UNIVERSITÉ PARIS SUD
FACULTÉ DES SCIENCES D'ORSAY

présentation de

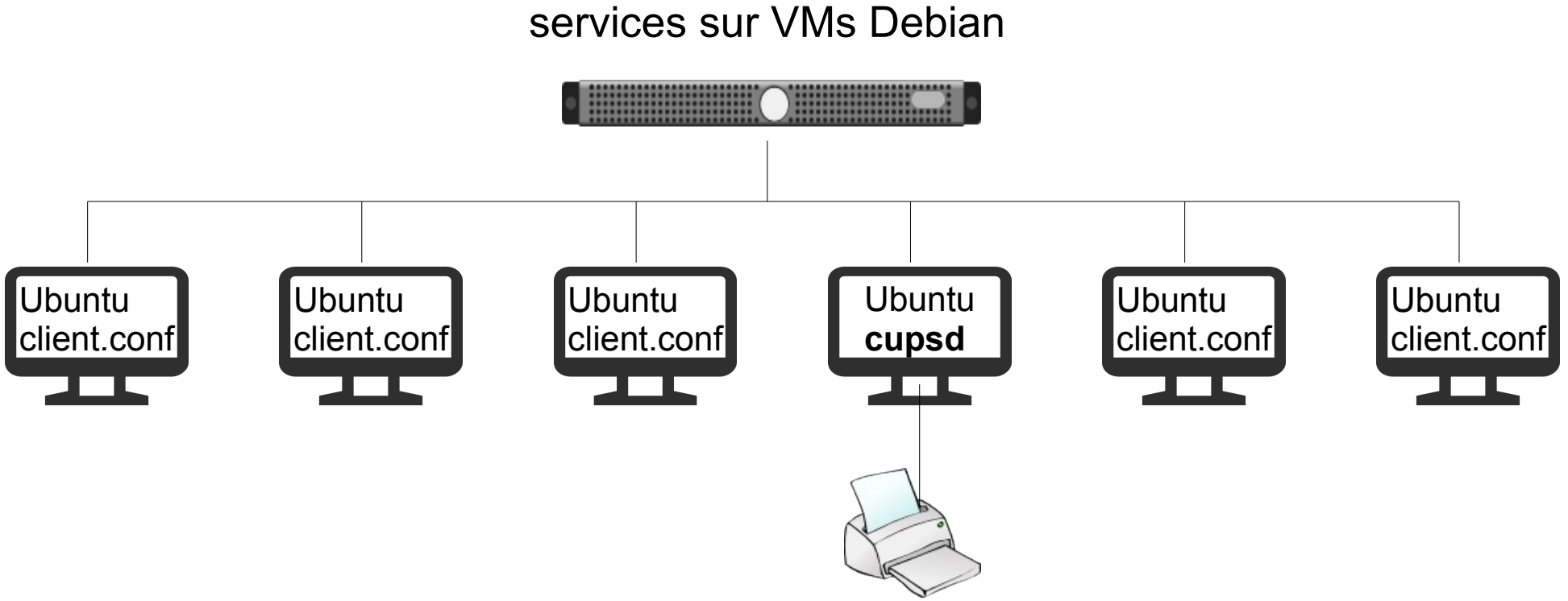
PyKota

service de quotas d'impressions

Journées Mathrice – octobre 2015 – Bures, Orsay

Olivier Chaudet

Contexte : salles de TP du LMO



Dans chaque salle, un poste de TP connecté à l'imprimante via USB ou port parallèle fait office de serveur d'impression. Le service CUPS est coupé sur les autres, qui ne reposent que sur l'instruction ServerName de leur fichier /etc/cups/client.conf

=> système simple qui fonctionne depuis longtemps, **mais...**

Limiter le gaspillage

Les impressions sont en libre service depuis des années, mais face aux coûts grandissants (et pas seulement financiers) que cela représente, on a envisagé divers types de solutions...

- Retirer les imprimantes, ne plus fournir de papier, désactiver le service d'impression en dehors des examens...
- Surveiller les logs pour remonter les bretelles aux indélicats
- Mettre en place un système de quotas d'impressions

Journal d'impressions de CUPS

Par défaut la journalisation des jobs d'impressions se trouve dans le fichier `/var/log/cups/page_log`, mais cela peut être modifié dans la configuration (`/etc/cups/cups-files.conf`). Les entrées de journal ont cette tête là :

```
imp33 user 3769 [06/Jan/2015:08:49:05 +0100] 2 1 - 192.168.31.85 exercices8.pdf - -
```

Où on a dans l'ordre :

- nom de l'imprimante
- login du propriétaire du job
- numéro du job
- date et heure
- numéro de la page
- nombre d'exemplaires de la page
- IPP job billing → chaîne de texte prévue pour simplifier la facturation (« - » par défaut)
- adresse de la machine émettrice du job
- nom du document
- champ IPP optionnel media (« - » par défaut)
- champ IPP optionnel sides (« - » par défaut)

Problèmes avec l'exploitation des journaux

1) Dans la configuration d'origine, il y a un serveur d'impression par salle.

