Technical Note: EFI & GPT install of Debian & CentOS

Toshikazu Aiyama
lab.aiyama@gmail.com

A complete procedure from a scratch personal computer(PC) system is described finally arriving fairly modern Unified Extensible Firmware Interface (UEFI), and Globally unique identifiers Partition Table (GPT) based multiple 64-bit linux distribution bootable systems. Basic set up for the firmware system is described first, then two typical linux distributions install are illustrated as examples. Several configuration files are also attached to ease the install process.

Subject classification: boot loader, elilo, GPT, UEFI, linux, debian, CentOS

Introduction

We will limit our discussion to 64-bit x86 Personal Computer (PC) systems with Unified Extensible Firmware Interface (UEFI) based Read-Only-Memory (ROM). Thus 16-bit, 32-bit, non-x86, or non-UEFI systems should be consulted elsewhere. The installation of a linux Operating Systems (OS)to a traditional BIOS (Basic Input/Output System) MBR (Master Boot Record) based ROM is very well documented elsewhere also. Most of these previous methods imply less than two tera bytes (TB) restriction on a bootable hard disk drive (HDD) size due to MBR data definitions.

UEFI systems are still very buggy mainly because their usages are very limited, and partly because they are not used widely even the most of the modern PC ROMs are UEFI based; thus a lot of bugs are left unexamined. Unless you use larger than 2TB HDD to bootstrap the system, you do not need to use UEFI boot capability. Even you use greater than 2TB HDD, however, there exist some ways to boot up the system, but we will not discuss these techniques here; thus you should consult about them elsewhere.

The traditional MBR data resides at the first 512-byte sector of a bootable HDD. Within 512-byte MBR data, there are four primary partition table entries of 16 bytes each. Even this data area is extremely important to any PCs, the traditional MBR based system sadly misses any backup mechanism of the table as a standard.

Just the MBR partition table is for the traditional BIOS ROM, the Globally unique identifiers Partition Table (GPT) is for the new UEFI ROM. The MBR partition table has allowed up to four primary partitions, on the other hand the GPT allows up to 128 primary partitions with each of its data occupying 128 bytes each. Towards the ends of any GPT disk, it has a complete backup data of GPT partition entries data. At this point of time, linux can recognize 15 partitions; thus we do not know the use of partitions beyond 15th partitions in linux. But we can partition an HDD up to 128 partitions anyway.

There are two widely used linux boot loaders: grub, and lilo. Grub (GRand Unified Bootloader) is the default bootloader of most of linux distribution, and it is well-documented elsewhere. On the

other hand, lilo (LInux LOader) is not used widely used now. And elilo is a boot loader for UEFI systems. Elilo's development may seem to be ceased for the time being.

Debian, and RedHat are two of the most famous linux distributions. Debian is a widely used linux distribution. Ubuntu is derived from debian. Both debian and ubuntu is a free distribution. Nowadays ubuntu is widely installed due to the ease of both its install, and its use.

RedHat Enterprise Linux (RHEL) is another widely used distribution. It is a paid subscription based distribution, and is widely used in linux commercial application platform. CentOS is derived from RedHat Enterprise Linux. CentOS is a free distribution, and supposed to be fully compatible with RHEL.

First we will briefly review the literature of PC's basic hardware, its associated firmware, linux, and its surrounding system software. Next, we will discuss the preparation of the whole installation process. Then we will present two linux distribution installation procedures. Finally, we describe elilo boot loader configuration for rebooting a respective distribution OS.

Literature

The IBM PC is released in 1981. It is the first PC marketed by then a computer giant IBM. Most of its parts are off the shelf from other original equipment manufacturers. Later IBM PC/AT is announced 1984. IBM took open architecture showing off its hardware design in detail, for example, IBM PC/AT Technical Reference (1984). There we can find schematic diagrams, and ROM source codes. Its architecture is well cloned in detail in modern PCs. Most of various hardware aspects of IBM PCs are very well documented by Hogan (1991).

MBR occupy the the very first sector of a disk whether it is a floppy diskette, or a hard disk. It is 512 bytes data of the following three sections:

1. Bootstrap code area.
2. Data area showing four primary partitions' locations, and sizes.
3. Signature area.

This data structure has accommodated various technological need thus far. But the recent ever increasing size of HDD finally necessitates a new standard.

UEFI is originally developed by Intel for its Itanium boot loader as Extensible Firmware Interface (EFI). The UEFI can be thought of a specification that defines a software interface between an operating system and PC hardware's firmware. We can access UEFI specifications at `http://uefi.org/specifications`. It mimics basic functioning of DOS (disk operating system).

GPT is a part of the new standard: UEFI. It is a disk oriented table replacing BIOS MBR table. GPT shows the structure of HDD, and its associated partition table.

Linux is a micro kernel originally developed by Linus Torvalds. You can find the development of it in Linux Kernel (`https://www.kernel.org/`). It is free, and open source software. To create a full fledged complete OS, we need other system softwares often obtained from GNU (`http://www.gnu.org/`).

Debian (`https://www.debian.org/`) is a well-established free Unix compatible OS which consists mostly GNU system software, and micro kernel linux. It has been one of the most difficult distribution to install, but it is getting easier than previously. Debian is the upstream distribution of still another popular linux distribution: ubuntu (`http://www.ubuntu.com/`).

CentOS (Community ENTerprise Operating System, `http://www.centos.org/`) is another free open source OS. It is derived from upstream, and compatible RedHat Enterprise Linux (RHEL, `http://www.redhat.com/en`) widely used in business. Now RedHat has one distribution for development: Fedora, and another for maintenance: CentOS.

