Assignment #2: Booting(2)

Emil Sharifulllin, Innopolis University September 22, 2016

1 Loading the OS

1.1 What is an UEFI OS loader and where does the Ubuntu OS loader reside on the system?

An UEFI OS loader is a special firmware that defines boot device and starts OS loader from special partition on this device. Ubuntu OS loader must be placed at <UEFI System Partition>/efi/boot/. At my system OS loader is stored at ls /boot/grub/

```
$ ls /boot/grub
fonts  grub.cfg  grubenv  locale  unicode.pf2  x86_64-efi
```

1.2 Describe in order all the steps required for booting the computer (until the OS loader starts running.)

1. System started, POST check running.
2. UEFI is loaded from ROM.
3. UEFI starts load manager that defines what kind of applications to load.
4. Firmware runs UEFI application with FAT32 filesystem from UEFISYS partition.
5. UEFI application runs kernel and initramfs.

To show parameters of boot order we can use efibootmgr

```
$ efibootmgr
BootCurrent: 0000
Timeout: 0 seconds
BootOrder: 0000,0007,0001,0002,0003,0008,0004,0005,0006
Boot0000* ubuntu
Boot0001* DTO UEFI USB Floppy/CD
Boot0002* DTO UEFI USB Hard Drive
Boot0003* DTO UEFI ATAPI CD-ROM Drive
Boot0004* CD/DVD Drive
Boot0005* DTO Legacy USB Floppy/CD
Boot0006* Hard Drive
Boot0007* IP4 Intel(R) Ethernet Connection I217-LM
Boot0008* IP6 Intel(R) Ethernet Connection I217-LM
```

1.3 What is the purpose of the GRUB boot loader in a UEFI system?

GRUB loader can load OS from filesystems that can be not supported by UEFI as example NTFS. Also GRUB can show you beautiful wallpaper.

1.4 How does the UEFI OS loader load the GRUB boot loader?

Firstly UEFI loader searches special NVRAM variable that contains a pointer to the hardware device and to a file on that hardware device that contains the UEFI image with GRUB. After that UEFI loader call this loader, this process is called boot chaining. But this process cannot be possible if Secure Boot is enabled. For Secure Boot systems we need to install middleware loader for example Shim or PRELoader. This loaders allow you to run signed linux kernel code.

1.5 Explain how the GRUB boot loader, in turn, loads and run the kernel by answering these 3 questions:

1.5.1 What type of filesystem is the kernel on?

It can be any filesystem that can be supported by OS loader. In my case there is ext4.

```
$ df -T | awk '{print $1,$2,$NF}' | grep "^/dev"
/dev/sda2 ext4 /
/dev/sda1 vfat /boot/efi
```

1.5.2 What type(s) of filesystem does UEFI support?

UEFI defines a specific version of the FAT file system, which is maintained as part of the UEFI specification and independently from the original FAT specification, encompassing a variant of the FAT32 file system on ESPs, and FAT16 and FAT12 file systems on removable media. ESP is recommended to be at least 512 MiB in size and formatted with a FAT32 filesystem for maximum compatibility.

1.5.3 What does the GRUB boot loader therefore have to do to load the kernel?

Grub will find the /boot/grub/menu.lst which configures its interactive menu. The location of the menu.lst, as well as stage1.5 and stage2 files, is hard-coded into grub when it is installed to the boot sector. Grub locates and loads the kernel and kernel run /sbin/init or systemd. GRUB also can load kernel from different types of filesystems not only fat. GRUB have special UEFI applicatios which allows GRUB to work with different filesystems

1.6 Do you need an OS loader and/or boot loader to load a Linux kernel with UEFI? Explain why or why not.

It is not necessary to load GRUB loader, because UEFI can load OS if it is stored at ESP partition.

1.7 How many parts (or stages) does GRUB have in such a system, and what is their task?

1. boot.img is stored in the master boot record (MBR) or optionally in any of the volume boot records (VBRs), and addresses the next stage by an LBA48 address at installation time it is configured to load the first sector of core.img.
2. core.img is by default written to the sectors between the MBR and the first partition, when these sectors are free and available. For legacy reasons, the first partition of a hard drive does not begin at sector 1 (counting begins with 0) but at sector 63, leaving a gap of 62 sectors of empty space. That space is not part of any partition or file system, and therefore not prone to any problems related with it. Once executed, core.img will load its configuration file and any other modules needed, particularly file system drivers; at installation time, it is generated from diskboot.img and configured to load the stage 2 by its file path.
3. Stage 2: files belonging to the stage 2 are all being held in the /boot/grub directory, which is a subdirectory of the /boot directory specified by the Filesystem Hierarchy Standard (FHS).

1.8 Where are the different stages found on the disk?

All UEFI loader images must be placed at ESP at folder /boot/

2 Initializing the OS

2.9 Describe the entire startup process of Ubuntu 16.04 in the default installation. The subquestions below are leaders to help you along, they must be answered but by no means represent the entire startup process of Ubuntu...

The entire process of booting is following:

- Firmware starting. It works and then calls OS loader
- OS loader loads kernel from disk to RAM and then starts kernel
- Kernel should run initializer software which will start all daemons in particular order and with dependencies. In modern Linux based operating system as an initializer usually runs systemd.

This process you can see on plot, thiat is connected with this pdf file.

systemd is a special daemon that kernel calls immediatelly after loading itself. systemd is a modern version of init.d. systemd improve booting of services with parallel loading.
But with parallel starting become new problem: how to require order of loading. systemd contain mechanism of dependencies. Any unit may want or require one or more other units before it can run. These dependencies are set in unit files with the directives Wants and Requires.

