Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.28, NO.6, Dec. 2018 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2018.28.6.1297

S.O.S : Shield of Steam API 콜 버스마크 기반의 온라인 게임 ESD DRM 보호

오 동 빈,† 김 휘 강‡
고려대학교 정보보호대학원

S.O.S : Shield of Steam Protection Based on API Call Birthmark in Online Game ESD DRM

Dong Bin Oh,† Huy Kang Kim‡
Graduate School of Information Security, Korea University

요 약

Steam과 같은 온라인 게임 ESD(Electronic Software Distribution)이 등장하면서 게임 불법 복제 방법도 다양화되고 있다. 온라인 게임 ESD에서는 오프라인에서도 게임을 플레이할 수 있어야한다는 특징 때문에 소프트웨어 DRM을 이용하고 있다. 그러나 기본적으로 제공하는 보안 수준이 낮아 쉽게 우회되고 있는 것이 현실이다. 본 연구에서는 불법 복제된 게임의 크랙을 분석하여 Steam DRM을 우회하는 방법에 대해 알아보고 API 콜 버스마크를 이용해 DRM을 보호하는 방법을 개발하였다. 생성한 버스마크는 크랙 그룹을 나타내는 데 있어 85% 이상의 강인성을 보였고, 크랙 여부를 탐지하는 데 95%의 신뢰성을 보였다. 수행한 연구를 통해 온라인 게임 ESD의 보안 향상을 도모할 수 있고, Third Party DRM을 구매할 수 없는 영세한 인디게임 개발자들에게도 일정 수준 이상의 게임 복제 방지 기능을 제공할 수 있다는데 의의가 있다.

ABSTRACT

The rise of Online game ESD(Electronic Software Distribution) like Steam, the method of game piracy are more diversified. In Online Game ESD, Software DRM is applied to game because we have to play in offline situation, but it is easily bypassed due to low security level. In this study, we analyze crack files of pirated games to learn how to bypass Steam DRM and to establish countermeasures for based on API call birthmark. The generated birthmark showed more than 85% resilience in representing crack groups and 95% credibility in detecting cracked games. With this study, it is possible to enhance the security of the online game Electronic Software Distribution platform, and to provide a high level of game piracy protection for indie game developers, especially those who can not purchase Third Party DRM to protect their own games.

Keywords: Steam, Game ESD, Online Game Security, API Call birthmark

I. 서 론

2016년 국내 게임 시장의 규모는 10조 8945억 원으로, 2007년부터 지속적인 상승세를 나타내고 있다[1]. 게임 시장 규모의 상승에 따라 불법 복제된 게임의 유통도 증가하고 있다[2]. Steam으로 대표되는 온라인 게임 ESD(Electronic Software Distribution)에서 구매한 게임을 인터넷을 통해 배

Received(07. 09. 2018), Modified(10. 08. 2018), Accepted(10. 12. 2018)

† 주저자, sin90@outlook.com
‡ 교신저자, cenda@korea.ac.kr(Corresponding author)

포하는 방법이 등장하면서 게임을 불법으로 복제하는 방법도 다양해지고 있다. 이러한 불법 복제된 게임 중 일부는 악성코드가 삽입되어 개인 정보를 탈취해 2차적 피해를 유발하기도 한다[3].

온라인 게임 ESD는 게임 구매자들에게 온라인으로 인증 절차를 거친 이후 구매한 게임 파일이 실행된다. 하지만 매번 게임을 플레이할 때마다 온라인으로 인증을 받는 방법은 온라인 인증을 받을 수 없는 오프라인 환경에서는 자신이 구매한 게임을 플레이할 수 없는 상황이 발생한다. 이러한 상황을 해결하기 위해 DRM(Digital Rights Management)을 게임에 적용하게 된다.

그러나 온라인 게임 ESD에서 사용하는 소프트웨어 DRM은 게임 크랙 그룹에 의해 쉽게 우회되어 게임 불법 복제 시장에 유포되고 있는 현실이다[4]. Crackwatch[5]에서 제공하는 온라인 게임 ESD별 크랙 출시 정보와 게임 출시 정보, 크랙 개발 그룹의 정보를 조사한 결과, Steam DRM을 이용한 최근 200개의 게임은 평균적으로 14일 이내에 크랙되었고, 200개의 게임 중 80% 이상이 게임 출시 후 1일 이내에 크랙되었다. 나머지 20% 이하에 해당 하는 게임 크랙에 2주 이상 소모되었다. 이러한 게임들은 Steam DRM 이외에 Third Party DRM을 사용하거나 플레이어들에게 인기가 없어서 거의 플레이되지 않는 게임이었다. 또한 메이저 크랙 그룹에서 대부분의 출시된 게임을 크랙하고 있는 것을 확인할 수 있었다.

온라인 게임에서는 부정 행위가 다양하게 존재한다. 부정 행위에 대한 조치는 온라인 게임이 제공되는 환경과 상황을 고려해서 대응하도록 설계해야한다[6]. 본 논문은 게임 불법 복제를 막기 위해 온라인 게임 ESD의 DRM을 보호하는 방법을 개발하고자 한다. 2장에서는 기존의 소프트웨어 불법 복제 방지를 위하여 DRM과 관련한 어떠한 연구가 진행되었는지 알아보고 게임 크랙 파일의 고유한 특징을 추출하기 위해 소프트웨어 버스마크와 관련된 연구를 살펴본다. 3장에서는 Steam의 게임 개발자를 위해 제공되는 DRM과 DRM에 적용된 보안 특징에 대해서 알아본다. 4장에서는 Steam 게임 크랙 그룹에서 적용한 DRM 우회 방법에 대하여 알아본다. 5장에서는 제안한 방법에 따라 버스마크를 생성하고 생성한 버스마크를 강인성과 신뢰성에 따라 검증하였다. 검증 결과, 게임 크랙 그룹에 대해서는 85% 이상의 버스마크 강인성이 나타났고 크랙 여부를 판단하는 데 있어서는 95%의 신뢰성을 나타내었다. 6장에서는 실험 결과를 종합하고 실험에서의 한계와 향후 연구에 대하여 제시하였다.

