Traditional LAN Booting / OpenSLX Project

DAAD Summer School: Aspects of Large Scale High Speed Computing

21st March 2011

Dr. Dirk von Suchodoletz, Sebastian Schmelzer, Michael Janczyk

Faculty of Engineering, University Freiburg

Albert-Ludwigs-Universität Freiburg

UNI FREIBURG

Last Lecture

- Introduction / Motivation, why virtualize?
- Administrative and economic advantages
- History and main ideas
- Distinguish full, hardware assisted, para virtualization and tools using it, partitioning, emulation
- Practical application: Running Windows without pain in flexible lecture pools
- Classic server consolidation: Experiences and further usage scenarios
- Virtualization for preservation of complex digital objects

Overview of this Lecture

- Introduction
- Different Concepts of Operation
- Project Idea OpenSLX
- Getting started – Tools to use for basic and advanced setups
- ...

Structure: LAN Booting Linux

Different Concepts of Operation

Project Idea OpenSLX

Traditional LAN Boot

- Booting machines via PXE over Ethernet LAN connections
  - DHCP to provide basic IP setup
  - Next-Server and filename statements to provide information to load next stage bootloader via the net
  - Different variants possible – PXE/SysLinux the most common combination
- Boot conceptually does not differ much from traditional kernel and InitRamFS load of a modern Linux system
  - InitRamFS loads all necessary components to enable rootfilesystem the system later runs off
  - Easiest setup: Root filesystem via NFS, later experiments could use NBD/SquashFS

The OpenSLX Project

- To generalize the stateless Linux setups – OpenSLX project created
- Project focuses on Linux deployment in large setups
  - Active for a couple of years since end of 1990th
  - Developed mainly at Freiburg University
  - Deployed at some universities and public highschools in Germany
- Technologically based on the typical ingredients for diskless Linux systems