2.9.1 What is the first process started by the kernel?

The first process is an /sbin/init(on older systems) or a systemd daemon that starts all other daemons.

2.9.2 Where is the configuration kept for the started process?

Configuration of systemd can be stored in many files that is called units. Units can be stored at three folders:

1. **/usr/lib/systemd/system/** - units of software, which is installed from package managers(APT/RPM/YUM

2. **/run/systemd/system/** - runtime defined units

3. **/etc/systemd/system/** - units defined by system administrator

2.9.3 It starts multiple processes. How is the order of execution defined?

In unit file at **Unit** section boot order can be defined with using **After** command. It defines after which processes certain process should be run.

2.10 To understand the workings of daemons in Ubuntu, we are going to take a closer look at one aspect of the booting process: networking. Please describe the workings of Ubuntu desktop here (Ubuntu server networking is actually simpler):

2.10.1 How is the networking started? Include the names of all configuration files and utilities.

To enable networking systemd starts NetworkManager. This is software that manages network connections. NetworkManager can be seted up with /etc/network/interfaces file. This file contains information about all connections.

To see how it starts lets call journalctl| grep -i network

```
$ journalctl| grep -i network
Sep 22 15:48:04 sne-litleleprikon kernel: FUJITSU Extended Socket ↩
    Network Device Driver - version 1.0 - Copyright (c) 2015 FUJITSU ↩
    LIMITED
```

```
Sep 22 15:48:04 sne-litleleprikon kernel: e1000e: Intel(R) PRO/1000 Network Driver - 3.2.6-k
Sep 22 15:48:04 sne-litleleprikon kernel: e1000e 0000:00:19.0 eth0: Intel(R) PRO/1000 Network Connection
Sep 22 15:48:14 sne-litleleprikon systemd[1]: Starting Network Time Synchronization...
Sep 22 15:48:14 sne-litleleprikon systemd[1]: Started Network Time Synchronization.
Sep 22 15:48:15 sne-litleleprikon audit[779]: AVC apparmor="STATUS" operation="profile_load" profile="unconfined" name="/usr/lib/NetworkManager/nm-dhcp-client.action" pid=779 comm="apparmor_parser"
Sep 22 15:48:15 sne-litleleprikon audit[779]: AVC apparmor="STATUS" operation="profile_load" profile="unconfined" name="/usr/lib/NetworkManager/nm-dhcp-helper" pid=779 comm="apparmor_parser"
Sep 22 15:48:15 sne-litleleprikon kernel: audit: type=1400 audit(1474548495.543:4): apparmor="STATUS" operation="profile_load" profile="unconfined" name="/usr/lib/NetworkManager/nm-dhcp-client.action" pid=779 comm="apparmor_parser"
Sep 22 15:48:15 sne-litleleprikon kernel: audit: type=1400 audit(1474548495.543:5): apparmor="STATUS" operation="profile_load" profile="unconfined" name="/usr/lib/NetworkManager/nm-dhcp-helper" pid=779 comm="apparmor_parser"
Sep 22 15:48:15 sne-litleleprikon systemd[1]: Started Trigger resolvconf update for networkd DNS.
Sep 22 15:48:15 sne-litleleprikon systemd[1]: Starting Raise network interfaces...
Sep 22 15:48:17 sne-litleleprikon avahi-daemon[831]: Network interface enumeration completed.
Sep 22 15:48:17 sne-litleleprikon systemd[1]: Starting Network Manager...
Sep 22 15:48:17 sne-litleleprikon systemd[1]: Started Raise network interfaces.
Sep 22 15:48:18 sne-litleleprikon NetworkManager[890]: <info> [1474548498.7910] NetworkManager (version 1.2.0) is starting...
Sep 22 15:48:18 sne-litleleprikon NetworkManager[890]: <info> [1474548498.7913] Read config: /etc/NetworkManager/NetworkManager.conf (etc: default-wifi-powersave-on.conf)
Sep 22 15:48:18 sne-litleleprikon NetworkManager[890]: <info> [1474548498.9759] manager[0xf1c1a0]: monitoring kernel firmware directory '/lib/firmware'.
Sep 22 15:48:18 sne-litleleprikon NetworkManager[890]: <info> [1474548498.9759] monitoring ifupdown state file '/run/network/ifstate'.
Sep 22 15:48:19 sne-litleleprikon NetworkManager[890]: <info> [1474548499.2517] dns-mgr[0xf2d8b0]: set resolv-conf-mode: dnsmasq, plugin="dnsmasq"
Sep 22 15:48:19 sne-litleleprikon NetworkManager[890]: <info> [1474548499.2517] dns-mgr[0xf2d8b0]: using resolv.conf manager 'resolvconf'
Sep 22 15:48:19 sne-litleleprikon whoopsie[896]: [15:48:19] Could not get the Network Manager state:
Sep 22 15:48:19 sne-litleleprikon whoopsie[896]: [15:48:19] GDBus.Error:org.freedesktop.DBus.Error.ServiceUnknown: The name org.freedesktop.NetworkManager was not provided by any .service files
Sep 22 15:48:19 sne-litleleprikon systemd[1]: Started Network Manager.
Sep 22 15:48:19 sne-litleleprikon systemd[1]: Reached target Network.
Sep 22 15:48:19 sne-litleleprikon systemd[1]: Starting Network Manager Script Dispatcher Service...
Sep 22 15:48:19 sne-litleleprikon systemd[1]: Started Network Manager Script Dispatcher Service.
Sep 22 15:48:19 sne-litleleprikon NetworkManager[890]: <info> [1474548499.7972] init!
```

2.10.2 Which process manages the networking configuration?

For this purposes is used NetworkManager

2.11 As a final installation step, make your experimentation server reachable over IPv6.