

Guide de l'administrateur des outils de gestion de fichiers et batch DIET

G. Le Mahec

19 octobre 2010

Table des matières

1	Introduction	2
1.1	L'intergiciel DIET	2
1.2	Architecture globale de l'environnement logiciel	2
2	Installation	4
2.1	Dépendances logicielles	4
2.2	Installation à partir des sources	4
2.2.1	Compilation de DIET	4
2.2.2	Compilation des outils diet-fms et diet-bms	4
2.3	Configuration du système pour l'exécution des outils	5
3	Configuration des serveurs	5
3.1	Configuration des applications serveurs	6
3.2	Configuration DIET des serveurs	6
3.3	Ajout d'un utilisateur	7
4	Exécution des serveurs	8
4.1	Utilisation de tunnels ssh	8
5	Configuration du poste client	9
6	Problèmes identifiés et solutions	9
7	Exemples de configurations	10
7.1	Exemples de fichiers de configuration	10
7.2	Exemple de configuration pour l'intégration de la machine BGP	11
7.2.1	Configuration du MA	11
7.2.2	Configuration des SeDs	11
7.2.3	Configuration du client	12
7.2.4	Lancement de la plate-forme	12

Le présent document détaille l'installation et la configuration des différents éléments composant l'environnement logiciel de gestion des fichiers et des systèmes de batch par l'intermédiaire de l'intergiciel DIET. La première section présente le fonctionnement général de DIET et des services *DIET file management system* et *DIET batch management system*. La seconde partie détaille les différentes étapes de l'installation des différents logiciels nécessaires au fonctionnement de ces services. La section 4 est consacrée au lancement des applications serveurs, notamment dans un environnement sécurisé. La section 5, quant à elle, présente la configuration des applications clientes. Dans la section qui suit seront présentés les problèmes connus et les moyens de les résoudre. Enfin, la dernière section fournit des exemples de configurations aussi bien pour les postes clients que pour les serveurs.

1 Introduction

La plate-forme de calcul du département *Recherche et Développement* de la société EDF est composée de plusieurs clusters et super-calculateurs hétérogènes et géographiquement répartis. Ces différentes ressources matérielles sont reliées entre elle par un réseau informatique et chacune d'elle dispose de son propre système de fichiers et de soumission de jobs. L'objectif principal de l'environnement logiciel présenté dans ce document est d'uniformiser et simplifier l'accès à ces différentes ressources, permettant aux utilisateurs d'y accéder facilement, avec une gestion transparente de leurs droits d'accès. Cet environnement est construit comme un ensemble de clients/serveurs pour l'intergiciel DIET et propose un ensemble de services à même de répondre à cet objectif.

1.1 L'intergiciel DIET

L'intergiciel DIET développé par l'équipe GRAAL de l'ENS de Lyon (INRIA Rhône-Alpes) repose sur le paradigme d'appels de procédures distantes Grid-RPC. Il propose une interface de création de programmes client/serveur adaptée à la grille en rendant transparente la complexité de celle-ci pour l'utilisateur. Il permet de définir des politiques d'ordonnancement avancées grâce à un système de *plugins* et utilise un gestionnaire de données permettant la réplication et l'accès transparent aux données sur ce type de plates-formes.

L'architecture de l'intergiciel DIET est composée de trois éléments principaux :

- Les clients : Ce sont des applications qui font l'interface entre un utilisateur désirant effectuer un appel de service sur la grille et l'intergiciel.
- Les "Server Daemons" (SeD) : Ce sont les serveurs qui fournissent les services à la grille. Un SeD encapsule un serveur de calcul. Il peut être par exemple placé sur le point d'entrée d'une machine parallèle (super-calculateur, cluster etc.) ou simplement sur une machine individuelle du réseau. Un SeD stocke la liste des services qu'il offre aux utilisateurs, mais aussi des informations sur sa charge et ses capacités matérielles (nombre de processeurs, charges des processeurs, quantité de mémoire installée etc.)
- Les agents : Ce sont les éléments de DIET qui effectuent la sélection du ou des serveurs utilisés pour l'exécution d'un service. Ils sont interconnectés suivant une topologie d'arbre dont le sommet est un "agent maître" (Master Agent ou MA) qui sera le seul pouvant être contacté par les applications clientes à la recherche d'un service.

DIET est construit sur l'architecture CORBA et utilise la librairie *omniORB* comme implantation de son ORB (Object Request Broker).

