Haaga-Helia
University of Applied Sciences

Digital Rights Management in video games: their impact on performance and the legal structure connected to them

Timothy Colangelo

Bachelor's Thesis
Degree Programme in Business Information Technology
2020

Haaga-Helia
University of Applied Sciences

Abstract

Date

Author(s)
Timothy Colangelo

Degree programme
Degree Programme in Business Information Technology

Report/thesis title	**Number of pages and appendix pages**
Digital Rights Management in video games: their impact on performance and the legal structure connected to them	**36 + 3**

Video games in the past few years have become the biggest entertainment market industry, raking in over tens of billions of dollars in the past five years.

This thesis focuses on Digital Rights Management functions in videogames and how DRM intertwines with video games in order to avoid piracy. The research goes into detail about the history and concepts of DRM and anti-piracy measures in both the past and present. However, the main focus of this research is DRM relating hinderances in video game performance. The study conducted focused on six video games, using four criteria to determine whether video games equipped with DRM perform worse than their DRM-free counterparts.

This research uses a correlationall quantitative approach: the data gathered, as mentioned earlier, comes from a pool of six games, using four different criteria, and compare results by using mathematical averages to empirically describe increase and decrease.

The research shows that performance in DRM-free video games is marginally better comparing to video games using DRM technologies, while the size of DRM-free video games shrunk significantly after DRM removal.

Keywords
internet, video games, security, performance, drm

Table of contents

1 Introduction

My past has been full of activities and such, but none of them took me so much into it as gaming.
I have been gaming for over 14 years of my life, and my interest and passion to the medium has been growing at such an extent that I am not only interested in the games themselves in terms of entertainment.
I have also got an interest in what goes behind the creation of the medium and how it works from a technological perspective.
From the engine used to the back-end and how multiplayer games can handle over 100 players in a game, video games are so fascinating.
It is no secret that videogames are one of the most pirated medium in entertainment, and this can be from different reasons.

For over two decades software houses, publishing houses and small studios try to protect their intellectual properties and products using copy protection and anti-tamper software, in order to keep piracy and the phenomenon of “cracking games” at a minimum.
The topic of this research will be the use of Digital Rights Management software.
In a more specific manner, the research will be centered around anti-tamper software, and the impact on a game’s performance.
Regarding the importance of this research, it can change how people see anti-tamper software and it can help small studios deciding if it is worth or not investing tens of thousands of Euros in anti-tampering and copy protection software.

1.1 The research question and objectives

The main question regarding this research is: does anti-tamper software hinder performance?
There are three main objectives for this research, specifically:

- How does an anti-tamper software integrate with a game (anything regarding executable file size, raw performance, loading times, function stacks)
- What do these integrations mean in terms of performance (too much processing power going towards the anti-tamper software, buffer overflows, RAM memory usage, so on)
- Does an anti-tamper software actually reduce the risk of pirated copies of video games spreading around.

The main objective of this research is to show that anti-tamper software can hinder performance when run in a video game and subsequentially drift users away from said game due

to these issues. It has to be said that this is a phenomenon which correlates more with games on PC, since video games on consoles are protected by their hardware and close environment in which the game runs on.
The results which will come out of this research can have a profound impact on small developers.
Game studios might choose not to use anti-tampering software for reasons which go beyond the mere economics: they can find how of an impact running these types of software can have in terms of performance.

1.2 Digital property

Currently, it seems like people can own anything they would like, but the truth is, we do own way less than we think we do, especially in the digital market.
The music we listen to, the TV series and movies we watch, and the games we play tie to a system of digital distributors, licensing policies, copy-protection platforms and anti-tamper software which keep all of our products secure and virus-free.
Still, they limit our freedom on what we can do with them.
For instance, if a track we like gets removed from our streaming platform of choice, we will not be able to listen to it, unless we do have access to the original file of said track.
This is the case with streaming platforms as the track is either not accessible or strongly encrypted to avoid copyright infringement and, therefore, piracy.

1.3 Piracy

Piracy is the practice of using someone's work without authorization or without getting the author retributed for their work (i.e. downloading copyright songs which the user should have paid for). Software piracy started in the 1970s as software enthusiasts were freely exchanging programs as a way to improve their understanding of the technology or to improve on their programming skills. As commercial software became more widespread, a community formed around the circumvention and distribution of copyright-infringed material through bulletin boards and Usenet networks, thus laying the groundwork for what is known today as the *scene*. During the late 1990s and early 2000s, as internet speeds rose and access to the internet was more economically viable, new ways of acquiring pirated content started to spread, such as using peer-to-peer protocols in applications such as *Napster, LimeWire, Kazaa,* and *eMule*. During the late 2000s and early 2010s a new peer-to-peer protocol technology emerged, which allowed users to download large files while being able to stop and resume downloads without affecting file integrity: BitTorrent.

1.4 BitTorrent

BitTorrent is a peer-to-peer technology established in 2001, and it allows to download multiple files simultaneously, by having a stream of packet data flow relating to different users which have previously completed the download (these users are known as “seeds”). This protocol enables download interruption as well as integrity check per packets to avoid data corruption during download.

1.5 Terminology

In this dissertation there will be some terms that might be unfamiliar to some people, here is an explanation of each term:

- CPU: Short for Central Processing Unit, also known as a “processor”, is the main unit of a computing machine and a flow of high speed calculation to run applications. It also directs and dictates how other components within the machine should behave;
- RAM: Short for Random Access Memory, it is a type of memory that stores chunks of application data temporarily in order to enable multi-tasking. This memory is volatile, which means that all the data within the memory will be erased once the machine shuts down.
- GPU: Short for Graphical Processing Unit, also known as a “graphics card” it is the unit responsible to manage highly intensive graphical tasks such as the rendering of 3D models and gaming environments. There are two types of GPUs commercially available: Integrated GPUs (or iGPU for short), a graphical processing unit integrated in the CPU which uses parts of the RAM as video memory. These GPUs are found in consumer grade CPUs for desktop and laptop computers. They are fairly low-powered and can not handle highly demanding graphical workloads, such as gaming. Dedicated GPUs, on the other hand, are dedicated processing units built specifically to calculate complicated mathematical calculations commonly found in tasks such as gaming, 3D rendering and cryptocurrency mining.
- Scene: the term refers to the community that focuses around exploiting, circumventing, and distributing copyrighted content free of charge. It is formed of groups which have strict rules when it comes to releasing material online: how is it released, the naming conventions and the distribution channels are all decided within the scene.

- Frame rate: the measuring of the number of frame changes in a second. It is measured in frames per second (FPS). The higher the frame rate, the smoother the reproduction of an image can be.
- 1% low frame rate: it is the average of the 1% lowest frame rate throughout a benchmark run. This data shows the consistency of frame times and how playable a game is.
- 0.1% low frame rate: it is the average of the 0.1% lowest frame rate throughout a benchmark run. This data shows the consistency of frame times and how playable a game is.
- Frame time: it is the time that passes by between the end of one frame, and the beginning of the following frame. It is measured in milliseconds. The higher the frame rate, the more stutter a player can experience while playing a game.

2 Digital Rights Management (DRM)

2.1 What is DRM?

Digital Rights Management is an “umbrella term” describing technologies, sets of tools and platforms that enable the control of protecting a piece of data, and it can be applied to any form of digital content, from simple PDFs to multi-layered executables and files within a video game.
Digital rights management tools and platforms have come a long way since their first inception in 1988, becoming more and more sophisticated and including even more and more layers of control and management.

2.2 History of DRM Implementation, digital distribution, and law enforcement regarding copyright matters

2.2.1 Software Service System (1988)

The first mention of a proto-DRM system was documented in “System and Computers in Japan, Volume 19, Issue 5” in the article “The Concept of Software Service System” by Ryoichi Mori and Shuichi Tashiro, where a system known as “Software service system” (or SSS) is described as a way to distribute and share content in which the customer pays an initial fee and then it is allowed to take a backup and is able to exchange the software without any restrictions. Its inception was a way “to protect software from unauthorized use and to encourage smooth marketing”.
This concept laid the groundwork for digital distribution services that billions of people nowadays use today, as it is a well-structured and fool-proof system to this day, as the article does also mention the introduction of a “property management function”, to help consolidate security and rights management.

2.2.2 Superdistribution (1990)

Ryuichi Mori reiterated on its “Software Service System” concept, using said ideas and concept to create the idea of “Superdistribution”, an approach which to digital distribution which divides itself in to three core functions: administration, accounting, and defense. To be able to take part in a superdistribution channel, one’s computer must have an “S-box”, which is “a digitally protected module containing microprocessors, RAM, ROM and a real-time clock”, as the primary goal of the latter is to ensure that information such as deciphering keys, and proprietary features of the system. This may sound like the CPU, GPU, RAM, ROM, and computer clock have to be physically encapsulated. However Mori concluded that the S-box could be created by printing an integrated circuit to the motherboard, which allows the components to communicate on a deeper layer of the system, thus protecting it in a digital form.

2.2.3 WIPO Copyright Treaty (1996)

The World Intellectual Property Organization Treaty Act (or WIPO WCT Act) is an act that ratifies and stipulates rules and terms for copyrighted content. In this document Computer Programs are classified as “literal works within the meaning of Article 2 of the Berne Convention”. (World Intellectual Property Organization Copyright Treaty Act, 1996, WIPO)
This act was ratified and implement within the United States of America law by the Digital Millennium Copyright Act (or DMCA) and by the European Union law under the European Union Directives, such as Directive 91/250/EC, Directive 96/9/EC and Directive 2001/29/EC for the implementation of the creation of copyright protection for software programs, databases and establishing the illegality of devices that allow the circumvention of DRM technologies.

2.2.4 Digital Millenium Copyright Act (1998)

The Digital Millennium Copyright Act of 1998 is an act which enforces in to law the 1996 WIPO WCT Act and includes exceptions for copyright infringement activities for law enforcement, intelligence, and other governmental activities.
These six exceptions are:

1. Nonprofit library, archive and educational institution exception
2. Reverse engineering
3. Encryption research
4. Protections of minors
5. Personal privacy
6. Security testing

This act also takes in to account Article 12 of the WCT which states that the removal of a DRM without any authority consenting the procedure is considered illegal, as well as the paid distribution of tampered copyrighted content.
The DMCA is, now, the base for copyright standards across the Internet.

2.2.5 OMA DRM (2004 – present)

OMA DRM is the first proper Digital Rights Management standard platform used on a large scale.

Created by the Open Mobile Alliance, a consortium of mobile phone manufacturers, mobile operating system development companies, phone carriers and information technology companies. Its main scope was to provide a way to avoid that any sort of digital content could be duplicated and share by users who happened to own a handset of one of these companies.
There are two versions of OMA DRM Standards: OMA DRM 1.0 and OMA DRM 2.0, and OMA DRM 2.1.

2.2.6 OMA DRM 1.0 (2004)

The first implementation of OMA DRM was set in force in 2004 and this laid out the basic set of standards which companies would have to obey to deploy an instance of OMA DRM within their lines of devices. This basic DRM methods included: Forward locking, which disables the option of forwarding protected content between two devices, for instance ringtones and wallpapers; Combined Delivery, which enabled the delivery of both content and the “Rights Object” (a key) in the same DRM message; and Separate Delivery, which sees the content and the Rights Object being delivered separately.
Both content and Rights Object are delivered by the HTTP or WAP protocol.

2.2.7 OMA DRM 2.0 (2006)

The second version of OMA DRM implements new security features, such as a Public Key Infrastructure (PKI), and a new protocol called Rights Object Acquisition Protocol (ROAP), which are not only crucial to maintain the systems secure using these protocols for authentication. They are used to check file integrity.

2.2.8 OMA DRM 2.1 (2009)

OMA DRM 2.1 introduces new features, which include content usage metering, Rights Object upload and Rights Object installation confirmation. Other new features include the introduction of a Content Issuer (CI), which handles the delivery of the protected content, and a Rights Issuer (RI), which handles the issuing of Rights Objects to the DRM Agent. The DRM Agent is an agent that dictates what users can or can not do with content downloaded from an RI or CI, even though the user controls it.

2.3 DRM in the present day

Nowadays DRM platforms, tools and systems are geared towards a new, and ever-growing market: gaming. As conventional media consumption fully entered the digital age thanks to streaming services, most of the companies working in the DRM industry gear towards video games and software houses. PC gaming has always been hit harder by piracy, so DRM implementations have been effective in to limiting damages. At the time of writing, more and more publishers decide to move towards having their own “launchers”, programs that implement a storefront to access games.

