NAME

auth.props - Sun Ray authentication daemon configuration file.

SYNOPSIS

/etc/opt/SUNWut/auth.props

DESCRIPTION

The **auth.props** file contains the Sun Ray Authentication Manager's configuration properties. Changes to many of these properties are not supported and should not be set to other than the default values.

OPTIONS

The following options are available:

adminConfigFile=*filename*
This file contains the administrative database configuration information.

allowAnnotations=*boolean*
UNSUPPORTED. When set true, any application can connect from any IP address and annotate a session. Annotations are restricted to keywords prefixed by "x_". Values are not restricted.

allowFWLoad=*boolean*
Specifies whether or not the **utload** command is allowed to download firmware to DTUs connected to this Authentication Manager.

allowLANConnections=*boolean*
When set true the DTU connections are allowed from public LAN interfaces, as well as from private Sun Ray interconnect interfaces.

cbport=*portNumber*
UNSUPPORTED. The Authentication Manager listens on this port for connections from the **utsessiond** daemon and other programs, such as **utload**.

cbtimeout=*seconds*
UNSUPPORTED. Specifies the read timeout in seconds for programs that connect to the cbport.

controllers=*maximum*
UNSUPPORTED. Specifies the maximum number of spare threads that are available for handling new connections from applications such as **utload**(1M).

enableGroupManager=*boolean*
UNSUPPORTED. Flag to turn on the group manager function.

enableLoadBalancing=*boolean*
Flag to turn on group manager load balancing.

enableMulticast=*boolean*
UNSUPPORTED. Flag to enable/disable use of multicast in group manager. If disabled, group manager will use broadcast.

forceSessionLocation=*boolean*
UNSUPPORTED. Flag to force use of **sessionHost** and **sessionPort** settings from this file regardless of the various authentication modules.

gmDebug=*level*
UNSUPPORTED. Group manager debugging level.

gmKeepAliveInterval=*seconds*
UNSUPPORTED. The group manager uses this as the time in seconds between broadcast keepalive messages.

gmport=*port*
UNSUPPORTED. The group manager uses this port to send and receive keepalive/discovery messages from other Authentication Managers.

gmSignatureFile=*file*

The group manager can "sign" messages to other group managers based on the contents of a signature file. Other group managers with the same signature file contents are "trusted". To be usable, the file must be owned by ’root’ and must not be readable, writable, or executable by anyone else; it must contain at least 8 bytes, at least two of which are letters and at least one which is a non-letter printable character.

log=*filename*

UNSUPPORTED. This option specifies a file that contains the log messages.

logAddTimeStamp=*boolean*

UNSUPPORTED. Add your own timestamp to syslog messages. This may be appropriate for debugging or in cases where a remote syslog server is being used and higher resolution timestamps are required.

logFacility=*value*

The logFacility can be one of the following:kern, user, mail, daemon, auth, syslog, lpr, news, uucp, cron, local0, local1, local2, local3, local4, local5, local6, local7

Log files

Log priorities for different **utauthd** message classes can be one of the following: emerg, alert, crit, err, warning, notice, info, debug, OFF.

The message classes are:

logPriClientError=*value*

logPriDebug=*value*

logPriNotice=*value*

logPriWarning=*value*

logPriConfigError=*value*

logPriUnexpectedError=*value*

maxStarting=*maximum*

UNSUPPORTED. Specifies the maximum number of threads that can be simultaneously initiating a session. Additional threads wanting to start or verify a session wait for previous threads to finish starting or verifying a session.

moduleDir=*directoryName*

UNSUPPORTED. Specifies the location of the authentication modules.

multicastTTL=*integer*

UNSUPPORTED. Time-to-live parameter for forwarding multicast packets. If set above one, keepalive messages can pass through routers.

noClaimSleepTime=*seconds*

UNSUPPORTED. The amount of time in seconds to sleep after a token has been offered to all of the authentication modules and before notifying the DTU that the authentication failed.

policy=*filename*

Specifies the location of the authentication policy specification.

port=*portNumber*

The **utauthd** daemon listens on this port for connections from Sun Ray DTUs.

remoteSelect=*boolean*
If true, the remote server selection option of the **utselect**(1) command is enabled by default.

reportAllDesktopEvents=*boolean*
UNSUPPORTED. When true, all desktop events are reported instead of being filtered to just those events that change the "exists" state of the DTU.

selectAtLogin=*boolean*
If true, activates a **utselect -L** GUI allowing the user to select a Sun Ray server before logging into CDE. If only one server is available, the GUI exits automatically. Refer to the **utselect** man page for more information on the **-L** option.

sessionHost=*hostname*
UNSUPPORTED. Specifies the host name of the server that is running the default **utsessiond** for this Authentication Manager.

sessionPort=*portNumber*
UNSUPPORTED. Specifies the port number of the server that is running the default **utsessiond** for this Authentication Manager.

sessionTypesFile=*filename*
Specifies a file that contains mappings from session types to the associated session startup and shutdown commands.

smartcardConfigSource = *<list of space-separated sources>*
The keys in this property specify the order in which to search for the configuration files. The special reserved key **LDAP** means go to the configured LDAP database. Any other value refers to a local **probe order** file.

smtimeout=*seconds*
UNSUPPORTED. Specifies the read timeout in seconds for reading messages from the **utsessiond** daemon.

termAddrIsSecret=*boolean*
UNSUPPORTED. When true, the IP address and port of DTUs are not reported in the dynamic status information provided on port cbport in response to the string.

terminalokens = *<list of space-separated tokens>*
UNSUPPORTED. Define the types of tokens that are handled by the terminal rather than by the auth manager. If a token of this type is seen, the auth manager will use the ID value generated by the terminal rather than trying to determine the ID on the server.

terminateEnable=*boolean*
UNSUPPORTED. Enable the cleanup of empty sessions based on notification from the session manager. The default is enabled.

timeout=*seconds*
UNSUPPORTED. DTUs are required to send a message to the Authentication Manager at least once every time period specified by *seconds*.

tokenDir=*directory*
UNSUPPORTED. Specifies a directory that contains the mappings from logical token names to session identifiers. The persistent storage of these mappings allows the **utauthd** daemon to recover its state after restarting. This state is reset on reboot of the system.

token.equiv=*filename*
UNSUPPORTED. Specifies a file that contains mappings from one raw token name to another.

workers=*maximum*
UNSUPPORTED. Specifies the maximum number of spare threads that are available for handling new connections from Sun Ray DTUs.

FILES

The following files are used:

/etc/init.d/utsvc

This is the system startup script that invokes the daemon **/opt/SUNWut/lib/utsessiond**. The session manager performs the actual session switching function.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWutr

SEE ALSO

utauthd(1M), **utpolicy(1M)**, **utsessiond(1M)**, **utselect(1)**

NAME

kiosk - Kiosk Session Service.

DESCRIPTION

The Kiosk Session Service provides an infrastructure which can be used to support unauthenticated access to an application environment in a controlled manner. Unauthenticated access is useful in scenarios where users cannot be expected to provide authentication credentials (e.g. public kiosks etc.) and in scenarios where authentication is expected to be carried out by an application other than the standard login application e.g. Sun Ray Windows Connector.

Kiosk Session Service Overview

The Kiosk Session Service can be described in terms of a number of high level concepts/elements.

Kiosk Session Service User Pool

The Kiosk Session Service maintains a pool of user accounts dedicated specifically to the running of Kiosk sessions. These accounts have the following characteristics.

- They are locked for normal login.
- They belong to a local unix group that has minimal rights on the system.
- No two sessions use the same Kiosk user account at the same time. The Kiosk Session Service User Pool may be configured using **kioskuseradm(1M)**.

Kiosk Session Service Prototypes

Kiosk prototypes are directories whose contents are copied to Kiosk users' home directories. Three specific types of prototypes exist. These are:

- user prototypes:
  These are prototypes which are copied to Kiosk users' home directories prior to every invocation of a Kiosk session and independent of the specific session to be launched.
- session prototypes:
  These are prototypes associated with a specific Kiosk session which are copied to Kiosk users' home directories prior to invocation of that session.
- application prototypes:
  These are prototypes associated with a specific Kiosk application. When and how these prototypes are copied to Kiosk users' home directories is a Kiosk session specific matter and will vary from session to session.

Kiosk Session Service Sessions and Applications

Kiosk sessions consist of a Kiosk Primary Session and optional added Kiosk Applications.

Every Kiosk Primary Session is defined by a Kiosk Session Descriptor. A Kiosk Session Descriptor defines, at a minimum, a session executable which will be launched as the user session. Kiosk Session Descriptors may also define other session specific properties such as a session prototype, pre and post execution scripts and a list of added Kiosk Applications.

Kiosk Application Descriptors are used to define Kiosk applications which may be added to a Kiosk Primary Session. Similar to session descriptors, a Kiosk Application Descriptor defines, at a minimum, an application executable which will be launched when the associated application is invoked. Kiosk Application Descriptors may also define other application specific properties such as an application prototype and pre and post execution scripts.

Not all Kiosk Primary Sessions will support the ability to add Kiosk Applications. For example, while it is reasonable for a Kiosk JDS session to support the ability to launch additional Kiosk Applications, it may make no sense for a Kiosk Full Screen Browser session to have this capability. Kiosk Primary Sessions which support added Kiosk Applications must define an "**application launcher**" identified by the **KIOSK_SESSION_APPLAUNCHER** property in the associated Kiosk Session Descriptor. This application launcher will be invoked by the Kiosk Session Service while processing the list of added applications

and is responsible for launching applications in a session specific manner.

Note: Kiosk Session and Application Descriptors are not delivered as part of the Kiosk Session Service but are provided independently by Kiosk Session authors. Each Kiosk Session provided in this way should clearly document its operation, configuration etc.

Kiosk Session Service Configuration

There are a number of different types of configuration relevant to the Kiosk Session Service.

Kiosk Session Service Framework Configuration

The Kiosk core libraries and utilities use the Kiosk Session Service Framework Configuration to identify locations of various files and directories needed for correct operation of the Kiosk Session Service. The Kiosk Session Service Framework Configuration is available in **/etc/opt/SUNWkio/kioskrc**.

Kiosk Session Service User Configuration

Kiosk Session Service User Account Configuration is used to configure the Kiosk Session Service user account pool and the Kiosk Session Service user accounts maintained within that pool. This configuration may be managed using the **kioskuseradm(1M)** utility.

Kiosk Session Service Policy

Kiosk Session Service Policy indicates whether or not the Kiosk Session Service should be active for a given display. Kiosk Policy is a subset of Kiosk Session Service Session Configuration. For more information see **kioskconfig(1M)** and **session.conf(4)**.

Kiosk Session Service Session Configuration

Kiosk Session Service Session Configuration is used to identify per display Kiosk Session information including Kiosk Session Service Policy, Kiosk Session Descriptor, various session properties used to control resource limitations etc. For more information see **kioskconfig(1M)** and **session.conf(4)**.

Kiosk Session Service Application Lists

The list of applications which should be added to a Kiosk session are collected together in a Kiosk Session Service Application List File. The location of an Application List File to be used by a given Kiosk Session is identified using the KIOSK_SESSION_APPLIST setting in the associated Kiosk Session Service Session Configuration. The format of a Kiosk Session Service Application List is a newline separated list of application entries, each entry having the form

exec|**desc**:*app-name*:*start-mode*;[*arg1*,*arg2*...*argn*]

where

- **exec|desc** indicates whether the subsequent *app-name* refers to an executable or an application descriptor.
- *app-name* is an absolute path or a name indicating an application executable or an application descriptor.
- *start-mode* indicates how the application should/can be started. Valid values are:
- **auto**: the application is started automatically.
- **critical**: the application is started automatically and registered as critical.
- **user**: the application may be started by the user.
- *arg1*,*arg2*...*argn* are the command line arguments to be passed to the application executable.

Enabling and Configuring Kiosk Session Service

Enabling Kiosk sessions on your host is a simple three step procedure.

- Configure the **pam(3PAM)** stack for your login manager. See **pam_kiosk(5)** for details on how to do this.

- Configure the Kiosk Session Service user account pool using the **kioskuseradm(1M)** utility.
- Configure your preferred Kiosk session using the **kioskconfig(1M)** utility.

Locating Prototypes, Descriptors and Application Lists

Prototypes, descriptors and applications lists may be identified either by an absolute path or a well-known name. When a well-known name is used to identify any of these objects, the Kiosk Session Service will search a number of predefined locations when trying to locate the object. The search path used depends on the type of object being searched for and is summarised as follows.

Note: the **KIOSK_*** environment variables mentioned below are specified and may be configured in the Kiosk Session Service Framework Configuration as described above.

- **prototypes**:
  prototypes are searched for using **$KIOSK_PROTOS_DIR/<prototype name>**
- **descriptors**:
  session descriptors are searched for using **$KIOSK_SESSIONS_DIR/<session name>.conf**
  application descriptors are searched for using **$KIOSK_APPS_DIR/<application name>.conf**
- **application lists**:
  application lists are searched for using **$KIOSK_APPS_DIR/<application list name>.list**

ISSUES

gdm(1) and the gdm greeter

The gdm login manager will invoke its configured greeter regardless of whether or not authentication credentials are needed. In the case of a Kiosk session, a greeter is neither needed nor desired. In some instances, the greeter may become temporarily visible during the initialisation of Kiosk sessions. As a workaround for this, an alternative greeter, **/opt/SUNWkio/lib/gdm/kioskgreeter** is supplied with the Kiosk Session Service. kioskgreeter depends on non public interfaces of gdm and, as such, is supplied unsupported and initially unconfigured. To configure kioskgreeter you should edit your gdm configuration file, usually **/etc/X11/gdm/gdm.conf**, modifying the **Greeter** setting as follows.

Greeter=/opt/SUNWkio/lib/gdm/kioskgreeter <original_greeter>

where **<original_greeter>** is the previous value of the Greeter setting, if available.
You must restart gdm after this modification.

Kiosk sessions and screen lock applications

Some environments support a screen lock function which locks the session after a period of inactivity or under other circumstances. Access to such a locked session is only granted when the user has reauthenticated, typically by providing a password. Kiosk user accounts have no password set and are locked for ordinary login. Because of this Kiosk users will not be able to unlock their session, if a screen lock is activated.

Session authors are encouraged to make sure their Kiosk sessions have screen lock functionality disabled. If this is not possible, and if the screen lock program supports the **pam(3PAM)** authentication framework, you can guard against this lockout scenario by adding the **pam_kiosk(5)** module to the PAM stack for the screen lock program in **pam.conf(4)** with the *reentry* option.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWkio, SUNWkior
Interface Stability	Committed

SEE ALSO

kioskdesc(1), **kioskstatus(1)**, **kioskconfig(1M)**, **kioskuseradm(1M)**, **kioskrestart(1M)**, **session.conf(4)**, **pam_kiosk(5)**

NAME

kioskconfig - generate Kiosk session configuration

SYNOPSIS

/opt/SUNWkio/bin/kioskconfig enable [**-d** *display*|**-f** *display-file*] [**-c** *config-file*]

/opt/SUNWkio/bin/kioskconfig apply [**-d** *display*|**-f** *display-file*] [**-c** *config-file*]

/opt/SUNWkio/bin/kioskconfig disable [**-d** *display*|**-f** *display-file*|**-a**]

/opt/SUNWkio/bin/kioskconfig print [**-d** *display*|**-f** *display-file*|**-a**]

/opt/SUNWkio/bin/kioskconfig list

/opt/SUNWkio/bin/kioskconfig reset

/opt/SUNWkio/bin/kioskconfig help

/opt/SUNWkio/bin/kioskconfig -h

DESCRIPTION

kioskconfig is used to generate or show Kiosk session configuration for a given display or set of displays. Kiosk session configuration describes the desired user environment of Kiosk sessions. This description defines Kiosk policy (indicating if Kiosk mode is enabled or disabled) and identifies the actual Kiosk session to be used along with settings used to configure Kiosk sessions. For a full description of Kiosk session configuration see **kiosk(5)**.

kioskconfig can perform its operations either on a single display, specified by the **-d** option, on a set of displays listed in a **display-file** specified by the **-f** option or on all displays for which kiosk configuration currently exists. For operations that can operate on selected displays, operation on all currently enabled displays is requested using the **-a** option. If no display selection option is provided, the default display file **/etc/opt/SUNWkio/displays** is used.

For the operations that can enable Kiosk mode for a display, a kiosk session configuration file must be provided. If no *config-file* is specified by the **-c** option, the default location **/etc/opt/SUNWkio/session.conf** is used. For details on the form and content of the configuration file see **session.conf(4)**.

The Kiosk Session Service does not provide standard display or session configuration files in the default locations. If you wish to use one of these files, you must create them yourself.

Configuration changes apply the next time a session is started on a configured display. If kiosk sessions are already running on the affected displays, you can use **kioskrestart(1M)** to force a restart of the sessions.

SUBCOMMANDS

The following subcommands are supported.

enable

Enable and configure Kiosk mode on one or more displays. Kiosk session configuration is created for the specified display(s), based on the specified configuration file. The **enable** command will always enable kiosk on the specified display(s). If the configuration file specifies Kiosk policy itself (i.e. it contains the **KIOSK_ENABLED** setting), this policy setting is ignored.

apply Apply Kiosk session configuration which includes a policy setting to one or more displays. The specified configuration file should contain the **KIOSK_ENABLED** setting to indicate whether kiosk should be enabled or disabled on the specified display(s). If the configuration file does not contain a policy setting, kiosk will be disabled on the specified display(s).

disable

Disable Kiosk mode for the specified display(s).

print Print the current Kiosk session configuration for the specified display(s).

list List the displays on which kiosk mode is currently enabled.

reset Remove all current kiosk session configuration. Executing the **reset** command disables Kiosk mode on all displays and removes internal data structures. This command should only be used to

deconfigure Kiosk mode completely, for example after deconfiguring all kiosk user accounts.

help Show a help message.

OPTIONS

The following options are supported.

-a Operate on all currently configured displays.

-d *display*
Identifies a single display to be configured.

-f *display-file*
Identifies a file which lists the displays to be configured. This file must contain a list of X11 display identifiers, one per line. Comment lines starting with '#' are supported. If neither the **-d** nor **-f** nor **-a** options are specified, a display file from a default location is used. The location of the default display file is **/etc/opt/SUNWkio/displays**. The Kiosk Session Service does not provide a default display file. If you wish to use one, you must provide it yourself.

-c *config-file*
Identifies a file containing Kiosk configuration. If the **-c** option is not specified, a default configuration file location is used. The default location of the configuration file is **/etc/opt/SUNWkio/session.conf**. The Kiosk Session Service does not provide a default configuration file. If you wish to use one, you must provide it yourself.

-h Show a usage message.

EXAMPLES

Example 1: Enabling a set of displays using the default configuration file

Given a default display file, **/etc/opt/SUNWkio/displays** containing the following:

```
:1
:2
:3
:4
```

and a default configuration file, **/etc/opt/SUNWkio/session.conf** containing the following:

```
KIOSK_ENABLED=no
KIOSK_SESSION=gnome
KIOSK_SESSION_PROTOTYPE=/export/home/prototypes/gnome
```

the command

```
kiosk# kioskconfig enable
```

will enable Kiosk sessions for displays :1, :2, :3 and :4. Kiosk sessions on these displays will use the **gnome** Kiosk Session Descriptor and the **/export/home/prototypes/gnome** prototype directory. The **KIOSK_ENABLED** setting will be ignored.

Example 2: Applying configuration to single display

Using the configuration file shown above, the command

```
kiosk# kioskconfig apply -d :2
```

will apply all Kiosk configuration, including the **KIOSK_ENABLED** setting, in the default configuration file **/etc/opt/SUNWkio/session.conf** to the :2 display. This will effectively disable Kiosk sessions for the :2 display. This could also be achieved using the command

```
kiosk# kioskconfig disable -d :2
```

Example 3: Listing displays for which Kiosk Mode is enabled

After executing the commands from the previous examples, you can list the displays on which kiosk is now enabled using the command

```
kiosk% kioskconfig list
:1
:3
:4
```

EXIT STATUS

The following exit values are returned.

0 Success

1 Failure

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWkio
Interface Stability	Uncommitted

SEE ALSO

kiosk(5), **session.conf(4)**

NAME

kioskdesc - list or print contents of Kiosk Session Service descriptors and application lists

SYNOPSIS

/opt/SUNWkio/bin/kioskdesc list **-a**|**-A**|**-s** [*name*]

/opt/SUNWkio/bin/kioskdesc print {**-a**|**-A**|**-s**} *name*

/opt/SUNWkio/bin/kioskdesc help

/opt/SUNWkio/bin/kioskdesc -h

DESCRIPTION

kioscdesc may be used to list available Kiosk Session Service descriptors and application lists.

SUBCOMMANDS

The following subcommands are supported:

list List available Kiosk Session Service descriptors and application lists or test for the existence of a named descriptor or application list. If no *name* is provided, a list of names is printed, which enumerates all available Kiosk Session Service descriptors or application lists. If *name* is provided, **kioskdesc** tests for the existence of the named Kiosk Session Service descriptor or application list. If the descriptor or application list exists, the full path to the descriptor or application list file is printed.

print Print the contents of the Kiosk Session Service descriptor or application list named *name*.

help Prints a help message.

OPTIONS

The following options are supported:

-a The list or print subcommand is applied to Kiosk Session Service application descriptors.

-A The list or print subcommand is applied to Kiosk Session Service application lists.

-s The list or print subcommand is applied to Kiosk Session Service session descriptors.

-h Prints a usage message.

OPERANDS

The following operands are supported:

name The name of a Kiosk Session Service descriptor to be listed or whose contents should be printed. *name* may be a full path to a descriptor or application list or a well-known name used to search predefined locations for descriptors or application lists. For more details on the use of well-known names to locate descriptors and applications lists, see **kiosk(5)**.

EXAMPLES

Example 1: Listing available descriptors

You may use the **list** subcommand to list all available session descriptors as follows.

```
kiosk% kioskdesc list -s
jds3
uttsc
kiosk%
```

Similarly, use the **list** subcommand to list all available application descriptors as follows.

```
kiosk% kioskdesc list -a
browser
calculator
charmap
editor
```

```
imageviewer
messenger
sun_webstart
kiosk%
```

Example 2: Testing for the existence of a descriptor

You may use the **list** subcommand as follows to check if a given session descriptor exists.

```
kiosk% kioskdesc list -s jds3
/etc/opt/SUNWkio/sessions/jds3.conf
kiosk%
```

Similarly, use the **list** subcommand as follows to check if a given application descriptor exists.

```
kiosk% kioskdesc list -a browser
/etc/opt/SUNWkio/applications/browser.conf
kiosk%
```

Example 3: Printing the contents of a descriptor

You may use the **print** subcommand as follows to print the contents of a given session descriptor.

```
kiosk% kioskdesc print -s uttsc
KIOSK_SESSION_ARGS=-t 1800 -- -m -b
KIOSK_SESSION_DESCRIPTION=A full screen Sun Ray Windows Connector session.
KIOSK_SESSION_EXEC=/etc/opt/SUNWkio/sessions/uttsc/uttsc
KIOSK_SESSION_ICON=
KIOSK_SESSION_LABEL=Sun Ray Connector for Microsoft Windows OS
kiosk%
```

Similarly, use the **print** subcommand as follows to print the contents of a given application descriptor.

```
kiosk% kioskdesc print -a sun_webstart
KIOSK_APP_DESCRIPTION=Java Web Start
KIOSK_APP_EXEC=/usr/java/jre/javaws/javaws
KIOSK_APP_ICON=/usr/java/jre/plugin/desktop/sun_java.png
KIOSK_APP_LABEL=Java Web Start
KIOSK_APP_POST=
KIOSK_APP_PRE=
KIOSK_APP_PROTOTYPE=
kiosk%
```

EXIT STATUS

The following exit values are returned.

0 Success

1 Failure

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWkio
Interface Stability	Uncommitted

SEE ALSO

kiosk(5)

NAME

kioskrestart - terminate kiosk sessions to restart them in a clean state

SYNOPSIS

/opt/SUNWkio/bin/kioskrestart

/opt/SUNWkio/bin/kioskrestart -a

/opt/SUNWkio/bin/kioskrestart -d *display ...*

/opt/SUNWkio/bin/kioskrestart -u *username ...*

/opt/SUNWkio/bin/kioskrestart -h

DESCRIPTION

The **kioskrestart** utility may be used to terminate running kiosk sessions. This usually causes the sessions to be restarted from scratch. Restarting a kiosk session can be used to let changes to kiosk configuration take effect or to reset sessions to a known, clean state.

Sessions are restarted by simulating termination of a critical application. This works only for sessions in the "running" state, as reported by **kioskstatus -x**. In particular this command may report errors, if any affected kiosk sessions are currently starting or terminating on their own.

The **kioskrestart** command can be used to restart the session within which it is invoked or one or more specified sessions. To restart specified sessions one of the **-a**, **-d** or **-u** options must be specified. Only one of these options can be specified. Root privileges are required to restart specific sessions.

The first form of the **kioskrestart** command, without any options, is used to restart the current kiosk session. This form can be used only within a kiosk session.

The second form of the **kioskrestart** command, with the **-a** option, is used to restart all currently running kiosk sessions.

The third form of the **kioskrestart** command, with the **-d** option, is used to restart the kiosk sessions on one or more specified displays.

The fourth form of the **kioskrestart** command, with the **-u** option, is used to restart the kiosk sessions running under one or more specified kiosk user accounts.

The last form of the **kioskrestart** command, with the **-h** option, displays a usage message.

OPTIONS

The following options are supported. The **-a**, **-d** and **-u** options are mutually exclusive. Without any option, **kioskrestart** terminates the current kiosk session.

-a Terminate all currently active kiosk sessions.

-d Terminate the kiosk sessions for the specified displays.

-u Terminate the kiosk sessions for the specified user accounts.

-h Displays a help message.

EXAMPLES

Example 1: Terminating all kiosk sessions.

To terminate all kiosk sessions, for example after disabling kiosk mode on all displays, use the **kioskrestart** command with the **-aoption:**

```
kiosk# kioskrestart -a
kiosk#
```

Example 2: Terminating a specific kiosk session.

To terminate a kiosk session on a specific display, for example after changing kiosk configuration for this display, use the **kioskrestart** command with the **-doption:**

```
kiosk# kioskrestart -d :4
kiosk#
```

Example 3: Terminating a kiosk session for a specific user account.

To terminate a kiosk session running under a specific kiosk user account, use the **kioskrestart** command with the **-uoption:**

```
kiosk# kioskrestart -u ku13
kiosk#
```

EXIT STATUS

The following exit values are returned:

0 All specified sessions were terminated successfully.

non-zero
Terminating at least one specified session failed.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWkio
Interface Stability	Uncommitted

SEE ALSO

kiosk(5), **kioskconfig(1M)**, **kioskuseradm(1M)**

NAME

kioskstatus - indicate the current Kiosk Session Service status for a given display or user.

SYNOPSIS

/opt/SUNWkio/bin/kioskstatus [**-hqx**]

/opt/SUNWkio/bin/kioskstatus -d [**-qx**] [*display* ...]

/opt/SUNWkio/bin/kioskstatus -u [**-q**] [*username* ...]

/opt/SUNWkio/bin/kioskstatus -c [**-q**]

/opt/SUNWkio/bin/kioskstatus -h

DESCRIPTION

The **kioskstatus** utility may be used to query whether Kiosk mode is active in a system session and the overall configuration status of Kiosk Session Service.

The first three forms of the **kioskstatus** command indicate whether or not a given X11 display or user account is currently running a valid Kiosk session. The fourth form of **kioskstatus** indicates whether Kiosk Session Service is configured, so that kiosk sessions can be started.

Without any options, **kioskstatus** will report whether or not the current session (identified by the current value of **$DISPLAY**) is a valid Kiosk session. In addition to checking the status for the session allocated to the current display, it is also verified that this session is allocated to the current user's login name, as reported by logname(1).

kioskstatus may be used to check sessions on other displays by specifying the **-d** option. No user identity check is performed in this case. If no *display* is specified, the current session (identified by the current value of **$DISPLAY**) is checked.

The kiosk status of a session can be determined by inspecting a marker property set on the X11 root window or by querying the kiosk user account allocation data maintained by Kiosk Session Services. By default session status is determined using kiosk user account allocation queries. The **-x** option may be used to perform an X property query instead. Generally the X property query more accurately reflects the state of an active kiosk session, but such a query requires authorization to connect to the X server for the inspected session and may hang, if the X server is currently grabbed, for example by a display manager showing a login greeter.

Alternatively, **kioskstatus** can determine whether specified user accounts are currently assigned to an active kiosk session by specifying the **-u** option. If no *username* is specified, the current user's login name, as reported by **getlogname(1)**, is checked.

kioskstatus may also be used to check whether Kiosk Session Service configuration is complete, so that kiosk sessions can be started. It is checked that kiosk user accounts are configured. You can configure kiosk user accounts using the **kioskuseradm(1M)** tool.

The last form of **kioskstatus** displays a short help message.

OPTIONS

The following options are supported.

-c Query whether Kiosk Session Service is configured.

