Gateway Installation Guide

Institute of Information Technology
http://icn.itec.aau.at/
Sebastian Theuermann (sebastian.theuermann@aau.at)
Daniel Posch (daniel.posch@itec.aau.at)

February 2016

Contents

1 Introduction

For a more secure / centralized access to the Pi-network a dedicated server is used as gateway. This server possesses two network cards where one is connected to the Pi management network and one is connected to the local network.

2 Hardware Requirements

At least the minimal hardware-requirements of Ubuntu 14.04 should be met (`https://wiki.ubuntu.com/TrustyTahr/ReleaseNotes/UbuntuGNOME`).

Additionally, two (preferably Gigabit) Ethernet Network Interfaces are required.

3 Software Requirements

3.1 Operating System

Download and install (for example) Ubuntu 14.04.3 LTS Desktop-Version (`http://www.ubuntu.com/download/desktop`, username e.g. 'pi-gateway')

3.2 Packages

A list of all manually installed packages on our gateway-server can be found in **Appendix A** or in the embedded File **pkgList.txt**  (opening the embedded file may require a desktop-PDF-reader).

This list contains way more packages (also already installed system packages) than necessary to set up the gateway-server, you may choose to install only selected packages (manual using e.g. `sudo apt-get install ...`) or use the list and the following commands to ensure all of them are installed.

Update your local package index using

```
sudo apt-get update
```

Install the necessary packages from file

```
sudo apt-get install $(grep -vE "^\s*#" pkgList.txt  | tr "\n" " ")
```

3.3 Pi-Scripts

Download the latest version of the **pi_scripts** from `http://icn.itec.aau.at/download/`.

4 Basic installation

4.1 Prerequisites

Ensure that all requirements are met (hardware, OS installed, packages installed, pi_scripts downloaded)

4.2 Configure the gateway's IP-address for the internal-network

Edit the file `/etc/network/interfaces` in order to assign a static IP-address to the internal network interface (connected to the Pi-network [management], here eth2).

The content of our `/etc/network/interfaces` is as follows:

```
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback

auto eth2
iface eth2 inet static
address 192.168.0.1
netmask 255.255.255.0
```

4.3 Enable IP Forwarding

To enable IP-Forwarding, the following line in `/etc/sysctl.conf` needs to be un-commented (removal of preceding #-symbol).

```
# Uncomment the next line to enable packet forwarding for IPv4
#net.ipv4.ip_forward=1
```

4.4 Enable Network Address Translation (NAT)

To enable the gateway to do NAT for the Pis, the following script has to be executed during boot time. Please ensure that **EXTIF** matches your external network interface (here eth1) and **INTIF** matches your internal network interface (here eth2). Create a file named **nat.sh** with following content or download the attached file **nat.sh**  .

```
echo -e "\n\nLoading simple rc.firewall-iptables version $FWVER.."
echo -e "\n"
DEPMOD=/sbin/depmod
MODPROBE=/sbin/modprobe

EXTIF="eth1"
INTIF="eth2"
#INTIF2="eth0"
echo "   External Interface:  $EXTIF"
echo "   Internal Interface:  $INTIF"

#=================================================================
#== No editing beyond this line is required for initial MASQ tests
echo -en "   loading modules: "
echo "  - Verifying that all kernel modules are ok"
$DEPMOD -a
echo "----------------------------------------------------------"
echo -en "ip_tables, "
$MODPROBE ip_tables
echo -en "nf_conntrack, "
$MODPROBE nf_conntrack
echo -en "nf_conntrack_ftp, "
$MODPROBE nf_conntrack_ftp
echo -en "nf_conntrack_irc, "
$MODPROBE nf_conntrack_irc
echo -en "iptable_nat, "
$MODPROBE iptable_nat
echo -en "nf_nat_ftp, "
$MODPROBE nf_nat_ftp
echo "----------------------------------------------------------"
echo -e "   Done loading modules.\n"
echo "   Enabling forwarding.."
echo "1" > /proc/sys/net/ipv4/ip_forward
echo "   Enabling DynamicAddr.."
echo "1" > /proc/sys/net/ipv4/ip_dynaddr
echo "   Clearing any existing rules and setting default policy.."
```

```
iptables-restore <<-EOF
*nat
-A POSTROUTING -o "$EXTIF" -j MASQUERADE
 COMMIT
*filter
:INPUT ACCEPT [0:0]
:FORWARD DROP [0:0]
:OUTPUT ACCEPT [0:0]
-A FORWARD -i "$EXTIF" -o "$INTIF" -m conntrack --ctstate \
ESTABLISHED,RELATED -j ACCEPT
-A FORWARD -i "$INTIF" -o "$EXTIF" -j ACCEPT
-A FORWARD -j LOG
COMMIT
EOF

echo -e "\nrc.firewall-iptables v$FWVER done.\n"
```