II. 관련 연구

2.1 소프트웨어 DRM 관련 연구

DRM은 소프트웨어를 불법 복제로부터 보호할 수 있는 보호 메커니즘과 구매자의 라이센스 권한을 프로그램이 알아낼 수 있는 라이센스 정책이라는 두 가지의 기능으로 이루어져 있다[7]. 본 논문에서는 게임 ESD의 DRM을 보호를 통한 불법 복제 방지 측면인 보호 메커니즘 기능의 DRM에 초점을 맞추고자 한다. 불법 복제 방지를 위해 진행되는 연구는 하드웨어를 이용한 DRM과 소프트웨어를 이용한 DRM에 대한 연구로 분류할 수 있다.

오명신 등[8]은 불법 복제 방지를 위해 하드웨어 보안 칩을 설계하였다. 복제 방지를 위해서는 보안 강도가 높아야하며 범용적 사용을 위해서는 비용이 저렴해야하며 사용이 편리해야한다. 고안한 보안 칩 내부에는 역공학 방지에 강력한 KSE96이라는 암호 알고리즘을 사용하였다. 그러나 Steam과 같은 온라인 게임 ESD에서는 게임 소프트웨어를 인터넷을 이용하여 구매하고 다운로드 받기 때문에 하드웨어 장치를 사용하는 것은 물리적 한계가 있고 온라인으로 구매, 유통, 판매되는 온라인 게임 ESD라는 서비스 특성과 어울리지 못한다는 문제가 있다.

소프트웨어를 이용한 DRM에 관한 연구는 주로 DRM을 응용하여 실제 서비스에 적용시키는 방향으로 연구가 진행되고 있다. 김종우 등[9]은 네트워크에서의 디지털 콘텐츠의 보호를 위하여 DRM 프레임을 재설계하였다. 디지털 콘텐츠에 실행 파일 구조를 가지는 시큐어 컨테이너를 이용하여 콘텐츠를 암호화하여 보호하고 실행 시 복호화 하여 실행한다. 복제가 되더라도 하드웨어 고유 정보로 암호화하였기 때문에 다른 컴퓨터 환경에서는 실행되지 않는다. Steam과 같은 클라우드 환경의 게임 ESD에서는 여러 기기에서 접속할 수 있기에 게임을 다운로드 받을 때마다 해당 기기에서 암호화를 진행해야 하고 암호화를 진행한다하더라도 3D 게임, VR 게임과 같은 대용량 게임에 대해서는 복호화를 할 때 시간이 오래 소요된다는 문제로 적용하기 힘들다는 단점이 있다. 이진홍 등[10]의 연구에서는 스트리밍 환경에서의

DRM 시스템을 설계하였다. 스트리밍을 위한 DRM 시스템은 실시간으로 데이터가 전송되기 때문에 데이터 전송과 함께 감시와 추적이 가능해야한다. 스트리밍 환경에서의 DRM은 실시간으로 전송되는 방송이나 음원 에 유용할 수 있으나 오프라인 환경에서도 실행되어야하는 게임에 실시간으로 감시할 수 있는 DRM을 적용하는 것은 게임의 특성과 잘 맞지 않는다는 한계가 있다.

2.2 소프트웨어 버스마크에 대한 연구

소프트웨어 버스마크는 코드 재사용 및 표절을 탐지할 수 있는 소프트웨어의 고유한 특징이다[11]. 소프트웨어 버스마크를 생성하는 방법은 정적인 특징을 이용한 방법과 동적인 특징을 이용한 방법이 있다. 정적 특징을 이용한 버스마크는 실행시키지 않고 소프트웨어 내부의 코드를 기반으로 생성한다. 동적 특징을 이용한 버스마크는 소프트웨어를 실행하여 시스템에 나타나는 변화를 기반으로 생성한다[12].

정적 버스마크 생성을 위해 Choi 등[13]은 실행 파일 내부의 IAT(Import Address Table)를 이용하였다. IAT는 실행 파일에서 사용하는 함수와 변수의 포인터를 가지고 있는 테이블이다[14]. 함수와 변수의 포인터를 이용한 정보를 기반으로 버스마크를 생성하였다. 생성한 버스마크를 기반으로 유사한 기능을 가지고 있지만 다른 소프트웨어인 경우, 같은 소프트웨어지만 세부 버전이 다른 경우에 대하여 유사도를 측정하였다. 실험 결과 유사한 기능을 가지고 있지만 실제 소프트웨어는 다른 경우에는 10% 미만으로 유사도가 나타났고 세부 버전만 다른 동일 소프트웨어에 대해서는 60% 이상의 유사도를 보여주었다. 이 연구는 API 콜을 뽑을 필요 없이 실행 파일 구조만 이용하여 버스마크를 추출할 수 있다는 것에 강점이 있다. Choi 등[15]의 연구에서는 IDA 디스어셈블러를 이용해 Code Graph를 생성하고 버스마크로 사용했다. 생성한 Code Graph는 가중치 이분 그래프 정합 방법을 이용하여 유사도를 계산하였고 같은 소프트웨어이지만 세부 버전이 다른 경우 90%의 유사도를 보였다.

