VISHNU - Le guide de l'administrateur

COLLABORATORS			

	TITLE : VISHNU - Le guide de l'administrateur		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Benjamin Isnard, Daouda Traoré, Eugène Pamba Capo-Chichi, Kevin Coulomb, and Ibrahima Cissé	March 8, 2011	

REVISION HISTORY			

NUMBER	DATE	DESCRIPTION	NAME
1	08/03/2011	Version initiale pour le module UMS uniquement	K. COULOMB

Contents

Chapter 1

Présentation du document

1.1 Objectifs du document

Ce document présente l'administration de la plateforme VISHNU.

1.2 Structure du document

Ce document est divisé en quatre parties :

- Définitions
- L'installation
- Déploiement
- L'administration

Chapter 2

Définitions

2.1 Acronymes

- DB : Base de données. Elle est centralisée et utilisée pour sauvegarder toutes les données manipulées par le module UMS. Cette base de données sera considérée comme étant de type PostgreSQL dans ce document.

- MA : Le 'Master Agent', élément de Sysfera-DS distribuant les requêtes entre les clients et les services désirés.

- SQL : Langage de requêtes sur les bases de données

2.2 Références

- [ARCH] D1.1g-VISHNU Technical Architecture : description de l'architecture de l'application VISHNU

- [DIET_USERGUIDE] DIET User Guide : guide de l'utilisateur DIET (nom du projet OpenSource de l'application SysFera-DS

- [VISHNU_USERGUIDE] VISHNU User Guide: guide de l'utilisateur VISHNU.

2.3 Glossaire

- Client UMS : Cela désigne les classes intermédiaires permettant un accès à distance aux services du SeD UMS.

- Préfrontale : Cela désigne une machine mise avant la(les) frontale(s) des calculateurs.

- SeD UMS : Cela désigne le programme contenant et exécutant les services du module UMS. Une seule instance est nécessaire pour faire tourner VISHNU.

- Sysfera-DS : Middleware open source développé par SysFera.

- Se référer au glossaire du document [ARCH] pour des définitions additionelles.

Chapter 3

Installation (à partir des sources)

3.1 Prérequis

- GCC version 4.4.3 minimum

- CMAKE version 2.6 minimum

- BOOST version 1.45 minimum

- OMNIORB version 4.1.4

- PGSQL-API (api PostGreSQL) version 8.0 minimum

- libcrypt

- SWIG 1.3 (SWIG 2 n'est pas compatible)

- SSH

- Sysfera-DS (DIET uniquement pour l'instant, version 2.7 minimum)

- JAVA version 1.6 (Option)

- Python au moins 2.5 (Option)

3.2 Compilation des sources

Vishnu utilise CMake comme système de construction et se conforme aux pratiques communèment admises. Il y a quatre options principales utilisables :

- CLIENT_ONLY, qui permet de ne compiler que les éléments clients si le flag est activé (OFF par défaut).

- ENABLE_PYTHON, qui permet d'activer la compilation du code PYTHON (OFF par défaut).

- ENABLE_JAVA, qui permet d'activer la compilation des sources JAVA pour les webs services (OFF par défaut).

- CMAKE_INSTALL_PREFIX, répertoire d'installation (/usr/local par défaut sur les plateformes *nix)

Par exemple, pour compiler Vishnu (clients et serveurs) avec l'API Python puis l'installer dans le répertoire /opt/vishnu sous *nix:

- 1. créer un répertoire build à la racine du projet et se placer dedans

- $ mkdir build

- $ cd build

- 2. générer le Makefile

- $ cmake -DENABLE_PYTHON=ON -DCMAKE_INSTALL_PREFIX=ON

- 3. lancer la compilation avec 2 jobs

- $ make -j 2

- 4. installation (en mode super-utilisateur)

- # make install

Note: pensez à ajouter le répertoire d'installation dans le $PATH utilisateur pour avoir accès à la commande.

Chapter 4

Déploiement de VISHNU

4.1 Lancement des serveurs

Le module UMS peut être lancé soit manuellement, soit en utilisant l'outil GoDIET : un logiciel de lancement pour Sysfera-DS depuis une machine d'administration.

