Requirements Framework for Video Game Console Security

By: Hauton J Tsang

UNIVERSITY OF WATERLOO

Outline

- Introduction to the problem
- Explain the objective of the framework
- Initial requirements
- Additional requirements based on past cases
- Practicality of requirements
- Discussion and conclusion

Introduction

- Video game console development
- How can I protect my console from being exploited by hackers?

Objective of Protection

- Prevent unauthorized software from running on game consoles
  - Homebrew: software developed by people without permission of the console manufacturer
  - Pirated games: unauthorized copies of retail games

Initial Requirements

- Protecting physical games:
  - Physical game should contain a secret marker that is difficult to duplicate
  - Firmware of game reader should authenticate the marker
  - Physical game should boot up after marker is authenticated as genuine by the firmware of the console
- Protecting downloaded software:
  - Downloaded software should have cryptographic signatures issued by authorized developers
  - When any software is run on the console, the cryptographic signature should be verified before it is executed
- Is that enough?

Additional Requirements

- Case 1: PlayStation 1 [1]
  - Protection bypass:
    - Swapping disks after booting
    - Modchips

Image from: https://www.youtube.com/watch?v=XUwSOfQ1D3c&t=572s

- Additional Requirements
  - Physical game should be in a custom form-factor
  - Game reader should use cryptographic verification to ensure integrity
  - Firmware should authenticate physical game authenticity continually

Additional Requirements

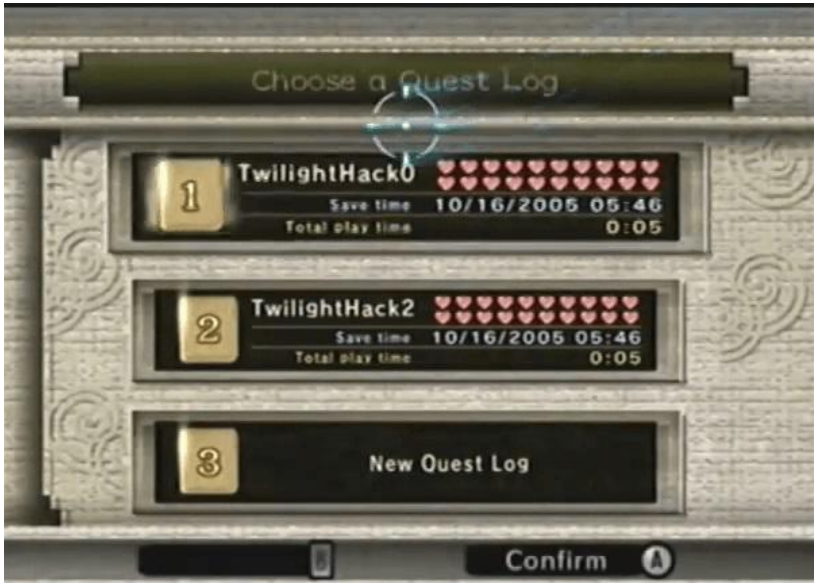
- Case 2: Original Xbox [2]
  - Protection bypass:
    - Drive swapping to modify secure drive contents

- Additional Requirements
  - Internal storage should be encrypted at rest
  - Encryption keys to decrypt storage should be hardcoded into a trusted element

Image from: https://www.youtube.com/watch?v=iV8B6eZVkBM&t=168s

Additional Requirements

- Case 3: Nintendo Wii [3]
  - Protection bypass:
    - Stack overflow due to a long character name in a Legend of Zelda: Twilight Princess save file
- Additional Requirements
  - Games should run in a sandbox that prevents custom code from running
  - Game save files should be cryptographically signed to prevent modification



