SteamOS - Adventures in Bootloading

Vivek Das Mohapatra

Collabora

November 24, 2023

COLLABORA

SteamOS - Adventures in Bootloading

Open First

2023-11-24

SteamOS - Adventures in Bootloading
└─Our Story Begins...
            └─SteamOS - Adventures in Bootloading

5 Years Ago

- More than 5 years ago
- Planning a major update for SteamOS

COLLABORA

2023-11-24

SteamOS - Adventures in Bootloading
└─Our Story Begins…

└─5 Years Ago

5 Years Ago

More than 5 years ago
Planning a major update for SteamOS
Goal: Atomic Updates

5 Years Ago

- More than 5 years ago
- Planning a major update for SteamOS
  - Goal: Atomic Updates

- The first commit 5 years ago, give or take a day
- There was some experimentation and research before that

Open First

What is an Atomic Update

> Either completely works or doesn't

Open First

What is an Atomic Update

- Either completely works or doesn't
- 2 OS image "slots"

What is an Atomic Update

- Either completely works or doesn't
- 2 OS image "slots"
- Update the one you're *not* booted from

Open First

What is an Atomic Update

- Either completely works or doesn't
- 2 OS image "slots"
- Update the one you're *not* booted from
- If it boots it's the new OS

Open First

SteamOS - Adventures in Bootloading

Open First

Implications of this approach

❯ Each image should be:

SteamOS - Adventures in Bootloading

2023-11-24

└─Implications

   └─Implications of this approach

Implications of this approach

❯ Each image should be:

Implications of this approach

- Each image should be:
  - Self contained

2023-11-24

SteamOS - Adventures in Bootloading
└─Implications
   └─Implications of this approach

Implications of this approach

- Each image should be:
  - Self contained

- Ideally knows *nothing* about other slots

Implications of this approach

➤ Each image should be:
  ➤ Self contained
  ➤ As "vanilla" as possible

Implications of this approach

➤ Each image should be:
  ➤ Self contained
  ➤ As "vanilla" as possible

➤ Ideally knows *nothing* about other slots

➤ changes from base OS should be as small as possible

Open First

Implications of this approach

- Each image should be:
  - Self contained
  - As "vanilla" as possible
  - Independently bootable

- Ideally knows *nothing* about other slots
- changes from base OS should be as small as possible
- Helps if we can take a single image and boot it

Implications of this approach

➤ Each image should be:
  ➤ Self contained
  ➤ As "vanilla" as possible
  ➤ Independently bootable

➤ Ideally knows *nothing* about other slots

➤ changes from base OS should be as small as possible

➤ Helps if we can take a single image and boot it

Open First

Who Boots the Bootloader?

➤ So how do we boot an image?

2023-11-24

SteamOS - Adventures in Bootloading
└─Implications

└─Who Boots the Bootloader?

Who Boots the Bootloader?

➤ So how do we boot an image?

Open First

Who Boots the Bootloader?

➤ So how do we boot an image?

➤ Universal Solution: Add a layer of indirection

Open First

Who Boots the Bootloader?

- So how do we boot an image?
- Universal Solution: Add a layer of indirection
- Stage 1 Loader

Who Boots the Bootloader?

- ❯ So how do we boot an image?
- ❯ Universal Solution: Add a layer of indirection
- ❯ Stage 1 Loader
  - ❯ Has 1 Job: Pick an image and boot it

SteamOS - Adventures in Bootloading
└─Implications

2023-11-24

        └─Who Boots the Bootloader?

Who Boots the Bootloader?
❯ So how do we boot an image?
❯ Universal Solution: Add a layer of indirection
❯ Stage 1 Loader
  ❯ Has 1 Job: Pick an image and boot it

- ❯ Instead of starting a kernel, start the next bootloader

Who Boots the Bootloader?

- So how do we boot an image?
- Universal Solution: Add a layer of indirection
- Stage 1 Loader
  - Has 1 Job: Pick an image and boot it
- Stage 2 (OS) Loader

SteamOS - Adventures in Bootloading
└─Implications

└─Who Boots the Bootloader?

2023-11-24

Who Boots the Bootloader?
- So how do we boot an image?
- Universal Solution: Add a layer of indirection
- Stage 1 Loader
  - Has 1 Job: Pick an image and boot it
- Stage 2 (OS) Loader

- Instead of starting a kernel, start the next bootloader

Who Boots the Bootloader?

- So how do we boot an image?
- Universal Solution: Add a layer of indirection
- Stage 1 Loader
  - Has 1 Job: Pick an image and boot it
- Stage 2 (OS) Loader
  - Does whatever it would normally do

2023-11-24

SteamOS - Adventures in Bootloading
└─Implications

└─Who Boots the Bootloader?