동적 버스마크 생성은 API 콜을 이용한 연구가 대표적이다. 최석우 등[12]의 연구에는 기능 단위 동적 API 콜 시퀀스를 이용해 버스마크를 생성하였다. 매개변수에 따른 실행 결과가 변하는 것을 고려하지 않고 함수 자체만 고려하였다. 생성된 버스마크는 준전체 정렬 알고리즘을 이용하여 유사도를 측정하였다. 실험 결과 같은 소프트웨어이지만 세부 버전이 다른 경우 82%, 같은 기능을 가지고 있지만 프로그램이 다른 경우 68%로 유사도가 측정되었다. Kim 등[16]의 연구에서는 API 콜 빈도를 이용해버스마크를 생성하였다. 목표가 되는 어플리케이션에 주요 함수들이 실행되는 빈도를 기반으로 버스마크를 만들어 코사인 유사도를 이용해 유사도를 측정하였다. 생성한 유사도는 동일 소프트웨어이지만 다른 버전을 가졌을 경우 99% 이상의 유사도를 보였다.

본 논문에서는 API 콜 호출 빈도를 이용한 동적 버스마크를 이용하였다. 동적 버스마크의 경우 정적 버스마크와 달리 소프트웨어가 실행되기 전에 초기화하는 과정을 버스마크로 생성할 수 있다. 이를 이용해 크랙 게임 소프트웨어 자체의 버스마크를 생성하는 것이 아니라, 각 크랙 그룹에서 게임을 크랙하기 위해 사용하는 방법에 대한 버스마크를 생성하여 유사도를 측정해 크랙 여부를 탐지하는 데 사용하였다.

III. Steam DRM

Steam DRM은 Steamworks API에서 제공하는 DRM이다. Steam이라는 온라인 게임 ESD을 통하여 게임을 배포하려면 Steam으로부터 일정 절차를 거친 뒤 게임 개발자라는 인증을 받아야한다. 인증을 받은 이후, Steamworks API를 이용하여 Steam DRM을 적용하고 게임 배포 서버로 게임을 업로드할 수 있다[17].

Steam DRM은 버전 1부터 체크섬 검사 및 계정 이용자 검증과 같은 보안적 측면을 제공하였다. 특히 버전 2 이후 불법 복제를 방지하기 위한 기능이 추가

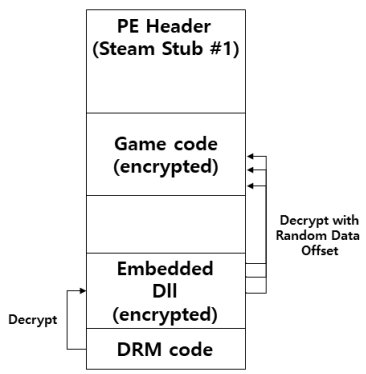

Fig. 1. Structure of Steam DRM Version 2.x

Table 1. Security Features of Steam DRM

Version	1.0	1.5	2.x	3.x
Checksum	O	O	O	O
Protection ID	X	Steam Stub	Steam Stub (new #1)	Steam Stub (new #2)
Code Encryption	X	CBC	XTEA (DLL) AES-256 (Game)	not used
Debugger Detection	X	X	O	O
Verification	Basic Ownership Check	Valid Digital Signature (steam.dll)	Cryptography Signature (steamclient.dll)	Cryptography Signature (steamclient.dll)

되었다. 대표적으로 steamclient.dll을 이용한 암호화 기반의 사용자 검증 기능과 안티디버깅이 포함되었다. 불법 복제를 위해 게임 코드를 추출하기 위해서는 게임 실행 파일 내에 있는 Dll 파일을 이용하여 복호화를 해야 한다. Dll 파일은 임의의 오프셋 순서로 배치된 게임 코드 데이터를 가지고 있다. 게임 코드를 복호화 하여 실행 파일을 추출하기 위해서는 관련된 오프셋 정보를 이용해야한다. 하지만 게임 파일 내의 Dll 또한 암호화 되어있어서 DRM 코드를 먼저 실행하여 Dll의 복호화가 선행되어야 게임 코드를 복호화 할 수 있다[18].

Steam DRM의 버전 3은 버전 2와 다르게 게임 코드 실행을 위해 게임 파일 내의 Dll을 복호화 할 필요가 없이 DRM 코드가 게임 코드와 Dll을 모두 복호화 한다. 기존의 버전 2에서는 DRM 코드와 Dll 파일이 나란히 배치되어 있어 Dll의 위치를 찾아 복호화하기 수월하였다. 버전 3에서는 DRM 코드와 Dll의 위치가 분리되어 있어서 암호화된 Dll의 시작점을 찾기 어려워 복호화 난이도가 상승하였다.