-d Query kiosk status for the specified displays or for the current X display.

-u Query kiosk status for the specified user accounts or for the current user.

-x Read the kiosk status from the X server for the display(s). This option requires authorization to connect to the X server for the session. The X server access may hang, if the X server is grabbed.

-q Quiet mode. Indicates that the status should not be printed to standard output. This option is most useful for scripting.

-h Displays a short help message.

EXAMPLES

Example 1: Querying users' Kiosk status

Given a Kiosk user, ku1000, who is currently running a Kiosk session, a Kiosk user ku1001, who is not currently running a Kiosk session and a non Kiosk user user1002, using kioskstatus to query their status will produce the following output.

```
kiosk% kioskstatus -u ku1000 ku1001 user1002
ku1000: active
ku1001: not active
user1002: not a kiosk account
kiosk%
```

Example 2: Querying a display's Kiosk status

Given a display, :2, which is currently running a Kiosk session and a display, :3, which is not currently running a Kiosk session, using kioskstatus to query their Kiosk statuses will produce the following output.

```
kiosk% kioskstatus -d :2 :3
:2: active
:3: not active
kiosk%
```

Assuming the current user has access to both X servers, using kioskstatus to query the X servers for their Kiosk status will produce the following output.

```
kiosk% kioskstatus -dx :2 :3
:2: running
:3: disabled
kiosk%
```

EXIT STATUS

The following exit values are returned:

0 If the **-c** option is used, an exit status of 0 indicates that Kiosk Session Service is configured. Otherwise an exit status of 0 indicates that a Kiosk session is active for all specified displays or usernames.

1 If the **-c** option is used, an exit status of 1 indicates that the Kiosk Session Service is not configured. Otherwise an exit status of 1 indicates that there is no active Kiosk session for at least one of the specified displays or usernames.

2 An exit status of 2 indicates that determining the status of Kiosk Session Service configuration or of at least one of the specified displays or usernames failed due to an error.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWkio
Interface Stability	Uncommitted

SEE ALSO

kiosk(5)

NAME

kioskuseradm - Kiosk Session Service user administration tool

SYNOPSIS

/opt/SUNWkio/bin/kioskuseradm show [**-p**]

/opt/SUNWkio/bin/kioskuseradm create [**-q**] **-l** *prefix* **-g** *group* [**-i** *gid*] **-u** *first-uid* **-c** *count*

/opt/SUNWkio/bin/kioskuseradm modify [**-fq**] [**-l** *prefix*] [**-g** *group* [**-i** *gid*]] [**-u** *first-uid*] [**-c** *count*]

/opt/SUNWkio/bin/kioskuseradm extend [**-q**] -c *count*

/opt/SUNWkio/bin/kioskuseradm delete [**-fq**]

/opt/SUNWkio/bin/kioskuseradm leakcheck [**-p**]

/opt/SUNWkio/bin/kioskuseradm cleanup [**-q**]

/opt/SUNWkio/bin/kioskuseradm status [**-pv**]

/opt/SUNWkio/bin/kioskuseradm help

/opt/SUNWkio/bin/kioskuseradm -h

DESCRIPTION

The Kiosk Session Service maintains a pool of user accounts dedicated to running Kiosk sessions. **kioskuseradm** may be used to administer this pool of accounts.

SUBCOMMANDS

The following subcommands are supported.

show Print the current Kiosk User account configuration. With the **-p** option the output has the form of an option line that can be passed to the **create** or **modify** commands to restore the same configuration.

create

Create *count* Kiosk user accounts in the group *group*. Consecutive uids starting at *first-uid* will be used for the accounts. The user name for the created accounts will take the form *prefix***number** where **number** runs from 0 to *count*-1.

If the **-i** option is not provided, the group *group* must exist on the system. If **-i** is specified, a group named *group* is created with the group-id *gid*. A group of that name must not exist on the system.

The **create** command will fail, if a Kiosk user account configuration already exists.

modify

Modify an existing Kiosk user account configuration. The existing configuration is deleted and recreated with the new settings. Settings that are not specified in the invocation of this command are kept unchanged from the old configuration. Otherwise the effect of this command is the same as running the **delete** command followed by the **create** command.

The **modify** command will fail, if Kiosk user accounts are not yet configured. If there are existing Kiosk sessions, the **modify** command will fail unless the **-f** option is specified.

extend

Extend the current range of Kiosk user accounts by *count* additional accounts.

The **extend** command can be used to add more Kiosk accounts even while there are existing Kiosk sessions.

delete Delete the existing Kiosk user accounts.

If the group used for the Kiosk user accounts was created by **kioskuseradm** by specifying the **-i** *gid* option to the **create** or **modify** command, the group will also be deleted. Otherwise the group will not be deleted.

If there are existing Kiosk sessions, the **delete** command will fail unless the **-f** option is specified. If the **-f** option is provided the **delete** command will attempt to terminate existing Kiosk sessions before deleting the Kiosk user account.

If deleting one or more user accounts fails, you can use the **leakcheck** command to find any leftover accounts and the **cleanup** command to delete these accounts once the circumstances that prevented their deletion are resolved.

leakcheck
Search for Kiosk user accounts left over from a prior configuration. Use the **cleanup** command to delete any accounts found once the circumstances that prevented their deletion are resolved. With the **-p** option, the output of the command is a list of user account names, one per line.

cleanup
Force the removal of "orphaned" Kiosk accounts, which could not be removed when a prior configuration was deleted or modified. Use the **leakcheck** command to find such accounts.

status Print the number of configured and currently used Kiosk user accounts. With the **-p** option, the output of the command is a pair of numbers

total-count used-count

With the -v option, the command also lists all Kiosk user accounts currently in use and the session (the X11 display) to which they are allocated. With both the **-p** and **-v** options, the output of the command is a list of the used user account names and their associated displays, one account per line.

help Prints a help message.

OPTIONS

-l *prefix*
Specify the prefix for all Kiosk user account names. The prefix must start with a letter which can be followed by up to three more letters or digits.

-g *group*
Specify the name of the group which all Kiosk user accounts will belong to. Any valid group name may be specified. If the **-i** *gid* option is also specified, this group will be created with the given *gid*. In that case the group will also be deleted when the Kiosk user account configuration is deleted. Otherwise the group must exist on the system and will not be deleted with the Kiosk accounts.

-i *gid* Specify the group-id of the group which all Kiosk users account will belong to. The group specified using the **-g** *group* option will be created using this group-id. Any valid unused group id can be specified.

The special value "auto" can be specified as the *gid*. In that case selection of a group id is left to the groupadd(1M) tool.

-u *first-uid*
Specify the starting user-id to be used for the Kiosk user account pool. A number greater than 100 must be specified. All uids in the range *first-uid* ... *first-uid*+*count*-1 must be available for Kiosk account configuration to succeed.

-c *count*
Specify the number of new Kiosk user accounts to create. When using the **create** or **modify** commands, *count* indicates the the total number of Kiosk user accounts requested. When using the **extend** command, *count* indicates the number of new Kiosk user accounts that should be added to the user account pool. You may specify any positive number of up to 4 digits.

-f Force termination of existing session when deleting user accounts.

-p Produce parseable instead of human readable output.

-q Quiet Mode. Suppress extensive progress messages.

-v Verbose mode. Provide more detail in output.

-h Print a usage message.

EXAMPLES

Example 1: Initial configuration of Kiosk user pool

The command

```
kiosk# kioskuseradm create -l ku -g kiosk -u 1000 -c 10
```

will create 10 Kiosk user accounts, starting with uid 1000. The accounts will be named, ku0, ku1, ku2 etc. The accounts will belong to the preexisting group named kiosk.

Example 2: Displaying the current configuration of the Kiosk user account pool

Assuming the Kiosk user account pool has been created using the command shown in Example 1, the pool configuration may be displayed as follows

```
kiosk# kioskuseradm show
Current kiosk user account settings:
  user name prefix:   ku
  first account uid:  1000
  number of accounts: 10
  kiosk group name:   kiosk
kiosk# kioskuseradm show -p
-l ku -g kiosk -u 1000 -c 10
```

Example 3: Extending the Kiosk user account pool

Assuming the Kiosk user account pool has been created using the command shown in Example 1, the pool could be increased in size to allow for 20 Kiosk users using the command

```
kiosk# kioskuseradm extend -c 10
```

Example 4: Removing the Kiosk user pool

If you no longer need to run Kiosk sessions, you may delete all accounts in the Kiosk user pool using the command

```
kiosk# kioskuseradm delete
```

EXIT STATUS

The following exit values are returned.

0 Success

non zero

Failure

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWkio
Interface Stability	Uncommitted

SEE ALSO

kiosk(5)

NAME

libusbut.so.1 - Sun Ray libusb plugin

SYNOPSIS

`/opt/SUNWut/lib/libusbut.so.1`

DESCRIPTION

The Sun Ray **libusb** plugin provides Sun Ray-specific support for **libusb**.

The **SUNWlibusbut** package delivers the Sun Ray libusb plugin **libusbut.so.1** in **/opt/SUNWut/lib**. This plugin is loaded by the **libusb** wrapper library **libusb.so**, which is delivered as part of the **SUNWlibusb** package.

For the wrapper library to be able load the plugin the following link must exist in /usr/sfw/lib/libusb_plugins:

libusbut.so.1 -> /opt/SUNWut/lib/libusbut.so.1

If the **SUNWlibusb** package is installed in the system prior to Sun Ray Server Software installation, then the required **symlink** is created automatically.

If the **SUNWlibusb** package is not installed, the administrator needs to install the **SUNWlibusb** package and then create the symlink as follows:

```
# cd /usr/sfw/lib/libusb_plugins
# ln -s /opt/SUNWut/lib/libusbut.so.1
```

See also /usr/sfw/share/doc/libusb/libusb.txt

NOTES

The Sun Ray **libusb** plugin currently does not support:

- usb_interrupt_write
- timeouts for the I/O calls.

Note: Do not fork processes to do I/O. Use threads instead.

The **SUNWlibusb** package can be found in the Supplemental area of the Sun Ray Server Software.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	**SUNWlibusb**, **SUNWlibusbut**
MT-Level	UnSafe
Interface Stability	External

SEE ALSO

intro(3), **attributes(5)**

NAME

pam_kiosk - Kiosk Session Service pam module.

SYNOPSIS

/opt/SUNWkio/lib/pam_kiosk.so

DESCRIPTION

pam_kiosk is the Kiosk Session Service pam module used to bypass normal authentication for Kiosk sessions. The **pam_kiosk** module is invoked by the display manager during session startup.

pam_kiosk provides the following functions.

1. **[all modules]** Return **PAM_IGNORE** and perform no further action, if the session is known not to be a Kiosk session.
2. **[auth module]** Evaluates Kiosk policy to find out if Kiosk is enabled.
3. **[auth module]** If a user identity was already set, **pam_kiosk** can clobber it and try to establish the Kiosk user anyway or ignore the Kiosk policy and proceed to login that user. The default behaviour is to ignore Kiosk policy, which can be changed by setting the **ignoreuser** option.
4. **[auth module]** Identifies the user account to use for the session and establishes the user identity for the session.
5. **[auth module]** Returns **PAM_SUCCESS** from **pam_sm_authenticate(3PAM)** to login the account without requiring any actual credentials.
6. **[auth module]** In **pam_sm_setcred(3PAM)** the X session is marked as a Kiosk session. Returns **PAM_IGNORE** to allow normal credential setting (**pam_unix_cred(5)**) to proceed.
7. **[account module]** Returns **PAM_SUCCESS**, if this is a Kiosk session, to short-circuit account checking, as the Kiosk user accounts should be locked.
8. **[session module]** Creates/Resets the user account home directory at session start.
9. **[session module]** Cleans up the user account at session end.

OPTIONS

pam_kiosk supports the following options.

debug
: Send debug messages to syslog using the **auth.debug** mode (see **syslog.conf(4)**). This option can be provided multiple times to increase the amount of debug output generated.

nowarn
: Don’t output non-fatal warning or error messages.

ignoreuser
: Ignore (clobber) an existing **PAM_USER** item. Default behavior is to override the Kiosk policy and fall back to an authenticated session.

log=*facility*
: Use *facility* for logging events to the system log. Supported values for *facility* **are AUTH**, **USER** and **LOCAL0-7**,

reentry
: Check if the current X display and PAM_USER user account belong to an active kiosk session. If they do, **pam_sm_authenticate(3PAM)** returns success. Use this option only in the authentication stack for a screen lock program, to allow kiosk sessions for which a screen lock has been activated to unlock. This option has an effect only on the authentication module.

USAGE

To use the **pam_kiosk** module you must correctly configure the pam stack for your login manager as follows

Configuring the pam stack on Solaris

Both **dtlogin(1X)** and **gdm(1)** are supported on Solaris. To configure either of these you must modify the pam configuration file **/etc/pam.conf** as follows, replacing **service** with dtlogin **or** gdm **as appropriate.**

- Insert the line

  service auth sufficient /opt/SUNWkio/lib/pam_kiosk.so

  This line **must** be before any line that features pam_authtok_get for your login manager.

- Insert the line

  service account sufficient /opt/SUNWkio/lib/pam_kiosk.so

  This line must be first in the **account** stack for your login manager.

- Insert the line

  service session required /opt/SUNWkio/lib/pam_kiosk.so

  This line must be first in the **session** stack for your login manager.

The following is an example of the resulting pam stack for dtlogin.

```
dtlogin auth    sufficient  /opt/SUNWkio/lib/pam_kiosk.so
dtlogin auth    requisite   pam_authtok_get.so.1
dtlogin auth    required    pam_dhkeys.so.1
dtlogin auth    required    pam_unix_cred.so.1
dtlogin auth    required    pam_unix_auth.so.1
dtlogin account sufficient  /opt/SUNWkio/lib/pam_kiosk.so
dtlogin account requisite   pam_roles.so.1
dtlogin account required    pam_unix_account.so.1
dtlogin session required    /opt/SUNWkio/lib/pam_kiosk.so
dtlogin session required    pam_unix_session.so.1
```

Configuring the pam stack on Linux

Only **gdm(1) is supported on Linux. To configure gdm you must modify the gdm pam configuration file /etc/pam.d/gdm** as follows.

- Insert the line

  auth sufficient /opt/SUNWkio/lib/pam_kiosk.so

  This line **must** be before any line that features pam_unix.

- Insert the line

  account sufficient /opt/SUNWkio/lib/pam_kiosk.so

  This line must be first in the **account** stack for gdm.

- Insert the line

  session required /opt/SUNWkio/lib/pam_kiosk.so

  This line must be first in the **session** stack for gdm.

The following is an example of the resulting pam stack for gdm.

```
auth     sufficient /opt/SUNWkio/lib/pam_kiosk.so
auth     required   pam_unix2.so   nullok #set_secrpc
account  sufficient /opt/SUNWkio/lib/pam_kiosk.so
account  required   pam_unix2.so
password required   pam_unix2.so   #strict=false
session  required   /opt/SUNWkio/lib/pam_kiosk.so
session  required   pam_unix2.so   debug # trace or none
session  required   pam_devperm.so
session  optional   pam_console.so
```

Prevent fallback to an ordinary session

The **pam_kiosk** module can be specified multiple times in the same PAM stack. In this case only the first invocation will perform the full function of the module. Subsequent invocations will detect that the function has already been performed and will simply return an approriate result code reflecting the success, ignore or error state of the primary invocation (see section ERRORS). This can be used to prevent PAM processing to continue when kiosk setup has failed due to an error, by specifying **pam_kiosk** twice as follows (in **pam.conf(4)** syntax):

```
service auth sufficient /opt/SUNWkio/lib/pam_kiosk.so
service auth requisite  /opt/SUNWkio/lib/pam_kiosk.so
...
```

Configuring for screen lock programs

Kiosk user accounts have no password set and are locked for ordinary login. Because of this Kiosk users will not be able to unlock their session, if a screen lock is activated.

To guard against this lockout scenario, you can use the **pam_kiosk** module with the *reentry* **option, so that the screen lock program will instantly unlock without user action, if pam_kiosk detects a kiosk session. To do this specify pam_kiosk** at the very top of the PAM stacks for screen lock services as follows:

```
lock-service auth sufficient /opt/SUNWkio/lib/pam_kiosk.so reentry
...
```

ERRORS

All modules return the following error codes:

PAM_SUCCESS

if the current session is a kiosk session and the module has performed its function successfully.

PAM_IGNORE

if the current session is not a kiosk session.

PAM error codes

if determining the kiosk policy has failed or if Kiosk Mode is enabled for the session, but setting up the kiosk session failed.

Some reasons for failure are

- There are no kiosk user accounts available.
- The kiosk session configuration is invalid.
- Eliminating remnants of a prior kiosk session using the same account has failed.
- Setting up the environment for the kiosk session has failed.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWkio
Interface Stability	Committed

SEE ALSO

kiosk(5), **pam(3PAM)**, **syslog.conf(4)**

NAME

session.conf - Kiosk Session Service session configuration file.

SYNOPSIS

/etc/opt/SUNWkio/session.conf

DESCRIPTION

session.conf is the default configuration input file for **kioskconfig(1M)**. The contents of session.conf indicate, at a minimum, which Kiosk session should be used for Kiosk enabled displays. Other Kiosk session properties configuring resource limitations, session arguments etc. may also be specified in session.conf. Kiosk session properties specified in session.conf override properties set in the associated Kiosk session descriptor.

The format of session.conf is that of a valid Bourne shell script where Kiosk session properties are defined using environment variables.

The following is a list of the environment variables which may be used to configure your preferred Kiosk session. With the exception of **KIOSK_SESSION**, all environment variables are optional. Unless otherwise stated, no default values are applied.

KIOSK_ENABLED
Indicates whether or not Kiosk should be enabled. Valid values are **yes** or **no** indicating that Kiosk is enabled or disabled respectively. The default value is **no**.

KIOSK_SESSION
Identifies which Kiosk session should be used. You may specify either the well-known name of or absolute path to a Kiosk Session Descriptor here. For more details on Kiosk Session Descriptors and the use of well-known names to locate them, see **kiosk(5)**.

KIOSK_SESSION_APPLAUNCHER
Identifies the absolute path to a Kiosk session specific executable used to launch applications added to the session.

KIOSK_SESSION_APPLIST
Identifies a Kiosk Session Application List to be used. You may specify either the well-known name of or absolute path to a Kiosk session application list. For more details on Kiosk Session Application Lists and the use of well-known names to locate them, see **kiosk(5)**.

KIOSK_SESSION_ARGS
Identifies the command line arguments to be passed to the Kiosk session executable when the session is launched.

KIOSK_SESSION_DIR
Identifies a directory reserved for use by the Kiosk session.

KIOSK_SESSION_EXEC
Identifies the absolute path to the Kiosk session executable.

KIOSK_SESSION_ICON
Identifies an icon that can be used to represent the session in an administration tool.

KIOSK_SESSION_LABEL
Identifies a label that can be used to represent the session in an administration tool.

KIOSK_SESSION_LOCALE
Identifies the locale to be used by the Kiosk session. The default value is the default system locale.

KIOSK_SESSION_LIMIT_CPU
Identifies the maximum cpu usage (in seconds) per process for the Kiosk session. The default value is the default system limit. For more details, see **ulimit(1)**.

KIOSK_SESSION_LIMIT_DESCRIPTORS
Identifies the maximum number of open file descriptors per process for the Kiosk session. The default value is the default system limit. For more details, see **ulimit(1)**.

KIOSK_SESSION_LIMIT_FILESIZE
Identifies the maximum file size (512 byte blocks) for the Kiosk session. The default value is the default system limit. For more details, see **ulimit(1)**.

KIOSK_SESSION_LIMIT_VMSIZE
Identifies the maximum swap size (in KB) per process for the Kiosk session. The default value is the default system limit. For more details, see **ulimit(1)**.

KIOSK_SESSION_POST
Identifies the absolute path to an executable which will be executed immediately after the Kiosk session executable (**KIOSK_SESSION_EXEC**) terminates.

KIOSK_SESSION_PRE
Identifies the absolute path to an executable which will be executed immediately before the Kiosk session executable (**KIOSK_SESSION_EXEC**) starts.

KIOSK_SESSION_PROTOTYPE
Identifies a Kiosk Session Prototype which should be applied to the Kiosk user's home directory before launching the Kiosk session. You may specify either the well-known name of or absolute path to a Kiosk Session Prototype here. For more details on Kiosk Session Prototypes and the use of well-known names to locate them, see **kiosk(5)**.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWkio
Interface Stability	Committed

SEE ALSO

kiosk(5), **kioskconfig(1M)**, **ulimit(1)**

NAME

sunray - Sun Ray virtual device driver.

SYNOPSIS

/dev/sunray

DESCRIPTION

The **/dev/sunray** file refers to a pseudo-device driver that provides frame-buffer compatible information for configuring the **Xsun**(1) X server. The **sunray** driver's only function is to properly respond to the VIS_GETIDENTIFIER **ioctl**(2).

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto

SEE ALSO

visual_io(7I)

NAME

uninstaller - Removes Sun Ray Connector software.

SYNOPSIS

uninstaller [-a *admin*]

DESCRIPTION

uninstaller removes the Sun Ray Connector 2.0 software from the system.

OPTIONS

The following options are supported.

-a *admin*

An alternate admin(4) file can be specified with *admin*. Solaris only option.

FILES

/opt/SUNWuttsc/etc/admin_default is the default admin file.

EXIT STATUS

0 if operation performed successfully, 1 if error found

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuttsc

SEE ALSO

NAME

utaction - Sun Ray DTU connect/disconnect action

SYNOPSIS

/opt/SUNWut/bin/utaction [**-c** *ccmd*] [**-d** *dcmd*] [**-e**] [**-D** *display*] [**-i**] [**-t** *sec*]

DESCRIPTION

The **utaction** program provides a way to execute commands when a Sun Ray DTU session is connected or disconnected. The *ccmd* is invoked using **sh**(1) whenever the session is connected to a DTU. Similarly, the *dcmd* is invoked using **sh**(1) whenever the session is disconnected from a DTU. Normally, action is not taken on the initial state of the session (when **utaction** is first run) unless the -i option is used.

Note: In earlier releases, this command resided in **/opt/SUNWut/lib/utaction** ; now, however, it resides in **/opt/SUNWut/bin/utaction** .

OPTIONS

The following options are supported.

-c *ccmd*
: Run this command when the current session is connected to a DTU.

-d *dcmd*
: Run this command when the current session is disconnected to a DTU.

-D *display*
: This option will set the X display variable that is to be used in determining the Sun Ray DTU session. Otherwise the DISPLAY environment variable is used.

-e This option causes utaction to exit after encountering a command.

-i Run the connect or disconnect command immediately, whichever is appropriate.

-t *sec* This option specifies a time-delay in seconds for the actions. In that case, the *ccmd* or *dcmd* will not be invoked unless the session remains in the connected or disconnected state, respectively, for at least *sec* seconds.

EXAMPLES

Example 1: This command invokes the CDE screen lock whenever the session is disconnected

% **utaction -d '/usr/dt/bin/dtaction LockDisplay' &**

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto

NOTES

The *ccmd* and *dcmd* are each only one argument to **utaction**. Quotes should be used if a command contains multiple words.

NAME

utadem - Sun Ray audio driver emulator.

SYNOPSIS

/dev/utadem

DESCRIPTION

utadem provides a generic virtual audio interface to Sun Ray DTU's. The actual interface to the DTU is through a daemon that is session-aware. The daemon connects to **utadem** through a master port and is responsible for creating the slave device nodes which connect to normal audio applications.

API

The exact capabilities of the audio device emulated depend on the OS and audio daemon.

Solaris Applications that normally open **/dev/audio** may use **utadem** as long as they have some way of selecting the audio device, such as through the **-d** *device* switch, or the AUDIODEV environment variable. Compliance to the standard **audio**(7I) interface is handled in the following manner:

Audio Data Formats
: The data formats supported depend on the daemon. Please refer to the daemon documentation for its capabilities.

Audio Ports
: Input and output audio ports are directly dependent on the Sun Ray DTU and not on the daemon. The daemon is capable of discovering the type and quantity of input ports available and report them in the **record.avail_ports** and **play.avail_ports** fields of the **audio_info** structure. Although the ports can be controlled directly, the actual audio output is generally a mix of multiple services, so the **play.gain** setting is the contribution of this audio device to the total experience. Since recording is exclusive of a single service, the **record.gain** and **record.balance** controls directly affect the hardware gain.

Sample Granularity
: Since the **utadem** driver is working through a daemon which transfers the audio data over an interconnect, larger granularities and jitter in the reporting of sample counts is possible. At any given time, the reported input and output sample counts will vary from the actual sample count by no more than the size of the buffers it is transferring. Programs should not rely upon the absolute accuracy of the **play.samples** and **record.samples** fields of the **audio_info** structure.

Audio Status Change Notification
: As described in **audio**(7I), it is possible to request asynchronous notification of state changes in an audio device.

Linux Compliance to the standard Open Sound System (OSS) interface is handled in the following manner:

Audio Data Formats
: Same as Solaris.

Audio Ports
: Input and output audio ports are directly dependent on the Sun Ray DTU and not on the daemon. The daemon is capable of discovering the type and quantity of input ports available. Use the OSS mixer interface to deter- mine which ports are available

Sample Granularity
: Same as Solaris.

The following OSS features are unsupported:

SNDCTL_DSP_REALTIME
Realtime capabilities are not supported.

SNDCTL_CAP_MMAP
Device DMA memory mapping is not supported.

/dev/audio
/dev/audio interface is not currently supported. Use /dev/dsp with appropriate format settings instead

ERRORS

Solaris **utadem** errors are defined in the **audio(7I)** man pages. If the daemon has exited, further audio operations are no longer possible on the slave ports. Audio programs must exit in order to clear this error. New opens will return ENODEV. Data writes and **ioctl** operations will return ENXIO. Data reads will complete normally and then return end-of-file.

Linux are there any?

FILES

The following file is used:

- **/dev/utadem**

Master port for daemons.

The logical device name of the slave port depends on the daemon.

ATTRIBUTES

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

Attribute Type	Attribute Value
Solaris	
Availability	SUNWutu
MT-Level	Safe
Linux	
Availability	SUNWutkau

SEE ALSO

Solaris and Linux
utaudio(1), **ioctl(2)**

Solaris **attributes(5)**, **audio(7I)**, **streamio(7I)**

NAME

utadm - Sun Ray network and DHCP configuration utility.

SYNOPSIS

/opt/SUNWut/sbin/utadm -a *interface-name*

/opt/SUNWut/sbin/utadm -c

/opt/SUNWut/sbin/utadm -d *interface-name*

/opt/SUNWut/sbin/utadm -f

/opt/SUNWut/sbin/utadm -l

/opt/SUNWut/sbin/utadm -n

/opt/SUNWut/sbin/utadm -p

/opt/SUNWut/sbin/utadm -r

/opt/SUNWut/sbin/utadm -x

/opt/SUNWut/sbin/utadm -A *subnetwork*

/opt/SUNWut/sbin/utadm -D *subnetwork*

/opt/SUNWut/sbin/utadm -L on | off

DESCRIPTION

The **utadm** command manages the network and DHCP configuration for a dedicated Sun Ray interconnect or a shared subnet on a LAN. It configures the name lookup, host, network, netmask, and DHCP database files so that Sun Ray DTUs can be connected to a central server host over one or more private interconnects or shared subnets. One of the following option flags must be specified: **-a**, **-c**, **-d**, **-f**, **-l**, **-n**, **-p**, **-r**, **-x**, **-A**, or **-D**. The command is run with superuser privileges.

OPTIONS

The following options are supported.

-a Configure the network interface specified by *interface-name* as a dedicated interconnect. In the default case, an available private address is selected from the range 192.168.128.0 to 192.168.254.0. If the subnet selected is 192.168.*N*.0, entries for the hosts, networks, and netmasks files are generated using the *hostname* of the server and *interface-name*:

File	Entry		
/etc/hosts	192.168.N.1	hostname-interface-name	
/etc/networks	SunRay-interface-name	192.168.N.0	SunRay
/etc/netmasks	192.168.N.0	255.255.255.0	

Once the appropriate entries are established, the network interface is activated as hostname-interface-name using **ifconfig(1M)**. If the interface is already up and configured, the user will be given the option to bypass configuration of the network interface and only configure DHCP on the interface. This allows configuration of a Sun Ray interconnect on the primary interface of the server. IP addresses on the Sun Ray subnets are managed using the DHCP protocol, which requires the addition of several macro entries to the DHCP table to control parameters on Sun Ray subnets. A pool of IP addresses for assignment to Sun Ray DTUs is created. It is possible to disable offering IP addresses by entering 0 as the first unit address or entering N to the query to offer IP addresses for the interface when prompted. Once the interface is configured and activated, **utfwadm(1M)** is invoked to add the current version of the firmware to the DHCP macros for the new network. The user is prompted for approval of all the default options, and may change them as desired. The auth server list is the list of IP addresses of Sun Ray servers. The value input overrides the default value prompted for. The auth server list can also be provided in a file as a list of IP addresses separated by spaces.

