

Pykota Installation

Installing the base system

- Install base system of Debian Etch (Sarge does not use a new enough version of CUPS, etc. to provide full functionality)

Installing and configuring Samba/Winbind

- Install Samba and Winbind, then configure `/etc/samba/smb.conf` to be similar to below:

```
[global]
    idmap gid = 10000-20000
    obey pam restrictions = yes
    dns proxy = no
    netbios name = <hostname here>
    printing = cups
    invalid users = root
    idmap uid = 10000-20000
    workgroup = <domain here>
    os level = 20
    printcap name = cups
    security = ads
    max log size = 1000
    winbind separator = \
    log file = /var/log/samba/log.%m
    load printers = yes
    socket options = TCP_NODELAY
    wins server = <IP Address of WINS Server>
    encrypt passwords = true
    public = yes
    realm = <FQDN IN CAPS>
    winbind use default domain = yes
    wins proxy = no
    server string = Cups Server
    winbind enum users = yes
    password server = *
    winbind gid = 10000-20000
    winbind enum groups = yes
    syslog = 0
    preferred master = no
    panic action = /usr/share/samba/panic-action %d

[printers]
    printable = yes
    writable = no
    path = /var/spool/samba
    comment = All Printers
    public = yes
    create mode = 0700
    printer admin = @"domain admins"

[print$]
    comment = Printer Drivers
    path = /etc/samba/drivers
    browseable = yes
    read only = no
    guest ok = yes
    write list = administrator, @"domain admins"
```

- Install krb5-config and krb5-user then configure /etc/krb5.conf similar to below:

```
[realms]
    <FQDN IN CAPS> = {
        kdc = <PDC IN CAPS>
        kdc = <BDC IN CAPS>
        admin_server = <PDC IN CAPS>
    }
```

- Join server to the domain with:

```
# net ads join -U administrator
```

- The /etc/nsswitch.conf file should contain the following lines:

```
passwd:      files winbind
group:       files winbind
shadow:      files winbind
```

- At this point, the configuration can be tested using the command:

```
# wbinfo -u
```

- If Samba and Winbind are working correctly at this point, a list of domain users should be displayed

Installing/configuring CUPS

- Install the CUPS packages (including the Foomatic printer drivers for CUPS – the full package makes life easier when using the web interface)
- Edit the /etc/cups/cupsd.conf file to allow remote access to the necessary users
- It should now be possible to add printers to CUPS using the web interface at <http://<cups server>:631>
- It is advisable to use the Postscript drivers where possible so that printing is as quick as possible
- Older and more basic JetDirect interfaces also seem to print quicker using the socket 9100 connection method
- Download the CUPS Driver for Windows from <http://www.cups.org/windows/index.php>
- Extract the following files from the CUPS Driver package to /usr/share/cups/drivers:
 - cups6.inf
 - cups6.ini
 - cups6.ppd
 - cupsps6.dll
 - cupsui6.dll
- Copy the following files from a Windows installation to /usr/share/cups/drivers:
 - ps5ui.dll
 - pscript.hlp
 - pscript.ntf
 - pscript5.dll
- Ensure all the files within the directory /usr/share/cups/drivers have **lower case filenames**
- Change the ownership of /etc/samba/drivers using the following command:

```
# chown -R administrator:"domain admins" /etc/samba/drivers/
```

- Run the following command to export the drivers for all the existing printers in CUPS:

```
# cupsaddsmb -U administrator -a
```

- To test that this has worked (assuming that there are no error messages), browse to the CUPS server in Windows and try to connect to a printer

Installing/Configuring Pykota

- Download the Debian Etch package for Pykota
- Install the package using the following command:

```
# dpkg -i <filename>
```

- After resolving any dependency issues, it may be necessary to run the command again
- Run the following command:

```
# pksetup debian
```

- Follow the on-screen instructions to configure Pykota
- At this point, it may be necessary to edit the /etc/pykota/pykota.conf file so that Pykota looks for 127.0.0.1 instead of localhost for the SQL server:

```
storageserver : 127.0.0.1
```