2.4 DRM technologies and anti-piracy measures

Today when talking about digital rights management technology, majority of people would think of anti-piracy software. This, however, is not the case.
Digital rights management platform started to implement anti-piracy measures and technologies in the early 2000s, as piracy awareness efforts were at their peak.

2.5 History of anti-piracy in video games

Video game piracy was a prominent phenomenon during the late 1980s through the 1990s and it is still an issue that affects certain regions (namely Russia, South-East Asia, and South America). However, the first piracy efforts began in the 1980s. There were two distinguished scenes, which implied different circumvention techniques: the home and personal computer video game scene, and the console scene.

2.6 Computer video game piracy and copy-protection methods in the past

Computer video game piracy was and still is the most widespread type of piracy, due to its relative simplicity and low-cost barrier.

2.6.1 Punch cards, and cassettes tapes (late 1970s – early 1980s)

Early home computers executed code using punch cards and cassettes tapes. These storage methods would allow copies to be duplicated and distributed easily, as there were no copy-protection techniques available.

2.6.2 On-disk copy protection (1980s)

With the rise in popularity of floppy-disks, floppy-disks readers and floppy-disks burners, developers needed to find a way to make video games and software harder to copy. The main process was to create bad sectors, odd memory address calls and marks, intentional differentiation of track layout and file encryption. These methods would be used during printing to deter copying, as the software would not run if the disk were not inserted it in the reader. Software with on-disk copy protection would not be able to be backed up in case the disk would malfunction, as floppy disks were fragile and susceptible to dust, sudden magnetic field changes, temperature variations and so forth.

2.6.3 Off-disk copy protection (late 1980s – early 1990s)

Off-disk copy protection became a more widespread and harder to defeat method of copy protection in the late 1980s and early 1990s. The main reason why off-disk copy protection was popular is that it could be creatively implemented: passphrases found in the game manual, multi-layered code wheels with different combinations, etc. The game would become unplayable or impossible to finish if one would not have said codes.

2.6.4 Physical locks (1980s – 1990s)

Physical locks were objects that the user need to have in order to access levels later in a specific game. These methods would range from specifically colored lenses to track certain objects to a set of lenses with prisms, known as Lenslok.

2.6.5 Lenslok (1980s – 1990s)

Lenslok^{TM} was a physical copy protection method which used a lens with prisms to decode a two-letter code which was used by the program to load. The biggest issue with this method was that it barely worked most of the times, due to calibration issues and screen sizes. It was used in eleven releases across the 1980s and 1990s.

2.6.6 Dongles (1980s – 1990s)

Dongles were physical objects that required to be connected to a communication port in one's computer to enable code execution. This method was largely used with high-end software, although some video games used it.

2.6.7 Serial numbers and Alphanumeric keys (1990s – Late 2000s)

During the 1990s serial numbers and alphanumeric keys became the most used copy protection mechanism in video games and computer software in general. The user was required to enter a serial number or key made up of letters and numbers during installation. Each copy of a video game was tied to a unique serial key or number, and this method was used in conjunction with CD checks in order to run the program once installed.

2.6.8 Dummy files, fake TOCs and oversized CDs (1990s)

This copy protection method was effective but short lived. Game developers created fake large files and table of contents (TOC) to deter CD burning, and it was effective. CD burners at the times could not see these files, so the burner would refuse to copy the CD due to its size. But after a few years new CD burners were capable of seeing these files and bypass burning of said files, so the game would fit on a normal 600MB CD-ROM.

2.7 DRM technologies become anti-piracy measures

After entering the new millennium, game developers and publishers would reach companies to research, create, and apply new anti-piracy technology. Software piracy reached its peak in the early 2000s, largely thanks to the increase of internet speeds worldwide. This shift in network technology made F2P sites and P2P protocols such as Torrent easier and faster to use. Download and upload speed were increasing significantly, but so were game file sizes due to more complex technologies and techniques getting introduced in this era.

2.7.1 SecuROM (1997-2016)

SecuROM^{TM} was a third-party digital rights management, content manager and authentication technology developed by Sony DADC in 1997. It has become one of the most controversial digital rights management platforms in history.
The way SecuROM^{TM} worked is by using different methods in different stages of production and mastering of a video game to achieve a high level of security. It must be said that SecuROM^{TM} was a modular technology, allowing developers to use features they would consider important. The main technique used with SecuROM^{TM}, however, was to attach a unique key to every disk during printing, which was impossible to copy and burn on a CD-

ROM. This key is used to authenticate the copy and decrypt parts of the executable, which allowed the game to boot and run. If the key check failed, the game would not boot.
Crackers found a way around by checking what the DRM was calling during its operation. It turns out that SecuROM triggers an interrupt from MSCDex (Microsoft CD-ROM Extension). It is also worth noticing that SecuROM uses self-modifying code to obfuscate parts of the executable thanks to a Windows API call, *WriteProcessMemory*. An original copy of a game using SecuROM would call this function four times, while a backed-up copy would call it three times and fail.
The first SecuROM cracks would capture the decrypted code from the aforementioned function calls, patch it into the executable using *SoftDice*, a tool which allowed to patch in memory code in executables and return a value of 0 on every *WriteProcessMemory* call, which allowed the code to run.
This process was key in to creating what are known as *NoCD Cracks*, executables which would replace the original executable stored on the hard disk, allowing one to play a game without a CD.
During the 2000s, SecuROM introduced online authentication in the form of alphanumeric serial keys on top of the CD checks to toughen up security.
In 2008 one of the biggest and most ambitious PC games ever would launch. This game was *Spore*, a project created by Will Wright, a visionary game designer known for industry-defining products such as *SimCity* and *The Sims*.
Spore would also boast a newer and even tougher version of SecuROM, with a revised online authentication method, which would require the user to log in every ten days to keep its copy active. This method was dropped after multiple complaints. The second iteration of this new version of the DRM used limited online activations, which means that the game can be activated only on a set number of computers, three in this case. However, this method was really sensitive, due to the fact that the DRM treated a computer as a new system even if only one component was swapped, thus using an authentication slot.
In September of 2008, a class-action lawsuit was filed against publisher *Electronic Arts* for not disclosing the inclusion of SecuROM on the box and its functioning mechanism, which had more in common with malware.
Allegedly, SecuROM's main source code would sit in a computer's kernel (also known as Ring 0), granting access to all features, functions, and low-level embedded programs in the system, becoming what is known as a *Rootkit*. For this specific purpose, SecuROM would install an executable known as *UA7service.exe* to allow the main source code to interact with machine registry, process handles, calls and much more. This created a lot of issues to end users which would see programs such as Task Manager completely unable to being used, system instability, crashes, freezes and Blue Screens of Death (also known as

BSODs). Over three thousand reviews and complaints were filed. After the lawsuit, Electronic Arts decided to launch a new version of the game with the copy-protection technology removed. Despite all of this, many Electronic Arts releases at the time would implement SecuROM, creating the same amounts of issues to end users worldwide.
In the 2010s SecuROM declined in popularity to the point that Microsoft dropped support for this technology on its new operating system, *Windows 10*, in 2016, deeming it a “security flaw”.

2.7.2 SafeDisc (1999 – 2009)

SafeDisc was a DRM technology created and owned by Macrovision Corporation. Market share figures show that SafeDisc had a 40% market share in the DRM market at its peak, and it started to decline after SecuROM disrupted the industry.
SafeDisc was applied as a copy-protection technology using a digital signature to the disk during the manufacturing process, which was hard to bypass and burn at the time. Other than that, SafeDisc also used an authentication technology to check the signature during the executable launch. Some products using SafeDisc had all the code and assets stored in the end user’s hard drive after installation, meaning that the optical media was used only for authentication purposes. A non-marketed feature of SafeDisc was support for virtual drives, in other words, it was possible to create a disk image of the optical media to use to launch the application, although the original CD or DVD was needed for authentication, as the digital signature was not copied over.
As mentioned in a data sheet provided by Macrovision, SafeDisc’s latest version, SafeDisc Advanced, shipped with a feature known as Asymmetric Code Blending (ACB), a technology that allows to tie Secure Data Types to an executable to obfuscate and prevent any tempering attempt.
SafeDisc was discontinued in March 2009, and since then Microsoft dropped support for the technology in 2016 in the same Windows 10 update that dropped support for SecuROM.

2.7.3 StarForce DRM

StarForce DRM was a copy-protection and DRM technology developed by StarForce Technologies.
Its main operating technique conceals the executable and .dll files in bytecode and runs those files in a virtual environment. These files get decrypted, translated, and run by the application.

StarForce was the first digital rights management technology and copy-protection software that became controversial: on one hand, if well implemented, the technology could protect a game for well over a year; on the other it was claimed that StarForce needed Ring-0 kernel permissions for installation, it would degrade mechanical hard drives by bottlenecking their speed, and it would cause system instability.
StarForce is still used today, and the company expanded its technology to any means of digital media. Some of StarForce's biggest clients include game developers like *Riot Games, Wargaming*, and many Russian companies including *Rostelekom, Gazprom,* and *Aeroflot*.

2.8 Console video game piracy history: the rise of ROMs and emulators

Video games on PC are easily getting the upper hand when it comes to piracy: due to a computer's open nature, technologies like the aforementioned (see Section 3.3) became vulnerable very quickly.
It can not be said the same thing for video game piracy on video game consoles. The current generation of consoles (Sony's *PlayStation 4*, Microsoft's *Xbox One*, and their intermediate updates as the *PlayStation 4 Pro* and *Xbox One X* respectively) make video game piracy a useless matter, as many console video games require online connectivity, on top of hard-ware copy-protection and anti-tamper technologies, which are really hard to bypass. On top of that, the close nature of these systems makes running unprotected copies of a video game far harder than expected.
All this progress made up for a series of mistakes and flaws that video game consoles manufacturers left on their systems. In the fourth, fifth and sixth generation of video game consoles, enthusiasts would find their way around a console's copy protection by reverse engineering chips and instruction sets to exploit flaws and bypass security measures. After that, they would start to manufacture modification chips (ModChips) which were soldered to a console's motherboard in order to bypass security measures. ModChips would allow bootlegged cartridges or optical media to run, as well as bypassing geographical restrictions (also known as Region Locking).
Another reason why console video game piracy was rising in the 1990s was the rise in popularity of emulators, programs that would make possible to run code not meant for a computer hardware by mimicking hardware functionalities using code. Emulators would need Read-only memory dumps of a cartridge or a CD-Rom to make games run, thus the popularity of ROMs would skyrocket.
In the last few years, many game companies, with Nintendo on the frontline, started to acknowledge how damaging this practice was to the industry and started to crack down on

ROM-hosting websites for copyright and intellectual property infringement, as many of the video games provided by these websites were and are still not part of the public domain.
It must be said that Nintendo is the company which was and still is mostly hurt by ROMs and emulators, as their latest handheld consoles are heavily affected by the phenomenon, as they are easily reversed engineered and emulated.