=> rédaction de scripts pour centraliser et consolider les logs en un seul listing avec possibilité d'extraire un nombre de pages par imprimante ou par utilisateur.

2) Il arrive que dans les entrées du journal, le champ « numéro de page » soit remplacé par le mot clef **`total`**, auquel cas le champ « nombre d'exemplaire » devient un nombre total de pages.

=> ajout d'une condition dans les scripts.

3) Le journal inclut la plupart des jobs qui ont échoué, conduisant à des résultats parfois matériellement invraisemblables.

=> l'opération remontage de bretelles est quelque peu délicate, et je n'ai pas trouvé de solution à ce problème

4) Lorsqu'on réagit, c'est que le mal est déjà fait, au moins en partie.

=> on a décidé d'opter pour une cure préventive à base de quotas d'impressions

Quotas intégrés à CUPS

CUPS intègre un système de quotas avec les principales caractéristiques suivantes :

- Le quota est fixé en nombre de pages autorisées, avec une remise à zéro du compteur suivant une période paramétrable, exprimée en secondes.

- Ce quota est fixé par imprimante, mais il est global pour tous les comptes et groupes => pas possible de prévoir des besoins particuliers, ni de désactiver simplement le décompte pour la durée d'un examen, par exemple.

Question souplesse, ça laisse à désirer...

Autres systèmes de quotas

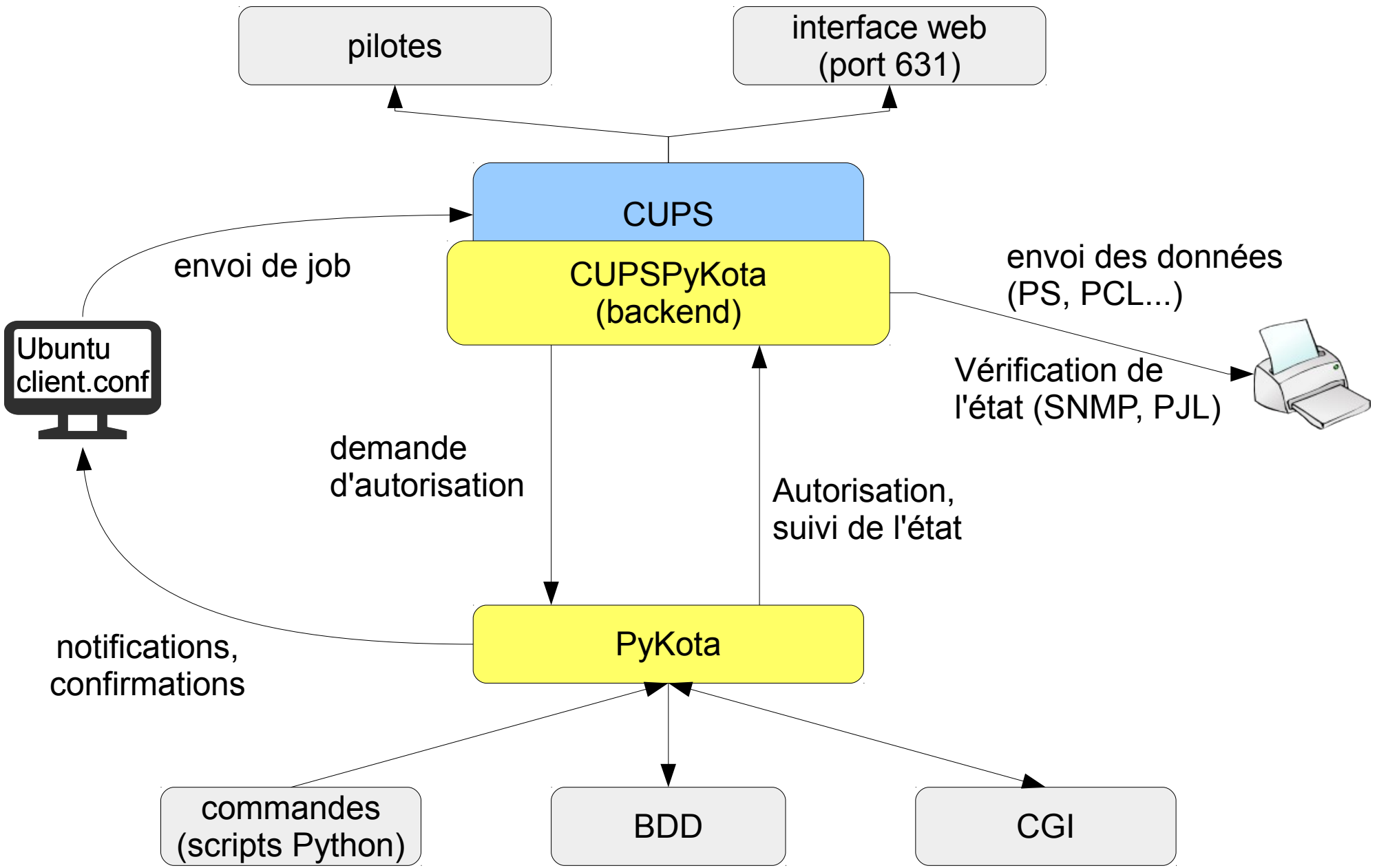
- Développé par l'entreprise australienne éponyme (environ 60 personnes)
- Licence propriétaire
- Multiplateforme
- Payant au nombre d'usagers (~550€ pour 1000), sans abonnement
- Mature (commercialisé depuis 1998)
- Supports technique et commercial actifs, mais mises à jour payantes
- Liste de fonctionnalités longue comme plusieurs bras

