VISHNU User Manual

COLLABORATORS			
	TITLE : VISHNU User Manual		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Benjamin Isnard, Daouda Traoré, Eugène Pamba Capo-Chichi, Kevin Coulomb, and Ibrahima Cissé	February 2011	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME
1	25/02/2011	First version of the VISHNU user manual which concerns only the UMS package.	SysFera

Contents

Chapter 1

Document presentation

1.1 Document objectives

This documents is a quick start guide of VISHNU software for users. The main objective of this document is to describe the VISHNU installation procedure and the way to use it.

1.2 Document structure

- Chapter 1 presents the document structure.
- Chapter 2 describes the VISHNU software (installation procedure, usage description and troubleshooting).
- Chapter 3 contains the VISHNU commands reference.
- Chapter 4 contains the C++ API reference.
- Chapter 5 contains the Python API reference.

1.3 References

- [D1.1b]: VISHNU "Spécifications techniques des besoins"

Chapter 2

Installation and usage

The VISHNU software is based on SysFera-DS which is an open-source middleware developped by SysFera. VISHNU is primarly designed to facilitate the access to high-performance computing resources by providing the following services:

- User management services (UMS): authentication and session management.
- Information management services (IMS): monitoring and control services.
- Tasks management services (TMS): submission of tasks (jobs) on computing resources.
- File management services (FMS): display and transfer of files between storage resources.

2.1 Installation procedure

This section details the main steps of the installation process, including the installation requirements [D1.1b]. VISHNU is based on SysFera-DS software which must be installed before.

Installation requirements:

- GCC V4.4.3
- CMAKE V2.6
- OMNIORB 4.1.4
- SYSFERA-DS V2.7 (available at: http://graal.ens-lyon.fr/DIET/)
- BOOST V1.45
- PYTHON V2.5
- JAVA V1.6
- SWIG V1.3
- LIBCRYPT

Installation procedure:

- Download the VISHNU install sources
- Decompress it and go to the vishnu directory

- Create a build directory and run CMake as follows:

  > mkdir build

  > cd build

  If your install directory is for example: /opt/vishnu

  > cmake -DCLIENT_ONLY=1 -DCMAKE_INSTALL_PREFIX=/opt/vishnu ..

  > make && make install

2.2 Sofware usage description

2.2.1 User account creation

The first step to access VISHNU is to request a new account to a VISHNU administrator. The only information required to create a new account is your full name and email address. You will automatically receive an email containing your userId and password.

2.2.2 Connection to VISHNU

To connect, use the **vishnu_connect** command in the shell terminal (all bourne shell are supported). The password received by email is temporary and must be changed at the first connection by using the **vishnu_change_password** command.

2.2.3 Reconnection to VISHNU

Reconnection is done using the **vishnu_reconnect** command. This command allows using an existing session that was previously opened but not closed. It makes it possible to simultaneously use the same session in different shell terminals.

2.2.4 Session management in VISHNU

After a successful call to the **vishnu_connect** command, a session is created. The session is required for calling any other commands. It avoids systematic authentification by userId and password. Only commands **vishnu_connect**, **vishnu_reconnect** and **vishnu_change_password** can be used outside a session by using userId and password. The **vishnu_list_history_cmd** command lists all the commands launched within a session.

To prevent unclosed sessions when the **vishnu_close** command is not used, the session is automatically closed on timeout or on disconnect (from the terminal).

2.2.4.1 Session close on timeout

In this mode, the session is automatically closed after an inactivity delay specified by the system or configured by the user using the **vishnu_configure_option** command.

2.2.4.2 Session close on disconnect

In this mode, the session is automatically closed when the shell terminal is closed. It is important to note that the system makes it impossible to close a session while commands are running. In this case, a session with automatic close on disconnect changes the close mode to automatic close on timeout.

2.2.5 Local user configuration management

2.2.5.1 Local user configuration creation

To access a UNIX account on a specific machine defined on VISHNU, the user must create a local user configuration by using the **vishnu_add_local_account** command. The **vishnu_list_machines** command gives information about the machines in which a local user configuration can be created or where a local user configuration has already been created. The information required to create a new local user configuration is: the userId, the machineId, the login of the UNIX account on the specified machine, the absolute path to the user's private SSH key (used for file transfers) and the home directory path.

The ssh public key of the machine named *"userId-machineId"* is returned and stored in the $HOME/.vishnu/localAccountPublicKey/ directory and must be added by the user in the ssh authorized key directory of the UNIX account. Doing this allows VISHNU to be directly connected on this UNIX account, running tasks as if it was the owner of the UNIX account.

2.2.5.2 Local user configuration update

All previous parameters used to create a local user configuration can be updated by using the **vishnu_update_local_account** command except for userId and machineId.

2.2.5.3 Local user configuration remove

A local user configuration can be removed by using the **vishnu_delete_local_account** command.

It is possible to display the local user configurations with the **vishnu_list_local_account** command. Other commands which are not cited above can be used to display information, such as the **vishnu_list_options** command, which displays all the options configured by the user, or the **vishnu_list_sessions** command, which displays information about the sessions.

2.3 Troubleshooting functions

"There is no session in this terminal" error can be solved by making a connection to VISHNU using the **vishnu_connect** command.

Chapter 3

Command reference

3.1 vishnu_connect

vishnu_connect — opens a session

Synopsis

`vishnu_connect` [`-h`] [`-p` *`closePolicy`*] [`-d` *`sessionInactivityDelay`*] [`-s` *`substituteUserId`*] *`userId`*

DESCRIPTION

Opening a VISHNU session is the first step before using any other VISHNU command. This command authenticates you. You must have been registered in the VISHNU system by an administrator. It also creates a session that remains open after the command is completed and until the session is either manually or automatically closed.

OPTIONS

-h ***help*** help about the command.

-p ***closePolicy*** is an option for closing session automatically. The value must be an integer. Predefined values are: 0 (UNDEFINED), 1 (CLOSE_ON_TIMEOUT), 2 (CLOSE_ON_DISCONNECT).

