checkm8 exploit
&
checkra1n jailbreak

Student seminar: security protocols and applications

03/2020

Louis Merlin (247565)

Plan

1. What is a jailbreak ?

2. History

3. Deep Dive

4. Conclusion

1. What is a jailbreak ?

1. What is a jailbreak ?

- Privilege escalation on Apple's iOS

1. What is a jailbreak ?

- Privilege escalation on Apple's iOS

- Mobile carrier unlocking and device customization

1. What is a jailbreak ?

- Privilege escalation on Apple's iOS

- Mobile carrier unlocking and device customization

- First jailbreak came out a few days after the first iPhone (2007)

1. What is a jailbreak ?

- Privilege escalation on Apple's iOS

- Mobile carrier unlocking and device customization

- First jailbreak came out a few days after the first iPhone (2007)

- Can be tethered, semi-tethered, untethered or semi-untethered

2. History

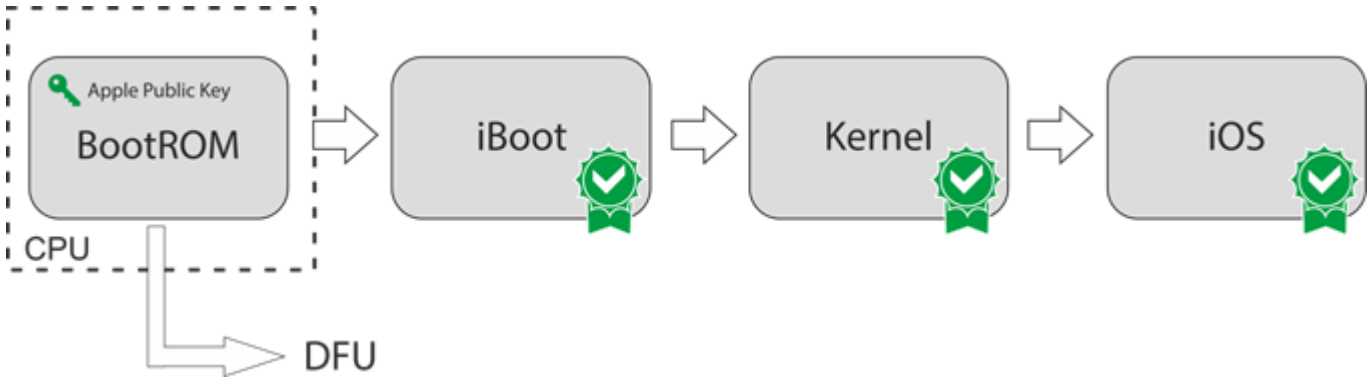
- Late 2019, by axi0mX

- Targets SecureROM (immutable code), so unpatchable

- Leads to

3. Deep Dive

SecureROM



https://habr.com/en/company/dsec/blog/472762/

3.1 - Normal use of DFU

- Start USB DFU protocol:
  - Input output buffer is allocated by the device in the ram
  - Address is given to usb stack via global variable
  - Image is transfered to the device
  - Image is copied to the memory location from where the boot will happen
  - Image is verified
- On DFU exit:
  - i/o buffer is freed
  - If parsing the image fails, the USB stack is activated again

3.2 - checkm8

- Start USB DFU protocol:
  - Input output buffer is allocated by the device in the ram
  - Address is given to usb stack via global variable
  - Image is transfered to the device INTERRUPT
  - ~~Image is copied to the memory location from where the boot will happen~~
  - ~~Image is verified~~
- On DFU exit:
  - i/o buffer is freed => use-after-free
  - If parsing the image fails, the USB stack is activated again

3.3 - A8 and A9 processors

- iPhone 6, 6+, 6s and 6s+

- Use-after-free overwrites USB stack

- USB tasks are a linked list of tasks

- Overwrite the current task's **next** value, and create that next task

- 🙌

3.4 - The rest...

- The ROM is deterministic, an the USB stack is not where we need it to be.

3.4 - The rest...

