Support for Sun Ray Kiosk Session Development Kit (All Topics)

Contents

Support for Sun Ray Kiosk Session Development Kit (All Topics)

Purpose

This guide is intended to enhance the understanding, use of, and development for the Kiosk Mode feature of Sun Ray Server Software (SRSS). This guide contains a more in-depth look at the the Kiosk Mode feature than is found in the official SRSS Documentation. It should be viewed as both a companion and supplemental guide to the official documentation, but it should be viewed as not a replacement. This guide assumes the reader has a basic understanding of the operation of SRSS and does not cover items such as installation and configuration of SRSS, with the exception of where those items may have implications on Kiosk Mode.

Support Statement for 3rd Party Provided Kiosk Session Types

Oracle fully supports the creation of Kiosk Session Types by 3rd parties for use with Sun Ray Server Software. Any documented interfaces, unless specifically noted otherwise, used by the 3rd Party Provided Kiosk Session Type are supported. For Customers with support contracts, this means service calls can be logged if a reproducible issue occurs with any supported interfaces do not perform as documented.

Oracle does not provide support for the actual application used in the custom Kiosk Session Type. The definition of an "application" as it pertains to support of the Kiosk Session Type is the specified script or binary that is referenced by the 'KIOSK_SESSION_EXEC' variable in the custom Kiosk Session Type Descriptor.

The only exceptions are Kiosk Session Types that are provided with Oracle Products. This includes Kiosk Session Types included with:

- Sun Ray Connector for Windows OS
- Sun Ray Connector for VMWare View Manager
- Oracle Virtual Desktop Infrastructure
- Kiosk Session Types bundled with Sun Ray Server Software for Solaris

Example: A 3rd party wishes to develop a Kiosk Session Type that uses the Citrix ICA Client to display a full screen Microsoft Windows Desktop. Oracle will support the use of Kiosk Mode for this purpose along with the Kiosk Session configuration files to create this type of Session. Oracle will not provide support for the Citrix ICA Client executable, any custom shell scripts written to launch the Citrix ICA Client, or the configuration files required by the Citrix ICA Client for operation.

Support for Examples provided in this Guide

This guide contains information on public and private interfaces, as well as public and private details . An interface is any method used to interact with the provided software to obtain an desired result. Details for the purpose of support, are considered to be configuration files, start up files, log information, and design concepts.

Examples scripts in this guide may use interfaces and details that are not deemed public. While these interfaces and details are highly unlikely to change, they may in fact change or may be non-existent with future versions. Unless otherwise state, examples that use private interfaces or private details are considered unsupported. While care has been taken developing these examples, Oracle assumes no risk from their use.

Examples code that uses these private interfaces or details will highlighted by following note:

Private Interface
Additional information may be included with the note

Some examples in this guide use public interfaces that may have the potential to cause a performance issues in certain situations. Such examples will use the following warning:

Performance Impact
Additional information may be included with the warning.

Support for information related to System Internals

Some of the information in this guide covers low-level design and implementation details. It is provided to allow the reader to gain a deeper technical understanding behind the architecture and design of the Kiosk Mode feature of Sun Ray Server Software. This information is provided for the sake of completeness only and is not intended to be modified in any manner.

Information related to System Internals will carry the following warning:

System Internals
Unless otherwise directed by Oracle Support, do not modify the files covered in this section. Please see the support note on this topic.

About Kiosk Mode (All Topics)

Contents

About Kiosk Mode (All Topics)

What is Kiosk Mode?

Sun Ray Server Software (SRSS) offers two modes of operation, Regular and Kiosk.

Regular Mode delivers a desktop based on the platform that SRSS is installed on. For instance, if SRSS is installed on Oracle Enterprise Linux users get an Oracle Enterprise Linux desktop using either Gnome or KDE. Users authenticate on this platform and traditionally execute applications located on the platform.

Kiosk Mode delivers an almost unlimited variety desktops or applications to users, even though the actual desktop or application may be running elsewhere. Kiosk mode bypasses the normal authentication methods of the platform and runs anything that the administrator defines. A Kiosk Session could be anything from a full screen instance of a Firefox web browser, to a full screen Windows 7 Desktop running in a virtual machine. See How a Kiosk Session Works

A few examples of Oracle products that use Sun Ray Server's Kiosk Mode feature are:

Oracle Virtual Desktop Infrastructure

Sun Ray Connector for VMware View Manager

Sun Ray Connector for Windows OS

While the above examples deliver a traditional desktop, Kiosk Mode is not limited in that regard. Here are just a few example use cases for Kiosk Mode:

- Airport Arrival and Departure Screens
- Digital Signage / Informational Kiosks
- Trade Show Badging Stations
- Internet Access Terminals
- Point of Sale Terminals
- Call Center Agent Stations
- And many more

How A Kiosk Session Works

When a Sun Ray Server is configured for Kiosk Mode, Sun Ray clients are granted non-authenticated access to the server to display a "non-regular session". A regular session is a standard desktop available on the operating system that Sun Ray Server Software is installed on. For an Oracle Solaris based Sun Ray Server, a regular session would be a Java Desktop System session. For an Oracle Enterprise Linux based Sun Ray Server, a regular session would be a KDE or Gnome desktop session. See What is Kiosk Mode.

These non-regular sessions are called Kiosk Sessions, the actual applications launched during a Kiosk Session are defined by an administrator and are called Kiosk Session Types. A Kiosk Session Type is a combination of configuration files, scripts, data files, and binary applications which are assembled in a specific manner for use with Kiosk Mode in a Kiosk Session Descriptor. The Kiosk Session Type that is defined by the Kiosk Session Descriptor is stored in the Sun Ray Data Store as the Kiosk Session Configuration.

There can only be a single Default Kiosk Session Configuration per Sun Ray Host Group, but many different Kiosk Session Types can be defined. Many different, or Alternate Kiosk Session Configurations can be configured to use the different Kiosk Session Types. Administrators can override the Default Kiosk Session Configuration and assign Alternate Kiosk Session Configurations to something called Token ID. Every Sun Ray

session has a Token ID that uniquely identifies that session. Like the MAC address of a network interface card, no two sessions will have the same Token ID. See How Can Kiosk Mode Be Applied.

Kiosk Sessions run under a local user account called a Kiosk User Account. Kiosk User Accounts come from a configurable pool of locked down user ID's called the Kiosk User Pool. Accounts in the Kiosk User Pool can only be granted access to the system via a specialized pluggable authentication module (PAM) called pam_kiosk. pam_kiosk checks the Sun Ray Server's system policy and/or override data in the Sun Ray Data Store to verify if that client's Token ID should be granted a kiosk session. If the Token ID for that Sun Ray client is granted access to a Kiosk Session, a Kiosk User Account is assigned to the session and that Kiosk User's Account ID is marked as unavailable for another Kiosk Session.

As the Kiosk Session starts, the Kiosk Session Service creates a new, dynamically provisioned, home directory for the assigned Kiosk User Account. Administrators pre-define a template/skeleton structure of any required user-specific state and/or application-specific configuration files that are needed by the Kiosk Session Type used by the Kiosk Session Configuration. These are known as Kiosk Session Prototypes. Any files in the Kiosk Session Prototype, which are specific to the corresponding Kiosk Session Type, are copied by the Kiosk Service into the assigned Kiosk User Account's home directory. The Kiosk Service modifies the files with appropriate ownership and permissions to enable them to be used by the Kiosk User Account associated with the Kiosk Session. This process allows all Kiosk Sessions to start in an identical and predictable state.

When the Kiosk Service finishes configuring the Kiosk Session based on the assigned Kiosk Session Configuration, the session is handed over to the Kiosk User Account to execute the scripts and or applications defined in the Kiosk Session Descriptor.

Without having to authenticate against the OS that the Sun Ray Server is installed, the end user now has direct access to the application(s) defined in the Kiosk Session Type and can interact with the application until either they end the application or certain session limits are reached.

Kiosk Sessions Types always have at least one “critical” process running, which normally is the application defined in the Kiosk Session Descriptor. If Kiosk Service senses that the critical process is no longer running, whether ended by the end user, having reached a session limit, or by some type of failure, it begins a shutdown and clean up of the Kiosk Session.

As the Kiosk Session ends, all user-specific state and files, including home directory content, are deleted by the Kiosk Session Service. Any secondary or residual processes or files owned by the current Kiosk User are also terminated and deleted by the Kiosk Service. At this point the Kiosk User Account's ID that was previously used by the session is marked as available for another Kiosk Session.

How is Kiosk Mode applied?

Based on Default System Policy

There are two classes of users on Sun Ray Server which administrators can configure the system policy to enable Kiosk Mode for:

- Card Users: Sessions are initiated or resumed when a smart card is inserted in the the Sun Ray Client
- Non-Card Users: Session are initiated when the Sun Ray Client boots or resumed when a smart card is removed.

Being able to set Kiosk Mode based on user class alone is a powerful tool. Consider the following example:

> An organization with a limited number of licenses for Microsoft Operating Systems wishes to enhance their license control by limiting the users who can access a Microsoft environment.
>
> This organization can use the system policy to:
>
> - Enable Regular Mode for Non-Card Users. These users will get a desktop based Oracle Solaris or Oracle Enterprise Linux
> - Enable Kiosk Mode for Card Users. Users who received smart cards will connect to Windows 2008 Remote Desktop Services using the Sun Ray Connector Windows OS

System Policy Configuration
Sun Ray System Policy can be configured via either the Sun Ray Admin GUI or via the command line utility utpolicy.

Based on Token ID

It is also possible to assign Kiosk Mode (or Regular Mode) to any User or Sun Ray Client based on their "Token ID". All Sun Ray Sessions, whether card or non-card based, are tracked by unique identifier called a Token ID.

By default the Token ID tells the Sun Ray Server where to display the session. This, at a very basic level, is how session mobility, or "Hot Desking" works. When a Token ID moves, the session follows. Token IDs can also be registered in the Sun Ray Data Store, which allows the administrator to configure what to display to that session.

Once a Token ID has been registered you can use the Sun Ray Admin GUI to override the default policy. For smart card tokens, choose the "Tokens" tab and for pseudo tokens, choose the "Desktop" tab. Find your Token ID, click Edit and you can select the following types of Sun Ray Sessions:

- Default: What ever happens to be the System default for your Card or Non-Card Session. Either Regular Mode or Kiosk Mode.
- Kiosk: Always Use Kiosk Mode for this Token. This Token will get the Default Kiosk Session Type, regardless of the System Policy.
- Regular: Always Use Regular Mode for this Token. This Token will get a normal Solaris or Linux based Desktop, regardless of System Policy.

Registered Only Mode
Sun Ray System Policy has an option to allow only registered Token ID's to gain access to system. While token registration is by product of using Registered Only Mode, it should not be considered the only method of registering Tokens.

In addition to Registered Only Mode, tokens can also be registered using:

- A Token Reader
- The Sun Ray Admin GUI
- The 'utuser' command line utility

Based on Token ID and overriding Default Kiosk Session Configuration

Not only do registered Token IDs allow administrators to to override default System Policies, the Default Kiosk Session Configuration can also be overridden when using the 'utkioskoverride' utility. This feature makes Sun Ray the most flexible and granular desktop solution available on the market. Consider the following example:

Sun Ray Servers have been deployed at the main branch of a County Library. Sun Ray Clients have been deployed at all of the local Library branches and access the centralized Sun Ray Servers in the main branch remotely. All of the patrons of the library are issued smart cards that they can use to check out materials and access to the internet from Sun Ray Clients. Unfortunately the default System Policy and Default Kiosk Session Configuration does not meet all of their needs.

The Sun Ray Servers are configured as follows:

- A default System Policy of no-access for Non-Card User based sessions
- A default System Policy of Kiosk Mode for Card User based sessions.
- The Default Kiosk Session Configuration displays a locked down web browser running in full screen mode.

Requirements

- Library Technical Staff need access to a Regular session based to the underlying Sun Ray Server OS from any client in any branch for troubleshooting and administration
- Librarians at the service counter need access to Microsoft Windows based Library Management System.
- Each branch has their own Windows 2008 server with Remote Desktop Services enabled that is running the Library Management System for that branch.
- The Librarians do not wish to use smart cards since the Sun Ray clients at the service counter are for staff use only
- Two large wall mounted flat panel displays need to display local branch information such hours of operation, current happenings, and new books available for loan
- Smart cards should not have to be used with the Sun Rays connected to the informational displays since they are never used interactively by patrons
- The local branch information to be displayed is in PDF format and created by the Library's Art Department that works from the main branch. The files are published weekly on a centralized web server using a branch specific URL.

Sun Ray Server Configuration

- Default System Policy of Kiosk Mode for card users can be overridden for the smart card Token ID's of the Library Technical Staff giving them a full desktop on the underlying OS that is hosting Sun Ray Server Software from any DTU connected to the centralized servers
- Default System Policy of No Access for Non-Card User based sessions can be overridden for the Token ID's of the Sun Ray Clients located at the service counter by assigning those client's Token IDs a session type of Kiosk Mode
- The Default Kiosk Session Configuration can be overridden for the Token ID's of the Sun Ray Clients located at the service counter and assign them an Alternate Kiosk Session Configuration that uses the Sun Ray Connector for Windows OS to access the local branch's Library Management System
- The Alternate Kiosk Configuration's Session Type used by the Sun Ray Clients located at the service counter can read the Location field of that Token ID to determine which branch's server the Sun Ray Connector for Windows OS should connect to
- The Default System Policy of No Access for Non-Card User based sessions can be overridden for the Token IDs of the Sun Ray Clients connected to the large wall mounted flat panel displays assigning those client's Token IDs a policy of Kiosk Mode
- The Default Kiosk Session Configuration can be overridden for the Token IDs of the Sun Ray Clients connected to the large wall mounted flat panel displays and assign them an Alternate Kiosk Session Configuration that uses Adobe Acrobat Reader in full screen presentation mode which will inform patrons of that branches hours of operation, this week's events, and new books available for loan
- The Alternate Kiosk Configuration's Session Type used by the Sun Ray Clients connected to the large wall mounted flat panel display can read the Location field of that Token ID to determine which branch specific URL Adobe Acrobat Reader will pull the branch specific PDF from

Advanced Topic
How to override the Default System Policy and assigning Alternate Kiosk Session Configurations for individual Token IDs covered in the Advanced Topics section.