-d ***sessionInactivityDelay*** is the maximum delay in seconds between two user requests when the CLOSE_ON_TIMEOUT policy is set.

-s ***substituteUserId*** is an admin option which allows an admin to open a session as if she was another user identified by her userId.

ENVIRONMENT

VISHNU_CLOSE_POLICY The value of this environment variable represents the session close policy. Overriden by the -p option.

VISHNU_CONFIG_FILE Contains the path to the local configuration file for VISHNU.

DIAGNOSTICS

The following diagnostics may be issued on stderr and the command will return the code provided within brackets:

"Unknown user" [20]

"The userId is unknown" [21]

"The user is locked" [23]

"The user is not an administrator" [25]

"The value of the timeout is incorrect" [43]

3.2 vishnu_reconnect

vishnu_reconnect — reconnects to a session that is still active

Synopsis

`vishnu_reconnect` [-h] *`userId`* *`sessionId`*

DESCRIPTION

This command allows you to resume a session that has been opened previously and that has not yet been closed. You can disconnect from a session without closing it (for example if there are running commands in that session) by setting the session's close policy (at connection time) to CLOSE_ON_TIMEOUT. As sessions are linked to a specific client system, you cannot reconnect to a session that was opened on another client system.

OPTIONS

`-h` ***`help`*** help about the command.

ENVIRONMENT

`VISHNU_CONFIG_FILE` Contains the path to the local configuration file for VISHNU.

DIAGNOSTICS

The following diagnostics may be issued on stderr and the command will return the code provided within brackets:

"Unknown user" [20]

"The user is locked" [23]

"The session key is unrecognized" [28]

"The sessionKey is expired. The session is closed." [29]

"The session Id is unknown" [30]

"The machine does not exist or it is locked" [36]

3.3 vishnu_close

vishnu_close — closes the session

Synopsis

`vishnu_close [-h]`

DESCRIPTION

This command closes the session that is currently active in the terminal. It will return an error if there are still some active requests that were been submitted by the user during the session (e.g., job submission or file transfers). After the session is closed, it cannot be re-opened.

OPTIONS

`-h` ***`help`*** help about the command.

ENVIRONMENT

`VISHNU_CONFIG_FILE` Contains the path to the local configuration file for VISHNU.

DIAGNOSTICS

The following diagnostics may be issued on stderr and the command will return the code provided within brackets:

"There is no open session in this terminal" [10]

"The session key is unrecognized" [28]

"The sessionKey is expired. The session is closed." [29]

"Commands are running" [31]

3.4 vishnu_change_password

vishnu_change_password — changes the password

Synopsis

`vishnu_change_password [-h]` *`userId`*

DESCRIPTION

This command is used to change the password. It can be done voluntarily or when the password is only temporary: for your first connection to VISHNU or after your password is reset by an administrator .

OPTIONS

-h ***help*** help about the command.

ENVIRONMENT

VISHNU_CONFIG_FILE Contains the path to the local configuration file for VISHNU.

DIAGNOSTICS

The following diagnostics may be issued on stderr and the command will return the code provided within brackets:

"Unknown user" [20]

"The user is locked" [23]

3.5 vishnu_add_local_account

vishnu_add_local_account — adds a new local user configuration

Synopsis

`vishnu_add_local_account` [-h] *userId machineId acLogin sshKeyPath homeDirectory*

DESCRIPTION

A local user configuration must be added to allow you (identified by userId) to connect to machine (identified by machineId). This configuration must match an existing system account on that machine with a login that matches acLogin. The parameters sshKeyPath parameter (the absolute path to your private SSH key, used for file transfers) must be provided as well as the homeDirectory path .

OPTIONS

-h ***help*** help about the command.

ENVIRONMENT

VISHNU_CONFIG_FILE Contains the path to the local configuration file for VISHNU.

DIAGNOSTICS

The following diagnostics may be issued on stderr and the command will return the code provided within brackets:

"There is no open session in this terminal" [10]

"The userId is unknown" [21]

"The session key is unrecognized" [28]

"The sessionKey is expired. The session is closed." [29]

"The machine id is unknown" [32]

"The machine is locked" [34]

"The local account already exists" [37]

3.6 vishnu_update_local_account

vishnu_update_local_account — updates a local user configuration

Synopsis

`vishnu_update_local_account` [`-h`] [`-l` *`acLogin`*] [`-s` *`sshKeyPath`*] [`-d` *`homeDirectory`*] *`userId`* *`machineId`*

DESCRIPTION

The local user configuration can be updated. You can modify information of its local configuration such as acLogin, sshKeypath or homeDirectory.

OPTIONS

`-h` ***`help`*** help about the command.

`-l` ***`acLogin`*** accLogin represents the login of the user on the associated machine.

`-s` ***`sshKeyPath`*** sshKeyPath is the path of the ssh key of the user on the associated machine.

`-d` ***`homeDirectory`*** HomeDirectory is the path of the home directory of the user on the associated machine.

ENVIRONMENT

`VISHNU_CONFIG_FILE` Contains the path to the local configuration file for VISHNU.

DIAGNOSTICS

The following diagnostics may be issued on stderr and the command will return the code provided within brackets:

"There is no open session in this terminal" [10]

"The userId is unknown" [21]

"The session key is unrecognized" [28]

"The sessionKey is expired. The session is closed." [29]

"The machine id is unknown" [32]

"The local is unknown" [38]

3.7 vishnu_delete_local_account

vishnu_delete_local_account — removes a local user configuration (for a given user on a given machine) from VISHNU

Synopsis

`vishnu_delete_local_account` [-h] *userId* *machineId*

DESCRIPTION

The local user configuration can be deleted from VISHNU. When a local user configuration is deleted, all the information about it is deleted from VISHNU.

OPTIONS

-h *help* help about the command.

ENVIRONMENT

VISHNU_CONFIG_FILE Contains the path to the local configuration file for VISHNU.

