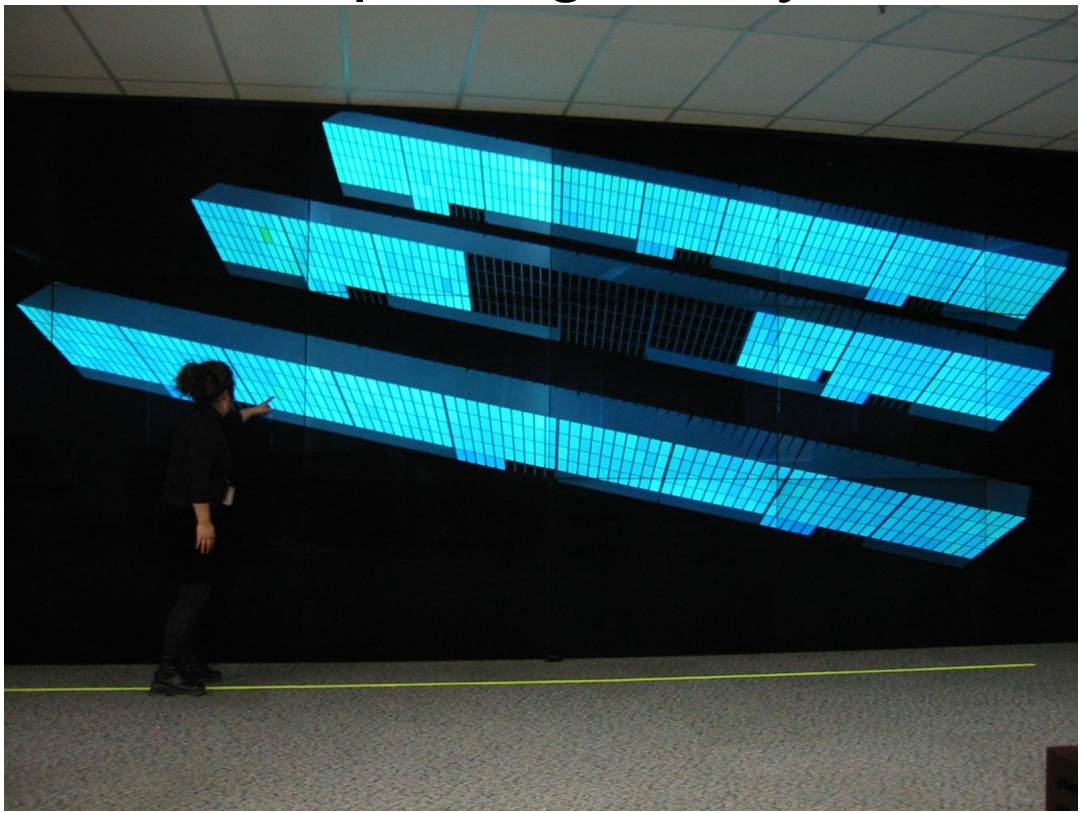
# LinuxBoot Ready is not Ready

Ron Minnich
Google

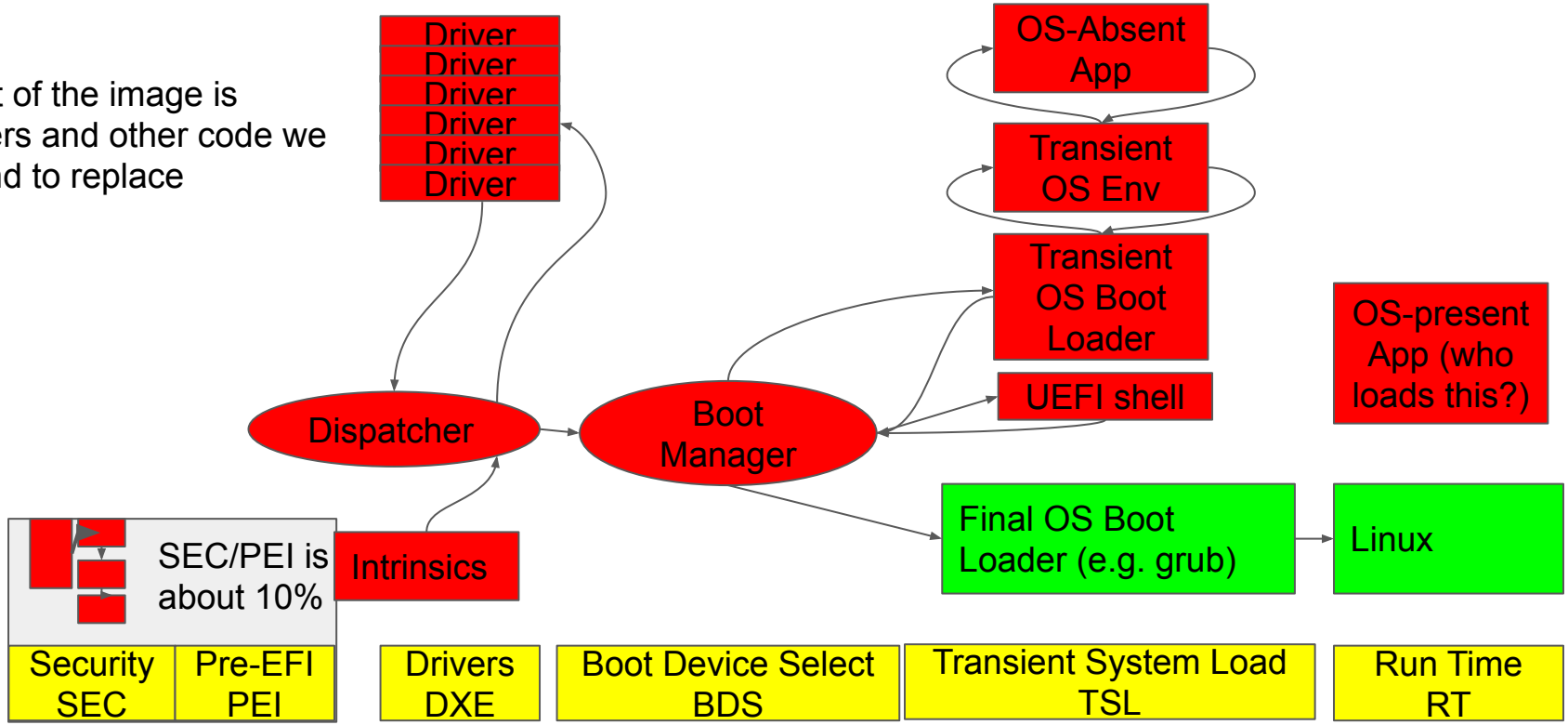
# LinuxBoot: Linux in firmware, replacing binary blobs

- Not really new
- LinuxBIOS supercomputer, 2002
- 100% GPL BIOS
- Used “kexec” (2-kernel-monte)
- Had to make devices work without BIOS

