

Embracing Open Firmware in HPC for Faster and More Secure Provisioning

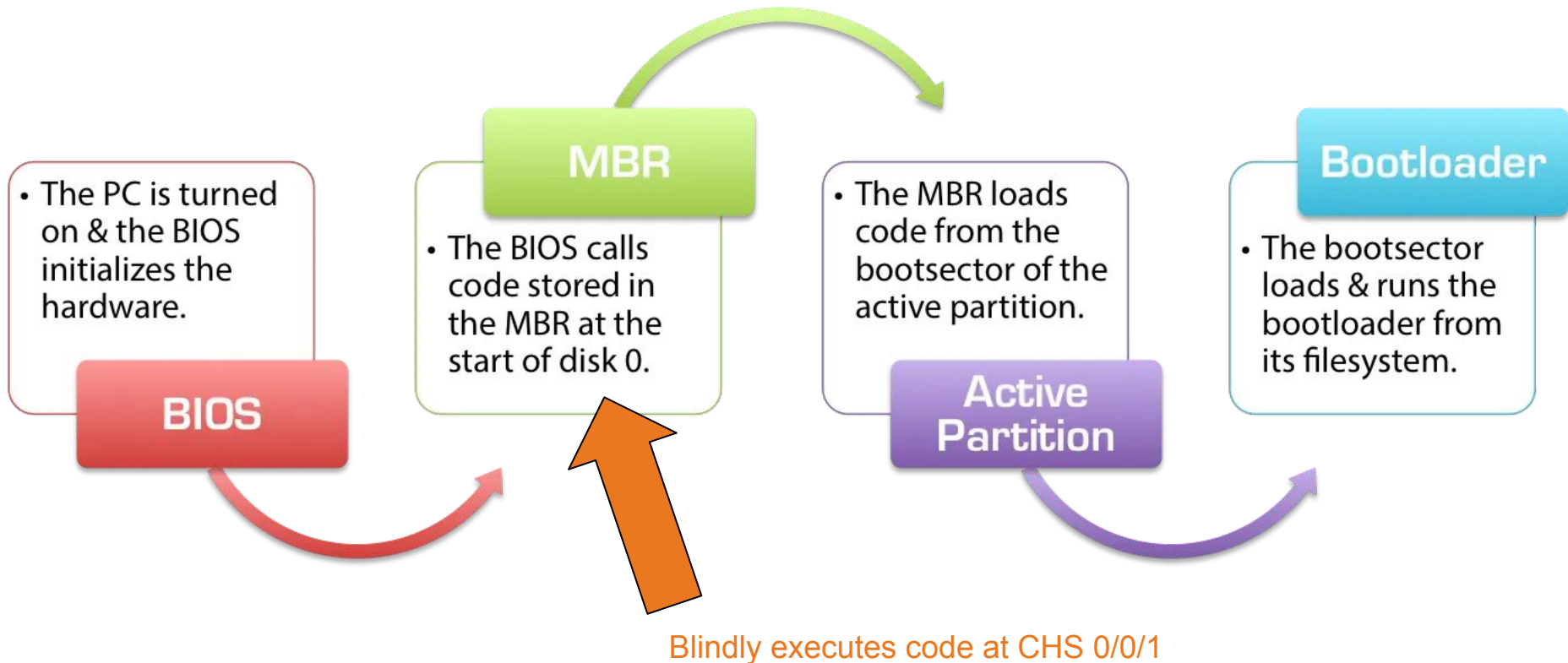


Devon T. Bautista

USRC Showcase
12 August 2020



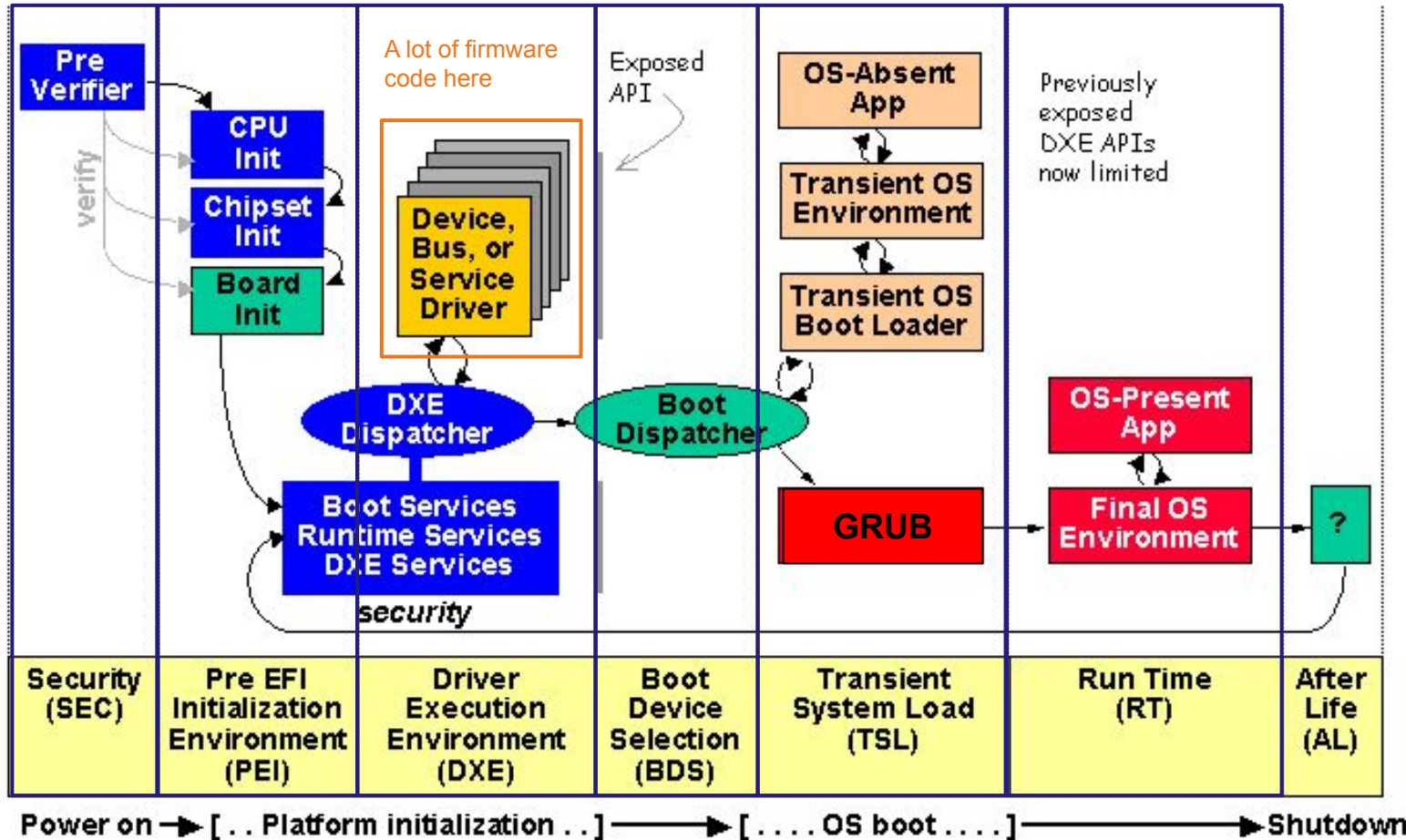
BIOS: The Old Way of Booting

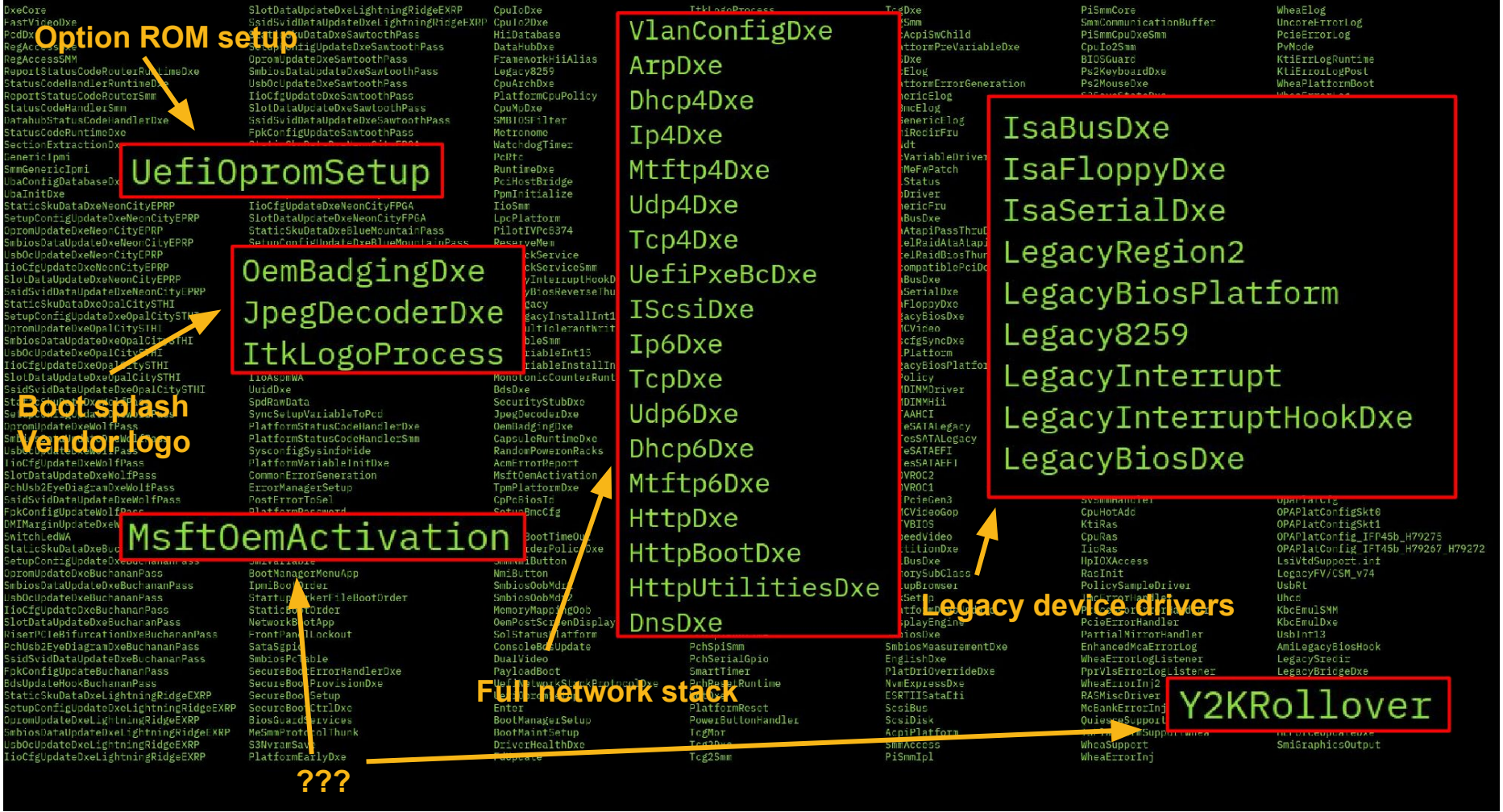


From: <https://neosmart.net/wiki/mbr-boot-process/>



UEFI: The Current Way of Booting





From: https://trmm.net/LinuxBoot_34c3

Network Drivers

tianocore / edk2

<> Code Pull requests 3 Actions Projects

master edk2 / NetworkPkg / TcpDxe /

Coeur and mergify NetworkPkg/TcpDxe/Tcp: Fix

- ..
- ComponentName.c
- SocketImpl.c
- SocketImpl.h
- SocketInterface.c
- Socket.h
- TcpDispatcher.c
- TcpDriver.c
- TcpDriver.h
- TcpDxe.inf
- TcpDxe.uni
- TcpDxeExtra.uni
- TcpFunc.h
- TcpInput.c