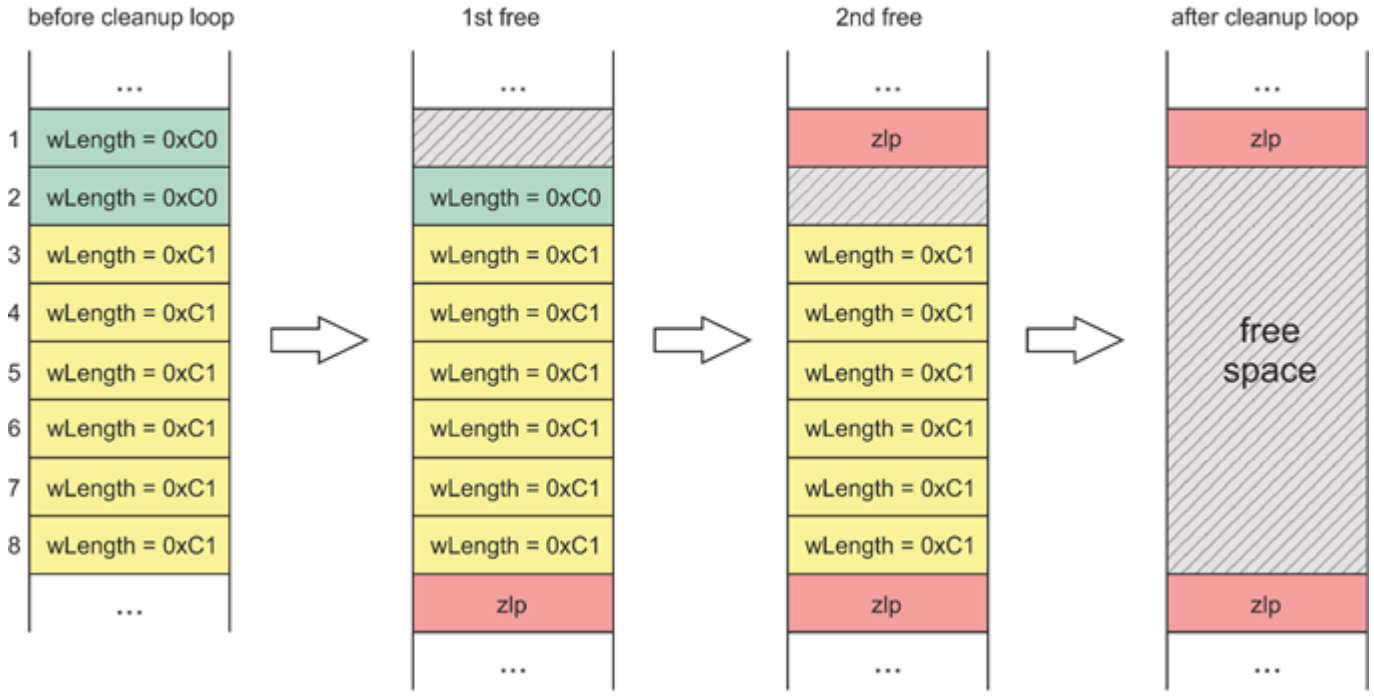
- The ROM is deterministic, an the USB stack is not where we need it to be.

- Heap Feng Shui to the rescue !

Heap Feng Shui

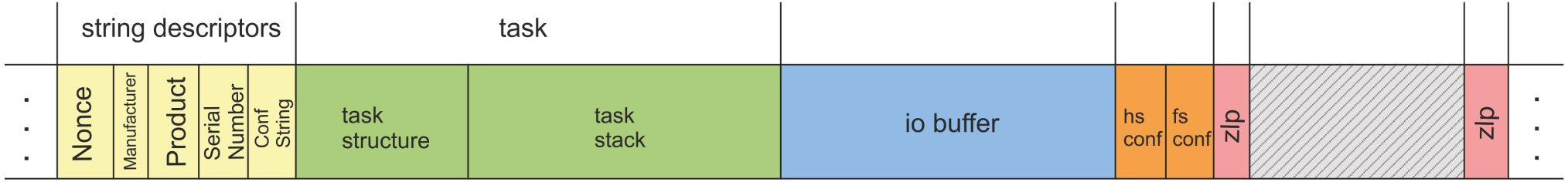
Requests that are not a multiple of 64 bytes do not trigger the leak !