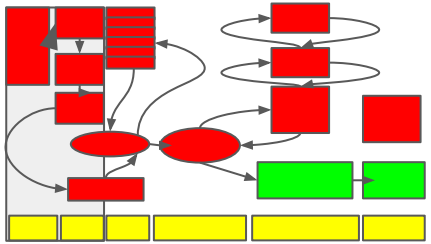


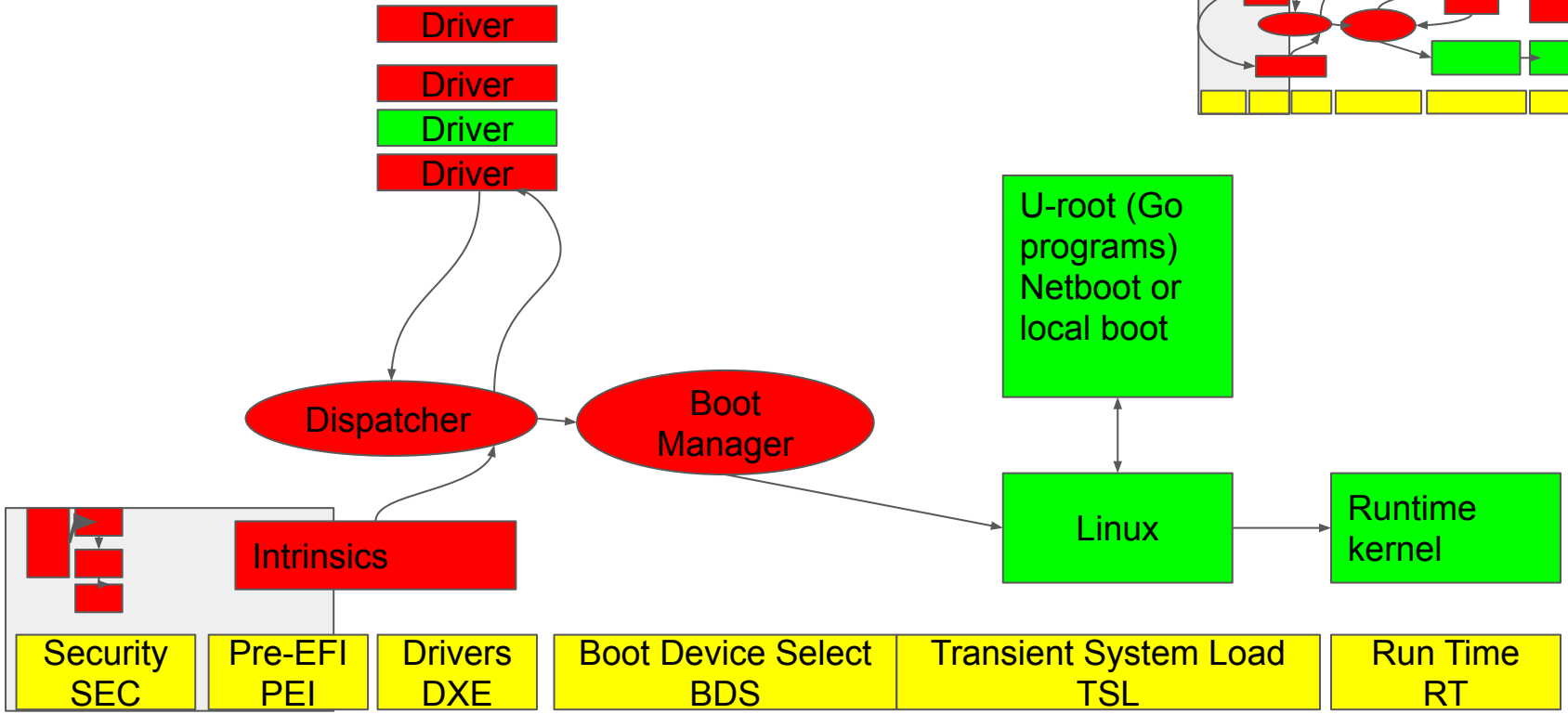
# UEFI components (Red is bad)