Intel's EDKII Firmware

git index : grub.git

GNU GRUB

summary refs log tree commit diff

path: root/grub-core/net/tcp.c

blob: e8ad34b84d4e9a049a7d190ac2d6bfeb6d547ab9 (plain)

```
1 /*
2  * GRUB -- GRand Unified Bootloader
3  * Copyright (C) 2010,2011 Free Software Found
4  *
5  * GRUB is free software: you can redistribute
6  * it under the terms of the GNU General Public
7  * the Free Software Foundation, either version
8  * (at your option) any later version.
9  *
10 * GRUB is distributed in the hope that it will
11 * but WITHOUT ANY WARRANTY; without even the i
12 * MERCHANTABILITY or FITNESS FOR A PARTICULAR
13 * GNU General Public License for more details.
14 *
15 * You should have received a copy of the GNU G
16 * along with GRUB. If not, see <http://www.gn
17 */
18
19 #include <grub/net.h>
20 #include <grub/net/ip.h>
21 #include <grub/net/tcp.h>
22 #include <grub/net/netbuff.h>
23 #include <grub/time.h>
24 #include <grub/priority_queue.h>
25
26 #define TCP_SYN_RETRANSMISSION_TIMEOUT GRUB_NET_
27 #define TCP_SYN_RETRANSMISSION_COUNT GRUB_NET_TR
28 #define TCP_RETRANSMISSION_TIMEOUT GRUB_NET_INTE
29 #define TCP_RETRANSMISSION_COUNT GRUB_NET_TRIES
30
31 struct unacked
32 {
33   struct unacked *next;
34   struct unacked **prev;
35   struct grub_net_buff *nb;
36   grub_uint64_t last_try;
37   int try_count;
38 };
39
40 enum
41 {
42   TCP_SYN = 0,
43   TCP_FIN = 1,
44   TCP_RST = 2,
45   TCP_PUSH = 3,
46   TCP_ACK = 4,
47   TCP_DATA = 5,
48   TCP_WINDOW_UPDATE = 6,
49   TCP_KEEP_ALIVE = 7,
50   TCP_INVALID = 8,
51   TCP_MAX = 9
52 };
```

GRUB Bootloader

/ net / ipv4 / tcp.c

```
1 // SPDX-License-Identifier: GPL-2.0-or-later
2 /*
3  * INET      An implementation of the TCP/IP
4  *          operating system. INET is imple
5  *          interface as the means of commu
6  *
7  *          Implementation of the Transmissi
8  *
9  * Authors:  Ross Biro
10 *           Fred N. van Kempen, <waltje@uWae
11 *           Mark Evans, <evansmp@uhura.astc
12 *           Corey Minyard <wf-rchlminyard@e
13 *           Florian La Roche, <fla@stud.ur
14 *           Charles Hedrick, <hedrick@klins
15 *           Linus Torvalds, <torvalds@cs.hs
16 *           Alan Cox, <gw4pts@gw4pts.ampr.c
17 *           Matthew Dillon, <dillon@apollo.
18 *           Arnt Gulbrandsen, <agulbra@nvg.
19 *           Jorge Cwik, <jorge@laser.satliir
20 *
21 * Fixes:
22 *           Alan Cox      :   Numerot
23 *           Alan Cox      :   Set the
24 *           Alan Cox      :   Stoppec
25 *           sk->int
26 *           (tcp_err
27 *           Alan Cox      :   All icr
28 *           pointer
29 *           socket
30 *           tested
31 *           Alan Cox      :   tcp_err
32 *           wakes f
33 *           behaves f
34 *           has gor
35 *           Alan Cox      :   tcp_ser
36 *           everytl
37 *           unknowr
38 *           tcp opt
39 *           Alan Cox      :   Reset t
40 *           syn rui
41 *           Herp Rosmanith :   More re
42 *           Alan Cox      :   No lonc
```

Linux

USB Drivers

```
tianocore / edk2

<> Code  1 Pull requests 3  Actions  Projects  Security  Insights

master  edk2 / MdeModulePkg / Bus / Usb / UsbBusDxe / UsbBus.c

Coeur MdeModulePkg/UsbBus: Fix various typos ...

A7 contributors

1537 lines (1274 sloc)  43.5 KB

1  /** @file
2
3      Usb Bus Driver Binding and Bus IO Protocol.
4
5  Copyright (c) 2004 - 2018, Intel Corporation. All rights reserved.<BR>
6  SPDX-License-Identifier: BSD-2-Clause-Patent
7
8  **/
9
10 #include "UsbBus.h"
11
12 EFI_USB_IO_PROTOCOL mUsbIoProtocol = {
13   UsbIoControlTransfer,
14   UsbIoBulkTransfer,
15   UsbIoAsyncInterruptTransfer,
16   UsbIoSyncInterruptTransfer,
17   UsbIoIsynchronousTransfer,
18   UsbIoAsyncIsynchronousTransfer,
19   UsbIoGetDeviceDescriptor,
20   UsbIoGetActiveConfigDescriptor,
21   UsbIoGetInterfaceDescriptor,
22   UsbIoGetEndpointDescriptor,
23   UsbIoGetStringDescriptor,
```