- Check that BOTH the symlink in /usr/lib/cups/backend AND the original file /usr/share/pykota/cupspykopta have the execute bit set and restart the cupsysd service
- Also check the permissions on the cupspykota file

```
# chown root.root /usr/share/pykota/cupspykota
# chmod 700 /usr/share/pykota/cupspykota
```

- Check that the Debian package 'pysnmp2' is NOT installed as this is known not to work with Pykota. A working version should have been installed by pksetup
- Add a printer via the web interface in CUPS – the device type should be of type 'PyKota managed....'
- Add the following lines to /etc/pykota/pykota.conf (software accounting is used because the hardware accounting increases the time taken to print the first page)

```
[CupsPrinterName]
preaccounter : software()
accounter : software()
```

- Add accounting options to the printer using pkprinters – e.g. A cost of 1p per page

```
# pkprinters -a -c 0.01 CupsPrinterName
```

- Add a user to PyKota using the pkusers command – for example Joey who will be charged for printing and given an initial allowance of £2

```
# pkusers -a -l balance -b 2 joey
```

- Print to the published printer (either via Samba or through Linux) as Joey and the job should successfully flow through CUPS
- To check that the accounting is working, first check in the CUPS web interface that the job has logged the number of pages correctly, then run the following commands to display the user's balance and the printer's accounting details

```
# pkusers -L joey
# repykota -P CupsPrinterName
```

- To use automatic addition of users, edit /etc/pykota/pykota.conf so that the 'policy:' line reads the following (the default does not work correctly due to the description not having quotes around it!)

```
policy : external(pkusers --add --skipexisting --limitby noquota
$PYKOTAUSERNAME && edpykota --add --skipexisting --printer $PYKOTAPRINTERNAME
$PYKOTAUSERNAME)
```

- Due to the way that I want to charge, a script is needed to handle the creation of new users. I have placed the script in /usr/share/pykota and it is called bainesNewUser.sh:

```
#!/bin/bash

# Script to create new users for PyKota and assign initial
# balance.
# Script should be specified in the 'policy' section of
# pykota.conf

# Variable assignment
PYKOTAUSERNAME=$1
PYKOTAPRINTERNAME=$2
INITIALBALANCE=2

# If a pupil username limit by balance and assign initial quota
```

```

if echo "$PYKOTAUSERNAME" | grep -q ^[0-9]
    then pkusers -a -s -l balance -b $INITIALBALANCE $PYKOTAUSERNAME &&
    edpykota -a -s -P $PYKOTAPRINTERNAME $PYKOTAUSERNAME

# If not a pupil, must be a staff member or test account so do
# not limit printing
    else pkusers -a -s -l noquota $PYKOTAUSERNAME && edpykota -a -s -P
$PYKOTAPRINTERNAME $PYKOTAUSERNAME
fi

exit

```

- To run the script, alter the policy line in `pykota.conf` so that it reads:

```

policy : external(/usr/share/pykota/bainesNewUser.sh $PYKOTAUSERNAME
$PYKOTAPRINTERNAME)

```

- To allow use of the CGI web pages, a new website configuration file named `pykota` was made and added to `/etc/apache2/sites-enabled/`:

```

NameVirtualHost *
<VirtualHost *>
    ServerAdmin webmaster@localhost

    DocumentRoot /usr/share/pykota/stylesheets
    <Directory />
        Options FollowSymLinks
        AllowOverride None
    </Directory>
    <Directory /usr/share/pykota/stylesheets/>
        Options Indexes FollowSymLinks MultiViews
        AllowOverride None
        Order allow,deny
        allow from all
        # This directive allows us to have pretty PyKota CGIs
        RedirectMatch ^/$ /usr/share/pykota/stylesheets/
    </Directory>

    ScriptAlias /cgi-bin/ /usr/share/pykota/cgi-bin/
    <Directory "/usr/share/pykota/cgi-bin">
        AllowOverride None
        Options ExecCGI -MultiViews +SymLinksIfOwnerMatch
        Order allow,deny
        Allow from all
    </Directory>

    ErrorLog /var/log/apache2/error.log

    # Possible values include: debug, info, notice, warn, error, crit,
    # alert, emerg.
    LogLevel warn

    CustomLog /var/log/apache2/access.log combined
    ServerSignature On

</VirtualHost>