Important information regarding Sun Ray Token IDs

- Non-Card sessions are commonly referred to as pseudo sessions due to their Raw Token ID of pseudo.<Terminal ID of Client>. Ex: pseudo.080020861234
    - Certain System Policies, specifically Registered Mode or Non-Smart Card Mobility, will translate the Raw Token ID to a Logical Token ID.
- Card sessions will normally have a Raw Token ID of <smart card model>.<unique identifier>. Ex: mondex.9998007668077709
    - Card sessions using Registered Mode for card based sessions will also translate the Raw Token ID to a Logical Token ID.
- Logical tokens are dynamic and only exist for the lifetime of the session
    - Therefore only Raw Token IDs should be registered as only they pertain to physical clients or smart cards

Translating Logical Tokens
If a desired System Policy option translates the Raw Token ID, you can use the following example in a script to get the Raw Token ID from a Logical Token ID:

```
#!/bin/sh
cardToken=`/opt/SUNWut/sbin/utuser -p $SUN_SUNRAY_TOKEN | tail -2 | awk '/./ {print $1}'`
```

Installation and Configuration (All Topics)

Contents

Installation (All Topics)

Installing Kiosk Mode

The necessary packages for Kiosk Mode are installed during the initial installation of Sun Ray Server Software. Kiosk mode is not configured as active by default after installation.

Configuring Kiosk Mode

In order to use Kiosk Mode, the Sun Ray Server Software must be configured using the utconfig utility.

> Tip
> If you don't initially configure Kiosk Mode with utconfig, you can configure it at any time by running utconfig -k command.

If you choose to configure Kiosk mode during the configuration of Sun Ray Server, you will be asked to create the Kiosk User Pool.

The Kiosk User Pool contains Kiosk User Accounts that run the applications defined in the Kiosk Session Type. See How A Kiosk Session Works.

> Local Accounts Only
> Kiosk User Accounts are always local accounts. If your server is configured for NIS or LDAP and a user with the same User ID exists, the Kiosk User Pool will not be configured correctly. Temporarily change nsswitch.conf to only use files for user authentication and re-run utconfig or utconfig -k.

The administrator can choose the following for the Kiosk User Pool (defaults):

- User prefix (utku)
- Group (utkiosk)
- User ID range start (150000)
- Number of users (25)

> How Many Users?
> You must specify enough user accounts on each server for the number of kiosk sessions that are desired to run. Rarely is the default of 25 accounts per server enough to satisfy the needs of most environments. You must configure the same number of users on each Sun Ray Server in the Host Group to ensure proper load balancing of Kiosk Sessions.

When deciding on the number of kiosk users to create, there are a few important items to consider, such as:

- How many Sun Ray Thin Clients are deployed
- Is Oracle Virtual Desktop Client use allowed
- Smart card usage
- Session Timeout for disconnected sessions
- Sizing and Scalability of Customized Kiosk Session
- Server Outage/DR scenarios.

Example:

```
  Kiosk Mode enabled for both Non-Card and Card based sessions
+ One hundred Sun Ray clients deployed
+ 200 users with smart cards
+ 300 users with the Oracle Virtual Desktop client
-------------------------------------------------------------------------
600 possible Kiosk Session Users
```

Deciding on the number of Kiosk User Accounts to create should never be a guess. It should be based on:

- Sizing information gathered from scalability testing of the Kiosk Session Type
- The number of servers are in the Sun Ray Host Group, taking into account outages or disaster scenarios

Over-subscribing and Under-subscribing the Kiosk User Pool
The ability to configure an arbitrary number of Kiosk User Accounts in the Kiosk User Pool does not mean that the server is capable of handling that many Kiosk Sessions. Having too many kiosk sessions per server can result in a poor user experience and in some cases a server failure. Conversely, If you don't have enough Kiosk users in the pool, the next session after the final Kiosk User has been allocated will result in an error and the Kiosk Session will not be able to start.

Enabling Kiosk Mode

Via System Policy

The primary method of enabling a Default Kiosk Session Type for Non-Card and Card based sessions is via a System Policy. System Policy can be changed to enable Kiosk Mode using either the Sun Ray Admin GUI or via the command-line interface using 'utpolicy'.

Per Token ID

Along with enabling Kiosk Mode based on the System Policy for Non-Card and Card based sessions, it can also configured or overridden on a per token basis using the 'utkioskoverride' utility. See How is Kiosk Mode applied.

Kiosk Session Components (All Topics)

Contents

Kiosk Session Components (All Topics)

Kiosk User Accounts

All computer applications must run under some type of user account and Kiosk Sessions are no different. To enable real people to access applications without requiring the need to authenticate to the underlying operating system of SRSS, Kiosk Mode manages a pool of local user accounts. If the Kiosk Service determines that the system administrator has configured either the system policy or the current Token ID to run a Kiosk Session, unauthenticated access to system is granted.

While Kiosk User Accounts do not correspond to the real person, their role in Kiosk Mode allows a real person to use the applications defined by the administrator in a unauthenticated manner. Without a Kiosk User Account, a Kiosk Session cannot run.

Account Characteristics

Kiosk user accounts have the following characteristics:

- Have a default naming scheme of utkux, where x is a range from 0 to N-1 where N = Chosen number of Kiosk Users Accounts to create
- Naming prefix can be chosen if the default of utku has risk of a collision
- UID by default start at 150000 (starting UID can be specified by the administrator)
- UID range must be contiguous
- Home directories are located in /var/opt/SUNWkio/home/<$USER>
- Local accounts only (/etc/passwd). Centralized NIS or LDAP Kiosk User Accounts are not supported.

Preventing unauthenticated access from becoming uncontrolled access

In an attempt to limit the impact a kiosk user can have on the system, the following is also true of Kiosk User accounts:

- Kiosk User Accounts are locked for normal login (GDM, SSH, Telnet, etc)
- Kiosk User Accounts belong to a local Unix group (utkiosk) that has minimal rights on the system.
- No two sessions use the same Kiosk user account at the same time on the same server
- The home directory associated with a Kiosk User Account is cleared completely after a session ends
- The home directory associated with a Kiosk User account is rebuilt from scratch when a session is started from a prototype directories
- Residual processes owned by the Kiosk User Account are killed when a Kiosk session ends and/or before a new one is started.
- Any and all files in /tmp and /var/tmp that are owned by the Kiosk User Account are deleted when a Kiosk Session ends and/or before a new one is started.

Administering the Kiosk User Pool

If the initial creation of the Kiosk User Pool is well thought out and based on the tips provided in the configuration section it should not require any changes. However, if changes do need to occur, you can manage the user pool after the initial configuration using the 'kioskuseradm' utility located in /opt/SUNWkio/bin. With this tool you can view the pool settings, see how many Kiosk user accounts are in use, and modify the pool settings such as adding or decreasing the number of kiosk user accounts. Increasing or decreasing the pool of users can be done while kiosk sessions are active, but changing other pools settings such as group membership or UID range requires that no kiosk sessions are active.

Modifying Kiosk Accounts
The Kiosk User Accounts in the Kiosk User Pool are required to have contiguous User IDs. If other user accounts were added to the local system that interrupt the contiguous User ID's of the Kiosk User Pool, you can not extend the pool. You must modify the pool which requires deleting existing the accounts and recreating them. Thus Kiosk Sessions cannot be actively using the accounts that need to be modified.

Kiosk Session Descriptors

Kiosk Session Descriptors define, at a minimum, a session executable or "application" which will be executed by the Kiosk User Account associated with the Kiosk Session. Kiosk Session Descriptors may also define other session type specific properties. The session executable does not need to be an application binary, typically it is considered a best practice to call an application binary from a script.

Kiosk Session Type Descriptor Characteristics

Session Descriptors have the following characteristics:

- Kiosk Session Descriptor files need to have a .conf extension to be identified by the Kiosk Service
- Kiosk Session Descriptors are stored /etc/opt/SUNWkio/sessions
- Kiosk Session Descriptors can use a "short name" to identify themselves if stored in /etc/opt/SUNWkio/sessions.
- Kiosk Session Descriptors are a plain text file that contain certain “key=value” statements

Kiosk Session Type Descriptor Key=Value Statements

Key	Value	Requirement
KIOSK_SESSION_EXEC	The fully qualified path to the application, or script, that calls the application to be run under a Kiosk Session	Mandatory
KIOSK_SESSION_LABEL	A short description used to represent the session type in the administration tool. If not set, the "short name" will be used.	Recommended
KIOSK_SESSION_PROTOTYPE	A prototype directory used to populate $HOME of the Kiosk user account with any user-specific state and/or application-specific configuration files needed by KIOSK_SESSION_EXEC	Optional
KIOSK_SESSION_DESCRIPTION	A description of the Session Type	Optional – Only viewable via CLI tools and not the Admin GUI
KIOSK_SESSION_DIR	A directory under /etc/opt/SUNWkio/<name> where <name>matches the short name used for the session descriptor.	Optional
KIOSK_SESSION_ARGS	Application arguments to pass to the application or script defined under KIOSK_SESSION_EXEC	Optional – Better to use a script
KIOSK_SESSION_PRE	Pre-session scripts that can be used for advanced session preparation or cleanup	Optional
KIOSK_SESSION_POST	Post-session scripts that can be used for advanced session preparation or cleanup	Optional
KIOSK_SESSION_ICON	An icon that can be used to represent the session type in an administration tool	Optional-- Not currently viewable
KIOSK_SESSION_TIMEOUT_DETACHED	Timeout (in seconds) after which a Kiosk session on a disconnected Sun Ray DTU will be terminated.	Optional
KIOSK_SESSION_LIMIT_CPU	Identifies the maximum usage of cpu time (in seconds) per process for the Kiosk session.	Optional
KIOSK_SESSION_LIMIT_DESCRIPTORS	Identifies the maximum number of open file descriptors per process for the Kiosk session.	Optional
KIOSK_SESSION_LIMIT_FILESIZE	Identifies the maximum file size (512 byte blocks) for the Kiosk session.	Optional

KIOSK_SESSION_LIMIT_VMSIZE	Identifies the maximum swap size (in KB) per process for the Kiosk session.	Optional

About PRE and POST Session Scripts
KIOSK_SESSION_PRE and KIOSK_SESSION_POST scripts run as root. Any variables set in these scripts will NOT be inherited by the Kiosk User Account's environment.

About Kiosk Session Limits
-If not specified in the Descriptor file or Sun Ray Admin GUI, KIOSK_SESSION_LIMIT_* key=value pairs will default to the system defaults (if set).
-If specified in the Descriptor file or Sun Ray Admin GUI, KIOSK_SESSION_LIMIT_* key=value pairs will override the system defaults (if set).
-If specified in the Descriptor file, alternate values typed in the Sun Ray Admin GUI will override.
-If not specified in the Descriptor file, values can still be entered using Sun Ray Admin GUI.
-If not specified in the Descriptor file, Sun Ray Admin GUI, or system defaults, there are no limits.

Kiosk Session Limits
Exercise caution when using Kiosk Session limits. Choosing unsuitable values can cause unpredictable results for Kiosk Sessions. Consult the ulimit man pages for more information on system limits.

Example Kiosk Session Type Descriptor File

All Kiosk Session Types need a Kiosk Session Descriptor File in order to be executed by Kiosk User Accounts. The following is an example of a minimally configured Kiosk Session Descriptor that will launch a notepad like application called 'gedit'. First step in creating a Kiosk Session Descriptor is making file in /etc/opt/SUNWkio/sessions called NotePad.conf which would contain the following contents:

```
KIOSK_SESSION_EXEC=/usr/bin/gedit
KIOSK_SESSION_LABEL=Gnome Notepad
KIOSK_SESSION_DESCRIPTION="NotePad"
```

Listing all the defined Kiosk Session Descriptors

The Session drop down box of the Edit Kiosk Mode screen of the Sun Ray Admin GUI is one way to list all of the defined Kiosk Session Descriptors, one can also use the 'kioskdesc' utility to do the same.

```
# /opt/SUNWkio/bin/kioskdesc list -s
cde
NotePad
Terminal
jds3
uttsc
vda
```

Likewise, the 'kioskdesc' utility can be used to print the contents of the Kiosk Session Descriptor.

```
# /opt/SUNWkio/bin/kioskdesc print -s NotePad
KIOSK_SESSION_DESCRIPTION=NotePad
KIOSK_SESSION_DIR=
KIOSK_SESSION_EXEC=/usr/bin/gedit
KIOSK_SESSION_LABEL=Gnome Notepad
```