NOTE: l'outil GoDIET sera disponible lors de la livraison finale du projet VISHNU

Pour mieux comprendre l'architecture de déploiement, se référer au document [ARCH], le chapitre 4, 'Technical Architecture'.

1. Avoir une base de données PostGreSQL accessible et initialisée (tables créées et premières données créées). Des scripts SQL sont fournis pour cela. Avoir la base PostGreSQL configurée pour qu'elle soit accessible par VISHNU (voir le fichier de configuration pg_hba.conf qui se situe dans un répertoire de la forme '/etc/postgresql/8.3/main' pour configurer la base).

2. Lancer le service de nommage CORBA sur la machine préfrontale. La commande est 'omniNames -start' pour la première fois, sinon 'omniNames' suffit. Attention, dans la configuration de l'omninames, bien utiliser l'adresse de l'hôte et non pas 'localhost' ou '127.0.0.1'.

3. Lancer le MA avec son fichier de configuration sur la machine préfrontale : 'dietAgent config.cfg'. Le fichier de configuration peut contenir les 3 lignes suivantes :

   • 'traceLevel = 1' : Le niveau de verbosité du master agent, cette valeur peut être entre 1 et 10 compris.
   • 'agentType = DIET_MASTER_AGENT' : Le type de l'agent, l'autre type disponible est DIET_LOCAL_AGENT mais dans notre cas il faut DIET_MASTER_AGENT.
   • 'name = MA0' : Le nom que l'on veut donner à l'agent.

4. Lancer le SeD UMS sur la préfrontale. Un exemple de commande de lancement est : './umssed ~/myClient_server/mathOp/SeD.cfg 1 pg localhost vishnu_user vishnu_user ~/Bureau/sendmail.py'

   • '~/myClient_server/mathOp/SeD.cfg' : le chemin jusqu'au fichier de configuration du SeD. Ce fichier peut par exemple contenir les 2 lignes suivantes :
     – 'traceLevel = 10' : Le niveau de verbosité du SeD UMS, ce niveau peut être entre 1 et 10 compris.
     – 'parentName = MA0' : Le nom du MA auquel le SeD UMS doit se lier. Ce doit être exactement le même nom que celui donné au même dans le champs 'name' du MA en question.
   • '1' : L'id de VISHNU à utiliser dans la DB, une seule est utilisable actuellement .
   • 'pg' : Pour utiliser une base postgrel. Actuellement la base oracle n'est pas utilisable.
   • 'localhost' : La DB à utiliser.
   • 'vishnu_user' (le premier) : Le nom d'utilisateur pour se connecter à la DB.
   • 'vishnu_user' (le second) : Le mot de passe pour se connecter à la DB.
   • '~/Bureau/sendmail.py' : Le script à utiliser pour envoyer les mails. Il sera fourni avec le module UMS.

5. Le module UMS de VISHNU est prêt à être utilisé. Pour ce faire, un client doit se connecter et soumettre des requêtes à VISHNU au moyen de clients UMS.

4.2   Test d'exécution d'un service depuis une machine client par shell

1. Une fois que la plateforme a été installée, se mettre sur un poste client avec VISHNU d'installé. Se référer au document [VISHNU_USERGUIDE] pour l'installation de la partie client.

2. Exporter la variable d'environnement VISHNU_CONFIG_FILE dans un script de configuration client. Se référer au guide d'installation du client [VISHNU_USER_GUIDE] pour connaître le contenu d'un fichier client.

3. Lancer la commande dans le shell 'vishnu_connect user', ou user représente un nom d'utilisateur dans la DB

4. Taper le mot de passe 'password', correspondant à l'utilisateur 'user'

5. Sur le client, un affichage doit signaler que le service a réussi. Dans le terminal ou le SeD UMS a été lancé et dans le terminal ou le MA tourne, selon le niveau de verbosité, plus ou moins d'informations, concernant le service effectué, doivent apparaitre.

