Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.25, NO.6, Dec. 2015 ISSN 2288-2715(Online)
http://dx.doi.org/10.13089/JKIISC.2015.25.6.1465

FPS 게임 서버 로그 분석을 통한 클라이언트 단 치팅 탐지 기능 개선에 관한 연구

김 선 민,† 김 휘 강‡
고려대학교 정보호호대학원

A research on improving client based detection feature by using server log analysis in FPS games

Seon Min Kim,† Huy Kang Kim‡
Graduate School of Information Security, Korea University

요 약

일반적으로 온라인 게임에서 사용하는 치팅 프로그램 탐지 모델은 클라이언트 단의 치팅 흔적을 조사하여 이를 실시간으로 탐지하는 방식과 서버의 누적 로그에 탐지 알고리즘을 적용하여 치팅 유저를 분류하는 서버 단 탐지 모델로 나누어진다. 클라이언트 탐지 모델은 Anti-reversing 기능 제공과 게임 메모리 변조와 같은 다양한 치팅 공격에 대한 탐지가 가능하나, 탐지 모듈이 클라이언트 단에 배포되어 동작하여 분석 및 우회가 가능할 뿐 아니라 오탐지로 인해 기능의 확장에 한계가 있다는 단점이 존재한다. 이로 인해 많은 주목을 받고 있는 서버 로그 분석 탐지 모델은 강력한 탐지 및 높은 정확도를 자랑하나, FPS와 같이 저 사양의 쉽고 단순한 접근성을 가진 로그 데이터만으로는 치팅 유저와 일반 유저 사이의 유의미한 차이점을 찾기가 어려워 이를 활용하지 못하는 단점이 존재하였다. 본 논문에서는 상기한 두 탐지 모델의 단점을 보완하기 위해, 클라이언트 단의 게임 보안 솔루션의 로그 정보와 서버 로그를 융합한 실험을 통해 효율적인 탐지 모델을 재설계 하였으며 실제 서비스 중인 FPS 게임에 이를 검증해 보았다.

ABSTRACT

Cheating detection models in the online games can be divided into two parts. The one is on client based model, which is designed to detect malicious programs not to be run while playing the games. The other one is server based model, which distinguishes the difference between benign users and cheaters by the server log analysis. The client based model provides various features to prevent games from cheating, For instance, Anti-reversing, memory manipulation and so on. However, being deployed and operated on the client side is a huge weak point as cheaters can analyze and bypass the detection features. That Is why the server based model is an emerging way to detect cheating users in online games. But the simple log data such as FPS's one can be hard to find validate difference between two of them. In this paper, In order to compensate for the disadvantages of the two detection model above, We use the existing game security solution log as well as the server one to bring high performance as well as detection ratio compared to the existing detection models in the market.

Keywords: Online game security solution, FPS Online game security, FPS Server log analysis

Received(09. 08. 2015), Modified(11. 17. 2015), Accepted(11. 24. 2015)

† 주저자, sos2899@korea.ac.kr
‡ 교신저자, cenda@korea.ac.kr(Corresponding author)

I. 서 론

전 세계 게임 시장의 규모는 지속해서 성장해 왔으며, 여러 분야에서 게임을 이용한 다양한 연구가 진행되고 있다.

이 중 온라인 게임의 봇(bot)탐지는 괄목할 만한 성장과 함께 새롭게 떠오른 연구 분야 중 하나로 게임 내의 균형을 무너뜨리며, 게임사의 수익 구조에 악영향을 끼치는 행위 및 프로그램을 효율적으로 탐지하는 방법을 연구하는 분야이다.

오늘날 온라인 게임의 보안 영역은 게임 아키텍처 전 분야[1]로 확대됨에 따라 이를 탐지하기 위한 많은 연구가 이루어지고 있으며 그 중 MMORPG (Massive Multiplyer Online Role Playing Game)의 봇 탐지[2,3,4]와 FPS의 치팅 프로그램 탐지 연구[5,6,7,8,9]는 가장 활발한 연구 분야로 온라인 게임 보안을 이끄는 주축이 되었다.

1.1 게임 봇과 치팅 프로그램의 차이점

MMORPG의 봇 프로그램과 FPS의 치팅 프로그램은 서로 다른 이름으로 명명되듯이 그 사용 방법과 목적이 다르다.

MMORPG의 봇 프로그램은 프로그램화된 반복적인 행위로 자동 사냥이나 자원 채집 등을 통해 게임 내의 균형을 무너뜨리지만, FPS의 치팅 프로그램이란 게임에서 제공하지 않는 다양한 정보 및 기능을 제공하여 치팅 유저의 승리 확률을 높여준다.

일반적으로 FPS의 치팅 프로그램에서 제공하는 기능은 Table 1과 같이 분류할 수 있다.