Intel's EDKII Firmware

```
git index : grub.git
GNU GRUB

summary refs log tree commit diff
path: root/grub-core/bus/usb/usb.c

blob: 8da5e4c7491b55df25c01e420700f7d78b583051 (plain)

1  /* usb.c - Generic USB interfaces. */
2  /*
3   * GRUB -- Grand Unified Bootloader
4   * Copyright (C) 2008 Free Software Foundation, Inc.
5   *
6   * GRUB is free software: you can redistribute it and/or modify
7   * it under the terms of the GNU General Public License as published by
8   * the Free Software Foundation, either version 3 of the License, or
9   * (at your option) any later version.
10  *
11  * GRUB is distributed in the hope that it will be useful,
12  * but WITHOUT ANY WARRANTY; without even the implied warranty of
13  * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
14  * GNU General Public License for more details.
15  *
16  * You should have received a copy of the GNU General Public License
17  * along with GRUB. If not, see <http://www.gnu.org/licenses/>.
18  */
19
20 #include <grub/dl.h>
21 #include <grub/mm.h>
22 #include <grub/usb.h>
23 #include <grub/misc.h>
24 #include <grub/list.h>
25 #include <grub/term.h>
26
27 GRUB_MOD_LICENSE ("GPLv3+");
28
29 static struct grub_usb_attach_desc *attach_hooks;
30
31 #if 0
32 /* Context for grub_usb_controller_iterate. */
33 struct grub_usb_controller_iterate_ctx
34 {
35   grub_usb_controller_iterate_hook_t hook;
36   void *hook_data;
37   grub_usb_controller_dev_t p;
38 };
39
40 /* Helper for grub_usb_controller_iterate. */
41 static int
42 grub_usb_controller_iterate_iter (grub_usb_controller_t dev, void *data)
43 {
44   struct grub_usb_controller_iterate_ctx *ctx = data;
45 }
46
```

GRUB Bootloader

```
/ drivers / usb / core / usb.c

1 // SPDX-License-Identifier: GPL-2.0
2 /*
3  * drivers/usb/core/usb.c
4  *
5  * (C) Copyright Linus Torvalds 1999
6  * (C) Copyright Johannes Erdfelt 1999-2001
7  * (C) Copyright Andreas Gal 1999
8  * (C) Copyright Gregory P. Smith 1999
9  * (C) Copyright Deti Fliegl 1999 (new USB architecture)
10 * (C) Copyright Randy Dunlap 2000
11 * (C) Copyright David Brownell 2000-2004
12 * (C) Copyright Yggdrasil Computing, Inc. 2000
13 * (usb_device_id matching changes by Adam J. Richter)
14 * (C) Copyright Greg Kroah-Hartman 2002-2003
15 *
16 * Released under the GPLv2 only.
17 *
18 * NOTE! This is not actually a driver at all, rather this is
19 * just a collection of helper routines that implement the
20 * generic USB things that the real drivers can use..
21 *
22 * Think of this as a "USB library" rather than anything else.
23 * It should be considered a slave, with no callbacks. Callbacks
24 * are evil.
25 */
26
27 #include <linux/module.h>
28 #include <linux/moduleparam.h>
29 #include <linux/string.h>
30 #include <linux/bitops.h>
31 #include <linux/slab.h>
32 #include <linux/interrupt.h> /* for in_interrupt() */
33 #include <linux/kmod.h>
34 #include <linux/init.h>
35 #include <linux/spinlock.h>
36 #include <linux/errno.h>
37 #include <linux/usb.h>
38 #include <linux/usb/hcd.h>
39 #include <linux/mutex.h>
40 #include <linux/workqueue.h>
41 #include <linux/debugfs.h>
42 #include <linux/usb/of.h>
43
44 #include <asm/io.h>
```

Linux

Filesystem Drivers

tianocore / edk2

<> Code Pull requests 3 Actions Projects

master edk2 / FatPkg / EnhancedFatDxe /

Coeur and mergify FatPkg/EnhancedFatDxe: Fix v

- ..
- ComponentName.c
- Data.c
- Delete.c
- DirectoryCache.c
- DirectoryManage.c
- DiskCache.c
- Fat.c
- Fat.h
- Fat.inf
- Fat.uni
- FatExtra.uni
- FatFileSystem.h
- FileName.c
- FileSpace.c

Intel's EDKII Firmware

git index : grub.git

GNU GRUB

summary refs log tree commit diff

path: root/grub-core/fs/fat.c

blob: 71775a17038b83d52482389c39a4a2f3f07db263 (plain)

```
1 /* fat.c - FAT filesystem */
2 /*
3  * GRUB -- Grand Unified Bootloader
4  * Copyright (C) 2000,2001,2002,2003,2004,2005,2
5  *
6  * GRUB is free software: you can redistribute it
7  * it under the terms of the GNU General Public
8  * the Free Software Foundation, either version
9  * (at your option) any later version.
10 *
11 * GRUB is distributed in the hope that it will
12 * but WITHOUT ANY WARRANTY; without even the im
13 * MERCHANTABILITY or FITNESS FOR A PARTICULAR F
14 * GNU General Public License for more details.
15 *
16 * You should have received a copy of the GNU Ge
17 * along with GRUB. If not, see <http://www.gnu
18 */
19 #include <grub/fs.h>
20 #include <grub/disk.h>
21 #include <grub/file.h>
22 #include <grub/types.h>
23 #include <grub/misc.h>
24 #include <grub/mm.h>
25 #include <grub/err.h>
26 #include <grub/dl.h>
27 #include <grub/charset.h>
28 #include <grub/datettime.h>
29 #include <grub/strftime.h>
30 #ifndef MODE_EXFAT
31 #include <grub/fat.h>
32 #else
33 #include <grub/exfat.h>
34 #endif
35 #include <grub/fshelp.h>
36 #include <grub/ll8n.h>
37
38 GRUB_MOD_LICENSE ("GPLv3+");
39
40 enum
41 {
42   GRUB_FAT_ATTR_READ_ONLY = 0x01,
43   GRUB_FAT_ATTR_HIDDEN = 0x02,
```

