Services d'impression, Partie 1: Berkeley LPD et LPRng

Sylvain Ferrand

Journées Mathrice, Paris IHP, 18-20 Octobre 2005

Pour quoi faire?

- Gèrer une queue d'impression
  - Gestion d'un spool sur le disque
  - Permettre des interventions sur la queue
- Rendre accessible des imprimantes sur un réseau

Pour quoi faire?

- Gèrer une queue d'impression
  - Gestion d'un spool sur le disque
  - Permettre des interventions sur la queue
- Rendre accessible des imprimantes sur un réseau
  $\Rightarrow$ Fonctions définits dans la RFC 1179 "Line printer daemon protocol"

Pour quoi faire?

- Gèrer une queue d'impression
  - Gestion d'un spool sur le disque
  - Permettre des interventions sur la queue
- Rendre accessible des imprimantes sur un réseau
  $\Rightarrow$ Fonctions définits dans la RFC 1179 "Line printer daemon protocol"
- Gestion de quota, redirections, filtre, Pilote d'imprimantes ...

Historique

- Berkeley LPD (BSD)

Historique

- Berkeley LPD (BSD)
- AT/T LP (System 5) (proprietaire?)

Historique

- Berkeley LPD (BSD)
- AT/T LP (System 5) (proprietaire?)
- PLP (1986) (BSD)

Historique

- Berkeley LPD (BSD)
- AT/T LP (System 5) (proprietaire?)
- PLP (1986) (BSD)
- LPRng (1992) (GPL)

Historique

- Berkeley LPD (BSD)
- AT/T LP (System 5) (proprietaire?)
- PLP (1986) (BSD)
- LPRng (1992) (GPL)
- Cups (V1.0 1999) (GPL et LGPL)

Berkeley LPD

- Conçu dans les années 70

Berkeley LPD

- Conçu dans les années 70
- Initialement dévellopé pour les imprimantes locales et textes

Berkeley LPD

- Conçu dans les années 70
- Initialement dévellopé pour les imprimantes locales et textes
- Implemente la RFC 1179 et pas plus

Berkeley LPD

- Conçu dans les années 70
- Initialement dévellopé pour les imprimantes locales et textes
- Implemente la RFC 1179 et pas plus
- Fichier de configuration (printcap) peut lisible et rigide

Berkeley LPD

- Conçu dans les années 70
- Initialement dévellopé pour les imprimantes locales et textes
- Implemente la RFC 1179 et pas plus
- Fichier de configuration (printcap) peut lisible et rigide
- Quatre commandes utilisateur:
  - *lpr* Assigne un job
  - *lpq* Affiche l'état de la queue
  - *lprm* Détruit un job
  - *lpc* Gestion de la queue (très limité)

Berkeley LPD

- Conçu dans les années 70
- Initialement dévellopé pour les imprimantes locales et textes
- Implemente la RFC 1179 et pas plus
- Fichier de configuration (printcap) peut lisible et rigide
- Quatre commandes utilisateur:
  - *lpr* Assigne un job
  - *lpq* Affiche l'état de la queue
  - *lprm* Détruit un job
  - *lpc* Gestion de la queue (très limité)

$\Rightarrow$ Aujourd'hui obsolète

Les dérivés

- PLP - Hack de Berkeley LPD, développé jusqu'a 1992
- LPRng - Essentiellement une réimplementation de PLP en GPL

Les améliorations apportées par LPRng

- Gestion de queue améliorée avec le nouveau *lpc*

Les améliorations apportées par LPRng

- Gestion de queue améliorée avec le nouveau *lpc*
  - Passer des job en tête de queue

Les améliorations apportées par LPRng

- Gestion de queue améliorée avec le nouveau *lpc*
  - Passer des job en tête de queue
  - rediriger des jobs vers une autre queue

Les améliorations apportées par LPRng

- Gestion de queue améliorée avec le nouveau *lpc*
  - Passer des job en tête de queue
  - rediriger des jobs vers une autre queue
- Sécurité améliorée

Les améliorations apportées par LPRng

- Gestion de queue améliorée avec le nouveau *lpc*
  - Passer des job en tête de queue
  - rediriger des jobs vers une autre queue
- Sécurité améliorée
- Meilleur gestion des permissions et de l'authentification (Kerberos, SSL)

Les améliorations apportées par LPRng

- Gestion de queue améliorée avec le nouveau *lpc*
  - Passer des job en tête de queue
  - rediriger des jobs vers une autre queue
- Sécurité améliorée
- Meilleur gestion des permissions et de l'authentification (Kerberos, SSL)
- Logging, quota

Les améliorations apportées par LPRng

- Gestion de queue améliorée avec le nouveau *lpc*
  - Passer des job en tête de queue
  - rediriger des jobs vers une autre queue
- Sécurité améliorée
- Meilleur gestion des permissions et de l'authentification (Kerberos, SSL)
- Logging, quota
- Fichier de configuration moins rigide mais compatible