6. Fermer la session avec 'vishnu_close'. Aucune erreur ne doit être remontée lors de ces tests.

Chapter 5

Administration

5.1 Présentation

Le module UMS correspond à la gestion des utilisateurs et des machines de VISHNU. Il permet aussi de sauvegarder la configuration de VISHNU à chaud et de la restaurer si besoin est. Dans toute la suite du chapitre, on supposera que l'utilisateur est déjà connecté avec un compte administrateur de VISHNU pour pouvoir réaliser ces manipulations. De plus, on présentera l'utilisation des commandes depuis le shell, mais cela reste valable depuis les API Python ou C++.

5.2 Gestion des utilisateurs

1. L'ajout d'un utilisateur se fait à l'aide de la commande 'vishnu_add_user'. Elle prend en paramètre le prénom de l'utilisateur, son nom de famille, les droits qui lui sont associés dans VISHNU (administrateur ou simple utilisateur) et son adresse de couriel. Tout ces paramètres sont obligatoires. Un privilège à 1 signifie administrateur, un privilège à 0 signifie un utilisateur. L'identifiant de l'utilisateur est généré et renvoyé.

2. La mise à jour d'un utilisateur ne peut être faite que par un administrateur. Cette mise à jour se fait avec un appel à la commande 'vishnu_update_user' et permet de modifier les paramètres de l'ajout (nom, prénom, statut, couriel). Il faut avoir l'identifiant de l'utilisateur (généré lors de la création de l'utilisateur) pour le désigner lors de la mise à jour.

   Note: le changement du statut d'un utilisateur à l'état "INACTIVE" correspond à un blocage de son compte.

3. La suppression d'un utilisateur efface toutes les informations liées à l'utilisateur de la base de donnée. Cette suppression se fait à l'aide de la commande 'vishnu_delete_user'.

4. La liste des utilisateurs ne peut être faite que par un administrateur. Cela se fait avec la commande 'vishnu_list_user'. Cette commande peut prendre en paramètre l'identifiant d'un utilisateur pour n'avoir les informations que concernant cet utilisateur.

5. Seul un administrateur peut réinitialiser le mot de passe d'un utilisateur de VISHNU. Pour ce faire, il doit appeller la commande 'vishnu_reset_password' en fournissant l'identifiant de l'utilisateur dont l'administrateur veut réinitialiser le mot de passe. Le nouveau mot de passe est temporaire et renvoyé par la commande. Lors de la prochaine connexion, l'utilisateur devra changer son mot de passe avec 'vishnu_change_password'.

5.3 Gestion des machines

1. L'ajout d'une machine se fait à l'aide de la commande 'vishnu_add_machine'. Cette commande prend en paramètre le nom de la machine, le site ou elle se trouve, le language de la description qui sera donnée pour la machine, le fichier contenant la clé publique et la description. Ces paramètres sont obligatoires, en passant par le shell, la description n'a pas besoin d'être fournie en paramètre mais elle est alors demandée à l'administrateur avant d'ajouter la machine. A la fin de l'ajout, l'identifiant généré pour la machine est renvoyé.

2. La mise à jour d'une machine se fait à l'aide de la commande 'vishnu_update_machine' et permet de modifier les paramètres mis lors de l'ajout de la machine. Il faut utiliser l'identifiant de la machine pour l'identifier lors de la mise à jour.

   Note: le changement du statut d'une machine à l'état "INACTIVE" correspond à un blocage de la machine. Cela rend la machine inaccessible aux utilisateurs de VISHNU mais toujours visible pour les administrateurs.

3. La suppression d'une machine se fait à l'aide de la commande 'vishnu_delete_machine' avec l'identifiant de la machine à supprimer. Cela supprime la machine de la base de données, ainsi que toutes les informations qui y sont attachées (Attention: cette commande est irréversible).

4. Les utilisateurs peuvent lister les machines, mais un administrateur a en plus une option qui est l'identifiant d'un utilisateur. Ceci lui permet de lister les machines sur lesquelles l'utilisateur a un compte VISHNU.

5.4 Gestion de la plateforme

1. L'administrateur peut faire une sauvegarde à chaud à un moment donné de VISHNU. Ceci sauvegarde les utilisateurs, les machines et les comptes des utilisateurs. Le fichier, dans lequel la configuration est, est retourné par la fonction. La fonction est 'vishnu_save_configuration', pas besoin de paramètres.