DIAGNOSTICS

The following diagnostics may be issued on stderr and the command will return the code provided within brackets:

"There is no open session in this terminal" [10]

"The session key is unrecognized" [28]

"The sessionKey is expired. The session is closed." [29]

"The local is unknown" [38]

3.8 vishnu_list_local_account

vishnu_list_local_account — lists the local user configurations

Synopsis

`vishnu_list_local_account` [-h] [-a] [-u *userId*] [-i *machineId*]

DESCRIPTION

A local configuration is used to configure the access to a given system for a given user through VISHNU. It is related to an account on that system that is identified using its login. This command allows you to check all the local configurations related to your VISHNU account.

OPTIONS

-h *help* help about the command.

-a *adminListOption* is an admin option for listing all local configurations of all users .

-u *userId* is an admin option for listing the local configurations of a specific user .

-i *machineId* is an option for listing local user configurations on a specific machine.

ENVIRONMENT

VISHNU_CONFIG_FILE Contains the path to the local configuration file for VISHNU.

DIAGNOSTICS

The following diagnostics may be issued on stderr and the command will return the code provided within brackets:

"There is no open session in this terminal" [10]

"The userId is unknown" [21]

"The session key is unrecognized" [28]

"The sessionKey is expired. The session is closed." [29]

3.9 vishnu_list_machine

vishnu_list_machine — lists the machines that are accessible through VISHNU

Synopsis

`vishnu_list_machine [-h] [-u userId] [-a] [-m machineId]`

DESCRIPTION

This command is used to display the machines that you can use for VISHNU services. The machines you can access through VISHNU are those that are configured in VISHNU by the VISHNU administrator, and that have been added to your personal VISHNU configuration using the vishnu_add_local_account command. The results contain, for each machine, a machine identifier that you can use as a parameter for other VISHNU commands.

OPTIONS

-h *help* help about the command.

-u *userId* is an admin option for listing machines in which a specific user has a local configuration.

-a *listAllmachine* is an option for listing all VISHNU machines.

-m *machineId* is an option for listing information about a specific machine.

ENVIRONMENT

`VISHNU_CONFIG_FILE` Contains the path to the local configuration file for VISHNU.

DIAGNOSTICS

The following diagnostics may be issued on stderr and the command will return the code provided within brackets:

"There is no open session in this terminal" [10]

"The userId is unknown" [21]

"The session key is unrecognized" [28]

"The sessionKey is expired. The session is closed." [29]

3.10 vishnu_list_history_cmd

vishnu_list_history_cmd — lists the commands

Synopsis

`vishnu_list_history_cmd` [-h] [-a] [-u *userId*] [-i *sessionId*] [-s *startDateOption*] [-e *endDateOption*]

DESCRIPTION

This command displays a history of the commands you ran. Several options can be used to specify which commands to list.

OPTIONS

-h ***help*** help about the command.

-a ***adminListOption*** is an admin option for listing all commands of all users.

-u ***userId*** is an admin option for listing commands launched by a specific user identified by his/her userId.

-i ***sessionId*** lists all commands launched within a specific session.

-s ***startDateOption*** allows the user to organize the commands listed by providing the start date (the UNIX timestamp of the start date is used).

-e ***endDateOption*** allows the user to organize the commands listed by providing the end date (the timestamp of the end date is used). By default, the end date is the current day.

ENVIRONMENT

`VISHNU_CONFIG_FILE` Contains the path to the local configuration file for VISHNU.

DIAGNOSTICS

The following diagnostics may be issued on stderr and the command will return the code provided within brackets:

"There is no open session in this terminal" [10]

"The userId is unknown" [21]

"The session key is unrecognized" [28]

"The sessionKey is expired. The session is closed." [29]

3.11 vishnu_list_options

vishnu_list_options — lists the options of the user

Synopsis

`vishnu_list_options` [`-h`] [`-a`] [`-u` *`userId`*] [`-n` *`optionName`*]

DESCRIPTION

This command displays the options you configured.

OPTIONS

`-h` ***`help`*** help about the command.

`-a` ***`listAllDeftValue`*** is an option for listing all default option values defined by VISHNU administrator.

`-u` ***`userId`*** is an admin option for listing the options of a specific user.

`-n` ***`optionName`*** allows the user to get the value of a specific option identified by its name .

ENVIRONMENT

`VISHNU_CONFIG_FILE` Contains the path to the local configuration file for VISHNU.

DIAGNOSTICS

The following diagnostics may be issued on stderr and the command will return the code provided within brackets:

"There is no open session in this terminal" [10]

"The userId is unknown" [21]

"The user is not an administrator" [25]

"The session key is unrecognized" [28]

"The sessionKey is expired. The session is closed." [29]

"The name of the user option is unknown" [41]

3.12 vishnu_list_sessions

vishnu_list_sessions — lists all sessions of the user

Synopsis

`vishnu_list_sessions` [-h] [-t *status*] [-p *sessionClosePolicy*] [-d *sessionInactivityDelay*] [-m *machineId*] [-a] [-u *userId*] [-i *sessionId*] [-s *startDateOption*] [-e *endDateOption*]

DESCRIPTION

This command is used to display and filter the list of all your sessions. For each session, a session identifier is provided which you can use to reconnect to a given session using the vishnu_reconnect command. The session's status is either 0 (inactive) or 1 (active).

OPTIONS

-h *help* help about the command.

-t *status* specifies the status of the sessions which will be listed. The value must be an integer. Predefined values are: 0 (INACTIVE), 1 (ACTIVE).

-p *sessionClosePolicy* specifies the closure mode of the sessions which will be listed (CLOSE_ON_TIMEOUT or CLOSE_ON_D
The value must be an integer. Predefined values are: 0 (UNDEFINED), 1 (CLOSE_ON_TIMEOUT), 2 (CLOSE_ON_DISCONNE

-d *sessionInactivityDelay* specifies the inactivity delay in seconds of the sessions which will be listed.