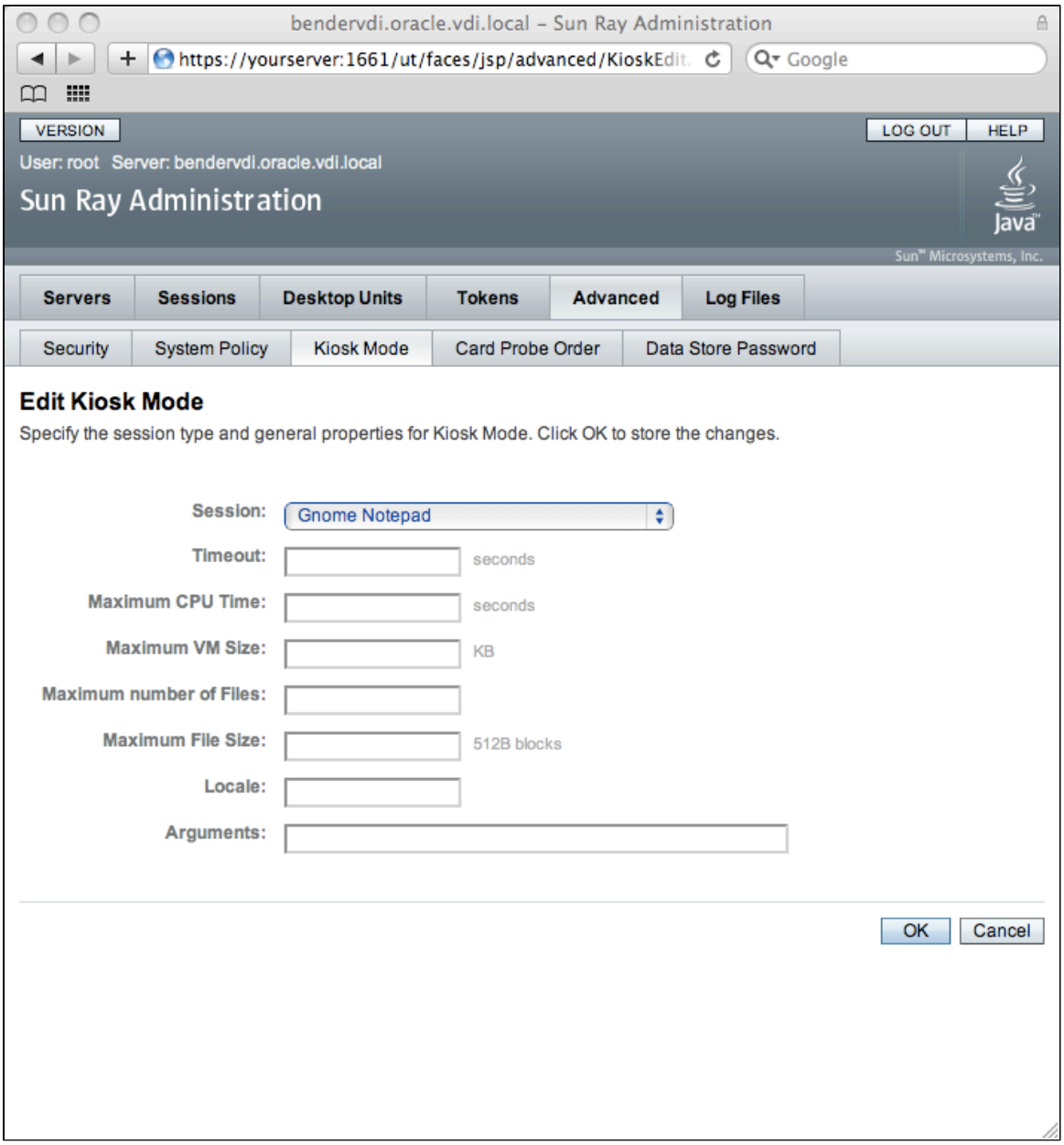
Setting the Default Kiosk Session Configuration

Once a Kiosk Session Descriptor file has been created, it can be selected for the system wide, or "Default" Kiosk Session Configuration. Selecting or changing the Default Kiosk Session Configuration can be done by either the Sun Ray Admin GUI or the command line interface 'utkiosk'

Sun Ray Admin GUI Method

- Point a web browser at http://<your Sun Ray Server Name>:1660 and log in with an administrator ID of the Sun Ray Server. Navigate to Advanced --> Kiosk Mode and Click Edit. The example Kiosk Session Type will be available as a choice under the Session drop down list.

> Session Name
> If the Key=Value pair of KIOSK_SESSION_LABEL was declared in the example Kiosk Session Descriptor file, this is the name that will appear in the Sun Ray Admin GUI instead of the "short name".



- Select this Session Type and click OK.

This is now the default Kiosk Session Configuration for all Token IDs configured to run a Kiosk Session.

> Changing the Default Session Configuration
> If the default Session Type was changed, existing Kiosks Sessions will need to be restarted.

Command Line Method

It is also possible to use the command line interface 'utkiosk' to set the Default Kiosk Session Configuration for all Token IDs.

To use the "short name":

```
echo KIOSK_SESSION=NotePad | /opt/SUNWut/sbin/utkiosk -i session
```

To use the fully qualified path:

```
echo KIOSK_SESSION=/etc/opt/SUNWkio/NotePad.conf | /opt/SUNWut/sbin/utkiosk -i session
```

Session Name
The Key=Value pair of KIOSK_SESSION_LABEL declared in the example Kiosk Session Descriptor file does not apply to command line configuration. You must use the "short name" or full path to the Kiosk Session Type Descriptor file.

About Kiosk Session Prototypes

Typical Application Behavior

Generally speaking, any application that is ran for the first time will result in the end user being prompted with a variety different routines that must be acknowledge or accepted before the application can actually be used. This may include, but not limited to:

- End User License Agreement (EULA) acceptance screens
- Registration Wizards
- Preference Wizards

Typically, applications will store any state or configuration information generated by these routines in the user's home directory. This information is now available for any subsequent use of the application, and the user will not prompted for the same information again. Most end users are familiar with and expect this "first run" behavior of applications. Likewise, they expect not to have to go through those routines again the next time they launch the application.

Similar to first run routines, applications may perform certain tasks every time they are executed. These may include:

- Display a product "Splash Screen"
- Tip of the Day Screens
- Check for Update notifications
- New document wizards

Normally application behaviors such as these, along with other preferences, are configurable. These user preferences are traditionally stored in users home directory using the same methods as the information collected by the first run routines.

Default Application Behavior in a Kiosk Session

By default, the initial execution of an application in a Kiosk Session will behave in exactly the same manner as a regular session. The key difference is that unlike a regular session, any application state or preferences selected by the end user are never stored. Without this information, the application will always behave as if it were being ran for the very first time. This is by design since:

- There is no correlation between the actual person using the application and the Kiosk User Account executing the application
- The Kiosk User Account's home directory is deleted at the end of the session. All application state or preference settings will be deleted as well.
- Kiosk User Accounts are dynamically assigned to Kiosk Sessions.

See How Kiosk A Kiosk Session Works for more information.

While this behavior is by design to limit the impact then person using a Kiosk Session can have the system, it doesn't provide for a very good end user experience. Frustration and displeasure would quickly grow if every time a Kiosk Session was accessed, the end user had run a configuration wizard or accept a license agreement.

To allow applications to run in a specific state and configuration, Kiosk Sessions can use Kiosk Session Prototypes to provide any required application state, configuration files, and preference settings to application.

The Role of Kiosk Session Prototypes

The primary design goal of any Kiosk Session Type is to have the application work as expected in an identical and predictable state for all users. Users would not expect configure applications each time they used them. Nor would allowing end user configuration of the application allow it to work in a identical and predictable state for all users. Having 100 different users configuring and specifying preferences for an application would most likely be anything but identical.

As we have already learned, Kiosk Session Type Descriptor files specify the application to launch using the KIOSK_SESSION_EXEC key=value pair. The Kiosk Session Descriptor file can use the KIOSK_SESSION_PROTOTYPE key=value pair to specify a the location of a directory that contains all the necessary application state, configuration files, and preference settings for the specified application.

Before the Kiosk Service allows the Kiosk User Account to execute the application defined in the Kiosk Session Type Descriptor, it will copy the entire contents of the prototype directory specified to to the Kiosk User Account's home directory. RWX permission on the files will be preserved as they exist in the prototype directory, but the ownership of the contents will be changed to that of the Kiosk User Account assigned to the session.

Determining the what files are needed for the Session Prototype

Typically the location and contents of files that control the configuration of desktop applications are not publicly documented. Primarily this is because the application always provides a "front-end" to these files such as a Preferences or Options menu. This lack of documentation can present a challenge in determining what files need to go into the Session Prototype directory.

The best way to determine what configuration files are required is to run the actual application from a terminal under a Kiosk User Account. This is preferred for determining Session Prototype contents over other methods because:

- If the application being tested closes, the kiosk session will continue to operate
- Many preferences are not written until the application is closed
- Easier to evaluate changes to the Kiosk User Account's Home directory. No guess which Kiosk User Account is being used
- Ability to become super user from the terminal to make a copy of the required files or move them to to the Kiosk Session Prototype directory

After stepping through the first run routines and changes to preferences, track the changes that occur to to the files in the home directory.

There may also be the need for configuration files of ancillary applications into the Kiosk Session Prototype directory. While it's easy to assume that a Kiosk Session Type that runs a Firefox web browser will most likely need Firefox configuration files, other applications that get called by web browser may need configuration files as well. This may include configuration and preference settings for things like Adobe Acrobat Reader, Adobe Flash, Real Player, etc.

Methods for determining which files should be included in the Session Prototype are covered in the Kiosk Session Tutorial section.

Kiosk Session Prototype Location

The repository for these types of files are called Kiosk Session Prototypes and are configured on a per Kiosk Session Type basis. The default location for Kiosk Session Prototypes is /etc/opt/SUNWkio/prototypes

Kiosk Session Prototype Characteristics

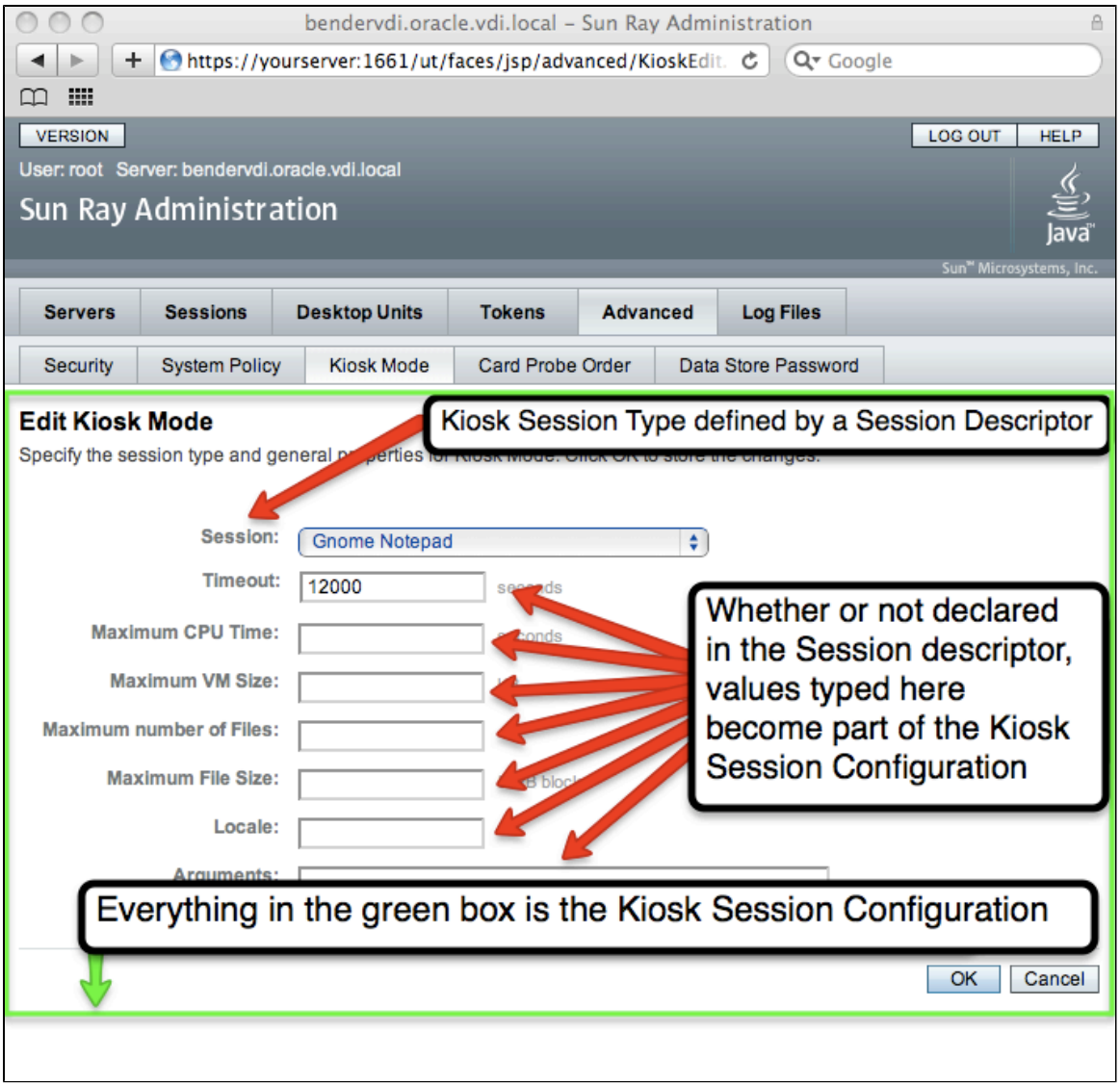
Kiosk Session Prototypes have certain characteristics:

- Kiosk Session Prototype directories are stored in /etc/opt/SUNWkio/prototypes/<name>.
    - Where <name> should match the short name of the Kiosk Session Type Descriptor
- If the name of the Kiosk Session Prototype names does not match the short name of the Kiosk Session Type, the full path to the KIOSK_SESSION_PROTOTYPE can be specified in the Kiosk Session Descriptor file.

Kiosk Session Configuration

Where Kiosk Session Descriptors define, at a minimum, an executable or "application" which will be executed for the Kiosk Session Type, Kiosk Session Configuration defines, , at a minimum, what Kiosk Session Descriptor should be used.

Kiosk Session Configurations are stored in the Sun Ray Data Store. Most who familiar with Kiosk Mode, may be unaware of the concept of the Kiosk Session Configuration. In reality, they create or change the "Default Kiosk Session Configuration" every time the Sun Ray Admin GUI is used to configure Kiosk Mode. Consider the Edit Kiosk Mode screen in the Sun Ray Admin GUI:



Kiosk Session Configurations are known by their names, and the name of the Default Kiosk Session is called "session". All defined Kiosk Session Configurations that are stored in the Sun Ray Data Store can be viewed using the 'utkiosk' utility.

```
# /opt/SUNWut/sbin/utkiosk -l
session
```

This shows that only the Default Kiosk Configuration is defined.

Likewise, using 'utkiosk', the Kiosk Session Type Descriptor and other Kiosk Configuration Settings (i.e. limits, additional arguments, etc which are set in the Sun Ray Admin Tool) used by the Kiosk Session Configuration can be viewed.

```
# /opt/SUNWut/sbin/utkiosk -e session
KIOSK_SESSION=NotePad
KIOSK_SESSION_TIMEOUT_DETACHED=12000
KIOSK_ENABLED=yes
```

The output shows that the Default Session Configuration for this machine is for a Kiosk Session Type that launches a Notepad that was created in the section that covered Kiosk Session Descriptors. The KIOSK_SESSION variable is pointing the short name of the Kiosk Session Descriptor, /etc/opt/SUNWut/sessions/Notepad.conf. To know whether KIOSK_SESSION_TIMEOUT_DETACHED was set using the Kiosk Session Descriptor or via the Kiosk Session Configuration (i.e. the Sun Ray Admin GUI in most cases), one would have to compare the contents of the actual Kiosk Session Type Descriptor file with the Kiosk Session Session Configuration.