2. L'administrateur peut recharger une configuration précédente de VISHNU à l'aide de la commande 'vishnu_restore_configuration' qui a besoin du fichier de sauvegarde pour recharger la base. Avant de pouvoir lancer cette restauration, tous les utilisateurs de VISHNU doivent être déconnectés.

3. Un administrateur peut également définir les valeurs par défaut des options de VISHNU pour tout les utilisateurs (ces options sont le temps de déconnexion par défaut et le type de fermeture d'une session par défaut). Cela se fait en appelant 'vishnu_configure_default_option' en donnant le nom de l'option et sa nouvelle valeur.

5.5 Options propres à l'administrateur dans les commandes utilisateurs

1. Dans la fonction 'vishnu_connect', un administrateur peut donner l'identifiant d'un utilisateur pour se connecter sous le nom de cet utilisateur dans VISHNU.

2. Dans la fonction 'vishnu_list_history_cmd', l'administrateur peut lister toutes les commandes de tout les utilisateurs ou les commandes d'un utilisateur en particulier en fournissant l'identifiant de l'utilisateur.

3. Dans la fonction 'vishnu_list_local_accounts', l'administrateur peut lister toutes les comptes de tout les utilisateurs ou les comptes d'un utilisateur particulier en fournissant l'identifiant de l'utilisateur.

4. Dans la fonction 'vishnu_list_options', l'administrateur peut lister les toutes les options de tout les utilisateurs ou les options d'un utilisateur en particulier en fournissant l'identifiant de l'utilisateur.

5. Dans la fonction 'vishnu_list_sessions', l'administrateur peut lister les toutes les session de tout les utilisateurs ou les sessions d'un utilisateur en particulier en fournissant l'identifiant de l'utilisateur, ou les sessions sur une machine particulière en fournissant l'identifiant de la machine.

Chapter 6

Command reference

6.1 vishnu_add_user

vishnu_add_user — adds a new VISHNU user

Synopsis

vishnu_add_user [-h] *firstname lastname privilege email*

DESCRIPTION

This command allows an admin to add a new user in VISHNU. Several user information are necessary such as: lastname, firtsname and email adress. The admin also gives a VISHNU privilege to the new user and a new userId and password are sent to the user by email.

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 already exists in the database" [22]

"The user is locked" [23]

"The user is not an administrator" [25]

"The mail adress is invalid" [27]

"The session key is unrecognized" [28]

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

"The machine is locked" [34]

6.2 vishnu_update_user

vishnu_update_user — updates the user information except the userId and the password

Synopsis

```
vishnu_update_user [-h][-f firstname][-l lastname][-p privilege][-m email][-s status] userId
```

DESCRIPTION

This command allows an admin to update a VISHNU user information or to lock a user. When a user is locked, she/he can not uses VISHNU. However, it is not possible to change the privilege of another admin.

OPTIONS

-h *help* help about the command.

-f *firstname* represents the updated firstname of the user.

-l *lastname* represents the updated lastname of the user.

-p *privilege* represents the updated privilege of the user. The value must be an integer. Predefined values are: 0 (USER), 1 (ADMIN).

-m *email* represents the updated email adress of the user.

-s *status* represents the status of the user (LOCKED or ACTIVE). The value must be an integer. Predefined values are: 0 (INACTIVE), 1 (ACTIVE).

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 locked" [23]

"Trying to lock a user account that is already locked" [24]

"The user is not an administrator" [25]

"The mail adress is invalid" [27]

"The session key is unrecognized" [28]

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

6.3 vishnu_delete_user

vishnu_delete_user — removes a user from VISHNU

Synopsis

`vishnu_delete_user` [-h] *userId*

DESCRIPTION

This command allows an admin to delete a user from VISHNU. When a user is deleted from VISHNU all of her/his information are deleted from VISHNU. However, it is not possible to delete the VISHNU root user.