1.2 Architecture globale de l'environnement logiciel

L'environnement logiciel est constitué de sous-ensembles principaux : *DIET file management system* et *DIET batch management system*. Le premier ensemble est consacré à la gestion des fichiers répartis sur les différents systèmes de fichiers de la plate-forme et le second à la soumission et au suivi des jobs sur les différents systèmes de batchs de celle-ci. Pour interagir avec l'environnement des différents utilisateurs, les serveurs réalisent les opérations demandées par l'intermédiaire de la commande *ssh*. La figure 1

présente l'architecture globale de l'environnement logiciel :

Les différents éléments de la hiérarchie DIET communiquent par des appels CORBA. Ils se déclarent auprès du service de nom de la plate-forme afin de s'identifier entre eux. Les SeD publient les différents services de gestion de fichiers et des systèmes de batch. Les clients soumettent leurs requêtes au MA en utilisant également des appels CORBA. Ce dernier retourne une liste de SeDs pouvant répondre à la requête, classés suivant un ordre configurable par l'utilisation de *plugins schedulers*. Les clients s'adressent alors directement au SeD choisi pour exécuter le service demandé. Afin d'accéder à l'environnement de l'utilisateur, les SeDs utilisent des commandes *ssh* pour l'exécution des différents services.

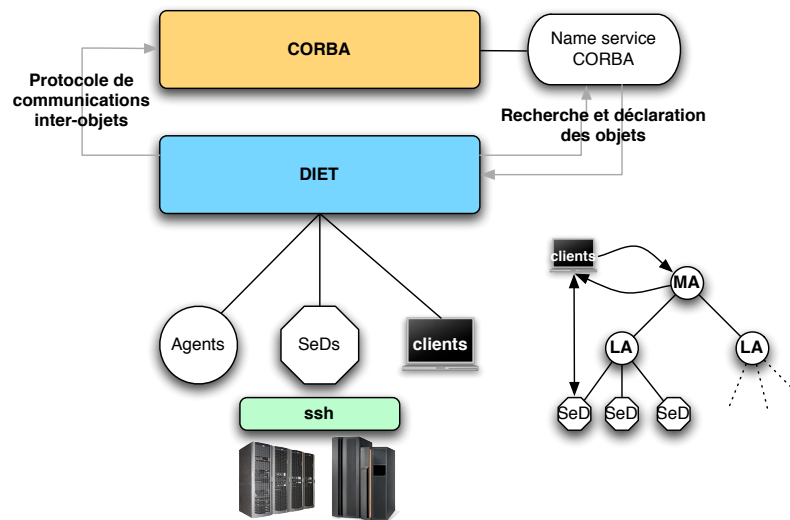


FIGURE 1 – Architecture globale de l'environnement logiciel

Les services proposés et leurs clients respectifs sont les suivants :

- Pour le système de gestion des batchs :
 - **diet-cancel** : Annule une soumission de job.
 - **diet-list** : Liste l'ensemble des jobs d'un système de batch.
 - **diet-submit** : Soumet un job à un système de batch.
- Pour le système de gestion des fichiers :
 - **diet-chgrp** : Change le groupe d'appartenance d'un fichier distant.
 - **diet-chmod** : Change les permissions d'accès d'un fichier distant.
 - **diet-cp** : Copie un fichier depuis un système de fichiers distant vers un autre système de fichiers distant.
 - **diet-head** : Retourne les n premières lignes d'un fichier distant.
 - **diet-ls** : Liste les fichiers d'un répertoire distant et/ou retourne les informations usuelle quant à ce fichier (permission, propriétaire etc).
 - **diet-mkdir** : Crée un répertoire sur un système de fichiers distant.
 - **diet-mv** : Déplace un fichier depuis un système de fichiers distant vers un autre système de fichiers distant.
 - **diet-rm** : Supprime un fichier d'un système de fichiers distant.
 - **diet-rmdir** : Supprime un répertoire d'un système de fichiers distant.
 - **diet-status** : Retourne la progression d'un transfert initié par la commande **diet-cp**.
 - **diet-tail** : Retourne les n dernières lignes d'un fichier distant.

Reportez-vous à la documentation relative aux différents services pour plus de détails quant aux fonctionnalités de ceux-ci.

2 Installation

Les logiciels de gestion de fichiers et de batchs sont fournis sous la forme de code sources ou en fichiers binaires pré-compilés. Dans les deux cas, ils présentent des dépendances avec plusieurs logiciels libres disponibles sur Internet. Afin de faciliter l'installation de l'environnement logiciel, ces dépendances sont également fournies (à l'exception de ssh, dont on suppose la disponibilité sur les machines cibles).

2.1 Dépendances logicielles

L'ensemble logiciel dépend pour son fonctionnement de plusieurs autres logiciels et bibliothèques :

- **omniORB 4** : Un ORB CORBA sous licence LGPL disponible à l'adresse suivante
<http://omniorb.sourceforge.net/>
- **SSH** : Secure SHell, une implantation du logiciel SSH par exemple *openSSH* disponible à l'adresse suivante :
<http://www.openssh.com/>
- **libuuid** (Dépendance optionnelle) : Une bibliothèque de génération d'identifiants uniques décentralisée.
- **Les commandes de bases des systèmes UNIX.**
- **Un ou plusieurs systèmes de gestion de batchs** (Torque, PBS, OAR, etc.)
- **CMake** : Un système de génération de fichiers *Makefile* multi-plates-formes et open-source. *Uniquement dans le cas d'une installation à partir des sources*
<http://www.cmake.org>
- **L'environnement de développement GNU** : Le compilateur g++ ; l'outils de compilation *make* etc. *Uniquement dans le cas d'une installation à partir des sources*

2.2 Installation à partir des sources

Pour installer *diet-fms* et *diet-bms* à partir des sources, les utilisateurs doivent disposer des outils classiques de compilation, d'une installation fonctionnelle de *omniORB* > 4.0.7 et de l'outil *CMake* ≥ 2.6 .