Who Boots the Bootloader?
- So how do we boot an image?
- Universal Solution: Add a layer of indirection
- Stage 1 Loader
  - Has 1 Job: Pick an image and boot it
- Stage 2 (OS) Loader
  - Does whatever it would normally do

- Instead of starting a kernel, start the next bootloader
- Doesn't have to know anything special is going on

Who Boots the Bootloader?

- ❯ So how do we boot an image?
- ❯ Universal Solution: Add a layer of indirection
- ❯ Stage 1 Loader
  - ❯ Has 1 Job: Pick an image and boot it
- ❯ Stage 2 (OS) Loader
  - ❯ Does whatever it would normally do

SteamOS - Adventures in Bootloading
└─Implications

      └─Who Boots the Bootloader?

2023-11-24

Who Boots the Bootloader?
- ❯ So how do we boot an image?
- ❯ Universal Solution: Add a layer of indirection
- ❯ Stage 1 Loader
  - ❯ Has 1 Job: Pick an image and boot it
- ❯ Stage 2 (OS) Loader
  - ❯ Does whatever it would normally do

- ❯ Instead of starting a kernel, start the next bootloader
- ❯ Doesn't have to know anything special is going on

SteamOS - Adventures in Bootloading

Open First

UEFI : What is it?

SteamOS - Adventures in Bootloading
└─The Details

└─UEFI : What is it?

2023-11-24

UEFI : What is it?

▸ Lets you use a PE32+ executable as a bootloader

▸ Lets you use a PE32+ executable as a bootloader

Open First

UEFI : What is it?

> Lets you use a PE32+ executable as a bootloader
> Provides a sort of kernel+libc layer

Open First

2023-11-24

SteamOS - Adventures in Bootloading
└─The Details

└─UEFI : What is it?

UEFI : What is it?

> Lets you use a PE32+ executable as a bootloader
> Provides a sort of kernel+libc layer

UEFI : What is it?

- Lets you use a PE32+ executable as a bootloader
- Provides a sort of kernel+libc layer
- Looks weird if you're used to POSIX

UEFI : What is it?

- ❯ Lets you use a PE32+ executable as a bootloader
- ❯ Provides a sort of kernel+libc layer
- ❯ Looks weird if you're used to POSIX
- ❯ Still <u>much</u> nicer than the Old Way™

2023-11-24

SteamOS - Adventures in Bootloading
└─The Details

          └─UEFI : What is it?

UEFI : What is it?

- ❯ Lets you use a PE32+ executable as a bootloader
- ❯ Provides a sort of kernel+libc layer
- ❯ Looks weird if you're used to POSIX
- ❯ Still <u>much</u> nicer than the Old Way™

Let's Count Our Yaks

➤ For our setup we need:

Open First

Let's Count Our Yaks

- ❖ For our setup we need:
- 🐃 UEFI firmware to start our loader

Let's Count Our Yaks

- For our setup we need:
  - 🐃 UEFI firmware to start our loader
  - 🐃 Our loader to find the other boot partitions

- OS images self contained, so each has own boot partition

Let's Count Our Yaks

➤ For our setup we need:
- 🐃 UEFI firmware to start our loader
- 🐃 Our loader to find the other boot partitions
- 🐃 To choose the right image boot partition

➤ OS images self contained, so each has own boot partition

Let's Count Our Yaks

> For our setup we need:
>> 🐃 UEFI firmware to start our loader
>> 🐃 Our loader to find the other boot partitions
>> 🐃 To choose the right image boot partition
>> 🐃 To start the next bootloader

> OS images self contained, so each has own boot partition

> It's not on the partition we started from - is that Ok?

Let's Count Our Yaks

➤ For our setup we need:
  🐃 UEFI firmware to start our loader
  🐃 Our loader to find the other boot partitions
  🐃 To choose the right image boot partition
  🐃 To start the next bootloader

Open First

SteamOS - Adventures in Bootloading
└─The Details

    └─Let's Count Our Yaks

2023-11-24

➤ OS images self contained, so each has own boot partition

➤ It's not on the partition we started from - is that Ok?

Yak Zero: Starting Our Loader

➤ UEFI boots:

2023-11-24

SteamOS - Adventures in Bootloading
└─The Details

└─Yak Zero: Starting Our Loader

Yak Zero: Starting Our Loader

➤ UEFI boots:

Yak Zero: Starting Our Loader

- UEFI boots:
  - A location specified by an NVRAM variable

Yak Zero: Starting Our Loader

- UEFI boots:
  - A location specified by an NVRAM variable
  - **OR** a well-known location on the 1st boot partition

Open First

Yak Zero: Starting Our Loader

- UEFI boots:
  - A location specified by an NVRAM variable
  - **OR** a well-known location on the 1st boot partition
- This Yak practically shaved itself. Onwards!

