Karlsruher Institut für Technologie

Diskless Linux für 100 Clients

Plattenloser Betrieb eines universitären PC-Pools

ATIS - Abteilung Technische Infrastruktur, Fakultät für Informatik

Gliederung

- Vorstellung
- Problemstellung / Motivation
- Voraussetzungen / Grundlagen
  - PXE-Boot
  - UnionFS bzw. auFS
- Die Umsetzung im Detail
  - Individualisierung mit Konfigurationsgruppen
- Zusammenfassung

Vorstellung

- KIT – Karlsruher Institut für Technologie
  - Fusion der Universität Karlsruhe und dem Forschungszentrum Karlsruhe
- ATIS – Abteilung technische Infrastruktur
  - zentraler IT-Dienstleister der Fakultät für Informatik
- Studentenpool mit über 100 Arbeitsplätzen, über 2.500 Accounts
  - Dual-Boot unter Windows und Linux (Fedora)
  - Windows Installation konventionell via WDS auf lokale HDD
  - Linux komplett via NFS ohne lokale HDD
- Vortragender:
  - Leiter des Bereich „IT-Dienste“ und stellv. Abteilungsleiter
  - Verantwortlich u.a. für den Linux-Betrieb im Pool

Motivation

- Verschiedene Deployment-Strategien wurden im Laufe der Zeit getestet:
  - Imaging à la Clonezilla
  - Automatisierte Installation via Anaconda/Kickstart
  - ...
- Nachteile
  - Langsam, fehleranfällig, erfordert Wartungsfenster mit Downtime
  - Aufsicht beim Deployment, Fehler durch defekte Hardware, Nacharbeiten
  - Windows Deployment „fummelt“ am MBR und der Partitionstabelle herum
  - Nachinstallationen / Patches via cron-Krücken
    - Nicht immer alle PCs laufen unter Linux
  - Installationen sind zu 99,9... % identisch
    - Hardwarekonfiguration wird unter Linux größtenteil beim Booten dynamisch angepasst

Zielvorstellungen

- Booten der Rechner übers Netz
- Betriebssystem via NFS vom Server
- Einfache Wartung
  - Alleinige Pflege **eines** Klientensystems in einer „chroot“-Umgebung auf dem Server (ein „Golden Image“)
- Vorhandene lokale Festplatte soll nicht angefasst werden
  - Windows kann sich darauf austoben
- **Kein** kastriertes Linux à la X-Terminal / Thin-Client bei dem fast alles im RAM und auf der CPU des Servers läuft
  - lokaler RAM und CPU der PCs sollen genutzt werden
- Die Rechner sollen *diskless* und *stateless* sein
  - `<RESET>` soll den originären Zustand wieder herstellen
- Heimatverzeichnisse (`/home`) und proprietäre, nicht-distributions-SW (`/opt`) kommen wie gewohnt via NFS

PXE-Boot:
oder „wie man sich am RJ45-Kabel aus dem Sumpf zieht“

PXE-Boot: Zutaten

- PXE: **P**re-boot E**x**ecution **E**nvironment
  - De-facto Standard von Intel um über das Netz/Netzwerkkarte einen Betriebssystemkernel zu booten
- Client
  - PXE-fähige Netzkarte
  - Im PC-BIOS muss der Netzwerkboot an erster Stelle stehen
- Server
  - DHCP-Server
  - tftp-Server
  - NFS-Server
  - DNS-Server

PXE-Boot: Ablauf

- PC bootet
- Netzkarte sendet DHCP-Request
- DHCP-Server antwortet mit IP-Parametern und der „next-server“-IP
- Client lädt vom „next-server“ das Bootmenü per tftp
  - User wählt Windows
    - Boot von der lokalen Festplatte
  - User wählt Linux
    - Client lädt per tftp den Kernel und die initrd und bootet
    - Beim booten wird „/“ via NFS gemountet

PXE-Boot: tftp Dialog

- Log-Auszug des tftp-Servers

```
RRQ from 141.3.8.242 filename pxelinux/pxelinux.0
RRQ from 141.3.8.242 filename pxelinux/pxelinux.cfg/2fb5a208-3f9b-dc11-409c-b5a30e436129
RRQ from 141.3.8.242 filename pxelinux/pxelinux.cfg/01-00-1b-38-92-fc-ee
RRQ from 141.3.8.242 filename pxelinux/pxelinux.cfg/8D0308F2
RRQ from 141.3.8.242 filename pxelinux/pxelinux.cfg/8D0308F
RRQ from 141.3.8.242 filename pxelinux/pxelinux.cfg/8D0308
RRQ from 141.3.8.242 filename pxelinux/pxelinux.cfg/8D030
RRQ from 141.3.8.242 filename pxelinux/pxelinux.cfg/8D03
RRQ from 141.3.8.242 filename pxelinux/pxelinux.cfg/8D0
RRQ from 141.3.8.242 filename pxelinux/pxelinux.cfg/8D
RRQ from 141.3.8.242 filename pxelinux/pxelinux.cfg/8
RRQ from 141.3.8.242 filename pxelinux/pxelinux.cfg/default
RRQ from 141.3.8.242 filename pxelinux/pxelinux.msc/german.kbd
RRQ from 141.3.8.242 filename pxelinux/vesamenu.c32
RRQ from 141.3.8.242 filename pxelinux/pxelinux.cfg/default
RRQ from 141.3.8.242 filename pxelinux/pxelinux.cfg/config/menugraphics.conf
RRQ from 141.3.8.242 filename pxelinux/pxelinux.msc/bootsplash.jpg

RRQ from 141.3.8.242 filename pxelinux/KERNEL/vmlinuz-f18
RRQ from 141.3.8.242 filename pxelinux/INITRD/initrd-f18.img
```

zuerst immer pxelinux.0
UUID des PCs
MAC-Adresse der Netzkarte
IP in HEX (141.3.8.242)