Les améliorations apportées par LPRng

- Gestion de queue améliorée avec le nouveau *lpc*
  - Passer des job en tête de queue
  - rediriger des jobs vers une autre queue
- Sécurité améliorée
- Meilleur gestion des permissions et de l'authentification (Kerberos, SSL)
- Logging, quota
- Fichier de configuration moins rigide mais compatible
- Fournit avec des filtres

Les améliorations apportées par LPRng

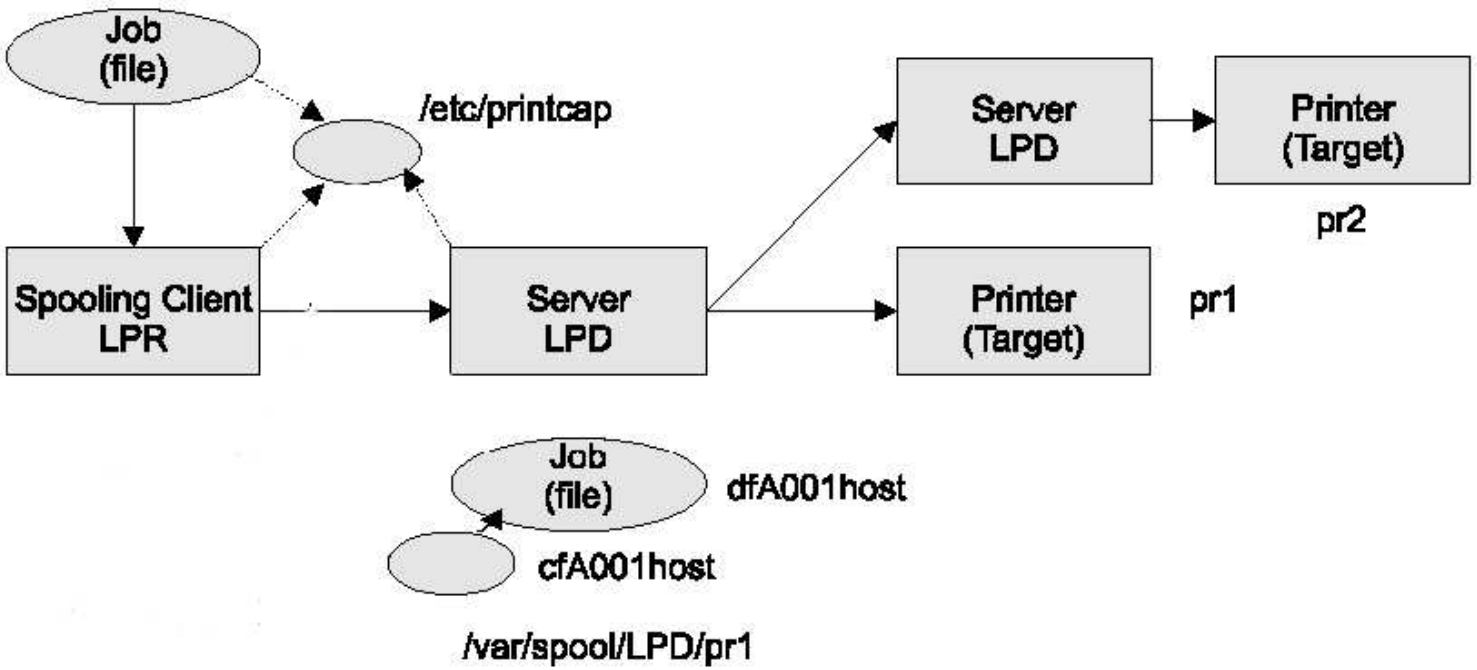
- Gestion de queue améliorée avec le nouveau *lpc*
  - Passer des job en tête de queue
  - rediriger des jobs vers une autre queue
- Sécurité améliorée
- Meilleur gestion des permissions et de l'authentification (Kerberos, SSL)
- Logging, quota
- Fichier de configuration moins rigide mais compatible
- Fournit avec des filtres
- Emulation du LP System V

Les améliorations apportées par LPRng

- Gestion de queue améliorée avec le nouveau *lpc*
  - Passer des job en tête de queue
  - rediriger des jobs vers une autre queue
- Sécurité améliorée
- Meilleur gestion des permissions et de l'authentification (Kerberos, SSL)
- Logging, quota
- Fichier de configuration moins rigide mais compatible
- Fournit avec des filtres
- Emulation du LP System V

Mais compatibilité conservée avec la RFC 1179

Achitecture Berkeley LPD/RFC1179



lpd est le daemon d'impression, il utilise le port 515 pour communiquer avec lpq, lpr, lprm. Les clients se connectent via des ports source < 1024.