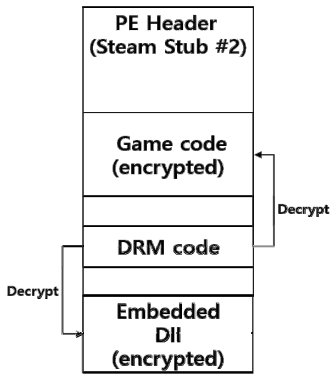

Fig. 2. Structure of Steam DRM Version 3.0

IV. Steam DRM 우회 방법

4.1 Steamless

Steam DRM은 버전 1.5부터 Steam Stub라는 패커를 지원하고 있다. Steamless는 Steam Stub이라는 패커를 언패킹하기 위해 Steam 어플리케이션의 Steam DRM이 게임 코드를 복호화 하는 과정과 동일하게 진행된다. 그러나 Steam 어플리케이션의 경우, 복호화 된 이후 게임 코드를 실행하는 반면에 Steamless는 실행하면 게임 코드를 복호화 후 실행 파일을 재구성하여 파일 형태로 저장한다[19]. Steam Stub가 실행 파일에 적용되었는지를 알기 위해서는 Protection iD[20]라는 도구를 이용해 확인할 수 있다.

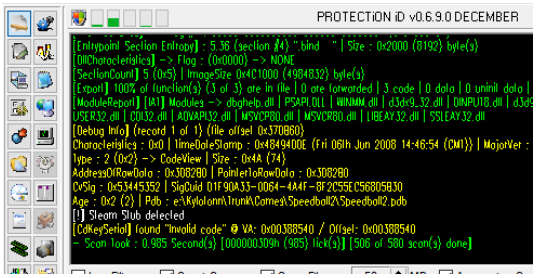

Fig. 3. Use Protection iD to detect Steam Stub

4.2 Dll Emulation

Dll Emulation은 로드되어야하는 Dll의 기능과 유사한 기능을 가진 Dll을 재구성하는 것이다. Steam 게임이 실행되기 위해서는 steam_api.dll의 SteamAPI_Init을 통해 게임 코드를 호출해야한다. 정상적인 방법으로 Steam을 통해 게임을 실행하는

경우, 사용자 검증을 위해 steam_api.dll은 steamclient.dll을 임포트한다. 크랙된 게임 파일의 경우, steamclient.dll을 임포트하면 사용자 검증과 관련된 기능까지 임포트하기 때문에 해당 Dll 을 임포트하지 않아야한다. steamclient.dll의 내부에 사용자 검증에 필요한 함수 이외에도 Steam 게임 실행에 필요한 함수가 존재하기 때문에 이들을 이용해야만 게임을 실행할 수 있다. Table 2는 게임 크랙 그룹이 Steam 게임을 실행하기에 필요한 steamclient.dll의 함수를 이용하기 위해 사용하는 Dll Emulation의 예이다. 해당 함수들을 이용하기 위해 SteamAPI_Init 함수를 포함하고 있는 Dll인 steam_api.dll을 에뮬레이팅하여 사용하게 된다. steamclient.dll 안에 있는 Steam의 게임 실행에 필요한 함수를 steam_api.dll에 포함시키고 게임을 실행할 때 호출하기 위해 Table 2와 같이 구별을 위한 접두사를 통해 함수의 출처를 구분한다.

Table 2. Example of Dll Emulating

steam_api.dll (Cracked)	Steamclient.dll (Pure)
SteamAPI_ISteamClient_ ConnectToGlobalUser	steam_ConnectTo GlobalUser
SteamAPI_ISteamClient_ CreateSteamPipe	Steam_CreateSte amPipe
SteamAPI_ISteamClient_ BReleaseSteamPipe	Steam_BReleaseS teamPipe
SteamAPI_ISteamClient_ CreateLocalUser	Steam_CreateLoc alUser

4.3 Dll Search Order Hijacking

Dll Search Order Hijacking은 Dll의 실행 경로를 역이용하는 방식이다. Windows 운영체제에서는 실행 파일에 필요한 Dll을 로드할 때 일정한 순서에 따라 Dll 존재 여부를 검색한다. 검색 결과에 매칭된 Dll이 존재하면 실행 파일은 해당 Dll을 로드하게 된다[21].

정상적인 Steam을 이용한 게임 실행은 Steam 어플리케이션을 통해 이루어진다. Steam 어플리케이션을 이용해 게임을 로드하게 되면 Dll 검색 순서에 따라 2순위인 Steam 어플리케이션 디렉터리에 존재하는 Dll을 로드하게 된다. 반면에 Steam 크랙 게임은 Steam 어플리케이션을 통해서 실행시키지 않고 크랙 게임 디렉터리의 실행 파일이나 별도의 런쳐를 통해 게임을 실행하게 된다. 실행된 크랙 게임은 Steam의 어플리케이션 디렉터리에 있는 Dll을 로드하지 않고 5순위인 현재 디렉터리에 해당하는 런쳐나 크랙 실행파일의 디렉터리에 있는 에뮬레이팅된 Dll을 로드하게 된다.

Table 3. Dll Search Order

Search Order	Description
1	Known Dll
2	Application loaded directory
3	System directory
4	Windows directory
5	Current directory
6	PATH directory