heap layout

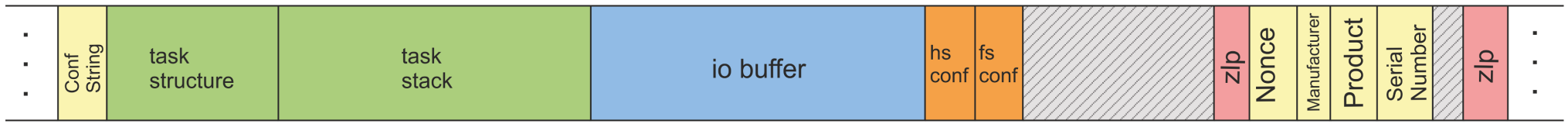


https://habr.com/en/company/dsec/blog/472762/

1st DFU iteration

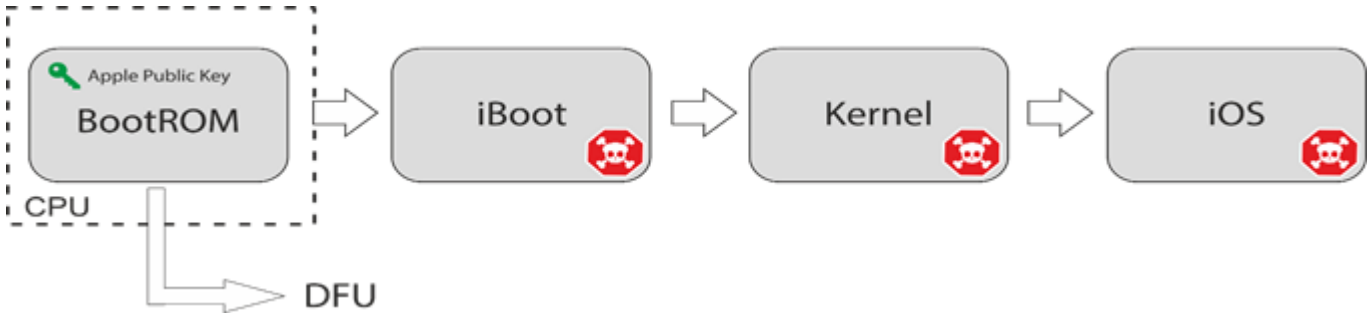


2nd DFU iteration



https://habr.com/en/company/dsec/blog/472762/

We want this



https://habr.com/en/company/dsec/blog/472762/

Payload

Final steps of exploit

- Restore USB configuration
- USB Serial Number = "PWND:[checkm8]"
- Replace USB request handler pointer

3.5 - The Bootkit

- Boot normally and compromise the stages one after the other

3.5 - The Bootkit

- Boot normally and compromise the stages one after the other
- **But** iBoot will reset all registers and wipe memory ("trampoline")

3.5 - The Bootkit

- Boot normally and compromise the stages one after the other
- **But** iBoot will reset all registers and wipe memory ("trampoline")
- We can hook bzero to avoid that

3.5 - The Bootkit

- Boot normally and compromise the stages one after the other
- **But** iBoot will reset all registers and wipe memory ("trampoline")
- We can hook bzero to avoid that
- Embed tiny ramdisk in kernel with hijack code

3.5 - The Bootkit

- Boot normally and compromise the stages one after the other
- **But** iBoot will reset all registers and wipe memory ("trampoline")
- We can hook bzero to avoid that
- Embed tiny ramdisk in kernel with hijack code
- We now have a fork(2) during system boot

4. Conclusion

- latest full jailbreak was on iOS 8 (2014)
- enables researchers to do security research
- dual booting
- Linux on iPhone ?

Illustrations and content

- https://habr.com/en/company/dsec/blog/472762
- https://media.ccc.de/v/36c3-11238-the_one_weird_trick_securerom_hates