-m *machineId* allows the user to list sessions opened on a specific machine.

-a *adminListOption* is an admin option for listing all sessions of all users.

-u *userId* is an admin option for listing sessions opened by a specific user.

-i *sessionId* allows the user to list all commands launched within a specific session.

-s *startDateOption* allows the user to organize the commands listed by providing the start date (the UNIX timestamp of the start date is used).

-e *endDateOption* allows the user to organize the commands listed by providing the end date (the timestamp of the end date is used). By default, the end date is the current day.

ENVIRONMENT

VISHNU_CONFIG_FILE Contains the path to the local configuration file for VISHNU.

DIAGNOSTICS

The following diagnostics may be issued on stderr and the command will return the code provided within brackets:

"There is no open session in this terminal" [10]

"The userId is unknown" [21]

"The session key is unrecognized" [28]

"The sessionKey is expired. The session is closed." [29]

3.13 vishnu_configure_option

vishnu_configure_option — configures an option of the user

Synopsis

`vishnu_configure_option` [-h] *`optionName`* *`value`*

DESCRIPTION

Options in VISHNU corresponds to the parameters of some VISHNU commands (e.g., the close policy for vishnu_connect) that can be preset in the user configuration stored by the VISHNU system. This command is used to set the value of an option for the current user (the user who opened the session).

OPTIONS

`-h` ***`help`*** help about the command.

ENVIRONMENT

`VISHNU_CONFIG_FILE` Contains the path to the local configuration file for VISHNU.

DIAGNOSTICS

The following diagnostics may be issued on stderr and the command will return the code provided within brackets:

"There is no open session in this terminal" [10]

"The session key is unrecognized" [28]

"The sessionKey is expired. The session is closed." [29]

"The name of the user option is unknown" [41]

Chapter 4

C++ API Reference

4.1 connect

connect — opens a session

Synopsis

int **vishnu::connect**(const string& userId, const string& password, Session& session, const ConnectOptions& options = ConnectOptions());

DESCRIPTION

Opening a VISHNU session is the first step before using any other VISHNU command. This command authenticates you. You must have been registered in the VISHNU system by an administrator. It also creates a session that remains open after the command is completed and until the session is either manually or automatically closed.

ARGUMENTS

userId Input argument. UserId represents the VISHNU user identifier.

password Input argument. Password represents the password of the user.

session Output argument. The session object that contains the created session details.

options Input argument. Options is an object which encapsulates the options available for the connect method. It allows the user to choose the way for closing the session automatically on TIMEOUT or on DISCONNECT and the possibility for an admin to open a session as he/she was a specific user.

EXCEPTIONS

The following exceptions may be thrown:

"Unknown user" [20]

"The userId is unknown" [21]

"The user is locked" [23]

"The user is not an administrator" [25]

"The value of the timeout is incorrect" [43]

4.2 reconnect

reconnect — reconnects to a session that is still active

Synopsis

int **vishnu::reconnect**(const string& userId , const string& password, const string& sessionId, Session& session);

DESCRIPTION

This command allows you to resume a session that has been opened previously and that has not yet been closed. You can disconnect from a session without closing it (for example if there are running commands in that session) by setting the session's close policy (at connection time) to CLOSE_ON_TIMEOUT. As sessions are linked to a specific client system, you cannot reconnect to a session that was opened on another client system.

ARGUMENTS

userId Input argument. UserId represents the VISHNU user identifier.

password Input argument. Password represents the password of the user.

sessionId Input argument. SessionId is the identifier of the session defined in the database.

session Output argument. The session object containing session information.

EXCEPTIONS

The following exceptions may be thrown:

"Unknown user" [20]

"The user is locked" [23]

"The session key is unrecognized" [28]

"The sessionKey is expired. The session is closed." [29]

"The session Id is unknown" [30]

"The machine does not exist or it is locked" [36]

4.3 close

close — closes the session

Synopsis

int **vishnu::close**(const string& sessionKey);

DESCRIPTION

This command closes the session that is currently active in the terminal. It will return an error if there are still some active requests that were been submitted by the user during the session (e.g., job submission or file transfers). After the session is closed, it cannot be re-opened.

ARGUMENTS

sessionKey Input argument. The sessionKey is the encrypted identifier of the session generated by VISHNU.

EXCEPTIONS

The following exceptions may be thrown:

"The session key is unrecognized" [28]

"The sessionKey is expired. The session is closed." [29]

"Commands are running" [31]

4.4 changePassword

changePassword — changes the password

Synopsis

int **vishnu::changePassword**(const string& userId, const string& password, const string& passwordNew);

DESCRIPTION

This command is used to change the password. It can be done voluntarily or when the password is only temporary: for your first connection to VISHNU or after your password is reset by an administrator .

ARGUMENTS

userId Input argument. UserId represents the VISHNU user identifier.

password Input argument. Password represents the password of the user.

passwordNew Input argument. PasswordNew represents the new password of the user.

EXCEPTIONS

The following exceptions may be thrown:

"Unknown user" [20]

"The user is locked" [23]

4.5 addLocalAccount

addLocalAccount — adds a new local user configuration

Synopsis

int **vishnu::addLocalAccount**(const string& sessionKey, const LocalAccount& newAccount, string& sshPublicKey);

DESCRIPTION

A local user configuration must be added to allow you (identified by userId) to connect to machine (identified by machineId). This configuration must match an existing system account on that machine with a login that matches acLogin. The parameters sshKeyPath parameter (the absolute path to your private SSH key, used for file transfers) must be provided as well as the homeDirectory path .

ARGUMENTS

sessionKey Input argument. The sessionKey is the encrypted identifier of the session generated by VISHNU.

newAccount Input argument. NewAccount is the object which encapsulates the new local user configuration.

sshPublicKey Output argument. The SSH public key generated by VISHNU for accessing a local account.