Next, make the file **nat.sh** executable by running:

```
chmod a+x nat.sh
```

You may test the script by running it as root:

```
sudo sh nat.sh
```

Now you can login to a Pi and try to reach the outside world (e.g. the Google public DNS) via the gateway.

```
# logged in on a Pi
ping -c 3 -W 10 8.8.8.8
```

If the tests succeed, schedule **nat.sh** to be run during system-startup

```
sudo cp nat.sh /etc/init.d/
sudo ln -s /etc/init.d/nat.sh /etc/rc2.d/S95masquradescript
```

4.5 Information on Domain Name Service (DNS)

By default, the Pis are configured to use the gateway-server as DNS-server.

Ensure that the package `dnsmasq` is installed (part of software-requirements).

```
sudo apt-get install dnsmasq
```

With this package installed, the gateway-server should be able to act as DNS-server for the Pis without any further configuration.

You can test the DNS-server by logging in on a Pi and trying a DNS-lookup:

```
# logged in on a Pi
host www.google.com
```

4.6 Installation of the Pi-Scripts

Extract the downloaded archive **pi_scripts.zip** in the home-folder `/home/pi-gateway/`.

5 Installation of a network-status overview page

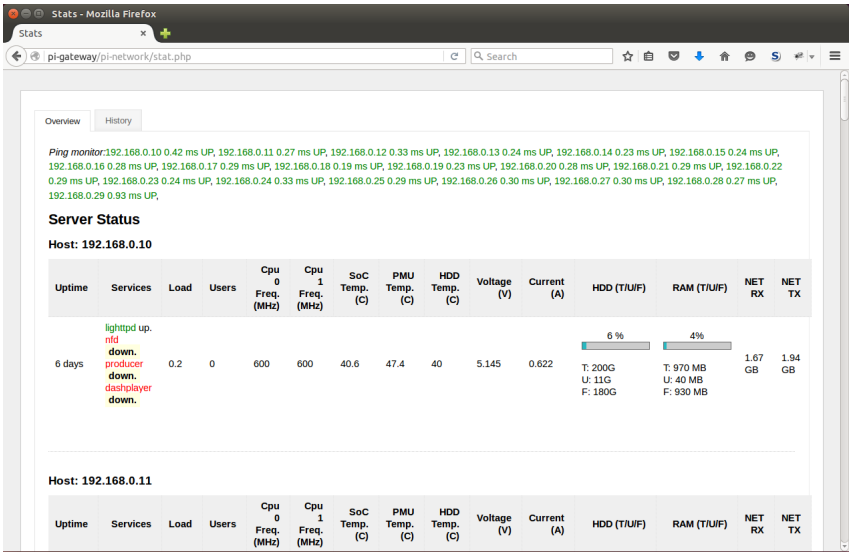
This is a optional step, not necessary for running emulations on the network.