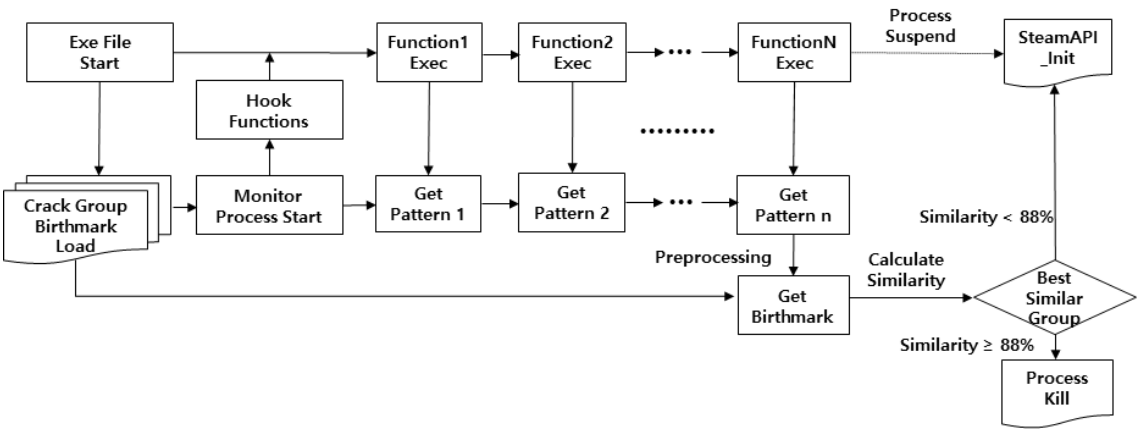

Fig. 4. Overview of S.O.S (Shield of Steam)

V. S.O.S (Shield of Steam)

5.1 개 요

S.O.S(Shield of Steam)는 본 논문에서 제안하는 API 콜 버스마크를 이용한 DRM 보호 방법이다. 게임을 실행하면 파이썬 디버거 기반의 API 콜 모니터링 프로세스를 생성하고 사전에 정의된 API들을 후킹하여 게임 실행 파일이 호출하는 API들을 체크한다. 체크한 API는 정해진 규칙에 따라 각각의 패턴을 생성한다. 생성된 패턴은 전처리를 통해 게임 소프트웨어 고유의 버스마크로 활용한다. 생성한 버스마크를 사전에 생성된 크랙 그룹의 버스마크와 비교하여 특정 크랙 그룹의 버스마크와 유사도가 88% 이상으로 나타나면 크랙 게임으로 판단하고 게임 프로세스를 종료하게 된다. 만약 모든 크랙 그룹과의 버스마크 유사도가 88% 미만이라면 크랙되지 않은 게임으로 판단하고 게임이 실행되게 되도록 크랙 여부 모니터링 방안을 구상하였다.

5.2 데이터 수집

실험을 위해 크랙이 적용된 Steam 게임 65개와 크랙되지 않은 게임 19개를 수집하였다. 그 중 메이저 크랙 그룹의 고유한 버스마크를 생성하기 위해 4개의 메이저 게임 크랙 그룹으로부터 제작된 46개의 크랙된 게임 파일을 이용하였다. 나머지 38개의 마이너 크랙 그룹의 크랙 게임과 크랙되지 않은 게임 은 버스마크의 신뢰성을 검증하는데 사용하였다.

API 콜을 수집하기 위해 마이크로스프트에서 개발한 Detours라는 소프트웨어 패키지를 활용하였다. Detours는 실행 파일의 API를 후킹하여 API 콜을 모니터링하거나 API 콜을 이용하는 데 사용할 수 있다. 또한 다양한 윈도우 운영체제 환경에서 작동할 수 있다는 장점이 있다[22]. 게임 실행 파일 내에서 API 콜을 추출하기 위해 Detours를 이용해 게임 소프트웨어를 실행하고 나타나는 API 콜들을 텍스트 파일로 저장하였다.

Table 4. Dataset Description

Birthmark Test	Type	Group Num	# of Games
Resilience	Cracked	Group #1	15
		Group #2	11
		Group #3	10
		Group #4	10
Credibility	Cracked	Group #5	3
		Group #6	3
		Group #7	3
		Group #8	3
		etc	7
	Pure	-	19

5.3 데이터 추출

게임을 크랙하는데 사용되는 Dll 파일과 게임을 실행하는 데 사용되는 exe 파일을 대상으로 역공학을 이용해 API 콜 중 게임 크랙과 관련된 함수 14개를 선정하였다. 선정한 함수는 파일시스템 접근, 문자열 비교, 라이브러리 로드와 기타 게임 실행에 필요한 함수를 대상으로 하였다. 선정한 함수를 기반으로 SteamAPI를 초기화하는 SteamAPI_Init이 호출되기 전까지의 API 콜을 선별하였다.

Table 5. Patternized Function list

FileSystem	Createfile FindFirstFile GetFileAttributes CreateFileMapping GetFullPathName
String	CompareString IstrcmpiW lstrlen
Library	LoadLibrary GetMouduleFileName
etc	GetCommandLine CreateEventW GetPrivateProfileString SetEnvironmentVariable

5.4 크랙 그룹 버스마크 생성

선별된 API 콜을 이용해 패턴을 생성하기 위해 매핑을 진행하였다. 매핑은 일반적인 API 호출을 포함하여 Public 계정의 디렉터리에 접근하거나 특정 크랙 그룹과 관련된 디렉터리 생성, Steam 관련된 문자열 비교와 같은 정상적으로 Steam을 통해 게임을 실행시켰을 때에는 나타나지 않는 API 콜과 변수를 대상으로 하였다.

선별한 14개의 함수로부터 함수 별 최대 6개의 번

수가 나타낼 수 있는 특징들을 이용하여 총 55 종류의 패턴을 생성하여 크랙 그룹의 버스마크를 생성하였다.

55개의 패턴을 기반으로 매핑된 버스마크는 전처리 과정을 거친다. 전처리 방법으로는 버스마크 내의 단순히 반복되는 패턴에서 중복된 부분을 제거하는 방법을 사용하였다. 사용자 환경에 따라 반복문을 이용해 게임 실행 환경을 세팅하는 경우와 같이 실제로는 동일한 행위이지만 의미 없이 패턴의 길이를 증가시키는 경우에는 동일 행위로 치환하여 버스마크의 길이를 줄일 수 있다[23]. 버스마크의 길이를 줄임으로 버스마크 간의 유사도를 비교할 때 정규화 처리하여 유사도 비교를 위한 버스마크의 타당성을 보장할 수 있다.