EXCEPTIONS

The following exceptions may be thrown:

"The userId is unknown" [21]

"The session key is unrecognized" [28]

"The sessionKey is expired. The session is closed." [29]

"The machine id is unknown" [32]

"The machine is locked" [34]

"The local account already exists" [37]

4.6 updateLocalAccount

updateLocalAccount — updates a local user configuration

Synopsis

int **vishnu::updateLocalAccount**(const string& sessionKey, const LocalAccount& LocalAccUpd);

DESCRIPTION

The local user configuration can be updated. You can modify information of its local configuration such as acLogin, sshKeypath or homeDirectory.

ARGUMENTS

sessionKey Input argument. The sessionKey is the encrypted identifier of the session generated by VISHNU.

LocalAccUpd Input argument. Is an object which encapsulates the local user configuration changes except the machineId and the userId.

EXCEPTIONS

The following exceptions may be thrown:

"The userId is unknown" [21]

"The session key is unrecognized" [28]

"The sessionKey is expired. The session is closed." [29]

"The machine id is unknown" [32]

"The local is unknown" [38]

4.7 deleteLocalAccount

deleteLocalAccount — removes a local user configuration (for a given user on a given machine) from VISHNU

Synopsis

int **vishnu::deleteLocalAccount**(const string& sessionKey, const string& userId, const string& machineId);

DESCRIPTION

The local user configuration can be deleted from VISHNU. When a local user configuration is deleted, all the information about it is deleted from VISHNU.

ARGUMENTS

sessionKey Input argument. The sessionKey is the encrypted identifier of the session generated by VISHNU.

userId Input argument. UserId represents the VISHNU user identifier of the user whose local configuration will be deleted for the given machine .

machineId Input argument. MachineId represents the identifier of the machine whose local configuration will be deleted for the given user .

EXCEPTIONS

The following exceptions may be thrown:

"The session key is unrecognized" [28]

"The sessionKey is expired. The session is closed." [29]

"The local is unknown" [38]

4.8 listLocalAccount

listLocalAccount — lists the local user configurations

Synopsis

int **vishnu::listLocalAccount**(const string& sessionKey, ListLocalAccounts& listLocalAcct, const ListLocalAccOptions& options = ListLocalAccOptions());

DESCRIPTION

A local configuration is used to configure the access to a given system for a given user through VISHNU. It is related to an account on that system that is identified using its login. This command allows you to check all the local configurations related to your VISHNU account.

ARGUMENTS

sessionKey Input argument. The sessionKey is the encrypted identifier of the session generated by VISHNU.

listLocalAcct Output argument. ListLocalAccount is the list of the local user configuations .

options Input argument. Allows an admin to list all local configurations of all users or a simple user to list his/her local user configurations on a specific machine.

EXCEPTIONS

The following exceptions may be thrown:

"The userId is unknown" [21]

"The session key is unrecognized" [28]

"The sessionKey is expired. The session is closed." [29]

4.9 listMachine

listMachine — lists the machines that are accessible through VISHNU

Synopsis

int **vishnu::listMachine**(const string& sessionKey, ListMachines& listMachine, const ListMachineOptions& options = ListMachineOptions());

DESCRIPTION

This command is used to display the machines that you can use for VISHNU services. The machines you can access through VISHNU are those that are configured in VISHNU by the VISHNU administrator, and that have been added to your personal VISHNU configuration using the vishnu_add_local_account command. The results contain, for each machine, a machine identifier that you can use as a parameter for other VISHNU commands.

ARGUMENTS

sessionKey Input argument. The sessionKey is the encrypted identifier of the session generated by VISHNU.

listMachine Output argument. ListLocalAccount is the list of the local configs .

options Input argument. Allows a user to list all VISHNU machines or information about a specific machine and an admin to list machines used by a specific user.

EXCEPTIONS

The following exceptions may be thrown:

"The userId is unknown" [21]

"The session key is unrecognized" [28]

"The sessionKey is expired. The session is closed." [29]

4.10 listHistoryCmd

listHistoryCmd — lists the commands

Synopsis

int **vishnu::listHistoryCmd**(const string& sessionKey, ListCommands& listCommands, const ListCmdOptions& options = ListCmdOptions());

DESCRIPTION

This command displays a history of the commands you ran. Several options can be used to specify which commands to list.

ARGUMENTS

sessionKey Input argument. The sessionKey is the encrypted identifier of the session generated by VISHNU.

listCommands Output argument. ListCommands is the list of commands.

options Input argument. Allows the user to list commands by using several optional criteria: a period, specific session and for admin to list all commands of all VISHNU users or commands from a specific user.

EXCEPTIONS

The following exceptions may be thrown:

"The userId is unknown" [21]

"The session key is unrecognized" [28]

"The sessionKey is expired. The session is closed." [29]

4.11 listOptions

listOptions — lists the options of the user

Synopsis

int **vishnu::listOptions**(const string& sessionKey, ListOptionsValues& listOptValues, const ListOptOptions& options = ListOptOptions());

DESCRIPTION

This command displays the options you configured.

ARGUMENTS

sessionKey Input argument. The sessionKey is the identifier of the session generated by VISHNU.

listOptValues Output argument. ListOptValues is an object which encapsulates the list of options.

options Input argument. Allows to list a specific option or all default options values or for an admin to list options of a specific user.

EXCEPTIONS

The following exceptions may be thrown:

"The userId is unknown" [21]

"The user is not an administrator" [25]

"The session key is unrecognized" [28]

"The sessionKey is expired. The session is closed." [29]

"The name of the user option is unknown" [41]

4.12 listSessions

listSessions — lists all sessions of the user

Synopsis

int **vishnu::listSessions**(const string& sessionKey, ListSessions& listsession, const ListSessionOptions& options = ListSessionOptions());

DESCRIPTION

This command is used to display and filter the list of all your sessions. For each session, a session identifier is provided which you can use to reconnect to a given session using the vishnu_reconnect command. The session's status is either 0 (inactive) or 1 (active).