To examine the content of any Kiosk Session Type Descriptor, the 'kioskdesc' utility can be used

```
# /opt/SUNWkio/bin/kioskdesc print -s Terminal
KIOSK_SESSION_EXEC=/usr/bin/gedit
KIOSK_SESSION_LABEL=Gnome NotePad
KIOSK_SESSION_DESCRIPTION="NotePad"
```

Since KIOSK_SESSION_TIMEOUT_DETACHED is not listed here, it must have been manually entered in the Sun Ray Admin GUI and is now part of the Kiosk Session Configuration, or more specifically the Default Kiosk Session Configuration.

Multiple Session Configurations
Sun Ray Server 4.1 or later supports defining multiple Kiosk Session Configurations. These configurations can be assigned to different Token IDs, allowing for a very granular application of different Kiosk Session Types. This topic is covered in the Advanced Section

Session Type Design Considerations (All Topics)

Contents

Session Type Design Considerations (All Topics)

Before a new Kiosk Session Type can be created, use cases and requirements should be planned out. Not only will this aid in creating the Session Type, but will help refine any possible rough edges with the initial design.

Support Statement for 3rd Party Provided Kiosk Session Types

Oracle fully supports the creation of Kiosk Session Types by 3rd parties for use with Sun Ray Server Software. Any documented interfaces, unless specifically noted otherwise, used by the 3rd Party Provided Kiosk Session Type are supported. For Customers with support contracts, this means service calls can be logged if a reproducible issue occurs with any supported interfaces do not perform as documented.

Oracle does not provide support for the actual application used in the custom Kiosk Session Type. The definition of an "application" as it pertains to support of the Kiosk Session Type is the specified script or binary that is referenced by the 'KIOSK_SESSION_EXEC' variable in the custom Kiosk Session Type Descriptor.

The only exceptions are Kiosk Session Types that are provided with Oracle Products. This includes Kiosk Session Types included with:

- Sun Ray Connector for Windows OS
- Sun Ray Connector for VMWare View Manager
- Oracle Virtual Desktop Infrastructure
- Kiosk Session Types bundled with Sun Ray Server Software for Solaris

Example: A 3rd party wishes to develop a Kiosk Session Type that uses the Citrix ICA Client to display a full screen Microsoft Windows Desktop. Oracle will support the use of Kiosk Mode for this purpose along with the Kiosk Session configuration files to create this type of Session. Oracle will not provide support for the Citrix ICA Client executable, any custom shell scripts written to launch the Citrix ICA Client, or the configuration files required by the Citrix ICA Client for operation.

Choosing an Application

When creating a custom Kiosk Session Type, the minimum requirement is that application specified in the Session Type Descriptor is compatible for use on the on a supported operating system that Sun Ray Server Software is installed on.

For instance, while it not possible to run Microsoft Word natively on Solaris or Linux, a Kiosk Session could run Oracle's Secure Global Desktop or Citrix ICA and access Microsoft Word as a published application.

Keep in mind that the term "application" as it pertains to Kiosk Mode does not necessarily have to stick to the traditional meaning of the word. An "application" running under a Kiosk Session could be almost anything:

- A full screen Web Browser configured only to accesses a web based call center application
- A full Windows 7 Desktop accessed via an RDP client such as the Sun Ray Windows Connector
- A custom script that allows end users a choice of local applications to run such as OpenOffice, An Email Client, and a Web browser
- A Java based front end to a VDI allowing access to a multitude of different desktops and therefore a multitude of different applications

Application Security Concerns

Since Kiosk Mode bypasses the system login mechanism, you must consider the security of the applications provided via the kiosk session.

For example, adding an application such as standard Gnome Terminal could provide users with access to a command-line interface of the underlying operating system and is not advised. However, deploying a terminal that was pre-configured via a script to connect to a different host for training or system administration purposes and required login credentials to that host would be acceptable.

Application exploits should also be researched and fixed wherever possible to prevent unauthorized access to other applications or the underlying operating system. An example of such an exploit used to be possible with Netscape Navigator. The print function allowed the end user to modify the actual print command (i.e. 'lp') used by the browser. Thus a user could replace the print command with any application or script the system and run it once they clicked print.

Application Suitability

When choosing an application to deploy with Kiosk Mode, keep in mind that this is a multi-user environment and certain questions must be answered:

- Will this application run for more than one user at a time?
- Does this application require permissions to resources on the system that are unavailable to the kiosk user?
- Can the application run as a kiosk user without requiring that user to run an installation or setup program?
- Does this application rely on hardware requirements not found in a Sun Ray environment such as a 3D GPU?

Application Scalability

Some applications, such as those that are graphically intensive, might run very slowly on a Sun Ray client or might consume too many system resources. While one user running this application might not negatively impact the system, several hundred could Always test how the application performs and note its system resource requirements at scale. See the Best practices section on how to perform scalability tests.

Example Digital Signage Kiosk Session Type (All Topics)

Contents

Example Digital Signage Kiosk Session Type (All Topics)

Digital Signage Tutorial

In this example, A large health care facility uses Sun Rays as their default desktop. All staff members are issued smart cards to assist with the compliance of the Health Insurance Portability and Accountability Act (HIPAA) Privacy Rule, which states that patient data can never be left unattended on computer display. The current Sun Ray System Policy is set for No Access for Non-Card based sessions. Therefore when a Sun Ray is not in use, the screen just displays an "Insert Card" icon.

Instead just displaying an icon, the Human Resources Department would like to use the Non-card session as an electronic billboard. When not in use, Sun Rays could display important information about workplace health, safety, waste management, and energy saving tips to the staff.

Digital Signage Requirements

Our Digital Signage Kiosk Session Type has the following requirements:

- The content will be created by the Art Department in OpenOffice Impress and provided by in PDF format
- The PDF will be located on a centralized server and can be accessed via NFS path of /centralserver/Billboard/Kiosk.pdf
- No product splash screens should appear.
- The content should be displayed fullscreen in Adobe Reader presentation Mode
- Content should be loaded automatically without any end user interaction
- The content should should loop continuously
- The mouse cursor should be hidden while the content is display
- Users should not be able to exit the presentation by touching the keyboard or moving the mouse, only by inserting a smart card.

Digital Signage Kiosk Session Descriptor

Now that the requirements have been provided, the first thing that needs to be done is to create a Session Type Descriptor for the “Digital Signage” session type.

Create a new file in /etc/opt/SUNWkio/sessions called Billboard.conf

The Kiosk Session Type Descriptor will use the following Key=Value Pairs

```
KIOSK_SESSION_EXEC=/opt/Adobe/Reader9/bin/acroread
KIOSK_SESSION_LABEL=Digital BillBoard
KIOSK_SESSION_DESCRIPTION="Adobe Acrobat setup in full screen mode"
```

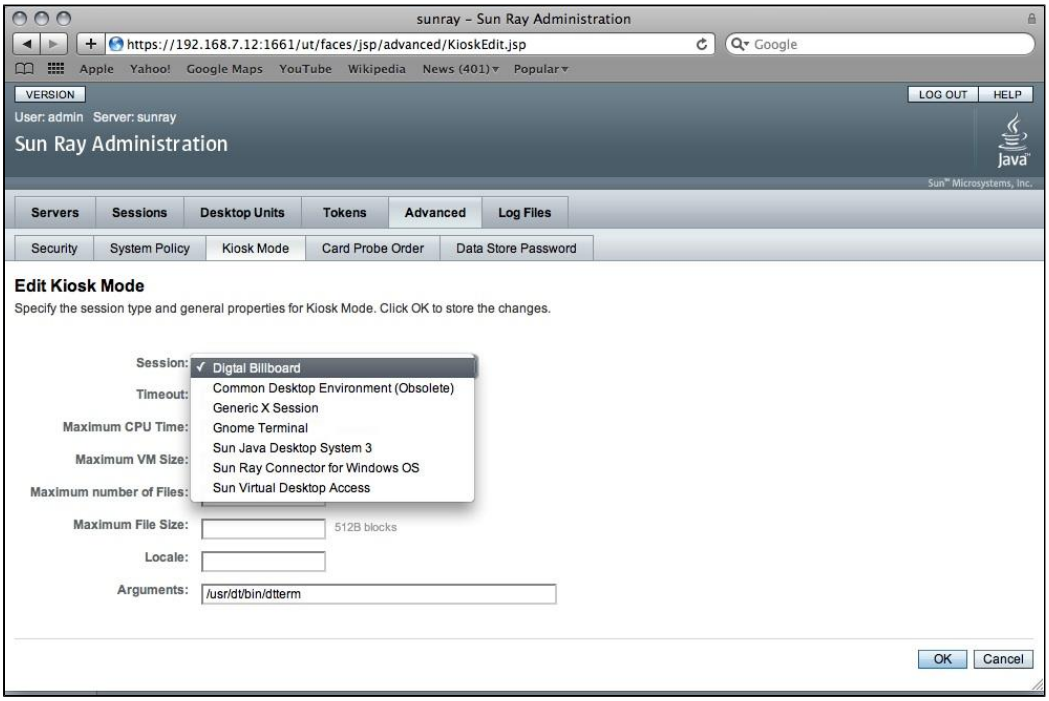
See Kiosk Session Descriptors for more information on Session Type Descriptor Key Value Pairs

Enable the New Session Type

Sun Ray Admin GUI Method

Using a web browser, access the Sun Ray Admin GUI at http://<yourserver>:1660. After logging in as an administrator, do the following:

- Select Advanced -> Kiosk Mode -> Edit
- Under the Session selection box the Kiosk Session Type Digital Billboard appears
- Select the Digital Billboard Kiosk Session Type and click OK to save



Command Line Method

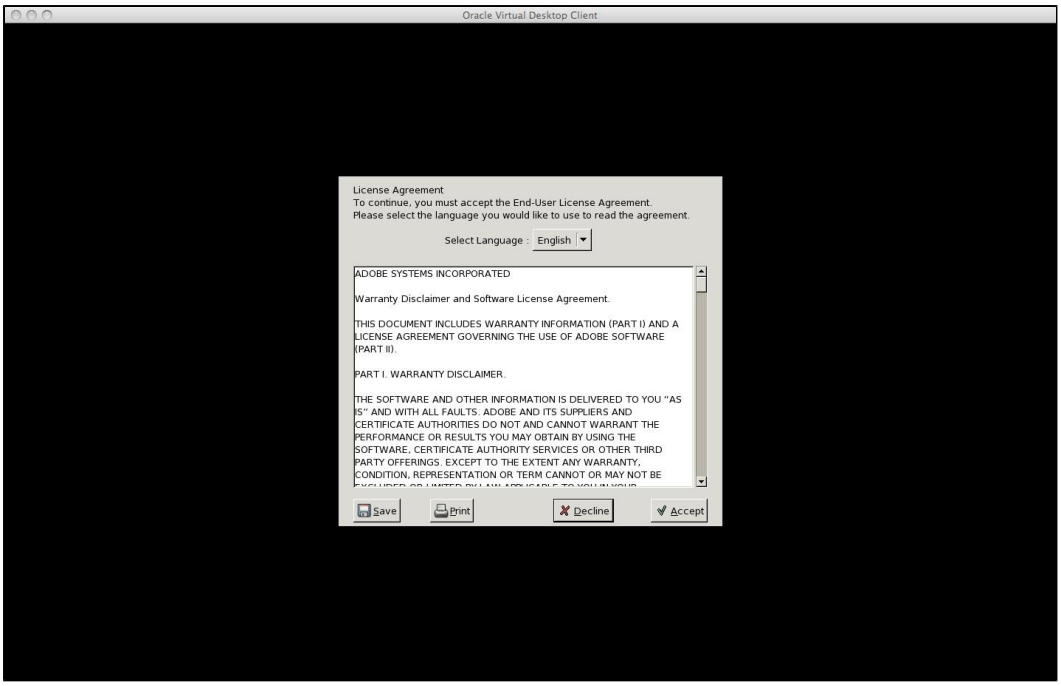
> Shortcut
> No need to use a web browser, it is possible change the Default Kiosk Session Type using the 'utkiosk' utility located in /opt/SUNWkio/bin. Since the Session Type Descriptor was stored where the Kiosk Service looks for them, the KIOSK_SESSION variable for the Default Kiosk Session Type can be set to the short name of the desired Kiosk Session Session Type

```
# echo KIOSK_SESSION=Billboard | utkiosk -i session
```

Access the Session

Using either a Sun Ray Client or the Oracle Virtual Desktop Client, start a new Sun Ray Session.

The following should screen should appear:



While Adobe Acrobat was successfully launched, none of the defined requirements for the Digital Signage Session Type requirements have been met.

Like the Typical Application Behavior covered in the the Kiosk Session Prototypes section, the application, as currently defined, is exhibiting first run behaviors and not performing as expected:

- A product splash screen comes up
- The user must acknowledge an end user license agreement (EULA)
- User interaction is required select a file
- The file when selected is not displayed in full screen mode
- Displaying the file in presentation mode is a manual process
- Hitting escape allows the slide show to exit
- The mouse cursor is visible

If you get the following error when launching the Kiosk Session, you most likely have a typo in the Kiosk Session Descriptor file.

Debugging problems with the Digital Signage Session Type

As noted when then the Digital Signage Session Type was accessed for the first time, a few problems need to be debugged:

- A product splash screen comes up
- The user must acknowledge an end user license agreement (EULA)
- User interaction is required select a file
- The file when selected is not displayed in full screen mode
- Displaying the file in presentation mode is a manual process
- Hitting escape allows the slide show to exit
- The mouse cursor is visible

Understanding the application

Creating custom session types often requires that the administrator assume the role of a detective to determine default application behavior and how to change it.