This PHP-script displays an overview of the current status of the Pi-network by requesting status-information of the individual Pis (in form of JSON-files served by the Pis own lightweight-webservers) and rendering it. Additionally, it is capable of displaying a history of some selected characteristics of every Pi.

This overview was created with our needs in mind, please feel free to customize the scripts creating the JSON-files and stat.php itself.

It fulfills the following information needs:

- Which Pis are 'up' at the moment (reachable by pinging them)
- What is the status of every Pi (cpu-load, voltage, ...)
- How did some important characteristics of a PI change over time (history)



5.1 Requirements

- Make sure the source code management system **git** (e.g. package `git`) and a **webserver** (e.g. package `apache2`) are installed.
- Make sure that **php** (e.g. package `php5`) and the **php-module for your webserver** (e.g. package `libapache2-mod-php5`) are installed.

  The default document-root of the webserver is (on our gateway) the directory `/var/www/html`.

5.2 Set up PHP

Enable the **php-module for your webserver**:

```
sudo a2enmod php5
sudo service apache2 restart
```

If you want to check if your PHP-setup works, create the file `/var/www/html/phpinfo.php` with following content:

```
<?php
phpinfo();
?>
```

And try to open in in your browser via `http://localhost/phpinfo.php`.

5.3 Download the source-files

Change into the document-root of your webserver and clone the GitHub-Repository containing the source-files.

```
cd /var/www/html
sudo git clone https://github.com/theuerse/netstat.git
```

5.4 Adapt settings to your network

In the top-portion of the file **stat.php**, change the names/IP-addresses in the array **$hostlist** to suit your network.

5.5 Create the history-folder

Use the following command to create a sub-directory wherein the script (stat.php) can accumulate a history of Pi-status-information / JSON-files.

```
# in the source-directory
sudo mkdir history
```

5.6 Make sure that the webserver has permission to write in the source-directory

In order for the PHP-script to locally save the status-JSON-files, the webserver has to have write-permission in the source-directory (especially in the history-sub-directory). One way to achieve this, is by transferring ownership of the source-directory (and its sub-directories) to the webserver's system-user (www-data in our case). Another one is making it and it's sub-directories writable to all users / members of a group using 'chmod -R...' .

```
# 'outside' the source-directory
sudo chown -R www-data:www-data netstat
```

5.7 Set up cron-jobs to manage the status-file history

Two cron-jobs manage the JSON-file-history, one of them imports the newest JSON-files every hour and the second one removes all archived files that are older than two days. To set them up, you have to edit the gateway's crontab,

```
sudo crontab -e
```

and add the last two lines found in the following excerpt of our crontab:

```
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
0 * * * * find /var/www/html/netstat/history -type f -mtime +1 -delete
15 * * * * wget -qO /dev/null "http://143.205.122.89/netstat/stat.php?action=save&key=8A29691737D"
```

Where 143.205.122.89 is one of the gateway's IP-addresses (local network). Please make sure to save before exiting the editor, or the changes will not take effect.

The **first line** (`0 * * * * find ...`) is executed hourly and deletes all files which haven't been modified for more than two days (/ are older than two days).

The graphs in the history-tab of the overview-page display an overview of the data of all files in the history folder, you may change the amount of entries in the history-graphs by changing the number of JSON-files kept in the history-folder. (e.g. by modifying the aforementioned 'first line')

The **second line** (`15 * * * * wget ...`) is called on the 15th minute of every hour (hourly) and orders the stat.php script to request the Pi's current JSON-files and add them to the history. If we would do this at the beginning of every hour (0 * * * *) or half hour (*/30 * * * *), we would risk reading old files a second time because the new ones might not have been created yet (race-condition, given JSON-file creation all 30 minutes / once at the beginning of every hour).