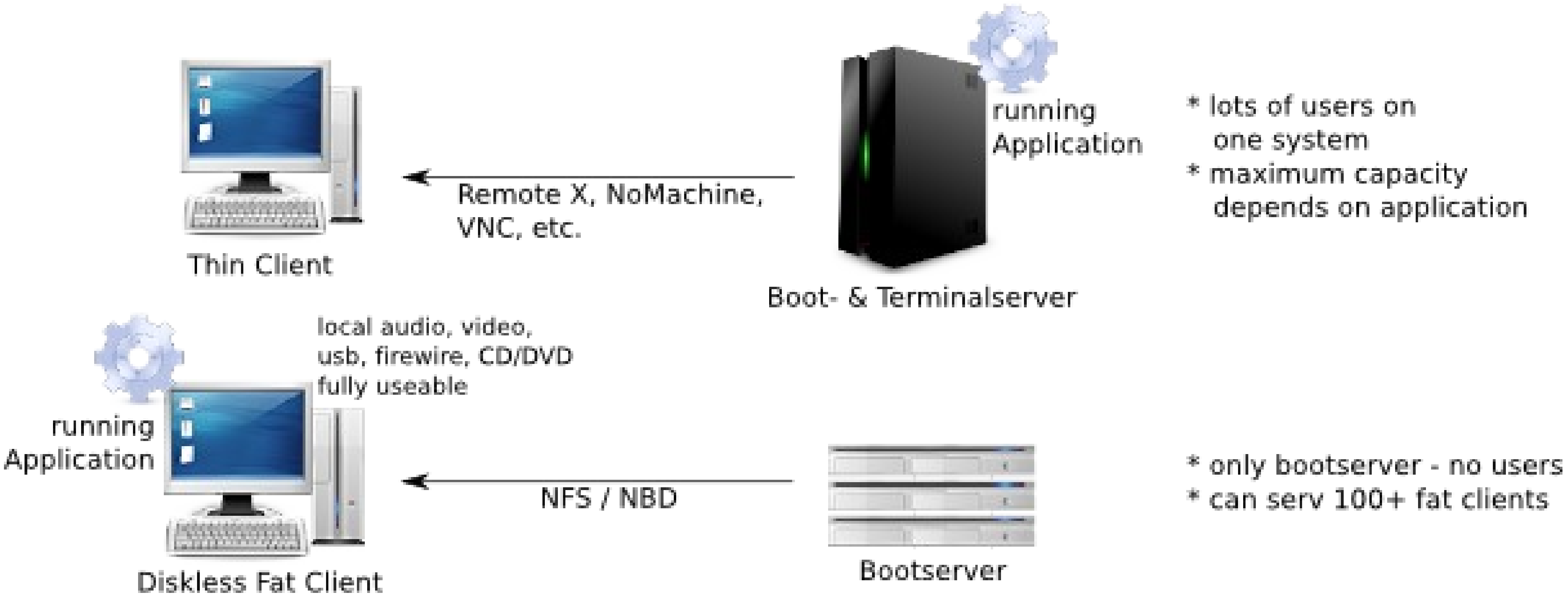
The OpenSLX Project

- Idea: Stateless clients offer all the functionality a user can expect
  - Exclusive CPU, direct and fast 3D/video output, direct hardware access to CD/DVS, for audio and periphery connectors
  - No restrictions regarding local USB, IEEE1394 devices
  - Easy deployment of virtual machines, like VirtualBox, VMware, QEMU/KVM as introduced in second lecture
- Standard Linux Workstation without a fixed disk installation of the the Operating System
- *Abstraction layer* for using standard Linux Distributions

The OpenSLX Project

Albert-Ludwigs-Universität Freiburg

- Different to projects like X2GO, LTSP, ThinStation, ...
  - Not a terminalserver but a full desktop/node setup
  - Complete stateless client with all tools and services for a workstation or cluster node



The OpenSLX Project

- PXE
  - Requires properly configured pointer from DHCP to TFTP
  - Difficult
    - If no PXE available on the client hardware and no PXE alternatives could be installed (too cheap, other architecture)
    - If subnet DHCP could not be reconfigured, offers no TFTP
    - Non-standard LAN media like WLAN, USB Ethernet, ...
- Mentioned projects handle typical LAN setups for large pools

→ OpenSLX offers with the introduction of it's PreBoot environment more generic boot options (other boot methods available – presented, experiments in 4th lecture)

OpenSLX Use-Cases

- *Provide standardized Linux Desktops* – Workdesk for students, employees, ...
- *Flexible proprietary desktop environments* – untie software from hardware and run it on-demand easily in floating manner using virtualization
- *Automatic maintenance and backup* – reboot normal desktop machines for malware checking and backup independent of the standard OS running
- *Fast switch between day and night mode* – use optimal software configuration for comfortable desktop and number crunshing