Je n'ai pas (encore) testé, donc voir : www.papercut.com

- Projet d'un seul homme (Jérôme Alet)
- Open source, licence libre (GPLv2)
- Fonctionne en interface avec CUPS
- Gratuit
- Utilisé depuis 2004, notamment par l'Université de la Nouvelle-Calédonie
- En jachère depuis quelques années faute de disponibilité de son développeur => **avis aux cracks de Python que ça intéresserait d'y participer**

Site officiel : http://www.pykota.com

Structure générale de PyKota



Prérequis : CUPS

PyKota s'appuie sur un serveur CUPS centralisé, au sens où la fiabilité du contrôle des imprimantes et du décompte des pages nécessite que chaque imprimante ne réponde qu'à un unique serveur.

PyKota et CUPS doivent être sur la même machine (un élément important de PyKota est en fait un filtre de CUPS : `cupspykota`).

Pour que PyKota fonctionne convenablement, il faut que CUPS transmette les informations relatives au propriétaire du job, car il est difficile d'imposer des quotas lorsque tous les documents appartiennent à `unknown`. Connaître la machine d'origine est aussi utile pour le feedback et les confirmations. Du coup, on doit avoir dans `/etc/cups/cupsd.conf` :

```
<Policy default>
  JobPrivateAccess all
  JobPrivateValues none
  ...
</Policy>
```

NB : si on ne veut pas que chacun puisse voir qui est en train d'imprimer quoi, on peut éventuellement restreindre le `JobPrivateAccess` :

```
...
  JobPrivateAccess @OWNER @SYSTEM
...
```

Installation

Récupérer l'archive de PyKota, ainsi que celles de pkipplib et pkpgcounter depuis le site officiel de PyKota : http://www.pykota.com/software/

La première contient notamment un script `checkdeps.py` qui peut nous renseigner sur la liste des dépendances manquantes sur le système (il faut au moins avoir python 2.x pour l'exécuter, mais c'est généralement le cas par défaut). En ce qui me concerne, sur une Debian 8.0, j'ai du installer depuis les dépôts :

```
python-pysnmp4 python-jaxml python-egenix-mxdatetime python-osd python-mysqldb
```

Les besoins précis dépendront notamment de la base de données sur laquelle on souhaite s'appuyer.

Une fois qu'on est bon pour les dépendances, il reste à lancer les scripts d'installation des bibliothèques de PyKota en exécutant dans le dossier de chacune :

```
python setup.py install
```

Utilisateur pykota

PyKota a besoin d'exister en tant qu'utilisateur système, avec pour `home` le dossier de configuration `/etc/pykota/`, dans lequel on copie les exemples avant de les éditer. Il est également impératif d'associer le backend `cupspykota` à CUPS.

```
adduser --system --group --home /etc/pykota --gecos PyKota pykota
adduser lp pykota

# Modèles de configuration
cp /usr/local/share/pykota/conf/pykota.conf.sample /etc/pykota/pykota.conf
cp /usr/local/share/pykota/conf/pykotadmin.conf.sample \
/etc/pykota/pykotadmin.conf

# droits sur les fichiers de conf
chown -R pykota.pykota /etc/pykota
chmod 755 /etc/pykota
chmod 644 /etc/pykota/pykota.conf
chmod 640 /etc/pykota/pykotadmin.conf

# backend pour CUPS
cp /usr/local/share/pykota/cupspykota /usr/lib/cups/backend/
chown root.root /usr/lib/cups/backend/cupspykota
Chmod 700 /usr/lib/cups/backend/cupspykota
```

À noter : CUPS doit pouvoir accéder à la configuration, on ajoute donc `lp` au groupe `pykota`

Base de données

PyKota est théoriquement capable de s'appuyer sur quatre types de bases de données, mais j'ai obtenu des résultats contrastés.

