Linux Desktop Management
with old school NFS-root boot

Christopher Huhn
C.Huhn@gsi.de
IT, Linux and HPC group
GSI Darmstadt

Hepix 2009, Berkeley

Outline

- Evolution of NFS-root system at GSI
- Technology of the current implementation and its strength
- Pitfalls
- Further directions

Operating System via Network

- Common concept for
  - X-Terminals, nowadays called Thin Clients
    - Diskless systems
  - Embedded systems
  - Support in the Linux-Kernel “since ever”
    - 1996: „NFS-Root-Client Mini-HOWTO“
- Consolidation effort in the days before sophisticated installation and configu-ration management infrastructure

Isn't it anachronistic?

- Thin clients are still a topic (e. g. LTSP)
- Crosslinks to virtualization issues:
  - OS image management
    - How do you make sure that OS instances are up to date on boot?
      - Might be a VM
      - Might also be a “forgotten” lab PC that was offline for a long time

GSI Linux desktops

- More a “fat“ client
  - Standard desktop PCs
  - Usually not diskless
  - Hardware zoo ranges from Pentium 3 to Nehalem server hardware
  - Some work as servers (NFS, SSH)
  - NFS-boot also used for batch worker nodes
- ~ 400 desktops running Debian Etch, Lenny in preparation

© Neil McGovern <neilm@debian.org>

“Classic” concept

- Booting via Network
  - Etherboot loader on floppy, kernel and network settings served via BOOTP
- NFS-root filesystem
  - Mounted via kernel subsystem
  - Dedicated root file system for each client
  - /usr shared between all clients and their OS server
    - Read-only for the clients
    - Client and Server OS not independent

Current concept

- Network boot via PXE, kernel + initrd via TFTP
- read-only OS image shared between clients
  i.e. no write access to / for the clients
- Distinct from OS server installation
  - OS servers are plain NFS servers (no HA)
  - Moving Clients from one server to another only requires DHCP reconfiguration

tmpfs and unionfs

- Some areas of the FS must still be writable
- Implementation relies on special purpose file systems:
  1)tmpfs: automatically resizing ramdisk
  2)unionfs/aufs: stackable copy-on-write file system with semi-transparency
- Changes to standard Debian installation mainly consists of 2 init scripts

© Savannah Grandfather
http://www.flickr.com/photos/savannahgrandfather/364949591

Init-script #1

- As early as possible
- /local: tmpfs
- /etc: tmpfs + NFS unification
  - Files checked out from central SVN-repository
    - fstab, X config, SSH host keys, ...
- /root, /media on tmpfs

Init-script #2

- Immediately after mounting of local disks
- /tmp: local partition or tmpfs
- /var: local partition or tmpfs
  + NFS unification

Putting it all together

PXE	✔Basic network configuration ✔Load boot menu ✔Load kernel and initrd, start kernel
Kernel	✔Extract and start initial ramdisk
Initrd	✔Gather additional boot options from DHCP ✔Mount the root file system and start init
Init	✔**Setup tmpfs and unions for /etc, /root, /media** ✔Mount local partitions ✔**Setup union for /var** ✔Start the DHCP client ✔...

Image creation

- `fai dirinstall ...`
- Almost identical to bare metal install
- No “golden image” copied from server to server
  - Implicit configuration by ad-hoc administration
    - Often undocumented
  - → Mutation problems

Client configuration management

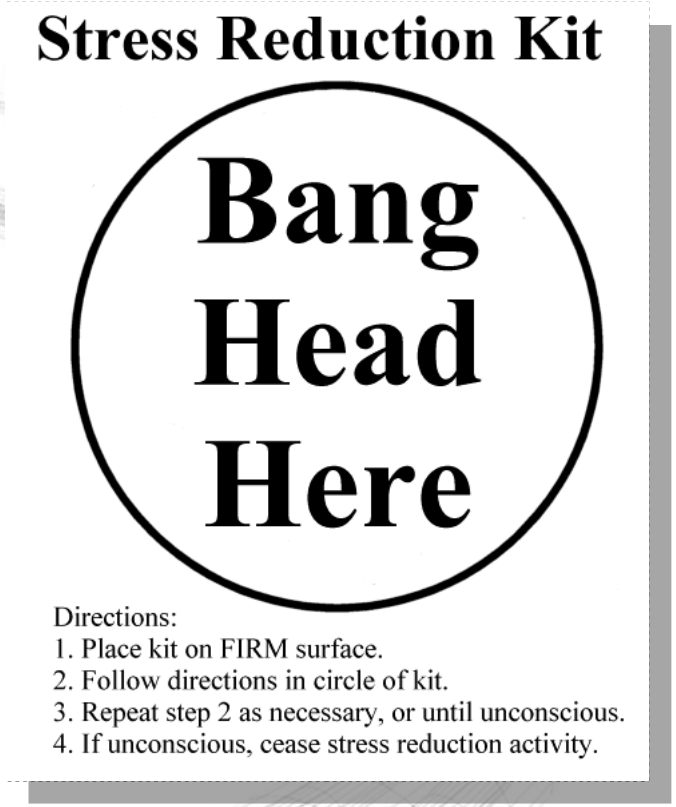
- Configuration management (cfengine/cron)
  1) Tasks on OS server outside client image (mainly server configuration)
  2) Tasks to run on OS server inside client images (settings for all clients)
     - No manual ad-hoc administration here!
  3) Tasks to run on clients directly
     - e.g. access control rules
- Standard cron jobs must be split:
  - Logfiles must be rotated on the clients
  - but updatedb should run on the server

Strengths of NFS-root concept

- Advanced security
  - Read-only system installation
  - „Offline“ security updates
- Tight control of running systems
- Fast OS deployment
- Fast upgrades
- Potential diskless operation
  - Works nicely as a rescue system

Drawbacks

- Totally network dependent
  - But: Can you work with a standalone desktop when the network is broken?
- Unification file system woes
  - esp. background updates
- Service restarts after upgrades
  - Libc upgrade requires ssh restart on all clients
- Permission mismatch problems
  - Permissions very important on /var
  - Unionfs copyup bugs
  - System account name – UID mappings may be different after OS change
    - Dynamically assigned at installation time



Outlook

- Replace 32bit compatibility layer (chroot) on 64bit by complete 32bit NFS-root
- Alternatives to NFS (v3):
  - NFS v4, Live-CD/DVD, HTTP, Lustre, ...
- HA cluster for OS servers
  - Load-balanced / active-active
- Netboot NFS-root VMs: GSI Linux Desktop inside Windows/MacOS, offsite?
- Secure (PXE-)Boot process
- This is no concept for mobile computers

Questions?

Thank you for your attention!