Présentation des scripts du Cr@ns

Yann Duplouy

Cachan Réseau à Normale Sup'

Mardi 29 novembre 2011

Introduction

Pourquoi un séminaire sur les «scripts du Cr@ns» ?

- ils permettent de simplifier des tâches qui autrement seraient lourdes
- et il faut donc les maintenir pour qu'ils continuent à être à jour avec les modifications du Cr@ns

Les scripts sont accessibles de la majorité des serveurs dans `/usr/scripts/` grâce au protocole NFS (ils sont stockés sur `fx`)
Leur mise à jour est gérée via `darcs` et `git`.

Sommaire

1 Structure des scripts

2 Scripts de gestion du Cr@ns et des comptes Cr@ns

3 Scripts pour les impression

Structure de `/usr/scripts/`

Les scripts y sont gérés par répertoires :

- Dans `gestion`, on retrouve tout ce qui est nécessaire pour la gestion du Cr@ns : `gest_crans.py`, `config.py`, ... que l'on détaillera plus tard
- Dans `impression`, on retrouve les scripts gérant le digicode, et ceux qui traduisent les paramètres rentrés sur l'intranet en paramètres pour l'imprimante.
- Dans `admin`, on a quelques scripts ayant pour but de faciliter les tâches de membres du CA (en particulier le contrôle des cartes d'étudiants et paiements)

Le dossier `respbats` sert de raccourcis pour certains des scripts. Il est contenu dans le `PATH` des membres actifs. Cette liste n'est pas exhaustive

Au début des scripts

Les scripts doivent fonctionner sur l'ensemble des machines, qui ne sont généralement configurées que pour faire apparaître `/usr/scripts` :

- Utilisation de `sys.path.append` nécessaire pour accéder rapidement aux autres scripts
- en particulier ce qui vient des autres scripts : `ldap_crans` (pour l'annuaire LDAP), `affich_tools` (gestion de l'encodage, des codes couleurs...)

Sommaire

1. Structure des scripts
2. Scripts de gestion du Cr@ns et des comptes Cr@ns
3. Scripts pour les impression

ldap_crans.py

Script dont les classes sont importées dans la plupart des autres scripts. Il permet l'accès à l'annuaire LDAP, en lecture et écriture :

- L'accès se fait avec des objets `CransLdap`
- Il permet de gérer les diverses données écrites dans l'annauire, chacune en les représentant avec des classes adaptées :
  - `Adherent`
  - `MachineFixe`
  - `MachineWifi`
  - `Club`
  - `Service`

  La gestion des modifications est effectuée par ces objets

annuaires.py

Ce «script» est importé par d'autres scripts pour simplifier la gestion des prises, des switchs et des locaux :

- `chbre_prises` relie les numéros de prises aux chambres correspondantes
- `uplink_prises` concerne les prises internes à l'infractruscture du Cr@ns
- `reverse(bat)` permet de relier les numéros de chambres à leur prises correspondantes
- `all_switchs([bat])` permet d'obtenir la liste des switchs d'un bâtiment en particulier ou de tous les bâtiments
- `locaux_clubs()` donne la correspondance local associatif - club correspondant, par une structure de dictionnaire

config.py

Dans ce «script», on retrouve des informations de configurations diverses, concernant les limites d'adhésion, d'upload, ou des configurations techniques pour les VLan. On y trouve en particulier :

- Les dates des blocages administratifs
- Les coûts des impressions
- Les paramètres de contrôle d'upload
- Des mails-types pour les adhésions, les déconnexions, la prévention...
- La correspondance nom VLan-numéro VLan

chgpass.py

Ce script permet pour un câbleur de changer le mot de passe d'un adhérent ou d'un club.

- Il s'utilise de la manière suivante : `chgpass <login>` et demandera le mot de passe du câbleur connecté, pour modifier le mot de passe de *login*
- Son utilisation est limitée : si on veut changer le mot de passe d'un adhérent ayant des droits, il faut posséder les droits du superutilisateur.

whos.py

`whos` permet de faire des recherches rapides dans l'annuaire LDAP. Il permet d'avoir accès à :

- diverses informations sur les adhérents (tapées à son adhésion ou à sa réadhésion) dont ses machines
- des informations sur les machines : leur adresse MAC, l'IP attribuée...) et l'IPsec des machines wifi avec l'option `-i`
- des informations sur les clubs : leurs machines, leurs responsables, leurs imprimeurs...
- des informations sur les bornes

Il permet l'accès à des données potentiellement sensibles : son utilisation est donc surveillée.

ressuscite.py

Ce script permet de récuperer les données d'anciens adhérents qui viendraient réadhérer après au moins une année d'absence. On peut remettre dans l'annuaire LDAP, à l'aide de ce script :

- les données d'une ancienne machine d'un adhérent
- les données d'une machine à partir de sa date de suppression
- les données d'une machine ou d'un adhérent à partir d'un fichier se trouvant dans le "cimetière" (`/home/cimetiere`)

gest_crans.py

C'est le script le plus lancé, sous le nom `crans`. Il permet aux membres actifs de faire tout un tas d'opérations :

- inscrire un nouvel adhérent ou modifier l'inscription d'un adhérent, ce qui inclus le rechargement de son compte impression, ou la gestion de ses alias mail...
- gérer les machines d'un club ou d'un adhérent
- gérer les adhésions des clubs

Comme `ressuscite`, il utilise un outil d'interface graphique en mode texte, via la bibliothèque `Dialog`. Par ailleurs, il vérifie les droits de l'utilisateur l'appelant pour restreindre les options auxquel il a accès.

Sommaire

1. Structure des scripts
2. Scripts de gestion du Cr@ns et des comptes Cr@ns
3. Scripts pour les impression

impression_canon.py

Ce script permet la traduction des options choisies sur l'interface web en un appel à CUPS permettant de lancer l'impression réelle. Il s'occupe aussi du calcul du coût de l'impression.

- La fonction `_exec_imprime` va lancer l'impression réelle en créeant la ligne de commande d'appel à `lpr` avec les options correspondantes
- La fonction `_calcule_prix` calcule le prix de l'impression en se servant des paramètres indiqués dans `config.py`

Il est importé par les scripts de l'intranet

digicode.py

Le script `digicode` gère l'utilisation des digicodes, qui sont stockés dans `/var/impression/codes/`. Il peut générer des codes avec `gen_code` (en tirant des nombres au hasard entre 100000 et 999999 et en vérifiant s'ils ne sont pas déjà utilisés) Il peut les lister, en précisant leur âge et en regardant leur contenu qui donne l'utilisateur pour lequel ils ont été générés Il s'occupe aussi de faire le ménage, en supprimant les codes non utilisés existant depuis 24h.
Le script `gen_code.py` n'a comme seule fonction de l'importer, puis d'utiliser la fonction de même nom.

Pour conclure...

La liste des scripts présentés dans ce séminaire n'est évidemment pas exhaustive : on s'est restreint à ceux que la plupart des membres actifs utilisent, directement ou indirectement.

Des Questions ?