Lilo (`http://lilo.alioth.debian.org/`) is a boot loader for MBR based Linux OS, and elilo (`http://sourceforge.net/projects/elilo/`)is for UEFI. Nowadays lilo, and elilo development seems to be ceased.

GRUB (`http://www.gnu.org/software/grub/`)is one of the most popular multi-boot capable boot loader including non-linux OS, Windows. Most current linux distribution uses it as a default boot loader. It is capable to support both MBR, and UEFI boot.

Next we will start the preparation process of leading to linux OS install.

Assumptions

Before we start the actual install of a linux distribution, we make all the necessary items explicit, in advance. At minimum, we need the following.

- UEFI bootable 64-bit x86 PC with some reasonable size of memory, an HDD, and internet connection.
- Three USB stick memory. First one is used for UEFI bootable stick, and its size must be 128MB or more. Second one is used for CentOS7 install image, and its size must be 256MB or more. Third one is used for SystemRescueCD image, and its size must be 512MB or more. Note that we do not use a formal debian installer image; thus we do not need another USB stick for it. If you need one, you need additional stick of 256MB or more.
- An environment where you can create SystemRescueCD on a USB stick memory (System-RescueCD, 2015).

If we have a writable optical drive, and its writable media, then we can modify the last two requirements. We can create three optical disks instead of the sticks. Technically speaking, we do not absolutely need the first USB stick, either. If everything goes well every time, we do not need it. But we feel it is very handy when we are in trouble in UEFI booting.

Not all PCs are the same. It may look alike, but the inside components drastically change the characteristics of the systems. For example, even the same 64-bit x86 CPU, number of cores/threads may influence on the performance of an application.

We are trying to present some common approach to implement UEFI, GPT based, elilo bootable procedures to boot up 64-bit x86-based systems using either debian or CentOS as an OS at one time. UEFI is a weak ground work so that a common OS is installed regardless of different configurations of hardwares. At another time we would like to boot up other OSes.

We assumes the followings:

1. PC is UEFI bootable. Most modern PCs have dual mode of traditional BIOS/MBR mode, and UEFI/GPT mode of boot up the system. And frequently the default is not UEFI, but the traditional mode: MBR. If so, you must set the hardware to boot from UEFI; otherwise the procedure explained in this paper should be interpreted with extreme cautions. Any device whether it is optical, USB stick, or HDD, you must select UEFI boot in the BOOT section of your PC's ROM configuration menu.
2. SystemRescueCD (`http://www.sysresccd.org/`) is available as a boot device: CD or USB stick. Probably any other rescue type of linux can do the same procedure. If you choose other live/rescue type of linux, you have to examine the following procedures all by yourself. The critical functions of this rescue type linux is:

- GPT partition tool. We use `gdisk` to partition. You can use any other software which can GPT partition an HDD.
- `vfat` format a partition.
- `chroot` capable.
- `wget` or other download software from the internet.

3. There should be at least one empty non-volatile RAM (NVRAM) bootloader entry. Depending on our hardware, the number of NVRAM UEFI boot entry varies. Some PCs have only one. If we do not have any empty space, naturally we cannot create a new entry; thus we cannot register elilo as a boot loader. But we can always boot from default UEFI boot loader, and then through it we can invoke elilo bootloader.
4. UEFI GPT based partition is properly configured. There are certain restrictions we need to follow. We will explain it in the subsequent section.
5. UEFI GPT EFI System partition (ESP) partition is properly configured. We will discuss about it later in the subsequent section.

Before we embark the actual linux distribution install, we need prepare an USB stick memory, an HDD, and a bootloader: elilo.

USB stick Preparation

USB stick memory is not absolutely necessary, but we use it, and feel it is extremely comfortable; thus we strongly recommend you to use at least one usb stick. In some cases, we may fail in USB boot from HDD one reason or another, but we can always safely boot from UEFI usb stick. Then we can switch to HDD's ESP partition to boot from a bootable linux partition of our choice.

1. Have a small USB stick more than 128MB capacity at hand to be used as a super-floppy, i.e. non-partitioned single disk. Most USB stick memory is originally this condition. Label it "UEFI boot".
2. CD or another USB stick with bootable SystemRescueCD. And boot from it; in this case it does not matter whether MBR, or UEFI boot. Label it "SystemRescueCD".
3. Insert "UEFI boot" stick. If it is not VFAT super-floppy formatted, then
   ```
   # mkfs.msdos /dev/sdX
   ```
   Device name `/dev/sdX` is the one associated with the inserted "UEFI boot" stick, not SystemRescuCD one. Mount it, and create a folder `EFI`, then `BOOT` within `EFI` folder.
   ```
   # mount /dev/sdX /mnt/windows
   # mkdir /mnt/windows/EFI
   # mkdir /mnt/windows/EFI/BOOT
   ```
4. Copy a binary file of `EFI` shell to `BOOT` folder. We can find 64-bit binary of `EFI` shell in `EFI` Development Kit in either of the following site. Let us call the first one simply `EDK`, and the last one `EDK II`.
   ```
   {EDK}\Other\Maintained\Application\UefiShell\bin\x64\Shell_full.efi.
      http://sourceforge.net/projects/efidevkit/files/Releases/
               Official Releases/Edk 1.06.zip/
   EFI Shell, EFI Toolkit, EFI Dev kit, EDK II
      http://sourceforge.net/projects/edk2/files/
   ```
   Copy either file: `Shell_Full.efi` to usb stick as `\EFI\BOOT\BOOTX64.EFI`. We have not been successful in using `EDK II`; thsu we always use old one: `EDK`. At this time, we are not interested in figuring out, or debugging `EDK II`; thus we simply use the version which works.

