM configuration and documentation.

Themes

/usr/share/gdm/themes Can be configured using the GraphicalThemeDir parameter in **gdm.conf**.

Face Browser

~/.face User-defined icon to be used by GDM face browser.

Gesture Listener Files

/etc/X11/gdm/modules/AccessDwellMouseEvents GNOME dwellmouselistener.

/etc/X11/gdm/modules/AccessKeyMouseEvents GNOME keymouselistener.

Logging

/var/log/gdm/*display*.log Output from Xserver for each session. This can be configured using the LogDir parameter in **gdm.conf**.

~/.xsession-errors Output from user's session.

/tmp/xsess-*user*.XXXXXX Output from session in failsafe mode or if **~/.xsession-errors** cannot be written.

Sockets

/tmp/.gdm_socket Temporary file used for GDM socket communications.

Process Id

/var/run/gdm.pid Stores the ProcessID for the running GDM daemon. This can be configured using the PidFile parameter in **gdm.conf**.

Xserver Authentication Directory

/var/lib/gdm Stores Xserver authentication files. This can be configured using the ServAuthDir parameter in **gdm.conf**.

ATTRIBUTES

See **attributes**(5) for descriptions of the following attributes:

tab() allbox; cw(2.750000i)| cw(2.750000i) lw(2.750000i)| lw(2.750000i). ATTRIBUTE TYPEATTRIBUTE VALUE AvailabilitySUNWgnome-display-mgr Interface stabilityExternal

SEE ALSO

Latest version of the *GNOME Desktop User Guide* for your platform.

gdmXnestchooser(1), **gdmflexiserver**(1), **gdmphotosetup**(1), **gdmsetup**(1), **gdmthemetester**(1), **gdm-restart**(1m), **gdmconfig**(1m), **gnome-std-options**(5), **Xserver**(1), **pam**(3pam), **logindevperm**(4)

NOTES

Original man page written by Martin K. Petersen <mkp@mkp.net>, George Lebl <jirka@5z.com>.

Updated by Brian Cameron, Sun Microsystems Inc., 2004.