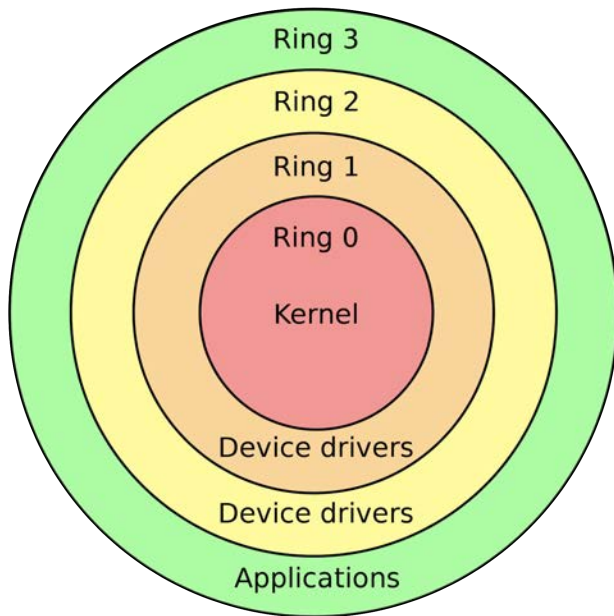
GRUB Bootloader

```
/ fs / fat / fat.h
1 /* SPDX-License-Identifier: GPL-2.0 */
2 #ifndef _FAT_H
3 #define _FAT_H
4
5 #include <linux/buffer_head.h>
6 #include <linux/nls.h>
7 #include <linux/hash.h>
8 #include <linux/ratelimit.h>
9 #include <linux/msdos_fs.h>
10
11 /*
12  * vfat shortname flags
13  */
14 #define VFAT_SFN_DISPLAY_LOWER 0x0001 /* conver
15 #define VFAT_SFN_DISPLAY_WIN95 0x0002 /* emulat
16 #define VFAT_SFN_DISPLAY_WINNT 0x0004 /* emulat
17 #define VFAT_SFN_CREATE_WIN95 0x0100 /* emulat
18 #define VFAT_SFN_CREATE_WINNT 0x0200 /* emulat
19
20 #define FAT_ERRORS_CONT 1 /* ignore
21 #define FAT_ERRORS_PANIC 2 /* panic
22 #define FAT_ERRORS_RO 3 /* remoun
23
24 #define FAT_NFS_STALE_RW 1 /* NFS RW
25 #define FAT_NFS_NOSTALE_RO 2 /* NFS RO
26
27 struct fat_mount_options {
28   kuid_t fs_uid;
29   kgid_t fs_gid;
30   unsigned short fs_fmask;
31   unsigned short fs_dmask;
32   unsigned short codepage; /* Codepage f
33   int time_offset; /* Offset of
34   char *iocharset; /* Charset us
35   unsigned short shortname; /* flags for
36   unsigned char name_check; /* r = relaxe
37   unsigned char errors; /* On error:
38   unsigned char nfs; /* NFS support
39   unsigned short allow_utime; /* permission
40   unsigned quiet:1, /* set = fake
41   showexec:1, /* set = only
42   sys_immutable:1, /* set = syst
43   dotsOK:1, /* set = hidd
```

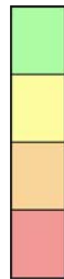
Linux

Privilege Rings

Traditional



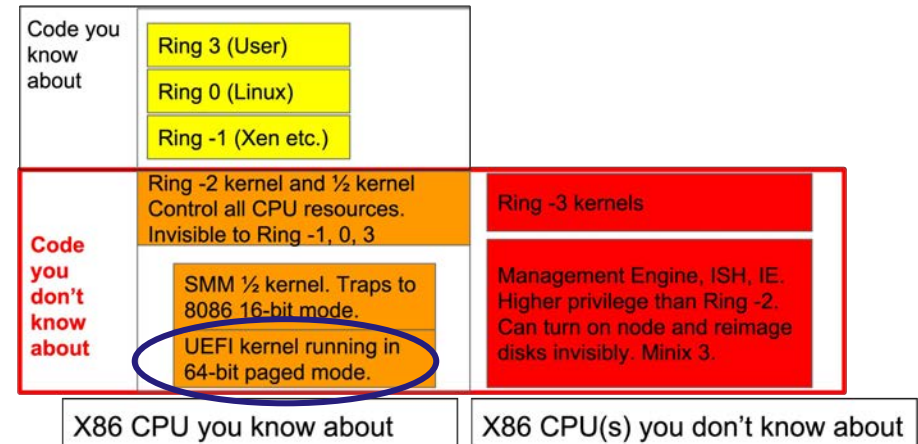
Least privileged



Most privileged

vs.