ARGUMENTS

sessionKey Input argument. The sessionKey is the encrypted identifier of the session generated by VISHNU.

listsession Output argument. Listsession is the list of sessions .

options Input argument. Allows the user to list sessions using several optional criteria such as: the state of sessions (actives or inactives, by default, all sessions are listed), a period, a specific session or for admin to list all sessions of all users or sessions of a specific user..

EXCEPTIONS

The following exceptions may be thrown:

"The userId is unknown" [21]

"The session key is unrecognized" [28]

"The sessionKey is expired. The session is closed." [29]

4.13 configureOption

configureOption — configures an option of the user

Synopsis

int **vishnu::configureOption**(const string& sessionKey, const OptionValue& optionValue);

DESCRIPTION

Options in VISHNU corresponds to the parameters of some VISHNU commands (e.g., the close policy for vishnu_connect) that can be preset in the user configuration stored by the VISHNU system. This command is used to set the value of an option for the current user (the user who opened the session).

ARGUMENTS

sessionKey Input argument. The sessionKey is the encrypted identifier of the session generated by VISHNU.

optionValue Input argument. The optionValue is an object which encapsulates the option information.

EXCEPTIONS

The following exceptions may be thrown:

"The session key is unrecognized" [28]

"The sessionKey is expired. The session is closed." [29]

"The name of the user option is unknown" [41]

4.14 vishnuInitialize

vishnuInitialize — initializes VISHNU

Synopsis

int **vishnu::vishnuInitialize**(const string& configPath);

DESCRIPTION

Calling this function is required before calling any function of the VISHNU API. It initializes the connection to the VISHNU infrastructure.

ARGUMENTS

configPath Input argument. ConfigPath is the path of VISHNU configuration file.

EXCEPTIONS

The following exceptions may be thrown:

"Internal Error: SysDS service failure" [1]

4.15 vishnuFinalize

vishnuFinalize — allows a user to go out properly from VISHNU

Synopsis

int **vishnu::vishnuFinalize**();

DESCRIPTION

Calling this function is necessary to free ressources consumed due to the VISHNU API

EXCEPTIONS

The following exceptions may be thrown:

"Internal Error: SysDS service failure" [1]

Chapter 5

Python API Reference

5.1 VISHNU_UMS.connect

VISHNU_UMS.connect — opens a session

Synopsis

VISHNU_UMS.connect(string userId, string password, Session session, ConnectOptions options = ConnectOptions());

DESCRIPTION

Opening a VISHNU session is the first step before using any other VISHNU command. This command authenticates you. You must have been registered in the VISHNU system by an administrator. It also creates a session that remains open after the command is completed and until the session is either manually or automatically closed.

ARGUMENTS

userId Input argument. UserId represents the VISHNU user identifier.

password Input argument. Password represents the password of the user.

session Output argument. The session object that contains the created session details.

options Input argument. Options is an object which encapsulates the options available for the connect method. It allows the user to choose the way for closing the session automatically on TIMEOUT or on DISCONNECT and the possibility for an admin to open a session as he/she was a specific user.

RETURNED OBJECTS

errorCode (integer) Output parameter. Contains 0 on success and the error code on failure.

sessionKey (string) Output parameter. Contains the session key.

EXCEPTIONS

The following exceptions may be thrown:

UMSVishnuException("Unknown user" [20])

UMSVishnuException("The userId is unknown" [21])

UMSVishnuException("The user is locked" [23])

UMSVishnuException("The user is not an administrator" [25])

UMSVishnuException("The value of the timeout is incorrect" [43])

5.2 VISHNU_UMS.reconnect

VISHNU_UMS.reconnect — reconnects to a session that is still active

Synopsis

VISHNU_UMS.reconnect(string userId , string password, string sessionId, Session session);

DESCRIPTION

This command allows you to resume a session that has been opened previously and that has not yet been closed. You can disconnect from a session without closing it (for example if there are running commands in that session) by setting the session's close policy (at connection time) to CLOSE_ON_TIMEOUT. As sessions are linked to a specific client system, you cannot reconnect to a session that was opened on another client system.

ARGUMENTS

userId Input argument. UserId represents the VISHNU user identifier.

password Input argument. Password represents the password of the user.

sessionId Input argument. SessionId is the identifier of the session defined in the database.

session Output argument. The session object containing session information.

RETURNED OBJECTS

errorCode (integer) Output parameter. Contains 0 on success and the error code on failure.

sessionKey (string) Output parameter. Contains the session key.

EXCEPTIONS

The following exceptions may be thrown:

UMSVishnuException("Unknown user" [20])

UMSVishnuException("The user is locked" [23])

UMSVishnuException("The session key is unrecognized" [28])

UMSVishnuException("The sessionKey is expired. The session is closed." [29])

UMSVishnuException("The session Id is unknown" [30])

UMSVishnuException("The machine does not exist or it is locked" [36])

5.3 VISHNU_UMS.close

VISHNU_UMS.close — closes the session

Synopsis

VISHNU_UMS.close(string sessionKey);

DESCRIPTION

This command closes the session that is currently active in the terminal. It will return an error if there are still some active requests that were been submitted by the user during the session (e.g., job submission or file transfers). After the session is closed, it cannot be re-opened.

ARGUMENTS

sessionKey Input argument. The sessionKey is the encrypted identifier of the session generated by VISHNU.

RETURNED OBJECTS

errorCode (integer) Output parameter. Contains 0 on success and the error code on failure.

EXCEPTIONS

The following exceptions may be thrown:

UMSVishnuException("The session key is unrecognized" [28])

UMSVishnuException("The sessionKey is expired. The session is closed." [29])

UMSVishnuException("Commands are running" [31])

5.4 VISHNU_UMS.changePassword

VISHNU_UMS.changePassword — changes the password

Synopsis

VISHNU_UMS.changePassword(string userId, string password, string passwordNew);

DESCRIPTION

This command is used to change the password. It can be done voluntarily or when the password is only temporary: for your first connection to VISHNU or after your password is reset by an administrator .