Fichiers de configuration

- /etc/lpd.conf
  - Definit les chemins vers les différents fichiers de logs et de conf
- /etc/printcap
  - Définit les queues et leur caractéristiques
- /etc/lpd.perm
  - Définit les authorisations pour des machines, des utilisateurs ou des groupes

printcap

Sert a définir les queues et leur propriétées

- Format très rigide sous Berkeley BSD
  - attention au retours à la ligne, aux espaces...
  - Tags de 2 caractères max -> peu lisible

  $\Rightarrow$ Des simplifications mais compatibilités ascendante avec LPRng

printcap - Coté client

Il suffit de connaitre le nom des queues et du serveur:

```
# Un exemple pour une imprimante nomee hp ou hp2eme
hp|hp2eme:lp=NomDeLaQueue@serveur
```

printcap - Coté serveur

Un peu plus complex, mais pas tant que ca.
La syntaxe est basée sur le format termcap

```
lprv:
    :server:
    :sd=/var/spool/lpd/lprv:
    :if=/var/spool/lpd/filtres/filter:
    :mx#0:
    :sh:
    :rm=lprv:
```

Quelques aspects du protocole LPD

Le client doit transferer un fichier de control et un fichier de donnée au serveur.
Le fichier de control est transferé en ASCII (fin de ligne \n)
Exemple:

```
Hmachine.polytechnique.fr          Host name
J(stdin)                           Job title
Lroot                              User name
fdfA001machine.polytechnique.fr    Datafile
N(stdin)                           Data file Name
...
```

Le fichier de control cfA001machine.polytechnique.fr et le fichier de donnée associé dfA001machine.polytechnique.fr sont copiés dans le repertoire de spool du serveur.

Quelques aspects du protocole LPD - suite

Le protocole définit par la RFC 1179 est très simple mais il y a des lacunes

Quelques aspects du protocole LPD - suite

Le protocole définit par la RFC 1179 est très simple mais il y a des lacunes

- Problèmes d'incompatibilité entre les clients et les serveurs.

Quelques aspects du protocole LPD - suite

Le protocole définit par la RFC 1179 est très simple mais il y a des lacunes

- Problèmes d'incompatibilité entre les clients et les serveurs.
- Problèmes d'authentification et donc de sécurité

Quelques aspects du protocole LPD - suite

Le protocole définit par la RFC 1179 est très simple mais il y a des lacunes

- Problèmes d'incompatibilité entre les clients et les serveurs.
- Problèmes d'authentification et donc de sécurité
- Pratiquement rien de prévu pour gérer la queue

Quelques aspects du protocole LPD - suite

Le protocole définit par la RFC 1179 est très simple mais il y a des lacunes

- Problèmes d'incompatibilité entre les clients et les serveurs.
- Problèmes d'authentification et donc de sécurité
- Pratiquement rien de prévu pour gérer la queue

$\Rightarrow$ introduit déja quelque extention

Quelques aspects du protocole LPD - suite

Le protocole définit par la RFC 1179 est très simple mais il y a des lacunes

- Problèmes d'incompatibilité entre les clients et les serveurs.
- Problèmes d'authentification et donc de sécurité
- Pratiquement rien de prévu pour gérer la queue

$\Rightarrow$ introduit déja quelque extention
$\Rightarrow$ Nouveau protocole IPP pour combler ces lacunes

Les filtres existants

Différent filtres existent déja:

- psfilter
- apsfilter
  - Détection automatique du type de donnée (image, texte...) et conversion en PS.
- LPRng ifhp filter

Créer ses propres filtres

Il faut écrire un programme qui lit l'entrée standard et qui écrit sur la sortie standard.
Pratiquement les filtres existants (comme apsfilter) savent très bien reconnaitre un type de fichier pour le transformer en postscript avec le bon utilitaire.
Il peut etre utile de rajouter des fonctionnalités maison, par exemple:

- vérifier que les imprimantes sont up et envoyer un mail si ce n'est pas le cas
- logger tout un tas de chose...
- bricoler le postscript (gérer le duplex...)

Structure d'un filtre

- Récuperer les arguments passés en ligne de commande
- reconaitre le type de donnée (un bete `file`)
- `if type_de_fichier -> appeler tel utilitaire de conversion...`
- faire tout les traitements additionnels nécessaire
- balancer le résultat sur la sortie standard

Conclusion de la premiere partie

- Berkeley LPD obsolète
- LPRng n'est pas mort, même si son développement est très ralentit (dernière version octobre 2004)
- Coté filtre, ASPFILTER semble encore etre en développement actif

LPRng+ASPFILTER, très robuste et efficace ... mais peut-etre y a-t-il a plus moderne? :)