Next, we will prepare an HDD for the first linux distribution install.

HDD preparation

An HDD must be configured prior to the final linux install. We will partition the hard drive using `SystemRescueCD`.

HDD's first partition must be VFAT filesystem formatted reserved (ESP), and the second one is allocated to the common swap: linux virtual memory; thus normal partition start the third one on. Note that Linux recognizes only up to and including fifteenth partition even though GPT allows up to 254 partitions.

After booting from `SystemRescueCD`, we invoke the HDD partition tool as follows:

```
# gdisk /dev/sda
```

`gdisk` is similar to `fdisk`, but it can handle GPT partition. We will attach sample HDD partition using `gdisk` out as Appendix 1.

There are a couple other items we observe.

- We keep bottom 2047 sectors, and top 2048 or more sectors unallocated. The bottom one is used as a primary UEFI GPT data area, and the top one used as a backup UEFI GPT data area.
- All linux partition must occupy even numbered sectors, but furthermore we set each of it as multiple of 2048 sectors. This works well for modern physical 4K advanced formatted HDD drive.
- From partition three to fifteen, I classify them into three categories: regular, testing, and data partitions. The last partition: the partition fifteen, is allocated as a common local linux data partitions available for all linux installed from partition three to fourteen. How many testing linux OSes install partitions are allocated depends on what you want to do. We set it four: from eleventh to fourteenth.

Before we start our first linux distribution install, we will set up bootloader: elilo.

elilo setup

We must obtain the boot loader: `elilo.efi`, and set up its configuration file: `elilo.conf`. We can use SystemRescueCD to do it all.

1. Boot from SystemRescueCD or other using another PC, create a folder.

```
$ ~/elilo
$ cd ~/elilo
```

2. Download `elilo-3.16-all.tar.gz` from `sourceforge`, or your local mirror site.

```
$ wget http://sourceforge.net/projects/elilo/files/elilo/elilo-3.16/
      elilo-3.16-all.tar.gz
```

3. Unpack `elilo`.

```
$ tar xvzf elilo-3.16-all.tar.gz
```

Create UEFI bootable device using a usb stick. Some ROMs fails to recognize HDD's GPT ESP partition as a valid UEFI bootable device. Thus it is safe to keep one which is always recognized as such.

1. Create a `dos` partition in the inserted USB stick.

```
# mkdosfs -F 32 -n usbxyzESP -S 512 /dev/sdY
```

2. Mount the USB partition (NSLU2-Linux, 2015).

```
# mount -o codepage=850,iocharset=iso8859-1,utf8 /dev/sdY /mnt/xxx
```

3. Create a directory in the dos partition.

```
# cd /mnt/xxx
# mkdir EFI
# mkdir EFI/BOOT
# mkdir EFI/ELILO
```

4. Copy 64-bit binary of EFI shell: Shell_full.efi as BOOTX64.EFI to EFI/BOOT/.

```
# cp Some/Where/Shell_full.efi EFI/BOOT/BOOTX64.EFI
```

5. Copy elilo-3.16-x86_64.efi as elilo.efi to EFI/ELILO/.

6. Create elilo.conf, and save it to usb stick folder EFI/ELILO/ with Appendix 2 as a sample file.

   *** Strictly speaking you cannot create a normal vfat32 text file using Unicode.

7. Copy elilo related files from target PC to usb stick /EFI/ELILO. Since we have not installed any OSes yet, we can simply skip this step now.

```
# cp /boot/*-amd64 /boot/efi/EFI/ELILO/
```

Since we use elilo as our boot loader, we must come back to configure elilo once again after we have installed one distribution, and another. We will explain a detailed installation steps for the most recent Debian jessie first, and next CentOS 7.0. If you follow these steps, you will obtain two separately UEFI bootable distribution working OSes.

Debian

The installation should be done to the third partition or after, neither the first partition (ESP), nor the second partition (swap). The swap partition does not necessarily reside at the second partition, but ESP partition must be the first partition of a UEFI bootable HDD.

Normal debian install is described elsewhere (Debian Installer, 2015); thus we show the text only rescue system based installation resulting extremely small debian install. Naturally, we can further install other packages such that we can create a full-blown system.

The method explained here is also applicable to ubuntu with some minor modification. It is discussed elsewhere (Pix Galore, 2015).

We assume the following. Some of them are already described in the previous sections, but we have listed bellow to emphasize.

1. PC is UEFI bootable.
2. SystemRescueCD is available as a boot device: CD or USB stick.
3. Find out what version of cdebootstrap-static is available. Currently 0.5.7, 0.5.9, and 0.6.4 are listed.

   http://ftp.debian.org/debian/pool/main/c/cdebootstrap/

   We will download, extract, install, and use it to obtain all necessary debian packages.
4. There should be at least one empty NVRAM bootloader entry; otherwise we cannot register our elilo boot loader.
5. EFI GPT based partition is properly configured.
6. EFI GPT ESP partition is properly configured.

Now we illustrate a rather length procedure with some explanations. Total number of major steps are about twenty. Please do not skip any of them.