2.8.1 Nintendo's Checking Integrated Circuit (CIC) (3rd – 5th Console Generation)

Nintendo has been working hard on stopping piracy on its system since the late 1980s, with the release of the *Nintendo Entertainment System* (known in Japan as *Famicom*). This machine used an unusual yet effective anti-piracy method.
In 1985 Katsuya Nakagawa, an engineer working at Nintendo filed a patent, *system for determining authenticity of an external memory used in an information processing apparatus*, which laid out the foundation of what is known as the *10NES*, or *CIC.*
It (*the patent*) describes a lock-key system: the cartridge would have a key-device on its board, while the console hosts a lock-device. When the cartridge gets inserted in the system, both the key-device and the lock-device would run the same custom instruction set in lockstep. Both chips know what the output of each chip would be, if the output is not the same, the system locks out, failing to run the game. On the other hand, if the operation had been successful, the game would have run properly.
The first people who attempted to bypass the CIC would spike the voltage on the lock-chip, to disable it and skip the check. When Nintendo acknowledged the issue, a new version of the key-chip was used, which detected a disabled lock-chip and failing the check, thus locking the system.
After years of attempts to reverse engineer the system, in 2010 the homebrew community was able to "decap" the CIC (decapping is a process where the top part of the chip gets removed by using chemicals or processes such as sanding, to expose the die and see the internal functioning). It revealed that the chip was a 4-bit chip manufactured by Sharp. The chip has two registers, A or the accumulator register, and X, the secondary register. On top of that, there is a 6-bit B register (also known as the pointer register) which is the registry responsible for accessing the RAM, and a carry flag, denominated with the letter C. Alongside the registers and carry flags, there is a process counter, which is 10 bits. The RAM used in this chip is 512 bytes split in to three 128-byte banks each. The counter used a polynomial counter, rather than a binary one.

```
                  +------------------+
  DATA_OUT <--  | 1 P0.0    +5V 16 |
   DATA_IN -->  | 2 P0.1        15 |  ?
      SEED -->  | 3 P0.2        14 |  ?
  LOCK/-KEY --> | 4 P0.3        13 |  ?
                | 5 Xout   P1.3 12 | <-- RESET_SPEED_B
                | 6 Xin    P1.2 11 | <-- RESET_SPEED_A
                | 7 RESET  P1.1 10 | --> SLAVE_CIC_RESET
                | 8 GND    P1.0  9 | --> -HOST_RESET
                +------------------+
```

1. The Checking Integrated Circuit pinout

```
"skip" means "do not execute next instruction"
"M" means "the RAM nybble addressed by B"
"BL" means "the low four bits of B"
"BM" means "the high two bits of B"
"PN" means "I/O port number BL"
"x.y" means "bit y of x"

00+N  adi N     "add immediate", A := A + N, skip if overflow   (00 is nop)
10+N  skai N    "skip acc immediate", skip if A = N
20+N  lbli N    "load B low immediate", BL := N
30+N  ldi N     "load immediate", A := N

40    l         "load", A := M
41    x         "exchange", swap A with M
42    xi        "exchange and increment", swap A with M, increment BL, skip if overflow
43    xd        "exchange and decrement", swap A with M, decrement BL, skip if underflow
44    nega      "negate acc", A := -A (two's complement)
46    out       "output", PN := A
47    out0      "output zero", PN := 0
48    sc        "set carry", C := 1
49    rc        "reset carry", C := 0
4a    s         "store", M := A
4c    rit       "return", pop PC from stack
4d    ritsk     "return and skip", pop PC from stack, skip
52    li        "load and increment", A := M, increment BL, skip if overflow
54    coma      "complement acc", A := ~A (ones' complement)
55    in        "input", A := PN
57    xal       "exchange acc and low", swap A with BL
5c    lxa       "load X with acc", X := A
5d    xax       "exchange X and acc", swap X with A
5e     ?        SPECIAL MYSTERY INSTRUCTION

60+N  skm N     "skip memory", skip if M.N = 1
64+N  ska N     "skip acc", skip if A.N = 1
68+N  rm N      "reset memory", M.N := 0
6c+N  sm N      "set memory", M.N := 1

70    ad        "add", A := A + M
72    adc       "add with carry", A := A + M + C
73    adcsk     "add with carry and skip", A := A + M + C, skip if overflow

74+N  lbmi N    "load B high immediate", BM := N

78+N NN   tl NNN    "transfer long", PC := NNN
7c+N NN   tml NNN   "transfer module long", push PC+2, PC := NNN
80+NN     t NN      "transfer", low bits of PC := NN
```

2. The Checking Integrated Circuit instruction set

Nintendo used this system for over three console generations, from the Nintendo Entertainment System in 1985 to the Nintendo 64 in 1995. In the homebrew community these chips are known as the CIC, the Super CIC resident in the Super Nintendo Entertainment System (*SNES* or *Super Famicom* in Japan), and the Ultra CIC, which is resident on the Nintendo 64. Today, all these chips are open source and readily available to the general public.

2.8.2 SEGA Dreamcast: GD-ROMs and exploitation of the MIL-CD format

Compared to Nintendo, SEGA had an easier time with piracy not only because their systems were not as popular as Nintendo's, but they were well protected. Their 6th generation console, the *Saturn*, had a copy protection scheme that remained undefeated after its lifecycle.

SEGA's 7th generation console, the *Dreamcast*, had a shorter lifecycle, as it was released in 1999 and was discontinued in 2001 due to poor sales figures. Not only that, but its copy protection system was defeated earlier in its lifetime, enabling to run burned games.

SEGA used a disc format known as a GD-ROM (Gigabyte Disk). GD-ROMs were optical media that could store over 1 GB of data, compared to a normal CD-ROM, which could store only over 600MB of data. The way SEGA achieved this was to print tracks closer to each other. GD-ROMs were split in to three sectors: the first sector was a low-density sector that stored game metadata and an audio file; while the second, high-density sector store the game data. The audio track was only readable via a CD-ROM drive, and it stated the following:

"*This disk is meant to be used only on SEGA Dreamcast*"

It must be said that the console did not have an operating system, so every time the system booted with a GD-ROM inserted, it would run a boot sequence. The boot algorithm runs as follows: during boot the BOOTROM chip would load the bootstrap sequence from the optical disk to the system's RAM. After that, the BOOTROM would locate a file known as *IP.BIN*, a binary file which stored metadata, including information regarding the game, region, the game's applicative name etc. If the *IP.BIN* check was successful, the SEGA license screen would appear on the screen. Once the license screen code was run, two bootstrap sequences would initiate. Bootstrap 1 would load and set up some hardware registers. Bootstrap 2 oversaw the CPU stack set up and loaded the Vector Base Counter (VBR for short).

As mentioned earlier, the name of the game executable was stored in the *IP.BIN* file. This file was searched by the bootROM chip in the disc's file system, and loaded in to memory at the memory address 0x8C010000. This file was known as *1ST_READ.BIN*.

During the development process, SEGA added a functionality to its console which would be used a backdoor to exploit the system, thus making the system able to run burned games. This functionality, known as *MIL-CD*, allowed the Dreamcast to run code from a CD-ROM as well as a GD-ROM. *MIL-CD* was used very sporadically during the system's lifecycle, and it was intended for multimedia applications, such as karaoke. SEGA was aware that this feature might have been a potential backdoor, so the company decided to add a scram-

bler in to the bootROM chip to randomly scramble *1ST_READ.BIN*. The scrambler seemingly allowed the Dreamcast to recognize whether a pirated copied was loaded in to the disc drive.
In 1999, the hacker group known in the scene as *Utopia* stole a Katana SDK (*Katana* being the codename of the Dreamcast during its development). Being an SDK, it featured a reverse-scrambler to allow *MIL-CD* application code to be run, thus exploiting the system.

2.8.3 Sony’s PlayStation 3: Install Other OS, Fail0verflow and PSJailbreak

Sony entered the home console market in 1994, with the *PlayStation*. Both the *PlayStation* and its successor, the *PlayStation 2*, were no stranger to system exploits and piracy. With its third home console, the *PlayStation 3*, Sony was confident that the system was safe and had no exploits, deeming it “unhackable”.
The *PlayStation 3* was launched in November 2006 at a retail price of 499$US. The high price was due to the use of a Blu-Ray disc drive, which was a new and expensive technology at the time. Not only that, but the use of a completely in-house built processing unit made the price float since it was an expensive semiconductor to manufacture.
The main processing unit of the *PlayStation 3* is called *CELL Broadband Engine*, and it was developed in conjunction with IBM.
The architecture of the *CELL* processor is much different than conventional processing units: it uses nine cores, a PowerPC processing element (PPE), and eight synergistic processing elements (SPEs). The PPE was used as the general processor, while the eight SPEs were used to perform high-speed calculation. This architecture was hard to develop for, as developers needed to specify to the processor which datasets the game needed to be executed. Despite being hard to develop for gaming purposes, the processor was used successfully in an array of cases and applications, including molecular folding simulation, and human brain modelling.
The *PlayStation 3* shipped with a feature that allowed end users to install a Linux distribution to the system. It was known as *Install Other OS*. The main distribution used for the system was known as *Yellow Dog Linux*. It allowed users to use their consoles as a home computer.
Hackers were turning to *Other OS* to gain access to the hypervisor, and in 2010 Sony decided to remove the feature in the 3.21 software update. Later that year, a tool known as *PSJailbreak* was released. It was a USB device that tricked the *PlayStation 3* system into thinking that a 6-port USB hub was plugged in. Port 1 stored the payload which was pushed into memory, and in Port 4 the tool made the system think that a Service Jig was plugged in. A Service Jig is a USB jig used in Sony’s service centers to force a *PlayStation 3* system into Factory/Reset Mode. Then, the *PSJailbreak* tool would execute

a heap overflow and the payload stored in Port 1 was executed, allowing the system to run unsigned code.
Again, later in the same year, a hacker group known as *Fail0verflow* announced that the *PlayStation3*'s private key was discovered. The flaw was in the key generation, which was supposed to be random. It turns out that the hypervisor random key generation was not random. With this information in hand, the group worked from the key, cracking the system.

2.9 Digital Rights Management in PC Gaming in the present day

PC gaming has always been hit by piracy harder than the console market. Availability and the structure of the market in the past made copyright infringing practices an easier task to achieve. PC games have always been weaker in terms of anti-piracy and rights management protection. Now, with modern anti-piracy and anti-tamper techniques being built in the digital distribution structure itself, PC gaming has received both praises regarding security and backlashes regarding privacy and how some of these anti-piracy measures tend to be more than just that.
In this chapter, the biggest digital storefronts and distributors will be discussed.

2.9.1 Steam by Valve (2004)

The PC gaming has not been the same since Valve released *Steam* in 2004. At its origin, *Steam* was an updater for Valve's releases at the time including *Half-Life 2* and *Counter-Strike v1.6*. A year later *Steam* started to support third-party titles. Since then, it became the biggest online game storefront. On top of that, Steam acts as a DRM.
The way *Steam* works is by using a licensing system: when a user buys a game on the platform, it acquires the license to download the game from its servers. It must be said that Valve obliges developers to use *Steam DRM Wrapper* to publish their products on the storefront. Valve itself advises on its documentation to use a third-party DRM in order to increase the chances of avoiding tampering.

2.9.2 Origin by Electronic Arts (2011)

Origin is a digital storefront developed, published, and managed by Electronic Arts in 2011. It is the only way to access games by Electronic Arts on PC. It does work similarly to *Steam* by using a licensing system. Its business model also include a subscription service, *Origin Access*.

2.9.3 Bypassing EA’s security servers: ZLOEmu

ZLOEmu is a software used to emulate *Origin*’s authentication servers, to allow users to freely download content off *Origin*’s servers. It uses server emulation to bypass Electronic Arts’ security protocols. It is mostly used in countries such as Russia, where the project originates from.

2.9.4 Uplay by Ubisoft (2012)

Uplay is Ubisoft’s own digital storefront and game library manager. It was released in 2012. Most of the Ubisoft catalog is available via *Uplay* or through other storefronts such as *Steam*. In order to play a Ubisoft game via *Steam*, the *Uplay* client must be installed on the user’s machine.

2.9.5 Epic Games Launcher by Epic Games (2019)

Epic Games, after the success of their free *battle royale* game, *Fortnite*, decided to re-shape the launcher used by the game into a digital storefront. Known as the *Epic Store*, the store features game library management, and a storefront. In terms of DRM, Epic stated that game developers can run whichever DRM technology they prefer.

2.9.6 The DRM-free approach: GOG by CD Projekt RED (2008)

Polish development game company CD Projekt Red has always been vocal regarding digital rights management matters. In 2008 the studio opened *GoodOldGames.com*, known today as *GOG*, an online retailer which sold older game stripped of DRM technology. When a user buys a game from *GOG*, it will receive an executable file, which is used to install the game, and the game is not tied to anything. The biggest benefit is that if one day the platform got shut down, the product would be still able to run, due to not having any ties to authentication servers.

2.9.7 Microsoft Store by Microsoft (2012)

Microsoft Store is a digital storefront owned by Microsoft and launched alongside *Windows 8* in 2012. Its working methods differ compared to the other storefronts, as Microsoft requires a tight application packaging in order to avoid any sort of tampering. Most applications require to be written using Microsoft’s own *Universal Windows Platform*, an Application Programming Interface (or API) used to allow application to be run seamlessly on different platforms without the need for code recompilation or rewriting (See Section 2.10.2)

2.10 The DRM platforms of the modern age: anti-tamper software

Storefronts and digital distribution platforms do offer some anti-tampering protection. It must be said that developers are still looking for more protection. That is when third-party anti-tamper platforms come into play.

2.10.1 Denuvo by Irdeto (2014)

Denuvo is an anti-tamper software developed by Irdeto and it was released initially in 2014. Irdeto bought Sony DADC in 2013, the same branch of Sony that developed Secu-ROM (see Section 3.3.1). *Denuvo* was quietly shipped with Electronic Arts' game *FIFA 15*. *Denuvo* has been considered "a DRM for DRM". Functionality and patent analysis will be discussed in the second part of this dissertation.