IP von hinten verkürzt

Immer noch nichts gefunden? „default“

„buntes“ Menü

User wählt Linux im Menü
Kernel
initrd

- Über die MAC / UUID / IP lässt sich steuern, welcher PC welches Menü bekommt
- Damit kann man mit „cron“ und Symlinks das OS individuell voreinstellen

PXE Bootmenü

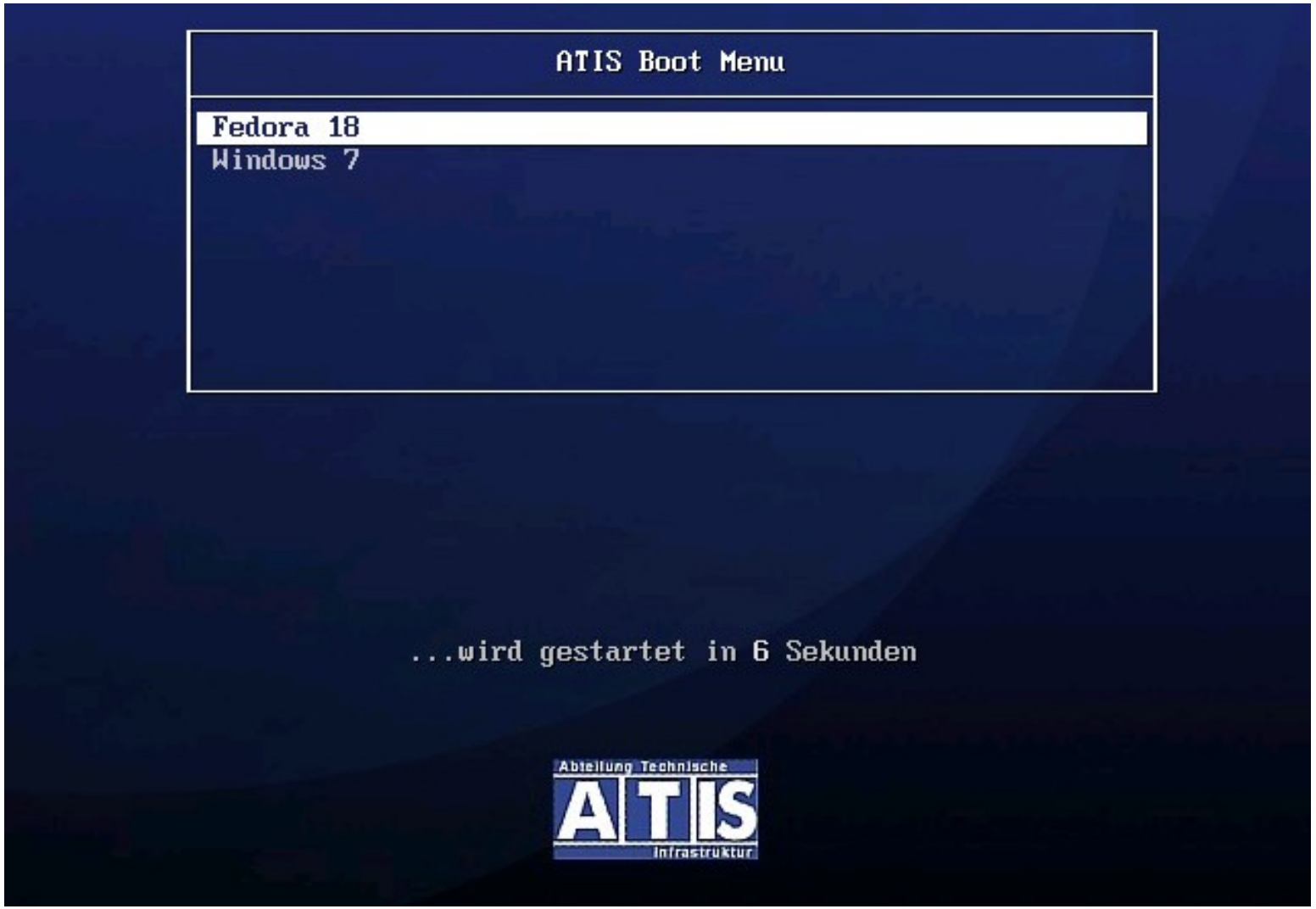
- **pxelinux.cfg/default:**

```
menu include pxelinux.cfg/config/menugraphics.conf
kbdmap pxelinux.msc/german.kbd
default vesamenu.c32    # this is a redirection to the graphic menu
prompt 0                # disables command prompt
allowoptions 0          # disallows any change of boot parameters
noescape 1              # disables the "hold down the Shift key“
timeout 100             # in 1/10th of seconds

label F18
    menu label Fedora 18
    kernel KERNEL/vmlinuz-f18
    append initrd=INITRD/initrd-f18.img root=NFSserver:/diskless/F18/root/

label windows
    menu label Windows 7
    localboot 0
```