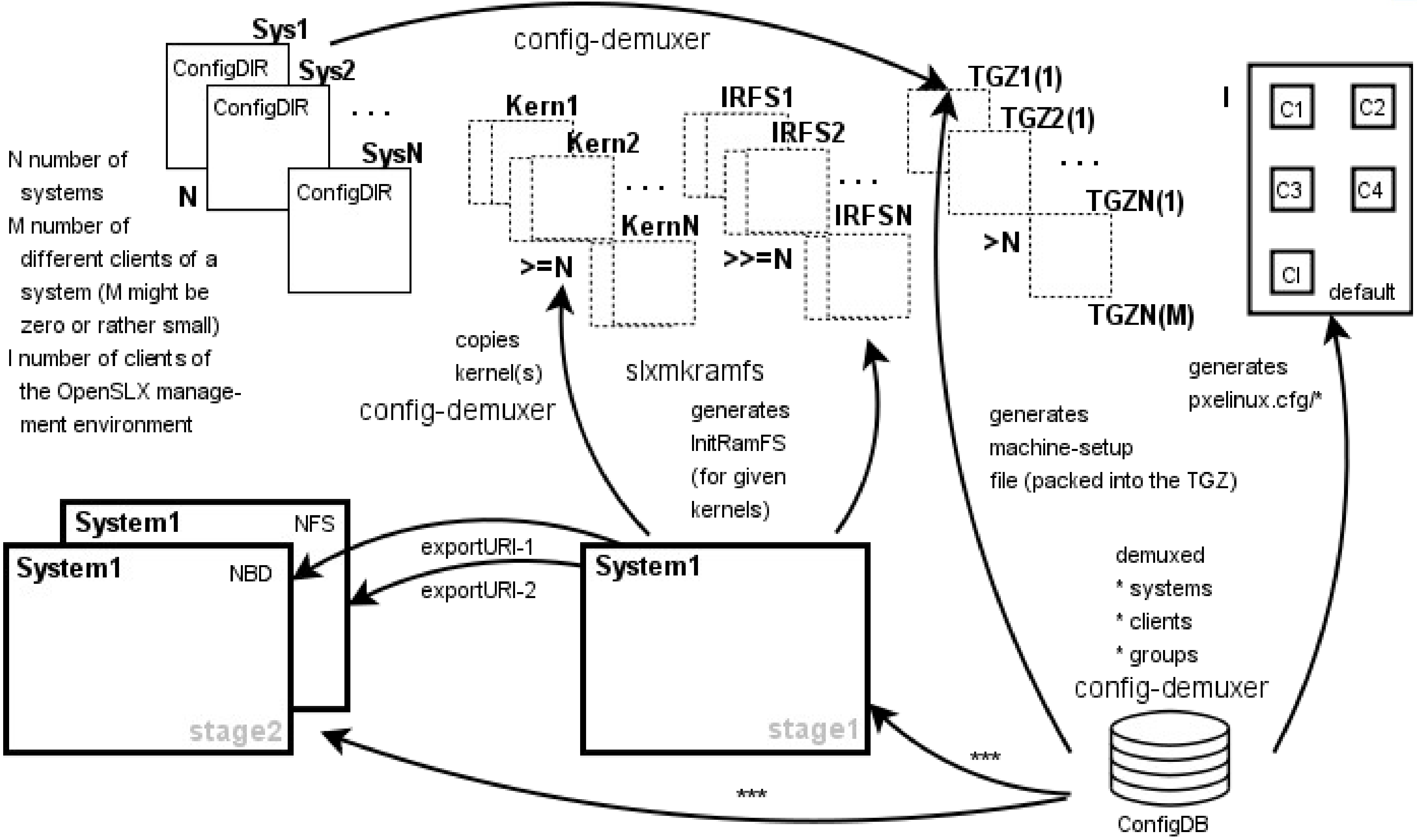
OpenSLX Use-Cases

- *Easy distributed software testing* – every potential user simply reboots his/her machine to check new versions or applications in an additional setup
- *Secure home banking terminal* (TPM secured boot)
- Of course the standard terminal server features available too
  - RDP, Citrix, XDMCP, ...

OpenSLX Implementation

- OpenSLX tools prepare the Linux distributions for export and deal with the setup of the clients
- Two major areas of action
  - Perl utilities for the interactive administration tasks on the server
  - Shell scripts for the automatic client setup
- Framework is meant to accommodate a larger number of different Linux variants and versions to boot with different options for the rootfilesystem