Modern



From: https://en.wikipedia.org/wiki/Protection_ring

From: <https://www.youtube.com/watch?v=iffTJ1vPCSo>



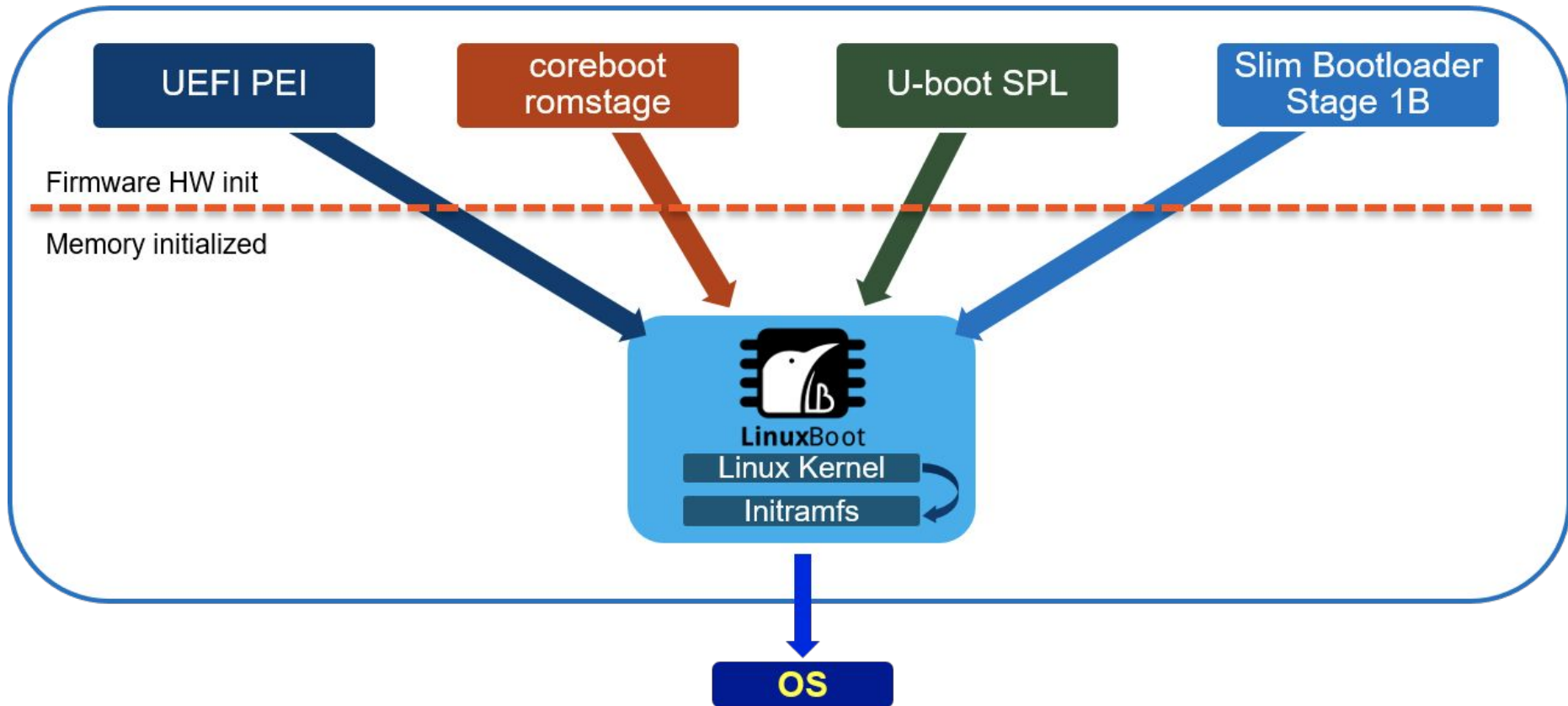
Problems

- Redundant drivers with different implementations
 - Increases attack surface
 - Too many unneeded or redundant drivers loading slows down boot
- Insufficiently audited code with the most privileged system access
 - Proprietary, closed-source
 - Reviewed by relatively small number of developers within company
- Reliant on vendor for updates and repairs



“Let Linux Do It”