PXE Bootmenü



PXE Boot

- Timer läuft ab oder Benutzer selektiert Linux
- Kernel und initrd werden via tftp geladen

```
RRQ from 141.3.8.242 filename pxelinux/KERNEL/vmlinuz-f18
RRQ from 141.3.8.242 filename pxelinux/INITRD/initrd-f18.img
```

- Via Kernelparameter wird dem Kernel das root-FS mitgegeben

```
append initrd=INITRD/initrd-f18.img root=NFSserver:/diskless/F18/root/
```

- Auf dem NFS-Server liegt unterhalb von `/diskless/F18/root` das „Golden Image“
- Done !

One Size Fits All !

Oder doch lieber individuell angepasste Arbeitsplätze ?

© www.f1online.de Bildnr./image no: 149694

Schwierigkeiten / Probleme (I)

- Schreibzugriffe
  - **Ein** Image via read/write-NFS für alle Clients ?
    - `/var` und sogar auch `/etc` (DHCP-Config !) werden beschrieben
    - Manchmal auch andere Stellen
  - Je Client ein Image ?
    - Skaliert nicht bei 100 Clients

Schwierigkeiten / Probleme (II)

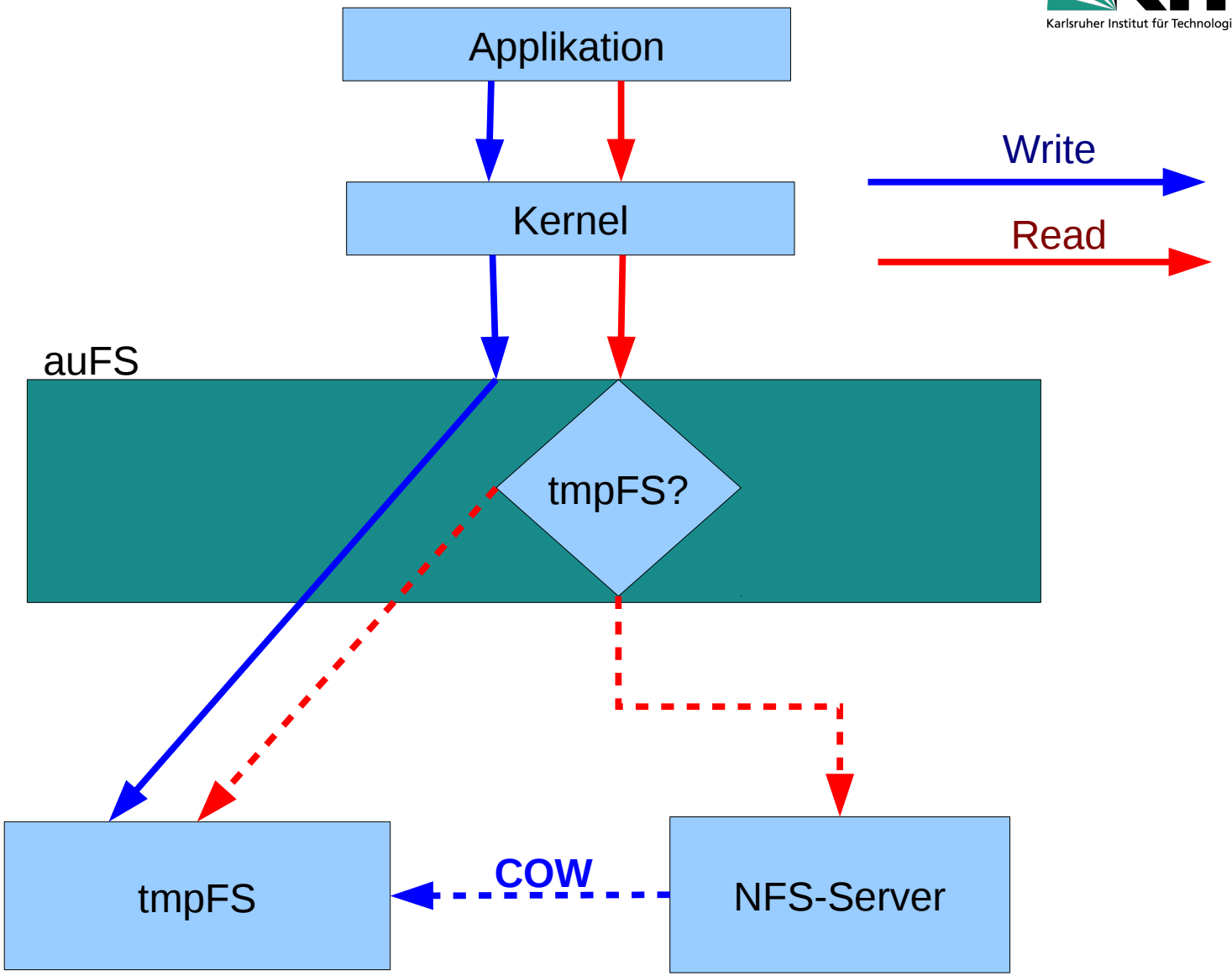
- Unterschiedliche Hardware-Konfigurationen
  - 4 verschiedene Generationen von F&L-PCs
  - Unterschiedliche Grafik-Karten (Intel, ATI und Nvidia)
  - Exoten-Lösungen
    - 3 Monitore
    - Wacom-Tablets
    - US-Tastatur
- Unterschiedliche Software-Konfigurationen
  - Netze mit unterschiedlichen Authentifizierungen
    - Unterschiedliche LDAP-Server / -Zweige
    - NIS
  - Rechner / Poolbereiche mit proprietärer Software
  - Bibliotheksrechner im Kiosk-Mode