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 user is locked" [23]

"The user is not an administrator" [25]

"The session key is unrecognized" [28]

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

6.4 vishnu_reset_password

vishnu_reset_password — resets the password of a user

Synopsis

`vishnu_reset_password` [-h] *userId*

DESCRIPTION

This command allows an admin to reset the password of the user. The password generated is temporary and must be changed for using 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 userId is unknown" [21]

"The user is locked" [23]

"The user is not an administrator" [25]

"The session key is unrecognized" [28]

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

6.5   vishnu_save_configuration

vishnu_save_configuration — saves the configuration of VISHNU

Synopsis

vishnu_save_configuration [-h]

DESCRIPTION

This commands allows an admin to save the VISHNU configuration. This configuration contains the list of users, the lists of machines and the list of local user configurations. It is saved on a xml format on a file registered on the directory $HOME/.vishnu/configurati

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 user is not an administrator" [25]

"A problem occurs during the configuration saving " [39]

6.6 vishnu_restore_configuration

vishnu_restore_configuration — restores the configuration of VISHNU

Synopsis

```
vishnu_restore_configuration [-h] filePath
```

DESCRIPTION

This function must be used carefully as it replaces all the content of the VISHNU central database with the information stored in the provided file. This file contains the list of users, the lists of machines and the list of local user configurations. It can be created using the vishnu_save_configuration command. The "root" VISHNU user is the only user authorized to call this function, and this action must be done without any other user connected to VISHNU. After restoring, the vishnu database is re-initialized.

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 user is not an administrator" [25]

"A problem occurs during the configuration restoring" [40]

6.7 vishnu_add_machine

vishnu_add_machine — adds a new machine in VISHNU

Synopsis

```
vishnu_add_machine [-h] name site language sshPublicKeyFile machineDescription
```

DESCRIPTION

This command allows an admin to add a new machine in VISHNU. Several machine information are mandatory such as: name, site, language and the public ssh key of the VISHNU system account on the machine. This public key will be provided automatically to all new VISHNU users who will have to add it to the authorized keys of their own account on the machine.

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 user is not an administrator" [25]

"The session key is unrecognized" [28]

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

"The machineId already exists in the database" [33]

"The closure policy is unknown" [42]

6.8 vishnu_update_machine

vishnu_update_machine — updates machine description

Synopsis

vishnu_update_machine [-h] [-n *name*] [-s *site*] [-d *machineDescription*] [-l *language*] [-t *status*] [--k *sshPublicKeyFile*] *machineId*

DESCRIPTION

This command allows an admin to update a VISHNU machine or to locked it. A machine locked is not usable.

OPTIONS

-h *help* help about the command.

-n *name* represents the name of the machine.

-s *site* represents the location of the machine.

-d *machineDescription* represents the description of the machine.

-l *language* represents the language used for the description of the machine.

-t *status* represents the status of the machine. The value must be an integer. Predefined values are: 0 (INACTIVE), 1 (ACTIVE).

-k *sshPublicKeyFile* contains the path to the SSH public key used by VISHNU to access local user accounts.

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 user is not an administrator" [25]

"The session key is unrecognized" [28]

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

"The machine id is unknown" [32]

"The closure policy is unknown" [42]

6.9   vishnu_delete_machine

vishnu_delete_machine — removes a machine from VISHNU

Synopsis

vishnu_delete_machine [-h] *machineId*

DESCRIPTION

This command allows an admin to delete a machine from VISHNU. When a machine is deleted all of its information are 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 user is not an administrator" [25]

"The session key is unrecognized" [28]

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

"The machine id is unknown" [32]

6.10   vishnu_list_users

vishnu_list_users — lists VISHNU users

Synopsis

```
vishnu_list_users [-h] [-i userIdOption]
```

DESCRIPTION

This command allows an admin to display all users information except the passwords.

OPTIONS

-h *help* help about the command.

-i *userIdOption* allows an admin to get information about a specific user identified by his/her userId.

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]

6.11   vishnu_configure_default_option

vishnu_configure_default_option — configures a default option value

Synopsis

`vishnu_configure_default_option` [-h] *optionName value*

DESCRIPTION

Options in VISHNU corresponds to 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 allows an administrator to configure the default value of an option; this is the value that will be applied when the user has no configuration defined for that option using the vishnu_configure_option command.

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 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]

Chapter 7

C++ API Reference

7.1 addUser

addUser — adds a new VISHNU user

Synopsis

int **vishnu::addUser**(const string& sessionKey, User& newUser);

DESCRIPTION

This command allows an admin to add a new user in VISHNU. Several user information are necessary such as: lastname, firtsname and email adress. The admin also gives a VISHNU privilege to the new user and a new userId and password are sent to the user by email.