2.10.2 Universal Windows Platform by Microsoft (2016)

Universal Windows Platform (or *UWP)* is an application programming interface (or API) developed by Microsoft and released in 2016. The main selling point of *Universal Windows Platform* is the ability to write code only once and be able to use said code across Microsoft platforms, including *Windows 10, Windows 10 Mobile,* Microsoft's home game console, the *Xbox One*, and Microsoft's mixed reality visor, *Hololens*. It must be said that *UWP* was not thought of as a DRM, however many of its functionalities coincide with anti-tamper software, case in point the constant encryption stream between machine and hard drive while a *UWP* application is running. *UWP*-based applications need to be installed and run through Microsoft's own storefront, as they are heavily encrypted.

3 Research

This research used a quantitative research method, more specifically a correlational research.

3.1 Quantitative research

Quantitative research is the main type of research. It is used to generate numerical data and generate statistical tools, such as graphs.
There are four different types of quantitative research: descriptive, correlational, casual/comparative, also known as quasi-experimental, and experimental.
Descriptive research uses data to create a hypothesis. This research method is used in projects where the phenomenon studied used systematic information.
Correlational research uses data to analyze patterns without digging deeper into the causes. The data is also gathered in its natural state, without working on the manipulation of variables.
Causal/comparative, also known as quasi-experimental research, is the method use to study causes and effects between variables. This method of research is used to understand how the causes of a phenomenon create the effects the research is studying.
Experimental research is the method known as true experimentation. This type of research uses the scientific method to study and connect causes and effects between a pool of variables.

3.2 Correlational research

Correlational research is the research method I have chosen to gather and compare data. The hypothesis which are at the base of this dissertation indicate that there is a pattern going on, in this case, performance hindrance in video games using anti-tamper software. Data is compared between two specimens of the same product. One of the specimen uses anti-tamper software, while the second specimen does not. The empirical variables gathered are raw performance, loading times, executable file size and runtime profiling (See Section 4).

3.3 Benchmarking

To acquire data that can be considered reliable, the games need to be benchmarked. Benchmarking is used to gather performance data to make the game easier to set up. There are games that have built-in benchmark tools that can assess performance and set the game accordingly.

Benchmarks have been used for years to gather empirical data on terms of performance. Benchmarks programs are developed with seven key criteria in mind (Dai, Berleant; 2019):

1. Relevance
2. Representativeness
3. Equity
4. Repeatability
5. Cost-effectiveness
6. Scalability
7. Transparency

All these criteria make sure that benchmark tools are fair across a range of different systems, as well as ascertain that the tests can be carried multiple times. The results provided by benchmarking programs need to be readily available and easy to comprehend. In terms of types of benchmarking programs, seven types of applications can be identified (Wikipedia, 2020):

1. Real program: a real-world application can be used as a benchmarking tool
2. Component benchmark: a program that tests a specific component or features of said component (e.g. reading and writing speeds on storage devices)
3. Kernel: low-level programs that measure “the basic architectural features of parallel machines” (Netlib, 1996)
4. Synthetic Benchmark: a program that test components using synthetic loads based on statistical calculations
5. I/O (input/Output) Benchmark: program that tests input-output interfaces in a system
6. Database Benchmark: program that measures database specific parameters such as response times and throughput values
7. Parallel Benchmark: program used to test multiple serialized machines or machines that run multiple cores or processing units

In this specific research the type of benchmark tool used is a combination of two real programs: the video game, which is in this research the program which performance is tested, and an acquiring data solution utility known as *Afterburner*, developed by MSI. Benchmarking is considered challenging, as many variables can change between a first run and a second run. Some of these challenges include: manufacturers cheating in benchmarks, disclosing higher values than actually achieved or the way in which benchmarking tools do not take into consideration factors such as heat and cooling.

In this case none of the games had built-in benchmarking tools.
In order to acquire data such as frame rate, I used an external tool called *MSI Afterburner*. *Afterburner* is a utility that allows users to monitor performance data, GPU overclocking, as well as being capable of logging data to simplify acquisition. *Afterburner* can monitor

tens of different data parameters, and help users controlling said parameters by using in-game overlays. These in-game overlays help users identify issues and troubleshoot them.

3.4 Benchmarking methodology

The benchmarking methodology used in this research is split into four stages. All four stages compare metrics regarding the performance of a game. The first stage measures raw performance by measuring the average frames per second, 1% frame per second lows and 0.1% frame per second lows. The second stage measures loading times in seconds. The third stage measures executable file size in megabytes. The fourth stage does not use empirical metrics, but functions to discover the inner workings of the DRM call stack system.

3.4.1 Stage one

Stage one is the stage in which raw performance gets measured. In stage one each copy of each game runs for 25 minutes. The same level and environment is chosen to make comparison fair. The first copy which gets tested is the one using Denuvo anti-tamper. At the end of the testing time, the machine gets switched off for five minutes to cool down the internal components. After five minutes the machine is switched on and the DRM-free copy of the game is tested. When the test is completed, the data gets acquired. The process has been the same across the pool of products I chose.

3.4.2 Stage two

Stage two concerns loading times. Both copies of the game are tasked to load a save game file. A save game file is a heavy loading process as the assets, routines and security calls get loaded in-memory. To achieve a correct result I used the machine’s internal stopwatch. The stopwatch starts counting when the loading request starts and stops when the loading has been completed. The results between the two specimens then get compared.

3.4.3 Stage three

Stage three consists in checking the executable file size of both game copies. To check the file size, the Windows *Properties* tab has been used. The tab can be accessed by right-clicking on the file and selecting the *Properties* option. The executable files sizes then get compared.

3.4.4 Stage four

Stage four consists in runtime profiling. An external program is used to monitor memory function call stacks. The program used in this research is *VTune Profiler* by Intel. The purpose of this stage is to control which function stacks are used by Denuvo anti-tamper to communicate with the game's source code. In this phase, the two copies of the game run for the same amount of time as in Stage one (25 minutes). After 25 minutes have passed, the data log from the profiler gets analyzed to check the call stack and see where Denuvo gets called in.

4 Data gathering and results

The main focus of this dissertation is to find out if anti-tamper software impact performance. To achieve an empirical result, the research will be conducted by using six computer games. The criteria used in the selection process revolves around availability: each game needs to have a version running Denuvo anti-tamper, and a DRM-free version readily and commercially available.
The games chosen for this research were:

- *Lords of the Fallen;*
- *ABZÛ;*
- *Mad Max;*
- *Sherlock Holmes: The Devil's Daugther;*
- *Yesterday Origin;*
- *Deus Ex: Mankind Divided;*

The research process in four stages, by checking different parameters:

- Stage 1 focuses on raw performance: the game will be ideally run using its in-game benchmarking tool. If this option is not available, the game will be run for 25 minutes using graphical settings suitable for the test machine. In this test the parameters which will be monitored are average frame rate, 1% frame late lows and 0.1% frame rate lows.
- Stage 2 focuses on loading times: in this stage the time between interacting with the game menu and the moment when the game is ready to play will be monitored.
- Stage 3 focuses on file size: both executable files will be compared in terms of size to ratify the claim that games running Denuvo are larger in size than their respective DRM-free counterparts.
- Stage 4 requires the use of a profiler to check the executable function stack and understand what functions in the stack trigger DRM functions and measures.

The test machine used for these tests uses the following internal components:

- CPU: Intel Core i5-7300 HQ (4 cores, 8 threads) @ 2.5 GHz with Intel Turbo Boost @ 3.2GHz;

- RAM: 8GB LPDDR4 @ 2400MHz;
- GPU: NVIDIA GeForce GTX 1050 4GB GDDR5 VRAM;
- HDD: 1TB @ 7200RPM;
- SSD: 128GB;

The most important components while running games are the CPU, as well as the RAM and VRAM usage: the higher the usage, the more likely that the game will be stuttering, freezing, or behaving unexpectedly.

4.1 Denuvo's history

Denuvo is a digital rights management and anti-tamper software created by Denuvo Technologies GmbH, a company risen from the acquisition of Sony DADC, the same branch of Sony that developed and worked on the SecuROM anti-tamper and DRM technology.
The first implementation of Denuvo was shipped with the Electronic Arts game *FIFA 15*, in September of 2014, as the cracking community was surprised not to see a crack getting released in the first 30-day sales window of the game.
The first somewhat successful attempt at cracking Denuvo came from the Chinese group 3DM, which released a functioning crack of the Electronic Arts game *Dragon Age: Inquisition* in December 2014, making in the first game using Denuvo anti-tamper to be cracked. However, the *3DM* release of *Dragon Age: Inquisition* was unstable, filled with bugs and prone to crashing.
During that time Italian scene group *Conspiracy* was able to get the inner functions of Denuvo sorted out and started releasing cracks upon cracks, in order to catch up with the backlog, as game studios started implementing Denuvo in their games.
In 2016 a Bulgarian cracker by the name of *Voksi* was able to bypass authentication for a few games by discovering a loophole that Denuvo used in the demo of Bethesda's game *DOOM*.
After that discovery, Denuvo started to work on a new version of their anti-tamper software. Still, crack groups were keeping up the pace and the time between the release of a game using Denuvo and its subsequent cracked release kept shortening, reaching the 30-day sales window, and in some cases, reaching the day-zero date, which means that the game's protection got bypassed on the same days of release.

4.2 Denuvo's patent filing

In August of 2015, an application patent has been filed by Denuvo Technologies GmbH which describes the process and application of a technique for enabling the nominal flow of an executable file. This patent describes a technique in which nominal constants are used on calculations to obfuscate and de-obfuscate parts of an executable by using an algorithm that uses said nominal constants: in the patent, the example shows the use of the nominal constant known as *Pi* (3.1415), which is required to calculate the volume of a cylinder. This nominal constant gets retrieved by using a series of calculations that comprise of hashed values of hardware information retrieved by the client machine. In other words, a system of a client, server, and a protection engine is used to retrieve hardware information and encrypt them using hash values that, if calculated randomly, can generate an exact nominal constant, which is used to enable nominal flow, thus, validating the executable file to be run. This process can be triggered during runtime, or while the client is retrieving information from the server for authentication purposes. The patent also mentions the use of this technique within a DRM environment, as a way to protect executables from being unauthorizedly run.

4.3 In-game triggers

Another method that Dneuvo uses for copy protection is in-game triggers.
In 2017 scene group *Codex* released a cracked version of the fighting game *Injustice 2: Legendary Edition*. In this release the group disclosed how in-game triggers are used by Denuvo for copy protection.
In the example stated by the group, when the character *Robin* performs an in-game action as “throwing a smoke bomb to the ground”, Denuvo starts a process of writing a private key to the memory from the memory address *000000014C113692*, generating this memory call:

```
000000014C113692  | 44 88 07                | mov byte ptr ds:[rdi],r8b
    000000014C113695  | 5F                      | pop rdi
    000000014C113696  | 50                      | push rax
    000000014C113697  | 21 C0                   | and eax,eax
    000000014C113699  | 9C                      | pushfq
    000000014C11369A  | 44 01 C1                | add ecx,r8d
    000000014C11369D  | 4C 89 F0                | mov rax,r14
    000000014C1136A0  | 48 89 C1                | mov rcx,rax
    000000014C1136A3  | 48 C7 C0 00 00 00 00    | mov rax,0
    000000014C1136AA  | 48 09 D0                | or rax,rdx
    000000014C1136AD  | 48 83 C1 01             | add rcx,1
    000000014C1136B1  | 49 89 CE                | mov r14,rcx
    000000014C1136B4  | C1 C1 08                | rol ecx,8
    000000014C1136B7  | 9D                      | popfq
    000000014C1136B8  | 58                      | pop rax
```

3. Memory call generated by the in-game trigger: to write the private key, variables get pushed from registry r8b, added to the values in registry r8d and moved to registry r14 to await execution.

