Customizing the Initial RAM Disk

LPIC-2: Linux Engineer (201-450)

Objectives:

At the end of this episode, I will be able to:

1. Describe the purpose and functionality of the Init Ram Disk (initrd).
2. Unpack, modify, and repackage the initrd.
3. Modify the grub bootloader to use a custom initrd.

> Additional resources used during the episode can be obtained using the download link on the overview episode.

- Init Ram Disk (initrd)
  - Contains files needed to boot the system
  - Miniature Operating System
    - Usually around 50MB
  - initrd is present during boot
  - Removed after boot completes
- Modifying the initrd
  - Kernel modules are loaded at boot time
  - Not all modules are present in the initrd
  - Most are stored in the root partition
  - Causes problems if you need to load drivers right away
    - Storage drivers
    - NIC/iSCSI
- Locating the initrd
  - Red Hat Based Systems
    - `/boot/initramfs`
    - `/boot/initramfs-4.18.0-193.14.3.el8_2.x86_64.img`
    - Generate the default initrd
      - `mkinitrd`
      - `mkinitrd /boot/initramfs-<version> <version>`
      - `mkinitrd /boot/initramfs-4.18.0-193.14.3.el8_2.x86_64.img 4.18.0-193.14.3.el8_2.x86_64`
      - Use `$(uname -r)` for the current version
  - Debian Based Systems
    - `/boot/initrd.img`
    - `/boot/initrd.img-5.8.0-55-generic`
    - Generate the default initrd
      - `mkinitramfs`
      - `mkinitramfs -o /boot/initrd-<version>.img <version>`
      - `mkinitramfs -o /boot/initrd-$(uname -r) $(uname -r)`
- Examining the initrd contents
  - Compress in an unusual format

        - cpio for the microcode
        - gzip for the rest
    - Extracting the contents
        - `unmkinitramfs /boot/initrd.img-$(uname -r) initramfs/`
    - Folder structure
        - `/initramfs`
            - `/early - x86_64 microcode (AMD)`
            - `/early2 - x86 microcode (Intel)`
            - `/main/lib/firmware`
            - `/main/lib/modules`
- Modifying and Repackaging the initrd
    - `sudoedit /etc/initramfs-tools/modules`
    - Add the appropriate module names
        - `raid10`
        - `btrfs`
    - Update the current initramfs
        - `update-initramfs -u`
    - Create a new initramfs in a different location
        - `update-initramfs -u -b ~/`
    - Create a new initramfs for a specific version
        - `update-initramfs -c -k 5.8.0-55-generic -b ~/`
        - Check `/lib/modules/*` for the version format
- Updating GRUB
    - Different configurations for different distros
      `/boot/grub/menu.lst`
      `/boot/grub/grub.conf`
      `/etc/default/grub`
- Example: Ubuntu
    - Menu is automatically generated at boot
    - Uses scripts in `/etc/grub.d` to build
    - Copy an existing entry
        - `less /boot/grub/grub.cfg`
        - Search for `menuentry`
    - `sudoedit /etc/grub.d/40_custom`
    - Push the Update
        - `sudo update-grub`
        - `sudo grub-mkconfig`
            - Displays potential config
        - `sudo grub-mkconfig -o /boot/grub/grub.cfg`
            - Installs the config

```
menuentry 'Ubuntu (Init Test)' --class ubuntu --class gnu-linux --class gnu --class os
$menuentry_id_option 'gnulinux-simple-a8af722f-c40a-4f72-844b-8afbbaa6b742' {
        recordfail
        load_video
        gfxmode $linux_gfx_mode
        insmod gzio
        if [ x$grub_platform = xxen ]; then insmod xzio; insmod lzopio; fi
        insmod part_msdos
        insmod ext2
        insmod raid10
        set root='hd0,msdos5'
        if [ x$feature_platform_search_hint = xy ]; then
          search --no-floppy --fs-uuid --set=root --hint-bios=hd0,msdos5 --hint-
efi=hd0,msdos5     --hint-baremetal=ahci0,msdos5  a8af722f-c40a-4f72-844b-8afbbaa6b742
        else
          search --no-floppy --fs-uuid --set=root a8af722f-c40a-4f72-844b-8afbbaa6b742
        fi
        linux   /boot/vmlinuz-5.8.0-55-generic root=UUID=a8af722f-c40a-4f72-844b-8afbbaa6b742
r    o find_preseed=/preseed.cfg auto noprompt priority=critical locale=en_US quiet
        initrd  /boot/initrd.img-5.8.0-55-don
}
```