5.8 Try it out

Now, you should be able to open the overview-page both locally and from your local network with your browser:

- `http://pi-gateway/netstat/stat.php` from your local network (where '`pi-gateway`' represents the name/IP-address of your gateway-server)
- or `http://localhost/netstat/stat.php` from the gateway-server

6 Managing the realtime_logging daemon

By default, when using the provided Pi-SSD-image (`http://icn.itec.aau.at/download/`), the realtime_logging daemon is enabled and automatically starts.

In case you do not use this kind of fine-grained logging (as mentioned in section 7) or you do not want the additional overhead caused by realtime-logging, you may deactivate it for the current session or permanently.

You may check the state of the realtime_logging daemon on a Pi by issuing:

```
sudo service realtime_logging status

ps -e | grep realtime_loggin
```

The daemon itself is modelled after the init-script-template of /etc/init.d/skeleton and therefore supports some common commands.

```
sudo service realtime_logging start
sudo service reatlime_logging stop
sudo service realtime_logging restart
sudo service realtime_logging status
```

To disable/enable the daemon from automatically starting on system startup, you may issue following commands.

```
cd /etc/init.d/ && sudo update-rc.d realtime_logging disable
cd /etc/init.d/ && sudo update-rc.d realtime_logging enable

man update-rc.d
```

Note: Using disable/enable does NOT stop/start the realtime_logging daemon (only at the next runlevel-change), either start/stop them yourself, change runlevel or reboot the Pi.

In order to perform the operations on all Pis, we suggest using a script (e.g. runcommand.py of the provided pi_scripts `http://icn.itec.aau.at/download/`).

7 Installation of a network-visualization

This is a optional step, not necessary for running emulations on the network.

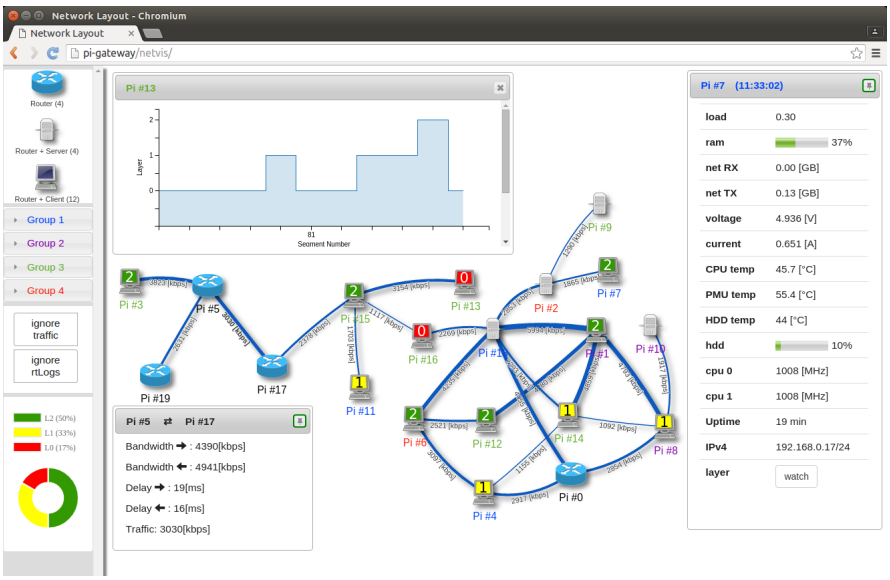
It provides an overview of the currently deployed network-topology, detail-information about the status of the individual Pis, and on demand both a visualization of the current traffic in the network and the last consumed SVC-Layer/Quality per client.

The realtime_logging-daemons (section 6) on the individual Pis supply it with information.

This visualization was created with our needs in mind, please feel free to customize the scripts creating the JSON-files/log-files and the visualization (networkLayout.html/js/css) itself.