Einschub: unionFS / auFS

- *unionFS* bzw. *auFS* sind Copy-On-Write-Filesystemtreiber, die sich zwischen Kernel und dem eigentlichen Filesystem (hier NFS) schieben
- Schreibzugriffe werden abgefangen und in eine RamDisk umgeleitet
- Lesezugriffe schauen zuerst in der RamDisk nach und danach auf dem NFS-Server
- Damit ist die ReadOnly-NFS-Freigabe aus Sicht des Clienten beschreibbar
- `<RESET>` vergisst alles wieder

- Findet auch Verwendung in den diversen Linux-Live-CDs
- unionFS
  - Alt und buggy, läuft im Userspace via fuse
- auFS (**a**nother **u**nion **F**ile **S**ystem)
  - Neuimplementierung im Kernelspace
  - z.B. Knoppix ab 5.1 (2007)

auFS


Applikation
Kernel
Write
Read
auFS
tmpFS?
tmpFS
COW
NFS-Server

Lösung mit auFS

- Mounten von `/var` via aufs
  - syslogd kann nach `/var/log` schreiben:
  - Logs landen im RAM
    - sind beim Reboot „weg“
    - zusätzliches Logging zu einem Remote-syslog-Server
- Hängt man statt `/var` gleich „`/`“ über auFS ein
  - kann man sogar lokal Pakete für Tests nachinstallieren
    - ...bis zum Reboot
  - und man hat die DHCP-Schreibzugriffe auf `/etc` auch abgefangen

Wie unterscheidet man Rechnerhardware / Netzumgebung / SW-Konfiguration ?

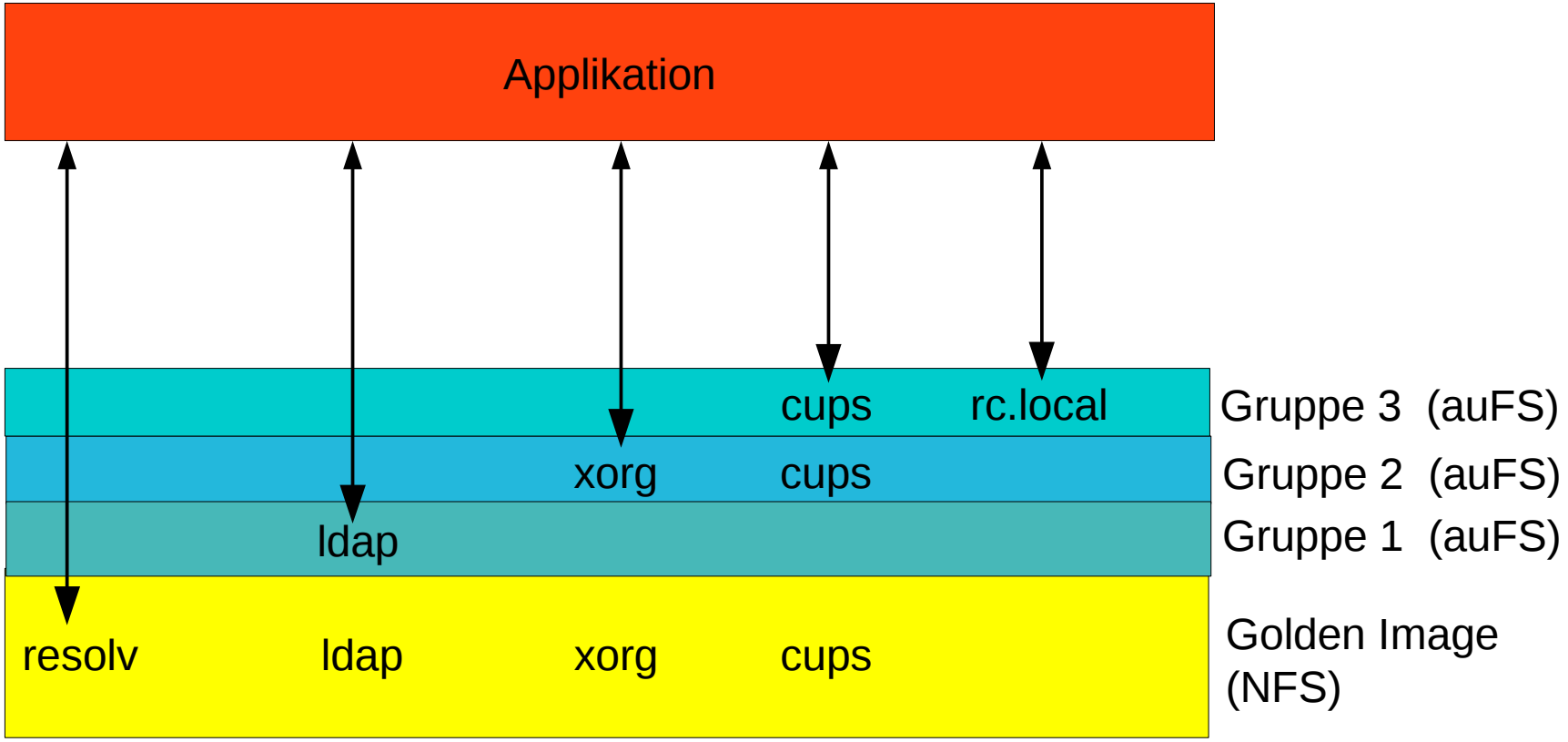
- Unterhalb von `/etc` liegt sowohl die Hardware- als auch die Netz- bzw. Software-Konfiguration