The **-a** option implies the **-c** option if the initial configuration has not yet been performed.

-c Initialize the basic configuration files for a Sun Ray interconnect or shared subnet without setting up any networks. This involves making sure that the network database files and framework for DHCP

exist, and setting the **/etc/nsswitch.conf** file so that network information for the local Sun Ray interconnect or shared subnets is obtained from local files.

-d Delete the network interface specified by *interface-name* from the list of configured Sun Ray private interconnects. The specified interface must have been previously configured using the **-a** option.

-f Take this server offline, preventing the creation of new sessions on this server when it is within a failover group. Existing sessions are not killed, but load balancing does not select this server for new sessions. Note: The GroupManager updates the host interface list (if there have been any changes) after every keep alive interval. So in order for the changes to be effective, the user has to wait at least the keepalive time interval.

-l Print the current configuration for all the Sun Ray dedicated interconnects or shared subnetworks. This includes remote subnetworks.

-n Bring this server back online. This restores normal operation of the server and allows new sessions to be created on this server. Note: The GroupManager updates the host interface list (if there have been any changes) after every keep alive interval. So in order for the changes to be effective, the user has to wait at least the keepalive time interval.

-p Print the current Sun Ray interconnect configuration, showing for each interface the hostname, network, netmask, and number of IP addresses out of an available pool of addresses assigned to Sun Ray DTUs by DHCP.

-r Unconfigure all active Sun Ray interfaces and remove all Sun Ray entries from the configuration databases. If LAN connection support is on, will prompt the user if it should be turned off. Default is yes (turn it off).

-x Print the current configuration in a machine-readable format

-A Configure the *subnetwork* specified as a Sun Ray subnetwork. This option only configures the DHCP service to allocate IP address and/or to provide Sun Ray parameters to Sun ray clients. It also will automatically turn on support for LAN connections from a shared subnetwork.

-D Delete the *subnetwork* specified form the list of configured Sun Ray subnetworks.

-L on| off
Turn support for LAN connections from a shared subnetwork on or off.

EXAMPLES

Example 1: The following example configures the Sun Ray private network on hme1

/opt/SUNWut/sbin/utadm -a hme1

FILES

The following files are used:

- **/etc/nsswitch.conf**

Name service switch configuration file.

- Solaris: **/var/dhcp/dhcptab** Linux: **/etc/dhcp.conf**

Solaris files or NIS+ table

- **/etc/inet/hosts**
- **/etc/inet/networks**
- **/etc/inet/netmasks**
- **/etc/hostname.***

Linux DHCP service configuration files

- **/etc/sysconfig/dhcpd**
- **/var/lib/dhcp/dhcpd.leases**
- **/etc/sysconfig/network-scripts/ifcfg - <interface_name>**

Interface information for Red Hat AS 3.0

- **/etc/sysconfig/network/ifcfg - <interface_name>**

Solaris: Hostname for each interface

Linux: Interface information for JDS and SuSE

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto

SEE ALSO

Solaris:

dhtadm(1M), **pntadm(1M)**, **dhcpconfig(1M)**, **dhcp(4)**, **dhcp_network(4)**, **dhcptab(4)**, **networks(4)**, **netmasks(4)**

Solaris and Linux:

ifconfig(1M), **syslogd(1M)**, **syslog(3)**, **nsswitch.conf(4)**, **hosts(4)**, **syslog.conf(4)**, **attributes(5)**, **utfwadm(1M)**

Alexander, S., and Droms, R., DHCP Options and BOOTP Vendor Extensions, RFC 1533, Lachman Technology, Inc., Bucknell University, October 1993.

Droms, R., Dynamic Host Configuration Protocol, RFC 1541,Bucknell University, October 1993.

NAME

utadmin.conf - Sun Ray server administration configuration file.

SYNOPSIS

/etc/opt/SUNWut/utadmin.conf

DESCRIPTION

The **utadmin.conf** file is a contains configuration parameters for the Sun Ray server administration database.

The **admin.defaultlocale** parameter (see below) is the only parameter that should be changed once the Sun Ray server is configured and in use. All other parameters are reserved.

PROPERTIES

The supported configuration parameters are listed below. For each one, the name, description, and an example are given.

Name Description

admin.defaultlocale

The default locale for Web-based Administration Tool. Supported values:

Solaris "en_US" (US English), "fr" (French), "ja" (Japanese), "zh" (Simplified Chinese).

Linux "en_us" (US English), "fr_FR" (French), "ja_JP.eucjp" (Japanese), "zh_CN" (Simplified Chinese).

admin.dstatus.dbfile

The name of the NDBM data files where the desktop status is stored.

admin.http.cfile

Configuration file for the Sun Ray Administration Server.

admin.http.port

The webserver port used by the Administration Tool.

admin.server.name

The name of the server where the administration database LDAP server process is running. This is usually the host name of the Sun Ray server.

admin.subtree

The subtree within the LDAP hierarchy where Sun Ray administration data for this server resides. This is an entry under the UT root entry that was specified by **utconfig**.

admin.user.name

The LDAP user that the administration clients should bind as to perform privileged operations.

admin.ustatus.dbfile

The name of the NDBM data files where the user status is stored.

EXAMPLES

Example 1: Configuration parameters for the LDAP and NDBM databases:

Solaris and Linux

```
admin.server.name      = sray-139
admin.server.port      = 7012
admin.user.name        = cn=utadmin,utname=sray-139,o=v1,o=utdata
admin.subtree          = utname=sray-139,o=v1,o=utdata
admin.defaultlocale    = en_US
admin.dstatus.dbfile   = /var/opt/SUNWut/ndbm/dstatus
admin.ustatus.dbfile   = /var/opt/SUNWut/ndbm/ustatus

admin.http.port        = 1660
admin.ssl.enable       =
```

Solaris

admin.http.cfile = /usr/apache/httpd.conf

Linux

admin.http.cfile = /usr/apache/conf/httpd.conf #(on JDS only)
admin.http.cfile = /etc/httpd/conf/httpd.conf #(on RHAS only)

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWutr

SEE ALSO

utinstall(1M), **utconfig(1M)**, **utuser(1M)**, **utdesktop(1M)**

Sun Ray Server Software 4.0 Administrator's Guide

NAME

utadminuser - Command used to manage authorized user list to administer the Sun Ray services through the Admin GUI.

SYNOPSIS

/opt/SUNWut/sbin/utadminuser

/opt/SUNWut/sbin/utadminuser -h

/opt/SUNWut/sbin/utadminuser -a *username* [*username* ...]

/opt/SUNWut/sbin/utadminuser -d *username* [*username* ...]

/opt/SUNWut/sbin/utadminuser -r

DESCRIPTION

This command is used to add, list and delete authorized users from the list which is stored in the datastore.

When invoked without any option, it prints the list of authorized users, one name per line.

The *username* must be a string consisting of characters from the set of alphabetic characters, numeric characters, period (.), underscore (_), and hyphen (-). The first character should be alphabetic.

When the Administration GUI is configured to use the default Sun Ray Datastore Authentication mechanism, only the "admin" user is supported and it is added automatically to the list of authorized users by utconfig. When the Administration GUI is configured to use the Unix Login Authentication mechanism, you can add any valid Unix username to the authorized user list.

Only super-user can add and delete entries.

OPTIONS

The following options are supported.

-a Adds specified users to the list

-d Deletes specified users from the list

-r Removes all authorized users

-h Prints the usage

EXAMPLES

Example 1: List all current authorized users:

/opt/SUNWut/sbin/utadminuser

Example 2: add a specific user "user1" into the list

/opt/SUNWut/sbin/utadminuser **-a** user1

Example 3: removes all authorized users.

/opt/SUNWut/sbin/utadminuser **-r**

EXIT STATUS

0 if operation performed successfully 1 if error found

SEE ALSO

useradd(1M), **utconfig**(1M)

NAME

utamghadm - Command used to configure Automated Multi-Group Hotdesking (AMGH), also known as Regional Hotdesking.

SYNOPSIS

```
/opt/SUNWut/sbin/utamghadm
/opt/SUNWut/sbin/utamghadm -d
/opt/SUNWut/sbin/utamghadm -l <library>
/opt/SUNWut/sbin/utamghadm -s <executable>
/opt/SUNWut/sbin/utamghadm -x
```

DESCRIPTION

This command is used by administrators to configure or disable the AMGH feature. AMGH can be configured with backend implementations which are either executable scripts or shared libraries, depending on the site specific implementation. This command provides a means to set the appropriate backend to be used. Setting up the backend implementation implicitly enables AMGH.

The scripts/libraries used to configure AMGH by this command must be owned by root only and should not have write permissions for others or group.

Only super-user can change the AMGH configurations using this command.

OPTIONS

The following options are supported.

-d Disable AMGH configurations

-l Specify the library to be used as the AMGH backend implementation. This should be the complete path to the library.

-s Specify the script to be used as the AMGH backend implementation This should be the complete path to the script.

-h Prints the usage

-x Prints the AMGH parameters in machine readable output (key/value form).

EXAMPLES

Example 1: Display current AMGH configuration

/opt/SUNWut/sbin/utamghadm

Example 2: Configure AMGH to use the script backend called utamghref_script which is available in /opt/SUNWutref/amgh directory.

/opt/SUNWut/sbin/utamghadm **-s** /opt/SUNWutref/amgh/utamghref_script

Example 3: Configure AMGH to use the libray backend called libutamghref_token.so which is available in /opt/SUNWutref/amgh directory.

/opt/SUNWut/sbin/utamghadm **-l** /opt/SUNWutref/amgh/libutamghref_token.so

Example 4: Disable AMGH.

/opt/SUNWut/sbin/utamghadm **-d**

EXIT STATUS

0 if operation performed successfully 1 if error found

SEE ALSO

ut_amgh_get_server_list(3), **ut_amgh_script_interface**(3)

NAME

ut_amgh_get_server_list, ut_amgh_free_server_list - APIs for Automated Multi-Group Hotdesking also known as Regional Hotdesking.

SYNOPSIS

```
cc [flag ...] file ... [library ...]

#include <utamgh.h>

int ut_amgh_get_server_list(const struct ut_amghargs *amghargs, struct ut_amghret
int ut_amgh_free_server_list(struct ut_amghret *amghret);
```

DESCRIPTION

The **ut_amgh_get_server_list**() function is called for retrieving a list of servers associated with the input parameters for the purposes of Automated Multi-Group Hotdesking. The underlying AMGH PAM services will invoke this function to obtain a list of servers which the Sun Ray should be redirected to. The first server in the list which is alive and online will be utilised.

The ut_amghargs structure has the following entries:

```
struct ut_amghargs {
  int  amghversion;     /* the version of the customer API */
  char *username;       /* the user name e.g. 'fred'*/
  char *token;          /* the token id e.g. 'Payflex.50045fda00130100'*
  char *terminal_cid;   /* the canonical terminal id e.g. 'IEEE802.08002
  char *terminal_ip_addr; /* the terminal ip address e.g. '10.23.146.12'
  char *insert_token;   /* the token id inserted  e.g. 'Payflex.50045fda
  char *display;        /* the display number e.g. ':12'*/
};
```

The ut_amghret has the following entries:

```
struct ut_amghret {
  char **server_list;   /* the reference to a server array */
  char *username;       /* iff the implementation returns one else NULL
  char *errorstr;       /* the error message */
  int  chain_amgh;      /* RESERVED for future use */
  int  use_firstserver; /* Use firstServer prop as the host */
};
```

The amghversion field is filled by the underlying PAM service before **ut_amgh_get_server_list**() is called. **ut_amgh_get_server_list**() should verify that amghversion is >= UT_AMGH_VERSION to protect itself from being deployed on old and potentially incompatible versions of AMGH. If the test fails it should return UT_AMGH_INCOMPATIBLE_VERSION before accessing other fields in the ut_amghargs or ut_amghret structures.

ut_amgh_get_server_list() may use the values of the ut_amghargs structure to determine the list of hosts to return to the AMGH service. It should set server_list to point to storage containing an array of pointers to fully-qualified hostnames or IP address strings expressed in the IPv4 standard '.' notation, with a NULL pointer as its last element. If no suitable hosts are found, **ut_amgh_get_server_list**() should assign NULL to the server_list field in the structure ut_amghret. If an error is encountered it should set errorstr in the ut_amghret structure to a suitable error string, set server_list to NULL and return one of the valid error codes listed below. **ut_amgh_get_server_list**() may allocate dynamic storage for use in the ut_amghret structure, which will be freed by a later call to **ut_amgh_free_server_list**().

ut_amgh_get_server_list() may also return a username. If a NULL username is supplied by the PAM service and a username is returned it will be used to initialize PAM_USER in the PAM context.

ut_amgh_free_server_list() frees up any memory dynamically allocated in the ut_amghret structure. It is

the responsibility of the calling application to invoke **ut_amgh_free_server_list**() to free the memory returned by **ut_amgh_get_server_list**().

RETURN VALUES

Upon successful completion, UT_AMGH_SUCCESS must be returned. In addition, the following values may be returned :

UT_AMGH_INCOMPATIBLE_VERSION
The version filled in struct ut_amghargs field amghversion by the customer is not compatible with UT_AMGH_VERSION value.

UT_AMGH_NO_PERMISSION
No permission to access the back-end database for the server lookups.

UT_AMGH_DB_ERROR
Error accessing the back-end database.

UT_AMGH_NULL_USERNAME
The implementation requires a username to perform its lookup, but the AMGH system was not configured to pass in a username and hence the ut_amghargs username field was NULL.

UT_AMGH_ERROR
Unknown error encountered by the implementation.

UT_AMGH_NOMEM
Unable to allocate memory.

UT_AMGH_USERNAME_CONFLICT
More than one non-NULL user names found for the token.

NOTES

A reference Makefile and source code for a file-based implementation of the AMGH APIs are available under /opt/SUNWutref/amgh/, as a learning aid for potential implementors.

SEE ALSO

ut_amgh_script_interface(3)

NAME

ut_amgh_script_interface - executable API for Automated Multi-Group Hotdesking (also known as Regional Hotdesking)

SYNOPSIS

```
<user_defined_executable_name> < input > output

input:
[username=username]
[token=token]
[terminal_cid=macaddr]
[terminal_ip_addr=dtu_IP]
[insert_token=token]
[display=display_number]

output:
[username=username]
[host=host]
[use_firstserver=true/false]
```

DESCRIPTION

The Sun Ray Server Software Automated Multi-Group Hotdesking (AMGH) system utilizes an API to obtain information about a username and/or a set of servers for a Sun Ray to be directed to. See the ut_amgh_get_server_list man page for more details.

AMGH also can be configured to run a customer-supplied executable to produce information for this API. This man page documents the interface that such an executable should implement.

The executable will be executed with no arguments. On its standard input stream (see **intro**(3)), stdin, data representing the ut_amghargs structure contents will be written. This data will be in the form of key/value pairs, one per line on stdin. The key name will be from the beginning of the line to the first character and continue to the next newline.

Valid lookup keys are :

```
username         /* the user name  e.g. 'fred' */
token            /* the token id   e.g. 'Payflex.50045fda00130100'*/
terminal_cid     /* the canonical terminal id  e.g. 'IEEE802.0800201e2347'*/
terminal_ip_addr /* the terminal ip address  e.g. '10.23.146.12'*/
display          /* the display number e.g. ':12'*/
insert_token     /* the token id inserted  e.g. 'Payflex.50045fda00130100'*/
                    /* This is useful in cases where inserted token is different
                    /* from the session token, for eg when the registered policy
                    /* is in effect */
```

Lines containing keys which are unrecognized should simply be discarded from stdin and the executable should continue parsing the remaining input until EOF is reached.

At that point the executable should perform its lookup. If a username is to be returned, it should write a key/value to standard output (stdout) of the form:

username=USERNAME

where USERNAME is the value returned by the lookup. It may also return any number of lines of the form:

host=HOST

where each HOST represents either a fully-qualified hostname or an IP address.

The script could also return the following values :

use_firstserver=[true | false]

The usage of these options is beyond the scope of this man page and will be discussed in details in the software documentation. These are documented here for completeness. The return values for these options can also be a non-zero number (true) or 0 (false).

ERRORS

If an error is encountered during the lookup, the executable may write a descriptive error message to standard error (stderr), which will be included in the error log and exit with a suitable error code (see EXIT VALUES).

EXIT VALUES

Exit values should correspond to the error codes in /opt/SUNWut/include/utamgh.h.

If no errors are encountered during the lookup, the executable should exit with a status of 0 (UT_AMGH_SUCCESS).

Note that if the lookup does not produce a match, this is not considered an error. In this case, no stdout should be generated, and the executable should exit with a status of 0 (UT_AMGH_SUCCESS).

If the executable exits with a non-zero exit status, an error will be logged corresponding to the value of the exit status.

SEE ALSO

ut_amgh_get_server_list(3)

NAME

utaudio - Sun Ray audio services connection utility.

SYNOPSIS

/opt/SUNWut/bin/utaudio

csh

setenv AUDIODEV ‘utaudio‘

ksh

export AUDIODEV=‘utaudio‘

sh

AUDIODEV=‘utaudio‘;export AUDIODEV

DESCRIPTION

utaudio enables standard Solaris audio services using the **utadem**(7D) audio device emulator driver. After connecting to a Sun Ray session, **utadem**(7D) creates a new audio device for which **utaudio** creates device files in the **/tmp/SUNWut/dev** directory. **utaudio** then echoes the root device name to standard output, setting the AUDIODEV environment variable. Standard audio applications can then open the new audio pseudo-device and perform audio play and record operations.

OPTIONS

There are no options for **utaudio**.

API

Applications that use the **/dev/audio** interface may open the device indicated by the AUDIODEV environment variable and use the AUDIO_GETDEV **ioctl** to determine which audio device is being used. The **utaudio** driver returns the string "SUNW,CS4231" in the name field of the **audio_device** structure to indicate compatibility with other Ultra platforms. The version field contains "a" and the config field contains "pseudo."

The AUDIO_SETINFO **ioctl** controls device configuration parameters. When an application modifies the **record.buffer_size** field using the AUDIO_SETINFO **ioctl**, the daemon will constrain it to be non-zero and up to a maximum of 8180 bytes.

Audio Data Formats

The **utaudio** daemon supports u-law and A-law with 8-bit precision or 16-bit linear PCM at any sample rate from 8000 Hz to 48 kHz for one or two channels. The Sun Ray standard sampling rate is 48 kHz as this yields the best quality. The input and output data formats for playing and recording do not have to match. Some input devices do not provide 2-channel capture, but two channels will be reproduced by duplication in the case where two channels are requested and the device supports only one.

Audio Ports

The **record.avail_ports** and **play.avail_ports** fields of the **audio_info** structure report the available input and output ports for the currently connected Sun Ray DTU. Only AUDIO_MICROPHONE and AUDIO_LINE_IN are supported and most devices will have both inputs. The Sun Ray audio model supports individual volume controls for the two, so it is possible that the volume setting will change with input.

For output, AUDIO_LINE_OUT is always selected and does not have variable gain. AUDIO_SPEAKER and AUDIO_HEADPHONE are supported and they share a level control. In general, comfortable settings for the speaker will also be comfortable for headphone use. Either or both outputs can be selected simultaneously. The Sun Ray specification supports a third, automatic switching mode that is accessed by deselecting both speaker and headphone or by selecting only line out. The **utsettings**(1) command may also be used to control the device’s outputs. In automatic mode, the settings track the physical connection of the headphone.

FILES

The following files are used:

- **/tmp/SUNWut/dev/utaudio/***n*

Numbered audio data pseudo-device file nodes.

- **/tmp/SUNWut/dev/utaudio/***n*ctl

Matching numbered control pseudo-device file nodes.

ENVIRONMENT VARIABLES

utaudio requires the DISPLAY environment variable contain an **X11**(7) display for which the user's session has access. This is set-up automatically in the Sun Ray environment.

An alternate driver emulator or different unit number can be specified in the UT_ADEM environment variable.

The results of **utaudio** should be placed in the AUDIODEV environment variable.

EXIT STATUS

The following exit values are returned:

0 Normal completion -- daemon back grounded

1 Either the X11 server, or the session could not be contacted, or there was a problem creating the new pseudo audio device.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto

SEE ALSO

utsettings(1), **X11(7)**, **utadem(7D)**, **audio(7I)**, **steamio(7I)**, **ioctl(2)**, **priocntl(2)**, **attributes(5)**, **environ(5)**

NOTES

The **audio**(7I) interface does not have an interface for dynamically changing audio devices such as that offered by the Sun Ray software. It is not possible to track the movement of sessions or changes in audio hardware using this device interface. The **utaudio** daemon makes a best-effort attempt to report changes in device control ability and to make the device appear as flexible as possible, matching that ability to the actual Sun Ray hardware being used.

If a session is disconnected, audio output continues as if there was an actual hardware connection, even though no samples are actually being transmitted or played. Conversely, audio input stops when there is no connected device.

NAME

utauthd - Sun Ray DTU authentication daemon.

SYNOPSIS

/opt/SUNWut/lib/utauthd -b | **-e** | **-n** | **-s**

DESCRIPTION

The **utauthd** daemon is responsible for authentication and access control for the Sun Ray DTUs attached to a server. This command should not be executed directly. It is invoked by a system startup script.

OPTIONS

The following options are supported.

-b Begin execution of the daemon.

-e End execution of the daemon.

-n Number of file descriptors to make available

-s Signal to send to utauthd

Without arguments, the default is **-b**.

FILES

The following files are used by the daemon:

- **/etc/init.d/utsvc**

This is the system startup script that invokes the daemon. **/opt/SUNWut/utsessiond**, the Session Manager, performs the actual session switching function.

- **/etc/opt/SUNWut/auth.props**

The Authentication Manager's configuration file.

- **/etc/opt/SUNWut/policy/utpolicy**

This file determines what policy is used by the Sun Ray server.

To start the authentication manger, it is customary to run **utrestart**; however, to kill existing user sessions and restart services, it is preferable to use **utrestart** -*c*.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto

SEE ALSO

auth.props(4), **utpolicy(1M), utrestart(1M)**

NAME

utcammigrate - Controlled Access Mode (CAM) application migration utility.

SYNOPSIS

/opt/SUNWut/sbin/utcammigrate [**-u**] [**-t**] [**-d** *directory*] **/opt/SUNWut/sbin/utcammigrate** **-h**

DESCRIPTION

The utcammigrate utility may be used to migrate existing CAM configuration to its Kiosk Mode equivalent with the intention of migrating from existing CAM CDE sessions to Kiosk CDE sessions. This migration includes the creation of Kiosk application descriptors, prototypes, session configuration and application lists. The migration doesn't include support for CDE wrapper scripts.

Without any arguments utcammigrate will migrate all available CAM configuration to Kiosk mode equivalents i.e. Kiosk mode session configuration and application descriptors, prototypes and lists corresponding to available CAM configureation will be created.

CAM Configuration

You may download and view the CAM configuration to be migrated using the command **/opt/SUNWut/sbin/utkiosk -e kiosk**. If you need to modify CAM configuration prior to migration, you may do so using the following steps:

- download the configuration to a temporary file using the command **/opt/SUNWut/sbin/utkiosk -e kiosk > /tmp/kiosk.conf**
- apply your modifications to the temporary file using any text editor
- upload the modified configuration using the command **/opt/SUNWut/sbin/utkiosk -i kiosk -f /tmp/kiosk.conf**

Note on dtsession

CAM and the CDE Kiosk session treat dtsession differently. CAM includes dtsession by default as a CAM application and dtsession may be removed from a given CAM configuration. The CDE Kiosk session, on the other hand, includes, by default, the launching of dtsession from its session script and, as such, dtsession never needs to be added as a Kiosk application. For this reason, the CAM dtsession application is excluded from the various migration activities described below. If a CAM configuration not containing the dtsession application is migrated, utcammigrate will configure the resulting Kiosk session to ensure that it does not launch dtsession. Additionally, if you have modified the CAM dtsession prototype and wish to use your modifications with the CDE Kiosk session, you will need to manually reapply your modifications to the new dtsession prototype or add them via a Kiosk application.

Migration of CAM applications and prototypes

For each CAM application defined in the Sun Ray administration database, utcammigrate will create an equivalent Kiosk application descriptor **/etc/opt/SUNWkio/applications/***name***.conf** where *name* is the Application Profile Name associated with the CAM application. If a descriptor named *name* already exists, utcammigrate will use the name **migrated.***name* instead.

Additionally, any available prototype associated with a migrated CAM application will be migrated and available for use in Kiosk Mode. A prototype for a CAM application named *name* available in **/var/opt/SUNWut/kiosk/prototypes/***name* will be migrated to **/etc/opt/SUNWkio/prototypes/***name*. If a prototype named *name* already exists, utcammigrate will use the name **migrated.***name* instead.

Kiosk session configuration and application list

utcammigrate will automatically create a Kiosk session configuration file, **/etc/opt/SUNWkio/cam-session.conf**, identifying the Kiosk CDE session as the selected session.

Additionally, utcammmigrate will create a Kiosk application list file,

/etc/opt/SUNWkio/applications/cam-applications.list, identifying all CAM applications enabled in the Sun Ray administration database. If no enabled applications are available, the application list file will not be created.

Once migration is complete, utcammigrate will automatically upload the resulting Kiosk session configuration file and application list to the Sun Ray administration database if the **-u** option is specified. You may also manually upload the configuration at any time by executing the following command:

/opt/SUNWut/sbin/utkiosk -i session -f /etc/opt/SUNWkio/cam-session.conf -A /etc/opt/SUNWkio/applications/cam-applications.list

Alternatively, you may use the Sun Ray administration GUI to select the Kiosk CDE session and add available applications.

OPTIONS

The following options are supported:

-u Automatically upload any generated Kiosk session configuration and application list to the Sun Ray administration database.

Note: In a failover group all servers should be migrated before uploading migrated configuration.

-t Create a tar archive, **/etc/opt/SUNWkio/cam-migrated.tar** containing the results of the migration. This tar file is useful for completing the migration task on other systems having the same CAM configuration. The steps to do this are:

- transfer the tar file to the target system
- untar the tar file from the directory **/etc/opt/SUNWkio**
- execute the utkiosk command listed above

-d *directory*

Use the specified directory, instead of **/etc/opt/SUNWkio** for all created files. Use this option to migrate data without installing the result of the migration to the local machine. It is not recommended to use this option with the **-u** option, as the generated session configuration is not valid until the migration results are deployed.

-h Print a usage message and exit.

EXIT STATUS

The following exit values are returned:

0 Success

1 Failure

FILES

/etc/opt/SUNWkio/cam-session.conf

/etc/opt/SUNWkio/cam-migrated.tar

/etc/opt/SUNWkio/applications/cam-applications.list

/etc/opt/SUNWkio/applications/*appname***.conf**

Generated files. Can be relocated by using the **-d** option.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuta
Interface Stability	Uncommitted

SEE ALSO

utkiosk(1M), **kiosk(5)**

NAME

utcapture - Capture packet information from the Authentication Manager.

SYNOPSIS

/opt/SUNWut/sbin/utcapture -h

/opt/SUNWut/sbin/utcapture [**-r**] [**-s** *server*] [*desktopID1 desktopID2 ...*]

/opt/SUNWut/sbin/utcapture -i *filename*

DESCRIPTION

The **utcapture** command connects to the Authentication Manager and monitors latency, packets sent, and packets dropped between the Sun Ray server and the Sun Ray DTU's.

utcapture writes the captured information to **stdout** in the following format:

TERMINALID TIMESTAMP TOTAL PACKET TOTAL LOSS BYTES SENT PERCENT LOSS LATENCY

OPTIONS

The following options are supported.

-h Help for using the command.

-i *filename*

Use an input file to search for Sun Ray DTU's that experienced dropped packets. A file is created using **utcapture**:

 # /opt/SUNWut/sbin/utcapture -r > /tmp/*filename*

The process is allowed to run for several minutes or hours. The utcapture command is used again:

 # /opt/SUNWut/sbin/utcapture -i /tmp/*filename*

The output is only the DTU's that experienced dropped packets.

-r Write captured data to stdout every 15 seconds in raw, continuous format.

-s *server*

Specify the Sun Ray *server* from which to capture data. If outside of the domain of the host running **utcapture**, the Sun Ray server hostname must be fully qualified. By default, the server monitored is the host running **utcapture**.

When no option is specified, **utcapture** writes to **stdout** at 15 second intervals if there is any change in packet loss or latency exceeds 10 ms for any Sun Ray DTU.

OPERANDS

The following operands are supported:

desktopID

Capture data for the specified Sun Ray DTU's only. Appliances are specified by their Ethernet address (*desktopID*) separated by spaces. By default, data for all DTU's is displayed.

EXAMPLES

Example 1: This command captures data from the Authentication Manager running on localhost every 15 seconds and then writes it to **stdout** if there is any change in packet loss for any Sun Ray DTU.

% **utcapture**

Example 2: This command captures data from the Authentication Manager running on localhost every 15 seconds and then writes it to **stdout** regardless if there is any change in packet loss.

% utcapture -r

Example 3: This command captures data from the Authentication Manager running on

netraj118.eng every 15 seconds and then writes it to **stdout** if there is any change in packet loss for DTU's with the Ethernet address of **080020a893cb** or **080020b34231**.