ARGUMENTS

userId Input argument. UserId represents the VISHNU user identifier.

password Input argument. Password represents the password of the user.

passwordNew Input argument. PasswordNew represents the new password of the user.

RETURNED OBJECTS

errorCode (integer) Output parameter. Contains 0 on success and the error code on failure.

EXCEPTIONS

The following exceptions may be thrown:

UMSVishnuException("Unknown user" [20])

UMSVishnuException("The user is locked" [23])

5.5 VISHNU_UMS.addLocalAccount

VISHNU_UMS.addLocalAccount — adds a new local user configuration

Synopsis

VISHNU_UMS.addLocalAccount(string sessionKey, LocalAccount newAccount, string sshPublicKey);

DESCRIPTION

A local user configuration must be added to allow you (identified by userId) to connect to machine (identified by machineId). This configuration must match an existing system account on that machine with a login that matches acLogin. The parameters sshKeyPath parameter (the absolute path to your private SSH key, used for file transfers) must be provided as well as the homeDirectory path .

ARGUMENTS

sessionKey Input argument. The sessionKey is the encrypted identifier of the session generated by VISHNU.

newAccount Input argument. NewAccount is the object which encapsulates the new local user configuration.

sshPublicKey Output argument. The SSH public key generated by VISHNU for accessing a local account.

RETURNED OBJECTS

errorCode (integer) Output parameter. Contains 0 on success and the error code on failure.

EXCEPTIONS

The following exceptions may be thrown:

UMSVishnuException("The userId is unknown" [21])

UMSVishnuException("The session key is unrecognized" [28])

UMSVishnuException("The sessionKey is expired. The session is closed." [29])

UMSVishnuException("The machine id is unknown" [32])

UMSVishnuException("The machine is locked" [34])

UMSVishnuException("The local account already exists" [37])

5.6 VISHNU_UMS.updateLocalAccount

VISHNU_UMS.updateLocalAccount — updates a local user configuration

Synopsis

VISHNU_UMS.updateLocalAccount(string sessionKey, LocalAccount LocalAccUpd);

DESCRIPTION

The local user configuration can be updated. You can modify information of its local configuration such as acLogin, sshKeypath or homeDirectory.

ARGUMENTS

sessionKey Input argument. The sessionKey is the encrypted identifier of the session generated by VISHNU.

LocalAccUpd Input argument. Is an object which encapsulates the local user configuration changes except the machineId and the userId.

RETURNED OBJECTS

errorCode (integer) Output parameter. Contains 0 on success and the error code on failure.

EXCEPTIONS

The following exceptions may be thrown:

UMSVishnuException("The userId is unknown" [21])

UMSVishnuException("The session key is unrecognized" [28])

UMSVishnuException("The sessionKey is expired. The session is closed." [29])

UMSVishnuException("The machine id is unknown" [32])

UMSVishnuException("The local is unknown" [38])

5.7 VISHNU_UMS.deleteLocalAccount

VISHNU_UMS.deleteLocalAccount — removes a local user configuration (for a given user on a given machine) from VISHNU

Synopsis

VISHNU_UMS.deleteLocalAccount(string sessionKey, string userId, string machineId);

DESCRIPTION

The local user configuration can be deleted from VISHNU. When a local user configuration is deleted, all the information about it is deleted from VISHNU.

ARGUMENTS

sessionKey Input argument. The sessionKey is the encrypted identifier of the session generated by VISHNU.

userId Input argument. UserId represents the VISHNU user identifier of the user whose local configuration will be deleted for the given machine .

machineId Input argument. MachineId represents the identifier of the machine whose local configuration will be deleted for the given user .

RETURNED OBJECTS

errorCode (integer) Output parameter. Contains 0 on success and the error code on failure.

EXCEPTIONS

The following exceptions may be thrown:

UMSVishnuException("The session key is unrecognized" [28])

UMSVishnuException("The sessionKey is expired. The session is closed." [29])

UMSVishnuException("The local is unknown" [38])

5.8 VISHNU_UMS.listLocalAccount

VISHNU_UMS.listLocalAccount — lists the local user configurations

Synopsis

VISHNU_UMS.listLocalAccount(string sessionKey, ListLocalAccounts listLocalAcct, ListLocalAccOptions options = ListLocalAccOptions());

DESCRIPTION

A local configuration is used to configure the access to a given system for a given user through VISHNU. It is related to an account on that system that is identified using its login. This command allows you to check all the local configurations related to your VISHNU account.

ARGUMENTS

sessionKey Input argument. The sessionKey is the encrypted identifier of the session generated by VISHNU.

listLocalAcct Output argument. ListLocalAccount is the list of the local user configuations .

options Input argument. Allows an admin to list all local configurations of all users or a simple user to list his/her local user configurations on a specific machine.

RETURNED OBJECTS

errorCode (integer) Output parameter. Contains 0 on success and the error code on failure.

EXCEPTIONS

The following exceptions may be thrown:

UMSVishnuException("The userId is unknown" [21])

UMSVishnuException("The session key is unrecognized" [28])

UMSVishnuException("The sessionKey is expired. The session is closed." [29])

5.9 VISHNU_UMS.listMachine

VISHNU_UMS.listMachine — lists the machines that are accessible through VISHNU

Synopsis

VISHNU_UMS.listMachine(string sessionKey, ListMachines listMachine, ListMachineOptions options = ListMachineOptions());

DESCRIPTION

This command is used to display the machines that you can use for VISHNU services. The machines you can access through VISHNU are those that are configured in VISHNU by the VISHNU administrator, and that have been added to your personal VISHNU configuration using the vishnu_add_local_account command. The results contain, for each machine, a machine identifier that you can use as a parameter for other VISHNU commands.

ARGUMENTS

sessionKey Input argument. The sessionKey is the encrypted identifier of the session generated by VISHNU.

listMachine Output argument. ListLocalAccount is the list of the local configs .

options Input argument. Allows a user to list all VISHNU machines or information about a specific machine and an admin to list machines used by a specific user.