It fulfills the following information needs:

- How does the network look like (connections, roles, bandwidth)
- What is the status of every Pi (cpu-load, voltage, ...)
- What is the current traffic in the network (per connection / edge)
- What is the last consumed layer (quality) per client (SVC)
- How does the consumed layer / quality per client change over time (SVC history)



7.1 Requirements

- Make sure the source code management system **git** (e.g. package `git`) and a **webserver** (e.g. package `apache2`) is installed. The default document-root of the webserver is (on our gateway) the directory `/var/www/html`.
- Make sure the packages `samba` and `samba-common` are installed in order to be able to share a directory of the gateway-server with the Pis.

7.2 Enable Headers-module of the webserver

The Headers-module of the webserver has to be enabled in order to be able to control/modify HTTP request and response headers.

```
sudo a2enmod headers
```

The downloaded project contains a **htaccess**-file using this module to set custom 'lifetimes' for the served JSON-files.

```
AddCharset UTF-8 .html .css .php .txt .js .json

<ifmodule mod_headers.c>
 <filesmatch "\\.(json|log)$">
 Header set Cache-Control "max-age=5, public"
 </filesmatch>
</ifmodule>
```

7.3 Set up a Samba share

The type of 'realtime-logging' we use to monitor/visualize the network-status involves the creation and transmission of a JSON-file (for the status) and a client-log (SVC-consumer log) every x seconds (where e.g. x=5).

In order avoid excessive strain (polling) to the gateway-server, the gateway-server shares a directory (Samba share) with the Pis, for them to save their JSON-files / log-files in.

The name of the share will be **pilogging**, the Samba username will be **www-data**, the directory shared with the Pis will by `/run/shm`.

7.3.1 Add a Samba-user who may access the share

```
sudo smbpasswd -a www-data
```

To not needlessly complicate things, we reuse the system-password of the system-user www-data.

7.3.2 Edit the Samba config file

Insert the following new entry to the Samba config file (`/etc/samba/smb.conf`)

```
[pilogging]
   comment = LOG IN HERE
   path = /run/shm
   guest ok = yes
   browseable = yes
   create mask = 0600
   directory mask = 0700
   writeable = yes
```

7.3.3 Restart the Samba-server

```
sudo service smbd restart
```

7.3.4 Check integrity of Samba config-file

After restarting the Samba-service, you may check the integrity of the Samba config-file by running the command:

```
testparm
```

7.3.5 Test the Samba share

After logging in on a Pi, you may try to put a arbitrary file on the gateway in order to test the Samba-share:

```
# logged in on a Pi
smbclient -U www-data //192.168.0.1/pilogging -c "put foo.txt" www
```

- **www-data** is the samba-user / system-user of the apache-webserver
- **192.168.0.1** is the static IP-address of the gateway-server
- **pilogging** is the name of the Samba-share hosted by the gateway-server
- **-c "put .... "** commands the Samba-client to put a certain file (from the current directory) in the shared directory
- **www** is the (example-) password of the Samba-user / system-user www-data

7.4 Download the source-files

Change into the document-root of your webserver and clone the GitHub-Repository containing the source-files.

```
cd /var/www/html
git clone https://github.com/theuerse/netvis.git
```

7.5 Set up input paths / directory

The visualisation displays the network-structure given in a network-topology-file and individual status-information per Pi given in JSON-files / .log-files (SVC info).

In the top portion of the file **networkLayout.js**, change the path to the network-topology-file and the path to the directory containing the JSON-files to fit your needs.

```
var topologyFilePath = "generated_network_top.txt";
var jsonDirectory = "network/";
...
```

In our case,
we created both a symlink (`ln -s TARGET LINK_NAME`) to the topology-file (called generated_network_top.txt) and a symlink to the directory that contains the Pis JSON-files. The path/directory in **networkLayout.js** points to this symlinks.

The symlink **generated_network_top.txt** points to
`/home/pi-gateway/pi-network/generated_network_top.txt`,

and the symlink **network** points to `/run/shm`.

Although this indirection was very handy for our development purposes, you may leave it out and directly specify the path to the topology-file / directory containing the JSON/log - files in **networkLayout.js** as mentioned above.