2.2.1 Compilation de DIET

La première étape de l'installation de l'environnement logiciel est l'installation de l'intergiciel DIET. La suite de commandes suivante, adaptée à la configuration du système cible permet la compilation et l'installation de DIET :

```
$ cd <répertoire des sources de DIET>
$ mkdir build
$ cd build
$ cmake .. -DOMNIORB4_DIR:PATH=<chemin d'installation d'omniORB> \
          -DCMAKE_INSTALL_PREFIX=<chemin d'installation souhaité pour DIET>
$ make && make install
```

2.2.2 Compilation des outils diet-fms et diet-bms

Une fois DIET installé, il reste à compiler les outils eux-mêmes. La suite de commandes suivante permet la compilation et l'installation de *diet-fms* et *diet-bms* :

```
$ cd <répertoire des sources de diet-bms>
$ mkdir build
$ cd build
$ cmake .. -DDIET_DIR:PATH=<chemin d'installation de DIET> \
          -DCMAKE_INSTALL_PREFIX=<chemin d'installation souhaité pour diet-bms> \
          -DCFG_FILE="<emplacement souhaité pour le fichier de configuration>"
$ make && make install
```

```
$ cd <répertoire des sources de diet-fms>
$ mkdir build
$ cmake .. -DDIET_DIR:PATH=<chemin d'installation de DIET> \
    -DCMAKE_INSTALL_PREFIX=<chemin d'installation souhaité pour diet-fms> \
    -DCFG_FILE="<emplacement souhaité pour le fichier de configuration>"
$ make && make install
```

Le paramètre `CFG_FILE` est utilisé pour désigner l'emplacement du fichier de configuration par défaut des serveurs. Sa valeur par défaut est `/etc/DIET/diet-fmd.cfg` ou `/etc/DIET/diet-bmd.cfg` respectivement pour *diet-fms* et *diet-bms*. L'utilisation d'un fichier de configuration par défaut facilite grandement la configuration des serveurs, il est donc conseillé de fixer ce paramètre de compilation.

2.3 Configuration du système pour l'exécution des outils

Une fois les outils installés, il convient de les rendre accessibles aux utilisateurs. Pour automatiser cette configuration, des fichiers exemples, adaptés aux différentes machines de la plate-forme EDF sont fournis. Ces fichiers contiennent divers initialisations de variables d'environnement du système permettant l'exécution simple des outils. Les variables importantes qui y sont fixées sont les suivantes :

- `PATH` : La liste ordonnée des chemins à parcourir pour trouver les exécutable. Dans notre cas, on y ajoutera le chemin vers le répertoire `bin` de l'installation d'omniORB, de DIET et des outils *diet-bms* et *diet-fms*.

```
$ export PATH=$PATH:<omniORB dir>/bin:<DIET dir>/bin:<diet-fms dir>/bin:\
    <diet-bms dir>/bin
```

- `LD_LIBRARY_PATH` : Le chemin de recherche des bibliothèques dans l'arborescence du système. Dans notre cas, on y ajoutera les chemins vers les répertoires `lib` d'omniORB, de DIET et des outils *diet-bms* et *diet-fms*.

```
$ export LD_LIBRARY_PATH=<omniORB dir>/lib:<DIET dir>/lib:<diet-fms dir>/lib:\
    <diet-bms dir>/lib
```

- `OMNIORB_CONFIG` : Le chemin d'accès au fichier de configuration d'omniORB. Un chemin par défaut est fixé à la compilation d'omniORB, il s'agit en général de `/etc/omniORB4.cfg`

Une fois ces variables fixées, les binaires sont prêts à être exécutés. Pour simplifier ces opérations et pour garantir la bonne exécution des outils *diet-bms* et *diet-fms* il est conseillé d'enregistrer ces opérations dans un fichier sur le modèle des fichiers `diet-env.sh` fournis. Il suffit alors d'exécuter la commande `source diet-env.sh` ou `. diet-env.sh` pour que les variables soit correctement initialisées. Les outils présentés effectuant des exécutions à distance de commandes d'omniORB, il est nécessaire qu'au moins les variables `PATH` et `LD_LIBRARY_PATH` permettent toujours l'accès au répertoire `bin` de son installation. Pour cela, le plus simple consiste à ajouter la commande `. diet-env.sh` à la fin du fichier `.bashrc` de l'utilisateur.