전처리된 버스마크를 동일한 그룹의 다른 크랙된 게임으로부터 추출한 버스마크들과 LCS(Longest Common Substring) 알고리즘을 이용해 공통 문자열을 생성하고 게임 크랙 그룹의 고유한 버스마크로 활용하였다.

게임 파일의 크랙 여부를 확인하기 위해서는 생성한 버스마크와 사전에 크랙 그룹의 고유한 버스마크

Table 6. Rules to map Parameter

Function Type	Parameter	Pattern
FileSystem	cdx	Fc
	dll	Fl
	ini	Fi
	other	Fd
	public	Fp
	steam	Fs
String	cdx	Sc
	Environment	Sv
	mono library	Sm
	Steam	Ss
Library	cdx	Lc
	mono library	Lm
	other	Ld
	Steam	Ls
Etc	cdx	Ec
	Environment	Ev
	other	Ed
	SmartSteam	Ee

간의 유사도를 측정하였다. 유사도 측정을 위해 TF-IDF를 이용하여 크랙 게임 버스마크를 벡터로 변환하였다. TF-IDF는 문자열에 있는 특정 요소가 얼마나 자주 등장하는 지를 나타내는 통계적 값으로 변환할 때 주로 사용한다[24]. 이를 이용해 Steam 게임을 실행함에 있어서 특정 패턴의 빈도와 중요도를 기반으로 크랙 그룹 별 특징을 판별할 수 있다. 생성된 버스마크 각각을 하나의 문서로 취급하여 버스마크 간의 단어 빈도와 역 문서 빈도를 계산하였다. 위의 방법으로 문서를 구성하는 단어, 즉 버스마크를 구성하는 패턴에 세분화된 가중치를 부여할 수 있다. 가중치를 이용해서 아래의 식과 같이 벡터로 표시할 수 있다.

$$\vec{a} = \sum_{w=1}^{k} (tf \times (\log(\frac{N+1}{N_w+1})+1) \times \overrightarrow{e_w}) \quad (1)$$

TF-IDF를 이용해 벡터로 변환된 버스마크는 코사인 유사도를 측정할 수 있다.

$$similarity = \frac{\vec{a} \bullet \vec{b}}{|\vec{a}||\vec{b}|} \quad (2)$$

만약 유사도가 특정 크랙 그룹과 제일 높으면 게임 프로세스를 종료하게 되고, 정상적인 Steam을 이용하여 실행한 경우의 버스마크와 유사하면 모니터링 프로세스만 종료하게 되고 게임 프로세스는 진행되도록 구상하였다.

5.5 버스마크 테스트

버스마크를 테스트하기 위해서는 강인성과 신뢰성을 대상으로 한다. 소프트웨어를 위한 버스마크에서 강인성은 동일 소소코드라도 난독화나 컴파일러를 이용하여 소프트웨어에 변화가 있더라도 같은 소프트웨어임을 나타낼 수 있는 정도를 의미한다. 신뢰성은 같은 기능을 가진 소프트웨어이지만 명백히 다른 소프트웨어일 경우에 다르다고 판단할 수 있는 기준으로 버스마크를 테스트할 수 있다, 본 논문에서는 소프트웨어가 아닌 크랙 그룹에서 사용하는 크랙 방식을 기반으로 버스마크를 생성하였다. 크랙 그룹의 API 호출 시퀀스를 기반으로 생성된 버스마크 또한 신뢰성과 강인성을 평가할 수 있다.

게임 크랙 그룹 버스마크의 강인성을 판단하기 위해 수집한 46개의 게임 실행 파일에 대해 생성한 크랙 그룹 버스마크와 유사도를 비교한 후 평균을 내어 각 그룹 버스마크가 그룹의 특징을 대표하고 있는지를 확인하였다. 실험을 진행한 결과, 강인성 평가에 사용된 4가지 크랙 게임 그룹 모두 자신의 그룹에서 생성한 버스마크에서 85% 이상의 유사도를 보이면서 높은 강인성을 나타내고 있었다. 다른 크랙 그룹과 크랙 그룹이 사용하는 Steam 게임의 크랙 방법인 Dll Emulation이나 Dll Search Order Hijacking과 같이 게임 크랙에 사용하는 방법은 유사하지만 파라미터에 들어가는 변수 이름과 같이 세부적인 사항에서만 차이가 존재하기에 높은 유사도를 나타낸다고 볼 수 있다.

버스마크의 신뢰성을 테스트하기 위해서는 마이너 크랙 그룹에서 크랙한 게임과 Steam을 통해 정상적으로 받은 크랙되지 않은 게임을 대상으로 하였다. 해당 게임들로부터 동일한 방법으로 버스마크를 생성한 뒤 기존에 생성한 4가지 그룹의 버스마크와 유사도를 측정하는 방법으로 실험을 진행하였다.