After this memory call ends, Denuvo fills a buffer at the address *000000014779F593.* After the buffer has been filled and the private key from the Denuvo server has been retrieved, anti-tamper functions are executed from the memory address *000000014774C37E*, generating this memory response:

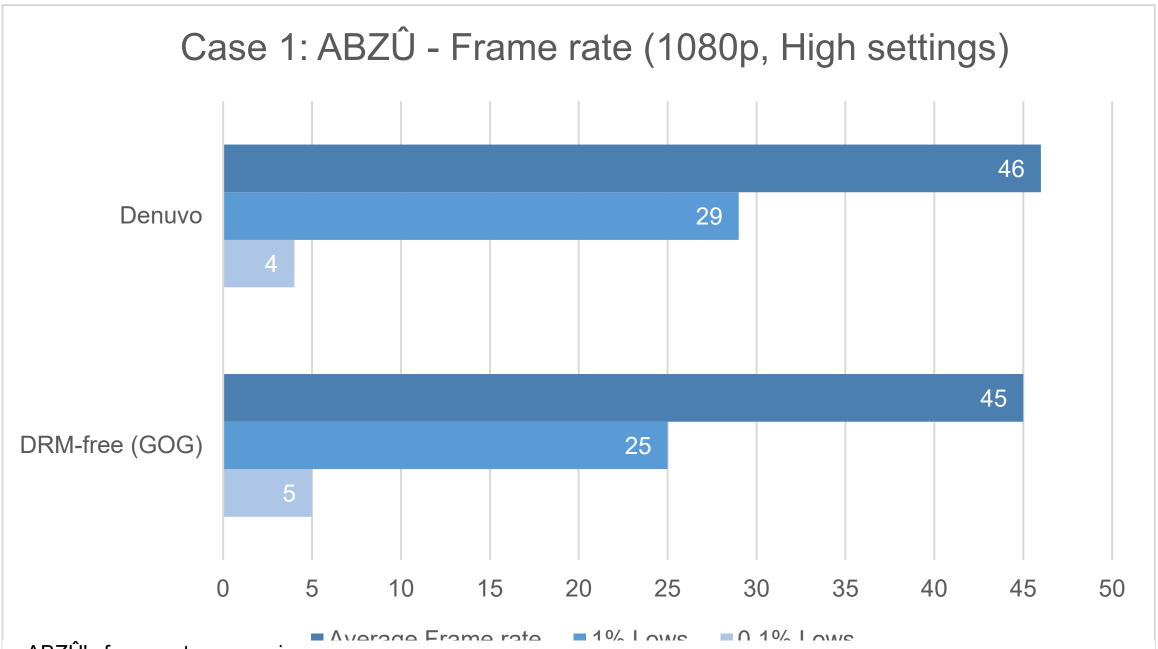
```
000000014774C37E  | 41 89 7D 00             | mov dword ptr ds:[r13],edi
      000000014774C382  | 48 29 F3                | sub rbx,rsi
      000000014774C385  | 41 54                   | push r12
      000000014774C387  | C1 CB 0D                | ror ebx,D
      000000014774C38A  | BE D4 72 4D 3E          | mov esi,3E4D72D4
      000000014774C38F  | 4C 8D 25 4F B5 06 FE    | lea r12,qword ptr ds:[1457B78E5]
      000000014774C396  | 4C 33 24 24             | xor r12,qword ptr ss:[rsp]
      000000014774C39A  | 48 8B 1C 24             | mov rbx,qword ptr ss:[rsp]
      000000014774C39E  | 4C 21 E3                | and rbx,r12
      000000014774C3A1  | 4C 09 24 24             | or qword ptr ss:[rsp],r12
      000000014774C3A5  | 0F BA F8 06             | btc eax,6
      000000014774C3A9  | 0F BA F6 0D             | btr esi,D
      000000014774C3AD  | 48 29 1C 24             | sub qword ptr ss:[rsp],rbx
      000000014774C3B1  | 4C 89 E3                | mov rbx,r12
      000000014774C3B4  | 48 23 1C 24             | and rbx,qword ptr ss:[rsp]
      000000014774C3B8  | 4C 0B 24 24             | or r12,qword ptr ss:[rsp]
      000000014774C3BC  | 49 29 DC                | sub r12,rbx
      000000014774C3BF  | C3                      | ret
```

4. Memory response generated by Denuvo: after the private key gets retrieved from the Denuvo servers, the private key calculated in the previous memory call gets pulled from the registry r12 and matched against the Denuvo server private key.

This response is used by Denuvo to overwrite any patches previously applied to these addresses by using functions extrapolated from a register called by Denuvo's code; in this case, the register is known as *r13*.
What *Codex* disclosed in this description is that their crack reads a large amount of encrypted code and patches the calls, therefore circumventing Denuvo's obfuscation system. The group also stated that this specific process causes micro-lag and stutters, as the process is highly dependent on the CPU, thus raising the CPU usage, in some cases over 99%, which is responsible for stutters.

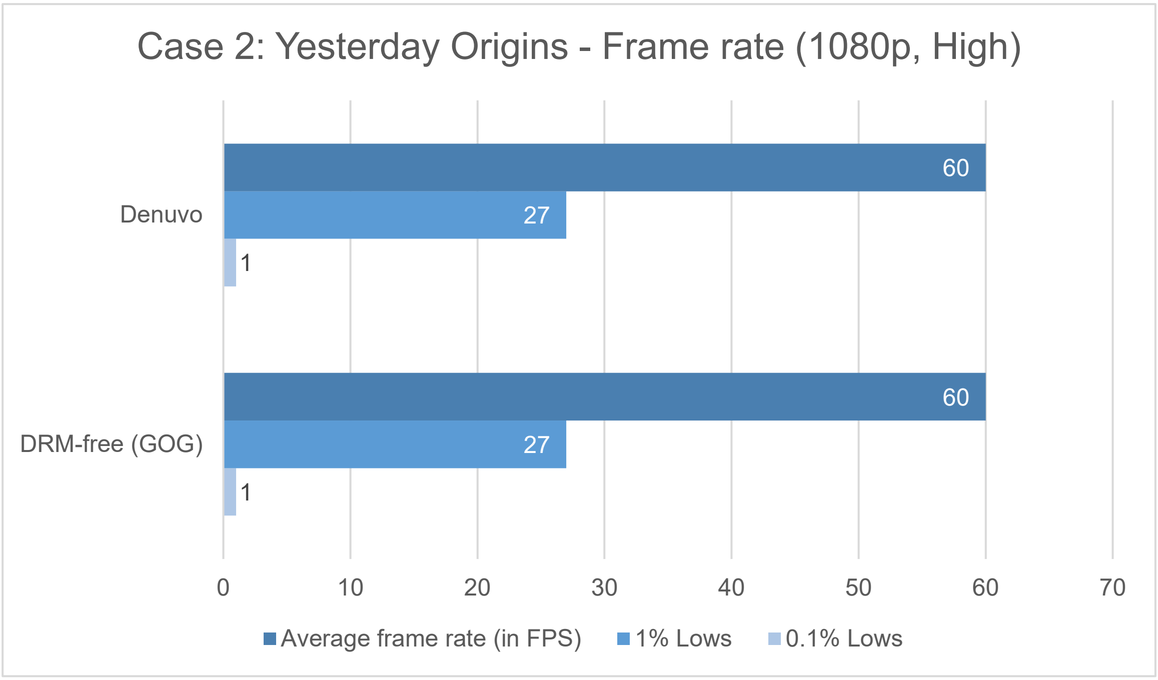
4.4 Stage 1: raw performance

In the first stage of the research, both games will be measured on raw performance by playing the same levels, in order to make it as scientifically accurate as possible. The values used to measure raw performance are frame rate, 1% frame rate lows, 0.1% frame rate lows and frame time. These values are interconnected and depict a picture of the general performance as well as how the game feels: the higher the disparity between the average frame rate and the 1% lows and 0.1% lows subsets, the choppier and more stutter the game feels. When it comes to frame time analysis, the lower the number, the smoother the game runs.
Denuvo anti-tamper has been claimed to hinder general performance by spiking CPU usage causing micro freezes and stutters, which can make the difference in the feel of the game.
All games have been tested for a maximum time of 25 minutes. In every version, CPU usage has been oscillating between 95% and 99%, as this machine's processing unit at the



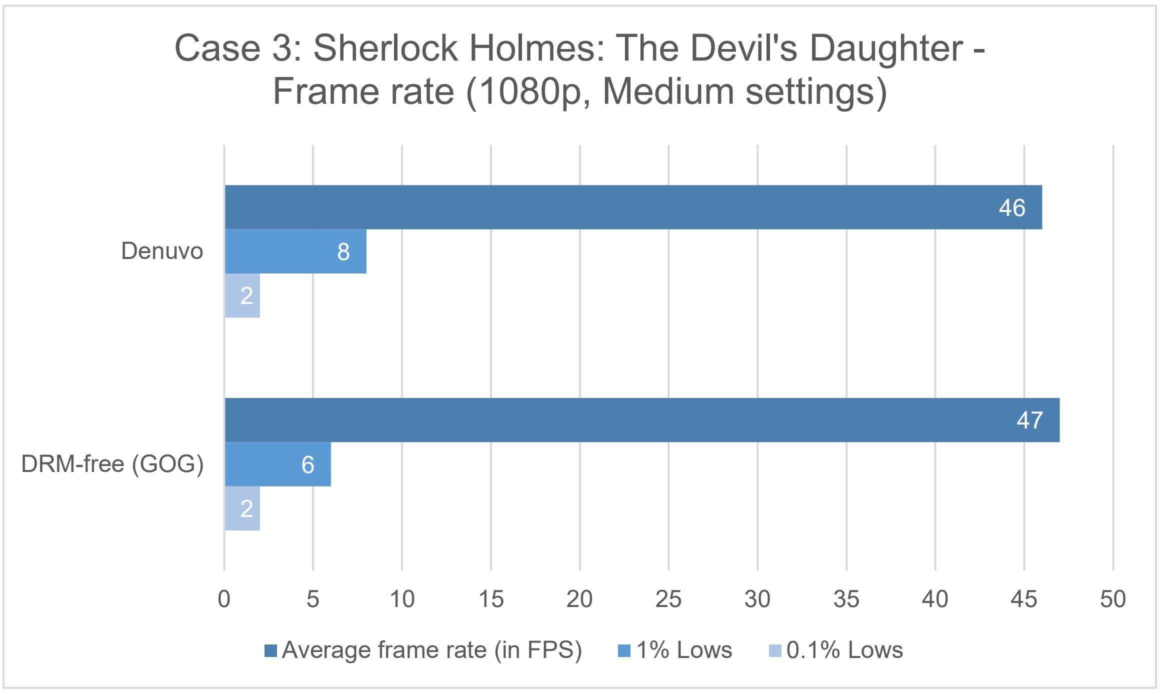
a. ABZÛ's frame rate comparison

time of writing is considered a “mid-range” option. Hence the CPU usage data can not be considered a reliable source of data as every computer has different processing units that handle processes differently or more efficiently.
In this first case, *ABZÛ*’s version running *Denuvo* anti-tamper ran at a slightly higher average frame rate, 46 frames per second, compared to its DRM-free version which ran at an average frame rate of 45 frames per second. Both versions had similar figures on the 1% and 0.1% lows, 29 and 4 frames per second on the *Denuvo* version, and 25 and 5 frames per second on the DRM-free version. These figures show that the DRM-free version runs at a more stable frame rate throughout the playthrough with fewer stutters, despite the average frame time on the *Denuvo* version being slightly lower, due to the higher frame rate. In terms of frame times, the *Denuvo* version registered an average of 21.7 milliseconds, while the DRM-free version registered an average of 22.2 milliseconds. This concludes that the difference in performance between the two versions is minimal, at 2.23 percent.