3 Configuration des serveurs

Afin d'effectuer les opérations nécessitant une identification de l'utilisateur, les serveurs doivent être convenablement configurés. De même, pour permettre le stockage temporaire des fichiers avant leur transmission vers d'autres machines, un répertoire de stockage temporaire disposant de suffisamment d'espace doit être indiqué dans la configuration du serveur. Chaque serveur dispose par ailleurs d'une configuration de DIET qui lui est propre. Les sections 3.1 et 3.2 présentent en détail la configuration des serveurs *DIET file management Daemon (diet-fmd)* et *DIET batch management Daemon (diet-bmd)*.

3.1 Configuration des applications serveurs

Le système de gestion des fichiers et le système de gestion des batchs partagent trois paramètres communs qui doivent être définis pour l'exécution des serveurs :

- Le paramètre **dietConfig** : Il s'agit du chemin vers le fichier de configuration DIET du serveur. Ce paramètre peut être fixé dans le fichier de configuration générale du serveur, par les variables d'environnement `DFMS_DIET_CONFIG` et `DBMS_DIET_CONFIG` respectivement pour le *File management Daemon* et le *Batch management Daemon* ou encore sur la ligne de commande en paramètre de l'option `--config`.
- Le paramètre **userTable** : Il s'agit du chemin vers un fichier contenant une table de correspondance entre utilisateurs distants et utilisateurs locaux. Ce paramètre peut être fixé dans le fichier de configuration générale, par l'utilisation des variables d'environnement `DFMS_USER_TABLE` et `DBMS_USER_TABLE` ou par la ligne de commande en paramètre de l'option `--user-table`.
- Le paramètre **groupTable** : Il s'agit du chemin vers un fichier contenant une table de correspondance entre groupes distants et groupes locaux. Ce paramètre peut être fixé dans le fichier de configuration générale, par l'utilisation des variables d'environnement `DFMS_GRP_TABLE` et `DBMS_GRP_TABLE` ou par la ligne de commande en paramètre de l'option `--grp-table`.

Le format employé pour les tables de correspondances est décrit dans la section suivante.

À ces trois paramètres communs s'ajoutent deux paramètres spécifiques aux serveurs :

- Le paramètre **tmpDirectory** : C'est un paramètre spécifique au *File management Daemon*. Il s'agit du répertoire à utiliser pour le stockage temporaire des fichiers transférés depuis un compte utilisateur spécifique. Ce répertoire devrait être en lecture seule pour l'utilisateur exécutant le démon et en écriture/exécution pour tous (mode octal 733). Ce paramètre peut être fixé dans le fichier de configuration générale, par l'utilisation de la variable d'environnement `DFMS_TMP_DIR` ou par la ligne de commande en paramètre de l'option `--tmp-dir`.
- Le paramètre **batch** : C'est un paramètre spécifique au *Batch management Daemon*. Il s'agit de la désignation du système de batch à utiliser localement. Ce paramètre peut être fixé dans le fichier de configuration générale, par l'utilisation de la variable d'environnement `DBMS_BATCH_NAME` ou par la ligne de commande en paramètre de l'option `--batch`.

Remarque : Il est important de noter qu'une configuration dans le fichier de configuration générale est masquée par la définition de la variable d'environnement correspondante et que cette dernière est elle-même supplantée par la configuration sur la ligne de commande.

3.2 Configuration DIET des serveurs

Chaque serveur doit disposer d'une configuration DIET lui permettant de se connecter au reste de la hiérarchie. Cette configuration est regroupée dans le fichier désigné par le paramètre de configuration **dietConfig** (voir section 3.1).

Les paramètres importants qui doivent être fixés dans ce fichier sont les suivants :

- Le paramètre **parentName** : Ce paramètre désigne le nom de l'agent de la hiérarchie auquel doit se connecter le serveur.
- Le paramètre **storageDirectory** : Il s'agit du chemin vers le répertoire utilisé par le gestionnaire de données de DIET pour stocker temporairement les données transmises.
- Le paramètre **sshPath** : Ce paramètre désigne le chemin d'accès au client SSH. Il n'est utile que dans le cas d'une utilisation de DIET à travers des tunnels SSH. Pour plus de détails consulter la section 4.1.
- Le paramètre **sshProxies** : Il s'agit du chemin vers le fichier de configuration des tunnels SSH. Ce paramètre n'est indispensable que dans le cadre de l'utilisation de DIET à travers des tunnels SSH.

