Stateless Linux with OpenSLX

CC University of Freiburg

March 31, 2011

Stage 0: Server installation

For installation you will need somne additional packages, depending on the type of installation. In many cases you need to add some additional Perl packages as well. Example for Ubuntu, on RPM based Systems packages can have slightly different names:

```
# build requirements
$ aptitude install git-core make gcc
# server environment
$ aptitude install rsync hpa-tftpd nfs-kernel-server
# perl environment
$ aptitude install perl perl-base perl-modules libclone-perl libconfig-general-perl \
                   libdbi-perl liburi-perl libdigest-sha1-perl libwww-perl libjson-perl
```

For installation the GIT repository is recommended:

```
linux:~ # git clone git://git.openslx.org/openslx/core.git openslx
linux:~ # git checkout -b stable -t origin/stable
linux:~ # cd openslx
linux:~/openslx # make install
```

The configuration and setup programs all have the prefix `slx`, most of which are Perl scripts. Most tools deliver some short help with the option `--help`, the manpage can be accessed through `--man`.

All global settings of the packages are placed beneath `/etc/opt/openslx`. All static files are placed in `/opt/openslx`. In `/var/opt/openslx` the client system and configuartions are stored. `/srv/openslx` is used for the exported systems. Global settings can be changed through `slxsettings`.

DHCPD, TFTPD configuration

You'll need a DHCP and a TFTP server. Your ISC-DHCP-Server needs to be configured to boot stateless, where `next-server` means the TFTP server:

```
next-server 10.8.4.1
filename "pxelinux.0";
```

If you need to setup a DHCP server you can use this short example:

```
ddns-update-style none;
subnet 10.8.4.0 netmask 255.255.255.0 {
  server-identifier 10.8.4.1;
  next-server 10.8.4.1;
  filename "pxelinux.0";
  option routers 10.8.4.254;
  range 10.8.4.101 10.8.4.200;
}
```

You'll need to check if the DHCP server is listening on the correct interface. It can be usefull to set up a TFTP server on the server runnung OpenSLX. For this you'll have to edit the configuration file depending on the distribution. A small example for in.tftp follows (`/etc/xinetd.d/tftp`).

```
service tftp
{
        socket_type             = dgram
        protocol                = udp
        wait                    = yes
        user                    = root
        server                  = /usr/sbin/in.tftpd
        server_args             = -s /srv/openslx/tftpboot
        disable                 = no
}
```

Testing the TFTP server:

```
linux:~ # tftp 10.8.4.1
tftp> get pxelinux.0
tftp> quit
```

Stage 1: Installation and configuration of the client systems

First you'll need a reference system which can be used as a client system. Here you can use virtual machines like VMware or VirtualBox. For a better idetification you can use a suffix `-mysuffix` after the distribution name. The clone of the reference system occurs in the background with the command `rsync`. On an Ubuntu reference system you must ensure that `root` can login through SSH.

```
linux:~ # slxos-setup clone 10.60.4.50:/ ubuntu-10.10-mysystem
Cloning vendor-OS from '10.60.4.50:/' to '/space/stage1/ubuntu-10.10-mysystem'...
...
Vendor-OS 'ubuntu-10.10-mysuffix' has been cloned successfully.
Vendor-OS 'ubuntu-10.10-mysuffix' has been added to DB (ID=1).

linux:~ # slxos-setup list-installed
List of installed vendor-OSes:
        ubuntu-10.10-mysuffix
```

Plug-ins - modular extensions of the base package

OpenSLX can be extended by implementing special extensions (OS-plugins). Installation and configuration of the plug-ins done by the command `slxos-plugin`:

```
linux:~ # slxos-plugin install ubuntu-10.10-mysuffix desktop
Plugin desktop has been installed into vendor-OS 'ubuntu-10.10-mysuffix'.
```

Stage 2: Export of the client system

For a client rootfilesystem the installation from Stage 1 has to be transitioned into Stage 2, the export. The command is called `slxos-export`. Currently supported are `nfs` and `nbd-squash` exports.

```
linux:~ # slxos-export export ubuntu-10.10-mysuffix nfs
vendor-OS '/var/opt/openslx/stage1/ubuntu-10.10-mysuffix' successfully exported to
  '/srv/openslx/export/nfs/ubuntu-10.10-mysuffix'!
Export 'ubuntu-10.10-mysuffix::nfs' has been added to DB (ID=1)...
################################################################################
```

```
Please make sure the following line is contained in /etc/exports
in order to activate the NFS-export of this vendor-OS:
        /srv/openslx/export/nfs/ubuntu-10.10-mysuffix
          *(ro,no_root_squash,async,no_subtree_check)
################################################################################
system 'ubuntu-10.10-mysuffix::nfs' has been successfully added to DB (ID=1)

linux:~ # slxos-export list-exported
List of exported OSes:
        ubuntu-10.10-mysuffix::nfs
```

Stage 3/4: System and client specific boot setup

Preparation of the Kernel with the appropriate InitialRamFS and a matching client configuration. If you want to change the settings, you can use the command `slxconfig`.

```
linux:~ # slxconfig list-system
List of systems:
        <<<default>>>
        ubuntu-10.10-mysuffix::nfs

linux:~ # slxconfig add-client slx-test01 mac=00:01:02:03:04:06
client 'slx-test01' has been successfully added to DB (ID=1)

linux:~ # slxconfig change-system ubuntu-10.10-mysuffix::nfs country=de
```

The client configuration can be done in `/var/opt/openslx/config/<system-name>` as well. This directory will be packed and used during the boot of a client (ConfTGZ).

To complete the Stage 3 configuartion you'll have to use the tool `slxconfig-demuxer`. This creates the directory and all needed configuartions in `<public-path>/tftpboot` and craetes the ConfTGZ in `<public-path>/tftpboot/client-config`.

If the boot stops complaining about a missing network device driver it can be necessary to specify it. To avoid the same problem in the future change the 'defualt' system, which defines attributes for all systems.

```
linux:~ # slxconfig change-system default ramfs_nicmods='e1000 e1000e forcedeth tg3 e100'
```

If you are using a different NFS server than the TFTP server you should specify your export.

```
linux:~ # slxconfig change-export ubuntu-10.10-mysuffix::nfs server_ip=10.8.4.2
```

The clients are now configured for network boot. You'll need to enable booting from network on your client (PXE).