- **SQLite** : la configuration en a été simple, et au premier abord tout a semblé fonctionner comme attendu. Malheureusement, l'inscription des jobs dans la base de données et la mise à jour des quotas par le backend échoue parfois, alors même que les commandes manuelles continuent à fonctionner normalement.
- **PostGres** : il faut ajuster le script de création de la base PostGres fourni dans `/usr/local/share/pykota/mysql/` et beaucoup d'explications dans les fichiers `README` et `SECURITY` de l'archive PyKota.
- **LDAP** : il est possible d'utiliser un annuaire LDAP comme base de données, en spécifiant les branches où on souhaite enregistrer les différents éléments.
- **MySQL** : il faut éditer le script de création de la base de données qui est fourni avec PyKota (`/usr/local/share/pykota/mysql/pykota-mysql.sql`), pour y mettre à jour les mots de passe et aussi y remplacer la directive obsolète `TYPE=INNODB` par `ENGINE=INNODB`. Cela fait, tout va comme sur des roulettes.

Configuration de PyKota : pykotadmin.conf

La configuration de PyKota est hébergée dans deux fichiers abondamment commentés.

`/etc/pykota/pykotadmin.conf`

⚠ Ce fichier contient les informations relatives à l'administrateur de la base de données, **il ne doit être lisible que par `root`, `pykota` et `lp`.**

`/etc/pykota/pykota.conf`

La plupart des options de ce fichier y sont directement expliquées en détail, et certaines seront abordées dans la suite de la présentation. Mais il faut au minimum se préoccuper :

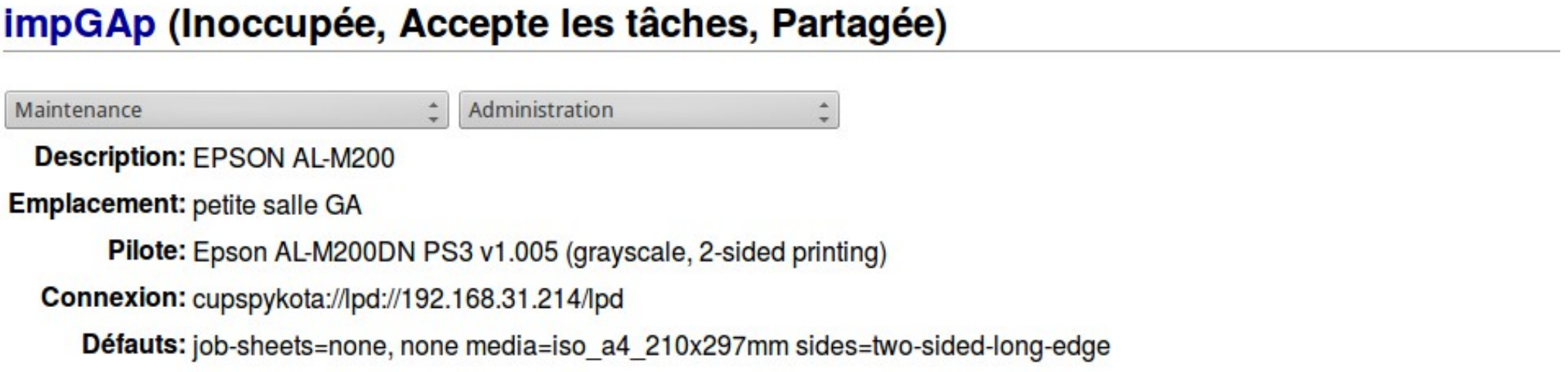
- des options de connexion utilisateur à la base de données choisie
- de la directive accounter qui spécifie comment déterminer le nombre de pages imprimées

Afin de ne plus avoir le problème de décompte des travaux échoués qui se posait avec CUPS, j'ai opté pour :

```
accounter: hardware(snmp)
```

Ajout d'imprimantes

D'abord, installer l'imprimante comme d'habitude dans CUPS, avec le pilote et le protocole voulus, puis indiquer au service qu'il faut utiliser le backend de PyKota en ajoutant `cupspykota:` ou `cupspykota://` au début de l'URL, ce qui peut aussi être réalisé automatiquement par l'option `--cups` de la commande `pkprinters`. Résultat dans CUPS :



Il faut ensuite ajouter cette imprimante dans la base de PyKota, à l'aide de la commande :

```
pkprinters -a -D "description intelligible" imprimanteCUPS
```