% utcapture -s netraj118.eng 080020a893cb 080020b34231

EXIT STATUS

The following exit values are returned:

0 Successful completion

1 Error

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuta

SEE ALSO

utauthd(1M), **utdesktop(1M)**

NOTES

utcapture does not report packet information for Sun Ray DTU's using firmware versions of 1.1 or less.

utcapture does not report latency for Sun Ray DTU's using firmware versions of 1.3 or less.

When using **-r** 0.000 will listed for PERCENT LOSS as 0.000 for every interval that has no loss. When not using **-r** this column will be blank. If the output is to be processed by commands that are column oriented, you must use **-r**.This includes using it as input to **utcapture -i**.

NAME

utcard - Sun Ray server smart card configuration utility

SYNOPSIS

/opt/SUNWut/sbin/utcard -a *filename*

/opt/SUNWut/sbin/utcard -d *name,version*

/opt/SUNWut/sbin/utcard -h

/opt/SUNWut/sbin/utcard -l

/opt/SUNWut/sbin/utcard -p *name,version*

/opt/SUNWut/sbin/utcard -r *name,version,new-position*

/opt/SUNWut/sbin/utcard -u

DESCRIPTION

The **utcard** command allows configuration of different types of smart cards in the Sun Ray administration database.

The administrator must first place a configuration file for a specific smartcard in the **/etc/opt/SUNWut/smartcard** directory. This file must have a **.cfg** extension. The smartcard definition in the **.cfg** file is added to the LDAP datastore by using the **-a** option. When a smartcard definition is added, it is automatically assigned the last position in the probe order.To modify the probe order, use the **-r** option.

OPTIONS

The following options are supported.

-a *filename*
: Add the card specified within *filename* that is in the **/etc/opt/SUNWut/smartcard** directory

-d Delete the card specified with *name, version*.

-h Show usage information

-l List all configured cards

-p Show the standard properties for the card specified with *name, version*.

-r Reorder the card specified with *name, version, to new-position*.

-u List unconfigured cards available for configuration as determined by the **.cfg** files in **/etc/opt/SUNWut/smartcard**

USAGE

Use this command only on a Sun Ray server that has been configured for administration by the **utconfig** command.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto

SEE ALSO

utconfig(1M)

NAME

utconfig - Sun Ray server software configuration utility.

SYNOPSIS

/opt/SUNWut/sbin/utconfig [**-u**] [**-k**] [**-w**]

DESCRIPTION

The **utconfig** command performs initial configuration of Sun Ray server and supporting administration framework software. Before taking any actions the command prompts the user for configuration parameters for each of the supporting software packages. The command must be run with superuser privileges.

OPTIONS

The following option is supported.

-u Unconfigure the Sun Ray server and administration software returning the mode of operation back to the default zero administration mode.

-k Configure or unconfigure only the Sun Ray Kiosk Mode software. Using this option requires that Sun Ray server and administration software has been configured already.

-w Configure or unconfigure only the Sun Ray Web Administration software. Using this option requires that Sun Ray server and administration software has been configured already.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto

SEE ALSO

Packaging related commands.

utinstall(1M)

NAME

utcrypto - Sun Ray privacy administration utility.

SYNOPSIS

/opt/SUNWut/sbin/utcrypto

/opt/SUNWut/sbin/utcrypto -a *key=value ...*

/opt/SUNWut/sbin/utcrypto -d

/opt/SUNWut/sbin/utcrypto -e [-f *filename*]

/opt/SUNWut/sbin/utcrypto -h

/opt/SUNWut/sbin/utcrypto -m *key=value ...*

/opt/SUNWut/sbin/utcrypto -o [**-f** *filename*]

DESCRIPTION

The **utcrypto** command allows the administrator to configure privacy options for the Sun Ray server. These settings include upstream/downstream encryption and upstream/downstream authentication. These settings will affect all sessions.

utcrypto operations that only display information may be run by any user. Operations that change or delete data must be run as superuser.

OPTIONS

The following options are supported.

-a Add the privacy settings for all sessions. The **-a** option must be followed by a series of *key=value* pairs separated by spaces. Valid *key=value* pairs are described below. If a *value* pair is not passed in, the value for that key will be set to a default value.

At least one key=value pair must be specified.

-d Delete the privacy settings for all sessions.

-e Take the privacy settings in comma-delimited format from **stdin** and add/modify the settings for all sessions. This will be a space-delimited list of *key=value* pairs. If followed by the **-f** option and a *filename*, it will read the settings from a file.

To replicate **utcrypto** settings, run **utcrypto -o** on the "source" server, then use the stdout string as an argument to **utcrypto -e** on the "target" server or servers.

-f Specifies a filename for the **-e** or **-o** option.

-h Displays the usage message.

-m Modify a privacy settings for all sessions. The **-m** option must be followed by a series of *key=value* pairs separated by spaces. Valid *key=value* pairs are described below. If a *value* is not passed in, the value for that key will remain unchanged.

-o Dump all privacy settings in comma-delimited format to stdout. If followed by the **-f** option and a *filename*, the settings will be dumped to the file. If the file specified by *filename* exists, then a warning message is generated and the script exits.

If an asterisk (*) appears in the Inherited column, then no value pair was specified, so it has been set to a hard-coded default value.

To replicate **utcrypto** settings, run **utcrypto -o** on the "source" server, then use the stdout string as an argument to **utcrypto -e** on the "target" server or servers.

The following are valid *key=value* pairs:

enc_up_type
ARCFOUR | **none** | **default**

enc_down_type
ARCFOUR | **none** | **default**

auth_up_type
none | **default**

auth_down_type
simple | **none** | **default**

mode **hard** | **soft** | **default**

The keyword **default** for all keys is set to the value of the default configuration, if it exists. If there a default configuration has not been set, then the value is set to hard-coded defaults. The hard-coded default values for the first four keys are **none** and **soft** for the **mode** key.

EXAMPLES

This adds the configuration for upstream ARCfour encryption and simple downstream authentication.

Note: Since enc_down_typeand **auth_down_type**are not specified, they take on the default values:

/opt/SUNWut/sbin/utcrypto -a enc_up_type=ARCFOUR auth_down_type=simple

This command modifies the default configuration. Upstream encryption is turned off and downstream encryption is set to ARCfour.

/opt/SUNWut/sbin/utcrypto -m enc_up_type=none enc_down_type=ARCFOUR

This command removes the default configuration.

/opt/SUNWut/sbin/utcrypto -d

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto

NAME

utdesktop - Sun Ray DTU administration utility.

SYNOPSIS

/opt/SUNWut/sbin/utdesktop **-a "***desktopID,location,otherinfo*"

/opt/SUNWut/sbin/utdesktop **-a -f** *filename*

/opt/SUNWut/sbin/utdesktop **-d** *desktopID*

/opt/SUNWut/sbin/utdesktop **-d -f** *filename*

/opt/SUNWut/sbin/utdesktop **-e "***desktopID,location,otherinfo*"

/opt/SUNWut/sbin/utdesktop **-e -f** *filename*

/opt/SUNWut/sbin/utdesktop **-h**

/opt/SUNWut/sbin/utdesktop **-l** [**-c** | **-g** | **-w** [**-t** *timeout*]]

/opt/SUNWut/sbin/utdesktop **-L** { **-c** | **-w** [**-t** *timeout*] }

/opt/SUNWut/sbin/utdesktop **-l -i** *substring*

/opt/SUNWut/sbin/utdesktop **-o**

/opt/SUNWut/sbin/utdesktop **-p** *desktopID*

DESCRIPTION

The **utdesktop** command allows the user to manage Sun Ray DTU's connected to the Sun Ray server the command is run on. The information that **utdesktop** displays and allows the user to add, edit, or delete is stored in the Sun Ray administration database. Other information is obtained from the Sun Ray authentication manager.

utdesktop operations that only display information may be run by any user. Operations that add, edit, or delete data must be run by the superuser.

OPTIONS

The following options are supported.

-a *desktopID,location,otherinfo*

Add DTU with the specified desktop-ID, location, and other information properties. Note that the 3 comma-delimited values should be enclosed within quotes. You must be root to use this option.

-a -f *filename*

Batch add multiple DTU's using input from the specified filename. The format of each line in the input file is: desktop-ID,location,other-info. You must be root to use this option.

-d *desktopID*

Delete the DTU with the specified desktop-ID. You must be root to use this option.

-d -f *filename*

Batch delete multiple DTU's using input form the specified filename. The format of each line in the input file is: desktop-ID. You may use the output of the **-o** option to feed this option as all arguments after the first comma are ignored. You must be root to use this option.

-e *desktopID,location,otherinfo*

Edit properties for the specified DTU by changing the location and other information properties to the specified values. Note that the 3 comma-delimited values should be enclosed within quotes. You must be root to use this option.

-e -f *filename*

Batch edit properties for multiple DTU's using input from the specified filename. The format of each line in the input file is: desktop-id,location,other-info You must be root to use this option.

-h Show usage information (this message).

-l List all DTU's currently registered in the admin database.

-l -c List all DTU's that are currently connected to the server (and note any that have been deleted with question marks in the Location field).

-L -c List all DTU's that are currently connected (long format).

-l -g List all currently connected DTU's and the servers they are connected to.

-l -w [**-t** *timeout*]
List all DTU's waiting for a session during the set *timeout* (short format). The default value of the timeout is 60 seconds.

-L -w [**-t** *timeout*]
List all DTU's waiting for a session during the set *timeout* (long format). The default value of the timeout is 60 seconds.

-l -i *substring*
List all DTU's with desktop IDs that contain the specified substring.

-o Dump DTU list in comma-delimited format. The format of each line output by this option is: desktop-id,location,other-info

-p Show desktop properties for the DTU with the specified ID.

EXAMPLES

Example 1: This command clears the location and the other information properties for DTU 080020a85112:

utdesktop -a "080020a85112,,"

Example 2: This command changes the location and the other information properties for DTU 080020a85112 to "SFO12-2103" and "John's Office", respectively:

utdesktop -e "080020a85112,SFO12-2103,John's Office"

Example 3: This command edits the properties of multiple DTU's using input from the file /tmp/desktops:

utdesktop -e -f /tmp/desktops

Example 4: This command displays all DTU's that contain "a851" in their desktop IDs:

% **utdesktop -l -i a851**

Example 5: This command lists all DTU's in an error state without sessions within the default timeout:

% **utdesktop -l -w**

Example 6: For a busy or slow network, this command lists (in long format) all DTU's in an error state without sessions for at least five minutes:

% **utdesktop -L -w -t 300**

Example 7: This command displays the current properties for DTU's 080020a85112:

% **utdesktop -p 080020a85112**

FILES

The following file is used:

/etc/opt/SUNWut/utadmin.conf

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuta

SEE ALSO

utuser(1M), **utadmin.conf(4)**, the *Sun Ray Server Software 4.0 Administrator's Guide*

NOTES

The **-G** option has been deprecated. Use the **-l -g** option pair instead.

NAME

utdetach - Detach the current session from the Sun Ray DTU.

SYNOPSIS

/opt/SUNWut/bin/utdetach

DESCRIPTION

The **utdetach** command disconnects the current session from its respective Sun Ray DTU. The session is not destroyed, but rather put into a disconnected state. The session can be accessed if the same user token is presented to the Sun Ray server.

This command is primarily executed by users of the non-smart card mobility feature so as to disconnect their "mobile" sessions.

The Sun Ray server starts an instance of **utslaunch** (1M) for each session whenever a user logs into a Sun Ray DTU via **dtlogin**. This makes the **utdetach** command available to users as a hotkey sequence. The default hotkey sequence is Shift + Pause and can be configured in the **utslaunch.properties** file.

OPTIONS

There are no options for **utdetach**.

EXAMPLES

Example 1: This command disconnects the current session from the DTU the user is currently using.

% utdetach

FILES

The following files are used:

- **/etc/opt/SUNWut/utslaunch_defaults.properties**

site-wide defaults

- **~/.utslaunch.properties**

user’s defaults

- **/etc/opt/SUNWut/utslaunch_mandatory.properties**

site-wide mandatory defaults

EXIT STATUS

The following exit values are returned:

0 Success

1 Failure

ATTRIBUTES

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

Attribute Types	Attribute Values
Availability	SUNWuto
Stability Level	Evolving

SEE ALSO

utslaunch(1M), **utslaunch.properties(4)**

NAME

utdevadm - Enable or disable Sun Ray device services.

SYNOPSIS

```
/opt/SUNWut/sbin/utdevadm

/opt/SUNWut/sbin/utdevadm [ -e | -d ] -s service

/opt/SUNWut/sbin/utdevadm -h
```

DESCRIPTION

The utdevadm command is used to enable/disable Sun Ray device services. This includes USB devices connected through USB ports, Embedded serial ports and Internal smartcard readers on the Sun Ray DTU.

After Sun Ray Server Software installation, by default all the device services will be enabled. utdevadm command can be used to enable/disable device services only in the configured mode, i.e. it is available only after **utconfig**(1M) has been run to activate Sun Ray data store. This is a site-wide property. When configured, it affects all units connected to the failover group.

It is required to do a cold restart of Sun Ray services before the change can take effect. When a successful change is made, the command reminds the administrator to restart services using the following command: /opt/SUNWut/sbin/utrestart **-c**

utdevadm command with no options or arguments shows the current state of the device services.

OPTIONS

The following options are supported:

-e Enables specific *service*.

-d Disables specific *service*.

-s Specify a *service*.
internal_serial | internal_smartcard_reader | usb | all

When internal_serial *service* is disabled, users will not be able to access embedded serial ports on the Sun Ray DTU. Sun Ray 170 has two embedded serial ports.

When internal_smartcard_reader *service* is disabled, users will not be able to access internal smartcard reader through PC/SC or SCF interface for reading/writing. It does not affect session access, hotdesking using unauthenticated smartcards.

When usb *service* is disabled, users will not be able to access any devices connected to USB ports. It does not affect HID devices such as keyboard, mouse, barcode reader.

Specifying all will enable/disable all device services.

-h Print the usage.

EXAMPLES

Example 1: The following command shows the current state of the device services.

```
# /opt/SUNWut/sbin/utdevadm

Sun Ray Device Service        Status
----------------------------- ------------
internal_serial               enabled
internal_smartcard_reader     enabled
usb                           enabled
```

Example 2: The following command disables usb service.

```
# /opt/SUNWut/sbin/utdevadm -d -s usb
```

Example 3: The following command enables usb service.

```
# /opt/SUNWut/sbin/utdevadm -e -s usb
```

ENVIRONMENT VARIABLES

None

EXIT STATUS

The following exit values are returned:

0 on success

1 on error

FILES

None

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto
Interface Stability	Public Evolving

SEE ALSO

utrestart(1M), **utconfig**(1M)

NAME

utdevmgrd - Sun Ray device manager daemon.

SYNOPSIS

/opt/SUNWut/lib/utdevmgrd [**-a** *authlist*] [**-c** *authfile*] [**-d**] [**-k** *authprops*] [**-o** *optroot*] [**-p** *port*] [**-r**] [**-s** *sigfile*] [**-t**]

DESCRIPTION

The **utdevmgrd** daemon is responsible for brokering devices that are attached to Sun Ray DTU's on the interconnect fabric for the purpose of remotely accessing the devices for various services. It is also responsible for approving services, keeping an inventory of devices and their controlling services, and locating devices on the interconnect.

If either the **-a** or the **-c** option is specified, the device manager daemon operates exclusively in call-back mode. In this mode, the device manager only communicates to authentication managers that are explicitly enabled by *authlist* or *authfile* and that have requested a call-back. The call-back feature provides a mechanism by which the device manager and the authentication manager establish each other's identity.

The *optroot* directory (default **/tmp/SUNWut**) is shared with other Sun Ray server components. Primarily it provides the location for a compatible device tree for each Sun Ray DTU's devices in the **sessions** and **units** subdirectories.

The **units** subdirectory contains a directory for each DTU on the interconnect named by the DTU's serial number. Within an DTU's directory, there are the familiar **dev** and **devices** directories that list logical names for devices and geographically hierarchical names for devices.

The **sessions** directory contains symbolic links into the **devices** directory that indicate which sessions are connected to which Sun Ray DTU's. The symbolic links are named after the X-Windows server display corresponding to a user's session by display number only (in other words, after removing the server name, which is always a name local to the current host, and the screen numbers). The user's **DISPLAY** environment variable can then be used to find the devices on the 'current' DTU. The user's **UTDEVROOT** environment variable achieves this, and can be used to find devices that are 'currently' accessible. The *optroot* directory also includes the named pipe with which the device manager communicates to device driver services and the **session_info** directory, which contains user information important to internal workings of the device manager.

The device manager works within a Sun Ray server group environment, which enables rapid switching to other servers and user load distribution. In order for device managers on each server in a group to communicate, the device manager must gain access to the group signature file. If the signature does not match the one used by other device managers in the group, then grouping will fail and not all devices on all DTU's on the interconnect will be available on the server, including devices on some DTU's being used by users on the server.

Normally, the device manager finds the group signature file by looking into the authentication manager's configuration file (**/etc/opt/SUNWut/auth.props**), but this can be changed by using the **-s** and **-k** options. If **-s** is specified, then *sigfile* is read and used as the group signature. If **-k** is specified, then the *authprops* file is scanned for the **gmSignatureFile** key and the listed file is used for the group signature.

Error messages from **utdevmgrd** are logged using **syslog**(3), with a facility value of LOG_DAEMON.

OPTIONS

The following options are supported.

-a *authlist*

Add the host and port pairs specified in *authlist* to the list of allowed authentication managers. The format of *authlist* is a comma separated list of *hostname*:*port* pairs.

-c *authfile*

Add the host and port pairs specified in the ASCII file *authfile* to the list of allowed authentication managers. The file contains a list of authentication manager specifications, one per line. The specifications take the form of hostname followed by port number, separated by white space. Blank lines and any line whose first printable character is "#" are ignored.

-d Enable debugging output.

-k *authprops*
Set the location for the authentication manager's configuration file to *authprops*. This file is used to find the group signature file in case the *sigfile* key was not specified. The default for this parameter is **/etc/opt/SUNWut/auth.props**. The key in this file that specifies the group signature is **gmSignatureFile**.

-o *optroot*
Set the device information root directory to *optroot*. This directory contains the service named **pipe**, and the **units**, **sessions**, and **session_info** directories. *optroot* is generally shared with other Sun Ray server components.

-p *port*
Set the device manager's listen port to the specified port value. The device manager defaults to port 7011. This is the port by which device services and authentication managers contact the device manager.

-r Automatically restart the device manager daemon if it exits. With this option, the device manager daemon creates two processes: a child that performs all the actual work and a parent monitoring process. The parent process will restart a child if the previous one exits. This enables existing services to re-attach to a new child device manager.

-s *sigfile*
Set the path of the group signature file to *sigfile*.

-t Test mode. Relax checking for error returns for files that are root access. Could cause unpredictable results on an operational device manager in case of true failure.

FILES

The following files are used:

/etc/opt/SUNWut/auth.permit
The customary location of the *authfile* for a system.

/tmp/SUNWut
The customary location for temporary files used by Sun Ray enterprise server managers, designated by *optroot*.

/tmp/SUNWut/.utdevmgr
The named pipe used for communication between the device manager and device driver services.

/tmp/SUNWut/units
The directory containing device directories for each DTU. The directory names are after the DTU's serial numbers. Each directory contains a **dev** directory and a **devices** directory.

/tmp/SUNWut/sessions
The directory containing links to DTU's in the **units** directory, named by X-Windows display number for each session. These links change as users move from one Sun Ray DTU to another.

/tmp/SUNWut/session_info
The directory containing information internal to the device manager for handling session mobility.

/etc/opt/SUNWut/auth.props
The customary location of the *authprops* file containing the authentication manager settings. The device manager looks for the gmSignatureFile key to extract the location of the group signature file.

/etc/opt/SUNWut/gmSignature
The customary location of the *sigfile* file containing the group signature.

ENVIRONMENT VARIABLES

The following environment variables are used:

DISPLAY

Use to get the default X-Windows display number from within the user's environment.

UTDEVROOT

Use to get the devices for the current session from within the user's environment.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto

SEE ALSO

utauthd(1M), **syslog(3)**, **syslogd(1M)**, **syslog.conf(4)**

NAME

utdisk, **utdiskctl** - Sun Ray Mass Storage device and controller drivers.

SYNOPSIS

$UTDEVROOT/dev/dsk/*partition_name*

$UTDEVROOT/dev/rdsk/*partition_name*

DESCRIPTION

The **utdisk** and **utdiskctl** drivers provide the **dkio(7I)** interface to mass storage devices connected to Sun Ray appliances (DTUs).

The actual interface to the DTU for each of these drivers is through the Sun Ray interconnect via the **utstoraged(1M)** daemon.

The **utmountd(1M)** daemon mounts devices with OS-recognizable filesystems on them.

API

Applications open a device link created by **utstoraged**. Links to raw device nodes are created in the **$UTDEVROOT/dev/rdsk** directory. Links to block device nodes are created in the **$UTDEVROOT/dev/dsk** directory.

Device nodes created by **utstoraged** comply with the **dkio(7I)** interface. Hardware limitations in mass storage devices may prevent compliance with these interfaces.

I/O requests to the device must be aligned on a 512-byte boundary, and all I/O request lengths must be in multiples of 512 bytes. Requests that do not meet these requirements will trigger an **EINVAL** error.

DEVICE STATISTICS SUPPORT

Each device maintains I/O statistics for all partitions allocated for that device. For each partition, the driver accumulates reads, writes, bytes read, and bytes written. The driver also initiates hi-resolution time stamps at queue entry and exit points to enable monitoring of residence time and cumulative residence-length product for each queue. Statistics are disabled by default but may enabled in the utdiskctl.conf configuration file.

IOCTLS

Refer to **dkio(7I)**.

ERRORS

EACCES	Permission denied.
EBUSY	The partition was opened exclusively by another thread.
EFAULT	The argument features a bad address.
EINVAL	Invalid argument.
ENOTTY	The device does not support the requested **ioctl()** function.
ENXIO	The device did not exist when attempting operation.
EAGAIN	Resource temporarily unavailable.
EINTR	A signal was caught during the execution of the **ioctl()** function.
ENOMEM	Insufficient memory.
EROFS	The device is read-only.
EPERM	Insufficient access rights.
EIO	An I/O error occurred.

CONFIGURATION

The **utdisk** and **utdiskctl** drivers can be configured by defining properties in the **utdiskctl.conf** file. The following properties are supported:

utdebug

The **utdisk** and **utdiskctl** drivers log errors to the system messages file. The value of the **utdebug** property determines the level of detail in the messages that are logged.

utdebug can accept the following values:

0 log errors only
1 log warnings
2 log error and warning details
3 log instance and device info
4 log operational sequence info
5 log everything else

The default value of this property is 2. Logging cannot be disabled.

utkstats

Setting **utkstats=1** causes **utdiskctl** to maintain I/O statistics for partitions. Set this value to zero to prevent the driver from recording partition statistics. This slightly reduces the CPU overhead for I/O, minimizes the amount of **sar(1)** data collected, and makes these statistics unavailable for reporting by **iostat(1M)**, even though the **-p/-P** option is specified. The default value for this property is 0.

FILES

The following files are used:

- **/usr/kernel/drv/utdiskctl.conf**

Driver configuration files.

- **/var/adm/messages**

System messages file.

- **$UTDEVROOT/dev/dsk/name**

Block interface to disk or partition.

- **$UTDEVROOT/dev/rdsk/name**

Raw interface to disk or partition.

Where *name* is a string descriptive of the device type with a suffix denoting the partition number or UNIX slice number.

EXAMPLES

Example 1: Zip disk with PCFS filesystem (SPARC Platform)

The block device link for the Zip disk is given by **$UTDEVROOT/dev/dsk/zip1s2**

The raw device link for the Zip disk is given by **$UTDEVROOT/dev/rdsk/zip1s2**

Example 2: Hard disk with 1 UFS slice numbered 6 (SPARC Platform)

The block device link for the backup slice is given by **$UTDEVROOT/dev/dsk/disk1s2**

The raw device link for the backup slice is given by **$UTDEVROOT/dev/rdsk/disk1s2**

The block device link for the slice 6 is given by **$UTDEVROOT/dev/dsk/disk1s6**

The raw device link for the slice 6 is given by **$UTDEVROOT/dev/rdsk/disk1s6**

Example 3: Zip disk with PCFS filesystem in the first partition (IA platform)

The block device link for the Zip disk is given by **$UTDEVROOT/dev/dsk/zip1p0**

The raw device link for the Zip disk is given by **$UTDEVROOT/dev/rdsk/zip1p0**

The block device link for the first partition is given by **$UTDEVROOT/dev/dsk/zip1p1**

The raw device link for the first partition is given by **$UTDEVROOT/dev/rdsk/zip1p1**

Example 4: Hard disk with 1 UFS slice numbered 6 in the third partition (IA Platform)

The block device link for the full disk is given by **$UTDEVROOT/dev/dsk/disk1p0**

The raw device link for the full disk is given by **$UTDEVROOT/dev/rdsk/disk1p0**

The block device link for the third partition is given by **$UTDEVROOT/dev/dsk/disk1p3**

The raw device link for the third partition is given by **$UTDEVROOT/dev/rdsk/disk1p3**

The block device link for the backup slice is given by **$UTDEVROOT/dev/dsk/disk1s2**

The raw device link for the backup slice is given by **$UTDEVROOT/dev/rdsk/disk1s2**

The block device link for the slice 6 is given by **$UTDEVROOT/dev/dsk/disk1s6**

The raw device link for the slice 6 is given by **$UTDEVROOT/dev/rdsk/disk1s6**

ENVIRONMENT VARIABLES

UTDEVROOT points to a symbolic link of the device root for the Sun Ray appliance associated with a user's session.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWutstk
MT-Level	Safe

SEE ALSO

utstoraged(1M), **utmountd(1M)**, **utdiskadm(1M)**, **dkio(7I)**, **iostat(1M)**, **sar(1)**

WARNINGS

Users are strongly advised to use the **utdiskadm(1M)** command to prepare the device for removal before unplugging the device or disconnecting their session.

NOTES

Users have access rights only to storage devices connected to the Sun Ray DTU on which their session is active and only for as long as it is active. When the session is disconnected from the Sun Ray DTU whether through logout, hotdesking, server switching, or any other reason, ownership of the storage device is lost and all pending data transfers are aborted. This can result in the filesystem on the media getting corrupted and data getting lost.

On the SPARC platform, only the first partition on disks with multiple FAT partitions is mounted.

NAME

utdiskadm - Sun Ray Mass Storage device management utility

SYNOPSIS

/opt/SUNWut/bin/utdiskadm -c *device_name*

/opt/SUNWut/bin/utdiskadm -e *device_name*

/opt/SUNWut/bin/utdiskadm -h

/opt/SUNWut/bin/utdiskadm -l [-a]

/opt/SUNWut/bin/utdiskadm -m *partition_name* [-p *mount_path*]

/opt/SUNWut/bin/utdiskadm -r *device_name*

/opt/SUNWut/bin/utdiskadm -s [-a]

/opt/SUNWut/bin/utdiskadm -u *mount_path*

DESCRIPTION

The **utdiskadm** command allows the user to perform administration tasks on mass storage devices connected to the Sun Ray DTU with which the current login session is associated. This command does not work on devices belonging to any other user except for some options available only to root. It also cannot be used on devices connected directly to the Sun Ray server.

OPTIONS

The following options are supported.

-c *device_name*

Check device for existence of media

-e *device_name*

Eject media from devices with removable media

-h

Show usage information

-l

List all storage devices of current session and their mount points. This option will show the *device_name*, *partition_name* and *mount_path* for a device.

-l -a

List all storage devices of on system. This option combination can be used by root only

-m *partition_name*

Mount partition partition_name on default mount point in **$DTDEVROOT/mnt**.

-m *partition_name* -p *mount_path*

Mount partition *partition_name* on directory *mount_path*

-r *device_name*

Prepare device *device_name* for removal by unmounting all its partitions

-s

List stale mount points for which physical devices do not exist

-s -a

List stale mount points on entire system All stale mount points on the Sun Ray server are displayed. Only root is allowed to use this option.

-u *mount_path*

Unmount *mount_path*

EXIT STATUS

The following exit codes are returned:

0

The operation was successful

1

The operation was unsuccessful

FILES

The following files are used:

$UTDEVROOT/dev/dsk

The directory containing links to block device names for each partition on the device.

$UTDEVROOT/dev/rdsk

The directory containing links to raw device names for each partition on the device.

ENVIRONMENT VARIABLES

UTDEVROOT points to a symbolic link of the device root for the Sun Ray appliance associated with a user's session.

DTDEVROOT points to a temporary directory associated with the user's session. The directory's lifetime is equal to the lifetime of the login session. It is removed, along with its contents, when the user logs out.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWutsto

SEE ALSO

utmountd(1M), **utstoraged(1M)**, **utdisk(7D)**

NAME

uteject - Sun Ray Mass Storage media eject utility

SYNOPSIS

/opt/SUNWut/bin/uteject *device_name*

DESCRIPTION

The **uteject** command has the same functionality as utdiskadm -e. It ejects media from removable media devices associated with the user's current Sun Ray session. If a filesystem is mounted on the device, the eject operation attempts to unmount it first.