8 Appendix A - list of all installed packages on the gateway-server

A list of all installed Packages on the gateway-server, created using the command `apt-mark showmanual`.

```
acpi-support
activity-log-manager-control-center
adduser
aisleriot
alsa-base
anacron
apache2
app-install-data-partner
apparmor
apt
apt-transport-https
apt-utils
avahi-autoipd
baobab
base-files
base-passwd
bash
bash-completion
bluez-alsa
bluez-cups
branding-ubuntu
brltty
bsdutils
busybox-initramfs
busybox-static
bzip2
checkbox-gui
cheese
command-not-found
console-setup
coreutils
cpio
cron
cups-bsd
dash
debconf
debconf-i18n
debianutils
deja-dup
dh-python
diffutils
dmsetup
dmz-cursor-theme
dnsmasq
dnsutils
doc-base
dpkg
e2fslibs
e2fsprogs
ed
eject
```

empathy
eog
evince
example-content
file
file-roller
findutils
firefox
firefox-locale-en
fonts-droid
fonts-freefont-ttf
fonts-kacst-one
fonts-khmeros-core
fonts-lao
fonts-liberation
fonts-lklug-sinhala
fonts-nanum
fonts-sil-abyssinica
fonts-sil-padauk
fonts-takao-pgothic
fonts-thai-tlwg
fonts-tibetan-machine
foomatic-db-compressed-ppds
friendly-recovery
ftp
gcc
gcc-4.8-base
gcc-4.9-base
gedit
ghostscript-x
git
gnome-accessibility-themes
gnome-disk-utility
gnome-font-viewer
gnome-mahjongg
gnome-orca
gnome-screenshot
gnome-session-canberra
gnome-sudoku
gnome-system-log
gnome-terminal
gnomine
gnupg
gpgv
grep
grub-common
grub-gfxpayload-lists
grub-pc
grub-pc-bin
grub2-common
gstreamer0.10-alsa
gstreamer0.10-plugins-base-apps
gstreamer1.0-alsa
gstreamer1.0-plugins-base-apps
gucharmap
gvfs-bin
gvfs-fuse
gzip
hostname
hplip
hyphen-en-us

ibus-pinyin
ibus-table
ifupdown
info
init-system-helpers
initramfs-tools
initramfs-tools-bin
initscripts
inputattach
insserv
iperf
iproute2
iputils-ping
iputils-tracepath
irqbalance
isc-dhcp-client
isc-dhcp-common
kbd
kerneloops-daemon
keyboard-configuration
klibc-utils
kmod
landscape-client-ui-install
language-pack-en
language-pack-en-base
language-pack-gnome-en
language-pack-gnome-en-base
language-selector-gnome
less
libacl1
libapache2-mod-php5
libapt-inst1.5
libapt-pkg4.12
libarchive-extract-perl
libatk-adaptor
libattr1
libaudit-common
libaudit1
libblkid1
libbsd0
libbz2-1.0
libc-bin
libc6
libcap2
libcap2-bin
libcgmanager0
libcomerr2
libdb5.3
libdbus-1-3
libdebconfclient0
libdevmapper1.02.1
libdrm2
libegl1-mesa-drivers-lts-utopic
libestr0
libexpat1
libffi6
libfribidi0
libgcc1
libgcrypt11
libgdbm3
libgnutls-openssl27