SPI Flash



From: <https://www.linuxboot.org/>

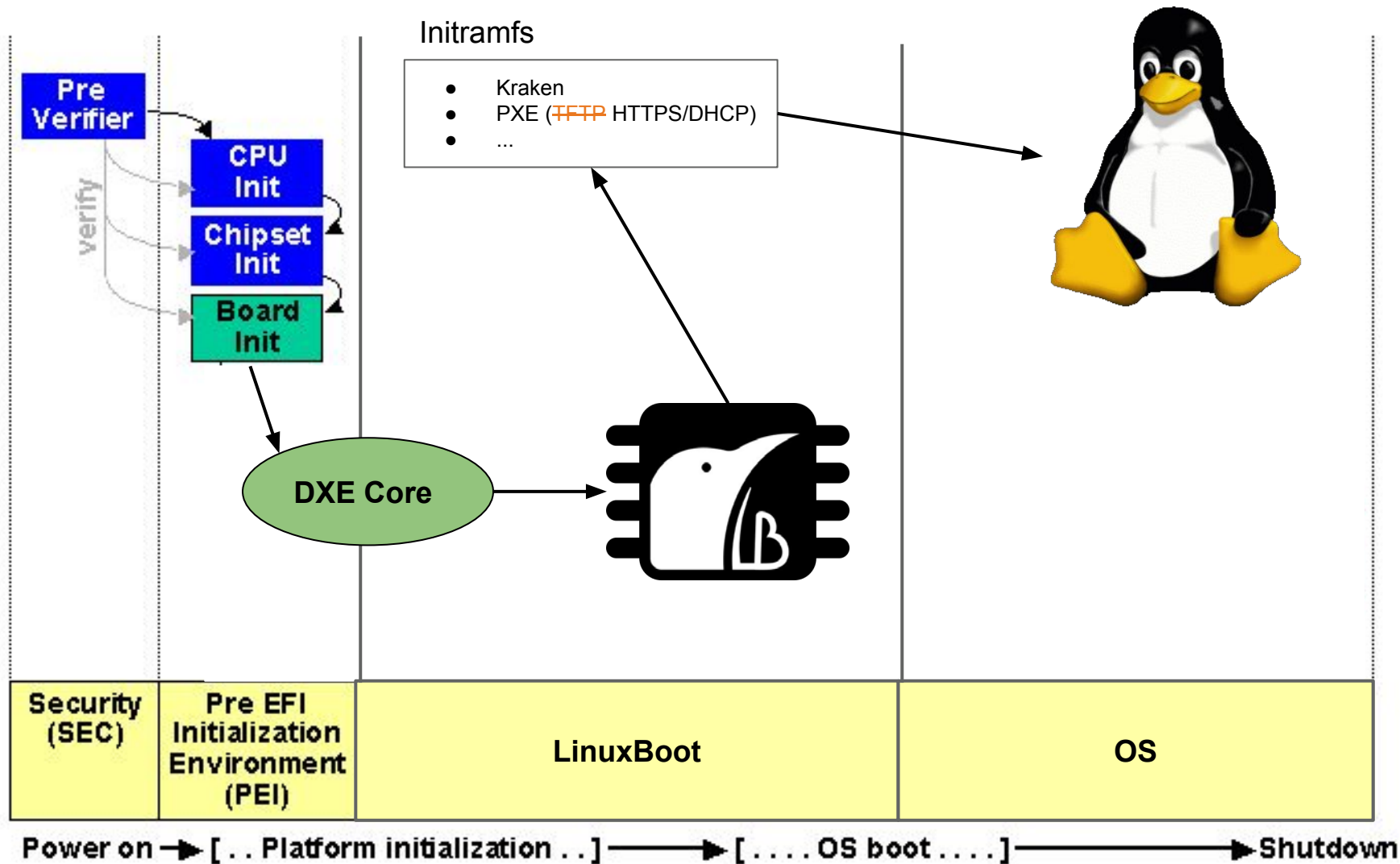


Benefits of Linux in Firmware



- Improves boot reliability
 - Replaces lightly-tested firmware drivers with hardened Linux drivers
- Improves boot time (up to 20 times faster in some cases)
 - Removes unnecessary/insecure code
- Allows customization of the initrd runtime to support site-specific needs (both device drivers as well as custom executables)
 - **Use Case:** Custom provisioning tools in the boot process
 - e.g. Replace TFTP with HTTPS for PXE booting
- Proven approach for almost 20 years in military, consumer electronics, and supercomputing systems – wherever reliability and performance are paramount

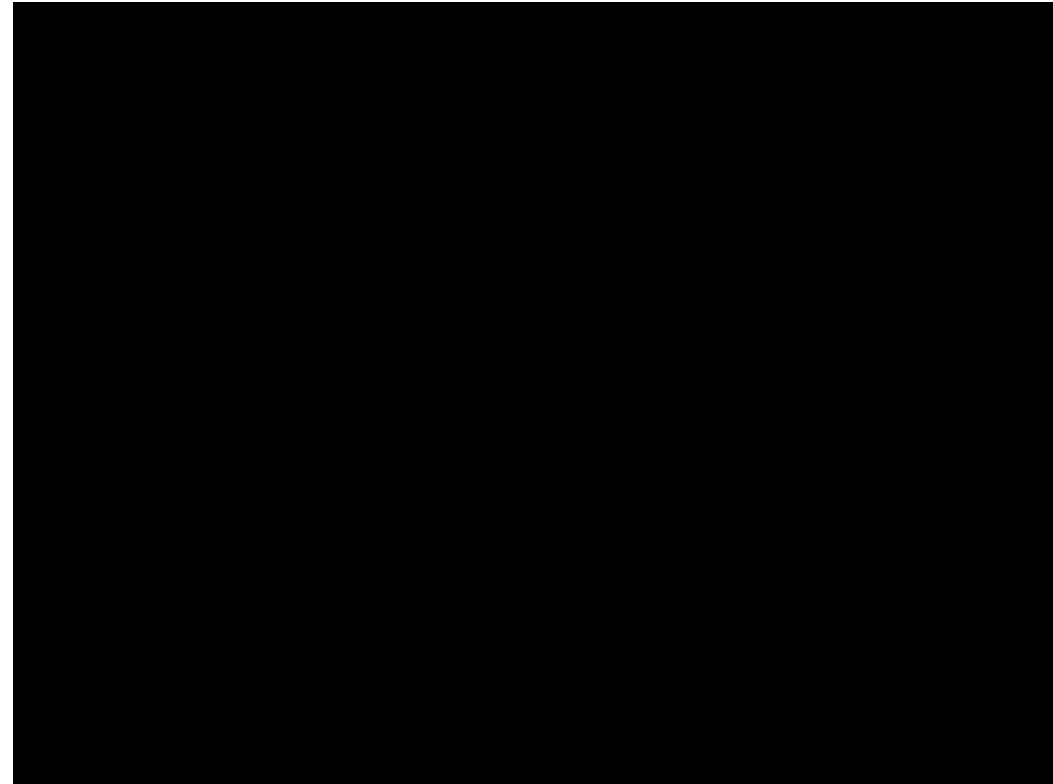
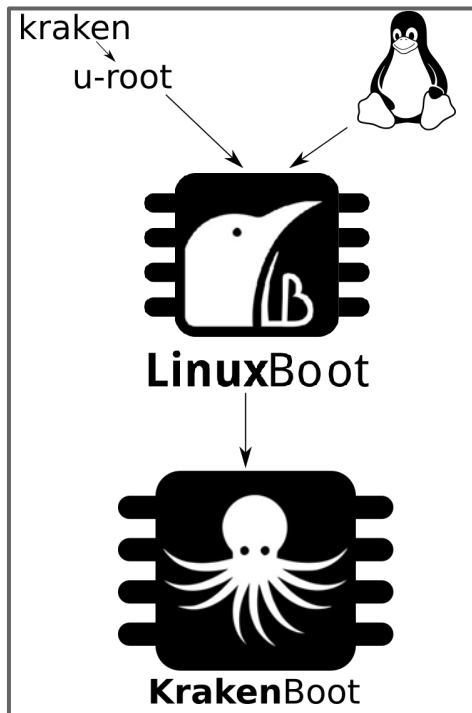




USRC's Research Into Provisioning with Open Firmware

Done

- Emulate a modified firmware image running a Linux kernel and custom initramfs
- Provision a VirtualBox cluster using kraken in a custom initramfs
 - Not a firmware image, but through VirtualBox



LA-UR-20-26019

Doing

- Create a working example of provisioning using emulatable firmware images
- Provision on *real* hardware in firmware



Problems Solved

Firmware Until Now	Firmware Now and Beyond
Contains an OS	Let Linux do it
Opaque, understood by few	Open, well-understood by many
Proprietary ecosystem	Auditable, debuggable
Product-specific	Portable, reusable
Vendor-specific tooling	Open source tools
Locked down	Customizable