1. Boot from UEFI `SystemRescueCD`. Make sure we are booting from EFI. If we cannot, then we cannot configure NVRAM until next boot.
   Booting may fail, if Intel Graphics (Atom graphics). It's probably that our default SystemRescueCD's kernel cannot detect where the brightness is. Luckily, this is easy enough to modify, as long as our current kernel version is 3.13.x or newer. Simply insert one of the following to Kernel configuration command line

   ```
   video.use_native_backlight=1     ; unknown parm, but it works
   video.use_native=1
   video.use=1
   ```

   to our kernel command line (Gentoo Linux, 2015).
2. Format the GPT partition we want to install the system, mount file systems to the partition to install, and create swap partition. Where `sdxy` is device name like `sdb3`, `sdwz` is device name like sda2. Before we proceed, it is always a good idea to examine the device, HDD, or SSD by proper procedure (`gdisk`), or other to avoid mishandling: accidentally formatting unintended device, or partitions.

   ```
   # mkfs.ext4 /dev/sdxy
   # mount /dev/sdxy /mnt/custom
   ```

   Create swap, if not made it as swap by other systems.

   ```
   # mkswap /dev/sdwz
   ```

3. Find out network connection is available.

   ```
   # route
   ```

   If not, reconnect/configure Ethernet connections.

   Retrieve, and extract `cdebootstrap-static` from debian site.

   ```
   # cd /tmp
   # wget http://ftp.debian.org/debian/pool/main/c/cdebootstrap/
       cdebootstrap-static_0.6.4_amd64.deb
   # ar xv cdebootstrap-static_0.6.4_amd64.deb
   # cd /
   # tar xvf /tmp/data.tar.xz
   ```

   Now we can employ debian image retrieval routine.

4. Retrieve minimal debian jessie system image from debian site, and place it in the specified partition.

   ```
   # cdebootstrap-static --allow-unauthenticated --arch=amd64
       --flavour=minimal jessie /mnt/custom
   ```

   If failed, try again ignoring all warnings until all files are retrieved, validated, installed, and configured properly. This is time consuming step Currently `wheezy, jessie, sid`, etc. are available. Flavor can be (Manpages, 2015):
   - `build`: Installs essential, `apt` and `build-essential`. Suitable for `sbuild` and `pbuilder` usage. All `rc.d` operations are disabled by a `policy-rc.d` script in `cdebootstrap-helper-rc.d` package.
   - `minimal`: Installs essential and `apt`. All `rc.d` operations are disabled by a `policy-rc.d` script in `cdebootstrap-helper-rc.d` package.
   - `standard`: Installs required and important priority packages.

We can also install other architecture, and/or other version of debian replacing word “amd64”, and/or “jessie”. But we will not discuss about it here.

5. Change the current and root directories to the provided directory, and then run command, if supplied, or an interactive copy of the user's login shell.

```
# LANG=C.UTF-8 chroot /mnt/custom /bin/bash
```

Confirm the prompt has changed.

6. Eliminate unnecessary.

```
# dpkg --purge cdebootstrap-helper-rc.d
```

7. Update, and install text editor: `nano`, and `apt` utility.

```
# apt-get update          # this access to the original site
# apt-get install nano apt-utils
```

8. Mount system directories, and create device files.

```
# mount -t proc proc /proc
# mount -t sysfs sys /sys
# mount -t devpts devpts /dev/pts
# apt-get install makedev
# cd /dev
# MAKEDEV generic          # takes time
```

9. Configure `apt` to retrieve additional debian packages over the internet. You can change the default to your favorite debian download site.

```
# cd /etc
```

Edit `/etc/apt/sources.list`. If non-standard packages are necessary, include `contrib`, and `non-free` as appropriate. If we use non-free device like a network interface chip in our hardware systems, then we must include it appropriately. Sample `/etc/apt/sources.list` is in Appendix 3. Then update, and upgrade.

```
# apt-get update
# apt-get dist-upgrade
# apt-get install (firmware-realtek) initramfs-tools
```

10. Mount partitions.

```
# cd /etc
```

Create, and edit `/etc/fstab`. An example is provided as Appendix 4. Next create a mount point `/boot/efi` for ESP partition.

```
# mkdir /boot/efi
```

Mount all the file systems.

```
# mount -a
```

11. Set time zone to your choice.

```
# dpkg-reconfigure tzdata
```

12. Configure networking.

```
# apt-get install ifupdown net-tools
# cd /etc/network
```

Edit, or create, and edit `interfaces.d/eth0` if `jessie` or later. A sample file is attached as Appendix 5.

Edit `/etc/resolv.conf`. A sample file is attached as Appendix 6.

Set `/etc/hostname`.

```
# echo HostName > ./hostname
```

Edit `/etc/hosts`. A sample file is attached as Appendix 7.

13. Configure `locales`, i.e. language, and character code.

```
# apt-get install locales
# dpkg-reconfigure locales
```

14. Install a linux `kernel`.

```
# apt-get install linux-image-amd64
# apt-get install efibootmgr
```

Edit `/etc/modules` to include `efivars`, if `wheezy` or earlier

15. Remove any cached package files from the folder `/var/cache/apt/archives`.

```
# apt-get clean
```

16. Set root pass word.

```
# passwd
```

*** The following two steps may depend on our final configuration of the system.

17. Create a general user, its home directory, and its pass word.

```
# useradd -m UserName
# passwd UserName
```

18. Configure EFI `elilo` boot assuming `efi` partition is already structured in the previous section.

```
# cd /boot
# cp initrd.img* efi/EFI/ELILO/
# cp vmlinuz* efi/EFI/ELILO/
```

Edit `efi/EFI/ELILO/elilo.conf`. A sample `elilo.conf` file is provided in Appendix 2.