- Le paramètre **convertiorPath** : Il s'agit du chemin vers l'application *convertior* d'omniORB. Ce paramètre n'est indispensable que dans le cadre de l'utilisation de DIET à travers des tunnels SSH. D'autres paramètres peuvent être fixés pour résoudre des problèmes de configuration réseau :
- Le paramètre **dietHostname** : Fixe le nom d'hôte à utiliser pour le serveur. Ce paramètre peut permettre de résoudre des problèmes posés par l'existence de plusieurs interfaces réseau dans une machine.
- Le paramètre **dietPort** : Fixe le port TCP à utiliser pour le serveur. Ce paramètre peut permettre de résoudre des problèmes de conflits d'utilisation de ports TCP par d'autres serveurs ou d'autres applications.

3.3 Ajout d'un utilisateur

Au démarrage du serveur, celui-ci consulte sa configuration afin de déterminer si des transformations de noms d'utilisateurs et de groupes sont nécessaires. Si le paramètre **userTable** est défini dans la configuration, le serveur lit le fichier désigné et enregistre les correspondances entre utilisateurs distants et utilisateurs locaux ainsi que le répertoire de départ à utiliser pour l'utilisateur visé et la clé SSH nécessaire à l'accès aux ressources pour ce dernier.

Un tel fichier se présente de la manière suivante :

La première colonne correspond aux identifiants globaux des utilisateurs. La seconde correspond à l'identifiant local de l'utilisateur, la troisième colonne définit le répertoire HOME à utiliser et la dernière colonne le chemin vers la clé privée utilisée pour les connexions *ssh* de cet utilisateur.

```
pierre.dupont  pdupont  /home/lyon/pdupont  /opt/diet/dfms/etc/keys/pdupont_rsa
paul.dupuis   pdupuis  /home/paris/pdupuis  /opt/diet/dfms/etc/keys/pdupuis_rsa
jacques.duval jduval   /home/papeete/jduval /opt/diet/dfms/etc/keys/jduval_rsa
jean.duflot   jduflot  /home/lyon/jduflot  /opt/diet/dfms/etc/keys/jduflot_rsa
```

Avec le fichier exemple donné ci-dessus, lorsque Pierre Dupont voudra accéder à un fichier de son répertoire utilisateur, la connexion *ssh* s'effectuera avec l'identifiant **pdupont** dans le répertoire **/home/lyon/pdupont** à l'aide de la clé RSA stockée dans le fichier **/opt/diet/dfms/etc/keys/pdupont_rsa**.

De la même manière, les groupes d'utilisateurs n'étant pas forcément identiques sur toutes les machines de la plate-forme, au démarrage, les serveurs peuvent consulter une table de correspondance des groupes. Le fichier contenant une telle table n'est composé que de deux colonnes : la colonne des noms de groupes distants à mettre en relation avec les noms de groupes locaux de la deuxième colonne.

Un tel fichier se présente de la manière suivante :

```
lyon    users
paris   users
papeete users
admins  admin
guests  temp
```

Avec le fichier exemple donné ci-dessus, un fichier appartenant au groupe d'identifiant global **admins** sera classé dans le groupe **admin** sur la machine locale. Il est à noter que lorsqu'il est impossible de maintenir une distinction de groupes d'utilisateurs existante au niveau globale sur la machine locale, les accès aux données peuvent être augmentés pour certains utilisateurs. Ainsi, avec la configuration donnée ici, un fichier du groupe **paris** dont les droits d'accès empêchaient l'accès à un utilisateur du groupe **lyon** verra cette restriction disparaître sur la machine locale. Le choix de correspondance des groupes doit donc être effectué soigneusement et à défaut de pouvoir faire correspondre chaque groupe global à un groupe local au moins aussi restrictif, l'utilisateur veillera à réduire les droits d'accès à ses fichiers ou changer le groupe de ceux-ci.

Ajouter un utilisateur au système consiste à ajouter la ligne correspondante à la table des utilisateurs des différents serveurs avant de relancer ceux-ci.

4 Exécution des serveurs

La configuration effectuée, les serveurs *diet-fmd* et *diet-bmd* peuvent être lancés depuis la ligne de commande, avec ou sans paramètres si ceux-ci sont fixés dans un fichier de configuration générale ou par les variables d'environnement.

Remarque : Il est possible qu'il faille que l'utilisateur définisse une variable d'environnement OMNIORB_CONFIG, désignant le fichier de configuration omniORB du serveur, notamment lorsqu'omniORB a été installé par l'utilisateur dans un répertoire différent du répertoire par défaut. Pour plus de détails quant à la configuration d'omniORB, se reporter à sa documentation (voir <http://omniORB.sourceforge.net/docs.html>).

4.1 Utilisation de tunnels ssh

Lorsque la hiérarchie DIET est répartie sur plusieurs réseaux dont les seules communications entre eux ne peuvent s'effectuer que par l'intermédiaire d'une connexion sécurisée SSH, il est nécessaire de renseigner DIET sur la manière de contacter ces réseaux. DIET crée alors automatiquement des "tunnels SSH" de "renvois" de ports distants vers le port local utilisé par le serveur et déclare les nouveaux objets sur l'hôte distant avec des références corrigées de telle manière que les communications s'effectuent par l'intermédiaire de ces tunnels.