- Es muss also `/etc` pro Rechner unterschieden werden
  - Aber nicht **N** fast gleiche `/etc`'s für **N** Rechner
  - Die Unterschiede sind marginal

- Idee:
  - Nehme ein normales „default“ `/etc`
  - Erzeuge extra NFS-Verzeichnisse mit den Deltas
  - Kopiere diese Deltas beim booten über das Default-`/etc` drüber
  - „kopieren“ heisst in diesem Fall mounten per auFS

  - Jedes extra NFS-Verzeichnis definiert eine sog. ***Konfigurationsgruppe***

Konfigurationsgruppen



Konfigurationsgruppen

- NFS-Freigabe auf dem Server
  - `/diskless/F18/root/etc`
    `.../usr`
    `.../var`
    `...`

  } dies ist das „Golden Image“
- „Daneben“ liegen die Konfigurationsgruppen mit den Deltas zum Golden Image
- Aufbau einer Konfigurationsgruppe „conf_pool“
  - `/diskless/F18/conf/`**`conf_pool`**`/etc`
    `.../usr`
  - ist ein winziger Teilbereich des Filesystems mit den Deltas zum Golden Image
- Diese Gruppen werden mit auFS über das Golden Image gemountet

Konfigurationsgruppen: Beispiel Pool-PC

- Beispiel einer Gruppe „conf_pool“:

```
# find /diskless/F18/conf/conf_pool -maxdepth 2 -mindepth 2 -type d
/diskless/F18/conf/conf_pool/etc/snmp
/diskless/F18/conf/conf_pool/etc/security
/diskless/F18/conf/conf_pool/etc/openldap
/diskless/F18/conf/conf_pool/etc/sysconfig
/diskless/F18/conf/conf_pool/etc/cups
/diskless/F18/conf/conf_pool/etc/X11
/diskless/F18/conf/conf_pool/etc/pam.d
/diskless/F18/conf/conf_pool/etc/cron.daily
/diskless/F18/conf/conf_pool/etc/exim
/diskless/F18/conf/conf_pool/etc/sssd
/diskless/F18/conf/conf_pool/etc/ssh
/diskless/F18/conf/conf_pool/usr/local

# du -sh /diskless/F18/conf/conf_pool
1.8M   /diskless/F18/conf/conf_pool

# find /diskless/F18/conf_pool -type f | wc -l
84
```

Konfigurationsgruppen: Beispiel Mitarbeiter-PC

- Neben der Konfiguration für die Abteilung („conf_atis“) hier noch individuelle Zusatz-Gruppe für den Mitarbeiter-Arbeitsplatz-PC

- Beispiel einer Gruppe „conf_hopp“:

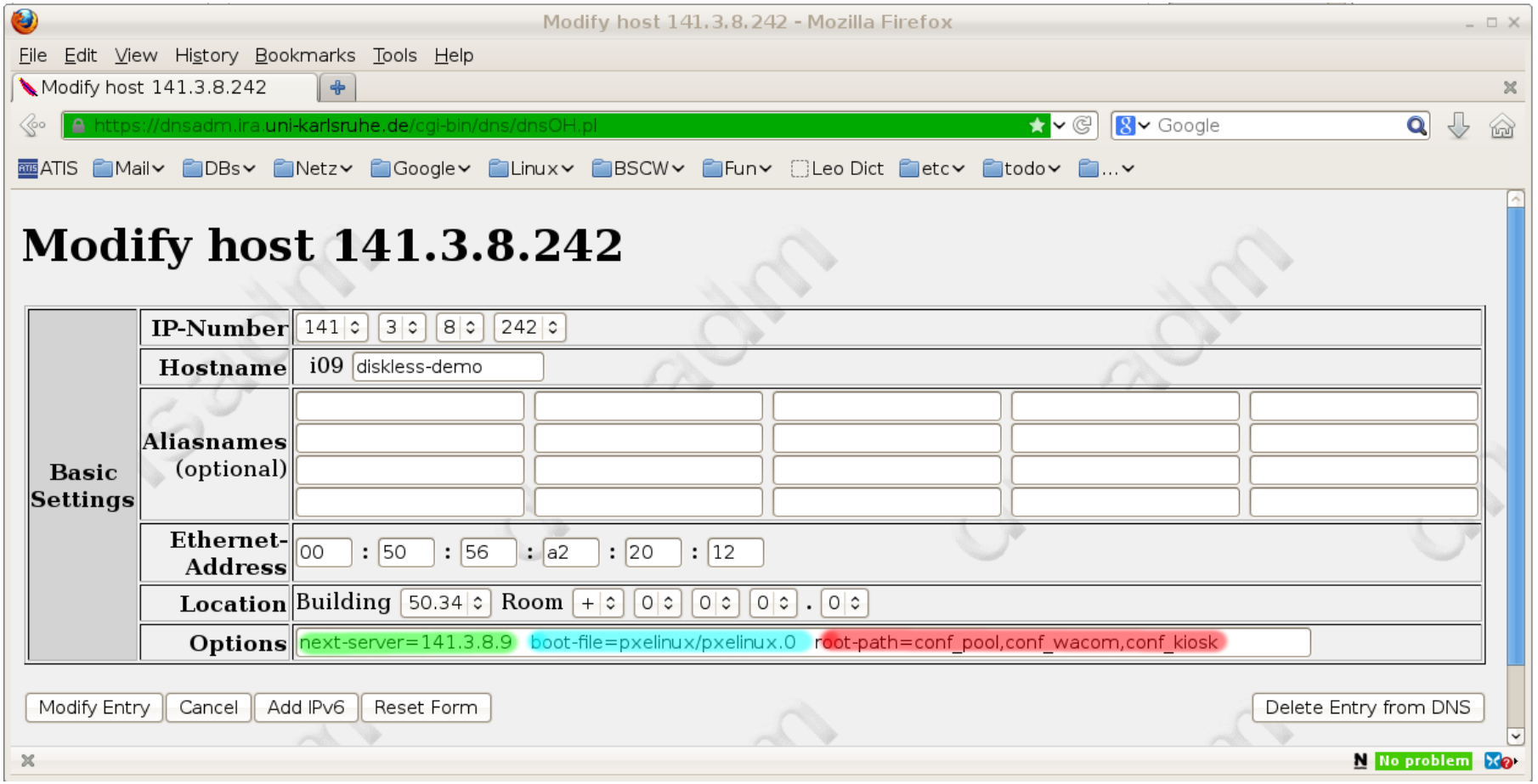
```
# find /diskless/F18/conf/conf_hopp/ -type f
/diskless/F18/conf/conf_hopp/etc/ssh/sshd_config
/diskless/F18/conf/conf_hopp/etc/locale.conf
/diskless/F18/conf/conf_hopp/etc/sysconfig/i18n
/diskless/F18/conf/conf_hopp/etc/sysconfig/keyboard
/diskless/F18/conf/conf_hopp/etc/sudoers
/diskless/F18/conf/conf_hopp/etc/rc.d/rc.local
/diskless/F18/conf/conf_hopp/etc/X11/xorg.conf
```

Wie bekommt der Client „seine“ Konfigurationsgruppe(n) ?

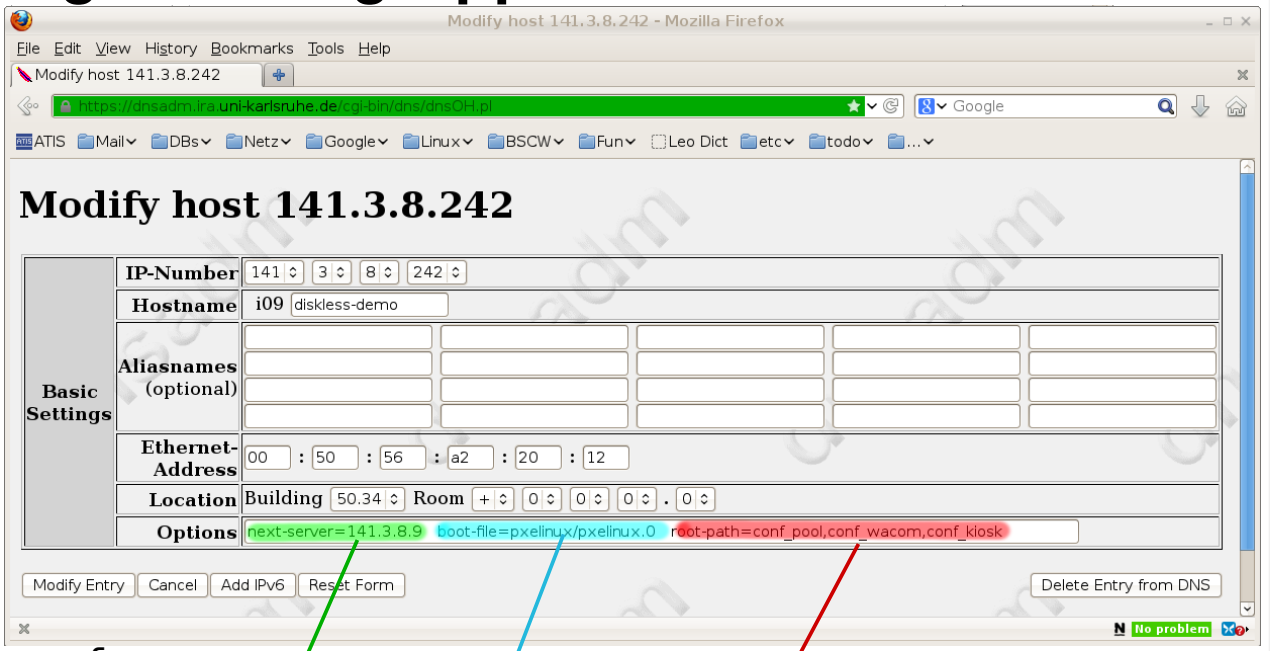
- Der DHCP-Server kann dem Client sog. Option-Strings übergeben
  - Diese benennen die jew. Konfigurationsgruppen des Clients
- Die Strings werden von der initrd abgefragt und dann die entsprechenden Gruppen über den „lokalen“ (NFS-) Dateibaum per auFS gemountet.
- Landen somit (wg. auFS) im RAM des Clients
  - Änderungen an den Gruppen zur Laufzeit werden i.d.R sofort aktiv
- Damit kann man auch Konfigurationen „stapeln“
  - d.h. mehrere Konfigurationsgruppen pro Client sind möglich
  - Jede Konfigurationsgruppe enthält nur die jeweils notwendige Rolle
  - Gleiche Dateien in einer „späteren“ Gruppen überschreiben ihre Pendants in einer vorherigen Gruppe