19. Unmount everything in chroot environment, exit from it, unmount the installed partition, and reboot.

```
# umount -a
# umount /dev/pts
# umount /sys
# umount /proc
# exit
```

Notice the prompt has changed.

```
# umount /mnt/custom
# shutdown -r now
```

Now we have completed an initial debian install.

Then we will boot from HDD `BOOTX64.efi`.

```
Shell > fs0:
fs0:\ > cd EFI\ELILO
fs0:\EFI\ELILO > elilo
```

After you type in “elilo”, and hit the return, then hit TAB immediately. Then all the elilo bootable linux images are listed, choose the one you have just installed.

Note that UEFI environment is similar to DOS; thus “/” is replaced by “\”. If failed to boot, boot from USB stick; some old PCs do not work with `EDK2`, then use old, simple, and reliable `EDK`. If still failed to boot, boot from SystemRescueCd, and fix any problems.

There are some good techniques to keep in mind. They cannot be listed in a good order. These are certainly helpful when we are in a trouble.

1. When we want to erase NVRAM registered bootloader in ESP,
   (a) Boot
   (b) Erase directory at /boot/efi/EFI

   ```
   # cd boot/efi/EFI
   # rm -fr centos
   ```

   (c) Format the partition body

   ```
   # mkfs.ext4 /dev/sda5
   ```

   (d) Remove NVRAM entry

   ```
   # efibootmgr -b 2 -B
   ```

   (e) Update grub entry, if grub is used as a bootloader.

   ```
   # update-grub
   ```

   *** Warning ***: it is very important to synchronize NVRAM booloader content, and bootable HDDs and SSDs ESP entry.

2. When we want to use grub as a bootloader (superuser.com, 2015),
   (a) Find there is other default debian grub bootloader

   ```
   # efibootmgr -v
   ```

   (b) If no debian loader, go to 5; otherwise rename the debian grub bootloader

   ```
   # cd /boot/efi/EFI/debian
   # mv grubx64.efi gurbB2.efi
   ```

   (c) Remove original debian loader assuming 0th entry

   ```
   # efibootmgr -b 0 -B
   ```

   (d) Create an old one as a new entry

   ```
   # cd /boot/efi
   # efibootmgr --create --gpt --disk /dev/sdb --part 1
      --write-signature --label "JeGrubB2"
      --load "\\efi\\debian$grubB2.efi"
   ```

(e) Install grub (*** this step may be unnecessary)

```
# apt-get install --reinstall grub-efi-amd64
( # update-grub ))
```

(f) Confirm the proper creation

```
# efibootmgr -v
```

*** It's a good idea, rename the new one to something else so that another new install does not conflict.

Now we are complete at installing debian jessie. Next we will install CentOS 7 which is RHEL 7 compatible.

CentOS

We would like to install in complete text mode, but the current version 7.0 default installer allows us only very limited configurations requiring a complete HDD format with xfs filesystem. We would like not to abandon the freedom of which partition to install, what filesystem to use, what to select as a bootloader, and etc. Thus the choice we have at present is only to use graphical install, but we can obtain a complete text mode OS, if we choose minimal base environment in the software selection later.

CentOS installation is extremely important because a lot of commercial applications on a linux distribution, if available, may only assume RHEL. Since CentOS is compatible to RHEL, we can clear this restriction with ease. Kishore(2015) has shown the brief installation steps of CentOS 7 including some screen shots; thus the interested reader should examine his described steps also.

The installation is done to the third partition or after, neither the first, nor second partition. The reasoning is the same with previously described debian install.

We will illustrate the steps which use elilo as a bootloader; thus we intentionally avoid the default grub install. Installation of grub as CentOS 7 bootloader is quite easy; thus we will not elaborate any further. Non-UEFI install of CentOS 7 is illustrated with screen shots by Cezar (2014) or elsewhere.

1. Create an installation media. We obtain the image from the following official site:

   ```
   http://isoredirect.centos.org/centos/7/isos/x86_64/
           CentOS-7.0-1406-x86_64-DVD.iso
   ```

   or we can obtain it from the following official mirror site:

   ```
   http://www.centos.org/download/mirrors/
   ```

   Then we burn its image to CD, or transfer its images to USB stick to create installation media.

2. At BIOS boot menu, choose "EFI boot from CD", or comparable one. We must choose EFI boot; otherwise some additional adjustment is required after all the install is complete, but before reboot.

3. At the first disk installer prompt, choose "Install CentOS 7". Then hit 'e' to enter command line edit menu, then append 'gpt' after 'quiet'.

4. Anaconda installer menu (Graphical).

(a) Choose language to use during installation, and click “Continue” at the bottom right.

(b) Installation summary.

i. Click Network, and hostname.

A. Set Hostname at the left-bottom.
Enter fully qualified host name, i.e. `my.complete.hostname.with.subdomain`, or `hostname.sub.domain.com`.

B. Click top right ON/OFF to set to ON. If network is connected, and if you’re satisfied the displayed network parameters, click “Done” at the top right; otherwise click “Configure” at the right-bottom.

1. IPv6 Settings.
   a. Choose Method: Ignore.
2. IPv4 Settings.
   a. Choose Method: Manual.
   b. Click Add at Addresses.
      1. Enter IP Address, i.e. 192.168.0.15.
      2. Enter Netmask 24, if 255.255.255.0.
      3. Enter Gateway, i.e. 192.168.0.1.
   c. Enter DNS servers, i.e. 192.168.0.1 192.168.0.254.
   d. Enter Search domains, i.e. sub.domain.com.
   e. Click Require IPv4 addressing.
3. Click “Save” at bottom of submenu.