1 TwilightHack0
Save time 10/16/2005 05:46
Total play time 0:05

2 TwilightHack2
Save time 10/16/2005 05:46
Total play time 0:05

3 New Quest Log

Image from: https://www.gamebrew.org/images/7/7f/Twilighthackwii2.jpg

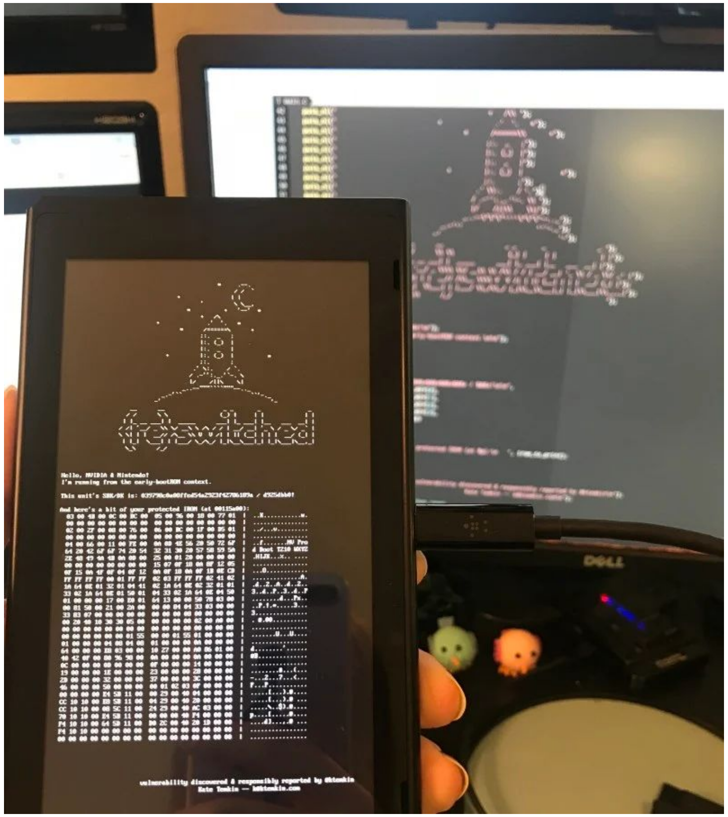
Additional Requirements

- Case 4: PlayStation Portable [4]
  - Protection bypass:
    - Service mode enabled by modifying battery serial number
    - Kernel exploit using vulnerable function
- Additional Requirements
  - Service mode should be secured using cryptographic keys
  - Console should contain hypervisor which isolates the kernel

Image from: https://www.youtube.com/watch?v=U8iZaxOPgjw&t=154s

Additional Requirements

- Case 5: Nintendo Switch [5]
  - Protection bypass:
    - An exploit found in firmware of a CPU from a third party
- Additional Requirements
  - Third party firmware should be audited for vulnerabilities



```
Hello, NVIDIA & Nintendo!
I'm running from the early-bootROM context.

This unit's SBK/DK is: 039798c0a00ffed54a2923f42706189a / d925dbb0f

And here's a bit of your protected IROM (at 00115a00):

vulnerability discovered & responsibly reported by @ktemkin
Kate Temkin -- k@ktemkin.com
```

Image from: https://www.bleepstatic.com/content/posts/2018/04/24/Fusee-Gelee.jpg

Final list of requirements

1. Physical game should contain a secret marker that is difficult to duplicate
2. Firmware of game reader should authenticate the marker
3. Physical game should boot up after marker is authenticated as genuine by the firmware of the console
4. Downloaded software should have cryptographic signatures issued by authorized developers
5. When any software is run on the console, the cryptographic signature should be verified before it is executed
6. Physical game should be in a custom form-factor
7. Game reader should use cryptographic verification to ensure integrity
8. Firmware should authenticate physical game authenticity continually
9. Internal storage should be encrypted at rest
10. Encryption keys to decrypt storage should be hardcoded into a trusted element
11. Games should run in a sandbox that prevents custom code from running
12. Game save files should be cryptographically signed to prevent modification
13. Service mode should be secured using cryptographic keys
14. Console should contain hypervisor which isolates the kernel
15. Third party firmware should be audited for vulnerabilities

Practicality of requirements

- Can I guarantee that my system is unhackable now?
    - No, because of the importance of **implementation**
        - Most exploits are due to oversights when implementing requirements, not requirements themselves
- How practical is it to implement all the requirements that were outlined before?