ARGUMENTS

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

newUser Input/Output argument. Object containing the new user information.

EXCEPTIONS

The following exceptions may be thrown:

"The userId already exists in the database" [22]

"The user is locked" [23]

"The user is not an administrator" [25]

"The mail adress is invalid" [27]

"The session key is unrecognized" [28]

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

"The machine is locked" [34]

7.2 updateUser

updateUser — updates the user information except the userId and the password

Synopsis

int **vishnu::updateUser**(const string& sessionKey, const User& user);

DESCRIPTION

This command allows an admin to update a VISHNU user information or to lock a user. When a user is locked, she/he can not uses VISHNU. However, it is not possible to change the privilege of another admin.

ARGUMENTS

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

user Input argument. Object containing user information.

EXCEPTIONS

The following exceptions may be thrown:

"The userId is unknown" [21]

"The user is locked" [23]

"Trying to lock a user account that is already locked" [24]

"The user is not an administrator" [25]

"The mail adress is invalid" [27]

"The session key is unrecognized" [28]

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

7.3 deleteUser

deleteUser — removes a user from VISHNU

Synopsis

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

DESCRIPTION

This command allows an admin to delete a user from VISHNU. When a user is deleted from VISHNU all of her/his information are deleted from VISHNU. However, it is not possible to delete the VISHNU root user.

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 who will be deleted from VISHNU.

EXCEPTIONS

The following exceptions may be thrown:

"The userId is unknown" [21]

"The user is locked" [23]

"The user is not an administrator" [25]

"The session key is unrecognized" [28]

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

7.4   resetPassword

resetPassword — resets the password of a user

Synopsis

int **vishnu::resetPassword**(const string& sessionKey, const string& userId, string& tmpPassword);

DESCRIPTION

This command allows an admin to reset the password of the user. The password generated is temporary and must be changed for using 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 password will be reset.

tmpPassword Output argument. The temporary password generated by VISHNU.

EXCEPTIONS

The following exceptions may be thrown:

"The userId is unknown" [21]

"The user is locked" [23]

"The user is not an administrator" [25]

"The session key is unrecognized" [28]

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

7.5 saveConfiguration

saveConfiguration — saves the configuration of VISHNU

Synopsis

int **vishnu::saveConfiguration**(const string& sessionKey, Configuration& configuration);

DESCRIPTION

This commands allows an admin to save the VISHNU configuration. This configuration contains the list of users, the lists of machines and the list of local user configurations. It is saved on a xml format on a file registered on the directory $HOME/.vishnu/configurati

ARGUMENTS

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

configuration Output argument. The configuration is an object which encapsulates the configuration description.

EXCEPTIONS

The following exceptions may be thrown:

"The user is not an administrator" [25]

"A problem occurs during the configuration saving " [39]

7.6 restoreConfiguration

restoreConfiguration — restores the configuration of VISHNU

Synopsis

int **vishnu::restoreConfiguration**(const string& sessionKey, const string& filePath);

DESCRIPTION

This function must be used carefully as it replaces all the content of the VISHNU central database with the information stored in the provided file. This file contains the list of users, the lists of machines and the list of local user configurations. It can be created using the vishnu_save_configuration command. The "root" VISHNU user is the only user authorized to call this function, and this action must be done without any other user connected to VISHNU. After restoring, the vishnu database is re-initialized.

ARGUMENTS

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

filePath Input argument. The filePath is the path of the file used to restore VISHNU configuration.

EXCEPTIONS

The following exceptions may be thrown:

"The user is not an administrator" [25]

"A problem occurs during the configuration restoring" [40]

7.7 addMachine

addMachine — adds a new machine in VISHNU

Synopsis

int **vishnu::addMachine**(const string& sessionKey, Machine& newMachine);

DESCRIPTION

This command allows an admin to add a new machine in VISHNU. Several machine information are mandatory such as: name, site, language and the public ssh key of the VISHNU system account on the machine. This public key will be provided automatically to all new VISHNU users who will have to add it to the authorized keys of their own account on the machine.