C. Click top right ON/OFF to set to ON.

D. Confirm network connection is established by the status.

E. Click top left “Done”.

ii. Click Date, and Time.
Choose an appropriate location. Click top left “Done”.

iii. Click Installation source.

A. Confirm “On the network” is marked.

B. Enter the following site or other appropriate one to retrieve the data over the internet.

```
http://mirror.centos.org/centos/7/os/x86_64/
```

C. Click “Done” at the top left.

iv. Click Installation destination.

A. Confirm proper local standard disk is set.

B. Click “I will configure partitioning”. at “Other Storage Options”, “Partitioning”.

C. Click the bottom left “Full disk summary and boot loader...”.

1. Choose disk.
2. If no bootloader be installed, then click “Do not install bootloader”, and confirm the bottom warning message.
3. Click “Close”.

D. Click “Done”.

E. At Manual partitioning menu.

1. Click all Unknown, and other to show all partitions.
2. Click the partition to install CentOS.

a. Confirm swap partition is properly set.
Check reformat, if necessary
If other linux system made it swap, we do not need to reformat.
b. Choose `/` partition.
1. Set partition's unique Label, i.e. CentOS7.
2. Check "Reformat".
3. Choose "FileSystem".
4. Set Mount point, i.e. `/`.
5. Click Update Settings.
c. Choose `/boot/efi` partition.
1. Set Mount Point, i.e. `/boot/efi`.
2. Click Update Settings.
If `ESP` is already configured, do not reformat.
e. Confirm three partitions: `/, swap, /boot/efi` settings are properly reflected at "New CentOS 7 Installation" table.
d. Click "Done" at the top left.
3. At "Summary of Changes" submenu, confirm, and if OK, click "Accept Changes".

v. Click "Software selection".
A. Choose appropriate base environment.
B. Choose appropriate Add-Ons for selected environment.
C. Click "Done" at the top left.
vi. Confirm Installation summary has no warning.
vii. Click "Begin Installation" at the bottom right.

5. Configuration

(a) Click Root password.
i. Enter root pass word.
ii. Enter confirmation pass word.
iii. Click "Done" at the top left; if you provide weak password, you have to click "Done" once more.

(b) Click User creation.
i. Enter Full name.
ii. Enter Username.
iii. Enter pass word.
iv. Enter confirmation pass word.
v. Click "Done" at the top left.

6. Confirm the installation process is started by the message of "Starting package installation process".

7. At the installation completion, click "reboot" near the bottom right.

8. Boot debian, or other linux including SystemRescueCD to configure elilo boot loader to include CentOS7.

9. At reboot, check “I accept the license agreement” if it appears, and click “Done” at the top left.
10. Click “Finish configuration” at the bottom right.

During the installation process, it may enter the screen saver mode. To recover from it, hit Esc. At reboot login, perform software update.

```
# yum update
```

Whenever we have a kernel update, we must perform manual bootloader `elilo.conf` update because we configure bootloader manually. If you forget to notice, just examine `/boot` to recognize a new kernel. If you know there is a new kernel, then

1. Copy new `vmlinuz`, `initramfs` files to `efi\EFI\ELILO\`.
2. Edit `efi\EFI\ELILO\elilo.conf` to reflect a new kernel.
3. Reboot to take a new kernel in effect.

If we would like to install grub as a bootloader not elilo, the following are rough procedures to follow after rebooting to CentOS.

1. Find there is other centos grub bootloader.

```
# efibootmgr -v
```

2. If either no centos loader, or we do not care that it will be over-written, go to 5; otherwise rename the `centos` grub bootloader folder. Note that `centos` is a a whole folder having several files.

```
# cd /boot/efi/EFI
# cp -R centos centos7      ; has to be some unique name
```

3. Remove original centos loader assuming 0th entry

```
# efibootmgr -b 1 -B
```

4. Create an old one as a new entry

```
# cd /boot/efi
# efibootmgr --create --gpt --disk /dev/sdb --part 1 --write-signature
              --label "CentOS7B7" --load "\\efi\\centos7\\shim.efi"
```

5. Install grub. (This step may be unnecessary.)

```
# yum reinstall grub-efi
```

6. Confirm the proper creation

```
# efibootmgr -v
```

Now we have completed CentOS install. Before we go to the concluding section, we would like to review, and reconfigure `elilo.conf` once again.

Configuration of elilo

We have completed two major linux distributions install. Now is the best time to re-examine `elilo.conf` so that it let us to choose the proper bootable partition to boot. Boot from USB stick, then GPT partitioned disk is in Target PC. If you have not prepared the first USB stick mentioned in "USB stick Preparation", you cannot perform the following tasks.

1. Boot UEFI:USB stick device.
   In Asus, at EFI EZ mode, choose Boot Menu(F8)
   In Intel, choose Boot Menu(F10)

2. After reboot from UEFI shell.

```
Shell> fs0:
fs0:\> cd EFI\ELILO
fs0:EFI\ELILO> elilo
```

3. If default image to boot, simply hit return key; otherwise to choose an image to boot, hit TAB key, and type any one label displayed.

4. After successful reboot to efi booted debian, just we would like to make sure we are in the right environment.

```
# apt-get install dosfstools efibootmgr
# efibootmgr -v
# modprobe efivars
```

   Edit `/etc/modules` to include `efivars`, if wheezy or earlier.