OPTIONS

The following options are supported.

device_name

Eject media from device_name

EXIT STATUS

The following exit codes are returned:

0

The operation was successful

1

The operation was unsuccessful

FILES

The following files are used:

$UTDEVROOT/dev/dsk

The directory containing links to block device names for each partition on the device.

$UTDEVROOT/dev/rdsk

The directory containing links to raw device names for each partition on the device.

ENVIRONMENT VARIABLES

UTDEVROOT points to a symbolic link of the device root for the Sun Ray DTU associated with a user's session.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWutsto

SEE ALSO

utdiskadm(1M), **utmount(1M)**, **utumount(1M)**, **utmountd(1M)**, **utstoraged(1M)**, **utdisk(7D)**

NAME

utfwadm - Sun Ray DTU firmware version management.

SYNOPSIS

/opt/SUNWut/sbin/utfwadm -A { **-a** | **-e** *enetAddr* } [**-f** *firmware*] { **-n** *intf* | **-N** *subnetwork* | **-V** } [**-d**] [**-F**] [**-u**] [**-i** *filename*] ...

/opt/SUNWut/sbin/utfwadm -D { **-a** | **-e** *enetAddr*} { **-n** *interface* | **-N** *subnetwork* | -V } ...

/opt/SUNWut/sbin/utfwadm -P [**-V**]

/opt/SUNWut/sbin/utfwadm -R

DESCRIPTION

The **utfwadm** command manages firmware upgrades to Sun Ray DTUs. The DTUs are capable of loading firmware upgrades and programming new firmware into their flash PROM (Programmed Read-Only Memory).

When a DTU is powered on, the firmware obtains an IP address and other configuration information using the DHCP protocol. Part of the configuration information is a firmware version identifier. If this identifier does not match the DTU's existing firmware, the DTU initiates an upgrade which replaces the current firmware with the new version.

The **utfwadm** command must be run when a new firmware version is installed to update the firmware version identifier and force the DTUs to load the new version on their next power cycle. utfwadm allows firmware identifiers to be set on either a per-network or per-unit basis enabling firmware upgrades to be targeted at entire Sun Ray subnetworks or individual DTUs.

In the SRSS 3.1 release, a new mechanism has been introduced to manage firmware without using DHCP alone. With increased deployment of Sun Rays remote from their servers, it became difficult to supply DHCP parameters from the server to the Sun Ray. Thus, a different mechanism that uses TFTP to access a firmware configuration file on the firmware server was developed. This mechanism involves the creation of *Model*.parms or *Model.*<*MAC*>.parms files for the firmware files corresponding to each Sun Ray *Model* (e.g., ''CoronaP1'') that **utfwadm** installs in the **/tftpboot** directory. The files containing <*MAC*> are unit-specifi, where <*MAC*> is the value defined using the **-e** option.

The modifications to **utfwadm** to support this have been made, and a few additional options have been defined to support control of firmware downloads through the creation of these configuration files. These options allow the forcing of a firmware downgrade where that would normally be prevented, the generation of only these files, without updating DHCP, the use of the frame buffer to hold the downloaded firmware file, and the inclusion of an additional file that specifies parameters other than firmware related ones, such as a list of servers to choose from.

It is possible to determine the firmware versions that are available and in use by each Sun Ray connected to the current server. See the **utfwload(1M)** command.

- In the **/tftpboot** directory are the firmware files. To verify the version of a firmware file, type:

/opt/SUNWut/lib/lzd < /tftpboot/*firmware-filename* **| what**

- To identify the version of firmware that a particular Sun Ray DTU is using, type:

  /opt/SUNWut/sbin/utdesktop -p *desktopID*

Where *desktopID* is the full MAC address. To display the MAC address, press the three audio keys on the Sun Ray DTU keyboard simultaneously.

The Sun Ray subnetworks must have been previously set up using the **utadm(1M)** command. The **utfwadm** command is run under super-user privileges.

OPTIONS

The following options are supported.

-A Add a unit or units to the list of DTUs to be upgraded. Please note: The **-A** option must be followed by either the **-a** or **-e** suboption.

-D Remove the defined DTUs from the list of units to be upgraded. This option causes the firmware version identifier to be unset. Please note: The **-D** option must be followed by either the **-a** or **-e** suboption.

-a Add or delete all units from the list of DTUs to be upgraded.

-e *enetAddr*
Apply the operation to only the specified unit with Ethernet address given by *enetAddr*, where all six hex bytes of the address are specified.

-f *firmware*
This option gives the pathname for the firmware to be downloaded to the DTUs. If *firmware* refers to a file, the hardware version is extracted from the version string within the file, and the file is copied to the **/tftpboot** directory to be downloaded only to that version of the hardware. If *firmware* refers to a directory, then all files in the directory are copied to the **/tftpboot** directory with their version strings appended. If the **-f** option is not given, a default location is used.

The 4.0 release of Sun Ray firmware is split into two varieties: one supports a new capability that allows limited local configuration of the Sun Ray, and the other does not. The default path installs firmware without the configuration capability. To install the firmware that allows configuration, use the "**-f** /opt/SUNWut/lib/firmware_gui" option.

-N *subnetwork*
Apply the given operation to units attached on the specified *subnetwork*. Multiple subnetworks may by given. It can also specify the special keyword **all** to apply the operation to all configured Sun Ray subnetworks.

-N all Apply the given operation to all subnetworks.

-n *intf* Apply the given operation to units connected to the Ethernet interface *intf*. Multiple interfaces may be given, or the special keyword **all**, which applies the operation to all configured Sun Ray interfaces.

-V With **-A**, generate the **.parms** files, but do not update DHCP. Note that this option should not be used if DHCP is still being used to provide firmware parameters to DTUs with old firmware. With **-P**, include a listing of the versions specified in the **.parms** files.

-F Force the Sun Rays to download this firmware, even though the *barrier* for this firmware is less than the *barrier* defined in the current firmware.

-u Turn on firmware load through the frame buffer. This is primarily for debugging purposes, to show the progression of the firmware loading process.

-i *parmfile*
Include the values contained in *parmfile* at the end of the **.parms** file for each firmware model. The key/value pairs are currently restricted to the specification of a servers list as ‘‘**servers=**<*name list*>’’ and a selection order as ‘‘**select=inorder**’’ or ‘‘**select=random**’’. These values are not saved across successive invocations of this command, so to retain them, the **-i** option must be specified each time.

-d Explicitly disable firmware download by creating **.parms** files with **version=_NONE_**, used with **-A**. This is different from **-D** which removes the **.parms** files completely. This is most useful with **-e** to prevent a particular Sun Ray from being upgraded. It also makes sense to use this to disable firmware upgrade completely, but provide a server list using the **-i** option.

-n all Apply the given operation to all interfaces.

-P This variant of the command prints out the version to which each domain should be upgraded on the next power cycle. A domain may be either an interconnect subnet or individual DTU. If it is a subnet, then the **Intf** column lists the interface device. If it is an individual DTU, then its Ethernet address is given in the **Domain** column, and the **Intf** column contains the interface name. If **-V** is specified, the versions from the **.parms** files are printed.

-R Remove the firmware files that were copied into the boot directory.

The **-z** option is RESERVED for use by the Sun Ray server software and should not be used.

FILES

The following files are used:

- **/var/dhcp/dhcptab**

File or NIS+ table

- **/tftpboot**

Default location of firmware boot file

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto

SEE ALSO

dhtadm(1M), **dhcpconfig(1M)**, **what(1)**, **dhcp(4)**, **dhcp_network(4)**, **dhcptab(4)**, **attributes(5)**, **utadm(1M)**, **utdesktop(1M)**

NAME

utfwload - Provide a compact summary of sessions and firmware maintenance.

SYNOPSIS

/opt/SUNWut/sbin/utfwload [**-a**] [**-l**] [**-L**] [**-H**]

DESCRIPTION

Without an argument, the **utfwload** command provides the display number of the current session for each connected Sun Ray in use, as well as the logged-in user, IP address, and firmware revision level. Options allow forcing the download of firmware to Sun Rays that are not running the current version supplied by the Sun Ray server.

Note:

utfwload upgrades Sun Rays that have been loaded with firmware from the SRSS 2.0 114880-04 patch or later or from SRSS 3.0 only. This command cannot force an upgrade of earlier firmware.

OPTIONS

The following options are supported.

-a In combination with other options, this option controls the selection of Sun Rays or sessions to display. Without-*a* , only sesssions that have logged in users are displayed. With *-a*, all sessions or Sun Rays are displayed, and those with no logged in user have a user id field of "????".

-l,**-L** With **utfwload**, force Sun Rays that are not running the version of firmware installed on the current server to upgrade. The command compares the "System Version" as provided by running **utfwadm** **-P** with the value that would be displayed with **-f**. If the version is different, a download is forced. The **-l** form of the command forces loading only on Sun Rays that are connected to sessions with no user logged in, while the **-L** form forces a download on all Sun Rays that are out of date. This option is available only to users with root privileges.

-H Output column headings above the regular output.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto

SEE ALSO

utwho(1M), **attributes(5)**

NAME

utfwsync - Synchronizes Sun Ray DTU firmware downloads.

SYNOPSIS

/opt/SUNWut/sbin/utfwsync [**-d**] [**-v**]

DESCRIPTION

The **utfwsync** command refreshes the firmware level on the Sun Ray DTUs (DTUs) with the default firmware version for the current Sun Ray Server Softrware release and patch level. It then forces all the Sun Ray DTUs connected to that Sun Ray server to restart. If the Sun Ray server is part of a failover group, then all Sun Ray DTUs within the failover group are forced to restart. The result is that each DTU tries to download the latest firmware offered by the primary Sun Ray server as it restarts, as described in the **utfwadm(1M)**man page.

This command is intended for use after software upgrades or after new firmware has been installed on all hosts as part of a patch.

As the command executes, access to user sessions is interrupted, but the sessions are not lost and are returned after the command completes.

The command must be run with superuser privileges.

OPTIONS

The following option is supported.

-d Disables system reconfiguration and forces all connected DTUs to load the currently configured version of the firmware. The currently configured version of the firmware may or may not be the default version for the current release and patch level.

-v Verbose mode. Additional messages regarding what is being done are written to **stdout**.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuta

SEE ALSO

utgstatus(1M), **utauthd(1M)**, **utfwadm(1M)**, **utinstall(1M)**

NAME

utgroupsig - Sets the group signature for Sun Ray servers in a failover group.

SYNOPSIS

/opt/SUNWut/sbin/utgroupsig

DESCRIPTION

The **utgroupsig** command sets the failover group signature.

The **utgroupsig** command prompts for the new signature twice. The group signature file is at least 8 bytes long and has similar content diversity characteristics as required by **passwd(1)**.

The signature is stored in clear in the location specified in the **auth.props** file with the **gmSignatureFile** property. The group signature file is created with owner **root** and mode 600 (read-write by root).

OPTIONS

There are no options for this command.

FILES

The following files are used:

- **/etc/opt/SUNWut/gmSignature**

Sun Ray group signature default file.

- **/etc/opt/SUNWut/auth.props**

Sun Ray authentication properties file.

EXIT STATUS

The following exit values are returned:

0 Success

1 Invalid input to command.

2 Unexpected failure. Signature file unchanged.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuta

SEE ALSO

utrcmd(1M), **passwd(1M)**, **auth.props(4)**

NAME

utgstatus - Display failover group status.

SYNOPSIS

/opt/SUNWut/sbin/utgstatus [**-s** *hostname*]

DESCRIPTION

The **utgstatus** command allows the user to view the failover group status information for the local server or for the named server. The information that the command displays is specific to that server at the time the command is run.

utgstatus displays information only and so can be run by any user.

OPTIONS

The following option is supported.

-s *hostname*

Display all the failover group status information for the specified *hostname*.

EXAMPLES

Example 1: This command displays the failover group status for the local Sun Ray server

% **/opt/SUNWut/sbin/utgstatus**

Example 2: This command displays the failover group status for the server sunray3:

% **/opt/SUNWut/sbin/utgstatus -s sunray3**

Information returned from this command looks like the following for a typical LAN-based configuration:

Or like this for a typical interconnect-based configuration:

```
 host     flags   interface     flags
                  192.24.0.0/24
 -------------    -------------------
 sunray3     TN    192.24.0.136  UAM
 sunray1     T-    192.24.0.93   UA-
 sunray2     TN    192.24.0.95   UAM
 sunray-sras TN    192.24.0.96   U--
```

For proper viewing, make sure the window is wide enough.

Explanation of **utgstatus** information:

The Network/Netmask values are denoted in CIDR (Classless Inter Domain Routing) network address notation, where the initial value (192.24.0.0) is the network address itself and the '/24' suffix signifies the number of bits that are the network identifier of the address. The remaining 8 bits are for specific host addresses.

Host Status Flags

T Trusted — The trusted hosts are members of this failover group because they share the same group signature.

N oNline — The server is configured to participate in load balancing (see the utadm man page for a description of the -n option).

Interface Status Flags

U Up — The interface is currently reachable by this host.

A Available — The interface is available for Sun Rays to connect to it and get service.

M Managing - The interface is configured to manage Sun Rays on its local subnet (in other words, utadm -a was run to configure the interface for Sun Ray service).

In the first example above, all hosts are part of the same failover group. All hosts but sunray1 are "online", which means they will participate in creating in normal session creation. sunray1 is "offline", which means it will not participate in session creation during load balancing for this failover group, although sessions can still be created on it, either explicitly through the use of **utswitch** or **utselect -R**, or implicitly if all other servers are down. The LAN interfaces for all of the hosts are Up, and all but sunray-sras are Available to Sun Rays (sunray-sras did not have **utadm -a** run to configure its interface for Sun Ray service, and it does not have allowLANConnections=true set in auth.props. It is a dedicated SRAS server for the failover group). Both sunray2 and sunray3 are Managing Sun Rays, because **utadm -a** was run for their LAN interfaces. They will offer DHCP parameters and possibly addresses during the Sun Ray bootup phase for Sun Rays on their local subnets.

In the second example above, all of the hosts are Trusted and Online, meaning that they will all participate in failover and load balancing for their Available interfaces. 193.25.0.0/24 is the LAN network, and the other networks are Sun Ray interconnects. All of the LAN interfaces are Up, meaning that they are reachable, but none are Available for Sun Ray service, and none are Managing Sun Rays on the local subnet. All of the interconnect interfaces are Up and Available, and all are Managing Sun Rays.

SEE ALSO

utadm, **auth.props**, **utswtich**, **utselect**

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuta

NAME

utinstall - Sun Ray Server Software installation, upgrade, and removal utility.

SYNOPSIS

/cdrom/cdrom0/utinstall [**-a** *admin-file*] [**-d** *media-dir*] [**-u**] [**-j** *jre*]

/opt/SUNWut/sbin/utinstall [**-a** *admin-file*] [**-d** *media-dir*] [**-u**] [**-j** *jre*]

DESCRIPTION

The **utinstall** command installs, upgrades, and removes Sun Ray Server Software. All software required to support the Sun Ray server is installed.

The **utinstall** command is run under superuser privileges and prompts the user before taking any action. The use of defaults is recommended.

OPTIONS

The following options are supported.

-a *admin-file*
: Instead of the default, use *admin-file* as the admin file for **pkgadd** operations (see **pkgadd(1M)**).

-d *media-dir*
: Instead of the default, use *media-dir* as the installation media root directory.

-j *jre* Instead of the default, use the particular jre specified in the argument.

-u Remove previously installed Sun Ray Server Software.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuti

SEE ALSO

patchadd(1M), **patchrm(1M)**, **pkgadd(1M)**, **pkgrm(1M)**, **admin(4)**

NAME

utkiosk - manage configuration for Sun Ray Kiosk Mode

SYNOPSIS

/opt/SUNWut/sbin/utkiosk **-i** *name* **-f** *file* [**-A** *applist-file*]

/opt/SUNWut/sbin/utkiosk **-e** *name* [**-s**|**-a**]

/opt/SUNWut/sbin/utkiosk **-d** *name*

/opt/SUNWut/sbin/utkiosk **-h**

DESCRIPTION

The **utkiosk** tool is used to manage Kiosk Mode configuration stored in the Sun Ray administration database. **utkiosk** allows storing, retrieving and deleting kiosk configuration data.

The data stored by **utkiosk** is instantly applied to newly started Sun Ray Kiosk Mode sessions. To enable Kiosk Mode for Sun Ray sessions, use the **-k** option to the **utpolicy** tool.

Kiosk configuration consists of a session configuration file, that references an installed kiosk session descriptor, as well as an optional application list. The session configuration file must be in the session.conf(4) **format, as processed by the kioskconfig(1M)** tool. The *KIOSK_SESSION_APPLIST* setting cannot be used here. Instead an application list for application container sessions can be specified as a file using the **-A** option. The application list will be embedded in the stored settings rather than maintained as a file reference.

In addition to the common **session.conf** settings, you can use the *KIOSK_SESSION_TIMEOUT_DETACHED* setting to specify a timeout (in seconds) after which a disconnected kiosk session will be terminated.

OPTIONS

The following options are supported.

-a Export only the application list contained in the configuration entry. The **-a** option cannot be used together with **-s**.

-A *applist-file*
Import an application list from *applist-file*.

-d *name*
Delete the kiosk configuration entry named *name* from the datastore. Kiosk Mode is disabled, if the **session** entry is deleted.

-e *name*
Export the kiosk configuration entry named *name* from the datastore. By default a complete session configuration is exported. A configured application list is embedded in the exported session configuration as **KIOSK_SESSION_APPLIST** entry. The **-s** and **-a** options can be used to separately export the session configuration and application list.

-f *file* Specify the file from which session configuration is read.

-h Print a usage message and exit.

-i *name*
Import session configuration into the kiosk configuration entry named *name* in the datastore. Configuration consists of a session configuration file and an optional application list.
If the session configuration file contains a non-empty **KIOSK_SESSION** value, but does not contain a **KIOSK_ENABLED** value, **KIOSK_ENABLED=yes** is implied for the resulting configuration.

-s Export the session configuration without policy or embedded application list. The output does not contain a **KIOSK_ENABLED** or **KIOSK_SESSION_APPLIST** setting. The **-s** option cannot be used together with **-a**.

OPERANDS

name The name of the kiosk configuration entry in the datastore. The following entries are currently defined for Sun Ray Kiosk Mode:

session

The **session** entry holds kiosk configuration data for Sun Ray Kiosk Mode.

kiosk

This entry was used in prior versions for configuring Controlled Access Mode 3.x and older. You can still access this configuration data by using the **kiosk** name. The **-a**, **-s** and **-A** options cannot be used with **kiosk**.

Other entries are reserved and should not be used.

EXAMPLES

Example 1: This command imports session configuration into the datastore.

utkiosk -i session -f mysession.conf

Example 2: This command imports configuration for an application container session and an associated application list into the datastore.

utkiosk -i session -f mydesktop.conf -A myapplist

Example 3: This command exports the complete configuration from the datastore into a file.

utkiosk -e session > mysession.conf

Example 4: These commands exchange the application list to be used with the current application container session.

utkiosk -e session -s > /tmp/mysession.conf
utkiosk -i session -f /tmp/mysession.conf -A newapplist

Example 5: This command exports the legacy configuration into a local file.

utkiosk -e kiosk > kiosk.conf

Example 6: This command deletes legacy configuration data from the datastore.

utkiosk -d kiosk

EXIT STATUS

The following exit values are returned:

0 Success

1 Failure

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuta
Interface Stability	Uncommitted

SEE ALSO

kioskconfig(1M), **session.conf(4)**, **kiosk(5)**

NAME

utkioskoverride - Set or query the session type assigned to tokens.

SYNOPSIS

/opt/SUNWut/sbin/utkioskoverride

/opt/SUNWut/sbin/utkioskoverride [**-r** *raw*] [**-t** *token*]

/opt/SUNWut/sbin/utkioskoverride **-s** *session* **-r** *raw*|**-t** *token*

DESCRIPTION

The **utkioskoverride** command provides a way to set the session type associated with a token or to query the session type currently associated with a token. Possible session types are "default" (where the system wide policy dictates the session type), "kiosk" (where a kiosk session is assigned to that token regardless of the system policy) and "regular" (where a regular, i.e non kiosk, session is assigned to the token regardless of the system policy).

OPTIONS

Without any parameters, the command will print out a usage message. Otherwise, the following options are supported:

-s *session*-**r** *raw*|**-t** *token*

Set the session type for a token to the specified *session*, which must be either "default", "kiosk" or "regular". Either a raw insert token (with the **-r** option) or a session token (with the **-t** option) must be specified as the target of the assignment. The command will assign the session type directly to the session token or to the logical token associated with the registered raw insert token. At this time, only logical tokens are supported as valid arguments for the **-t** option. Future releases may allow assigning a session type to other session token types. If the specified token fails to meet the requirements (i.e is neither a logical token for the **-t** option nor a registered token for the **-r** option), the command will return an error code.

-r *raw* **-t** *token*

Query and display the currently assigned session type for the specified token. At least one of a raw insert token (with the **-r** option) or a session token (with the **-t** option) must be provided, and the command will attempt to find the session type with the following logic: if a session token is specified and has an associated session type, it will be displayed, else if a raw insert token is specified and is registered as a logical token, that token's associated session type will be displayed.

The command displays the value of the session type in the following format:

SESSION_TYPE=<session type>

where <session type> can be either "default", "kiosk" or "regular". If the tokens provided don't have any explicit session type assigned, "default" will be printed out.

You can use the **utuser(1M)** command to view session type assignments for multiple logical tokens.

EXAMPLES

Example 1: This command sets the session type for the registered smart card "MicroPayFlex.12345678" to "kiosk":

 # /opt/SUNWut/sbin/utkioskoverride -s kiosk -r MicroPayFlex.12345678

Example 2: This command queries the session type for the registered smart card "MicroPayFlex.12345678":

 # /opt/SUNWut/sbin/utkioskoverride -r MicroPayFlex.12345678
SESSION_TYPE=kiosk

Example 3: This command queries the session type for the logical token user.123456-78:

 # /opt/SUNWut/sbin/utkioskoverride -t user.123456-78
SESSION_TYPE=default

EXIT STATUS

The following exit values are returned:

0 Success

1 If no token is specified or the session type is invalid

2 If the token is not a suitable token when using the **-s** option

3 If the executable is not run as root when using the **-s** option

4 Generic error

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuta

SEE ALSO

utpolicy(1M), **utuser(1M)**, **attributes(5)**

NAME

utlicenseadm - Utility to manage Sun Ray device licenses.

SYNOPSIS

/opt/SUNWut/sbin/utlicenseadm -l [**-t** *<licenseType>]* [**-k** *<licenseKey>]* [*<desktopid>*]

/opt/SUNWut/sbin/utlicenseadm -d [**-t** *<licenseType>]* [**-k** *<licenseKey>]* *<desktopid>*

DESCRIPTION

utlicenseadm is a utility to administer device licenses for Sun Rays. Currently, it can be used to list and delete MS-TSCAL licenses stored by the Sun Ray Windows Connector in SRDS.

OPTIONS

The following options are supported.

-l List the licenses. If no desktopid is specified then all matching licenses are listed for all DTUs.

-d Delete licenses for a specific DTU. The desktop id must be specified for this option. A specific license for a DTU may be deleted by providing appropiate license type and/or license key.

-t *licenseType*
Specify the license type to be administered. The current supported license type is only "RDPLicense".

-k *licenseKey*
A key for the license if applicable.

desktopid
Desktop id to administer the device license for.

EXIT STATUS

0 if operation performed successfully, 1 if error found

SEE ALSO

uttsc(1)

NAME

utmhadm - Sun Ray DTU multihead group configuration utility.

SYNOPSIS

/opt/SUNWut/sbin/utmhadm [*groupname*]

/opt/SUNWut/sbin/utmhadm **-a** *groupname* **-g** *COLS*x*ROWS* **-p** *primaryCID* **-l** *CID1,CID2,...,CIDn*

/opt/SUNWut/sbin/utmhadm **-d** *groupname*

/opt/SUNWut/sbin/utmhadm **-e** [**-f** *filename*]

/opt/SUNWut/sbin/utmhadm **-o** [**-f** *filename*]

/opt/SUNWut/sbin/utmhadm **-h**

/opt/SUNWut/sbin/utmhadm

DESCRIPTION

The **utmhadm** command provides a way to administer Sun Ray server multihead multihead groups. The information that **utmhadm** displays and that is editable is stored in the Sun Ray administration database.

The **utmhadm** operations that only display information may be run by any user. Operations that change data must be run as superuser.

OPTIONS

The following options are supported.

-a *groupname*
: Creates a new multihead group having identifier *groupname*. This name must be unique and not already exist on the system.

-d *groupname*
: Removes the multihead group for the specified *groupname*.

-e
: Populates the system multihead group database with input data of the format produced by **-o**, from standard input.

-f *filename*
: Specify a *filename* for use with **-e** or **-o** instead of standard input or output.

-g *COLS*x*ROWS*
: Specifies the geometry of the multihead group in the form *COLS*x*ROWS*. This number of columns and rows must not exceed the maximum number allowed and must match the number of DTUs specified with **-l**. This option can only be used with **-a**.

-h
: Prints the usage message.

-l *CID1, CID2,..., CIDn*
: Specifies the DTU canonical identifiers when creating a group. A canonical identifier has the form IEEE802.*nnnnnnnnnnn* or *nnnnnnnnnnn* (the 12-digit hexadecimal MAC address of the DTU) and the list must be comma-separated. The identifiers must be specified in row-major order. The maximum number of DTUs allowed is 16.

-o
: Dumps all system configured multihead group data, in comma-separated format, to standard output. Intended for subsequent use with **-e**.

-p *primaryCID*
: Identifies which in the list of canonical identifiers, specified with **-l**, is designated as the primary DTU within the group. The primary is repeated in the list specified by **-l**. This option can only be used with **-a**.

When no options are provided, **utmhadm** lists information about all multihead groups configured on the system.

EXAMPLES

Example 1: This command list all DTUs that are in the multihead group:

% **/opt/SUNWut/sbin/utmhadm tera**

Here is sample output:

```
Multihead Group    Geometry          CIDs
------------------ ------------------ --------------------------------
tera               geometry=2x1       IEEE802.080020b538dc (P)
                                      IEEE802.080020b56e2d
```

Example 2: This command creates a terminal group having two terminals with the first one being the primary:

 # /opt/SUNWut/sbin/utmhadm -a srgroupA -g 2x1 -p IEEE802.080020b0562f -l IEEE802.080020b0562f,IEEE802.080020b64574

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuta

SEE ALSO

utxconfig(1), *Sun Ray Server Software Administrator's Guide*

NAME

utmhconfig - Sun Ray mulithead GUI configuration utility.

SYNOPSIS

/opt/SUNWut/sbin/utmhconfig

DESCRIPTION

The **utmhconfig** utility allows the administrator to list, add, or delete multihead groups easily. The initial screen lists any existing multiheaded groups and allows the administrator to select those to delete. The utility can also be used to create a new group. To do this, the administrator starts the utility on the Sun Ray DTU that is to become the "primary" of the group (it has the keyboard, mouse, and all the devices for the group). The administrator selects "Create New Group" and follows the instructions in the wizard to identify all of the terminals in the new multihead group. The administrator run the utmhconfig command as superuser and must have a recognized smart card available.

OPTIONS

There are no options for **utmhconfig**

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuta

SEE ALSO

utmhadm(1M), **utxconfig(1)**

NAME

utmhscreen - Sun Ray multihead GUI screen display tool.

SYNOPSIS

/opt/SUNWut/lib/utmhscreen [**-l**]

DESCRIPTION

The **utmhscreen** tool provides a window showing the respective displays location in the multihead group. The display showing the widow is highlighted in white, while the other displays are darkened. The window is located in the upper right corner of the display.

This tool is automatically launched for users during the X server startup process (session creation). If the X server is not running in a multihead environment, the tool immediately exits.

OPTIONS

utmhscreen accepts the following option:

-l Indicates to **utmhscreen** that it is being auto launched by the windowing system. The use of this option is beyond the scope of this manual.

RESOURCES

The tool understands all of the core X Toolkit and Motif resource names and classes as well as:

enableAutoLaunch (class EnableAutoLaunch)
Specifies whether or not **utmhscreen** should be launched automatically during X session startup. The default is "true".

EXAMPLES

Example 1: To disable automatic launching of utmhscreen for a user, set the following X resource in their **$HOME/.Xdefaults** file:

Utmhscreen*enableAutoLaunch: false

ATTRIBUTES

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

Attribute Types	Attribute Values
Availability	SUNWuta
Interface Stability	Evolving

SEE ALSO

utmhadm(1M). **utmhconfig(1M)**, **utxconfig(1)**

NAME

utmount - Sun Ray Mass Storage media mount utility

SYNOPSIS

/opt/SUNWut/bin/utmount -m *partition_name*[-p *mount_path*]

DESCRIPTION

The **utmount** command has the same functionality as utdiskadm -m. It is used to mount partition_name on either the default mount point in $DTDEVROOT/mnt, or on a user supplied mount point mount_path.