OpenSLX Implementation

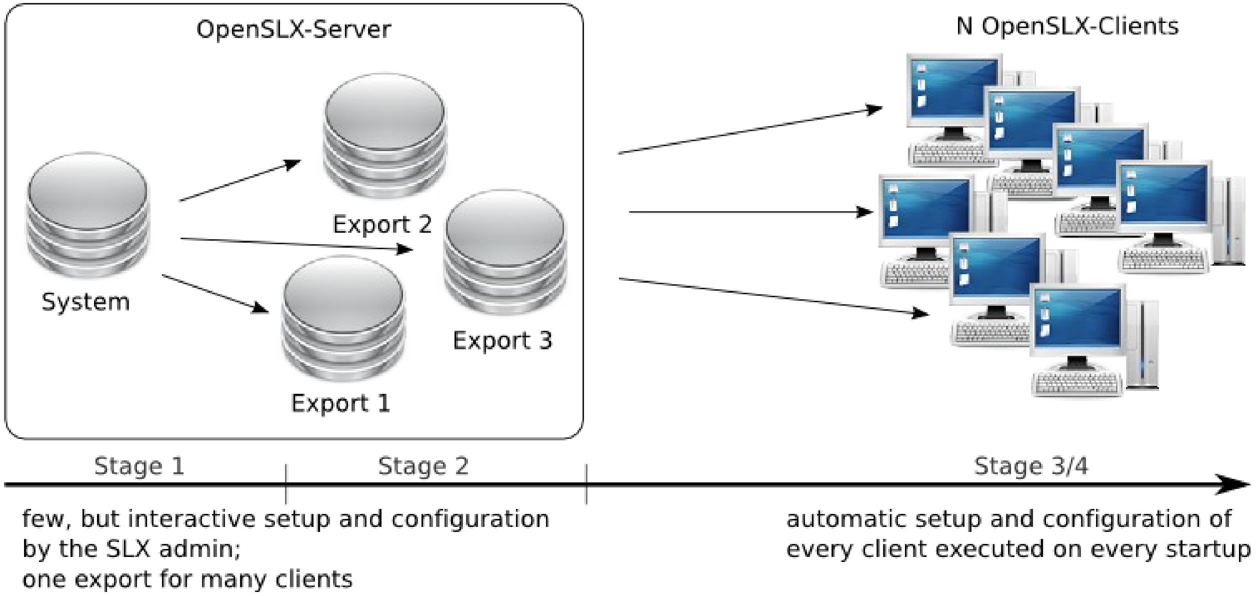


N number of systems
M number of different clients of a system (M might be zero or rather small)
I number of clients of the OpenSLX management environment

The OpenSLX Software

Albert-Ludwigs-Universität Freiburg

- Structured by defining stages
  - *PreBoot is a special stage to circumvent PXE/TFTP and boot media restrictions (discussed in fourth lecture)*
  - *Stage 1 is the base installation/preparation of a distribution to be exported in stage 2*



few, but interactive setup and configuration by the SLX admin;
one export for many clients

automatic setup and configuration of every client executed on every startup

The OpenSLX Software

- Structured by defining stages
  - *Stage 1 allows for additional packages – the OpenSLX plugins could be installed*
  - *Stage 2 defines (different) filesystem exports of Linux distributions mounted commonly by the clients*
  - *Stage 3 is the major client setup phase running within Initial RamFS preparing the root filesystem and configuration*
  - *Stage 4* is the client machine running the target Linux distribution allowing users to login graphically or running jobs of different types

Structure: OpenSLX

Albert-Ludwigs-Universität Freiburg

Getting Started

OpenSLX – Getting Started

- After installation of the toolset Linux distributions are to be prepared for later export, remember:
  - Challenge – generic filesystem mounted by a large number of different stateless clients read-only (hardware-, software-wise)
  - No per-client configuration possible at this level
- Several Linux distributions available for OpenSLX export depending on OpenSLX version
  - Ubuntu (8.04 … 10.10), SuSE (11.2,3) well supported
  - Debian, Scientific Linux, Gentoo in several stages of development

OpenSLX – Getting Started

- **Stage 1** actions to be run initially and for updates by the administrator on the server or preparation machine
  - Staging and file servers could be hosted on different machines
  - Staging servers do not need to be a power horse and run 24/7
  - File server should be able to serve the required number of clients (remember discussion in first lecture)

OpenSLX – Getting Started

- Setting up Stage 1 – preparing the *Vendor OS* (third party stuff)
  - **slxos-setup –** two possible methods implemented
  - *Cloning a running system* with the configured distribution which should be deployed
    - Source system named as stage 0 in OpenSLX wording (we use virtual machines for it) - installed rsync/ssh required
    - Easy to prepare, adapt to your needs on a running system
  - **slxos-plugin** to extend the base setup

OpenSLX – Prepare Exports

- Stage 2 prepares the actual exports (called *Systems* as they later run on the clients)
  - **slxos-exports** can produce different types
    - Classic NFS export
    - Producing SquashFS container file to be exported by Network Block Device (like NBD, DST, DNBD(2), ...)
- Yes – lots of files are duplicated, but
  - Disk space is not really an issue nowadays
  - Unavoidable to prepare exports using SquashFS
  - Avoids problems generated by working on NFS exports

OpenSLX – Prepare Exports

- Tools add *Vendor-OS* and *Systems* to a configuration database
  - Accessed by **slxconfig** tool
  - Takes the several configuration options of them, e.g. plugins installed
  - Could define client options

Client Configuration

- General setups for all clients (like authentication and user home sources) is normally done within the exported filesystem
- Several entry points for configuration
  - Using the database – possible for options made available by the base system and plugins
    - Switching off plugins
    - Defining different variable sets

Client Configuration

- Often not enough – more flexibility and additional configuration required
  - thus additional list of files could be added to stage 4 filesystem
    - Per system and per client
    - Scripts for execution during (tool restricted) stage 3
    - Files to be copied to /etc, /var …
  - Specific configuration directory for these files