5. Edit `/etc/fstab` to mount `/boot/efi`.

```
/dev/sda1   /boot/efi       vfat    defaults        0       1

# blkid /dev/sda1
  /dev/sda1: UUID="64A1-0836" TYPE="vfat"
```

6. Create VFAT partition (ESP) as the first partition of the boot HDD.

```
# mkdosfs -F 32 -n ESPxyz -S 512 /dev/sdX1
```

7. Create a directory `/boot/efi`, and mount the first partition on it (nslu2-linux, 2015).

```
$ cd /boot
# mkdir efi
# mount -o codepage=850,iocharset=iso8859-1,utf8 /dev/sdX1 efi
```

8. Create folders in the dos partition.

```
$ cd efi
$ mkdir boot
$ mkdir efi
$ mkdir efi/elilo
```

9. Copy `elilo-3.16-x86_64.efi` as `elilo.efi` to `efi/elilo/`.

10. Create `elilo.conf`, and save it to usb stick `efi/elilo/` with Appendix 2 as a sample.

Whenever we installed a linux distribution to a new partition, or whenever our linux image is updated, we need to edit `elilo.conf`. If in doubt, periodically compare the current running kernel version, i.e., the output of

```
$ uname -r
```

and the file list of `/boot`. If we find any new linux image in `/boot`, copy `vmlinuz`, and `initrd.img/initramfs` files to `/boot/efi/EFI/ELILO/`, then edit `elilo.conf` to reflect the update. Also whenever newly dated `initrd.img/initramfs` is found, simply copy it as above, but you do not need to edit `elilo.conf`.

Conclusion

A detailed procedure to boot x86 64-bit CPU based PC systems under UEFI ROM, GPT partitioned HDD, and the boot loader: elilo installing two famous representative linux distributions: debian, and CentOS is described. We have paid special attention to debian text install without an official debian installer. Since these two distributions are representative linux distributions, installing other distributions will be well-guessed.

We would like to extend an UEFI application elilo capability so that it can read linux filesystem: ext4, ext3, and etc. The current elilo version cannot read linux filesystem directly; thus it is necessary to copy both initrd.img/initramfs, and vmlinux image from /boot to VFAT formatted /boot/efi/EFI/ELILO, or ESP partition's EFI/ELILO folder to enable elilo reading these files. Certainly, enabling read-only linux filesystem does increase the size of elilo executable, but the resulting fruit is much larger. As we try to include more number of partitions to boot, not only the duplication of both initial ram disk file, and vmlinuz file, but also we must always update either or both of these files in the ESP partitions whenever either or both of them is updated; thus we need to pay attention to the upgrade process. Currently we need to update both files in two separate partitions: a partition which /boot belong, and ESP.

We do not like GRUB as a boot loader because it tries to wipe out the previous default boot-loader setting; thus it behaves it is the only bootloader in the system. GRUB is a chain boot loader, that is, its boot entries are all the previously bootable entries.

We believe that an OS and its boot loader is a separate entity which belongs to the separate levels of system software hierarchy in PC systems. Any bootloader in UEFI systems is an EFI application, not an OS application. Thus upgrading an OS should not influence upgrading bootloader directly.

We like non-monolithic, and non-chain-loading feature of elilo; thus we would like to assist elilo development.

A UEFI function to initialize NVRAM of PC must be developed. Failure/Success of NVRAM installation of boot loader makes the table maintained by ESP, and NVRAM easily inconsistent. Sometimes we can synchronize from an OS, but it may fail occasionally. Once it gets inconsistent, and we may fail to synchronize; then there is no way to synchronize except for hardware total reset, which may involve extraordinary CMOS clear. Thus we cannot recommend playing around with NVRAM boot entries.

We would like to label a bootloader by ourselves. Many installers do not allow us to label its bootloader now. Thus we need to follow the following procedure to label it.

1. Copy boot loader, and save it under another name.
2. Delete the original loader using efibootmgr.
3. Label the copied loader using efibootmgr.

Then we can make NVRAM entry consistent with ESP entry. This is tedious, and it may creates a certain mistake. An installer can show its default boot loader name, and let us override its default name by entering new name.

Also we need an EFI application to show maximum allowable number of UEFI bootloader entry. Most UEFI ROMs we encounter are based on very old UEFI specifications. Most of them are upgraded recently, but they still carry UEFI specification date of 2010 time stamp. Some UEFI PCs have a very limited UEFI bootloader entry size like just one. Through knowing this number, we can make a good planning of UEFI bootloader entry management for each PC.

We are developing CentOS 7 complete text install method without using any video graphics. This method is important for a Linux server install because some servers may have text only hardware; thus CentOS 7 install is not possible. And, of course, for most of linux server applications, graphics capabilities are not necessary at all except for just increasing resource requirement of its hardware. Preliminary installation is good, but we can not elaborate any further at this point in time.

Reference

Centos.org, “CentOS 7 Is Here.” CentOS Project. Web. 31 Jan. 2015. <http://www.centos.org/>.

Cezar, Matei. “Installation of ”CentOS 7.0 with Screenshots.” Tecmint Linux Howtos Tutorials Guides. Tecmint.com, 9 July 2014. Web. 25 Jan. 2015. <http://www.tecmint.com/centos-7-installation/>.

Debian.org, “Debian.” The Universal Operating System. Web. 31 Jan. 2015. <https://www.debian.org/>.