OPTIONS

The following options are supported.

-m partition name

Mount partition partition_name on default mount point in $DTDEVROOT/mnt.

-m partition_name -p mount_path

Mount partition partition_name on directory mount_path

EXIT STATUS

The following exit codes are returned:

0

The operation was successful

1

The operation was unsuccessful

FILES

The following files are used:

Table 1

$UTDEVROOT/dev/dsk
: The directory containing links to block device names for each partition on the device.

$UTDEVROOT/dev/rdsk
: The directory containing links to raw device names for each partition on the device.

ENVIRONMENT VARIABLES

UTDEVROOT points to a symbolic link of the device root for the Sun Ray DTU associated with a user's session.

DTDEVROOT points to a temporary directory associated with the user's session. The directory's lifetime is equal to the lifetime of the login session. It is removed, along with its contents, when the user logs out.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWutsto

SEE ALSO

utdiskadm(1M), **uteject(1M)**, **utumount(1M)**, **utmountd(1M)**, **utstoraged(1M)**, **utdisk(7D)**

NAME

utmountd - Sun Ray Mass Storage media mounter daemon

SYNOPSIS

/opt/SUNWut/lib/utmountd -m [-D *debug-level*] [-p *poll_interval*] [-t *max_threads*]

DESCRIPTION

utmountd performs mount and umount operations on Sun Ray mass storage devices managed by utstoraged(1M). Slices or partitions with OS recognizable filesystems are mounted on a directory inside the current sessions mount directory, $DTDEVROOT/mnt.

Error messages from utmountd are logged to **/var/opt/SUNWut/log/utmountd.log**.

OPTIONS

The following options are supported.

-D *debug-level*

Debug mode. Use is beyond the scope of this document. If a debug level is set, debug messages are sent to stderr.

-p *poll_interval*

Minimum time in seconds between polling removable media devices for media insertion events

-t *max_threads*

Maximum number of simultaneous service threads allowed. The more active threads, higher the load on the system. A lower number of threads mean slower response times to mounting or unmounting devices if the daemon is overloaded with work.

FILES

The following files are used:

$DTDEVROOT/mnt

A link pointing to the directory containing mount points

ENVIRONMENT VARIABLES

DTDEVROOT points to a temporary directory associated with the user's session. The directory's lifetime is equal to the lifetime of the login session. It is removed, along with its contents, when the user logs out.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWutsto

SEE ALSO

utdiskadm(1M), **utstoraged(1M)**, **utdisk(7D)**

NAME

utparallel, utserial - Sun Ray serial and parallel port device driver emulators.

SYNOPSIS

#include <sys/types.h>

#include <fcntl.h>

utserial

#include <sys/termios.h>

#include <termio.h>

utparallel

#include <sys/ecppio.h>

DESCRIPTION

utserial is a tty-style interface that provides a generic virtual interface to USB serial adaptors connected to the Sun Ray DTU.

utparallel is a parallel-style interface that provides a generic virtual interface to USB parallel adaptors connected to the Sun Ray DTU.

utserial and **utparallel** are each loadable **STREAMS** drivers.

EXTENDED DESCRIPTION

The actual interface to the DTU for each of these drivers is through the Sun Ray interconnect via either the **utseriald** daemon or the **utparalleld** daemon, each of which is session-aware. The daemons are connected to either **utserial** or **utparallel** through a master port and each is responsible for creating the slave device nodes through which normal applications will connect.

API

Applications open a device file created by either **utseriald** or **utparalleld**. Device files created by **utseriald** comply to the **termio**(7I) interface and device files created by **utparalleld** comply to the **ecpp**(7D) interface. Hardware limitations in USB adaptors might prevent compliance with these interfaces.

FILES

The following files are used:

- **/dev/utserial**

Master port for **utserial**

- **/dev/utparallel**

Master port for **utparallel**

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWutu
MT-Level	Safe

SEE ALSO

utseriald(1M), **utparalleld(1M)**, **termio(7I)**, **ecpp(7D)**

NAME

utparalleld - Sun Ray printer service daemon.

SYNOPSIS

/opt/SUNWut/lib/utparalleld [**-D** *debug-level*] [**-o** *optroot*] [**-r**]

DESCRIPTION

utparalleld provides printer support for Sun Ray DTUs. **utparalleld** supplies driver services for all USB parallel adaptors and USB printers that comply with the USB printer class.

utparalleld uses the Sun Ray parallel I/O driver on solaris and Sun Ray utio driver on linux to provide applications the same interface as standard workstation parallel ports, such as **/dev/ecpp** or **/dev/bpp** on solaris and **/dev/parport** on linux. Applications such as the **lp**(1) daemon can use device nodes that are provided by **utparalleld**.

When a parallel adaptor or a USB printer is attached to a DTU, **utparalleld** creates device nodes in the **$UTDEVROOT/devices** directory. A user can use the device link **$UTDEVROOT/dev/printers/***printer-name* to access a specific printer.

OPTIONS

The following options are supported.

-D *debug-level*
: Debug mode. Use is beyond the scope of this document.

-o *optroot*
: Use *optroot* as the parallel service's root directory for device node creation. The default value is **/tmp/SUNWut**. *optroot* should be the same directory as the *optroot* directory used by **utdevmgrd**(1M).

-r Automatically restart the printer service daemon if it exits. With this option, the printer service daemon creates two processes: a child that performs all the actual work, and a parent monitoring process. The parent process restarts a child if the previous one exits.

FILES

The following files are used:

$UTDEVROOT/**dev/printers**
: The directory containing links to parallel device names for each DTU..

ENVIRONMENT VARIABLES

UTDEVROOT points to a symbolic link of the device root for the Sun Ray DTU associated with a user's session.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto

SEE ALSO

utdevmgrd(1M), **utseriald(1M)**

NAME

utpolicy - Sun Ray authentication manager policy management command.

SYNOPSIS

/opt/SUNWut/sbin/utpolicy **-a** [**-g**] [**-p**] [**-r** *type*] [**-s** *type*] [**-z** *type*]

/opt/SUNWut/sbin/utpolicy

/opt/SUNWut/sbin/utpolicy **-h**

/opt/SUNWut/sbin/utpolicy **-g**

/opt/SUNWut/sbin/utpolicy **-k** {*card*|*pseudo*|both}

/opt/SUNWut/sbin/utpolicy **-d**

/opt/SUNWut/sbin/utpolicy **-M**

/opt/SUNWut/sbin/utpolicy **-S**

DESCRIPTION

The **utpolicy** command simplifies and writes the policy configuration of the Sun Ray authentication manager, **utauthd(1M)**.

OPTIONS

The following options are supported.

POLICY SETTING

The specified Policy Setting arguments completely replace the current active authentication policy. Only arguments that are specified become active. Policy Setting and Card Reader Assignment arguments can be specified together

-a This option, followed by valid Policy Setting, or Card Reader Assignment arguments, applies these arguments to the active authentication policy for the system. This option is not valid by itself.

-d Disable the Exit menu option from the Non-smart card mobility GUI. The Exit option is enabled by default.

-g Turn on session selection within a server group. Allows the user to select on which server the user's session is run.

-k {card|pseudo|both}

Enables Kiosk Mode for the specified session type. Selecting *card* enables Kiosk Mode for card sessions, selecting *pseudo* for terminal sessions; and selecting *both* enables Kiosk Mode for all types of sessions. For functionality, at least one of the **-r**, **-s**, or **-z** options must be invoked with the same argument as the -k option. The **-k** option is not considered until the **utconfig** application has configured for Kiosk Mode.

Note: Kiosk Mode policy enabled using the **-k** option may be overriden using **utkioskoverride(1M)**.

-M Enable non-smart card mobile sessions.

-m Enable multihead session capability, allowing multiple terminals to act as display devices for a single user session.

-p This option changes the behavior of the self-registration application so that it does not require a name and password before registering a token. Note that the self-registration application only verifies the name and password. They are not stored.

-r {card|pseudo|both}

Specify the token types that must be registered in the administrative database in order to be granted access to a login screen. Policy looks up and uses token database entry

-S *smartcard_type*

Specify the smart card type that should cause the smart card login GUI (utsclogin) to be invoked when a card of the type in the list is inserted into the DTU. This is the smart card token type (without the token ID) as provided by the Sun Ray software. If more than one smart card type is to be specified, multiple -S options must be specified.

-s {card|pseudo|both}

Specify the token types that will be presented with a registration screen if they do not have an entry in the administrative database. Policy allows self-registration of tokens.

-z {card|pseudo|both}

Specify the token types that do not require an entry in the administrative database in order to be granted access to a login screen. Policy grants access to tokens without database entry.

With the **-h** option, the **utpolicy** command prints out the usage message.

With no options, the **utpolicy** command prints out the policy in effect.

The following options are RESERVED for use by the Sun Ray Server Software and should not be used:

-G, -P, -Q, -b, -f, -l, -u, -x, +x

EXAMPLES

Example 1: This command configures the policy so that all access via smart card requires a valid administrative database entry before access is granted. If a database entry has not been created for a smart card, then a registration session is presented on the DTU. If no smart card is used, then the normal login screen is presented.

/opt/SUNWut/sbin/utpolicy -a -r card -s card -z pseudo

FILES

The following files are used:

- **/etc/opt/SUNWut/policy/utpolicy**

The policy configuration file

- **/etc/opt/SUNWut/auth.props**

Sun Ray authentication manager's configuration file

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto

SEE ALSO

utauthd(1M), **utreader(1M)**, **utrestart(1M)**, **utkioskoverride(1M)**, **auth.props(4)**, **pam(3PAM)**

NAME

utpreserve - Sun Ray configuration file preservation utility.

SYNOPSIS

/cdrom/cdrom0/utpreserve [**-d** *preserve-directory*]

DESCRIPTION

The **utpreserve** command stops Sun Ray services, terminates user sessions, and saves existing Sun Ray server configuration data into a compressed tar file, **/var/tmp/SUNWut.upgrade/preserve_4.0.tar.gz**.

OPTIONS

The following option is supported.

-d *preserve-directory*

Save the compressed tar file into the *preserve-directory*.

SEE ALSO

utinstall(1M), **utconfig(1M)**

NAME

utpw - Sun Ray administration password change utility.

SYNOPSIS

/opt/SUNWut/sbin/utpw

DESCRIPTION

The **utpw** command changes the Sun Ray administrator password (also known as the "UT admin" password). This password is entered by the administrator when logging into the Administration Tool and is used to make a privileged connection to the LDAP server.

utpw changes the password both in the administration database, and the password file on the local server.

In a failover group, **utpw** also affects the administration database of the secondary servers, but only the password file on the local server. The administrator must log into the secondary servers and run **utpw** on them to change the password files.

OPTIONS

There are no options for **utpw**.

EXAMPLES

Example 1: This command changes the administration password:

```
# /opt/SUNWut/sbin/utpw
Enter new UT admin password:
Re-enter new UT admin password:
Enter old UT admin password:

Changing LDAP password...
Done.
Changing password file...
Done.
```

FILES

The following files are used:

- **/etc/opt/SUNWut/utadmin.pw**
- **/etc/opt/SUNWut/utadmin.conf**

EXIT STATUS

The following exit values are returned:

0 Success

1 Error

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuta

SEE ALSO

utdesktop(1M), **utuser(1M)**, *Sun Ray Server Software Administrator's Guide*

NOTES

The **-f** option has been deprecated. Use **utpw** instead. If you use the **-f** option, you must supply the Sun Ray administration password though there is no prompt for it displayed.

NAME

utquery - query Sun Ray Desktop Units current parameter values

SYNOPSIS

/opt/SUNWut/sbin/utquery [-d] *IP_address*

/opt/SUNWut/sbin/utquery -h

DESCRIPTION

The **utquery** command allows administrators to query the current parameter values on a Sun Ray Desktop Unit (DTU). The *IP_address* may specify the network IP address of a single DTU to query, a subnet broadcast address to query all DTUs on the subnet or the broadcast address to query all DTUs associated with this Sun Ray server.

The utquery command can be used to aid in diagnosing problems when a DTU is unable to successfully connect to an authentication manager or when a Multihead group is redirected outside their "home" failover group.

OPTIONS

The following options are supported.

-d Report the DHCP parameters the DTU obtained at boot-up.

-h Print the usage.

EXAMPLES

Example 1: To display the DHCP values for the DTU at IP address 129.146.58.182

% **utquery -d 129.146.58.182**

The following is an example of the output for the command above:

```
terminalID=080020d15f23
    terminalIPA=10.6.102.224
    model=CoronaP3
    Subnet=255.255.255.0
    Router=10.6.102.1
    Broadcst=10.6.102.255
    LeaseTim=86400
    DHCPServer=10.6.102.3
    INFORMServer=10.6.102.113
    AuthSrvr=10.6.102.113
    AuthPort=7009
    LogHost=10.6.102.113
    FwSrvr=10.6.102.113
    NewTVer=3.1_12,REV=2005.04.13.06.03
    parmsVersion=3.1_12,REV=2005.04.13.06.03
    parmsBarrier=310
    parmsBarrierLevel=310
    parmsServers=
    parmsSelect=default
    currentAuth=10.6.102.111
    currentFW=3.1_12,REV=2005.04.13.06.03
    currentBarrier=310
    currentBarrierLevel=310
    dnsList=129.147.5.51,129.145.155.32,129.145.155.42
    dname=SFBay.Sun.COM
```

In the output above, there can be a number of values with *Barrier* in the name, and understanding the differences is subtle. In general, the *Barrier* value is the value associated with a firmware file, and its corresponding version. So, *currentBarrier* is the barrier value for the current firmware running on the DTU, and

parmsBarrier is the barrier level of the file residing on the firmware server. In the absence of any *BarrierLevel* values, the firmware file on the server will not be loaded if its *Barrier* value is less than the currently running firmware. For example, in the output above, the *currentBarrier* is 310, so any attempt to load SRSS 2.0 firmware, which has a barrier level of 200, would fail. In order to permit the forced loading of firmware with a lower barrier level, the *BarrierLevel* values are used. The *BarrierLevel* is either the *currentBarrier* value of the resident firmware, the *BarrierLevel* value provided from DHCP, or the *parmsBarrierLevel* obtained from the *.parms* file. Later values in this list take precedence if present. It is the *currentBarrierLevel* that is compared with the barrier level of the server's file to determine if the download is permitted or not.

Example 2: To display the DHCP values for all DTUs on subnet 129.146.58:.

utquery -d 129.146.58.255

Example 3: To display the DHCP values for all DTUs on this server:

utquery -d 255.255.255.255

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuta

NOTES

The utquery command will only report on DTUs that are operating with firmware version 2.0 or newer.

The utquery command will only report on DTUs that have successfully acquired DHCP parameters from a DHCP server.

The**NewTVer** value reported is the firmware revision level supplied by DHCP, which is used to determine whether a firmware upgrade is required. The **currentFW** value is the firmware revision level of the current firmware running in the Sun Ray device.

The **AuthSrvr** and **AltAuth** values reported are those provided by DHCP parameters at DTU boot time. The **currentAuth** value is the IP address of the Sun Ray server to which the Sun Ray is currently connected.

In firmware version 3.1 and newer, additional values obtained via TFTP from the *.parms* file are also printed. (See **utfwadm(1m)**.) If **parmsVersion** is defined, it overrides **NewTVer**.

The **utquery** command with a broadcast address does not work on remote subnets because most routers do not forward the **broadcast-address** packets.

NAME

utrcmd - Sun Ray remote administration utility.

SYNOPSIS

/opt/SUNWut/lib/utrcmd [**-n**] *hostname command* [*args*]

DESCRIPTION

The **utrcmd** program provides a way to run certain Sun Ray administration commands remotely. It contacts the **in.utrcmdd**daemon on the remote *hostname* and executes the specified *command* with the specified arguments *args* (if any).

utrcmd copies its standard input to the specified command, the standard output of the command to **utrcmd**'s standard output, and the standard error of the command to **utrcmd**'s standard error. Interrupt, quit, and terminate signals are propagated to the specified command; **utrcmd** terminates normally when the command does.

OPTIONS

The following option is supported.

-n Redirect the input of **utrcmd** to **/dev/null**. This option prevents interactions between **utrcmd** and the shell which invokes it. For example, if you are running **utrcmd** and invoke a **utrcmd** in the background without redirecting its input away from the terminal, it will block even if no reads are posted by the specified command. The **-n** option prevents this behavior.

USAGE

Official hostnames or nicknames may be given as the *hostname*.

The **utrcmd** and **in.utrcmdd** programs use the Sun Ray failover group configuration to perform a set of checks before allowing the specified command to proceed.

The program **utrcmd** runs with set-user-ID permission of root or superuser. The **utrcmd** command proceeds only if all of the following are true (on the initiating system):

- The user's real user-ID is superuser, or the user has membership rights in the **utadmin** group.
- The **/etc/opt/SUNWut/auth.props** file is owned by superuser and is not writable by anyone other than superuser.
- The **gmSignatureFile** property of **auth.props** specifies a group signature file.
- The group signature file exists and is owned by superuser and is not readable, writable, or executable by anyone other than superuser.
- The group signature file is at least 8 bytes long and has similar content diversity characteristics as required by **passwd**(1).
- The **utrcmd/tcp** service is enabled.

The **in.utrcmdd** program will accept the connection only if all of the following are true on the remote system:

- The **utrcmd/tcp** service is enabled and matches the configuration on the initiating system.
- The **in.utrcmdd** program is enabled in **/etc/inetd.conf**.
- The **utadmin** group is configured on the system.
- The **/etc/opt/SUNWut/auth.props** file is owned by superuser and is not writable by anyone other than superuser.
- The **gmSignatureFile** property of **auth.props** specifies a group signature file.The group signature file exists and is owned by superuser, and is not readable, writable, or executable by anyone other than superuser.
- The group signature file is at least 8 bytes long and has similar content diversity characteristics as required by **passwd**(1).

If the connection is accepted, the **utrcmd** program begins a challenge-response handshake with the

in.utrcmdd program, using the contents of the group signature file to sign messages (without revealing the contents of the signature file). Either **utrcmd** or **in.utrcmdd** will reject the transaction if the handshake fails. The specified command will not be run if the contents of the group signature files on the two systems differ.

Finally, **in.utrcmdd** rejects the specified command if it is not recognized. Specified commands always run in group **utadmin**.

The following commands are allowed and are always run in group **utadmin**:

- **/opt/SUNWut/sbin/utreplica**
- **/opt/SUNWut/sbin/utpolicy**
- **/opt/SUNWut/sbin/utfwadm**
- **/usr/sbin/dhtadm**
- **/usr/sbin/pntadm**
- **/opt/SUNWut/lib/utauthd**
- **/etc/init.d/utsvc**
- **/opt/SUNWut/sbin/utsession**
- **/opt/SUNWut/sbin/utreader**
- **/opt/SUNWut/sbin/utrestart**

Impact of disabling **utrcmd**:

utrcmd is used to run certain Sun Ray administration commands on remote systems. Hence disabling **utrcmd** means that the following functionality will not work as expected:

- Admin GUI
  - Restart Services
  - Sun Ray Session management
- CLI
  - **/opt/SUNWut/sbin/utreplica** to setup a failover group
  - **/opt/SUNWut/sbin/utdssync** to convert and synchronize the Sun Ray data store
  - **/opt/SUNWut/sbin/utfwsync** to synchronize firmware levels

EXAMPLES

Example 1: This command lists the configured token readers on a remote Sun Ray server

 # /opt/SUNWut/lib/utrcmd sun5 /opt/SUNWut/sbin/utreader

FILES

The following files are used:

- **/etc/hosts**

Internet host table

- **/etc/group**

Group file

- **/etc/inet/services**

Internet services table

- **/etc/inetd.conf**

Internet services daemon configuration table

- **/etc/opt/SUNWut/auth.props**

Sun Ray authentication properties file

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto

SEE ALSO

utauthd(1M), **inetd(1M)**, **group(4)**, **auth.props(4)**, **hosts(4)**, **nsswitch(4)**, **passwd(1)**, **rsh(1)**, **attributes(5)**

NOTES

utrcmd works in a manner similar to **rsh(1)**. However, it imposes multiple restrictions to maintain system security.

NAME

utreader - Sun Ray utility for configuring a terminal as a token reader.

SYNOPSIS

/opt/SUNWut/sbin/utreader

/opt/SUNWut/sbin/utreader -a *cid*

/opt/SUNWut/sbin/utreader -c

/opt/SUNWut/sbin/utreader -d *cid*

/opt/SUNWut/sbin/utreader -h

DESCRIPTION

The **utreader** command is used to designate Sun Ray terminals as dedicated token card readers.

When utreader is run with no options, the list of terminals currently configured as token readers within the current host group is displayed. The token reader list can be viewed by all users. Changes to the token reader configuration may be made only by a suitably privileged user.

Note:

After the token reader configuration been modified, a cold restart of services must be performed in order for the new configuration to take effect.

OPTIONS

The following options are supported.

-a *cid* Adds the specified terminal to the list of token card readers.

-c Clears (deletes all terminals from) the list of token card readers.

-d *cid* Deletes the specified terminal from the list of token card readers.

-h Displays the usage message.

A terminal is identified to the **-a** and **-d** options by its unique individual "canonical ID" or "CID". This is typically a string of the form **IEEE802.***EthernetMacID*. The **IEEE802.** portion of that string is the default CID prefix and may be omitted from the command line.

EXAMPLES

Example 1: This command configures a terminal whose ID is IEEE802.008020112233 as a token reader:

% **utreader -a 008020112233**

Example 2: This command unconfigures all token readers:

utreader -c

EXIT STATUS

The following exit values are returned:

0 Success

1 Failure

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuta

SEE ALSO

utrestart(1M), **utconfig(1M)**

NOTES

utreader is available only after **utconfig(1M)** has been run to activate the Sun Ray data store.

The number of token card readers that may be present on a host group's reader list at any one time may be limited by the implementation. Historically the limit has been eight readers. This implementation guarantees that at least 48 readers will be supported at one time. It is possible that more than 48 readers may be configured in some circumstances but a site should not depend on being able to exceed 48 in future SRSS releases.

NAME

utreplica - LDAP replication utility for Sun Ray servers.

SYNOPSIS

/opt/SUNWut/sbin/utreplica **-p [-a** | **-d]***secondary-server1* [*secondary-server2*

/opt/SUNWut/sbin/utreplica **-s** *primary-server*

/opt/SUNWut/sbin/utreplica **-l**

/opt/SUNWut/sbin/utreplica **-u**

/opt/SUNWut/sbin/utreplica **-z [***port#*]

DESCRIPTION

The **utreplica** command configures the Sun Ray LDAP server to replicate data from the primary server to each secondary server in a failover group. The command must be run with superuser privileges on the Sun Ray server to be configured.

OPTIONS

The following options are supported.

-l List the current failover administration status.

-p [-a|**-d]** *secondary-server*

Configure the primary server. *secondary-server* is the host name of the secondary server. List all seondary servers within the failover group.

-a is used to add the specified secondary servers to the current list of secondary servers.

-d is used to delete the specified secondary servers from the current list of secondary servers.

-s *primary-server*

Configure a secondary server. *primary-server* is the host name of the primary server.

-u Unconfigure this Sun Ray server for LDAP database replication.

-z[*port#*]

Update the port number with the one specified for the datastore service. If run on the primary server without specifying the port number, it simply updates all the necessary configuration files on the primary with the default port for the datastore service. If run on the secondary server without specifying the port number, it resyncs all necessary configuration files on the secondary server with the port number currently configured on the primary server.

USAGE

utreplica is used only on Sun Ray servers in a failover group. Configure the primary Sun Ray server first, then the secondary servers.

FILES

The following files are configured on the primary Sun Ray server:

- **/etc/opt/SUNWut/srds/current/utdsd.conf**
- **/etc/opt/SUNWut/srds/current/utdsd.ini**
- **/etc/services**
- **/etc/opt/SUNWut/utadmin.conf**

The following files are configured on the secondary Sun Ray server:

- **/etc/opt/SUNWut/srds/current/utdsd.conf**
- **/etc/opt/SUNWut/utadmin.conf**

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto

SEE ALSO

utconfig(1M)

NOTES

To replicate the LDAP information properly, all Sun Ray servers in the failover group must have the same group signature.

Use of the **-p**, **-s**, and **-u** options on a Sun Ray server interrupts all active sessions on that server.

Please note that "interrupt" in this context means that, while the screen will flash for approximately 30 seconds, all current sessions are preserved, not destroyed. Active users may need to remove and re-insert their smart cards to return to their sessions. If a CDE login screen is presented, the user should select Options/Reset Login Screen to return to an existing session.

NAME

utresadm - manage explicit monitor resolution settings for Sun Ray.

SYNOPSIS

/opt/SUNWut/sbin/utresadm [-v *vonum*]

/opt/SUNWut/sbin/utresadm -a [**-v** *vonum*] **-c** *CID* **-t** *token resname*

/opt/SUNWut/sbin/utresadm -d [**-v** *vonum*] **-c** *CID* **-t** *token*

/opt/SUNWut/sbin/utresadm -p [**-v** *vonum*] [**-c** *CID*] [**-t** *token*]

/opt/SUNWut/sbin/utresadm -o

/opt/SUNWut/sbin/utresadm -i

DESCRIPTION

The **utresadm** command allows an administrator to establish explicit video signal timings on the monitor video outputs of a Sun Ray desktop unit. The video signal timing controls the on-screen resolution and refresh rate of the attached monitor.

Resolutions configured through **utresadm** take precedence over resolutions discovered by the Sun Ray DTU through a DDC exchange with the monitor. Resolutions may be specified for a particular combination of Sun Ray DTU and access token, for a particular Sun Ray independent of access token, and for all Sun Ray DTUs controlled by servers in a failover group. In cases where multiple configuration records might apply to a given session, the most specific matching record is applied.

utresadm with no arguments shows the resolution that has been configured for the current access token in the current Sun Ray DTU.

OPTIONS

The following options are supported.

-a establishes the resolution *resname* as the preferred resolution for the given *CID*, *token* and *vonum*. If *token* is specified as the word **default** then *resname* becomes the preferred resolution for all tokens at the given *CID* for which no explicit resolution has been configured. If both *CID* and *token* are specified as the word **default** then *resname* becomes the preferred resolution for all CIDs and tokens for which no explicit resolution has been configured. This option is available only to the superuser. Changes become effective only after services are restarted.

-d deletes an explicit resolution previously configured for the given *CID*, *token* and *vonum*. This option is available only to the superuser. Changes become effective only after services are restarted.

-p shows the explicit resolution configured for the given *CID*, *token* and *vonum*. If no token is specified then all configuration records for the given *CID* are shown. If no *CID* is specified then all configuration records for the given *token* are shown. If neither *CID* nor *token* is specified then all configuration records are shown.

-o emits (to standard output) all configured resolution records in a format suitable for consumption by **utresadm -i**.

-i reads (from standard input) a list of resolution configuration records in the form emitted by **utresadm -o** and configures the corresponding resolutions. This option is available only to the superuser.

-c *CID*

CID is the canonical ID of the Sun Ray desktop unit, or the word **default** if this operation is to apply to all desktop units.

-t *token*

token is the access token for the session, or the word **default** if this operation is to apply to all tokens.

-v *vonum*

vonum is the monitor video output number that is to be affected by this operation. The first monitor video output on a Sun Ray DTU is numbered 1. Additional outputs, present only on certain models of Sun Ray DTU, are identified by successively higher numbers. (For example, a Sun Ray 2FS DTU has two outputs, numbered 1 and 2.) If no *vonum* is specified then output number 1 is assumed.

OPERANDS

resname

The name of a timing that will produce the desired resolution. A list of available timings may be obtained from **utresdef(1M)**.

EXIT STATUS

The following exit values are returned:

0 if the desired activity was completed without error.

1 if the command terminated because of a command line syntax problem.

2 if the Sun Ray data store was inaccessible.

3 if the resolution definition (provided on standard input) was unacceptable.

FILES

/etc/opt/SUNWut/utadmin.conf

/etc/opt/SUNWut/utadmin.pw

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuta

SEE ALSO

utsettings(1), **utset(1)**, **utresdef(1M)**, **utconfig(1M)**

NOTES

utresadm provides no assurance that the configured resolution timing can be understood by the monitor attached to a Sun Ray unit; it is possible to configure a resolution that can not be understood by the monitor. **utsettings(1)** provides a safer interactive means of establishing an explicit resolution for the current access token on the current Sun Ray unit.

utresadm is available only after **utconfig(1M)** has been run to activate the Sun Ray data store.

As with anything that must go through the LDAP database, there is a slight lag between the time an update is applied on the primary server and its propagation to the secondary server or servers; however, the length of the interval would normally be measured in seconds or fractions of a second.