강인성 실험을 기반으로 신뢰성에서 마이너 크랙 그룹에서 크랙한 Steam 게임과 크랙되지 않은 Steam 게임에 대해 크랙 여부를 판단할 수 있는 유사도의 임계값을 설정해야한다. 강인성 실험에서 대부분의 크랙 그룹의 자신을 나타내는 크랙 그룹 버스마크 유사도의 평균이 약 88%로 나타났으므로, 신뢰성 실험에서 사용할 유사도의 임계값을 88%로 설정하였다. 게임 버스마크의 유사도가 88% 이상 일 경우 크랙된 게임으로 판단하였고, 88% 미만일 경우 크랙되지 않은 게임으로 판단하여 버스마크의 신뢰성을 검증하는 실험을 진행하였다.

실험 결과, 실제로 크랙되었지만 버스마크를 이용해 크랙되었다고 예측된 경우 약 95%의 탐지율을 보였고 크랙되지 않았지만 크랙된 게임이라고 판단하는 약 10%의 오류를 나타내고 있었다.

Table 8. Result of Credibility Test

	Predicted Cracked	Predicted Pure
Actually Cracked	94.73%	5.27%
Actually Pure	10.52%	89.48%

위의 두 가지 실험을 통해 버스마크의 신뢰성과 강인성을 모두 검증하였다. API 호출 기반 버스마크를 생성한 데이터로부터 신뢰성을 검증한 결과 각 크랙 그룹의 버스마크의 신뢰도가 평균 85% 이상 나타났고, 강인성 실험의 경우 약 95%의 정확도를 기반으로 게임의 크랙 여부를 판단할 수 있었다. 강인성 실험에서 크랙된 게임 중에서 실제로 크랙되었다고 탐지된 경우의 유사도와 크랙되지 않은 게임 중에서 실제로 크랙되었다고 판단하지 않은 경우의 버스마크의 유사도가 각각 약 94%와 약 69%로 나타남을 알 수 있었다. 약 25%의 유사도의 차이는 크랙 여부에 따라 호출되는 API에 차이가 있음을 의미한다. 수집한 게임 데이터가 늘어나면 크랙 그룹에서 사용하는 크랙 방법이 반영된 더 정확한 버스마크가 생성되어 크랙 게임에서는 유사도가 더 높게, 크랙되지 않은 게임에서는 유사도가 더 낮게 측정되어 더욱 명백한 크랙 여부를 판단할 수 있을 것이다.

VI. 결 론

Steam DRM은 Steam이라는 세계 최대 규모의 온라인 게임 ESD임에도 불구하고, 게임의 불법 복제를 방지하기 위한 보안은 취약하다.

게임 불법 복제를 방지하기 위해 게임 개발자들은

Table 7. Result of Resilience Test

	Group #1 birthmark	Group #2 birthmark	Group #3 birthmark	Group #4 birthmark
Group #1 Resilience	**88.82%**	84.09%	84.73%	88.67%
Group #2 Resilience	84.54%	**86.62%**	84.91%	84.86%
Group #3 Resilience	83.36%	81.69%	**85.77%**	82.43%
Group #4 Resilience	79.70%	86.58%	85.37%	**88.75%**

Steam DRM 이외에도 Third Party DRM을 이용하여 게임에 추가 DRM을 적용할 수 있다. 그러나 Third Party DRM을 개발하기 위해 노력과 시간이 필요할 뿐 아니라 상용 DRM을 사용할 경우에는 추가적인 비용까지 소요된다. 특히 Steam 게임의 대부분을 차지하는 인디 게임들은 소규모 인력과 자금으로 개발하는 경우가 많아 Third Party DRM을 적용하기 어려운 실정이다[25].

본 논문에서는 API 콜 버스마크를 이용한 온라인 게임 ESD의 DRM 보호 방법을 개발하였다. 생성한 버스마크를 검증하기 위해 강인성과 신뢰성을 수집한 게임을 대상으로 실험하였다. 실험 결과 크랙 그룹의 자체적인 특징을 나타내는 강인성의 경우 각 그룹 모두 자신의 그룹에서 85% 이상의 유사도를 보이는 것으로 나타났고, 신뢰성의 경우 약 95%의 정확도를 기반으로 크랙 그룹을 탐지하는 버스마크를 생성했음을 알 수 있었다. 신뢰성과 강인성이 검증된 버스마크를 사용해서 파이썬으로 만든 디버거를 이용해 크랙된 Steam 게임을 탐지하는 모니터링 프로그램을 제작할 수 있었다.

그러나 실험에 사용한 게임 중 자체적으로 디버깅 탐지나 후킹 탐지 기능이 있는 경우에는 개발한 방법이 작동되지 않는 경우가 있었다. 또한 탐지되는 속도의 측면에서 실제 게임 서비스에 적용하여 온라인상으로 서비스하기에는 느리다는 한계가 있었다. 추후 연구에서는 경량화된 알고리즘을 적용하여 탐지 속도를 높일 수 있을 것이다. 후킹 탐지 회피와 관련해서는 Windows에서 유저 모드의 API를 이용한 후킹이 아니라 커널 모드에서의 후킹과 같은 다양한 후킹 탐지 회피 기법을 이용할 것이다.

추가적으로, 게임 크랙과 관련된 데이터를 더 확보해서 단순히 게임의 크랙 여부 탐지뿐 아니라 게임 크랙 파일이 어떠한 크랙 그룹에서 개발되었는지 탐지할 수 있는 게임 크랙 그룹 프로파일링 기법을 개발하는 연구를 진행하고자 한다.

References

[1] KOCCA, "White paper on Korean Games", Dec. 2012

[2] VPNrank, "Best Game Torrents Sites", https://www.vpnranks.com/best-game-torrents-sites/, Aug, 2018.