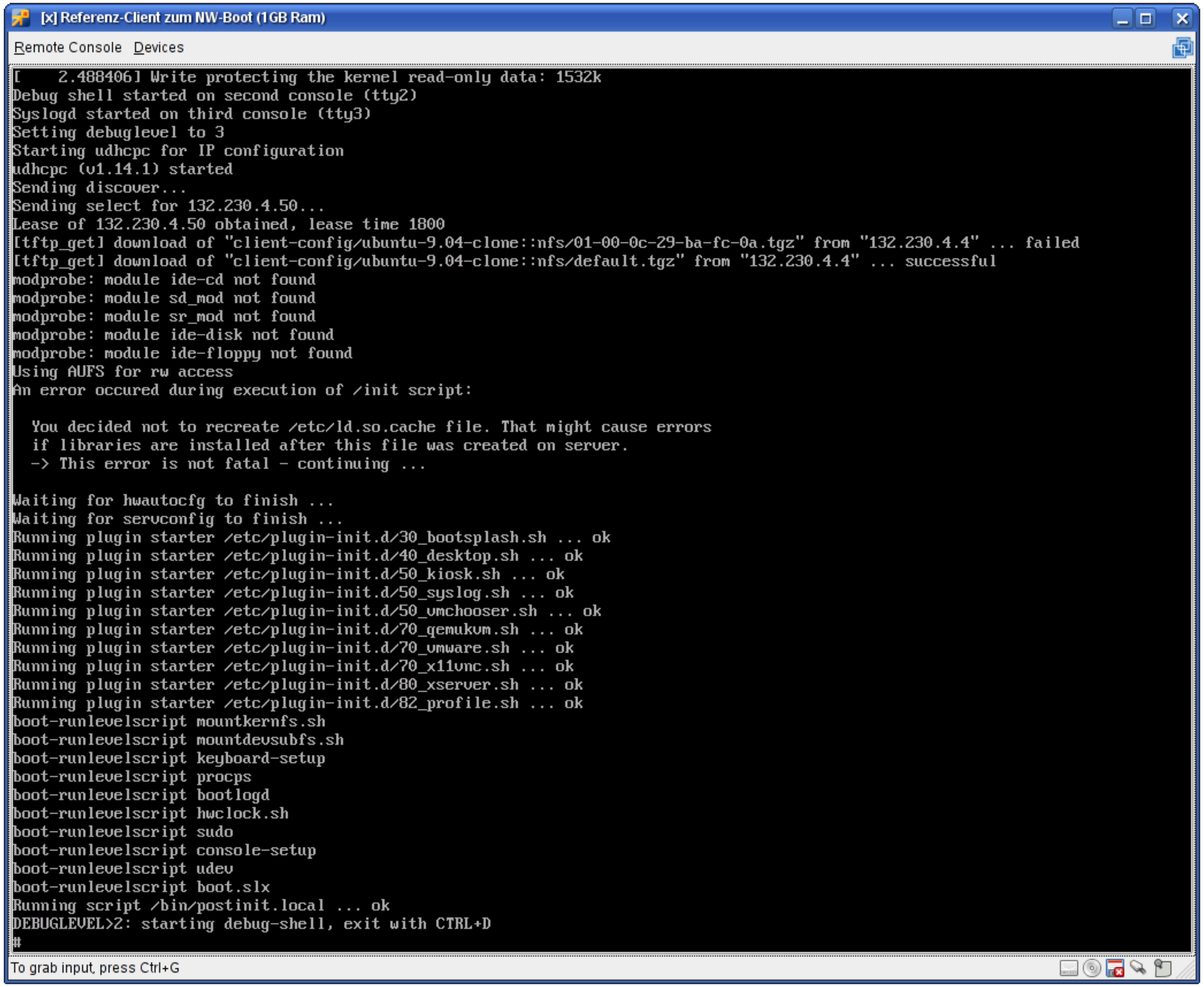
Boot: InitRamFS – Configuration Phase

- Configuration of OpenSLX clients during *every boot* within InitRamFS – should be fast and efficient
  - Remember first lecture: persistent storage is not desirable as adding overhead
- Provided bootscripts and the user land environment
  - Distribution independent mini environment, using *eglibc* and *busybox* – well known from embedded environments