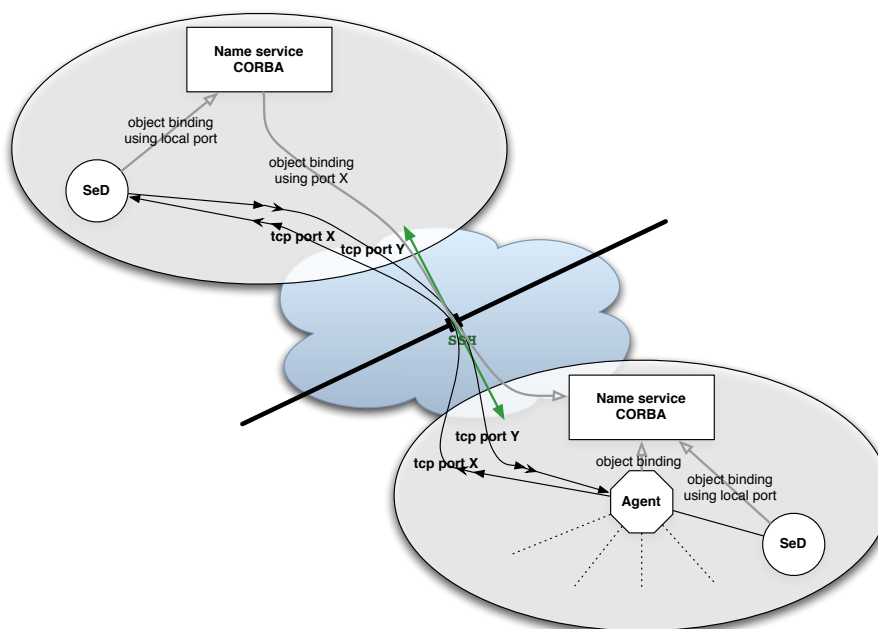


FIGURE 2 – Connexion de DIET à travers SSH

La figure 2 présente le fonctionnement général des connexions de DIET à travers SSH :

Au démarrage du serveur, celui-ci enregistre tous les objets CORBA dans le service de nom CORBA du réseau accessible. Les objets sont accessibles directement en utilisant le nom d'hôte et le port réels de l'objet. Les objets ainsi créés sont inaccessibles depuis le second réseau joignable uniquement par SSH. DIET crée alors un tunnel SSH depuis le second réseau. Celui-ci pointe vers l'hôte et le port de l'objet visé. Cependant, le service de nom "local" n'étant pas accessible depuis l'autre réseau, DIET doit enregistrer une référence à l'objet dans le service de nom distant. Or, l'IOR de l'objet local contient la référence à la machine et au port localement utilisés par l'objet CORBA et donc injoignables depuis le second réseau. DIET effectue alors la correction de l'IOR faisant pointer celui-ci sur le tunnel SSH "écoutant" sur le second réseau avant de l'enregistrer dans le second service de nom. Désormais lorsqu'une application du second réseau demande une référence à l'objet du premier réseau à son service

de nom CORBA, celui-ci lui retourne une référence d'objet pointant sur le tunnel SSH et donc ainsi accessible.

Toutes ces étapes conditionnent le format du fichier de configuration des communications à travers SSH dans DIET. En effet, il est nécessaire de fournir, pour chaque réseau à relier via SSH :

- Un identifiant du réseau à contacter
- Le nom d'hôte ou l'adresse IP de l'hôte sur lequel l'objet est enregistré.
- Le nom d'hôte ou l'adresse IP du serveur SSH à contacter.
- Le nom d'utilisateur utilisé pour la connexion SSH.
- Le port du serveur SSH.
- La clé privée employée pour la connexion SSH.
- Le chemin d'installation d'omniORB sur la machine distante.
- La référence du service de nom sur la machine distante.

Le nom du fichier de configuration des tunnels est initialisé dans le fichier de configuration DIET par l'intermédiaire de l'attribut **sshProxies**. Une ligne de ce fichier se présente de la manière suivante :

```
dist object.host n01.dist.fr login 22 /keys/login-key /usr \  
NameService=corbaname::n02.dist.fr:2809
```

Dans cet exemple, un tunnel SSH est créé depuis **object.host** vers la machine **n01.dist.fr** en se connectant en SSH sur le port 22 avec l'utilisateur **login** grâce à la clé enregistrée dans le fichier **/keys/login-key**. Sur l'hôte distant omniORB est installé dans le répertoire **/usr** et le service de nom est lancé sur la machine **n02.dist.fr** et écoute sur le port 2809.

Remarque : Cette configuration est propre à DIET et est donc valable, - après adaptation à chaque machine - et nécessaire sur tous les nœuds de la hiérarchie comme sur tous les clients.