b. Yesterday Origins’ frame rate comparison

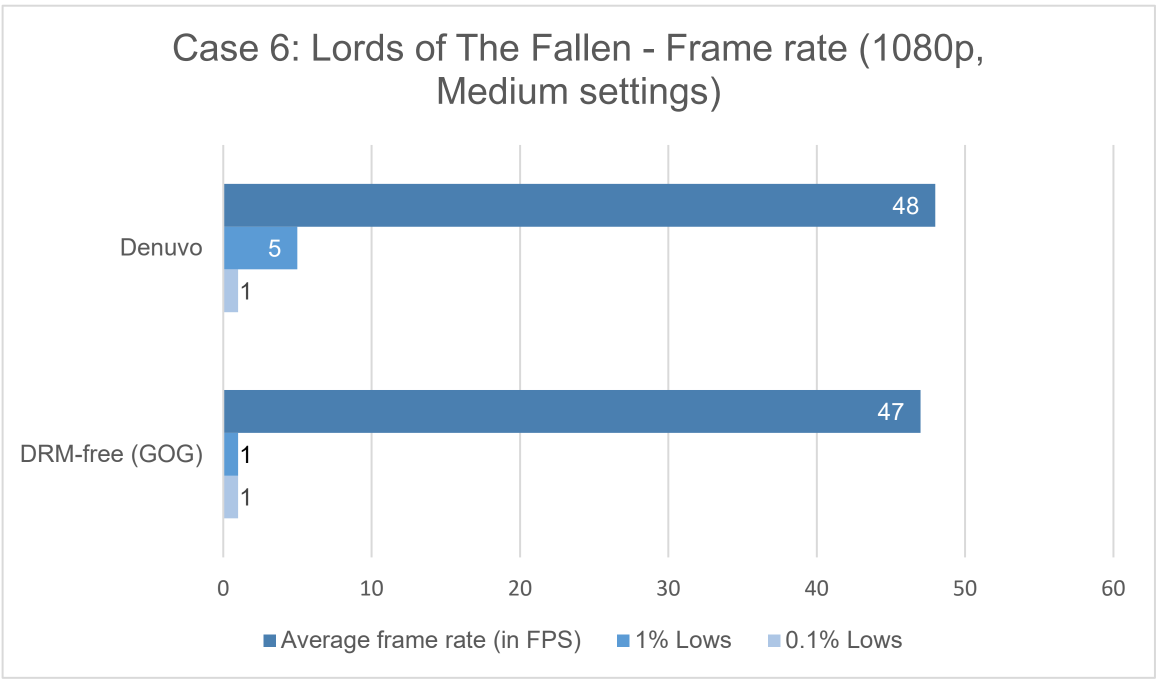
The second game in the analysis is *Yesterday Origins,* and the raw performance test yielded an unexpected result: both versions ran exactly at the same average frame rates, including 1% and 0.1% lows at 60, 27 and 1 frames per second, respectively. As the average frame rate is the same, the frame time analysis is equal in both versions clocking in at 16.7 milliseconds. This result is unexpected as it is rare and highly unlikely that two versions of the same game, with different anti-tamper technologies, run in the exact same way and yield the same results.



c. Sherlock Holmes: The Devil's Daughter’s frame rate comparison

Sherlock Holmes: The Devil’s Daughter is the third game taken into analysis. The graph shows that the version using *Denuvo* anti-tamper reported an average frame rate of 46 frames per second, with 1 percent lows at 8 frames per second, and 0.1 percent lows at 2 frames per second. In this case, the DRM-free version yielded a higher average frame rate at 47 frames per second, with 1 percent lows at 6 frames per second and 0.1 percent lows at 2 frames per second. Here, the disparity in performance between the two copies is on average 2.17 percent. When it comes to frame time, the version running *Denuvo* anti-tamper ran at an average frame time of 21.7 milliseconds, while the DRM-free version reported a slightly lower time of 21.3 milliseconds.

The fourth and fifth cases would have been *Mad Max* and *Deus Ex: Mankind Divided*. However, both versions of both games kept crashing at boot, meaning that the results at this stage could not be obtained.



d. Lords of The Fallen’s frame rate comparison

The last game taken into consideration in this stage is *Lords of The Fallen.* The data show that during runtime, the version using *Denuvo* anti-tamper achieved an average frame rate of 48 frames per second, with a 1 percent low average of 5 frames per second, and a 0.1 low percent average of 1 frame per second. In the second test run using the DRM-free version of the game, the results worsened slightly, yielding an average frame rate of 47 frames per second, with a 1 percent low and 0.1 percent average of 1 frame per second for both result subsets. Average frame times show that the *Denuvo* version of the game fared slightly better, at 20.8 milliseconds, while the DRM-free version ran at an average frame time of 21.3 milliseconds. The performance disparity between these two versions is 2.12 percent.

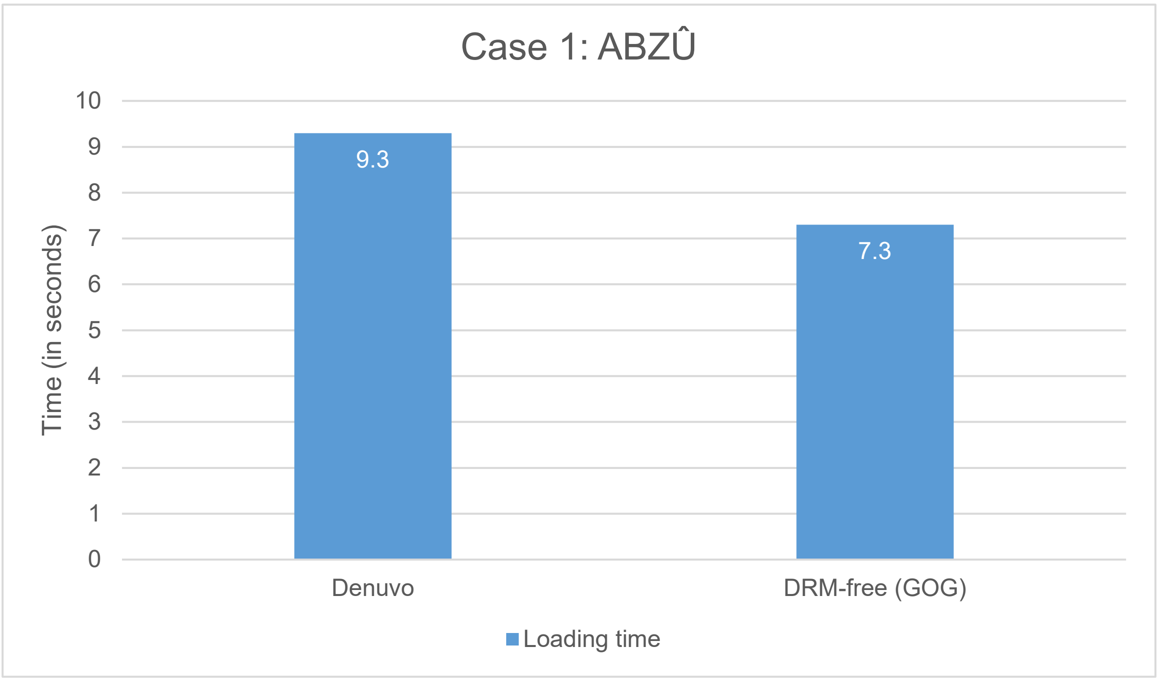
The conclusion of this stage shows that, using this specific combination of hardware components, games using *Denuvo* anti-tamper did perform minimally better than their DRM-free counterparts. The difference in performance, though, is insignificant, with a combined average of 1.63 percentile points.

4.5 Stage two: loading times

In the second stage of this research, loading times were analyzed.

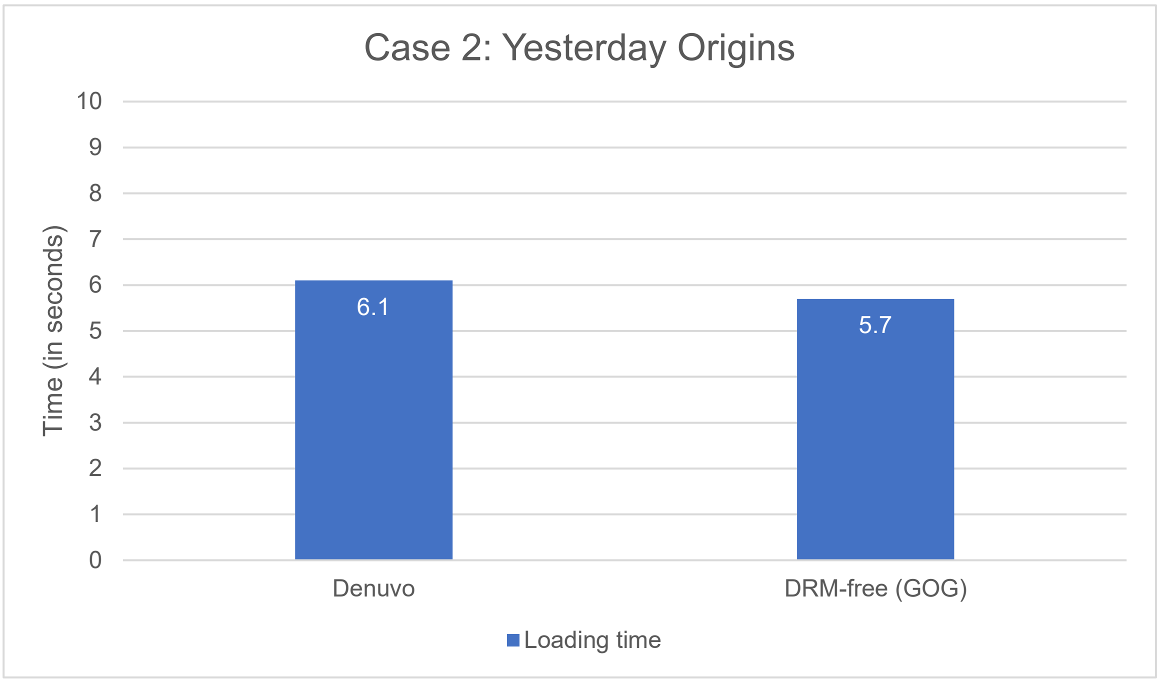
There have been claims that games using *Denuvo* as an anti-tamper solution tend to have higher loading times due to the higher number of function calls that *Denuvo* makes to check the integrity of the executable and if the executable has been tampered with.

The load time testing was conducted by entering the game’s menu and loading a save file, which is the longest loading time in a game as most of its assets and routines need to be loaded into memory. The time was monitored by an internal stopwatch, in order to avoid human error. The stopwatch starts monitoring when the last input from the game menu was registered and it stops when the loading ends.



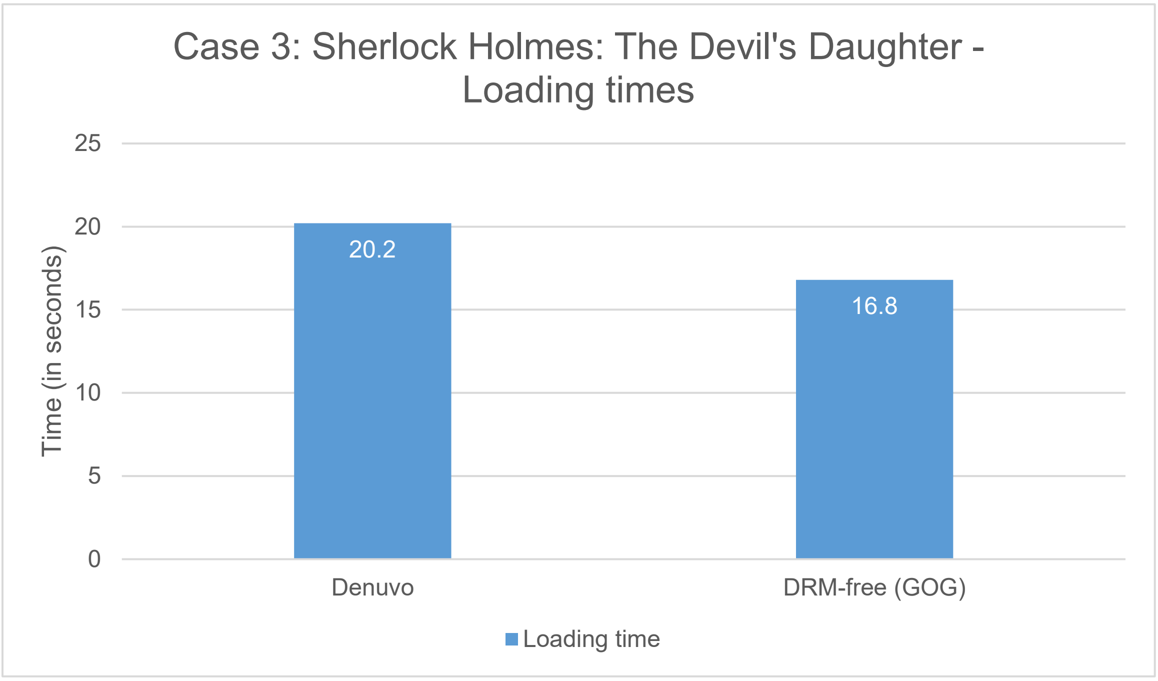
e. ABZÛ's loading times

ABZÛ's loading time analysis shows that the version running *Denuvo* anti-tamper loads the save file in 9.3 seconds, while the DRM-free version loads in 7.3 seconds. This comparison states that the DRM-free version of *ABZÛ* is capable of loading 21.5 per cent faster than its *Denuvo* counterpart.



f. Yesterday Origins' loading times

While performance between the two versions is the same, the loading times on *Yesterday Origins* are slightly different. The version running the *Denuvo* anti-tamper technology loads a save file in 6.1 seconds, while the DRM-free version achieves this task in 5.7 seconds. The loading time disparity between the two game versions is small, at 6.5 percentile points.