Practicality of requirements

- Low difficulty:
  - Physical game should contain a secret marker that is difficult to duplicate
  - Firmware of game reader should authenticate the marker
  - Physical game should boot up after marker is authenticated as genuine by the firmware of the console
  - Downloaded software should have cryptographic signatures issued by authorized developers
  - When any software is run on the console, the cryptographic signature should be verified before it is executed
  - Internal storage should be encrypted at rest
  - Service mode should be secured using cryptographic keys

Practicality of requirements

- Medium difficulty:
  - Physical game should be in a custom form-factor
  - Game reader should use cryptographic verification to ensure integrity
  - Firmware should authenticate physical game authenticity continually
  - Encryption keys to decrypt storage should be hardcoded into a trusted element
  - Game save files should be cryptographically signed to prevent modification

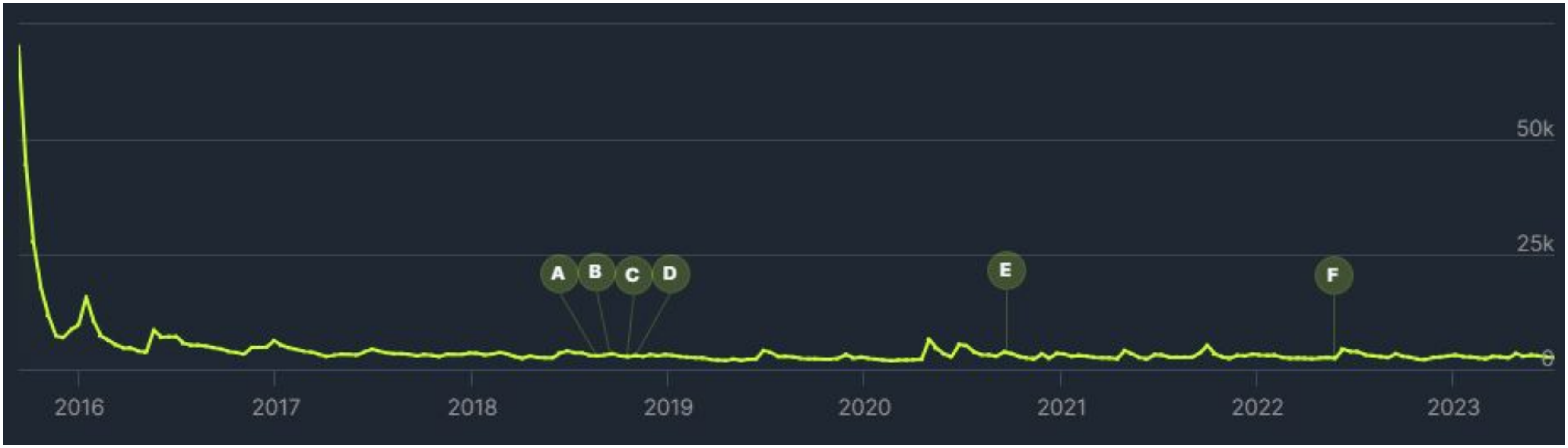
Practicality of requirements

- High difficulty:
    - Games should run in a sandbox that prevents custom code from running
    - Console should contain hypervisor which isolates the kernel
    - Third party firmware should be audited for vulnerabilities

Discussion and conclusion

- Is it **necessary** to create an unhackable console?
  - Economic impact lower if:
    - Prerequisite hardware needed to achieve exploit
      - Exploits requiring a specific game
      - Exploits requiring a modchip
    - Discovery of an exploit is delayed
      - Technical
      - Social

Discussion and conclusion



From: https://steamdb.info/app/287700/charts

Discussion and conclusion

- Changing scope
  - What is the motivation of game console hackers?
    - Hardware and software freedom [6]
  - What if we compromise and implement requirements that will delay possible hacks?
    - Xbox One (2013) developer mode

References

[1]https://wololo.net/2012/12/10/how-ps1-security-works/

[2]https://consolemods.org/wiki/Xbox:Hotswapping

[3]https://www.gamebrew.org/wiki/Twilight_Hack_Wii

[4]https://www.psdevwiki.com/psp/index.php/JigKick_Battery

[5]https://medium.com/@SoyLatteChen/inside-fus%C3%A9e-gel%C3%A9e-the-unpatchable-entrypoint-for-nintendo-switch-hacking-26f42026ada0

[6]https://www.youtube.com/watch?v=DUGGJpn2_zY

Thank you for listening!