Le nom doit bien sûr être le même dans CUPS et PyKota. Si l'imprimante existe déjà dans PyKota, elle est modifiée à moins d'utiliser l'option `-s` ou `--skipexisting`

Gestion des imprimantes

Il est possible de créer des groupes d'imprimantes. En fait, du point de vue de PyKota, un tel groupe est considéré comme une imprimante mais n'est pas rattaché à une entrée de CUPS. Ainsi, on peut créer un groupe et y ajouter l'imprimante précédemment configurée :

```
pkprinters -a -D "Ceci est un groupe d'imprimantes." mathens
pkprinters -g mathens imprimanteCUPS
```

Et puis pour retirer une imprimante d'un groupe :

```
pkprinters -r -g mathens imprimanteCUPS
```

Voire la supprimer entièrement :

```
pkprinters -d imprimanteCUPS
```

Pour lister les statuts des imprimantes dont le nom commence par « imp » :

```
pkprinters -l imp*
```

C'est la seule commande pour laquelle l'option « list » utilise une minuscule, les autres emploient `-L`

Gestion des utilisateurs

On peut créer des groupes « admin », « profs » et « eleves » par la commande :

```
pkusers -a -g admins profs eleves
```

Puis y ajouter des utilisateurs, par exemple :

```
pkusers -a -i admins,profs -D "par ex. le gecos" jerome prof2 enseignant3
```

Comme pour les imprimantes, si un utilisateur ou un groupe existe déjà, il est modifié à moins d'employer l'option `-s`

Pour lister le statut des utilisateurs dont le login commence par « ch » :

```
pkusers -L ch*
```

Et puis pour retirer un utilisateur d'un groupe :

```
pkusers -r -i admins enseignant3
```

Voire le supprimer entièrement :

```
pkusers -d enseignant3
```

Quotas : principes

Il y a plusieurs façons d'appliquer les quotas, et il faut en choisir une par utilisateur, par le biais de l'option « limitby » (ou `-l`) de `pkusers`. Les valeurs possibles sont :

- **`quota`** : utilise des limites en nombres de pages (méthode par défaut)
- **`balance`** : utilise des crédits d'impression, reflétant par exemple une somme d'argent
- **`noquota`** : il n'y a pas de quota appliqué, mais les pages sont comptées
- **`nochange`** : pas de quota, pas de décompte, c'est open-bar
- **`noprint`** : pas d'impression, pas de quota

Dans tous les cas, la première chose à faire avant que des quotas puissent fonctionner est de créer une association entre l'utilisateur et chaque imprimante concernée (ce sont techniquement des entrées dans la table `userpquota`), pour cela on utilise la commande :

```
edpykota -a -P impCUPS,hp4999 jerome
```

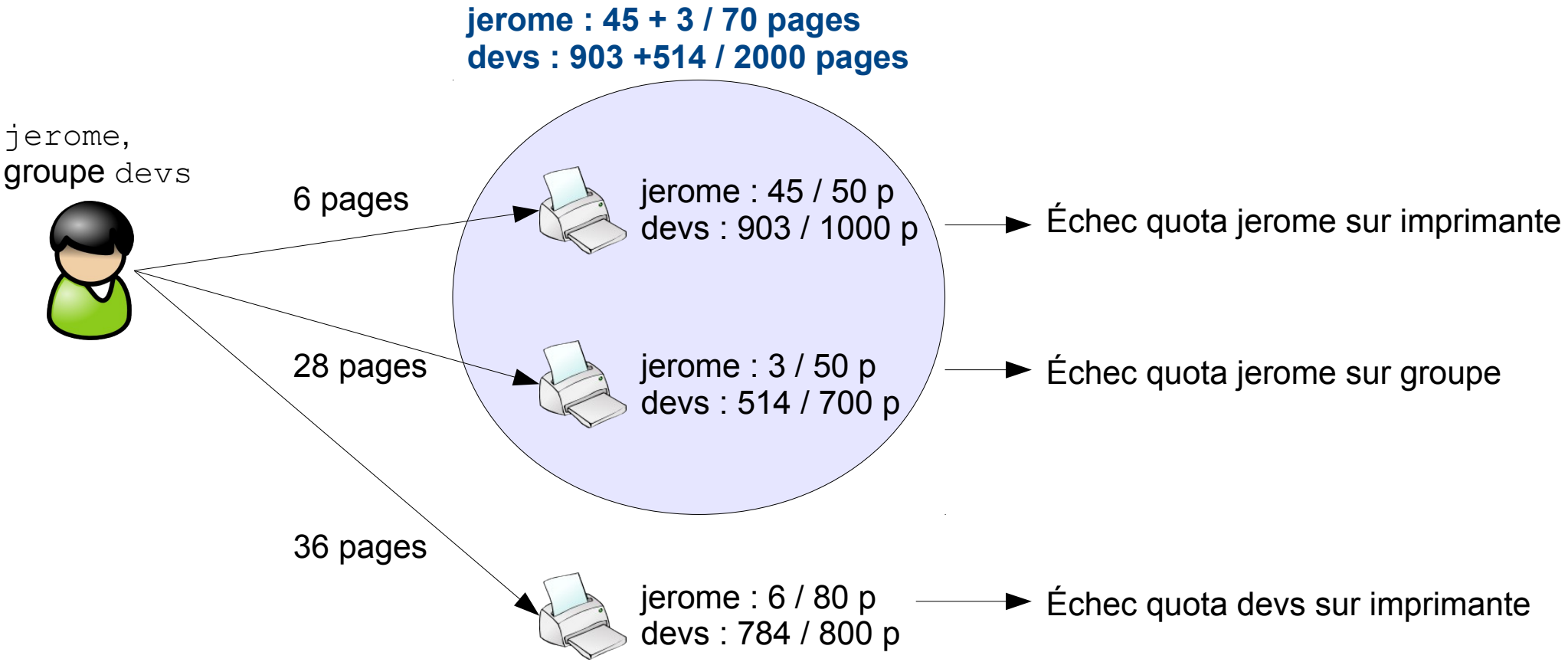
L'option -P peut être omise pour créer une association avec chaque imprimante existante (y compris les groupes).