In addition, once a Sun Ray session, which can persist across many logins, is attached to a given DTU, the monitor timing persists for the life of the session and can be changed only by modifying the timing explicitly from within that session while it remains attached to that DTU.External activities, such execution of **utresadm -a** or **utresadm -d** after the session has been associated with the DTU, have no effect on any existing association. However, **utrestart -c** destroys all existing sessions and thereby makes all previously existing associations obsolete. As DTUs reconnect after the restart and are granted new sessions, the newly configured timings are applied to the new sessions as they are created.

NAME

utresdef - manage monitor resolution definitions for Sun Ray.

SYNOPSIS

/opt/SUNWut/sbin/utresdef

/opt/SUNWut/sbin/utresdef *resname*

/opt/SUNWut/sbin/utresdef -a [**-c** *comment*] *resname dimensions*

/opt/SUNWut/sbin/utresdef -d *resname*

/opt/SUNWut/sbin/utresdef -o

/opt/SUNWut/sbin/utresdef -i

/opt/SUNWut/sbin/utresdef -h

DESCRIPTION

The **utresdef** command allows an administrator to create, delete and view resolution definitions (actually monitor signal timing definitions) for monitors attached to Sun Ray DTUs.

Running **utresdef(1M)** with no options or arguments produces a summary listing of all available resolutions. Running **utresdef(1M)** with no options and a *resname* argument produces a summary listing of the named resolution. The summary listing shows the name of the resolution, the on-screen dimensions (*WIDTHxHEIGHT*) that result from the use of this resolution, a flag indicating whether this resolution is built in (**B**) to SRSS or has been locally defined (**-**), and a brief description of the resolution.

Individual resolutions may be associated with specific Sun Ray units by **utresadm(1M).** Users may configure resolutions for their own personal access tokens through **utsettings(1)** or **utset(1)**.

OPTIONS

The following options are supported.

-a defines the resolution *resname*. This option is available only to the superuser.

-d deletes the resolution definition named *resname*. This option is available only to the superuser.

-o shows all locally-defined resolutions in a format suitable for consumption by **utresdef -i**.

-i reads (from standard input) a list of resolution definitions in the form emitted by **utresdef -o** and adds those definitions to the list of locally-defined resolutions. This option is available only to the superuser.

-c a **comment** explaining the nature and purpose of the resolution being defined.

-h Displays the usage message for this command.

OPERANDS

dimensions

the dimensions, in the form *WIDTHxHEIGHT*, of the on-screen display that results from applying this resolution definition to the Sun Ray desktop unit.

resname

the name of the resolution being defined, deleted or displayed. *resname* is often given in the form *WIDTHxHEIGHT@RATE* (e.g. 640x480@60) but this form is not required. The resolution name must begin with a letter or a number, may contain letters, numbers and the characters ´.´ (period), ´-´ (hyphen), ´_´ (underscore), ´@´ (at) or ´+´ (plus), and must not exceed 20 characters in length.

EXIT STATUS

The following exit values are returned:

0 if the desired activity was completed without error.

1 if the command terminated because of a command line syntax problem.

2 if the Sun Ray data store was inaccessible.

3 if the resolution definition (provided on standard input) was unacceptable.

FILES

/etc/opt/SUNWut/utadmin.conf

/etc/opt/SUNWut/utadmin.pw

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuta

SEE ALSO

utsettings(1), **utset(1)**, **utresadm(1M)**, **utconfig(1M)**

NOTES

utresdef is available only after **utconfig(1M)** has been run to activate the Sun Ray data store.

NAME

utrestart - Sun Ray utility for resetting and restarting services.

SYNOPSIS

/opt/SUNWut/sbin/utrestart

/opt/SUNWut/sbin/utrestart -c

/opt/SUNWut/sbin/utrestart -h

DESCRIPTION

The **utrestart** command is used for resetting and restarting Sun Ray services. It replaces the **utpolicy -i** option which has been deprecated in 2.0.

utrestart can only be run by the super-user.

The utrestart command without options causes a "warm" restart: Sun Ray services are restarted and existing sessions are preserved.

OPTIONS

The following options are supported.

-c Restarts Sun Ray services. Sessions will be lost.

-h Prints the usage for this command.

EXAMPLES

Example 1: This resets services.

/opt/SUNWut/sbin/utrestart

Example 2: This restarts services.

/opt/SUNWut/sbin/utrestart -c

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuta

NAME

utselect - Sun Ray failover group server selection tool.

SYNOPSIS

/opt/SUNWut/bin/utselect [**-L**] [**-R**] [**-S**] [**-X**]

DESCRIPTION

The **utselect** command is a graphical user interface (GUI) to the **utswitch** command. It allows the user to select manually to which Sun Ray server or session the Sun Ray DTU is to connect. The sessions in the GUI are sorted in order of most current. The second item in the list is highlighted by default to allow easy switching between two servers. The Refresh button executes the **utswitch -l** command and updates the information displayed in the GUI. The OK button executes a **utswitch -h** command to the server highlighted.

OPTIONS

The following options are supported.

-L Configures **utselect** to run in "login" mode before the log in screen is displayed. Where:

- If only one server is available, the command exits.
- The current server is set as the default.
- Selecting the current server causes the command to exit
- The locale is determined in a manner similar to CDE
- The screen is centered in the display

-R Remote server selection is enabled. This enables an entry field where a networked server name can be entered.

-S Remote server selection is disabled.

-X Exit after making a selection from the list.

EXAMPLES

Example 1: This command enables users to select which Sun Ray server or session to connect. The GUI exits after selection

% **/opt/SUNWut/bin/utselect -X**

FILES

Two properties in the **auth.props(4)**file impact the operation of utselect:

selectAtLogin=true	This setting will cause utselect to run before dtlogin and allow users to start their session on a particular machine. The default value is "false"
remoteSelect=true	This setting will cause utselect to behave as if the -R option were specified. This will allow users to input the name of a server outside of the default HA group, if the Sun Ray DTU can connect to it (i.e. the Sun Ray can be routed to the server, such as in a LAN deployment). The default value is false.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto

SEE ALSO

utswitch(1), **attributes(5)**, **auth.props(4)**

NAME

utserial, utparallel - Sun Ray serial and parallel port device driver emulators.

SYNOPSIS

#include <sys/types.h>

#include <fcntl.h>

utserial

#include <sys/termios.h>

#include <termio.h>

utparallel

#include <sys/ecppio.h>

DESCRIPTION

utserial is a tty-style interface that provides a generic virtual interface to USB serial adaptors connected to the Sun Ray DTU.

utparallel is a parallel-style interface that provides a generic virtual interface to USB parallel adaptors connected to the Sun Ray DTU.

utserial and **utparallel** are each loadable **STREAMS** drivers.

EXTENDED DESCRIPTION

The actual interface to the DTU for each of these drivers is through the Sun Ray interconnect via either the **utseriald** daemon or the **utparalleld** daemon, each of which is session-aware. The daemons are connected to either **utserial** or **utparallel** through a master port and each is responsible for creating the slave device nodes through which normal applications will connect.

API

Applications open a device file created by either **utseriald** or **utparalleld**. Device files created by **utseriald** comply to the **termio**(7I) interface and device files created by **utparalleld** comply to the **ecpp**(7D) interface. Hardware limitations in USB adaptors might prevent compliance with these interfaces.

FILES

The following files are used:

- **/dev/utserial**

Master port for **utserial**

- **/dev/utparallel**

Master port for **utparallel**

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWutu
MT-Level	Safe

SEE ALSO

utseriald(1M), **utparalleld(1M)**, **termio(7I)**, **ecpp(7D)**

NAME

utseriald - Sun Ray DTU serial services daemon.

SYNOPSIS

/opt/SUNWut/lib/utseriald [**-D** *debug-level*] [**-o** *optroot*] [**-r**]

DESCRIPTION

utseriald provides serial support for Sun Ray DTUs through driver services for USB serial adaptors. It also provides support for the built-in, non-USB serial ports on some Sun Ray models.

utseriald uses the Sun Ray serial I/O driver on solaris and Sun Ray utio driver on linux to provide applications the same interface as standard workstation serial ports, such as **/dev/term/a** or **/dev/term/b** on solaris and **/dev/ttyS0** on linux.

When a USB serial adaptor is attached to a DTU, **utseriald** creates device nodes in the **$UTDEVROOT/devices** directory. Sun Ray models with permanent, built-in serial ports will also have corresponding device nodes created in this directory.

A user can use the device link **$UTDEVROOT/dev/term/***terminal-name* to access the serial ports on the Sun Ray DTU.

OPTIONS

The following options are supported.

-D *debug-level*

Debug mode. Use is beyond the scope of this document.

-o *optroot*

Use *optroot* as the serial service's root directory for device node creation. The default value is **/tmp/SUNWut**. *optroot* should be the same directory as the *optroot* directory used by **utdevmgrd**(1M).

-r Automatically restart the serial service daemon if it exits. With this option, the serial service daemon creates two processes: a child that performs all the actual work, and a parent monitoring process. The parent process restarts a child if the previous one exits.

FILES

The following files are used:

$UTDEVROOT/**dev/term**

The directory containing links to serial device names for each DTU.

ENVIRONMENT VARIABLES

UTDEVROOT points to a symbolic link of the device root for the Sun Ray DTU associated with a user's session.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto

SEE ALSO

utdevmgrd(1M), **utparalleld(1M)**

NAME

utsession - List and manage the Sun Ray sessions on the local Sun Ray server.

SYNOPSIS

/opt/SUNWut/sbin/utsession -p [**-x**] [**-d** *disp#*] [**-u** *unix*] [**-t** *token*] [**-n** *name*]

/opt/SUNWut/sbin/utsession -k [**-a**] [**-x**] [**-d** *disp#*] [**-u** *unix*] [**-t** *token*] [**-n** *name*]

/opt/SUNWut/sbin/utsession -l [**-x**] [**-d** *disp#*] [**-u** *unix*] [**-t** *token*] [**-n** *name*]

/opt/SUNWut/sbin/utsession -h

DESCRIPTION

The first synopsis (**-p**) is used to print Sun Ray sessions for the specified user or token on the current server. When the **-u**, **-t**, **-n**, and **-d** options are not used, **utsession** prints all of the Sun Ray sessions on the current server.

When listing the Sun Ray sessions, **utsession** also lists the state for each session:

D Disconnected — The session is currently not attached to any Sun Ray. The session is considered connected if this flag is omitted.

I Idling — A **dtlogin** session that is currently waiting for user login (ie. **dtgreet**). User has already logged into the **dtlogin** session if the flag is omitted.

The second synopsis (**-k**) is used to kill Sun Ray sessions on the current server. At least one of the **-d**, **-u**, **-t**, or **-n** options must be specified. Unless the **-a** option is specified, more than one session matching the specified criteria will return an error.

The third synopsis (**-l**) gives detailed security status for the specified user or token on the current server. When the -u, -t, -n, and -d options are not used, utsession lists detailed security status for all terminal CIDs currently connected to the server.

The fourth synopsis (**-h**) displays the usage of this command.

Note:

This command must be run as root.

OPTIONS

The following options are supported:

-a Apply the operation to all matching sessions if more than one matches the search criteria. If **-a** is not specified, multiple matching sessions return an error.

-d *disp#*
Specify the X display number for search.

-h Display the usage of this command.

-k Kill the sessions matching the search criteria. You must also specify at least one of the **-d**, **-u**, **-t**, or **-n** options.

-l Gives detailed security status, normally a list of terminal CIDs currently used by the session, listed under the terminalCIDs key. For multihead sessions, it also displays the primary terminal CID under the primaryCID key.

-n *name*
Specify the registered Sun Ray username for search. Sessions belonging to users matching the username are listed. It is a case sensitive, exact match.

-p Print the sessions belonging to the specified user or token.

-t *token*
Specify the Sun Ray token for search. The token is in one of the following forms:

- Raw token form for unregistered users (**MicroPayflex.**####)

- Pseudo token form for terminal users (**pseudo.***macaddr*)
- Logical token form for registered users (**user.**#####)
- Mobile token form for NSC mobile users (**mobile.***username*)

-u *unix*

Specify the UNIX login name for search.

-x RESERVED for special handling. This is invoked by **utrcmd(1M)** to support remote operation of the Administration Tool interface.

EXAMPLES

Example 1: This command lists all sessions on the current server.

utsession -p

Example 2: This command finds the sessions for the UNIX user "jdoe".

utsession -p -u jdoe

Example 3: This command terminates a registered Sun Ray user's (john doe's) session.

utsession -k -n "john doe"

EXIT STATUS

The following exit values are returned:

0 Command completed successfully.

1 Entry not found.

-1 An error occurred.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuta

SEE ALSO

utuser(1M), **auth.props(4)**, **utdesktop(1M)**

NAME

utsessiond - Sun Ray session manager daemon.

SYNOPSIS

/opt/SUNWut/lib/utsessiond [**-a** *authlist*] [**-c** *authfile*] [**-d**] [**-h** *hostname*] [**-p** *port*] [**-P** *nport*] [**-r**] [**-t**]

DESCRIPTION

The **utsessiond** daemon provides a reliable rendezvous point for services in a Sun Ray session. It acts as an intermediary to forward session connection and disconnection messages from the Sun Ray authentication manager to the services and provides facilities for supporting distributed synchronization of clip-lists for the services.

If either the **-a** or the **-c** options is specified, the session manager daemon operates exclusively in call-back mode. In this mode, the session manager only takes session connect and disconnect commands from authentication managers that are explicitly enabled by *authlist* or *authfile* and that have requested a call-back. The call-back feature provides a mechanism by which the session manager and the authentication manager may establish each other's identity.

Error messages from **utsessiond** are logged using **syslog**(3) with a facility value of LOG_DAEMON.

OPTIONS

The following options are supported.

-a *authlist*
: Add the host and port pairs specified in *authlist* to the list of permitted Authentication Managers. The format of *authlist* is a comma separated list of *hostname*:*port* pairs.

-c *authfile*
: Add the host and port pairs specified in the ASCII file *authfile* to the list of permitted Authentication Managers. The file contains a list of Authentication Manager specifications, one per line. The specifications take the form of *hostname* followed by *port* number, separated by white-space. Blank lines and any line whose first printable character is "#" are ignored.

-d Enable debugging output.

-h *hostname*
: Set the *hostname* portion of the session IDs generated by the Session Manager to the specified *hostname* value. By default this is set to the machine's node name. This option can be used to handle servers supporting multiple IP addresses as part of a clustering solution.

-p *port*
: Set the Session Manager's listen port to the specified *port* value. The default is port 7007. This is the port by which session services and Authentication Managers contact the Session Manager.

-P *nport*
: This option is no longer used. Retained only for backward compatibility.

-r Automatically restart the Session Manager daemon if it exits. With this option the Session Manager daemon will create two processes: a child that performs all the actual work and parent monitoring process. The parent process will restart a child if the previous one exits.

-t Test mode. Use is beyond the scope of this document.

FILES

The following file is used:

- **/etc/opt/SUNWut/auth.permit**

The customary location of the **authfile** for a system.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto

SEE ALSO

utauthd(1M), **syslog(3)**, **syslogd(1M)**, **syslog.conf(4)**

NAME

utset - view or change the Sun Ray DTU device settings.

SYNOPSIS

/opt/SUNWut/bin/utset [**-d** *arg[,...]* [**-f**]] [**-i** *arg[,...]*] [**-o** *arg[,...]*] [**-v** *arg[,...]*]

/opt/SUNWut/bin/utset -l

DESCRIPTION

The **utset** command reports or modifies Sun Ray device settings. **utset** executed with no options prints the current status of all device settings to the standard output stream. **utset** executed with the **-d**, **-i**, **-o** or **-v** options modifies the values of the specified settings. **utset** executed with the **-l** option prints a list of predefined display timing names to the standard output stream.

Sun Ray device settings are grouped into four categories: Display, Audio Input, Audio Output, and Video Input. The setting to be modified is selected by an option letter, which identifies the category, and an argument consisting of a list of individual parameter settings within that category and the values that are desired for each of those parameters. The argument is a comma separated list of '*name=value*' pairs, where *name* specifies a device parameter and *value* specifies the desired value of that parameter.

Parameters reported as **read only** by **utset** may not be modified. Read-only status usually reflects some limitation of a particular model of DTU. For instance, the Sun Ray 150 supports only one display timing and therefore reports its display timing setting as a read-only parameter.

The **utsettings**(1) command provides equivalent device parameter reporting and modification capabilities in a GUI format.

The **xset**(1) command provides control over device parameters such as mouse acceleration which are reported by **utset** but may not be modified by **utset**.

OPTIONS

The following options are supported.

-d *arg[,...]* [**-f**]

Set the Display timing:

r[*vonum*]=*timing*

Display timing: *vonum* is the number of the monitor video output whose timing will be modified. The DTU's first (and perhaps only) monitor video output is number 1. Additional outputs, if present, are identified by successively higher numbers. If *vonum* is omitted then output number 1 is assumed. *timing* is the name of a timing. The **-l** option gives the list of available timings. Timing names are usually of the form *W*x*H*@*R* where *W* is the resulting on-screen width (in pixels), *H* is the resulting on-screen height (in pixels) and *R* is the resulting refresh rate (in Hz). **utresdef**(1m) provides additional information on monitor timings.

If **-f** is not given then after the new timing is put into effect the user is asked to confirm that the new Resolution/Refresh Rate is acceptable. If no confirmation is provided within a certain interval (currently 15 seconds) then the timing is returned to its previous setting.

-i *arg[,...]*

Set the Audio Input settings:

s=[m|l]	Input Select: microphone, line.
g=<0:75>	Microphone Gain: a range of 0 to 75.
l=<0:15>	Line In Gain Left: a range of 0 to 15.
r=<0:15>	Line In Gain Right: a range of 0 to 15.
v=<0:64>	Volume: a range of 0 to 64.

-l Show the list of predefined display timing names.

-o *arg[,...]*

Set the Audio Output settings:

s=[a|s|h]	Output Select: auto, speaker, headphone.
v=<0:31>	Volume: a range of 0 to 31.
b=<-32:32>	Balance: a range of -32 to +32.
m=[on|off]	Mute: on, off (also: 1, 0 respectively).
e=[on|off]	Stereo Enhance: on, off (also: 1, 0 respectively).
T=<-6:6>	Treble: a range of -6 to +6.
B=<-6:6>	Bass: a range of -6 to +6.

-v *arg[,...]*

Set the Video Input settings:

b=<0:255>	Brightness: a range of 0 to 255.
c=<0:63>	Contrast: a range of 0 to 63.
C=<0:127>	Color: a range of 0 to 127.
t=<0:255>	Tint: a range of 0 to 255.
f=<0:3>	Filter: a range of 0 to 3.
T=[on|off]	Color Trap: on, off (also: 1, 0 respectively).

EXAMPLES

Example 1: This command displays the current device settings for the Sun Ray DTU to which your session is attached:

```
% utset
Version: SunRayP3-3.1_10,REV=2005.03.30.15.44 (read only)
Audio Input Mic Gain: 58
Audio Input Line In Gain L: 0
Audio Input Line In Gain R: 0
Audio Input Select: Microphone
Audio Input Monitor Volume: 0
Audio Output Volume: 15
Audio Output Balance: 0
Audio Output Treble: 0
Audio Output Bass: 0
Audio Output Select: Auto
Audio Output Headphone Detected: no
Audio Output Mute: off
Audio Output Stereo Enhance: off
Mouse Threshold: 4
Mouse Acceleration: 2.0
Display Timing 1: 1024x768@60 (read only)
Display Blanking: off
Video Brightness: 128
Video Contrast: 7
Video Color: 46
Video Tint: 128
Video Filter: 3
Video Color Trap: on
```

This session happens to be attached to a Sun Ray 150, which has only one (internal) monitor video output

whose whose display timing is fixed and is therefore reported as a read-only parameter.

EXIT STATUS

The following exit values are returned:

0 Success

1 Failure

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto

SEE ALSO

utresdef(1m), **utsettings**(1), **xset**(1)

NAME

utsettings - View or change the Sun Ray DTU settings.

SYNOPSIS

/opt/SUNWut/bin/utsettings [**-H** [**-k** *hotkey*]]

DESCRIPTION

The **utsettings** command opens a Sun Ray Settings dialog box that allows the user to view or change audio, visual, and tactile interface settings for the Sun Ray DTU.

The **utsettings** application connects to the Session Manager, which tells the application which DTU it is being displayed upon. As the user moves a session from one Sun Ray DTU to another, the Session Manager keeps track of the session's current location and instructs the **utsettings** application to follow. With each session move, the **utsettings** application displays the current DTU's configuration.

By default, the Sun Ray server starts an instance of **utslaunch(1M)** for each session created by the user logging in via **dtlogin**. This enables the Sun Ray Settings dialog box to be available at the press of a hotkey or key combination. Subsequent presses toggle the dialog box on and off.

Users can initiate similar functionality by running **utsettings** with the **-H** flag. The hotkey can be specified using the **-k** option. Only one instance of **utsettings -H** or **utslaunch** can be running per session.

Settings selected through **utsettings** apply only to the DTU where **utsettings** is run; hotdesking to another DTU does not bring the new timing along as part of the session. However, the selected timing is retained and used again if a user hot desks back to the original DTU.

If the session is associated with a personal mobile token (a smart card or an NSCM credential), then **utsettings** offers to make the selected timing permanent. If a user accepts that offer, then the timing is retained and reused on that user's subsequent personal mobile token sessions on the same DTU. For shared session token types, such as *pseudo*, users are not allowed to establish long-term resolution settings because their settings would interfere with other people's use of the DTU.

OPTIONS

The following options are supported.

-H Start the **utsettings** application in *hotkey* mode. The **utsettings** application starts with the Sun Ray Settings dialog box hidden. Pressing the hotkey toggles the display of the dialog box. The default hotkey key combination is Shift + Props. The hotkey can be user or site defined according to the **utsettings.hotkey** property in the files listed below. See FILES

-k *hotkey*

Use the specified key or keys as the hotkey combinations when the **-H** option is specified. This option is dependent upon the **-H** option.

EXAMPLES

Example 1: This command displays the settings for the Sun Ray DTU you are currently logged into.

% **utsettings**

FILES

The following files are used:

- **/etc/opt/SUNWut/utslaunch_defaults.properties**

site-wide defaults

- **~/.utslauch.properties**

user's defaults

- **/etc/opt/SUNWut/utslaunch_mandatory.properties**

site-wide mandatory defaults

EXIT STATUS

The following exit values are returned:

0 Success

1 Failure

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto

SEE ALSO

utslaunch(1M), **dtlogin(1X)**, **dtsession(1X)**, **utslaunch.properties(4**)

NAME

utslaunch - Sun Ray DTU launch application.

SYNOPSIS

/opt/SUNWut/lib/utslaunch

DESCRIPTION

The **utslaunch** application is used to launch various Sun Ray applications via a "hotkey" key combination. The applications are enabled when the key combination is pressed.

The **utslaunch** application provides hotkey functionality while consuming fewer system resources.

Hotkey key combinations are defined in the **utslaunch.properties** files.

OPTIONS

There are no options for **utslaunch**.

EXAMPLES

Example 1: This command starts the utslaunch daemon in the background.

utslaunch &

FILES

The following files are used for hotkey configuration:

- **/etc/opt/SUNWut/utslaunch_defaults.properties**

site-wide defaults

- **~/.utslaunch.properties**

user's defaults

- **/etc/opt/SUNWut/utslaunch_mandatory.properties**

site-wide mandatory defaults

The following file is used:

/usr/dt/config/Xsession.d/0100.SUNWut

ENVIRONMENT VARIABLES

utslaunch uses the **DISPLAY** environment variable to get the user's X display number.

It also uses the **HOME** environment variable to get the user's home directory to be able to use user's hotkey settings.

EXIT STATUS

The following exit values are returned:

0 Success

1 Failure

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto

SEE ALSO

utslaunch.properties(4), **utsettings(1)**, **utdetach(1)**

NAME

utslaunch.properties - Default hotkey key combinations for various applications supported by **utslaunch**.

SYNOPSIS

/etc/opt/SUNWut/utslaunch_defaults.properties

~/.utslaunch.properties

/etc/opt/SUNWut/utslaunch_mandatory.properties

DESCRIPTION

The files listed above are standard Java properties files that can contain defaults which customize the operation of the **utslaunch** application. Each file contains entries in the format of:

name=value

where *name* is the property name and *value* is the value set.

EXTENDED DESCRIPTION

When the **utslaunch** application starts, it looks for and reads the properties files in the order listed below. Note that a hotkey key combination specified in a file can be overridden by a file read later in the search order.

1. **/etc/opt/SUNWut/utslaunch_defaults.properties**

   This file contains site-wide default properties which are used if the user has not specified any. These properties override any application defaults.

2. **~/.utslaunch.properties**

   This file contains the user's default properties. These properties override application and site-wide default properties.

3. **/etc/opt/SUNWut/utslaunch_mandatory.properties**

   This file contains site-wide mandatory default properties which supersede any application, site-wide, or user defaults.

PROPERTIES

The supported application properties are listed below. For each property, the name, description, application default, and some examples are given.

Name — **utdetach.hotkey**

Description — Specifies the hotkey or key combination that disconnects the current session from the DTU the user is currently using. The value is a valid X keysym name preceded by one or more of the supported modifiers (Ctrl, Shift, Alt, Meta).

Application Default — Shift Pause (Hold down Shift and press the Pause key)

Name — **utsettings.hotkey**

Description — Specifies the hotkey or key combination that invokes the Sun Ray Settings dialog box. The value is a valid X keysym name preceded by one or more of the supported modifiers (Ctrl, Shift, Alt, Meta).

Application Default — Shift SunProps (Hold down Shift and press the Props key)

Examples:

- F3
- Shift F4
- Ctrl Shift Alt F5

EXAMPLES

Example 1: The following is a sample of the contents of a properties file.

utdetach.hotkey=Shift Pause
utsettings.hotkey=Shift SunProps

ATTRIBUTES

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

Attribute Types	Attribute Values
Availability	SUNWutr
Stability Level	Evolving

SEE ALSO

utslaunch(1M), **utdetach(1M)**

NAME

utstoraged - Sun Ray Mass Storage service daemon

SYNOPSIS

/opt/SUNWut/lib/utstoraged [-D *debug-level*] [-o *optroot*] [-r]

DESCRIPTION

utstoraged provides mass storage support for Sun Ray DTUs. **utstoraged** supplies driver services for USB mass storage devices that comply with the USB Bulk Only Mass Storage Specification 1.0.

utstoraged uses the **utdisk**(7D) loopback driver to provide the **dkio**(7I) interface to Solaris applications. Applications can access Sun Ray storage devices through block and raw device links created in the $**UTDEVROOT/dev/dsk** and $**UTDEVROOT/dev/rdsk** directories respectively. **utstoraged** interacts with the mounter daemon, **utmountd**(1M), to mount devices containing OS-recognizable filesystems.

Users have access rights only to storage devices connected to the Sun Ray DTU on which their session is active and only for as long as the session is active. When the session is disconnected from the Sun Ray DTU through logout, hotdesking, server switching, or any other means, ownership of the storage device is lost and all pending data transfers are aborted. This can result in corruption of the filesystem on the media and loss of data.

Note:

Users are strongly advised to use the **utdiskadm**(1M) command to prepare the device for removal before disconnecting a session.

Error messages are logged to **/var/opt/SUNWut/log/utstoraged.log**

OPTIONS

The following options are supported:

-D *debug-level*

Debug mode. Use is beyond the scope of this document. If a debug level is set, debug messages are sent to stderr.

-o *optroot*

Use **optroot** as the parallel service's root directory for device node creation. **optroot** should be the same directory as the **optroot** directory used by **utdevmgrd**(1M).

-r

Automatically restart the storage service daemon if it exits. With this option, the storage service daemon creates two processes: a child that performs all the actual work, and a parent monitoring process. The parent process restarts a child if the previous one exits.

FILES

The following files are used:

$UTDEVROOT/dev/dsk

The directory containing links to block device names for each slice or partition on the device.

$UTDEVROOT/dev/rdsk

The directory containing links to raw device names for each slice or partition on the device.

ENVIRONMENT VARIABLES

UTDEVROOT points to a symbolic link of the device root for the Sun Ray DTU associated with a user's session.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWutsto

SEE ALSO

utdiskadm(1M), **utmountd(1M)**, **utdevmgrd(1M)**, **utdisk(7D)**

NAME

utsunmc - Sun Ray server software module for the Sun Management Center, providing addition, load, and removal utilities.

SYNOPSIS

/opt/SUNWut/sbin/utsunmc [**-u**]

DESCRIPTION

The **utsunmc** command adds the Sun Ray server software module to the Sun Management Center (SunMC) and loads it to permit monitoring of the Sun Ray software. The **utsunmc** command can also remove the Sun Ray server software module from the SunMC.

The **utsunmc** command is run with superuser privileges.

OPTIONS

The following option is supported.

-u Remove the previously added and loaded Sun Ray server software module.

Without arguments, addition and load of the Sun Ray server software module is performed.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWutesa

SEE ALSO

Sun Management Center 3.0 Software User’s Guide

NOTES

The **utsunmc** command requires the Sun Management Center 3.0 or Sun Management Center 2.1.1 to be installed.