Yak One: List Other Partitions

❯ The API is there, and mandatory

Open First

Yak One: List Other Partitions

- The API is there, and mandatory
- Unusual use pattern, might not work

Open First

SteamOS - Adventures in Bootloading
└─The Details

└─Yak One: List Other Partitions

2023-11-24

Yak One: List Other Partitions

➤ The API is there, and mandatory
➤ Unusual use pattern, might not work
➤ Tried, it works

Yak One: List Other Partitions

➤ The API is there, and mandatory

➤ Unusual use pattern, might not work

➤ Tried, it works

Open First

Yak One: List Other Partitions

- The API is there, and mandatory
- Unusual use pattern, might not work
- Tried, it works
- Another Yak Shaved

Open First

Yak Two: Pick an Image

➤ Can we use NVRAM variables?

Open First

Yak Two: Pick an Image

- ❯ Can we use NVRAM variables?
  - ❯ Erased on firmware update

Open First

Yak Two: Pick an Image

- ❯ Can we use NVRAM variables?
  - ❯ Erased on firmware update
- ❯ Config Files?

Open First

SteamOS - Adventures in Bootloading
└─The Details

└─Yak Two: Pick an Image

2023-11-24

Yak Two: Pick an Image

➤ Can we use NVRAM variables?
  ➤ Erased on firmware update
➤ Config Files?
  ➤ UEFI Can open files on FAT filesystems

Yak Two: Pick an Image

➤ Can we use NVRAM variables?
  ➤ Erased on firmware update
➤ Config Files?
  ➤ UEFI Can open files on FAT filesystems

COLLABORA

Open First

Yak Two: Pick an Image

- ❯ Can we use NVRAM variables?
  - ❯ Erased on firmware update
- ❯ Config Files?
  - ❯ UEFI Can open files on FAT filesystems
  - ❯ Easy to do from the OS too

Open First

Yak Two: Some Details

COLLABORA

- ▸ Config file: simple key/value pairs
    - ▸ eg. `boot-requested-at:  20231122221729`

Open First

COLLABORA

Yak Two: Some Details

➤ Config file: simple key/value pairs

➤ eg. `boot-requested-at: 20231122221729`

➤ File with highest `brat` wins

Open First

➤ Easy to parse

➤ Talk about high-watermarking & layout change

Yak Two: Some Details

- Config file: simple key/value pairs
  - eg. `boot-requested-at:  20231122221729`
- File with highest `brat` wins
- Yak Shaved

- Easy to parse

- Talk about high-watermarking & layout change

COLLABORA

Yak Three: The Next Bootloader

Open First

Yak Three: The Next Bootloader

> ❯ Open chosen image's bootloader

Yak Three: The Next Bootloader

- Open chosen image's bootloader
- Sanity check the contents

- "Is this even an executable" etc

Yak Three: The Next Bootloader

- Open chosen image's bootloader
- Sanity check the contents
- Launch it

2023-11-24

SteamOS - Adventures in Bootloading
└─The Details

└─Yak Three: The Next Bootloader

Yak Three: The Next Bootloader
- Open chosen image's bootloader
- Sanity check the contents
- Launch it

- "Is this even an executable" etc
- Was this actually going to work?
- Launching from a different partition?

Yak Three: The Next Bootloader

- Open chosen image's bootloader
- Sanity check the contents
- Launch it
- Minimum Viable Yaks Shaved

SteamOS - Adventures in Bootloading
└─The Details

2023-11-24

└─Yak Three: The Next Bootloader

Yak Three: The Next Bootloader
- Open chosen image's bootloader
- Sanity check the contents
- Launch it
- Minimum Viable Yaks Shaved

- "Is this even an executable" etc
- Was this actually going to work?
- Launching from a different partition?

Well that sounded easy…

- ❯ There *were* some stumbling blocks along the way

Well that sounded easy…

➤ There *were* some stumbling blocks along the way
  ➤ Weird compiler bugs (we think)

➤ function worked only when static

Open First

Well that sounded easy…

- There *were* some stumbling blocks along the way
  - Weird compiler bugs (we think)
  - Flaky UEFI implementations

- function worked only when static

- open/close not ref counted

Well that sounded easy…

- There *were* some stumbling blocks along the way
  - Weird compiler bugs (we think)
  - Flaky UEFI implementations
  - SecureBoot

- function worked only when static
- open/close not ref counted
- SB: can't use the stock exec function
- SB: have to do memory layout by hand

Well that sounded easy…

- There *were* some stumbling blocks along the way
  - Weird compiler bugs (we think)
  - Flaky UEFI implementations
  - SecureBoot
  - No debugger for you

Open First

- function worked only when static
- open/close not ref counted
- SB: can't use the stock exec function
- SB: have to do memory layout by hand
- Hello printf my old friend