Boot: InitRamFS – Configuration Phase

- Shell script *init* handling everything within InitRamFS
  - Loading of network adaptor module
  - IP configuration and mounting of the root filesystem via NFS or SquashFS NBD (DNBD2, ...)
  - Making all or parts of the later stage 4 root filesystem writeable (AUFS, UnionFS, COWloop or bind mounts)
  - Hardware autodetection and module loading
  - Distro specific configuration of software (**servconfig** script, client config via several methods)

Boot: InitRamFS – Configuration Phase


[x] Referenz-Client zum NW-Boot (1GB Ram)
Remote Console
Devices
To grab input, press Ctrl+G

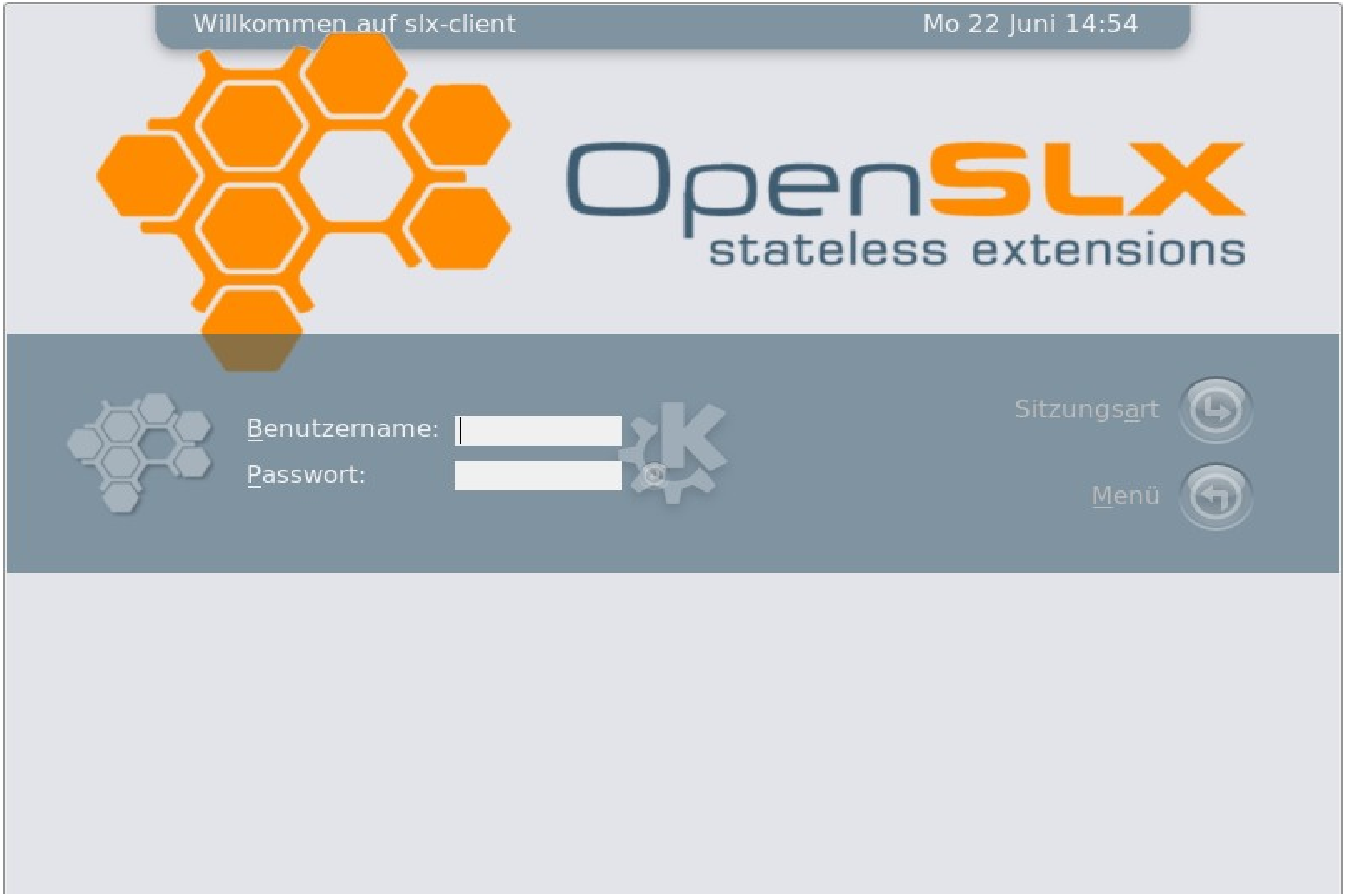
```
[    2.488406] Write protecting the kernel read-only data: 1532k
Debug shell started on second console (tty2)
Syslogd started on third console (tty3)
Setting debuglevel to 3
Starting udhcpc for IP configuration
udhcpc (v1.14.1) started
Sending discover...
Sending select for 132.230.4.50...
Lease of 132.230.4.50 obtained, lease time 1800
[tftp_get] download of "client-config/ubuntu-9.04-clone::nfs/01-00-0c-29-ba-fc-0a.tgz" from "132.230.4.4" ... failed
[tftp_get] download of "client-config/ubuntu-9.04-clone::nfs/default.tgz" from "132.230.4.4" ... successful
modprobe: module ide-cd not found
modprobe: module sd_mod not found
modprobe: module sr_mod not found
modprobe: module ide-disk not found
modprobe: module ide-floppy not found