The Sun Management Center agent is stopped while this command runs and restarted after the command completes. The agent may not properly restart on Sun Management Center 2.1.1. In this case, the command **/opt/SUNWsymon/sbin/es-start -a** should be run.

NAME

utswitch - Sun Ray server selection and session listing utility.

SYNOPSIS

/opt/SUNWut/bin/utswitch { **-l** | **-t** | **-h** *hostname* } [**-k** *token*] [**-p** *port*] [**-r**]

DESCRIPTION

The **utswitch** command allows switching a Sun Ray DTU among Sun Ray servers in a failover group. It can also list the existing sessions for the current token. One of the following option flags must be specified: **-l**, **-t**, or **-h**. The **utselect**(1) command implements a GUI-based interface to this command.

OPTIONS

The following options are supported.

-h *hostname*

Force an explicit switch to the server with *hostname*.

-k *token*

Specify the token ID *token* to be used in collecting session information from the servers in the failover group. The token normally used is the one connected to the current session.

-l List the servers accessible from the current Sun Ray DTU for the current token and show any existing sessions on those servers.

- The first field of the output is the server name.
- The second field is the X display number for an active user session. If no active user session exists, then -1 is printed or -2 is printed if the login screen is being displayed.
- The third field is the last connection time to an existing session, as a time value from the **time(2)** system call. If there is no session, the third field indicates status from the host as:

  -1 Server is up, but there is no session.

  -2 No response received from the server.

  -3 No path from the Sun Ray to the server.
- The fourth field is 1 if the server is offline and 0 otherwise.

-p *port*

Sets the port number of the Authentication Manager on the Sun Ray server to *port*, instead of the default 7009.

-r Forces a remote redirection outside of the current failover group to search for an existing session within an external failover group. If no session is available, load balancing is performed. Without this option, the Sun Ray DTU is bound explicitly to the target Sun Ray server, rather than to an appropriate server within the target failover group. This option may only be used with the **-h** option

-t Switch to the server whose session has the latest connection time among the existing sessions for the current token. Normally this would switch to the current session, so it has limited usefulness. However, it is useful in the case of logging out of an existing X session and back to the login screen. The connection time of a logged out session is biased back in time so that the session will not be selected if there is an existing logged-in X session on another server. From a CDE login screen, it is possible to force a call to **utswitch -t** by selecting Reset Login Screen from the Options menu. This allows switching back to a logged-in session from the login screen without having to log in.

FILES

The following files are used:

- **/var/opt/SUNWut/displays/***

X display files

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto

SEE ALSO

utselect(1), **attributes(5)**

NAME

uttsc - A terminal server client for the Sun Ray environment.

SYNOPSIS

uttsc [options] server

DESCRIPTION

uttsc is a Remote Desktop Protocol (RDP) client that enables users to launch and use Windows application from a Sun Ray desktop.

OPTIONS

The following options are supported.

-a *application*
Launch the Windows desktop with the specified application.

-b Disable the windows pulldown header in full-screen mode. This option can only be used when full screen mode (-m) is specified.

-c *working folder*
Needs -a provided as well. Applications specified with -a launch from the folder specified with -c.

-d *domain*
Domain for authentication. Needs -u provided as well.

-g *<widthxheight>[+|-<xoffset>+|-<yoffset>]*
Width and height specify size of Connector window. X offset specifies distance of top left corner of Connector Window from left edge of screen if positive and distance from right edge of screen if negative. Y offset specifies distance of top left corner of Connector window from top edge of screen if positive and distance from right edge of screen if negative. The -g option can not be provided together with the -m option.

-h Display complete usage of this command.

-i Read password from STDIN. It prompts for password with echo off if STDIN is a TTY, otherwise no prompt is displayed. This option can only be used when username option (-u) is specified and -p option is not used.

-k *keyboard*
Specify the keyboard type to be used for the connection. The supported values for keyboard are sun(type6), sun(type6jp) and sun(kr). The default keyboard is "sun(type6)". All Sun and PC USB keyboards should be supported by this default value, apart from Sun type6 Japanese, the Sun Korean keyboard and the PC Korean keyboard. The sun(type6jp) value is used for the Sun type6 Japanese keyboard. The sun(kr) value is used for the Sun Korean keyboard and the PC Korean keyboard.

-l *locale*
Specify the locale to be used for the Windows session. If this option is not specified, the LC_ALL or the LANG environment variable values are used. If these are not set, then "en-US" is the default locale. Any valid Windows locale can be specified in the standard format, eg: German is de-DE, Swiss French is fr-CH etc.

-m Enable fullscreen mode. This overrides the window manager and causes the uttsc client window to fully cover the current screen. It can not be provided together with the -g option.

-n *client name*
Specify a name to be used for this client. The DTU mac address is used as the default.

-O Enable optimized hotdesking behaviour. In this mode, hotdesking the Sun Ray session, does not relaunch the Sun Ray Connector.

-p Read password from controlling TTY. It prompts for password with echo off. This option can only be used when username option (-u) is specified and -i option is not used.

-r *device*
Allow device redirection of specified device on the client for access on the server. The devices currently allowed are :

-r *comport:<comport>=<device>,...*
Redirects serial devices on your client to the server.

-r *disk:<drive name>=<path>,...*
Redirects a path to the share \\tsclient\<drive name> on the server.

-r *printer:<printername>[=<driver>],...*
Redirects a printer queue on the client to the server. The <printername> is the name of the queue in your local system. <driver> defaults to a simple postscript driver unless you specify one. The first printer on the command line will be set as your default printer. If the driver name has white spaces, it must be quoted.

-r *sound:[low|high|off]*
Disable sound redirection from the server to the client or change the quality of transmitted sound. The sound quality in terms of bits per second can be specified . A "low" quality transmits 8khz and a "high"
quality does 22.2 khz. By default, High quality sound is enabled.

-r *scard:on*
Enable Smart Card redirection from client to the server. By default, Smart Card redirection is disabled.

-s Connects to the console session of a server.

-t *Timeout*
Specify the TCP connection timeout (in seconds) for the connection to the windows server. If this value is not specified or the Timeout value is set to zero, the system default value for the TCP connection timeout period will be used.

-u *username*
Username to use for authentication on the server.

-v Display version information of the uttsc command line. It can not be provided together with any other option.

-x *XDisplay*
Specify the X display to launch the session on. The DISPLAY environment variable is used to determine the display when this option is not used.

-z Disable RDP packet data compression. This is enabled by default.

-A *color depth*
Sets the colour depth for the connection (8, 15, 16 or 24). The colour depth may be limited by the server configuration in which case the server configuration is honored.

-B Disable backing store. Backing store is enabled by default if the X server supports it, this option disables it. Enabled backing store provides performance improvements during large and fast screen updates.

-C Use shared colourmap. By default, private colormap is used.

-E *window-attribute*
Enable window attributes from the defined set. The available set of options which can be enabled are : wallpaper, fullwindowdrag, menuanimations, theming, cursorshadow, cursorsettings. Keeping these attributes disabled improves display performance especially over lower bandwidth networks. Multiple -E options can be specified for more than one attribute if required.

-D *window-attribute*
Window attributes are disabled by default. The -D option is deprecated, ignored and not considered an error. It will be removed and considered an error in the next release.

-K Handle shortcuts in the local computer desktop session. If not specified then shortcuts are handled in the Windows server session. This option cannot be used in full screen mode.

-P *port* Specify the port over which to connect to RDP. The default port is 3389.

-T *window title*
Set the title on the client window. This option cannot be used when -b is specified in full screen mode.

Options taking values can not be provided multiple times.

EXAMPLES

1. An example to redirect a printer named "printer1" which uses the hp deskjet 990c driver, to a Windows session on say winserver1 :

% uttsc -u user1 -r printer:printer1="hp deskjet 990c" winserver1

EXIT STATUS

0 if operation performed successfully, 1 if error found

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuttsc

SEE ALSO

uttscadm(1m), **uttscprinteradm(1m)**, **utlicenseadm(1m)**

NAME

uttscadm - Sun Ray Connector for Windows Configuration utility

SYNOPSIS

uttscadm -c | -u | -h

DESCRIPTION

uttscadm is a utility used to configure the Sun Ray Connector for Windows. This command must be run with root permissions after installation to prepare the system to launch the Connector.

OPTIONS

The following options are supported.

-c This option configures the system to launch the Connector. It creates the required SRDS schema to store the printer configurations. It also prompts the administrator for the path to the OpenSSL library location, if required.

On Solaris, it also creates the required entries in the /etc/services file for the Sun Ray Connector Proxy daemon, and thereafter launches it.

-u This option is used to unconfigure the changes created during configuration, before removing the Connector.

-h Displays command usage information

EXIT STATUS

0 if operation performed successfully, 1 if error found

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuttsc

SEE ALSO

uttsc(1)

NAME

uttscprinteradm - uttsc printer cache administration tool

SYNOPSIS

uttscprinteradm -h
uttscprinteradm -l [*username*]
uttscprinteradm -d *username* [*printer*]

DESCRIPTION

The **uttscprinteradm** command is a utility to administer users and printers in the uttsc printer configuration cache. The cache is used to automatically preserve printer configuration changes done by users to printers provided on the uttsc command line during subsequent connections to Windows servers through uttsc.

Separate printer configuration entries are stored for every Windows user/printer combination.

Configuration entries of the same printer using different Windows drivers are not compatible with each other. If a printer is redirected with a new driver none of the configuration saved with the previous driver will be restored. If the configuration is changed while the printer is redirected with the new driver the new configuration will automatically overwrite the old configuration.

OPTIONS

The following options are supported.

-h Display usage information.

-l [*username*]
List printers of all users if no user specified or list printers of the specified user.

-d *username* [*printer*]
Delete specified printer of the specified user or (if no printer specified) delete user and all printers of user.

EXIT STATUS

0 if operation performed successfully, 1 if error found

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuttsc

SEE ALSO

uttsc(1)

NAME

uttscrestart - Sun Ray Windows Connector utility for restarting service.

SYNOPSIS

uttscrestart

DESCRIPTION

The **uttscrestart** command is used for restarting the Sun Ray Windows Connector daemon process. Existing Sun Ray Connector sessions will not be affected by the restart.

uttscrestart can only be run by the super-user.

OPTIONS

There are no options to this command.

EXAMPLES

Example 1: This starts the service if it is not running, or restarts it if it is already running.

uttscrestart

EXIT STATUS

0 if operation performed successfully, 1 if error found

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuttsc

SEE ALSO

uttsc(1)

NAME

utumount - Sun Ray Mass Storage unmount utility

SYNOPSIS

/opt/SUNWut/bin/utumount -u *mount_path*

DESCRIPTION

The **utumount** command has the same functionality as utdiskadm -u. It attempts to unmount the filesystem on *mount_path* if the related device is a Sun Ray storage device belonging to the user.

OPTIONS

The following options are supported.

-u *mount_path*

Unmount *mount_path*

EXIT STATUS

The following exit codes are returned:

0 Success.

1 Failure.

FILES

The following files are used:

$UTDEVROOT/dev/dsk

The directory containing links to block device names for each partition on the device.

$UTDEVROOT/dev/rdsk

The directory containing links to raw device names for each partition on the device.

ENVIRONMENT VARIABLES

UTDEVROOT points to a symbolic link of the device root for the Sun Ray DTU associated with a user's session.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWutsto

SEE ALSO

utdiskadm(1M), **uteject(1M)**, **utmount(1M)**, **utmountd(1M)**, **utstoraged(1M)**, **utdisk(7D)**

NAME

/opt/SUNWut/sbin/utusbadm - Enable or disable USB device services

SYNOPSIS

DESCRIPTION

Note that **utusbadm** command is deprecated. Use **/opt/SUNWut/sbin/utdevadm** command to enable/disable usb services.

The **utusbadm** ommand is used to disable/enable access to all USB ports on SunRay devices. It does not affect HID devices such as the keyboard and mouse; however, it does affect all other devices attached to the USB ports, which will not be accessible if the site is so configured.

This is a site-wide property. When set, it affects all units connected to the failover group.

Changing this configuration requires a cold system restart before it can take effect. When a successful change is made, the command reminds the administrator to restart the services.

OPTIONS

The following options are supported.

-h Print the current usage state.

-e Enable all USB device services.

-d Disable all USB device services.

ENVIRONMENT VARIABLES

None

EXIT STATUS

The following exit values are returned:

0 on success

1 on error

FILES

None

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	**SUNWuto**
Interface Stability	Public Evolving

SEE ALSO

utrestart(1M), utdevadm(1M)

NAME

utuser - Sun Ray user administration utility.

SYNOPSIS

/opt/SUNWut/sbin/utuser **-a "***tokenID,server-name,server-port,name,other-info*" [**-r** *tokenreader*]

/opt/SUNWut/sbin/utuser **-a** **-f** *filename* [**-r** *token-reader*]

/opt/SUNWut/sbin/utuser **-a** **-i** *current-tokenID new-tokenID* [**-r** *token-reader*]

/opt/SUNWut/sbin/utuser **-d** *tokenID*

/opt/SUNWut/sbin/utuser **-d** **-f** *filename*

/opt/SUNWut/sbin/utuser **-d** **-i** *current-tokenID*

/opt/SUNWut/sbin/utuser **-e "***tokenID,server-name,server-port,name,other-info*"

/opt/SUNWut/sbin/utuser **-e** **-f** *filename*

/opt/SUNWut/sbin/utuser **-e** **-i** *current-tokenID* [**enabledisable**]

/opt/SUNWut/sbin/utuser **-h**

/opt/SUNWut/sbin/utuser **-l**

/opt/SUNWut/sbin/utuser **-l** **-c**

/opt/SUNWut/sbin/utuser **-l** **-i** *substring*

/opt/SUNWut/sbin/utuser **-l** **-n** *substring*

/opt/SUNWut/sbin/utuser **-L**

/opt/SUNWut/sbin/utuser **-L** **-c**

/opt/SUNWut/sbin/utuser **-L** **-i** *substring*

/opt/SUNWut/sbin/utuser **-L** **-n** *substring*

/opt/SUNWut/sbin/utuser **-L** **-g**

/opt/SUNWut/sbin/utuser **-L** **-s** *session*

/opt/SUNWut/sbin/utuser **-o**

/opt/SUNWut/sbin/utuser **-p** *tokenID*

/opt/SUNWut/sbin/utuser **-r** *token-reader*

DESCRIPTION

The **utuser** command allows the administrator to manage users registered on the Sun Ray server which the command is run. The information that **utuser** provides is from the Sun Ray administration database and the Sun Ray Authentication Manager.

utuser operations that only display information may be run by any user. Operations that change or delete data are run under superuser privileges.

OPTIONS

The following options are supported.

-a Add user with the specified *tokenID*, *servername*, *serverport*, *name* and *other information* properties.

The 5 comma-delimited values should be enclosed within quotes. All values except *tokenID* and *name* are optional, although comma separators are required (for example "<tokenID>,,,<name>,").

-a -f Batch add multiple users using input from the specified *filename*. The format of each line in the input file is: *tokenID,server-name,serverport,name,other-info*. All values except *tokenID* and *name* are optional, although comma separators are required (for example "<tokenID>,,,<name>,").

-a -i Add the specified *new-tokenID* to the user that currently has token *current-tokenID*.

-d Delete the user with the specified *tokenID*. This command deletes the user and all of the user's tokens. (To delete a single token without deleting the user, use the **-di** option.)

-d -f Batch delete multiple users using input from the specified *filename*. The format of each line in the input file is: *tokenID*. However, you may use the output of the **-o** option as input to this option as all arguments after the first comma are ignored. For each token-id specified in the filename, this command deletes the associated user and all of the user's tokens. (To delete a single token without deleting the user, use the **-di** option.)

-d -i Delete token *current-tokenID* from the user that currently has ownership of it. The token to be deleted must not be the user's only token. This command does not delete the user or any of the user's other tokens. (To delete the user and all the user's tokens, use the **-d** option.)

-e Edit properties for the user with the specified *tokenID* by changing the *server-name*, *server-port*, *name* and *other-information* properties to the specified values.

The 5 comma-delimited values should be enclosed within quotes. All values except *tokenID* and *name* are optional, although comma separators are required (for example "<tokenID>,,,<name>,").

-e -f Batch edit multiple users using input from the specified *filename*. The format of each line in the input file is: *tokenID,server-name,server-port,name,other-info*. All values except *tokenID* are optional, although comma separators are required (for example "<tokenID>,,,,").

-e -i Enable or disable the specified *current-tokenID*.

-h Show usage information (this message).

-l List all users registered in the admin database..

-l -c List all users registered in the admin database that are currently logged in.

-l -i List all users registered in the admin database with token-ids that contain the specified substring.

-l -n List all users registered in the admin database with names that contain the specified substring.

-L List all users registered in the admin database (long format).

-L -c List all users registered in the admin database that are currently logged in (long format).

-L -i List all users registered in the admin database with token-ids that contain the specified *substring* (long format).

-L -n List all users registered in the admin database with names that contain the specified *substring* (long format).

-L -g List all users registered in the admin database currently logged in and the servers into which they are logged in.

-L -s List all users registered in the admin database with session types matching the specified session type, which can be either "default", "kiosk" or "regular" (long format).

-o Dump user list in comma-delimited format. The format of each line output by this option is: *tokenID*, *server-name*, *server-port*, *name*, and *other-info*.

-p Show user properties for user with the specified *tokenID*.

-r When specified alone, this option reads a token-id from the specified token reader. When specified with the **-a**, **-af** or **-ai** options, the **-r** flag instructs **utuser** to use the specified token reader to assist in adding users or tokens whenever the character "x" is used in place of a token-id. The command will prompt you to insert the token into the specified reader when its ready to read the token.

For the **-l -i**, **-l -n**, **-L -i**, and **-L -n** options, the substring comparisons are case-insensitive.

EXAMPLES

Example 1: This command displays all users that have "parker" in their usernames:

% **/opt/SUNWut/sbin/utuser -l -n parker**

Example 2: This command adds a user with unknown token-ID, server name "localhost", server port "7007", user name "John Anderson", and other information "C987" by using the token reader 08002086e18f to read the token-ID:

/opt/SUNWut/sbin/utuser -a "x,localhost,7007,John Anderson,C987" -r 08002086e18f

Example 3: This command adds multiple users using input from the /tmp/users file:

/opt/SUNWut/sbin/utuser -a -f /tmp/users

Example 4: This command reads a token from token reader 08002086e18f:

/opt/SUNWut/sbin/utuser -r 08002086e18f

FILES

The following file is used:

- **/etc/opt/SUNWut/utadmin.conf**

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuta

SEE ALSO

utdesktop(1M), **utadmin.conf(4)**

NOTES

The **-G** option has been deprecated in favor of using **utuser -L -g**.

The **-k** option has been deprecated. Use **utsession -k** instead.

NAME

utwall - Sun Ray user notification utility.

SYNOPSIS

/opt/SUNWut/sbin/utwall **-a** *aufile* [**-r** *n*] [**-v**] { **ALL** | user [**:***display*] | **:***display* ... }

/opt/SUNWut/sbin/utwall [**-d**] [**-m** "*subject*"] [**-t** "*message-text*"] [**-v**] { **ALL** | user [**:***display*] | **:***display*

/opt/SUNWut/sbin/utwall [**-u** "*message-text*"]

DESCRIPTION

utwall sends a message or an audio file to users having an **Xserver** process. The messages can be sent in email and/or displayed in a pop-up window. When sent to a multihead session, the pop-up window will appear on all displays for that session.

Options **-a** and **-d** require superuser privileges.

OPTIONS

The following options are supported.

-a *aufile*
: Annunciate mode. Plays the audio file *aufile* on the specified user's X session. Audio files of type **.au** can be found at **/usr/demo/SOUND/sounds**.

-d
: Pop up a window with the supplied message on each Xserver instance.

-m
: Send mail with the given subject "*subject*" and supplied message. If the text has white space, use single or double quotes. Substitution is supported.

-r *n*
: Repeat the annunciation *n* times. This option can only be used with **-a**. Default is 1.

-t
: Message text. Alternatively, the message can be supplied as **stdin**. If the text has white space, use single or double quotes. Substitution is supported.

-v
: Verbose mode.

OPERANDS

The following operands are supported:

ALL
: Action is performed on all user having an Xserver process.

user**:***display*
: Action is performed on the given users (optional display number *display*) having an Xserver process.

:*display*
: Action is performed on the users having display number *display*.

EXAMPLES

Example 1: This command sends email to all users:

 # /opt/SUNWut/sbin/utwall -m 'System policy change...' -t 'Tonight0lease log off' ALL

The email reads:

Subject: System policy change...
Tonight
Please log off

Example 2: This command pops a window up on all sessions stating "Log off now!"

 # /opt/SUNWut/sbin/utwall -d -t "Log off now!" ALL

Example 3: This command pops a window up on jsmith's session on display 26 with the text from *messagefile*

/opt/SUNWut/sbin/utwall -d jsmith:26 < *messagefile*

Example 4: This command pops a window up with a greeting to the user on display number 10

/opt/SUNWut/sbin/utwall -d -t "Hello" :10

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto

SEE ALSO

wall(1M), **mailx(1M)**, **utaudio(1)**

NOTES

When Sun Ray DTUs are configured for Xinerama, only the origin screen displays the **utwall** message.

NAME

utwho - Provide a compact summary of sessions and firmware maintenance.

SYNOPSIS

/opt/SUNWut/sbin/utwho -c [**-a**] [**-H**]

DESCRIPTION

The **utwho** command is a script that assembles information about display number, token, and logged-in user, and displays that information in a compact format. It can also display the IP address, Sun Ray model, and MAC address of a Sun Ray that is connected to a given session.

OPTIONS

The following options are supported.

-c With **utwho**, show connected Sun Rays with the display number, session token, logged in user, IP address, Sun Ray model, and Sun Ray MAC address. The display number is presented in the form d.m, where d is the X display number of the session, and m is the index within a multihead group of that particular Sun Ray.The session token is the value of **$SUN_SUNRAY_TOKEN** within a session. The model and MAC address are output as PxB.MAC, where Px is the ending part of the model type, e.g., P4, and MAC is the 6-byte ethernet MAC address in hexadecimal format. Without the **-c** option, the command displays only session information, including the X display number, token, and logged in user. In this mode, sessions are displayed even if they don't have a Sun Ray connected to them.

-a In combination with other options, this option controls the selection of Sun Rays or sessions to display. Without **-a**, only sesssions that have logged in users are displayed. With **-a**, all sessions or Sun Rays are displayed, and those with no logged in user have a user id field of "????".

-H Output column headings above the regular output.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWuto

SEE ALSO

utfwadm(1M), **attributes(5)**

NAME

utxconfig - Sun Ray DTU X server configuration utility.

SYNOPSIS

/opt/SUNWut/bin/utxconfig [**-a**] [**-c** *config-file*] [**-d** *display*] [**-D**] [**-k** *xkb-value*] [**-l**] [**-L** *geom*] [**-m** *multihead*] [**-p** *pcolor*] [**-r** *res*] [**-R** *geom*] [**-S** *screen-order*] [**-t** *token*] [**-v**] [**-x** *xinerama*]

/opt/SUNWut/bin/utxconfig **-e** [**-d** *display*] [**-t** *token*]

/opt/SUNWut/bin/utxconfig [**-o**] [**-f** *file*]

/opt/SUNWut/bin/utxconfig [**-i**] [**-f** *file*]

/opt/SUNWut/bin/utxconfig [**-X** *xserver*]

DESCRIPTION

utxconfig displays and configures X server parameters for Sun Ray sessions. The changes to the X server are not evident until a restart of the X server process. For example, log out, then log in.

OPTIONS

The following options are supported.

-a Allows the setting or listing of the default values. Only superuser may change the default settings.

-c *config-file*
Sets a specific *configfile* to use. The usage of this option is beyond the scope of this manual.

-d *display*
Sets the X display variable used to determine the Sun Ray DTU session. Otherwise, the DISPLAY environment variable is used. Users must have access to an X server attached to their session before they can read or change the settings for that session.

-D Debug flag.

-e Erases all specific settings for the session. All settings return to their default values.

-f *file* Specifies a file to be used in conjunction with **-o** or **-i**.

-i Populates the system settings database from a comma delimited text record such as the one produced by **-o**. Input is taken from the standard input unless **-f** is specified. You must be root to use this option.

-k *xkb-value*
Enable or disable the XKB extension. To enable use of XKB specify "on". To disable XKB specify "off". To revert to the system default specify "reset".

-l Lists the current settings for the session. If specific values have not been set for particular attributes, default values are printed.

-L *geom*
Lists out the X server screen device start-up arguments for the user preferred geometry set with **-R** or for *geom* if none is set. The use of this option is beyond the scope of this manual.

-m *multihead*
Enables or disables multihead mode for X session startup. By default, if a session is started on a multihead terminal group, then the session starts in multihead mode to match the terminal group with an appropriate number of screens and geometry. Specify "off" to disable this behavior and the session starts on a single terminal with one screen. The keyword "reset" may be specified to reset to the system default.

-o Output all system settings in a comma-delimited text record. Intended for use with **-i**. Outputs to standard output unless **-f** is specified.

-p *pcolor*
Parameter that specifies the level of support for the PseudoColor (8-bit) visual in the X server. The PseudoColor visual is not enabled by default. The accepted values for *pcolor* are "off", "on",

"default", and "reset". A *pcolor* value of "off" will disable the PseudoColor visual. A *pcolor* value of "on" will enable the PseudoColor visual, but the TrueColor visual (24-bit) will remain the default. A *pcolor* value of "default" will enable the PseudoColor visual and make it the default visual, although the TrueColor visual will still be available. A *pcolor* value of "reset" will revert to the system default.

-r *res* Parameter that specifies a resolution (number of pixels) that the X server should provide for the session. The format of *res* is *WIDTH*x*HEIGHT*, for example 1280x1024. **utxconfig** enforces restrictions on the possible widths and heights that can be specified. The keyword "auto" may also be specified, which will enable selection of a resolution that best matches the resolution capabilities of the terminal on which the X session is started. The keyword "reset" may be specified to revert to the system default.

-R *geom*

Specify a preferred screen geometry in the form *COLS*x*ROWS*. At X server startup this geometry overrides the terminal group geometry on which the session is started. See **-m**. The keyword "auto" may also be specified as a value, which will cause the geometry of the terminal group on which the X session is started to be used. The keyword "reset" may be specified to revert to the system default.

-S *screen-order*

Specify a preferred screen number order for the session's screen group. The order must be a legal for standard Xsun (Sun Ray X server) screen placement, or the keyword "auto" which will cause a default screen ordering to be chosen. Manipulating the geometry via the -R option will automatically change the screen order to an appropriate value.

-t *token*

Allows the setting of a specific token to use. The use of this option is beyond the scope of this manual.

-x *xinerama*

Enable or disable XINERAMA extension. To enable use of XINERAMA specify "on". To disable XINERAMA specify "off". To revert to the system default specify "reset".

-v Verbose output for -l option. When both -l and -v are specified attributes with default values are labeled as such.

-X *xserver*

Specify which X server to output options for.

EXAMPLES

Example 1: This command enables PseudoColor visual on a 1024 x 768 screen:

% **/opt/SUNWut/bin/utxconfig -r 1024x768 -p on**

Example 2: This command configures a multihead group with two screens, right and left:

% **/opt/SUNWut/bin/utxconfig -m on -R 2x1 -S 0,1**

Example 3: This command sets the screen geometry and ordering to automatic:

% **/opt/SUNWut/bin/utxconfig -R auto**

ATTRIBUTES

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

Attribute Types	Attribute Values
Availability	SUNWuta
Interface Stability	Evolving

SEE ALSO

Xserver(1)

NOTES

The settings are actually maintained on the basis of an authentication token and do not remain specific to a single X display number.

NAME

utxlock - Sun Ray utility for locking a window session.

SYNOPSIS

/opt/SUNWut/bin/utxlock

DESCRIPTION

The **utxlock** utility locks the current display in a manner specific to the current windowing environment. If the current environment is Gnome, it uses **xscreensaver-command**; if the current environment is CDE, it uses **dtsession**; otherwise **utxlock** is used.

> **Note:**
>
> Although some users may find screen locking an inconvenience, overriding it has security implications that should be obvious. Override at your own risk.

A user may disable any screen lock behavior by setting the environment variable **SUN_SUNRAY_UTXLOCK_PREF** to **NULL**. Any other value will be used as a command line to use for invoking a screen lock command instead of the default behavior.

SRSS invokes **utxlock** on any session disconnect. To disable this behavior, add the following line to your **$HOME/.dtprofile**:

SUN_SUNRAY_UTXLOCK_PREF=; export SUN_SUNRAY_UTXLOCK_PREF

As another example, if a user had their own screenlock program called mylock, and they wanted to pass it the argument **-l**, they should add the following line to their **$HOME/.dtprofile** :

SUN_SUNRAY_UTXLOCK_PREF="$HOME/bin/mylock -l"; export SUN_SUNRAY_UTXLOCK_PREF

OPTIONS

No options are supported.

ATTRIBUTES

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

Attribute Type	Attribute Value
Availability	SUNWutu