When investigating an application, it is very useful to run the application under a Kiosk User Account, but in a different manner than having the application defined for a Kiosk Session Type. The most flexible and robust way to debug an custom Kiosk Session Type is to run the defined application from the command line as a kiosk user via a terminal. In order to do this, we need to create a kiosk session type that just launches a terminal.

Refining the custom session type

Using the "Terminal Method" to launch the application, it is now possible to open and close the application without the Kiosk Session shutting down due to a critical processes exiting. This is important since many application state settings and user preferences do not get written to configuration files until the application is closed.

While the initial run of Abode Acrobat Reader using the Terminal Method, the application behaved just as it did when it was defined for the Digital Signage Session Type. However since the Terminal Method allows us to open and close the application at will. Subsequent runs of the program show that the splash screen only runs during the initial invocation and turns it self off for subsequent execution. Likewise, the EULA acceptance screen only appears once.

Typically a settings file that denotes that the user has accepted a EULA gets stored somewhere that is accessible to the current user. Usually these types of files are located in a hidden directory of the product name in the users home directory. Application preferences such as full screen mode operation and other preferences are also stored is a similar manner. Finding this file is the role of the application detective.

When the Acrobat Reader program is run via a terminal in Kiosk Mode, the administrator can watch the changes that occur to files in the Kiosk Users Home Directory while changes are made to the application.

Upon first run of the Acrobat reader program from a terminal, one could see that the application created a new file structure in a hidden directory in the users home directory called $HOME/.adobe/Acrobat.

As the centralized PDF was open and closed and changes were made using the applications preferences menu, it was observed that the file /.adobe/Acrobat/9.0/Preferences/reader_prefs was changing with each close of the application. Looking at this properties of this file, two things are noticed:

1.It's an ASCII Text File
2.It's been modified very recently

```
# file reader_prefs;ls -l reader_prefs
reader_prefs:   ascii text
-rw-------   1 utku6     utkiosk        5836 Jul 12 23:20 reader_prefs
```

Since this is an ASCII text file, the contents can be viewed. Looking at the contents of reader_prefs, a few entries are interesting:

```
<</AVGeneral [/c <<     /AppInitialized [/b true]
>>]
/AVPrivate [/c <<       /ChooseLangAtStartup [/b false]
        /EULAAcceptanceTime [/i 1279004804]
        /SplashDisplayedAtStartup [/b true]
        /UnixLanguageStartup [/i 4542037]
        /showEULA [/b false]
>>]
/unixAppPathPreferences [/c <<  /unixAppLastFileOpenPathPreference [/s
(/centralserver/Billboard/Kiosk.pdf)]
>>]
/unixAppSizePreferences [/c <<  /unixAppOpenMaximizedPreference [/b true]
```

These entries, plus the timestamp, confirm that reader_prefs is the file that gets updated with many of the settings that control the application in a manner that would meet our requirements. Unfortunately some of the requirements are still not met.

Further refinements

Using Adobe Acrobat Reader's Preferences menu, and administrator can control startup behaviors such as:

- Show Splash Screen (General → Application Startup)
- Warn when documents attempt fullscreen (Fullscreen → Setup)

- Escape Key Exits (Fullscreen → Fullscreen Navigation)
- Loop After Last Page (Fullscreen → Fullscreen Navigation)
- Advance Every X seconds (Fullscreen → Fullscreen Navigation)
- Mouse Cursor Always Hidden (Fullscreen → Fullscreen Appearance)

Once the desired changes are made and the application has been closed, it is once again noticed that the the reader_prefs file has been updated. This file is a prime candidate for inclusion in a Kiosk Session Prototype. While the format of this file doesn't lend itself to easy recreation or editing, it size is negligible.

Unable to render {include} Couldn't find a page to include called: Determine Prototype Requirements

Final Session Refinement

A few problems still remain with the behavior of our kiosk application:

- Acrobat Reader is not launching a PDF
- Acrobat Reader is not launching full screen

The reason Acrobat Reader is not launching a PDF is simple, it has not been told what PDF to open. Acrobat Reader typically uses the file name as an argument to the acroread binary. However KIOSK_SESSION_EXEC key is only used to tell Kiosk Mode what application to start, and cannot be used arguments to be passed to the application other wise an error will result regarding improper kiosk configuration. Passing arguments to the application is the job of the KIOSK_SESSION_ARGS key in the Session Descriptor.

Now the Session Descriptor file must be told what arguments to pass to the KIOSK_SESSION_EXEC key.

Edit /etc/opt/SUNWkio/sessions/Billboard.conf and add the following Key

```
KIOSK_SESSION_ARGS=/centralserver/Billboard/Kiosk.pdf
```

Save the file

Tip
Instead of calling a binary directly and using KIOSK_SESSION_ARGS, it is considered a best practice to have the KIOSK_SESSION_EXEC call a script which not only calls the binary but can also pass arguments to the binary.

Note
The location of the PDF must be reachable by all servers in the Sun Ray Host Group

Now Digital Signage Session type shows the following behavior.

One final requirement remains, launching full screen.

Despite all the efforts and research, the fact is that Adobe Acrobat does not have a mechanism to start in fullscreen mode.

The ability to display a PDF in full screen mode on it's initial view is a declaration stored in the PDF itself. This requires that you either have the full version of Adobe Acrobat to change the PDF's initial view mode or a program that is capable of generating PDF's that has the ability to set this declaration.

For instruction on how to do this with Adobe Acrobat, please see the documentation .

For Oracle OpenOffice, Choose File → Export as PDF... → User Interface → Open in Full Screen Mode → Export

Do not use the PDF Toolbar button to export as you will not be able to full screen declaration to be set.

Once the PDF has been set to open initially as full screen Digital Signage Session Type works as expected in an identical and predictable state on every DTU.

Advanced Topics (All Topics)

Contents

Advanced Topics (All Topics)

Multiple Kiosk Session Configurations

Key Concepts

To better understand this feature, it is important that one understands the difference between Kiosk Session Configurations and Kiosk Session Descriptors.

Kiosk Session Configurations are stored in the Sun Ray Data Store. The contain information such as which Kiosk Session Descriptor to use, limitations set by the administrator, and any other KIOSK_SESSION_* variables that are not set in Kiosk Session Descriptor.

Kiosk Session Descriptors are the actual files that contain what applications or scripts to launch, and other KIOSK_SESSION_* variables that are listed here.

The notion of "session.conf" as an actual file to store Kiosk Session Configurations, as discussed in the session.conf man page, does not exist when SUNWkio is integrated with Sun Ray Server Software. The Kiosk Session Configuration is always stored in the Sun Ray Data Store.

Based on the existing documentation, it is easy to confuse these two items. Both may contain many of the same KEY=VALUE pairs, in fact there are only two variables that are specifically designed for Kiosk Session Configuration which are the KIOSK_SESSION and the KIOSK_ENABLED variables. To add to the confusion, they share the same ".conf" file extension.

Key Value Pairs
If KEY=VALUE pairs are duplicated in the Kiosk Session Configuration and Kiosk Session Type Descriptor files, the values set in the Kiosk Session Configuration will take precedence.

Feature Overview

Kiosk Mode offers the ability to assign different Kiosk Session Configurations to different Token IDs. The concepts and use cases regarding how Kiosk Mode can be applied to sessions, whether via System Policy or on a per Token ID basis is discussed in the section How is Kiosk Mode applied and further information is available here on Token IDs.

Even though this feature is very powerful, it is one that is relatively unknown. There are two reasons for this, the first being that it is a fairly new feature, arriving in SRSS 4.1.

Prior to SRSS 4.1, the Session Type selected via the Sun Ray Admin GUI was the only Session Configuration that could be used. Thus, the Kiosk Session Type that is selected using the Sun Ray Admin GUI becomes the "Kiosk Session Configuration". Using an Oracle provided Kiosk Session

Type, such as the one for Sun Ray Windows Connector, was an all or nothing prospect. Sun Ray Clients either displayed that specific Kiosk Session Type or they ran a Regular Session with the only granularity offered between the two being provided at the System Policy level based on Card or Non-card use.

While many developers of custom Kiosk Session Types could use logic in their scripts to run different applications for different Token ID's, different MAC addresses, etc, it was still only possible to have one Kiosk Session Configuration available. This meant that the scripts used in various Kiosk Session Types could get very complex with if/then logic and case statements.

The second reason for its "unknown feature" status, is because there is no place in the Sun Ray Admin GUI to administer the feature today. It is configured via the command line interface. Most administrators only use the Sun Ray Admin GUI because, for historical reasons of only offering one Kiosk Session Configuration, that's the only thing they have ever used. In short, one could say it's a "hidden" feature.

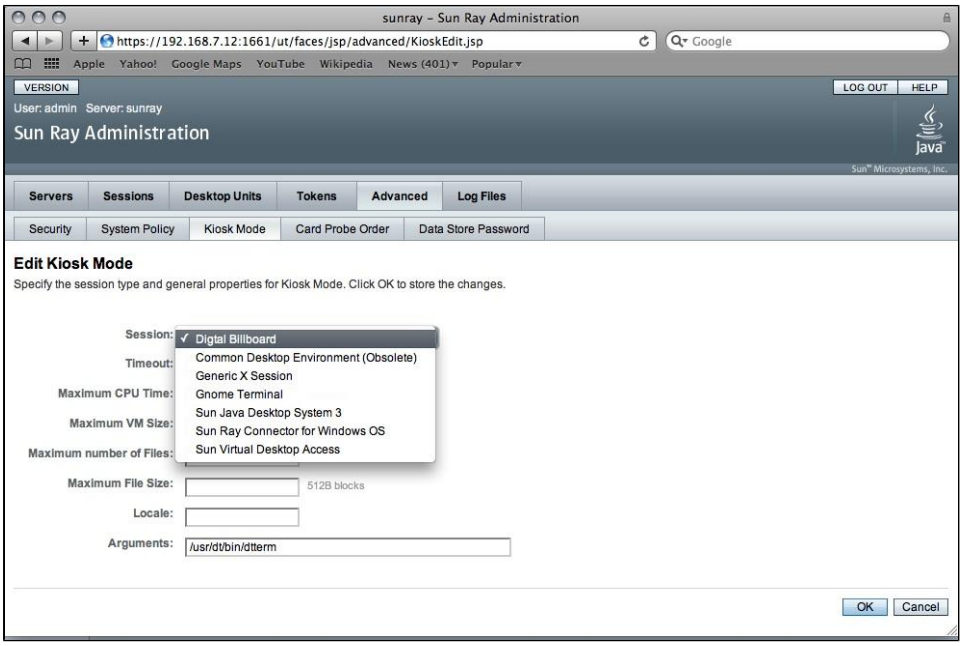
Default Kiosk Session Configuration

Like described above, if Kiosk Mode is enabled at the System Policy level, the Kiosk Session Type chosen via the Sun Ray Admin GUI becomes the "Kiosk Session Configuration". The actual name of this Kiosk Session Configuration in the Sun Ray Data Store is called "session".

Since 4.1 offers the ability to assign different Kiosk Session Configurations to different Token ID, a more descriptive term is used, and that is "Default Session Configuration". It's called the Default Kiosk Session Configuration because it gets assigned to all Token ID's marked for Kiosk Mode that have not been assigned an overriding Kiosk Session Configuration.

"Non-Default" or Alternate Kiosk Session Configuration

"Non-Default" or Alternate Kiosk Session Configurations are exactly what they sound like. They are any Kiosk Session Configuration that is not currently stored in the Sun Ray Data Store with the default name of "session". However, the administrator can create multiple Alternate Kiosk Session Configurations. Consider the following drop down list of available Kiosk Session Type Descriptors from the Sun Ray Admin GUI:



Any of the listed Kiosk Session Type Descriptors can be used to create a new Alternate Kiosk Session Configuration, which can be assigned to individual Token ID's.

Besides the Sun Ray Admin GUI, all available Kiosk Session Type Descriptors can be listed by their short names with the 'kioskdesc' utility.

```
# /opt/SUNWkio/bin/kioskdesc list -s

cde
Generic
Terminal
jds3
uttsc
vda
```

Short Names
The Kiosk Framework defines the locations where the Kiosk Service will look for files. Any files stored those defined locations can be referenced using what is a called a "short name". Short names are just the prefix of the Kiosk Session Type Descriptors or Directory Names of other KEY=VALUE pairs. The short name of the Kiosk Session Type Descriptor /etc/opt/SUNWkio/sessions/NotePad.conf would be "NotePad"

How to assign an Alternate Kiosk Session Configuration to a Token ID.

Two things are needed to assign Alternate Kiosk Session Configuration to a Token ID and simply enough those are 1) An Alternate Kiosk Configuration and 2) A Token ID that is registered in the Sun Ray Data Store. Note that Kiosk Mode does not have to be enabled via the System Policy to assign a Kiosk Session Configuration to a Token ID.

Registered Mode vs Registered Tokens
Don't confuse a registered token with Registered Mode System Policy. While token registration is byproduct of using Registered Only Mode, it is not the only method of registering tokens. One could also use a Token Reader, the Sun Ray Admin GUI, or the 'utuser' utility

Example: Assigning an Alternate Kiosk Session Configuration a pseudo Token ID

For simplicity, this example will one of the existing "NotePad" Kiosk Session Type Descriptor that was previously defined in the section that covered Kiosk Session Descriptors. The Kiosk Session Type Descriptor file can be referenced either using the full path and name of the file or it's short name:

```
/etc/opt/SUNWkio/sessions/NotePad.conf
```

Create a New Kiosk Session Configuration based on the Kiosk Session Type Descriptor

As previously discussed, there is only one Kiosk Session Configuration stored in the Sun Ray Data Store by default. This is the Default Session Configuration, thus it is stored with the name of "session". All of the defined Kiosk Session Configurations that are stored in the Sun Ray Data Store can be viewed using the 'utkiosk' utility.

```
# /opt/SUNWut/sbin/utkiosk -l
session
```

Likewise, using 'utkiosk', the Kiosk Session Type Descriptor and other Kiosk Configuration Settings (i.e. limits, additional arguments, etc which are set in the Sun Ray Admin Tool) used by the Kiosk Session Configurations can be viewed.

```
# /opt/SUNWut/sbin/utkiosk -e session
KIOSK_SESSION=vda
KIOSK_SESSION_TIMEOUT_DETACHED=12000
KIOSK_ENABLED=yes
```

The output shows that the Default Session Configuration for this machine is Oracle VDI. The 'KIOSK_SESSION=vda' is the short name Kiosk Session Type Descriptor, /etc/opt/SUNWut/sessions/vda.conf. To know whether KIOSK_SESSION_TIMEOUT_DETACHED was set using the Kiosk Session Type Descriptor or via the Kiosk Session Configuration (i.e. the Sun Ray Admin GUI in most cases), one would have to compare the contents of the actual Kiosk Session Type Descriptor file with the Kiosk Session Session Configuration.