⚠ Cela signifie qu'il faut aussi penser à créer ces associations lorsqu'on ajoute une nouvelle imprimante. Par exemple :

```
edpykota -a -P NouvelleImprimante *
```

Quotas : limitby quota (1)

Dans ce mode, le quota est fixé en nombre de pages pour un utilisateur ou un groupe donné sur une imprimante ou un groupe d'imprimante donné. Le décompte du nombre de pages imprimées pour un groupe (d'utilisateurs ou d'imprimantes) est la somme du nombre de pages imprimées par ses membres. Exemples :



Quotas : limitby quota (2)

En plus, pour chaque association utilisateur-imprimante, il peut y avoir deux limites : « soft » et « hard », suivant le même principe que les quotas disques sous Unix. Quand la limite « soft » est atteinte, on peut encore imprimer temporairement (durée précisée par l'option `gracedelay` du fichier de configuration de PyKota).

Exemples d'attribution de quotas :

```
# indiquer qu'on utilise le mode "quota" (qui est déjà par défaut) pour jerome
pkusers -l quota jerome
# limites soft et hard à 200 et 220 pages pour jerome sur les imprimantes h4 et imp
edpykota -a -S 200 -H 220 -P h4,imp jerome
# limtes à 120 et 200 pages pour toto sur toutes les imprimantes
edpykota -a -S 120 -H 200 toto
# limites à 800 et 1000 pages pour les groupes devs et admins
edpykota -a -S 800 -H 1000 -g devs admins
```

Pour remettre à 0 le compteur d'un utilisateur, sans supprimer son historique d'impressions :

```
edpykota -r -P h4,imp jerome
```