5 Configuration du poste client

Comme les serveurs, les applications clientes nécessitent un fichier de configuration DIET. Par défaut, l'application tentera d'accéder au fichier **client.cfg** situé dans le répertoire **.diet** du répertoire personnel de l'utilisateur. Le paramètre de la ligne de commande **--config** permet de redéfinir le chemin vers ce fichier ou de s'abstenir de le copier dans le répertoire par défaut.

Le fichier de configuration DIET d'un client doit contenir les paramètres suivants :

- Le paramètre **MAName** : Le nom du Master Agent à contacter pour la recherche de serveurs et la soumission des requêtes.
- Le paramètre **sshPath** : Ce paramètre désigne le chemin d'accès au client SSH. Il n'est utile que dans le cas d'une utilisation de DIET à travers des tunnels SSH. Pour plus de détails consulter la section 4.1.
- Le paramètre **sshProxies** : Il s'agit du chemin vers le fichier de configuration des tunnels SSH. Ce paramètre n'est indispensable que dans le cadre de l'utilisation de DIET à travers des tunnels SSH.
- Le paramètre **convertiorPath** : Il s'agit du chemin vers l'application *convertior* d'omniORB. Ce paramètre n'est indispensable que dans le cadre de l'utilisation de DIET à travers des tunnels SSH.

6 Problèmes identifiés et solutions

Durant les différents tests effectués sur la plate-forme de calcul d'EDF, deux problèmes principaux ont été rencontrés, il est à noter que ces problèmes apparaissent avec les comptes dont nous disposons pour effectuer les tests et qu'ils n'apparaissent peut être pas avec tous les comptes utilisateurs.

1. Problèmes de connexions SSH sans mot de passe, vers et depuis certaines machines : Les connexions SSH sans mot de passe sont impossibles depuis les machines *clavi2q2* et *rendvous* vers la machine *clamart2*.

Une solution temporaire, dans l'attente d'un changement de configuration du serveur SSH de *clamart2* est d'entrer le mot de passe de l'utilisateur ouvrant les tunnels SSH sur *clamart2* à chaque demande. Les applications clientes étant les plus pénalisées par ce problème, il est conseillé de les exécuter depuis la machine *bgp* qui ne nécessite pas de mot de passe pour se connecter vers *clamart2*

2. L'exécution de la commande de binding d'un nouveau contexte CORBA à distance, via SSH, est bloquante lorsqu'elle est effectuée sur la machine *frontier* vers la machine *bgp*.

Une solution temporaire permettant l'exécution de la plate-forme est d'identifier les processus bloquant par la commande :

```
$ ps ax | grep bind_new_context
```

Il suffit alors de tuer le processus grâce à la commande `kill`.

7 Exemples de configurations

Nous présentons ici quelques exemples de fichiers de configuration utilisés durant nos tests sur la plate-forme de calcul d'EDF.

7.1 Exemples de fichiers de configuration

Fichier de configuration générale de *diet-fms* - *diet-fmd.cfg*

```
dietConfig = /home/le-mahec/etc/SeD.cfg
userTable = /home/le-mahec/etc/users
groupTable = /home/le-mahec/etc/groups
tmpDirectory = /home/le-mahec/tmp
```

Fichier de configuration générale de *diet-bms* - *diet-bmd.cfg*

```
dietconfig = /home/le-mahec/etc/SeD-bmd.cfg
usertable = /home/le-mahec/etc/users
grouptable = /home/le-mahec/etc/groups
batch = torque
```

Fichier de configuration DIET d'un serveur - *SeD.cfg*

```
parentName = MAO
storageDirectory = /tmp/DIET-SeDs
convertiorPath = /home/le-mahec/softs/omniORB/bin/convertior
sshProxies = /home/le-mahec/etc/sshProxies
sshPath = /usr/bin/ssh
```

Fichier de configuration DIET d'un client - *client.cfg*

```
MAName = MAO
traceLevel = 0
convertiorPath = /home/le-mahec/softs/omniORB/bin/convertior
sshProxies = /home/le-mahec/etc/sshProxies
sshPath = /usr/bin/ssh
```

Fichier de configuration SSH pour DIET - sshProxies

```
bgp localhost bgp le-mahec 22 /home/le-mahec/.ssh/id_rsa \  
/gpfs/home/le-mahec/softs/omniORB NameService=corbaname::localhost  
frontier localhost frontier le-mahec 22 /home/le-mahec/.ssh/id_rsa \  
/gpfs2/home/le-mahec/softs/omniORB NameService=corbaname::localhost  
clai2q2 localhost clai2q2 root 22 /home/le-mahec/.ssh/id_rsa \  
/root/softs/omniORB NameService=corbaname::localhost  
clamart2 localhost clamart2 lemahec 22 /home/le-mahec/.ssh/id_rsa \  
/home/lemahec/softs/omniORB NameService=corbaname::localhost
```

7.2 Exemple de configuration pour l'intégration de la machine BGP

Nous présentons ici, en tant qu'exemple, la configuration des différents éléments de DIET pour permettre l'intégration de la machine *BGP* à la hiérarchie DIET. On supposera que le Master Agent est lancé sur la machine *clai2q2*.