To examine the content of any Kiosk Session Type Descriptor, the 'kioskdesc' utility can be used

```
# /opt/SUNWkio/bin/kioskdesc print -s vda
KIOSK_SESSION_ARGS=-- -r usb:on -E wallpaper -E theming
KIOSK_SESSION_DESCRIPTION=A full screen virtual desktop session. Additional applications are
-n -o -t -w] -- [<uttsc options>] ${vda_desc4}
KIOSK_SESSION_DIR=/etc/opt/SUNWkio/sessions/vda
KIOSK_SESSION_EXEC=/etc/opt/SUNWkio/sessions/vda/vda
KIOSK_SESSION_LABEL=Oracle Virtual Desktop Infrastructure
```

To override the Default Session Configuration for this for a particular Token ID, an Alternate Kiosk Session Configuration needs to be stored. This Alternate Kiosk Session Configuration will make use of the NotePad Kiosk Session Type Descriptor.

In order to provide clarity and prevent confusion between the Kiosk Session Type Descriptor files and Kiosk Session Configurations, the Alternate Kiosk Session Configuration will be called "GnomeNotepad".

At a minimum, an Alternate Kiosk Session Configuration can be created by setting the KEY=VALUE variable of KIOSK_SESSION to the short name (or full path) of an Kiosk Session Type Descriptor and then passing that to the 'utkiosk' utility using a unique name for the Alternate Kiosk Session Configuration.

Kiosk Session Configuration Names
Use a name that starts with a Capital letter. File names that use all-lowercase letters are reserved.

```
echo KIOSK_SESSION=NotePad | /opt/SUNWut/sbin/utkiosk -i GnomeNotePad
```

Where "NotePad" is the short name of the Kiosk Session Descriptor file: /etc/opt/SUNWkio/session/NotePad.conf

And "GnomeNotePad" is the name of the Kiosk Session Configuration that will be stored in the Sun Ray Data Store (SRDS). This name can be used later for overriding the Default Session Type.

Now that the new Kiosk Session Configuration has been stored in the SRDS, it can be listed and examined using 'utkiosk'

```
# /opt/SUNWut/sbin/utkiosk -l
GnomeNotePad
session
# /opt/SUNWut/sbin/utkiosk -e GnomeNotePad
KIOSK_SESSION=NotePad
KIOSK_ENABLED=yes
```

Assign the Alternate Kiosk Session Configuration to a Token ID.

For a token ID, this example will use a pseudo token of a Sun Ray Client. As previously mentioned, for any Sun Ray Client (Physical or OVDC), the Terminal ID is the identifying part of the pseudo Token ID. For physical Sun Ray Client, its MAC address is the same as it's Terminal ID. In this case the MAC address of the DTU that will be assigned the non-default Kiosk Session Type is 080020861234, thus the complete Token ID is pseudo.080020861234

First the Token needs to be registered. This can be done through the 'utuser' utility or the Sun Ray Admin GUI

'utuser' utility:

/opt/SUNWut/sbin/utuser -a "<token_id>, <server_name>, <server_port>, <name>, <other_info>"

```
# /opt/SUNWut/sbin/utuser -a "pseudo.080020861234,localhost,0,DTU 080020861234,Front Service
Counter"
Added one user.
```

Sun Ray Admin GUI:

Tokens -> New ->

In the "Enter Token Identifier Manually" box, enter your Token ID, i.e. pseudo.080020861234
In the "Name" box enter a name for this pseudo Token, i.e "DTU 080020861234"
In the "Other Information" box Enter any relevant information desired, i.e. "Front Service Counter"

> Override Feature
> If you only wish to override the Default System Policy, you can do so here by selecting Default, Regular, or Kiosk from the Advanced drop down.

Click OK

Second, assign the Alternate Kiosk Session Configuration via the 'utkioskoverride' utility:

/opt/SUNWut/sbin/utkioskoverride -r <Raw Token ID> -s <Session Type: Default, Regular, Kiosk> -c <Name of Alternate Kiosk Session Configuration>

```
# /opt/SUNWut/sbin/utkioskoverride -r pseudo.080020861234 -s kiosk -c GnomeNotePad
The session type has been successfully changed. Please note that changes
will only take effect the next time a session is started for the specified
token
```

> If you only wish to override the default System Policy, don't provide the -c option

Complete. Kiosk Mode is now enabled for this Sun Ray Client's Token ID. End any existing Session on the Client and it will come up in Kiosk Mode displaying "NotePad"

Advanced Kiosk Session Configuration Examples

The above is very simple example. It only uses a single KEY=VALUE pair for the Session Configuration. Any of the KEY=VALUE pairs listed in the session.conf man page or in the section describing Kiosk Session Type Descriptors can be used. This allows an administrator complete control over the Kiosk Environment. For example, it's possible to use different Kiosk Session Prototypes, different ulimits, for different Kiosk Session Configurations.

Using the above "GnomeNotePad" Session Configuration example, if a Kiosk Session Prototype Directory was not defined in the Kiosk Session Type Descriptor, or a different Prototype other than the one that was defined was desired, an Alternate Kiosk Session Configuration could be created specifying the KEY=VALUE pair variable KIOSK_SESSION_PROTOTYPE. Typically the more KEY=VALUE pair variables that get added, it is may be easier to work with a file. If so, one could create a temporary file and use that to store the desired KEY=VALUE pairs. For example, create a file called /tmp/MyConfig that will contain:

KIOSK_SESSION=NotePad
KIOSK_SESSION_PROTOTYPE=/etc/opt/SUNWkio/prototypes/NotePad

Then direct 'utkiosk' to use that file:

```
# /opt/SUNWut/sbin/utkiosk -i GnomeNotePad -f /tmp/MyConfig
# /opt/SUNWut/sbin/utkiosk -e GnomeNotePad
KIOSK_SESSION=NotePad
KIOSK_SESSION_PROTOTYPE=/etc/opt/SUNWkio/prototypes/NotePad
KIOSK_ENABLED=yes
```

Furthermore, assume that in this Prototypes directory, the Administrator placed a file called "ReadMe.txt" that should be opened at the start of the Kiosk Session. Arguments to the application specified in the Kiosk Session Type Descriptor could be passed from the Kiosk Session Configuration using the KIOSK_SESSION_ARGS variable. Add that variable to the file /tmp/MyConfig:

KIOSK_SESSION=NotePad
KIOSK_SESSION_PROTOTYPE=/etc/opt/SUNWkio/prototypes/NotePad
KIOSK_SESSION_ARGS=$HOME/ReadMe.txt

And again, direct 'utkiosk' to use that file:

```
# /opt/SUNWut/sbin/utkiosk -i GnomeNotePad -f /tmp/MyConfig
# ./utkiosk -e GnomeNotePad
KIOSK_SESSION=NotePad
KIOSK_SESSION_PROTOTYPE=/etc/opt/SUNWkio/prototypes/NotePad
KIOSK_SESSION_ARGS=$HOME/ReadMe.txt
KIOSK_ENABLED=yes
```

All this while the original Kiosk Session Type Descriptor of /etc/opt/SUNWkio/sessions/NotePad.conf never changed.

```
# /opt/SUNWkio/bin/kioskdesc print -s NotePad
KIOSK_SESSION_DESCRIPTION=NotePad
KIOSK_SESSION_DIR=
KIOSK_SESSION_EXEC=/usr/bin/gedit
KIOSK_SESSION_LABEL=Gnome Notepad
```

Advanced Session Type Examples

When looking for examples of what would be considered a more advanced example of a custom Kiosk Session Type, one does not have look any further than the software that is already installed. Both the bundled and unbundled session types provided by Oracle make for extremely useful reference designs. Copies of the descriptors are allowed as long copied version as a distinct name.

> Oracle does not support modification of any Oracle provided Kiosk Session Type. Since the provided Kiosk Session Types are part of installed packages, any changes you make will most likely be overwritten by a patch or upgrade. It is permissible to make a copy of a Oracle provided Kiosk Session Type. Support for this new Kiosk Session Type would be the same as any other custom Kiosk Session Type.

Oracle provides two Kiosk Session Types with Sun Ray Server Software for Solaris, both of which are locked down desktop environments. The first being a Common Desktop Environment (CDE) that is primarily for older Controlled Access Mode migrations and the second based on Oracle Java Desktop System 3, which provides a desktop environment with a more modern look and feel. Both of these Kiosk Session Types are fairly advanced in both their level of scripting and functionality since they also support the notion of additional application to be specified along with the primary session type. Both of these Kiosk Session Types are unique to Solaris, therefore are not available for SRSS for Linux OS.

Other Oracle products such as the Sun Ray Connector for Windows OS (SRWC), if installed, also come with their own session type. The SRWC Session Type is available for both Solaris and Linux. Oracle's Sun Ray Connector for VMWare View (SRVC) is a unbundled Kiosk Session Type for use with Sun Ray Server Software on Solaris.

Oracle VDI is a slightly different product as its unified installer not only installs SRSS and SRWC, but also configures the Sun Ray Server exclusively for Kiosk Mode Use. Like SRVC, Oracle VDI provides it's own Kiosk Session Type, but only for SRSS installations on Solaris.

If you are seeking information about the Oracle provided Kiosk Session Types that do not ship with SRSS, please see that products documentation.

> Copyright and License Notice
> If creating a new custom Kiosk Session Type based on one of Oracle's supplied Kiosk Session Types, all Copyright and License information in the original descriptors and scripts must be respected if you intend to sell or redistribute the copied session type.

Common or Default Prototypes

In addition to Session Type specific Prototype files, the Kiosk Service also has a Common or Default Prototype.

The Default Prototype is copied to ALL Kiosk User Account's home directories REGARDLESS of the Kiosk Session Type being used. The location of the Default Prototype directory is /etc/opt/SUNWkio/prototypes/default. Initially this prototype only contains a single file to disable Xscreensaver since the notion of "locking the screen" is incompatible with the use non-authenticated sessions. If Xscreensaver were to lock the screen, the end user would have no way to unlock it.

An administrator can use the Default Prototype to copy files or directories that may be required regardless of the Kiosk Session Type.

The Kiosk Configuration Framework

> System Internals
> Unless otherwise directed by Oracle Support, do not modify the files covered in this section. Please see the support note on this topic.

Various parts of the Kiosk Service use the Kiosk Framework Configuration file to identify the locations of files & directories required for operation. The configuration file is located at /etc/opt/SUNWkio/kioskrc

For instance, the web admin GUI automatically discovers and configures a Kiosk Session Type Descriptors located in the directory specified by the KIOSK_SESSIONS_DIR variable.

The Kiosk Configuration Framework also allows a Kiosk Session Type Developer to use "short names" instead of a fully qualified path when using command line utilities and Key=value pair variables in Kiosk Session Type configuration files.

> Note
> With the exception of enabling debug output, do not make changes to this file. Changes will be overwritten by patches to the SUNWkior package and may result in system instability.

File Locations Defaults

These settings specify the location of directories that contain configuration settings both the Kiosk Service and Kiosk Session Types

Key	Default Value	Description
KIOSK_CONFIG_DIR	/etc/opt/SUNWkio	A pointer to the top level directory where Kiosk configuration files are stored
KIOSK_VAR_DIR	/var/opt/SUNWkio	A pointer to the directory where variable files associated with the Kiosk Service are stored such as the Kiosk Session User's home directories and dynamically generated per display information
KIOSK_SESSIONS_DIR	$KIOSK_CONFIG_DIR/sessions	A pointer to the directory under which Kiosk Session Type descriptors are stored. These descriptors specify the primary session type to be used by the Kiosk Session
KIOSK_APPS_DIR	$KIOSK_CONFIG_DIR/applications	A pointer to the directory under which Kiosk Application Descriptors are stored. Kiosk Application Descriptors are used by certain Kiosk Session Descriptors to allow multiple applications to be run in a general purpose Kiosk Session Type that provides a locked down desktop operating environment such as Gnome, CDE, etc.
KIOSK_PROTOS_DIR	$KIOSK_CONFIG_DIR/prototypes	A pointer to the directory under which Kiosk Session Prototypes are stored.

Kiosk User Defaults

These settings get used when the Kiosk User Pool is created:

Key	Default Value	Description
KIOSK_USER_COMMENT	KioskSessionServiceUser	The comment for Kiosk User Accounts in /etc/passwd
KIOSK_USER_HOME	$KIOSK_VAR_DIR/home	The top level of the home directory for Kiosk User Accounts
KIOSK_USER_SHELL	/bin/sh	The specified shell (command-line interpreter/script host) for the Kiosk User Accounts

Configuration Consistency

Successful use of Kiosk Mode requires a consistent configuration across a Sun Ray Host Group. This ensures the user experience will not depend

on which server that their session gets assigned to.

Basic Configuration

Kiosk Session Configuration settings are kept consistent across the Sun Ray Host Group automatically. These settings are stored in the Sun Ray Data Store which get queried by the Kiosk Service to create Kiosk Session Configurations in real time as new Sun Ray Sessions start. In particular this covers:

- System Policy as it pertains to Enabling Kiosk Mode
- Kiosk core settings, e.g. resource limits, detach behavior, timeouts
- Kiosk Session Type used by the Kiosk Session Configuration

Environment Customization

All files and directories used by Kiosk Session Configuration (Session Type Descriptors, Prototype Directories, etc) need to be made available and kept consistent across the Sun Ray Host Group. It is the responsibility of the system administrator to ensure this requirement is met as there is no automatic method for keeping these resources consistent across a Sun Ray Host Group.