RETURNED OBJECTS

errorCode (integer) Output parameter. Contains 0 on success and the error code on failure.

EXCEPTIONS

The following exceptions may be thrown:

UMSVishnuException("The userId is unknown" [21])

UMSVishnuException("The session key is unrecognized" [28])

UMSVishnuException("The sessionKey is expired. The session is closed." [29])

5.10 VISHNU_UMS.listHistoryCmd

VISHNU_UMS.listHistoryCmd — lists the commands

Synopsis

VISHNU_UMS.listHistoryCmd(string sessionKey, ListCommands listCommands, ListCmdOptions options = ListCmdOptions());

DESCRIPTION

This command displays a history of the commands you ran. Several options can be used to specify which commands to list.

ARGUMENTS

sessionKey Input argument. The sessionKey is the encrypted identifier of the session generated by VISHNU.

listCommands Output argument. ListCommands is the list of commands.

options Input argument. Allows the user to list commands by using several optional criteria: a period, specific session and for admin to list all commands of all VISHNU users or commands from a specific user.

RETURNED OBJECTS

errorCode (integer) Output parameter. Contains 0 on success and the error code on failure.

EXCEPTIONS

The following exceptions may be thrown:

UMSVishnuException("The userId is unknown" [21])

UMSVishnuException("The session key is unrecognized" [28])

UMSVishnuException("The sessionKey is expired. The session is closed." [29])

5.11 VISHNU_UMS.listOptions

VISHNU_UMS.listOptions — lists the options of the user

Synopsis

VISHNU_UMS.listOptions(string sessionKey, ListOptionsValues listOptValues, ListOptOptions options = ListOptOptions());

DESCRIPTION

This command displays the options you configured.

ARGUMENTS

sessionKey Input argument. The sessionKey is the identifier of the session generated by VISHNU.

listOptValues Output argument. ListOptValues is an object which encapsulates the list of options.

options Input argument. Allows to list a specific option or all default options values or for an admin to list options of a specific user.

RETURNED OBJECTS

errorCode (integer) Output parameter. Contains 0 on success and the error code on failure.

EXCEPTIONS

The following exceptions may be thrown:

UMSVishnuException("The userId is unknown" [21])

UMSVishnuException("The user is not an administrator" [25])

UMSVishnuException("The session key is unrecognized" [28])

UMSVishnuException("The sessionKey is expired. The session is closed." [29])

UMSVishnuException("The name of the user option is unknown" [41])

5.12 VISHNU_UMS.listSessions

VISHNU_UMS.listSessions — lists all sessions of the user

Synopsis

VISHNU_UMS.listSessions(string sessionKey, ListSessions listsession, ListSessionOptions options = ListSessionOptions());

DESCRIPTION

This command is used to display and filter the list of all your sessions. For each session, a session identifier is provided which you can use to reconnect to a given session using the vishnu_reconnect command. The session's status is either 0 (inactive) or 1 (active).

ARGUMENTS

sessionKey Input argument. The sessionKey is the encrypted identifier of the session generated by VISHNU.

listsession Output argument. Listsession is the list of sessions .

options Input argument. Allows the user to list sessions using several optional criteria such as: the state of sessions (actives or inactives, by default, all sessions are listed), a period, a specific session or for admin to list all sessions of all users or sessions of a specific user..

RETURNED OBJECTS

errorCode (integer) Output parameter. Contains 0 on success and the error code on failure.

EXCEPTIONS

The following exceptions may be thrown:

UMSVishnuException("The userId is unknown" [21])

UMSVishnuException("The session key is unrecognized" [28])

UMSVishnuException("The sessionKey is expired. The session is closed." [29])

5.13 VISHNU_UMS.configureOption

VISHNU_UMS.configureOption — configures an option of the user

Synopsis

VISHNU_UMS.configureOption(string sessionKey, OptionValue optionValue);

DESCRIPTION

Options in VISHNU corresponds to the parameters of some VISHNU commands (e.g., the close policy for vishnu_connect) that can be preset in the user configuration stored by the VISHNU system. This command is used to set the value of an option for the current user (the user who opened the session).

ARGUMENTS

sessionKey Input argument. The sessionKey is the encrypted identifier of the session generated by VISHNU.

optionValue Input argument. The optionValue is an object which encapsulates the option information.

RETURNED OBJECTS

errorCode (integer) Output parameter. Contains 0 on success and the error code on failure.

EXCEPTIONS

The following exceptions may be thrown:

UMSVishnuException("The session key is unrecognized" [28])

UMSVishnuException("The sessionKey is expired. The session is closed." [29])

UMSVishnuException("The name of the user option is unknown" [41])

5.14 VISHNU_UMS.vishnuInitialize

VISHNU_UMS.vishnuInitialize — initializes VISHNU

Synopsis

VISHNU_UMS.vishnuInitialize(string configPath);

DESCRIPTION

Calling this function is required before calling any function of the VISHNU API. It initializes the connection to the VISHNU infrastructure.

ARGUMENTS

configPath Input argument. ConfigPath is the path of VISHNU configuration file.

RETURNED OBJECTS

errorCode (integer) Output parameter. Contains 0 on success and the error code on failure.

EXCEPTIONS

The following exceptions may be thrown:

SystemException("Internal Error: SysDS service failure" [1])

5.15 VISHNU_UMS.vishnuFinalize

VISHNU_UMS.vishnuFinalize — allows a user to go out properly from VISHNU

Synopsis

VISHNU_UMS.vishnuFinalize();

DESCRIPTION

Calling this function is necessary to free ressources consumed due to the VISHNU API

RETURNED OBJECTS

errorCode (integer) Output parameter. Contains 0 on success and the error code on failure.

EXCEPTIONS

The following exceptions may be thrown:

SystemException("Internal Error: SysDS service failure" [1])