7.2.1 Configuration du MA

Le Master Agent, doit disposer d'un accès à la machine *BGP* et celle-ci doit pouvoir le contacter. Étant donné que la seule possibilité d'accéder à la machine *BGP* depuis la machine *clai2q2* (et vice-versa) est d'utiliser une connexion *ssh*, nous allons devoir configurer le Master Agent afin que celui-ci ouvre un tunnel vers *BGP* et se déclare, avec les bons paramètres d'accès sur cette machine.

Le fichier de configuration du Master Agent doit ainsi contenir les lignes suivantes :

```
convertiorPath = /root/softs/omniORB/bin/convertior  
sshProxies = /root/etc/sshProxies  
sshPath = /usr/bin/ssh
```

- Le paramètre `convertiorPath` lui fournit le chemin d'accès au programme `convertior` permettant de modifier les informations de l'IOR de l'agent afin que l'hôte et le port qu'il contient pointent sur le tunnel ssh côté *BGP*.
- Le paramètre `sshProxies` fournit un fichier de configuration des tunnels ssh.
- Le paramètre `sshPath` fournit quant à lui le chemin vers le client ssh à utiliser pour réaliser les connexions nécessaires.

Le fichier `sshProxies`, doit alors contenir la ligne suivante :

```
bgp localhost bgp le-mahec 22 /root/.ssh/id_rsa \  
/gpfs/home/le-mahec/softs/omniORB NameService=corbaname::localhost
```

Avec cette configuration, lorsque le MA sera lancé, celui-ci se connectera à la machine *bgp* en tant qu'utilisateur *le-mahec*, sur le port 22 en utilisant la clé privée `/root/.ssh/id_rsa`. Grâce à cette connexion, il crée alors un tunnel ssh, permettant de le contacter depuis *bgp* puis il effectue la modification de son IOR de manière à ce qu'il puisse se déclarer dans le serveur de nom *omniNames* de *bgp* avec un IOR adapté.

7.2.2 Configuration des SeDs

Le SeD lancé sur *bgp* a accès au MA grâce au tunnel ssh que celui-ci a créé. Cependant, il n'est lui-même pas joignable depuis *clai2q2*. Il faut donc le configurer de telle manière qu'il ouvre un tunnel ssh vers cette machine. Pour cela, son fichier de configuration devra contenir les lignes suivantes :

```
convertiorPath = /home/le-mahec/softs/omniORB/bin/convertior  
sshProxies = /home/le-mahec/etc/sshProxies  
sshPath = /usr/bin/ssh
```

Afin de lui préciser vers quelles machines le SeD doit ouvrir des tunnels ssh, on éditera le fichier `/home/le-mahec/etc/sshProxies`. Dans notre exemple de connexion vers *clawi2q2*, ce fichier devra contenir la ligne suivante :

```
clawi2q2 localhost clawi2q2 root 22 /home/le-mahec/.ssh/id_rsa \  
/root/softs/omniORB NameService=corbaname::localhost
```

Ainsi, le SeD, une fois lancé, ouvre une connexion ssh en tant qu'utilisateur `root` sur le port 22 avec la clé `/home/le-mahec/.ssh/id_rsa`. Il crée alors un tunnel vers *clawi2q2*, puis procède aux corrections de son IOR pour que celui-ci pointe sur le tunnel ssh côté *clawi2q2*.

7.2.3 Configuration du client

De la même manière, les applications clientes ont besoin de communiquer avec le SeD, mais celles-ci doivent également pouvoir être contactées par les SeDs choisis afin de transmettre les données de sortie du problème. La configuration du client est, du point de vue des paramètres nécessaires à la création des tunnels, identique à celle du MA. Le fichier de configuration du client contient alors les lignes :

```
convertiorPath = /root/softs/omniORB/bin/convertior  
sshProxies = /root/etc/sshProxies  
sshPath = /usr/bin/ssh
```

Et le fichier `sshProxies` contient la ligne suivante :

```
bgp localhost bgp le-mahec 22 /root/.ssh/id_rsa \  
/gpf/home/le-mahec/softs/omniORB NameService=corbaname::localhost
```

7.2.4 Lancement de la plate-forme

Le lancement de la plate-forme s'effectue alors de la manière suivante :

- Lancement des serveurs de noms *omniNames* sur les machines *bgp* et *clawi2q2*.
- Lancement du Master Agent sur *clawi2q2*.
- Lancement du SeD sur *bgp*.

À partir de ce moment, le SeD de *clawi2q2* est accessible depuis un client configuré comme indiqué précédemment sur *clawi2q2*.