Using AUFS for rw access
An error occured during execution of /init script:

  You decided not to recreate /etc/ld.so.cache file. That might cause errors
  if libraries are installed after this file was created on server.
  -> This error is not fatal - continuing ...

Waiting for hwautocfg to finish ...
Waiting for servconfig to finish ...
Running plugin starter /etc/plugin-init.d/30_bootsplash.sh ... ok
Running plugin starter /etc/plugin-init.d/40_desktop.sh ... ok
Running plugin starter /etc/plugin-init.d/50_kiosk.sh ... ok
Running plugin starter /etc/plugin-init.d/50_syslog.sh ... ok
Running plugin starter /etc/plugin-init.d/50_vmchooser.sh ... ok
Running plugin starter /etc/plugin-init.d/70_qemukvm.sh ... ok
Running plugin starter /etc/plugin-init.d/70_vmware.sh ... ok
Running plugin starter /etc/plugin-init.d/70_x11vnc.sh ... ok
Running plugin starter /etc/plugin-init.d/80_xserver.sh ... ok
Running plugin starter /etc/plugin-init.d/82_profile.sh ... ok
boot-runlevelscript mountkernfs.sh
boot-runlevelscript mountdevsubfs.sh
boot-runlevelscript keyboard-setup
boot-runlevelscript procps
boot-runlevelscript bootlogd
boot-runlevelscript hwclock.sh
boot-runlevelscript sudo
boot-runlevelscript console-setup
boot-runlevelscript udev
boot-runlevelscript boot.slx
Running script /bin/postinit.local ... ok
DEBUGLEVEL>2: starting debug-shell, exit with CTRL+D
#
```