DNS / DHCP Verwaltungstool der Informatik

- Erstellt DNS- und DHCP-Konfigurationsdateien
- Ebenso die DHCP-Optionstrings



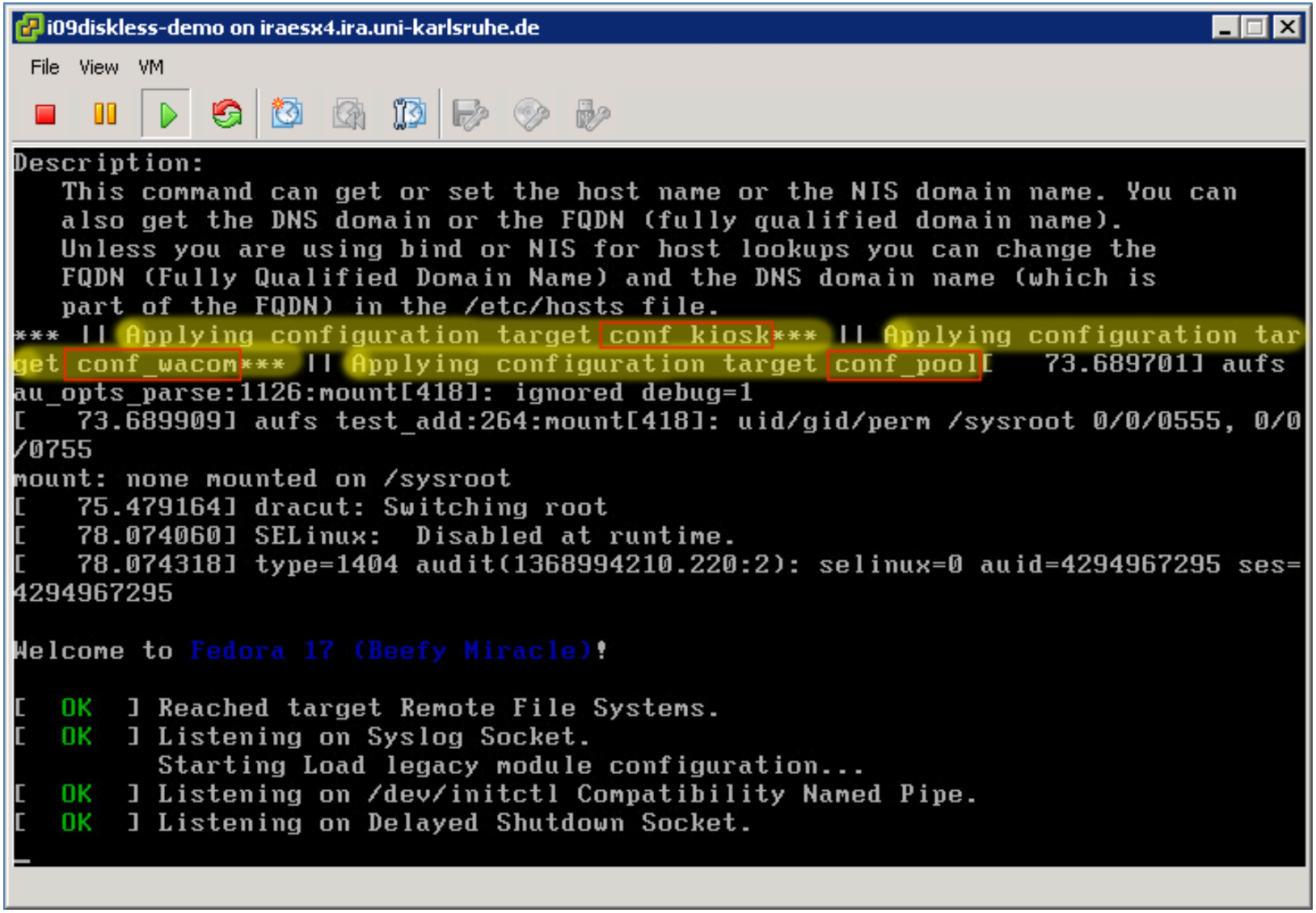
DNS-Tool und Konfigurationsgruppen



- **dhcpd.conf:**

```
host i09diskless-demo {
        hardware ethernet 00:50:56:a2:20:12;
        fixed-address i09diskless-demo.atis.uni-karlsruhe.de;
        option host-name "i08diskless-demo";
        option dhcp-client-identifier   "i09diskles-demo";
        option netbios-node-type 8;
        next-server 141.3.8.9;
        filename "pxelinux/pxelinux.0";
        option root-path "conf_pool, conf_wacom, conf_kiosk";
}
```

Client-Boot


i09diskless-demo on iraesx4.ira.uni-karlsruhe.de
File View VM
Description:
This command can get or set the host name or the NIS domain name. You can also get the DNS domain or the FQDN (fully qualified domain name). Unless you are using bind or NIS for host lookups you can change the FQDN (Fully Qualified Domain Name) and the DNS domain name (which is part of the FQDN) in the /etc/hosts file.
*** || Applying configuration target conf_kiosk*** || Applying configuration target conf_wacom*** || Applying configuration target conf_pool[73.689701] aufs au_opts_parse:1126:mount[418]: ignored debug=1
[73.689909] aufs test_add:264:mount[418]: uid/gid/perm /sysroot 0/0/0555, 0/0/0755
mount: none mounted on /sysroot
[75.479164] dracut: Switching root
[78.074060] SELinux: Disabled at runtime.
[78.074318] type=1404 audit(1368994210.220:2): selinux=0 auid=4294967295 ses=4294967295
Welcome to Fedora 17 (Beefy Miracle)!
[OK] Reached target Remote File Systems.
[OK] Listening on Syslog Socket.
Starting Load legacy module configuration...
[OK] Listening on /dev/initctl Compatibility Named Pipe.
[OK] Listening on Delayed Shutdown Socket.

Der Server im Backend ?

Der Server im Backend !

- NFS-Server: „Standard“ 2 CPUs, 4 Kerne, 8 GB RAM
  - Bedient Fedora-Diskless-Clients und die User-Homes per NFS / CIFS
  - NFS Parameter hochschrauben
  - Nur Last, wenn viele Clients gleichzeitig booten („iowait“)