Debian Installer team, “Debian GNU/Linux Installation Guide.” Debian GNU/Linux Installation Guide. Web. 12 Jan. 2015. <https://www.debian.org/releases/stable/amd64/index.html.en>.

Gentoo Linux, “Intel.” Gentoo Wiki. Web. 12 Jan. 2015. <http://wiki.gentoo.org/wiki/Intel>.

Hogan, Thom. The Programmer’s PC Sourcebook. 2nd ed. Redmond, WA: Microsoft Press, 1991. Print.

International Business Machines Corporation, PC/AT Technical Reference. Boca Raton, Florida, 1984. Print.

Kishore, Srijan. “How to Install a CentOS 7 Minimal Server.” How to Install a CentOS 7 Minimal Server. 23 Oct. 2014. Web. 14 Jan. 2015. <https://www.howtoforge.com/centos-7-server>.

Manpages, “Manpages.” Manpage for Cdebootstrap-static. Web. 12 Jan. 2015. <http://man.cx/cdebootstrap-static(1)>.

NSLU2-Linux, HowTo MountFATFileSystems Browse. Web. 20 Jan. 2015. <http://www.nslu2-linux.org/wiki/HowTo/MountFATFileSystems>.

Pix Galore, “Pix Galore.” : Use Cdebootstrap to Install Debian/Ubuntu Linux. Web. 12 Jan. 2015. <http://pix-galore.blogspot.jp/2010/01/use-cdebootstrap-to-install.html>.

superuser.com, “How to Reinstall GRUB2 EFI?” Web. 12 Jan. 2015. <http://superuser.com/questions/376470/how-to-reinstall-grub2-efi>.

SystemRescueCd, “SystemRescueCd Homepage.” SystemRescueCd. Web. 31 Jan. 2015. <http://www.sysresccd.org/SystemRescueCd_Homepage>.

SystemRescueCD, “Sysresccd-manual-en How to Install SystemRescueCd on an USB-stick.” Web. 21 Jan. 2015. <http://www.sysresccd.org/Sysresccd-manual-en_How_to_install_SystemRescueCd_on_an_USB-stick>.

Appendix 1: HDD partition

```
Disk /dev/sda: 7814037168 sectors, 3.6 TiB
Logical sector size: 512 bytes
Disk identifier (GUID): A86E0228-26AE-43F1-80C5-672C729F5E96
Partition table holds up to 128 entries
First usable sector is 34, last usable sector is 7814037134
Partitions will be aligned on 2048-sector boundaries
Total free space is 5741 sectors (2.8 MiB)

Number  Start (sector)    End (sector)  Size       Code  Name
   1            2048         1050623   512.0 MiB   EF00  EFI System
   2         1050624      1074792447   512.0 GiB   8200  Linux swap
   3      1074792448      1209010175   64.0 GiB    0700  Wh@p8z77
   4      1209010176      1343227903   64.0 GiB    8300  Linux filesystem
   5      1343227904      1477445631   64.0 GiB    8300  Linux filesystem
   6      1477445632      1611663359   64.0 GiB    8300  Linux filesystem
   7      1611663360      1745881087   64.0 GiB    8300  Linux filesystem
   8      1745881088      1880098815   64.0 GiB    8300  Linux filesystem
   9      1880098816      2014316543   64.0 GiB    8300  Linux filesystem
  10      2014316544      2148534271   64.0 GiB    8300  Linux filesystem
  11      2148534272      2165311487   8.0 GiB     8300  Linux filesystem
  12      2165311488      2182088703   8.0 GiB     8300  Linux filesystem
  13      2182088704      2198865919   8.0 GiB     8300  Linux filesystem
  14      2198865920      2215643135   8.0 GiB     8300  Linux filesystem
  15      2215643136      7814033407   2.6 TiB     8300  Linux filesystem
```

Appendix 2: EFI/ELILO/elilo.conf

```
prompt
timeout=50
delay=30
default=Wh64
verbose=5
chooser=simple
append="vga=0x31F splash showopts"

image=vmlinuz-2.6.32-5-amd64
label=Sq64
initrd=initrd.img-2.6.32-5-amd64
read-only
root=/dev/sda4
append=""

image=vmlinuz-3.2.0-3-amd64
label=Wh64
initrd=initrd.img-3.2.0-3-amd64
read-only
root=/dev/sda3
append=""

image=vmlinuz-3.2.0-3-amd64
label=TWh64
initrd=initrd.img-3.2.0-3-amd64
read-only
root=/dev/sda11
append=""
```

Appendix 3: /etc/apt/sources.list

```
deb http://ftp.debian.org/debian jessie main contrib non-free

deb http://security.debian.org/ jessie/updates main contrib non-free
```

Appendix 4: /etc/fstab

```
/dev/sda3 / ext3 errors=remount-ro 0 1
/dev/sda2 none swap sw 0 0
/dev/sda1 /boot/efi vfat defaults 0 1
```

Appendix 5: /etc/network/interfaces.d/eth0

```
# The primary

allow-hotplug eth0
iface eth0 inet static
address 192.168.0.15
netmask 255.255.255.0
network 192.168.30.0
broadcast 192.168.0.255
gateway 192.168.0.1 192.168.0.254
```

Appendix 6: /etc/resolv.conf

```
domain sub.domain.com
search sub.domain.com

nameserver 192.168.0.1
nameserver 192.168.0.254
```

Appendix 7: /etc/hosts

```
# IPv4
127.0.0.1 localhost.localdomain localhost
# If dhcp, omit the following line
192.168.0.15 hostname.sub.domain.com hostname


# IPv6
::1 localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```