Application and Resource Provisioning

Any scripts, applications, or files referenced in the Kiosk Session Type Descriptor or to be included in the the Kiosk Session Prototype, need to be made available and kept consistent across the Sun Ray Host Group. It is the responsibility of the system administrator to ensure this requirement is met as there is no automatic method for keeping these resources consistent across a Sun Ray Host Group.

Tips and Tricks (All Topics)

Contents

Tips and Tricks (All Topics)

Using these Tips and Tricks

As with anything else, there is usually more than one way to do something. While all of the information that these tips and trick can provide is available using the publicly supported interfaces provided by Sun Ray Server Software ('utdesktop', 'utuser', 'utdesktop', etc), the use of those utilities can come with a performance cost. This cost is spelled out clearly in the Best Practices section that deals with using non-invasive scripting techniques.

While the publicly supported interfaces work as expected, they were never designed to be invoked by hundreds of different users at the same time. Essentially, that is exactly what occurs when using Kiosk Mode.

Thus the need for "Tips and Tricks". However, these tips and tricks also come with a cost.

New features and functionality may also change how these Tips and Tricks behave, or prevent them from working altogether. For instance, many developers who create Kiosk Session Types have used the following script snippet in the past to determine the MAC address of a physical Sun Ray Client:

```
#!/bin/sh
MYMAC=`cd $UTDEVROOT; pwd | sed 's/.*\(...........\)/\1/'`
```

Since the Oracle Virtual Desktop Client (OVDC) does not support devices, it does not have $UTDEVROOT directory structure. If a Kiosk Session uses this code is ran from an OVDC instance, the session will most likely fail.

It is critical to not only note the shell specified in these examples provided for, but also the fact that the examples are meant to run from within a script. Unix tools such as awk, sed, and grep may behave differently not only shell to shell, but distro to distro. Differences between Linux and Solaris implementations of those tools will be highlighted when necessary. Most tools will behave differently and require scripting techniques such as escape characters and proper handling of white space, therefore you cannot simply cut and paste examples to your command prompt and have them work as expect. Always invoke these examples from a script, using the shell specified in the example.

Support for Examples provided in this Guide

This guide contains information on public and private interfaces, as well as public and private details . An interface is any method used to interact with the provided software to obtain an desired result. Details for the purpose of support, are considered to be configuration files, start up files, log information, and design concepts.

Examples scripts in this guide may use interfaces and details that are not deemed public. While these interfaces and details are highly unlikely to change, they may in fact change or may be non-existent with future versions. Unless otherwise state, examples that use private interfaces or private details are considered unsupported. While care has been taken developing these examples, Oracle assumes no risk from their use.

Examples code that uses these private interfaces or details will highlighted by following note:

> Private Interface
> Additional information may be included with the note

Some examples in this guide use public interfaces that may have the potential to cause a performance issues in certain situations. Such examples will use the following warning:

> Performance Impact
> Additional information may be included with the warning.

Script to Determine the Display

While $DISPLAY is standard environment variable, it's not always useful to have the extra data that comes with it.

```
# echo $DISPLAY
:3.0
```

Many files created by SRSS to keep track of session information also contain useful information for Kiosk Development scripting purposes. Typically the session's display is listed as the primary identifier in these files, but does not contain all the extra data that is contained in the $DISPLAY variable. We can make the $DISPLAY variable more useful for scripting purposes using the following technique:

```
#!/bin/bash

# Get the display
MYDISP=$(echo $DISPLAY | sed 's/.*:\([^.]*\).*/\1/')
```

Script to Determine the Terminal ID

> Private Interface
> Please read this support note before using this example

Common scripting methods to discover the MAC Address of a Sun Ray client that rely on $UTDEVROOT will not work:

- In some NAT'd environments
- When ran from the Oracle Virtual Desktop Client (ODVC)

The MAC address of a Physical Sun Ray is the sames as the Terminal ID of a Physical Sun Ray. Using the Terminal ID will prevent scripts from failing if the OVDC is used. We can get the current Terminal ID in an efficient and non-invasive manner if we first discover our display using the following technique:

```
#!/bin/bash

# Get the display
MYDISP=$(echo $DISPLAY | sed 's/.*:\([^.]*\).*/\1/')

# Get the MAC Address
MYTERMID=$(sed -n 's/^TERMINAL_ID=^.*\.//p' /tmp/SUNWut/config/dispinfo/$MYDISP)
```

Script to Determine the Model of the Sun Ray

Private Interface
Please read this support note

Kiosk Session Type Developers may find it useful to know which model of Sun Ray the session is being displayed on. For instance the Kiosk Session Type may display applications different if connected to Dual Display Sun Ray 2FS or 3 Plus. Likewise, the Developer may want to disable certain features, like USB-R, if those features are not supported when using the Oracle Virtual Desktop Client (OVDC)

We can get the current Model the session is running on in an efficient and non-invasive manner using the following technique:

```
#!/bin/bash

# Get the display
MYDISP=$(echo $DISPLAY | sed 's/.*:\([^.]*\).*/\1/')

# Get the Model
MYMODEL=$(sed -n 's/^MODEL_ID=//p' /tmp/SUNWut/config/dispinfo/$MYDISP)
```

Determine the Raw Token ID

Performance Impact
Please read this support note

Every session has a unique Token ID. A Token ID is used to tell the Sun Ray Server where to send the screen to, and it can also be used to tell screen what to display from a Kiosk, Non-default Kiosk, or Regular Session. There are two types of tokens, Raw Token IDs (technically referred to Insert Tokens) and Logical Token IDs.

- Non-Card sessions an Raw Token ID of pseudo.<Terminal ID of Client>
- Card sessions have an Raw Token ID of <smart card model>.<unique identifier>

All Sun Ray Sessions have an environmental variable called $SUN_SUNRAY_TOKEN that by default will match the Raw Token ID. However, certain system policies (such as Registered Mode or Non-Smart Card Mobility) will translate $SUN_SUNRAY_TOKEN into a Logical Token ID. Logical Token ID's are not useful for developing Kiosk Session Types as they are Dynamic and only exist for the lifetime of the session.

Kiosk Session Type Developers may want to add logic to a script based on the Token ID. For instance:

- It may be desirable to have an application behave differently based on Card and Non-Card sessions.
- It may be desirable to look up information in the Sun Ray Data Store associated with a smart card token that can be used for scripting logic.

In order to do this type of scripting, you must know the Raw Token ID. You can translate a Logical Token ID back to the Raw Token ID using the following method:

```
#!/bin/bash
cardToken=$(/opt/SUNWut/sbin/utuser -p $SUN_SUNRAY_TOKEN | tail -2 | awk '/./ {print $1}')
```

Script to Determine the IP Address

Private Interface
Please read this support note

Performance Impact
Please read this support note

In some cases, the developer of a Kiosk Session Type may want to know the IP address of the current Sun Ray Client.

We can find the IP by using the Display to determine the Terminal ID, which we can parse the output of the 'utwho' utility for a match.

```
#!/bin/bash

# Get the display
MYDISP=$(echo $DISPLAY | sed 's/.*:\([^.]*\).*/\1/')

# Get the MAC Address
MYTERMID=$(sed -n 's/^TERMINAL_ID=^.*\.//p' /tmp/SUNWut/config/dispinfo/$MYDISP)

# Print IP of current Sun Ray
MYTERMIP=$(/opt/SUNWut/bin/utwho -c | awk '/$MYTERM\$/ { print \$4 }')
```

Best Practices (All Topics)

Contents

Best Practices (All Topics)

Best Practices while developing Kiosk Session Types

Kiosk Applications should always call scripts, not binaries

- Always test custom scripts prior running them in a Kiosk Session
- Test custom scripts from a terminal running in a Kiosk Session

Keep scripts as simple as possible

- Complex scripts lead to longer root cause analysis of a failure

Except where noted, avoid modifying any of the system files provided by Kiosk Mode

- Modifying Kiosk Service scripts or configuration files is not supported and will most likely break Kiosk Mode

Understand your application before using it in Kiosk Mode

- Required variables
- Configuration files
- Required system resources

Remember to keep the configuration consistent

- Kiosk scripts and associated applications must be copied to and installed on each server in the Sun Ray Host Group
- Prototypes must be copied to every server in the FOG
- More information on this topic on the Configuration Consistency page in the Advanced Topics section

Environmental Variables

- While start up scripts are sourced, you still may have to augment any custom scripts with application or session-specific variables such as PATH, LD_LIBRARY_PATH, XSEARCHFILES, etc

Use non-invasive utilities when writing scripts for Kiosk Session Types.

When Kiosk Mode is operating in Regular Mode, the initial "bring up" or restart of Sun Ray Services is is fairly light weight. The biggest draw on the system is displaying a simple greeter. From an end user standpoint, no other load is placed on the Sun Ray Server.

In Kiosk Mode, the load can be considerably greater since all of the processing required to get the Kiosk Session Type displayed is happening for potentially hundreds of session simultaneously.

If the Sun Ray Server is already taxed due to a massive number session creations, making calls to the data store that requires "current state" from the session manager just worsens the situation. The definition of "current state" as it pertains to this discussion is actual current state of any Sun Ray Client, Token ID, or User that is currently connected to a Sun Ray Host Group.

A query can be performed that shows the "Location" field of all registered Sun Ray Clients, or the same query can be done for currently connected Sun Ray Clients. The former, which in some cases can result in a lot more information to parse, is far less "invasive" from a system

performance standpoint. For example, utdesktop -lc invoke the session manager, while utdesktop -l is just a simple LDAP query.

Any query that requires the current state of any Sun Ray Client, Token ID, or User connected to a Sun Ray Host Group will, at some point, make a call back to the authentication daemon (authd). If authd is too busy to service this request, it will time out. Most ut* utilities that require a loop back call to authd have a time out of 5000 ms.

Thus, if any part of a custom Kiosk Session Type relies on this information, it will fail to complete. If the failure is fatal to the session startup, the Kiosk Session will be torn down and restarted. This will create even more load and compound any existing performance issues of the initial bring up of Kiosk Sessions.

Many of the supported public interfaces of Sun Ray Server Software such as the utilities 'utdesktop', 'utuser', and 'utdesktop' can do both simple low-impact LDAP queries of the Sun Ray Data Store and queries that report back current state. Typically the usage notes for each will have an option that specifically declares something close to currently connected.

If you are unsure if the command you wish to use requires call back to authd, do the following on a test system (heavily used production systems may make it hard to notice the call back). Open a terminal window and truss the java process running authd. Open another terminal window and run the desired command. If the command does a call back to authd, you'll notice a substantial increase the truss output.

Troubleshooting and Debugging (All Topics)

Contents

Troubleshooting and Debugging (All Topics)

Troubleshooting and Debugging with a Terminal

An administrator can define a Kiosk Session Type that just runs a terminal and assign that to a specific pseudo token or make it this custom session type portable by assigning it to a smart card. This simple, yet robust solution provides the administrator with easy command line access to the underlying OS where any tool can be run. If administrator privileges are needed, they are just a few keystrokes away while at a terminal.

From Kiosk Session debugging standpoint, not only does the Terminal Method allow applications to be tested under actual Kiosk User Account, it also can provide valuable "behind the scenes" information that is, by design, hidden from view:

- Non-fatal scripting errors
- Non-fatal library errors
- Permission problems
- Result codes
- Changes to the Home directory (excellent for determining what files are needed for Kiosk Session Prototypes)

How to create a simple Kiosk Session to launch a terminal:

Create a new Kiosk Session Descriptor File

/etc/opt/SUNWkio/sessions/gterm.conf

KIOSK_SESSION_EXEC=/usr/bin/gnome-terminal
KIOSK_SESSION_LABEL=Gnome Terminal

Create an Alternate Kiosk Session Configuration

```
echo KIOSK_SESSION=gterm | /opt/SUNWut/sbin/utkiosk -i GnomeTerminal
```

Override Default Policy and/or Default Kiosk Session Configuration

/opt/SUNWut/sbin/utkioskoverride -r <Raw Token ID of a registered pseudo or smart card token> -s kiosk -c GnomeTerminal

> More information
> The concept of multiple Kiosk Session Configurations is explained in depth in the Advanced Topics section

Increase Logging Levels

To generate richer logging of the Kiosk environment, do the following steps:

Set KIOSK_DEBUG=1 in /etc/opt/SUNWkio/kioskrc

```
#!/bin/sh
#
#*******************************************************************************#
# Copyright 2008 Sun Microsystems, Inc.  All rights reserved.
# Use is subject to license terms.
#
#*******************************************************************************#

#
## Warning:
##
## This file is for use internally by Kiosk Session Service. Alterations to
## this file are not supported.
##
## Any changes to this file may break the Kiosk Session Service and will most
## likely be lost during future updates or patch applications.
#

KIOSK_CONFIG_DIR=/etc/opt/SUNWkio
KIOSK_VAR_DIR=/var/opt/SUNWkio

KIOSK_APPS_DIR=$KIOSK_CONFIG_DIR/applications
KIOSK_SESSIONS_DIR=$KIOSK_CONFIG_DIR/sessions
KIOSK_PROTOS_DIR=$KIOSK_CONFIG_DIR/prototypes

KIOSK_USER_COMMENT=KioskSessionServiceUser
KIOSK_USER_HOME=$KIOSK_VAR_DIR/home
KIOSK_USER_SHELL=/bin/sh

KIOSK_TEXTDOMAIN=kiosk
KIOSK_TEXTDOMAINDIR=/opt/SUNWkio/lib/locale

KIOSK_DEBUG=1
```

Add the debug option to pam_kiosk lines in /etc/pam.conf

```
dtlogin-SunRay auth sufficient /opt/SUNWkio/lib/pam_kiosk.so.1 log=user ignoreuser debug
dtlogin-SunRay auth requisite /opt/SUNWkio/lib/pam_kiosk.so.1 log=user debug
dtlogin-SunRay account sufficient /opt/SUNWkio/lib/pam_kiosk.so.1 log=user debug
dtlogin-SunRay session required /opt/SUNWkio/lib/pam_kiosk.so.1 log=user debug
```

> Tip
> The option can be used multiple times increase the amount of debug output generated.