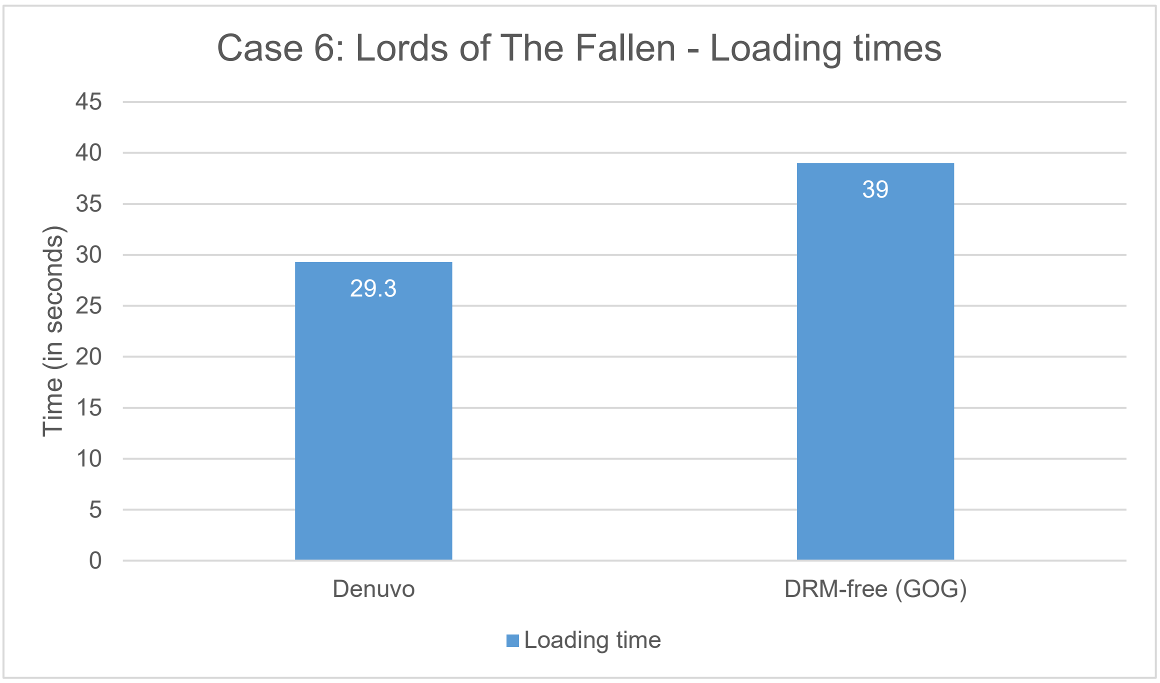


g. Sherlock Holmes: The Devil's Daughter's loading times

Sherlock Holmes: The Devil's Daughter's loading time chart shows a small but noticeable difference between the two versions. The loading time monitored in the *Denuvo* version of the game clocks in at 20.2 seconds, while the DRM-free version loads a save file in 16.8 seconds.

The comparison states that the DRM-free version of the games loads 16.8 per cent faster than the *Denuvo* version.

Mad Max and *Deus Ex: Mankind Divided*'s loading times have not been recorded as both versions of both games will not run (see Section 4.6).



h. Lords of The Fallen's loading times

Lords of The Fallen is the game that reverses the trend in terms of loading times, as the version running *Denuvo* loads faster than the DRM-free version. Loading a save game on the *Denuvo* version of *Lords of The Fallen* takes 29.3 seconds. The DRM-free version of the game loaded a save file in 39 seconds. In comparison, there is a 33.1 per cent increase in loading times between the game version running *Denuvo* anti-tamper technology and the DRM-free version.

The data gathered in this stage of research shows that in most of the cases the DRM-free versions of the samples load faster than the specimen using *Denuvo*'s anti-tamper technology by 2.9 percentile points. This result can be significant, as fewer functions to call while loading tend to speed up the process.

4.6 Stage three: executable size file comparison

The third stage of research evaluated the size of the executable files which the games use to start and run.

Rumors have been circulating that a game using *Denuvo* anti-tamper technology has a larger sized executable file than the same game without said technology.

The research was conducted by looking into the game directory files, locating the executable, and opening the executable using the *Properties* option in Windows. The file size will be revealed and matched with its DRM-free counterpart to check if this claim is true.

Table 1. ABZÛ's executable file size comparison

ABZÛ	Denuvo	DRM-free (GOG)
	96.2 MB	43.2 MB

The first case under scrutiny was *ABZÛ.* The executable file that is used during runtime is located in the *Binaries* folder within the game's main directory. The data shows that the executable shipped with *Denuvo* is 96.2 megabytes in size, while its DRM-free counterpart comes in at 43.2 megabytes. The decrease in size achieved by not including *Denuvo* anti-tamper measures in at 55.1 per cent.

Table 2. Yesterday Origins' executable file size comparison

Yesterday Origins	Denuvo	DRM-free (GOG)
	21.1 MB	21.7 MB

The second game under scrutiny is *Yesterday Origins*. This case has been the most surprising, as in the previous stages both versions yielded similar results. This stage is no exception. The executable file using *Denuvo* anti-tamper measures at 21.1 megabytes in size, while the DRM-free version is 21.7 megabytes. The size increase between the two versions is minimal, at 2.84 percentile points. This metric and case can be either considered an exception or it can be discarded. The increase in file size can be caused by further patching on the DRM-free specimen.

Table 3. Sherlock Holmes: The Devil's Daughter executable file size comparison

Sherlock Holmes: The Devil's Daughter	Denuvo	DRM-free (GOG)
	54 MB	55.6 MB

The third game under scrutiny is *Sherlock Holmes: The Devil's Daughter*, and it yielded similar results to the second case. The executable file using *Denuvo* anti-tamper measures at 54 megabytes, while the DRM-free version clocks in at 55.6 megabytes. These data show a size increase of 2.96 percentile points between the executable using the *Denuvo* anti-tamper technology and its DRM-free counterpart. As mentioned for the second case, this metric can be considered an exception or discarded. However, in this specific instance, the increase in size can be attributed to either a new iteration of *Denuvo* being shipped in an update, or further patching on the DRM-free specimen.

Table 4. Mad Max's executable file size comparison

Mad Max	Denuvo	DRM-free (GOG)
	73.2 MB	24.5 MB

Mad Max could not be tested in the previous stages of research, as both versions would crash and not run, however the game executables were accessible. The version of the game using *Denuvo* measures in at 73.2 megabytes, while the DRM-free version from *Electronic Arts'* storefront, *Origin,* is 24.5 megabytes in size. These data show a size decrease of 66.5 percentile points between the two specimen.

Table 5. Deus Ex: Mankind Divided executable file size comparison

Deus Ex: Mankind Divided	Denuvo	DRM-free (GOG)
	95.6 MB	42.6 MB

Deus Ex: Mankind Divided was the second game which was not tested due to crash at boot, however; as with *Mad Max*, the game directories were still accessible, so an executable file size comparison could be drawn. Data shows that the version of the game using the *Denuvo* anti-tamper technology measures in at 95.6 megabytes in size, while the DRM-free version of the game measures in at 42.6 megabytes. This shows a size decrease of 55.4 percentile points.

Table 6. Lords of The Fallen's executable file size comparison

Lords of The Fallen	Denuvo	DRM-free (GOG)
	45.9 MB	10.9 MB

The sixth and last game under scrutiny is *Lords of The Fallen*. Data shows that the version of the game using *Denuvo* measures in at 45.9 megabytes, while the DRM-free version is 10.9 megabytes in size. The data shown concludes that the removal of the anti-tamper decreased the executable file size by 76.2 percentile point.

The third stage of this research shown that, across the pool of samples, four out of six products had a noticeable reduction in file size when the DRM was not included. The average decrease in size is 41.23 percentile points.

4.7 Stage four: runtime profiling

Stage four of the research would have been the stage in which both specimen of the application were analyzed using a profiler to examine function call stacks during runtime to identify the call stacks used to trigger *Denuvo* anti-tamper. However, this research stage had to be aborted, as the profiling of these executable files are using so-called “wrappers” to avoid any sort of tampering. Wrapping is used in conjunction with the storefront as an additional level of security. This process causes to obfuscate most of the call stack by not disclosing any information. Other than that, repeated attempts at running said executables with a profiler resulted in the program shutting down, thus not allowing the monitoring of further data during runtime.

5 Conclusion

This research was conducted to validate the main four claims of this dissertation. After analyzing the results of the different test stages, only one point can be firmly validated: the use of *Denuvo* anti-tamper does increase a game's executable file size, as the technology used for code obfuscation requires more space. As for the other points of this research, slight average improvements are shown across the pool of samples, however, said improvements are not statistically significant enough to confirm that *Denuvo* anti-tamper hinders both game performance and loading times.

To expand on this research, using a bigger pool of games, as well as different combinations of hardware components would be beneficial, as different users use different hardware.

Regarding the ethical implications of using digital rights management platforms and anti-tamper technologies, game developers and game publishers have the legal right to protect their intellectual property by any means necessary. The ethical question comes to play when a DRM implementation is not disclosed, and the technology used can be harmful for the end user's machine (see section 3.3.1).

This research can be considered partially conclusive.

The knowledge acquired during this research and the documentation gathering on this dissertation will be crucial in the future for helping understand the past and look further ahead by developing new methods of copy protection, as well as making end users aware of the importance of protecting intellectual and digital property.

References

Multimedia Security Technologies for Digital Rights Management [0-12-369476-0; 1-280-63591-6] Lin, Ching-Yung; 2006 Accessed April 4

Wikipedia – Piracy; URL https://en.wikipedia.org/wiki/Copyright_infringement; Accessed April 4

Wikipedia – BitTorrent; URL https://en.wikipedia.org/wiki/BitTorrent; Accessed April 4

Wikipedia – Central Processing Unit; URL https://en.wikipedia.org/wiki/Central_processing_unit; Accessed April 4

Wikipedia – Random Access Memory; URL https://en.wikipedia.org/wiki/Random-access_memory; Accessed April 4

Wikipedia – Graphic Processing Unit; URL https://en.wikipedia.org/wiki/Graphics_processing_unit; Accessed April 4

Scene - Software Piracy Exposed; Honick, Ron; 2005

Frame rate – Wikipedia; URL https://en.wikipedia.org/wiki/Frame_rate; Accessed April 4

Frame time – What is frame time? Why is frame time important?; Ropaku; URL https://www.ropaku.com/what-is-frame-time-why-is-frame-time-important/; Accessed April 4

System and Computers in Japan – Volume 19, Issue 5; The Concept of Software Service System; Mori, Ryoichi; Tashiro, Shuichi; 1988; Accessed April 7

Superdistribution: The Concept and the Architecture; Mori, Ryoichi; Kawahara, Masaji; IEICE Transactions (1976-1990) Vol. E73 No.7 pp.1133-1146; Accessed April 7

World Intellectual Property Organization Copyright Treaty Act, 1996, WIPO; Accessed April 7

The Digital Millenium Coypright Act of 1998 – U.S. Copyright Office Summary - www.copyright.gov/legislation/dmca.pdf; Accessed April 7

Open Mobile Alliance OMA DRM – Digital Rights Management Version 1.0 – Version 05-September-2002; http://www.openmobilealliance.org/release/DRM/V1_0-20021104-C/OMA-Download-DRM-V1_0-20020905-C.pdf Accessed April 7

Open Mobile Alliance OMA DRM Content Format – Approved Version 2.0 – 03 Mar 2006; www.openmobilealliance.org/release/DRM/V2_0-20060303-A/OMA-TS-DRM-DCF-V2_0-20060303-A.pdf; Accessed April 7

Open Mobile Alliance OMA DRM Requirements – Approved Version 2.1 – 14 Oct 2008; http://www.openmobilealliance.org/release/DRM/V2_1-20081106-A/OMA-RD-DRM-V2_1-20081014-A.pdf; Accessed April 7

LGR – History of DRM & Copy Protection in Computer Games; https://www.youtube.com/watch?v=HjEbpMgiL7U&t=771s; Accessed April 7