Most of the image is drivers and other code we intend to replace

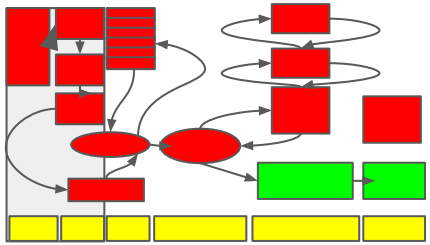


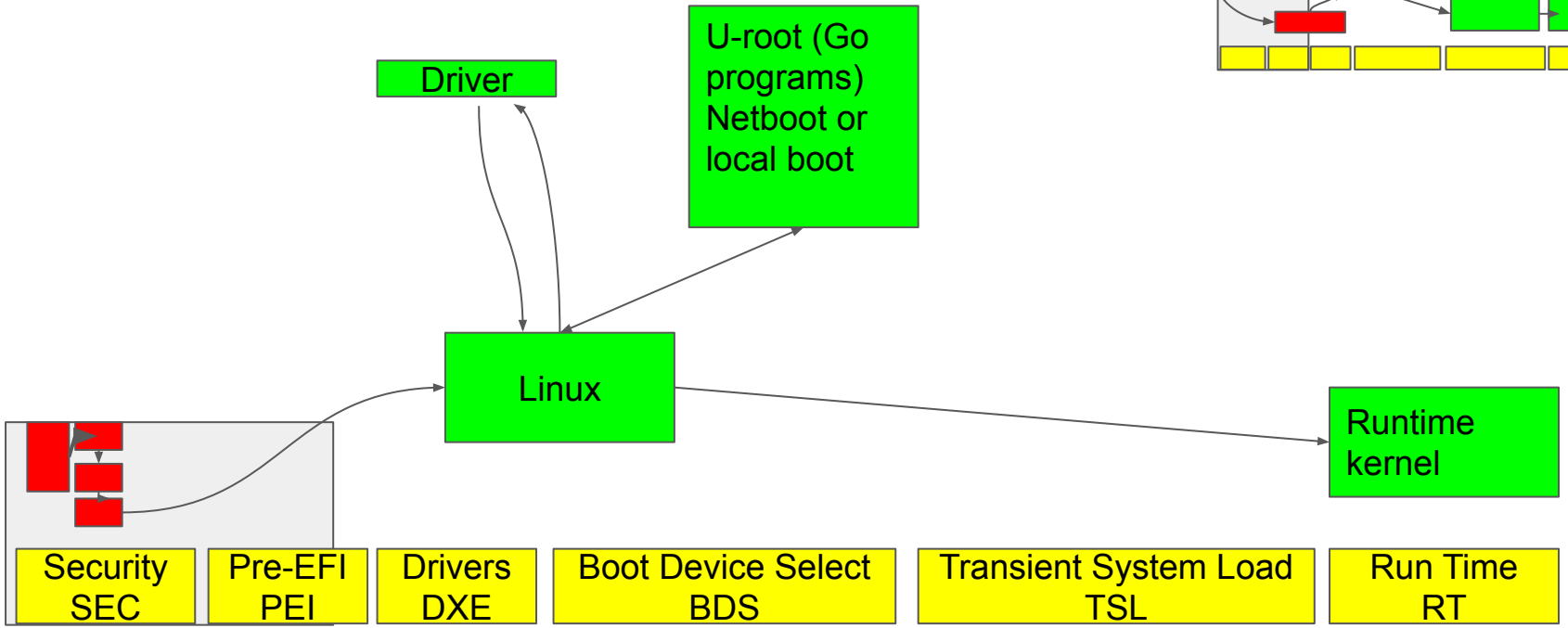
# Current UEFI LinuxBoot state





# 2021 Goal




Driver
U-root (Go programs) Netboot or local boot
Linux
Runtime kernel
Security SEC
Pre-EFI PEI
Drivers DXE
Boot Device Select BDS
Transient System Load TSL
Run Time RT

# Kexec *usually* works

- But driver problems can trip it up
- Can your driver start if the BIOS doesn’t help it?
- Some can, some can not

# 1999 : “... disabled by BIOS …”

```
/*
* Setup base registers for IDE command/control spaces for each interface:
*/
for (reg = 0; reg < 4; reg++)
        if (!dev->base_address[reg]) {
                fail...
        }
```