libgnutls26
libgpg-error0
libjson-c2
libjson0
libklibc
libkmod2
liblocale-gettext-perl
liblockfile-bin
liblockfile1
liblog-message-simple-perl
liblzma5
liblzo2-2
libmagic1
libmodule-pluggable-perl
libmount1
libmpdec2
libncurses5
libncursesw5
libnewt0.52
libnih-dbus1
libnih1
libnotify-bin
libp11-kit0
libpam-cap
libpam-modules
libpam-modules-bin
libpam-runtime
libpam0g
libpcre3
libplymouth2
libpng12-0
libpod-latex-perl
libpopt0
libprocps3
libproxy1-plugin-gsettings
libproxy1-plugin-networkmanager
libpython3-stdlib
libpython3.4-minimal
libpython3.4-stdlib
libqt4-sql-sqlite
libreadline5
libreadline6
libreoffice-calc
libreoffice-gnome
libreoffice-help-en-us
libreoffice-impress
libreoffice-math
libreoffice-ogltrans
libreoffice-presentation-minimizer
libreoffice-writer
libselinux1
libsemanage-common
libsemanage1
libsepol1
libslang2
libsqlite3-0
libss2
libssl1.0.0
libstdc++6
libtasn1-6
libterm-ui-perl

libtext-charwidth-perl
libtext-iconv-perl
libtext-soundex-perl
libtext-wrapi18n-perl
libtinfo5
libudev1
libusb-0.1-4
libustr-1.0-1
libuuid1
libwmf0.2-7-gtk
libxp6
linux-generic-lts-utopic
locales
lockfile-progs
login
logrotate
lsb-base
lsb-release
lshw
lsof
ltrace
makedev
mawk
memtest86+
mime-support
mlocate
module-init-tools
mount
mountall
mtr-tiny
multiarch-support
myspell-en-au
myspell-en-gb
myspell-en-za
mythes-en-us
nano
nautilus
nautilus-sendto
nautilus-share
ncurses-base
ncurses-bin
net-tools
netbase
netcat-openbsd
nfs-kernel-server
notify-osd
ntp
ntpdate
onboard
openoffice.org-hyphenation
openprinting-ppds
os-prober
overlay-scrollbar
passwd
pcmciautils
perl
perl-base
perl-modules
php5
plymouth
plymouth-theme-ubuntu-logo

plymouth-theme-ubuntu-text
policykit-desktop-privileges
popularity-contest
pppconfig
pppoeconf
printer-driver-c2esp
printer-driver-foo2zjs
printer-driver-min12xxw
printer-driver-pnm2ppa
printer-driver-ptouch
printer-driver-pxljr
printer-driver-sag-gdi
printer-driver-splix
procps
pulseaudio-module-bluetooth
python-dev
python-igraph
python-pip
python3
python3-minimal
python3.4
python3.4-minimal
qt-at-spi
readline-common
remmina
resolvconf
rfkill
rhythmbox
rhythmbox-plugin-magnatune
rsync
rsyslog
samba
samba-common
screen
seahorse
sed
sensible-utils
shotwell
simple-scan
sni-qt
software-center
speech-dispatcher
ssh
ssh-askpass-gnome
sshpass
strace
sudo
sysv-rc
sysvinit-utils
tar
tcpdump
telepathy-idle
telnet
thunderbird
thunderbird-gnome-support
thunderbird-locale-en
thunderbird-locale-en-us
time
totem
totem-mozilla
transmission-gtk

```
ttf-indic-fonts-core
ttf-punjabi-fonts
tzdata
ubuntu-artwork
ubuntu-desktop
ubuntu-docs
ubuntu-keyring
ubuntu-minimal
ubuntu-session
ubuntu-settings
ubuntu-sounds
ubuntu-sso-client-qt
ubuntu-standard
ubuntu-wallpapers-trusty
ucf
udev
ufw
unity
unity-settings-daemon
unity-webapps-common
upstart
ureadahead
usb-creator-gtk
util-linux
uuid-runtime
vim
vim-common
vim-tiny
vino
wamerican
wbritish
whiptail
whoopsie
wireless-tools
xcursor-themes
xdg-user-dirs
xdg-user-dirs-gtk
xdiagnose
xkb-data
xorg
xserver-xorg-input-all-lts-utopic
xserver-xorg-lts-utopic
xserver-xorg-video-all-lts-utopic
xterm
xul-ext-unity
xul-ext-webaccounts
zlib1g
```