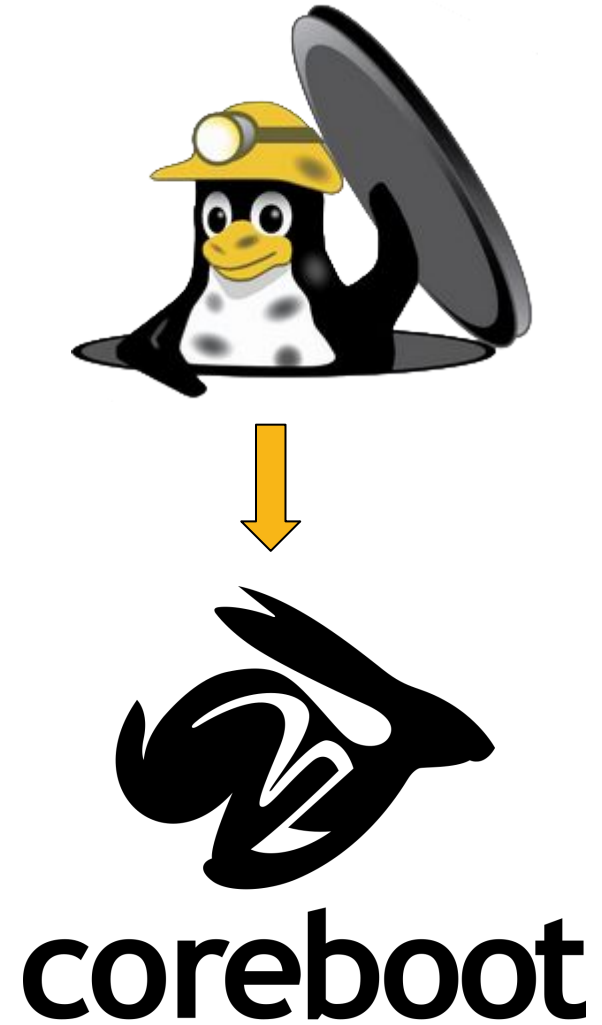
“The vendors will never support this.”



Open Firmware: Not a New Idea



Ron Minnich, creator of Coreboot (formerly LinuxBIOS), at LANL in 1999



Facebook



LinuxBoot in provisioning

- LinuxBoot can simplify provisioning a lot
 - Tested DHCP/TFTP implementations
 - Better protocols: HTTPS instead of TFTP
 - Consistent firmwares everywhere
 - We know and control what that we run



OPEN
Compute Project



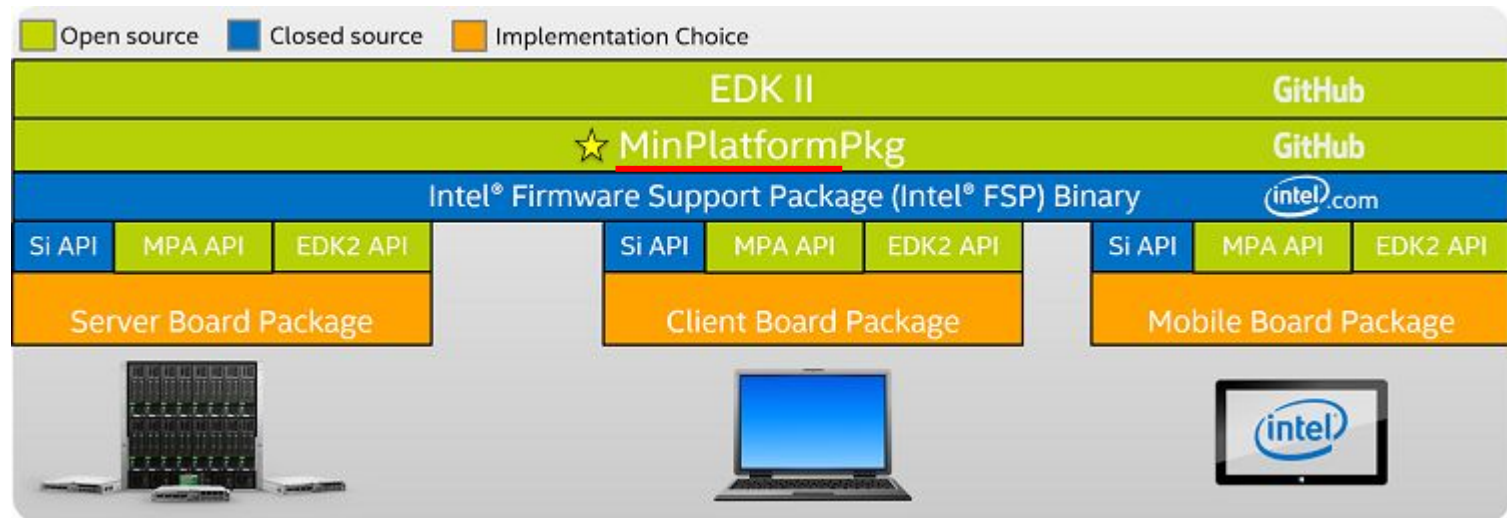
OpenBMC





<https://osfc.io/>





See: <https://www.youtube.com/watch?v=x3NFbUC3hkA>
and: <https://edk2-docs.gitbook.io/edk-ii-minimum-platform-specification>



Tianocore EDKII UEFI Firmware reference implementation

Overview

Arm is an active contributor to the EDKII project hosted by the Tianocore community.

The EDKII project is an open source project that provides a modern, feature-rich, cross-platform firmware development environment for the UEFI and PI specifications developed and maintained by the UEFI Forum.

Arm contributions make sure the EDKII project constantly keeps an up to date implementation of a UEFI compliant firmware on Arm systems.

Arm contributes to both the EDKII main repository, maintaining some core packages like DynamicTablesPkg and StandaloneMMPkg, and the EDKII platforms repository, hosting support for various Arm reference platforms as well as other 3rd party Arm-based platforms maintained by either Linaro or partners.



UNIFIED EXTENSIBLE
FIRMWARE INTERFACE



Questions?

Acknowledgements

J. Lowell Wofford
Cory Lueninghoener



Over 70 years at the forefront of supercomputing