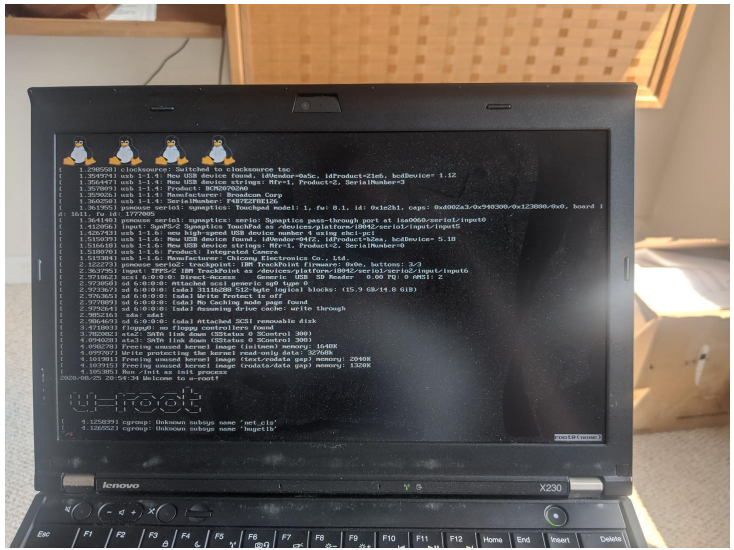
Tip: NOT ENABLED and DISABLED are not the same thing :-)

# 2018 AHCI driver

```
1.    diff --recursive -u ./clean/linux-4.9.80/drivers/ata/libahci.c linux-4.9.80/drivers/ata/libahci.c
2.    --- ./clean/linux-4.9.80/drivers/ata/libahci.c         2018-02-03 11:05:43.000000000 -0500
3.    +++ linux-4.9.80/drivers/ata/libahci.c       2018-02-07 18:02:32.526535910 -0500
4.    @@ -537,8 +537,12 @@
5.             }
6.
7.             /* fabricate port_map from cap.nr_ports for < AHCI 1.3 */
8.     -       if (!port_map && vers < 0x10300) {
9.     -               port_map = (1 << ahci_nr_ports(cap)) - 1;
10.    +       if (!port_map) { //  && vers < 0x10300) {
11.    +               printk("%s: saved_port=%02x\n", __func__, hpriv->saved_port_map);
12.    +               writel(0x1, mmio + HOST_PORTS_IMPL);
13.    +               port_map = readl(mmio + HOST_PORTS_IMPL);
14.    +
15.    +               //port_map = (1 << ahci_nr_ports(cap)) - 1;
16.                    dev_warn(dev, "forcing PORTS_IMPL to 0x%x\n", port_map);
```

# Shutdown issues

- Drivers don’t clear BME
- Drivers don’t reset state to “as powered on”



# Startup issues

- Drivers assume that BME is set (bug -- proper BIOS always clears it)
- Drivers assume that hardware is ‘pristine’
- Assumption that if a value is set in a register, it has to be good
  - E.g., interrupt vector numbering
- This is a bit harder than hotplug
- Hotplug needs to deal with hardware that has been reset
- This needs to deal with hardware that looks “misconfigured”

# Different kernel versions

- Booted a newer kernel from an older kernel
- The kernel/drivers had different ideas about interrupt numbering
  - “Off by 10”
- And, hence, interrupts were never delivered

# But it can work

- Recent test: Atomic PI, perpetual reboot, started Aug 18
- Ran for a week until “electric company event”
- So it could be branded LinuxBoot ready except
- I need to start having it boot different versions

# Decades-old problem

- More companies moving to LinuxBoot, problem is more visible
- We are proposing “LinuxBoot Ready”
- For any kernel, Linux 5.8 or later, a driver
  - Correctly shuts down so that kexec will work
  - Correctly restarts the driver no matter its state
  - Does not require BIOS interventions
- Can run on a totally open machine, in other words

# Certification

- Demonstrate 100,000 kexec cycles with no
  - Failure to boot
  - Reset to BIOS
  - Growth in memory footprint (e.g. by growth in E820 reserved areas)
  - Performance degradation
  - For any two kernel versions
- Will require driver/PCI subsystem changes
- May require hardware changes
- Some hardware may never work
- Need a way to mark drivers as “LinuxBoot Ready”
- Let competitive pressure do the rest for drivers and hardware designs