[3] Ji-young Woo, Huy-Kang Kim. "Survey and research direction on online game security." Proceedings of the Workshop at SIGGRAPH Asia. ACM, 2012.

[4] Game Donga, "the history of crack group", http://game.donga.com/51123/

[5] CrackWatch, "Crack Status of All Games", https://crackwatch.com/, Aug. 2018.

[6] Byung-Il Kwak, Huy-Kang Kim. "A survey and categorization of anomaly detection in online games," Journal of the Korea Institute of Information Security & Cryptology, Vol. 25, No. 5, pp. 1097~1114, Oct, 2015.

[7] Steven Davis. Protecting games: a security handbook for game developers and publishers. Charles River Media, Inc., 2009.

[8] Myung-Shin Oh and Seung-Jo Han, "The Secure Chip for Software Illegal Copy Protection," Journal of the Korea Institute of Information Security & Cryptology, Vol. 12, No. 4, pp. 87~98, Aug, 2002.

[9] JJong-Woo Kim, Won-Il Yang and Seung Jo Han, "Design of DRM Frame for Digital Contents Protection in Network," Journal of the Korea Institute of Information Security & Cryptology, Vol. 16, No. 3, pp. 101~113, Jun, 2006.

[10] Jin-Heung Lee, Tea-Jung Kim and Ji-Hwan Park, "Design and Implementation of Secure DRM System for Contents Streaming," Journal of the Korea Institute of Information Security & Cryptology, Vol. 13, No. 4, pp. 177~186, Aug, 2003.

[11] Seong-Soo Park and Hwan-Soo Han, "Detecting Software Similarity Using API Sequences on Static Major Paths," Journal of KIISE, Vol. 41, No. 12, pp. 1007~1012, Dec, 2014.

〔12〕 Seok-Woo Choi, Woo-Young Cho and Tai-Sook Han, "A Functional Unit Dynamic API Birthmark for Windows Programs Code Theft Detection," Journal of KISS : Software and Applications, Vol. 36, No. 9, pp. 767~776, Sep, 2009.

〔13〕 Jong-Cheon Choi, Yong-Man Han, Seong-Je Cho, Hae-Young Yoo, Jin-Woon Woo. "A static birthmark for MS windows applications using import address table." Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2013 Seventh International Conference on. IEEE, 2013.

〔14〕 Ho-Dong Lee, Reverse Engineering Volume 1 File Structure, Hanbit Meia, Sep, 2016.

〔15〕 Seok-Woo Choi, Hee-Wan Park, Hyun-Il Lim, Tai-Sook Han. "A static birthmark of binary executables based on API call structure." In Annual Asian Computing Science Conference Springer, Berlin, Heidelberg. pp. 2-16. Dec. 2007.

〔16〕 Dong-Jin Kim, Yong-Man Han, Seong-Je Cho, Hae-Young Yoo, Jin-Woon Woo, Yun-Mook Nah, Min-Kyu Park, Lawrence Chung. "Measuring similarity of windows applications using static and dynamic birthmarks." Proceedings of the 28th Annual ACM Symposium on Applied Computing. ACM, 2013.

〔17〕 Steam, "Steam DRM Documentation", https://partner.steamgames.com/doc/features/drm, Aug. 2018.

〔18〕 PCGamingWiki, "Cyanic / SteamDRM", http://pcgamingwiki.com/wiki/User:Cyanic/Steam_DRM, Aug. 2018.

〔19〕 Github, "Steamless, DRM remover", https://github.com/atom0s/Steamless, Aug. 2018.

〔20〕 Protection ID Home Page, "Protection iD", http://pid.gamecopyworld.com, Aug, 2018.

〔21〕 Tae-Ho Kwon, Zhendong Su. "Automatic detection of unsafe dynamic component loadings." IEEE Transactions on Software Engineering 38.2 (2012): 293-313.

〔22〕 Galen Hunt, Doug Brubacher. "Detours: Binary interception of win 3 2 functions." 3rd usenix windows nt symposium. 1999.

〔23〕 In-Kyeom Cho and Eul-Gyu Im, "Improvement of Performance of Malware Similarity Analysis by the Sequence Alignment Technique," KIISE Transactions on Computing Practices, Vol. 21, No. 3, pp. 263~268, Mar, 2015.

〔24〕 Eun-Soon You, Gun-Hee Choi Seung-Hoon Kim, "Study on Extraction of Keywords Using TF-IDF and Text Structure of Novels," Journal of the Korea Society of Computer and Information, Vol. 20, No. 2, pp. 121~129, Feb, 2015.

〔25〕 Starforce, "Game Protection Steam" , http://www.star-force.com/blog/index.php?blog=2683, Aug. 2018.

〈저자소개〉

오 동 빈 (Dong Bin Oh) 학생회원
2018년 2월: 경찰대학 행정학과 학사
2018년 3월~현재: 고려대학교 정보보호대학원 정보보호학과 석사과정
〈관심분야〉 온라인게임 보안, 악성코드, 디지털 포렌식

김 휘 강 (Huy Kang Kim) 종신회원
1998년 2월: KAIST 산업경영학과 학사
2000년 2월: KAIST 산업공학과 석사
2009년 2월: KAIST 산업및시스템공학과 박사
2004년 5월~2010년 2월: 엔씨소프트 정보보안실장, Technical Director
2010년 3월~2014년 12월: 고려대학교 정보보호대학원 조교수
2015년 1월~현재: 고려대학교 정보보호대학원 부교수
〈관심분야〉 온라인게임 보안, 네트워크 보안, 네트워크 포렌식