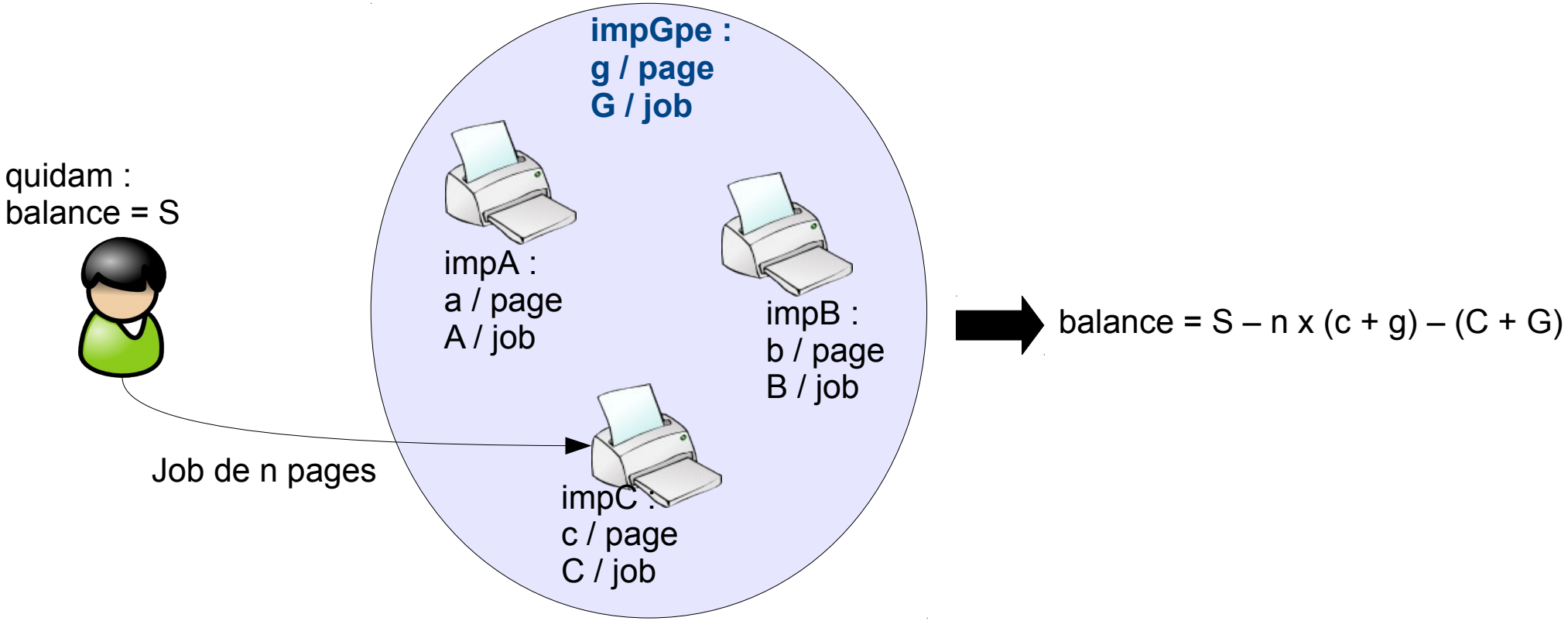
Pour supprimer une association entre un utilisateur et une imprimante (y compris l'historique) :

```
edpykota -d -P impA jerome
```

Quotas : limitby balance

Dans ce mode, l'utilisateur se voit attribuer un crédit (ou balance), pouvant éventuellement refléter une monnaie réelle. Chaque imprimante et groupe d'imprimante doit alors avoir un coût par page imprimée, et optionnellement aussi par job. Par exemple :

```
# Toutes les imprimantes ont un coût nul, mais le groupe impGpe a un coût de 1
pkprinters -c 0
pkprinters -c 1.0 impGpe
# quidam utilise un quota par balance et se voit attribuer 199,9 crédits
pkusers -l balance -b 199.9 quidam
```



Extraction d'information

Outre les options « list » des commandes précédentes, il y a un script dédié à la présentation des informations : `dumpykota`. Par défaut, le retour s'en fait au format CSV, mais il est aussi possible d'opter pour de l'xml (voir l'option `format`). Quelques exemples...

Historique des jobs d'un utilisateur au format CSV avec des tabulations :

```
dumpykota -d history -f tsv username="ilabusetrop"
```

Liste des utilisateurs dans l'ordre alphabétique, au format XML :

```
dumpykota -d users -f xml -O "+username"
```

Récapitulatif des quotas d'utilisateurs par compteur de pages décroissant, puis ordre alphabétique, pour un groupe d'imprimantes :

```
dumpykota -d upquotas -O "-pagecounter,+username" printername="GroupeImprimantes"
```

Historique des jobs du 11/09/2015 au 10/10/2015, au format `page_log` de CUPS :

```
dumpykota -d history -f cups  start=20150911 end=20151010
```

Notifications : côté serveur

Demandes de confirmation avant impression

Elles sont régies par l'option `askconfirmation`, qui exécute une commande dont le retour doit être `CANCEL` pour annuler le job, et n'importe quoi d'autre pour l'autoriser. Un tel script est proposé avec PyKota, qui est configuré ainsi pour les salles de TP :

```
askconfirmation: /usr/local/bin/pknotify \
-destination $PYKOTAJOBORIGINATINGHOSTNAME:7654 \
--timeout 120 --confirm "Message avec état du quota de l'utilisateur"
```

Alertes

Les messages d'alerte relatifs aux quotas sont spécifiés dans les options : `poorwarn`, `softwarn` et `hardwarn` du fichier de configuration, et la manière de les envoyer est fixée par l'option `mailto`. On peut soit les envoyer sous forme de méls aux adresses enregistrées pour les utilisateurs, soit faire appel à une commande, par exemple ce même `pknotify` :

```
mailto : external(/usr/local/bin/pknotify \
--destination $PYKOTAJOBORIGINATINGHOSTNAME:7654 --notify "%(message)s" \
2>&1 > /dev/null)
```