ARGUMENTS

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

newMachine Input/Output argument. Is an object which encapsulates the information of the machine which will be added in VISHNU except the machine id which will be created automatically by VISHNU.

EXCEPTIONS

The following exceptions may be thrown:

"The user is not an administrator" [25]

"The session key is unrecognized" [28]

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

"The machineId already exists in the database" [33]

"The closure policy is unknown" [42]

7.8 updateMachine

updateMachine — updates machine description

Synopsis

int **vishnu::updateMachine**(const string& sessionKey, const Machine& machine);

DESCRIPTION

This command allows an admin to update a VISHNU machine or to locked it. A machine locked is not usable.

ARGUMENTS

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

machine Input argument. Existing machine information.

EXCEPTIONS

The following exceptions may be thrown:

"The user is not an administrator" [25]

"The session key is unrecognized" [28]

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

"The machine id is unknown" [32]

"The closure policy is unknown" [42]

7.9 deleteMachine

deleteMachine — removes a machine from VISHNU

Synopsis

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

DESCRIPTION

This command allows an admin to delete a machine from VISHNU. When a machine is deleted all of its information are deleted from VISHNU.

ARGUMENTS

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

machineId Input argument. MachineId represents the identifier of the machine.

EXCEPTIONS

The following exceptions may be thrown:

"The user is not an administrator" [25]

"The session key is unrecognized" [28]

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

"The machine id is unknown" [32]

7.10   listUsers

listUsers — lists VISHNU users

Synopsis

int **vishnu::listUsers**(const string& sessionKey, ListUsers& listuser, const string& userIdOption = string());

DESCRIPTION

This command allows an admin to display all users information except the passwords.

ARGUMENTS

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

listuser Output argument. Listuser is the list of users .

userIdOption Input argument. Allows an admin to get information about a specific user identified by his/her userId.

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]

7.11   configureDefaultOption

configureDefaultOption — configures a default option value

Synopsis

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

DESCRIPTION

Options in VISHNU corresponds to 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 allows an administrator to configure the default value of an option; this is the value that will be applied when the user has no configuration defined for that option using the vishnu_configure_option command.

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 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]

Chapter 8

Python API Reference

8.1 VISHNU_UMS.addUser

VISHNU_UMS.addUser — adds a new VISHNU user

Synopsis

VISHNU_UMS.addUser(string sessionKey, User newUser);

DESCRIPTION

This command allows an admin to add a new user in VISHNU. Several user information are necessary such as: lastname, firtsname and email adress. The admin also gives a VISHNU privilege to the new user and a new userId and password are sent to the user by email.

ARGUMENTS

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

newUser Input/Output argument. Object containing the new user 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 userId already exists in the database" [22])

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

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

UMSVishnuException("The mail adress is invalid" [27])

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

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

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

8.2   VISHNU_UMS.updateUser

VISHNU_UMS.updateUser — updates the user information except the userId and the password

Synopsis

VISHNU_UMS.updateUser(string sessionKey, User user);

DESCRIPTION

This command allows an admin to update a VISHNU user information or to lock a user. When a user is locked, she/he can not uses VISHNU. However, it is not possible to change the privilege of another admin.

ARGUMENTS

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

user Input argument. Object containing user 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 userId is unknown" [21])

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

UMSVishnuException("Trying to lock a user account that is already locked" [24])

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

UMSVishnuException("The mail adress is invalid" [27])

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

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

8.3   VISHNU_UMS.deleteUser

VISHNU_UMS.deleteUser — removes a user from VISHNU

Synopsis

VISHNU_UMS.deleteUser(string sessionKey, string userId);

DESCRIPTION

This command allows an admin to delete a user from VISHNU. When a user is deleted from VISHNU all of her/his information are deleted from VISHNU. However, it is not possible to delete the VISHNU root user.