  - Daten liegen im (iSCSI-) SAN
- Clients cachen die NFS-Daten recht intensiv, ebenso der NFS-Server

Alles ganz einfach, oder ?

- Im Prinzip schon, aber:
  - Bestehendes System ist in einem langjährigem evolutionären Prozess entstanden
  - Viel Hirnschmalz steckt in der initrd und den Konfigurationsgruppen
  - Muss bei neuen Major-Releases reviewed werden
    - Dracut mit den Hooks ist ein Segen !
  - auFS bei Fedora leider nicht OnBoard -> manuelles Kernelbacken
  - Die Wechsel des System-Starts bei Fedora von „SysV-init“ über „upstart“ zu „systemd“ erforderte Überarbeitung der Start-Scripte
    - Auch hier muss eingegriffen werden
      - Bsp.: „ifdown“ vom NetworkManager
    - Ebenso beim Shutdown

Wirklich plattenlos ?

- Nicht ganz
  - Windows braucht sie sowieso
  - Wenn vorhanden, verwendet Linux sie als Swap und temp. Scratch-Bereich
  - Es geht aber auch ganz ohne !

Zusammenfassung (I)

- Über 100 PCs im ATIS-Pool, plus diverse „sonstige“ PCs
  - ~50% booten Linux per Default
- Clients: mind. 2 GB, besser 4 GB RAM
- Erster Login nach dem Reboot in eine schwergewichtige 3D-Desktop-Umgebung wie Gnome3 ist etwas zäh
  - Wenn der Cache gut gefüllt ist, wuppt es
- Clients überleben den Reboot des Servers !
- Clients verhalten sich erratisch bei Updates von systemnaher Software
  - z.B. glibc, aber auch Firefox
  - Pool-PCs werden jede Nacht neu gebootet
  - „entleert“ auch die RamDisk

Zusammenfassung (II)

- Pflege in einer chroot-Umgebung auf dem Server
  - „yum install XYZ“ / „yum update“
  - Sofort auf allen Clients verfügbar
- Rollout einer neuen Fedora-Release: boot und gut 
- Über die Konfigurationsgruppen werden
  - Pool- und Mitarbeiter-Arbeitsplatz-PCs
  - Informatik-Bibliothek im Kiosk-Mode
  - eine Forschungseinrichtung
  - Verfügbarkeit unterschiedlicher Drucker
  - individuelle Hardware- und SW-Lösungen abgebildet
- Aktuelle Baustellen:
  - NFS4, Kerberos, auFS-Kernel, Versionierung via LVM-Snapshots

Vielen Dank für Ihre Aufmerksamkeit

Olaf.Hopp@kit.edu