Notifications : côté client

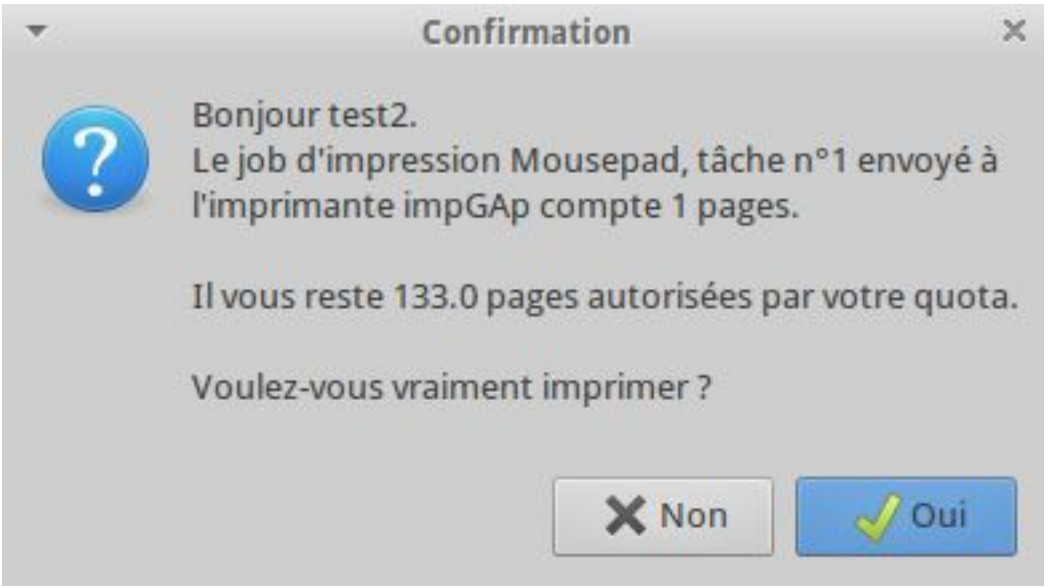
C'est bien d'envoyer des requêtes RPC depuis le serveur, encore faut-il que quelque chose soit à l'écoute sur le client. PyKota s'accompagne de quelque chose de ce genre : PyKotIcon, un service XML-RPC disponible comme binaire Windows ou en code source.

⚠ Sous Linux, PyKotIcon nécessite wx v3 pour fonctionner correctement. Sous Ubuntu, cette bibliothèque est disponible à partir de Vivid Vervet (paquets `libwxbase3.0-0`, `libwxgtk3.0-0` et `python-wxgtk3.0`). Il doit aussi être démarré à l'ouverture de session, par exemple via un lanceur dans `/etc/xdg/autostart/` appelant la commande :

```
/usr/local/bin/pykoticon –port=7654 192.168.31.9
```

Si tout se passe bien, on voit ceci dans la zone de notification : 

Et au moment d'imprimer :



CGI

PyKota propose aussi des pages CGI qui permettent d'obtenir via une interface web l'appel à `dumpykota`, les rapports sur les quotas ou, en fonction de l'authentification, le statut personnel du quota. Ces pages ont besoin d'une installation PyKota fonctionnelle, mais pas nécessairement sur le serveur d'impression.

Ci-contre, un exemple de rapport.

On peut les voir sur le site officiel :
http://www.pykota.com/wiki/WebInterface

Rapports PyKota

PyKota v1.27alpha5_unofficial

Rapport

Merci de cliquer sur le bouton ci-dessus

Imprimante : HP2100 () / EPSDX6050 () / PDFGen () / TeeCatcher ()

Masque Utilisateur / Groupe : * *e.g. jo**

Rapport pour les Groupes :

Rapport pour le quota user sur l'imprimante PDFGen ()

Délai de grâce: 7 jours

Prix par job: 0.000

Prix par page: 0.000

Utilisateur	LimitBy	surtaxe	util	soft	hard	balance	grace	total	payé	warn
administrateur	NQ	1.0	0	None	None	-0.70		0	0.00	0
ambre	NQ	1.0	0	None	None	-0.60		0	0.00	0
fleur	NQ	1.0	0	None	None	0.00		0	0.00	0
guest	NQ	1.0	0	None	None	0.00		0	0.00	0
jerome	-B	1.0	0	None	None	990.20		0	1006.50	0
rachel	NQ	1.0	0	None	None	-53.40		0	0.00	0
remroot	NQ	1.0	0	None	None	0.00		0	0.00	0
root	NQ	0.5	0	None	None	0.00		0	0.00	0
								Total : 0	1006.50	
								Réel : inconnu		

Rapport

MERCI

À l'assistance, qui est présente de si bon matin.

À Adrien, qui m'a gentiment notifié mon volontariat pour cette présentation.

À Jérôme Alet, qui a bien voulu répondre à mes questions et vérifier que ce document ne contenait pas trop de bêtises.