Change syslog to user.debug priority for Sun Ray Logging

Edit /etc/syslog.conf

Change:

```
user.info /var/opt/SUNWut/log/messages
```

To:

```
user.debug /var/opt/SUNWut/log/messages
```

Restart syslogd

```
# svcadm restart system-log
```

Watch /var/opt/SUNWut/log/messages for errors

Architectural Reference Guide (All Topics)

Contents

Architectural Reference Guide (All Topics)

Some of the information in this guide covers low-level design and implementation details. It is provided to allow the reader to gain a deeper technical understanding behind the architecture and design of the Kiosk Mode feature of Sun Ray Server Software. This information is provided for the sake of completeness only and is not intended to be modified in any manner.

> System Internals
> Unless otherwise directed by Oracle Support, do not modify files the files covered in this section. Please see the support note on this topic

Introduction

The Kiosk software package (SUNWkio) covered in the Architectural Reference Guide was developed with the intent that it could be used as a stand-alone product. While the software has never been available as a separate product, some of the documentation provided with the SUNWkio packages may reference configuration files and utilities that not used, or used differently than documented when SUNWkio is integrated with Sun Ray Server Software (SRSS).

For instance, when SUNWkio documentation discusses configuration, it covers the 'kioskconfig' utility and a single configuration file called session.conf. A lot of familiar concepts are covered such as enabling Kiosk Mode, Prototypes, and Applications to launch.

When Kiosk is integrated with SRSS, session configuration is stored in the Sun Ray Data Store instead of an actual file. This makes the configuration available to all members of the Sun Ray Host Group automatically. Additionally, the responsibility of enabling Kiosk Mode is defined at either the system policy level or on per-token basis.

There is also the potential for confusion between Session Configuration as discussed in the SUNWkio documentation, and Session Descriptors as discussed in both this guide and the SRSS documentation. It's easy to understand why: They both have .conf extensions and they both use many of the same KEY=VALUE assignments.

In short, while the concepts are the same, the implementation of that concept may be completely different. To this point, the documentation provided with the SRSS where Kiosk is concerned should supersede what would appear to be similar functionality in the SUNWkio man pages.

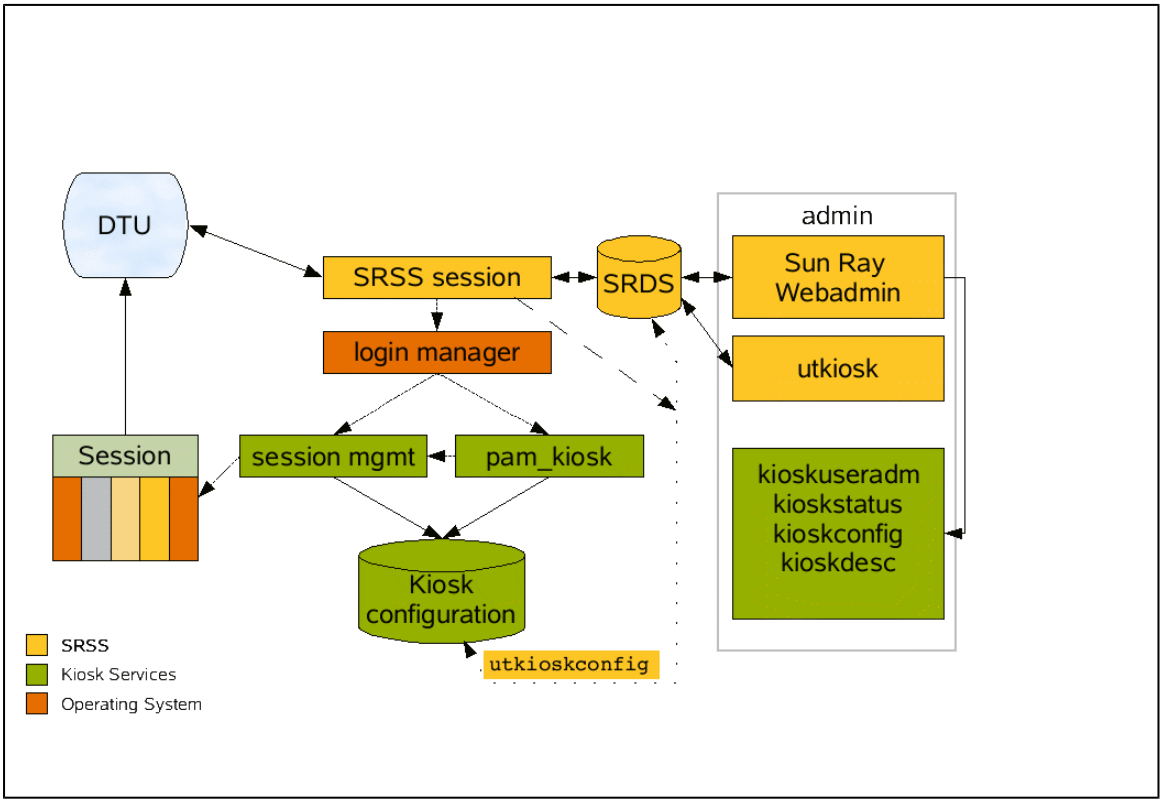
Architectural Overview

Previous sections in the Kiosk Software Development Kit have discussed at a high how a Kiosk Session works and how system determines if unauthenticated access should be granted, thereby initiating a Kiosk Session.

Some information that has been covered before will be discussed again in this section in much greater detail to help foster a better

understanding of the components of Kiosk Mode.

The following is graphical representation of the Kiosk architecture and brief description of components.



Login Manager

Kiosk Mode makes use of the standard login manager (e.g. GDM or dtlogin) of the system that Sun Ray Server is installed on. It is primarily responsible for:

- Invoking the Kiosk Pluggable Authentication Module pam_kiosk
- Starting graphical sessions

> Multi-Session GDM
> Some distributions of Linux ship with a version of GDM that is not multi-session aware. In this case, a version of GDM is included with Sun Ray Server Software that allows for multiple sessions

pam_kiosk

A pluggable authentication module (PAM) focusing on Kiosks Session specific tasks such as:

- Bypassing authentication if Kiosk Mode is enabled for the current display
- Selecting an available kiosk user account to run kiosk session
- Invocation of the Session Management module
- Marking session as a kiosk session
- Upon session completion declares Kiosk user accounts available for subsequent sessions

User Management Module

The user management module is responsible for:

- Allocation & de-allocation, selection & deselection of kiosk user accounts
- Responding to various "user account queries"
    - Identifies whether or not a given account is a kiosk account
    - Identifies whether or not a given kiosk account is active or not

Session Management Module

The session management module is responsible for setting up and tearing down a given kiosk session such as:

- Copy the correct Kiosk Session Prototypes to Kiosk User Account's home directory based on Kiosk Session Type
- Ensure the correct Kiosk Session Type will be started
- Starting any “run as root” processes associated with the Kiosk Session
    - e.g. The “Critical Application Monitor Daemon” (kioskcritd)

Kiosk Service Module

The Kiosk Service Module refers to framework and tools used to enable and configure a given Session Type. The framework is private interface used by the Kiosk Service to identify the locations of files & directories required for operation. The tools are both public and private interfaces to the Kiosk Service. The public interfaces allow for administrative control of the Kiosk Service. The private interfaces perform work for the Kiosk Service such as querying the Sun Ray Data Store at the beginning every Kiosk Session for information about:

- The Kiosk Session Type to be launched
- Any associated start-up applications with the Kiosk Session Type
- Any resource limitations associated with the Kiosk Session Type not declared in the Kiosk Session Type Descriptor

The results of the query create a Kiosk Session Configuration file that is used by the Kiosk Service to launch the Kiosk Session.

How pam_kiosk "marks" a Kiosk session

While the purpose and function of pam_kiosk is adequately covered in other parts of this guide, the Architectural Overview mentions that a Sun Ray gets "marked" by pam_kiosk as a Kiosk Session.

During the Kiosk PAM authentication sequence, the pam_kiosk module will evaluate whether or not the current session should be a Kiosk Session by:

- Checking for the existence of /var/opt/SUNWkio/config/$DISPLAY
- Checking if /var/opt/SUNWkio/config/$DISPLAY specifies KIOSK_ENABLED=yes

If the current session is identified as a kiosk session, pam_kiosk marks the session by setting the X property _SUN_KIOSK_STATUS on the root window of the relevant display. Similarly, this mark is removed by the pam_kiosk module when the Kiosk session exits.

Certain utilities such as 'kioskstatus' read this "mark" to see if a display is running a Kiosk Session. Since Kiosk Mode sources all Xsession.d or xinitrc.d scripts, administrators may find this utility useful to help decide which scripts shoul run in a Kiosk Session (or vice versa)

Architectural Kiosk Session Startup

Once a Sun Ray Client has connected to server, the following sequence occurs.

Kiosk Modules
Note the roles and responsibilities of the various Kiosk Software Modules previously covered in the Architectural Overview

- Login manager is started
- Xstartup/Init scripts offers a last chance to generate or adjust session configuration for the current display
- Login Manager invokes pam_kiosk
    - pam_kiosk consults display-specific configuration to determine if it is appropriate to start a kiosk session
    - If it is inappropriate to start a kiosk session at this time, pam_kiosk is done and acts in a pass through fashion having no effect on authentication or subsequent user session.
    - Otherwise pam_kiosk selects an available kiosk user account via the User Management module to run the kiosk session
    - pam_kiosk indicates that authentication is successful to login manager
- Login manager invokes pam_kiosk session module
    - pam_kiosk triggers session initialization via the pam_open_session phase
        - Session Management retrieves configuration data describing the kiosk primary session
        - Session Management deletes and creates the kiosk user's home directory and applies the kiosk default prototype to same

        - Session Management “installs” relevant primary session prototype into the kiosk user's home directory
        - Session Management executes the session-specific pre session script, if configured, which can be used for further setup
        - Session Management ensures the “Kiosk Session Starter” will be used as the user's session
        - Session Management starts the kioskcritd daemon as root
- Login manager starts session
    - Kiosk Session Starter launches the defined Session Type.
    - Defined Kiosk Sessions Type can do further setup as required
        - Install the prototypes to Kiosk User's home directory
        - Set any session specified resource limitations that are configured
    - Session starts any additional applications defined
    - Primary session runs and eventually ends
- Login manager invokes pam_kiosk session module
    - pam_kiosk triggers session cleanup during the pam_close_session phase
        - Session Management stops the kioskcritd daemon, if still running
        - Session Management executes the session specific post session script, if configured, which can be used for custom cleanup
        - Session Management terminates any residual processes from the kiosk user's session, deletes the kiosk user's home directory and any other files owned by the kiosk user
- Login manager restarts or ends the session

(include:Kiosk Session Starter}

User Management Module

The User Management Module is covered in depth in the section that discusses the concept of Kiosk User Accounts. The following information regarding how a Kiosk User Account is assigned to a Kiosk Session is provided for completeness.

How Kiosk User Accounts are assigned to Kiosk Sessions

When pam_kiosk determines that a session should be configured to run a Kiosk Session, it looks at /var/run/opt/SUNWkio/users/uid.next to get the next available Kiosk User Account. The username of the Kiosk User Account is stored in reservation file called /var/run/opt/SUNWkio/users/$DISPLAY.dpy where $DISPLAY is the display that the Kiosk Sesssion is using. When the session has ended, and the cleanup process has a occurred, this reservation file is deleted.

pam_kiosk also tracks the display number that reserved Kiosk User Account is using for the Kiosk Session in the file /var/run/opt/SUNWkio/users/$USER.session where $USER is username of the Kiosk User Account.

The use of two different reservation files allows Kiosk Mode to re-use the same Kiosk User Account for a new Kiosk Session on the same display should a problem occur logging the Kiosk User Account out. However, if another Kiosk Session never uses the same display, the Kiosk User Account will remain reserved until a reboot occurs.

Upon reboot, all reservation information in /var/run/opt/SUNWkio is purged.

The Kiosk Configuration Framework

System Internals
Unless otherwise directed by Oracle Support, do not modify the files covered in this section. Please see the support note on this topic.

Various parts of the Kiosk Service use the Kiosk Framework Configuration file to identify the locations of files & directories required for operation. The configuration file is located at /etc/opt/SUNWkio/kioskrc

For instance, the web admin GUI automatically discovers and configures a Kiosk Session Type Descriptors located in the directory specified by the KIOSK_SESSIONS_DIR variable.

The Kiosk Configuration Framework also allows a Kiosk Session Type Developer to use "short names" instead of a fully qualified path when using command line utilities and Key=value pair variables in Kiosk Session Type configuration files.

Note
With the exception of enabling debug output, do not make changes to this file. Changes will be overwritten by patches to the SUNWkior package and may result in system instability.

File Locations Defaults

These settings specify the location of directories that contain configuration settings both the Kiosk Service and Kiosk Session Types

Key	Default Value	Description
KIOSK_CONFIG_DIR	/etc/opt/SUNWkio	A pointer to the top level directory where Kiosk configuration files are stored
KIOSK_VAR_DIR	/var/opt/SUNWkio	A pointer to the directory where variable files associated with the Kiosk Service are stored such as the Kiosk Session User's home directories and dynamically generated per display information
KIOSK_SESSIONS_DIR	$KIOSK_CONFIG_DIR/sessions	A pointer to the directory under which Kiosk Session Type descriptors are stored. These descriptors specify the primary session type to be used by the Kiosk Session
KIOSK_APPS_DIR	$KIOSK_CONFIG_DIR/applications	A pointer to the directory under which Kiosk Application Descriptors are stored. Kiosk Application Descriptors are used by certain Kiosk Session Descriptors to allow multiple applications to be run in a general purpose Kiosk Session Type that provides a locked down desktop operating environment such as Gnome, CDE, etc.
KIOSK_PROTOS_DIR	$KIOSK_CONFIG_DIR/prototypes	A pointer to the directory under which Kiosk Session Prototypes are stored.

Kiosk User Defaults

These settings get used when the Kiosk User Pool is created:

Key	Default Value	Description
KIOSK_USER_COMMENT	KioskSessionServiceUser	The comment for Kiosk User Accounts in /etc/passwd
KIOSK_USER_HOME	$KIOSK_VAR_DIR/home	The top level of the home directory for Kiosk User Accounts
KIOSK_USER_SHELL	/bin/sh	The specified shell (command-line interpreter/script host) for the Kiosk User Accounts