Well that sounded easy…

- There *were* some stumbling blocks along the way
  - Weird compiler bugs (we think)
  - Flaky UEFI implementations
  - SecureBoot
  - No debugger for you

Open First

- function worked only when static

- open/close not ref counted

- SB: can't use the stock exec function

- SB: have to do memory layout by hand

- Hello printf my old friend

Mission Accomplished

- We have a bootloader that is
  - Simple (ish)
  - Silent
  - Invisible

SteamOS - Adventures in Bootloading

Open First

Simple Text Menu

❯ UEFI provides a text console

Open First

Simple Text Menu

- UEFI provides a text console
- Can register a key handler

Open First

Simple Text Menu

- UEFI provides a text console
- Can register a key handler
- Simple menu allows user to select boot target

Simple Text Menu

- UEFI provides a text console
- Can register a key handler
- Simple menu allows user to select boot target
- But no "keydown" or "key held" detection

Simple Text Menu

- UEFI provides a text console
- Can register a key handler
- Simple menu allows user to select boot target
- But no "keydown" or "key held" detection
  - So how do we trigger the menu?

Graphics Mode Menu

❯ UEFI provides framebuffer & basic blitter

Graphics Mode Menu

- UEFI provides framebuffer & basic blitter
- No text, only pixels

Graphics Mode Menu

- ❯ UEFI provides framebuffer & basic blitter
- ❯ No text, only pixels
  - ❯ Use PFF2 fonts (`grub-mkfont`)

Graphics Mode Menu

- UEFI provides framebuffer & basic blitter
- No text, only pixels
  - Use PFF2 fonts (`grub-mkfont`)
  - Render pixel-by-pixel to a buffer

Open First

Graphics Mode Menu

- ❯ UEFI provides framebuffer & basic blitter
- ❯ No text, only pixels
  - ❯ Use PFF2 fonts (`grub-mkfont`)
  - ❯ Render pixel-by-pixel to a buffer
  - ❯ Blit to the framebuffer

Open First

Graphics Mode Menu

- ❯ UEFI provides framebuffer & basic blitter
- ❯ No text, only pixels
  - ❯ Use PFF2 fonts (`grub-mkfont`)
  - ❯ Render pixel-by-pixel to a buffer
  - ❯ Blit to the framebuffer
  - ❯ Write compliant UTF-8 & UTF-16 en/de-coders

Boot Failure Detection

- Stage 2 increments a counter in the config

Open First

Boot Failure Detection

- Stage 2 increments a counter in the config
- OS resets counter if boot is "good enough™"

Open First

Boot Failure Detection

- Stage 2 increments a counter in the config
- OS resets counter if boot is "good enough™"
- Counter too high: Stage 1 can show the menu

Open First

COLLABORA

SteamOS - Adventures in Bootloading

Open First

Any Questions?

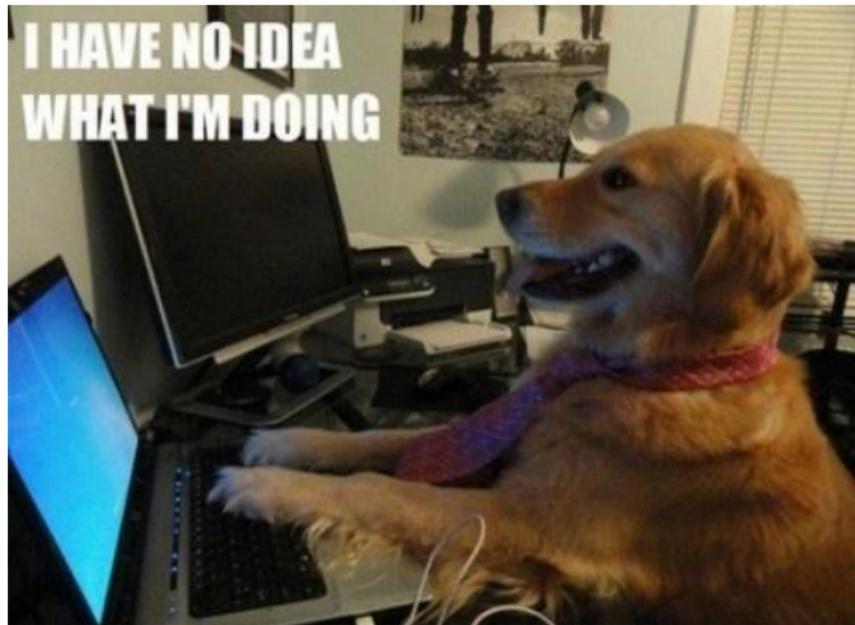2023-11-24

SteamOS - Adventures in Bootloading
└─That's All Folks

└─Any Questions?

Open First

Any Questions?

Open First