- Plugin setup
- At the end: *switch_root* into stage 4

Stage 4 – Booted OpenSLX Client



Willkommen auf slx-client

Mo 22 Juni 14:54

Benutzername:

Passwort:

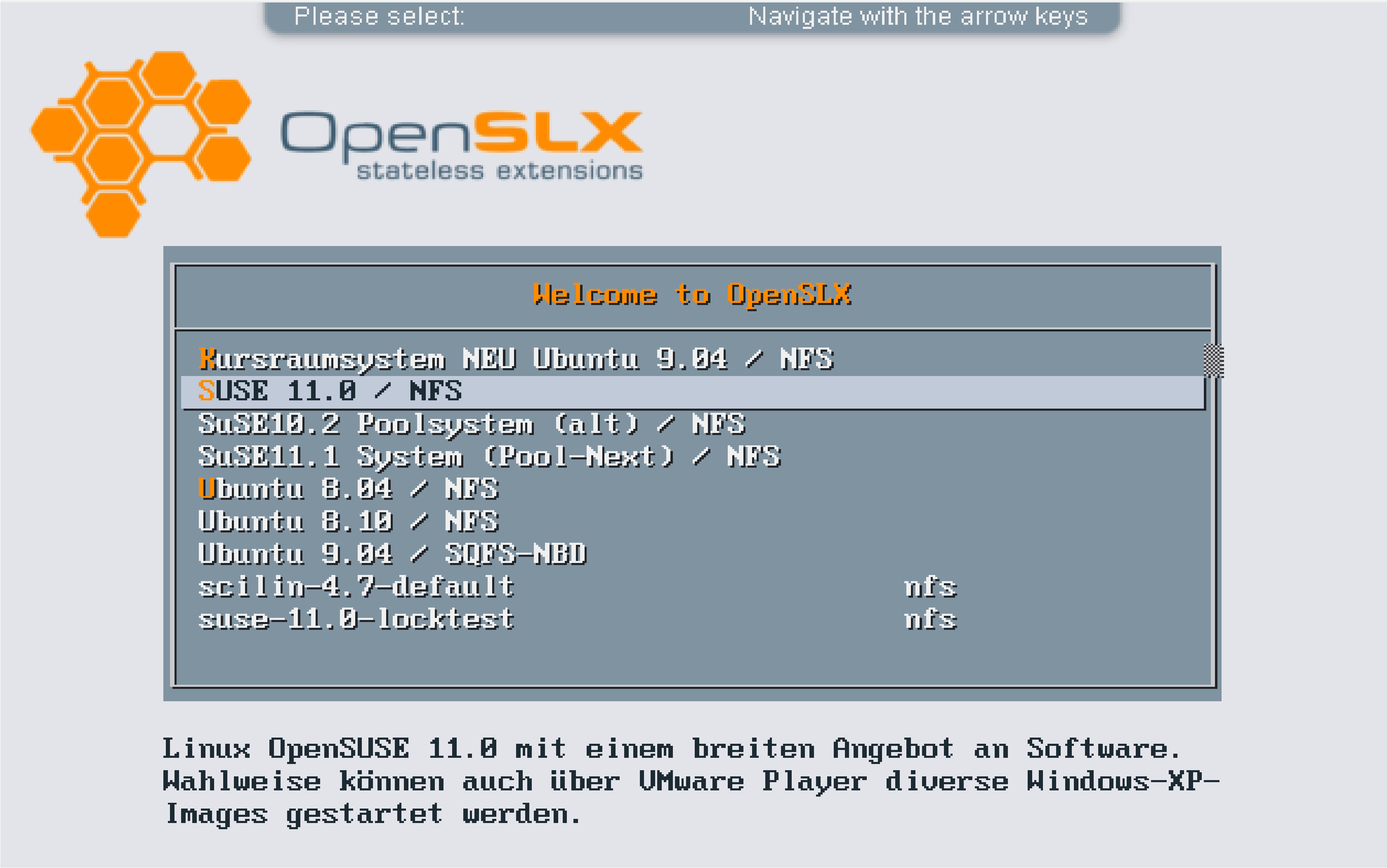
Structure: PXE Boot Menu

Configuring PXElinux

Boot: PXE Menu

- Using HPAs PXElinux suite
  - Really flexible:
  - Basic boot prompt (or just booting)
  - Offer sub menus, even on different TFTP servers
    - Standard menu
    - VGA menu
  - Add options like local boot, APM power off, installation of other OS via network

Boot: PXE Menu



```
Please select:                    Navigate with the arrow keys

OpenSLX
stateless extensions

Welcome to OpenSLX

Kursraumsystem NEU Ubuntu 9.04 / NFS
SUSE 11.0 / NFS
SuSE10.2 Poolsystem (alt) / NFS
SuSE11.1 System (Pool-Next) / NFS
Ubuntu 8.04 / NFS
Ubuntu 8.10 / NFS
Ubuntu 9.04 / SQFS-NBD
scilin-4.7-default                        nfs
suse-11.0-locktest                        nfs

Linux OpenSUSE 11.0 mit einem breiten Angebot an Software.
Wahlweise können auch über VMware Player diverse Windows-XP-
Images gestartet werden.
```

Practical Part

- Short break, then continue with
  - Network boot demonstration
  - DHCP configuration
  - TFTP setup
  - Simple PXE boot
  - More complex PXE menu setups
  - ...

Outlook

- Last lecture, Thursday
  - Double lecture starting 2pm again in Computer Lab #4
- Further practical part
  - Providing root filesystem via NFS or SquashFS on NBD
  - Demonstration of advanced booting via PreBoot environment
  - Configuring, extending PreBoot