EuroGamer – Banging the DRM, The History of anti-piracy, page 2; https://www.eurogamer.net/articles/banging-the-drm-article?page=2; Accessed April 8

Modern Vintage Gamer – SecuROM – The PC CD-ROM DRM that broke games; https://www.youtube.com/watch?v=u8ltfyqD3lM&t=548s; Accessed April 8

Justia, Thomas v. Electronic Arts, Inc, CLASS ACTION COMPLAINT, September 22, 2008; https://dockets.justia.com/docket/california/candce/5:2008cv04421/207287; Accessed April 8

CD Media World – Macrovision SafeDisc Advanced Datasheet; https://www.cdmedia-world.com/hardware/cdrom/files/sdadv_datasheet.pdf; Accessed April 8

CVE Details – Macrovision SafeDisc – List of known vulnerabilities; https://www.cve-details.com/vulnerability-list/vendor_id-4176/product_id-7301/Macrovision-Safedisc.html Accessed April 8

SafeDisc – Google Patents; https://patents.google.com/patent/US7464411B2/en?assignee=macrovision&oq=macrovision; Accessed April 8

STARFORCE DRM Protection – CrackWatch; https://crackwatch.com/drm/starforce; Accessed April 8

StarForce Clients Portfolio – StarForce; http://www.star-force.com/about/clients/; Accessed April 9

The weird and wonderful CIC – HackMii; https://hackmii.com/2010/01/the-weird-and-wonderful-cic/ Accessed April 9

CIC Patent – Google Patents; https://patents.google.com/patent/US4799635A/en; Accessed April 9

How the Dreamcast copy protection was defeated – Fabien Sanglard; http://fabiensanglard.net/dreamcast_hacking/; Accessed April 9

Dreamcast Programming – IP.BIN and 1ST_READ.BIN; http://mc.pp.se/dc/ip.bin.html;

IBM 100 – The Cell Broadband Engine; https://www.ibm.com/ibm/history/ibm100/us/en/icons/cellengine/; Accessed April 9

Modern Vintage Gamer – The Sony Playstation 3 – The “Unhackable” Console; https://www.youtube.com/watch?v=siOXFGZj_z0&t=744s; Accessed April 9

ValveFollower – The History of Steam (PC); https://www.youtube.com/watch?v=9781Q63xIMo; Accessed April 9

Valve – Steam DRM (Steamworks Documentation); https://partner.steamgames.com/doc/features/drm; Accessed April 9

Games 4 The World – ZLOrigin – made simple; https://www.games4theworld.club/t30092-zlorigin-made-simple; Accessed April 9

Denuvo – Wikipedia; https://en.wikipedia.org/wiki/Denuvo; Accessed April 9

Overlord Gaming – History of Denuvo: the DRM for DRMs; https://www.youtube.com/watch?v=y_6zYVcJIKM&t=739s; Accessed April 13

Universal Windows Platfom – Wikipedia; https://en.wikipedia.org/wiki/Universal_Win-dows_Platform; Accessed April 13
Microsoft Documentation – What is a Universal Windows Platform (UWP) app? – Microsoft; https://docs.microsoft.com/en-us/windows/uwp/get-started/universal-application-platform-guide; Accessed April 13
Overlord Gaming – The History of Denuvo: the DRM for DRMs; URL https://www.youtube.com/watch?v=y_6zYVcJIKM&t=739s; Accessed June 6, 2020

Benchmark (computing) – Wikipedia; 2020; URL https://en.wikipedia.org/wiki/Bench-mark_(computing) Accessed August 18

Dai, Wei; Berleant, Daniel – Benchmarking Contemporary Deep Learning Hardware and Frameworks: a Survey of Qualitative Metrics; 2019; URL https://dberleant.github.io/pa-pers/BenchmarkingContemporaryDeepLearningHardwareAndFrameworks.pdf Accessed August 19

Embedded Microprocessor Benchmark Consortium – EEMBC Benchmarks; 2019; URL https://www.eembc.org/products/ Accessed August 19

PCWorld, IDG News – NVIDIA’s Benchmark Tactics Reassessed; 2003; URL https://web.archive.org/web/20110606032058/http://www.pcworld.com/arti-cle/111012/nvidias_benchmark_tactics_reassessed.html Accessed August 19

Netlib – Introduction to Kernel Benchmarks; 1996; URL http://www.netlib.org/park-bench/html/kernel-rationale.html Accessed August 19

ArsTechnica – Evidence continues to mount about how bad Denuvo is for PC gaming performance; 2018; URL https://arstechnica.com/gaming/2018/12/evidence-continues-to-mount-about-how-bad-denuvo-is-for-pc-gaming-performance/; Accessed June 6

XSReviews – Games with built-in benchmarks 2020: How to benchmark your PC; 2020; URL https://xsreviews.co.uk/news/games-with-built-in-benchmarks-2019-how-to-bench-mark-your-pc/ Accessed August 18

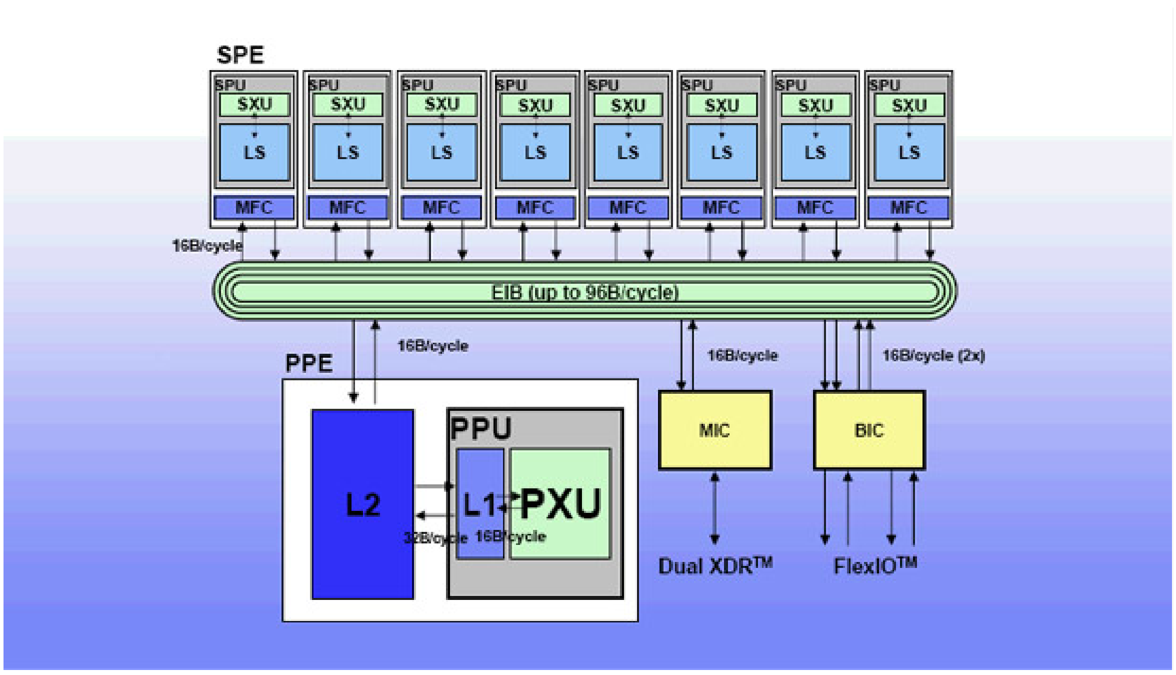
Afterburner _ MSI; 2020; URL https://www.msi.com/page/afterburner; Accessed August 18

ExtremeTech – Denuvo Really Does Cripple PC Gaming Performance; 2018; URL https://www.extremetech.com/gaming/282924-denuvo-really-does-cripple-pc-gaming-performance; Accessed June 7

Google Patents – Technique for enabling nominal flow of an executable file; 2014; URL https://patents.google.com/patent/EP2998895A1/en; Accessed June 5

Codex.nfo – CrackWatch – Injustice 2; 2017; URL https://crackwatch.com/game/injustice-2/cracks/injustice-2-legendary-edition-codex; Accessed June 6

Appendices



U.S. Patent Jan. 24, 1989 Sheet 2 of 10 4,799,635

FIG.2

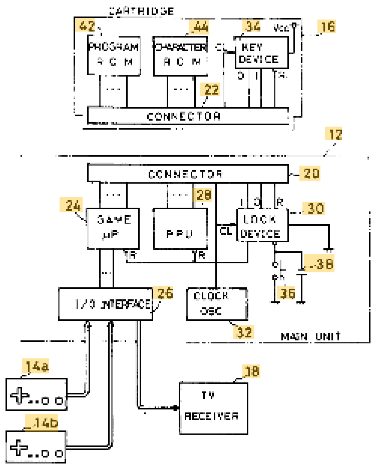
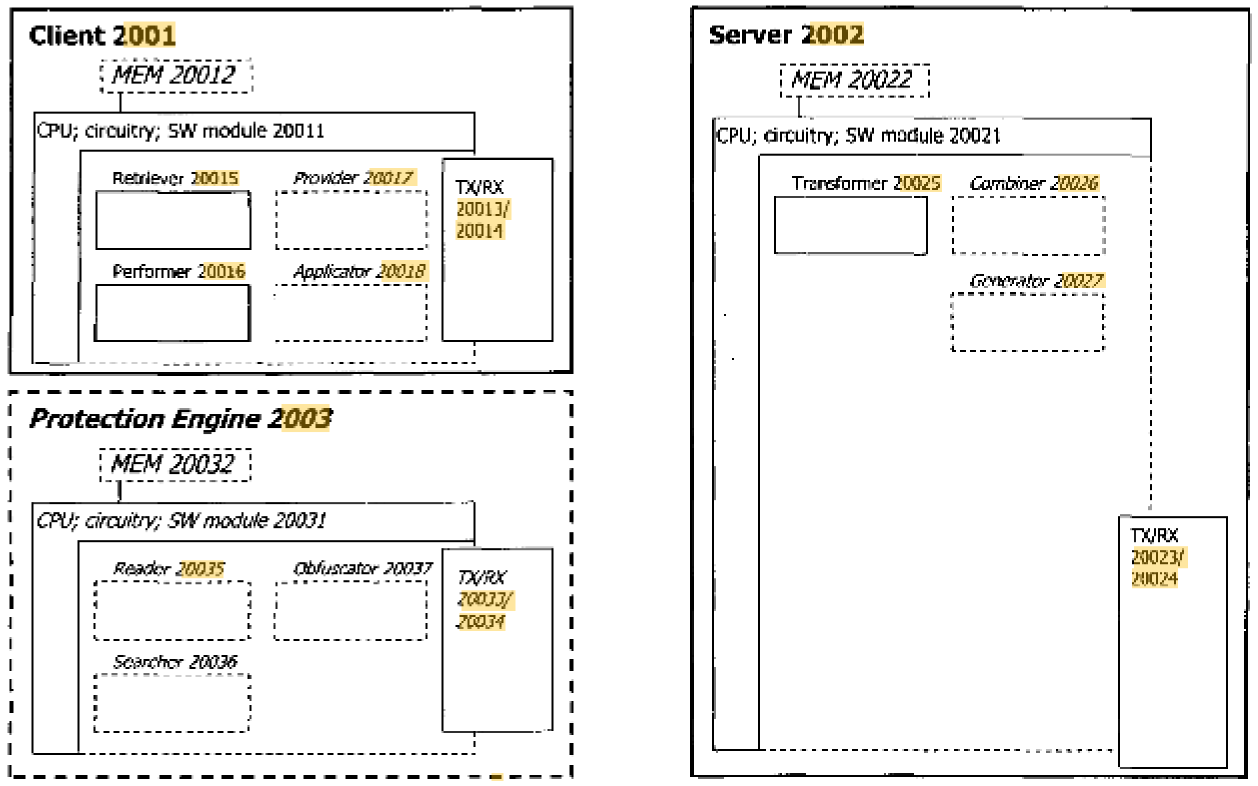


Fig. 1



Client 2001
MEM 20012
CPU; circuitry; SW module 20011
Retriever 20015
Provider 20017
Performer 20016
Applicator 20018
TX/RX 20013/ 20014
Protection Engine 2003
MEM 20032
CPU; circuitry; SW module 20031
Reader 20035
Obfuscator 20037
Searcher 20036
TX/RX 20033/ 20034
Server 2002
MEM 20022
CPU; circuitry; SW module 20021
Transformer 20025
Combiner 20026
Generator 20027
TX/RX 20023/ 20024