```

- Ideally, the CGIs will require authentication so that users can only see their own print quota account details... this has proved difficult so far.
- Change the 'poor man threshold' in `pykota.conf` so that the users get a warning at a time
- Change the enforcement to 'laxist' in `pykota.conf` so that users can go into debt on their last job (i.e. A job may cost 20p but a user only has 10p so their balance will be allowed to go to -10p but they will not be able to print again until a value greater than 10p has been added)
- Change the settings to prevent duplicates

- Above changes can be seen in pykota.conf below:

```
## pykota.conf
#
# IMPORTANT : many more directives can be used, and some of the directives
# below accept different and/or more complex parameters. Please read
# /usr/share/pykota/conf/pykota.conf.sample for more details about the
# numerous possibilities allowed.
#
[global]

# Database settings
storagebackend : pgstorage
storageserver : 127.0.0.1
storagename : pykota
storageuser : pykotauser
storageuserpw : readonlypw
storagecaching : No
disablehistory : No

# Logging method
logger : system

# Set debug to Yes during installation and testing
debug : Yes

# Who should receive automatic bug reports ?
crashrecipient : pykotacrashed@librelogiciel.com

# Should we keep temporary files on disk ?
# Set this to yes for debugging software accounting problems
keepfiles : no

# Logos for banners and CGI scripts
logourl : http://www.pykota.com/pykota.png
logolink : http://www.pykota.com/

# SMTP
smtpserver : <IP ADDRESS OF SMTP SERVER>
maildomain : MAIL DOMAIN HERE

# Print Administrator
admin : Administrator
adminmail : EMAIL ADDRESS HERE

# Use usernames as-is or convert them to lowercase ?
utolower : Yes

# Should we hide some fields in the history (title, filename) ?
privacy : no

# Should we charge end users when an error occurs ?
onbackenderror : nocharge

# Default accounting methods :
preaccounter : software()
accounter : software()
onaccountererror : stop

# Who will receive warning messages ?
# both means admin and user.
mailto : both

# Grace delay for pages based quotas, works the same
```

```

# as for disk quotas
gracedelay : 7

# Configurable zero, to give free credits
balancezero : 0.0

# Warning limit for credit based quotas
poorman : 0.20

# Warning messages to use
poorwarn : Your Print Quota account balance is low - you have less than 20p of
c
redit left. You will soon be unable to print unless you top up your account.

softwarn : Your Print Quota Soft Limit is reached.
This means that you may still be allowed to print for some
time, but you must contact your administrator to purchase
more print quota.

hardwarn : Your Print Quota Hard Limit is reached.
This means that you are not allowed to print anymore.
Please contact the network manager
as soon as possible to solve the problem.

# Number of banners allowed to be printed by users
# who are over quota
maxdenybanners : 0

# Should we allow users to ever be over quota on their last job ?
# strict means no.
enforcement : laxist

# Should we trust printers' internal page counter ?
trustjobsize : yes

# How to handle duplicate jobs
denyduplicates : no
duplicatesdelay : 0

# What should we do when an unknown user prints ?
# The policy below will automatically create a printing account
# for unknown users, allowing them to print with no limit on the
# current printer.
policy : external(/usr/share/pykota/bainesNewUser.sh $PYKOTAUSERNAME
$PYKOTAPRINTERNAME)

# Here are some lines that we suggest you add at the end
# of the pykota.conf file. These lines gives possible
# values for the way print jobs' size will be computed.
# NB : it is possible that a manual configuration gives
# better results for you. As always, your mileage may vary.
#
[PRINTER-01]
preaccounter : software()
accounter : software()

[PRINTER-02]
preaccounter : software()
Accounter : software()

...

```