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 who will be deleted from 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 userId is unknown" [21])

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

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])

8.4 VISHNU_UMS.resetPassword

VISHNU_UMS.resetPassword — resets the password of a user

Synopsis

VISHNU_UMS.resetPassword(string sessionKey, string userId, string tmpPassword);

DESCRIPTION

This command allows an admin to reset the password of the user. The password generated is temporary and must be changed for using 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 password will be reset.

tmpPassword Output argument. The temporary password 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 userId is unknown" [21])

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

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])

8.5   VISHNU_UMS.saveConfiguration

VISHNU_UMS.saveConfiguration — saves the configuration of VISHNU

Synopsis

VISHNU_UMS.saveConfiguration(string sessionKey, Configuration configuration);

DESCRIPTION

This commands allows an admin to save the VISHNU configuration. This configuration contains the list of users, the lists of machines and the list of local user configurations. It is saved on a xml format on a file registered on the directory $HOME/.vishnu/configurati

ARGUMENTS

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

configuration Output argument. The configuration is an object which encapsulates the configuration description.

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 user is not an administrator" [25])

UMSVishnuException("A problem occurs during the configuration saving " [39])

8.6   VISHNU_UMS.restoreConfiguration

VISHNU_UMS.restoreConfiguration — restores the configuration of VISHNU

Synopsis

VISHNU_UMS.restoreConfiguration(string sessionKey, string filePath);

DESCRIPTION

This function must be used carefully as it replaces all the content of the VISHNU central database with the information stored in the provided file. This file contains the list of users, the lists of machines and the list of local user configurations. It can be created using the vishnu_save_configuration command. The "root" VISHNU user is the only user authorized to call this function, and this action must be done without any other user connected to VISHNU. After restoring, the vishnu database is re-initialized.

ARGUMENTS

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

filePath   Input argument. The filePath is the path of the file used to restore VISHNU configuration.

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 user is not an administrator" [25])

UMSVishnuException("A problem occurs during the configuration restoring" [40])

8.7   VISHNU_UMS.addMachine

VISHNU_UMS.addMachine — adds a new machine in VISHNU

Synopsis

VISHNU_UMS.addMachine(string sessionKey, Machine newMachine);

DESCRIPTION

This command allows an admin to add a new machine in VISHNU. Several machine information are mandatory such as: name, site, language and the public ssh key of the VISHNU system account on the machine. This public key will be provided automatically to all new VISHNU users who will have to add it to the authorized keys of their own account on the machine.

ARGUMENTS

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

newMachine Input/Output argument. Is an object which encapsulates the information of the machine which will be added in VISHNU except the machine id which will be created automatically 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 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 machineId already exists in the database" [33])

UMSVishnuException("The closure policy is unknown" [42])

8.8 VISHNU_UMS.updateMachine

VISHNU_UMS.updateMachine — updates machine description

Synopsis

VISHNU_UMS.updateMachine(string sessionKey, Machine machine);

DESCRIPTION

This command allows an admin to update a VISHNU machine or to locked it. A machine locked is not usable.

ARGUMENTS

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

machine Input argument. Existing machine 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 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 machine id is unknown" [32])

UMSVishnuException("The closure policy is unknown" [42])

8.9 VISHNU_UMS.deleteMachine

VISHNU_UMS.deleteMachine — removes a machine from VISHNU

Synopsis

VISHNU_UMS.deleteMachine(string sessionKey, string machineId);

DESCRIPTION

This command allows an admin to delete a machine from VISHNU. When a machine is deleted all of its information are deleted from VISHNU.

ARGUMENTS

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

machineId Input argument. MachineId represents the identifier of the 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 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 machine id is unknown" [32])

8.10 VISHNU_UMS.listUsers

VISHNU_UMS.listUsers — lists VISHNU users

Synopsis

VISHNU_UMS.listUsers(string sessionKey, ListUsers listuser, string userIdOption = string());

DESCRIPTION

This command allows an admin to display all users information except the passwords.

ARGUMENTS

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

listuser Output argument. Listuser is the list of users .

userIdOption Input argument. Allows an admin to get information about a specific user identified by his/her 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 user is not an administrator" [25])

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

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

8.11 VISHNU_UMS.configureDefaultOption

VISHNU_UMS.configureDefaultOption — configures a default option value

Synopsis

VISHNU_UMS.configureDefaultOption(string sessionKey, OptionValue optionValue);

DESCRIPTION

Options in VISHNU corresponds to 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 allows an administrator to configure the default value of an option; this is the value that will be applied when the user has no configuration defined for that option using the vishnu_configure_option command.

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 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])