Table 1에서 분류한 치팅 기능은 실제 FPS 게임의 치팅 프로그램에 포함된 기능들로 Wallhack이란 게임 내 그래픽 함수의 변조 및 후킹을 통해 장애물 및 벽 뒤에 숨어 있는 적군의 움직임을 확인 할 수 있게 해주어 일반 유저보다 높은 Kill 점수를 획득하여 승리 확률을 높여주며, ESP 치팅 기능은 적군의 위치 및 헬스 포인트(health point)의 정보를 화면에 출력해주어 게임 플레이 시, 높은 Kill 점수뿐 만 아니라 위 헬스 포인트 정보를 통해 빠른 미션 완수가 가능하다. Aimbot이란 적군의 좌표를 계산하여 자동 조준해주는 치팅 기능으로 이를 통해 연속된 헤드 샷(head shot)이 가능하여 게임 내의 가장 큰 악영향을 끼치는 치팅 기능이다. Fly hack이란 자신의 x, y, z 좌표를 변조하여 공중에서 모든 플레이어의 움직임을 확인하면서 게임 플레이가 가능하며 이를 통해 높은 Kill 점수를 획득하여 미션 승리 확률을 높여 줄 수 있다. Teleport란 게임 내 적군의 좌표를 확인하여 자신의 좌표를 동일하게 변경하여 먼 거리의 적군의 좌표까지 빠르게 이동할 수 있는 기능으로, 적군의 위치로 바로 이동한 후 사격 하여 높은 Kill 점수와 더불어 가장 적은 시간 안에 많은 Kill 점수를 획득할 수 있다.

Table 1. An overview of cheating types

Feature	Description
Wallhack	It provides a plyer to see enemies through obstacles such as walls.
ESP	It stands for extrasensory perception, which provides extra information about playing conditions.
Aimbot	it automatically calculates the location of visible enemies and in what direction a shot would have to be made in order to get a headshot.
Flyhack	This means flying into the air, allowing the player to move in ways not intended by the game mechanics.
Teleport	It can be used to teleport to an enemy's position directly.
Infinite ammo	It removes the recoil that appears when firing.
Infinite health	It gains a huge amount of health points.

Infinite ammo와 Infinite health는 적군의 사격으로 타격을 받더라도 헬스 포인트가 무한대로 죽지 않거나 방어 능력이 탁월하여 죽지 않는 무적 상태를 만들어 주는 기능들이며, 이를 통해 일반 유저에 비해 높은 Kill 점수를 획득할 수 있다.

Table 1의 치팅 기능들은 모두 게임의 승패에 큰 영향을 끼칠 뿐 아니라 치팅 유저로 인해 공정한 게임 유저들의 이탈이 가속화될 수 있다. 따라서 게임사는 Third party제품의 클라이언트 단 탐지 모델인 게임 보안 솔루션을 사용하여 치팅 피해를 최소화하기 위해 노력하고 있으나, Third party인 보안 솔루션의 경우, 오 탐지 문제로 인해 게임 자원 접근 및 보호 영역이 매우 제한 적일 뿐 아니라, 일반적으로 게임 실행 후 보안 솔루션이 실행되는 구조이기 때문에 이를 악용하여, 보안 솔루션 실행 전 Dll인젝

션[14] 및 Code인젝션[15]을 통해 게임 코드 변조 및 중요 함수를 후킹 하는 방식으로 치팅 프로그램이 배포되고 있다.

게임 보안 솔루션의 경우, 게임 내 공정한 환경 유지를 위해 유저 컴퓨터에서 실행 중인 치팅 프로그램을 최대한 빠르게 탐지하여 게임을 종료하는 임무를 수행하는데, 이는 치팅 프로그램 제작자에게 보안 솔루션의 탐지 기능 우회 및 분석할 수 있는 공격 지점을 알려주며, 또한 새로 제작한 치팅 프로그램의 탐지 여부를 쉽게 확인하여, 치팅 프로그램을 빠르게 배포할 수 있는 환경을 제공해 주게 된다.

이와 같은 클라이언트 단 보안 솔루션의 문제점으로 인해 최근 서버의 누적 데이터 셋을 이용하여 탐지 알고리즘을 구현하는 서버 로그 분석 기법이 많이 연구되고 있지만, 속도에 민감한 FPS 게임 특성 상 게임 내의 이벤트를 세분화하여 상세한 로그를 적재하기 힘들 뿐 아니라, 접근성이 단순한 FPS의 누적 수치 로그만으로는 치팅 프로그램을 사용한 유저와 일반 유저 사이의 유의미한 차이점을 찾기가 힘들다는 문제점이 존재한다.

Table 2의 클라이언트 탐지 모델의 경우, 주로 Third party인 Gameguard[12]나 HackShield[13]와 같은 온라인 게임 보안 솔루션을 사용하여 게임 내 중요 데이터 보호 및 치팅 프로그램의 탐지 임무를 수행하지만, 클라이언트 단에 보안 및 탐지 모듈이 배포되어 쉽게 분석 및 우회할 수 있다는 문제점이 있을 뿐 아니라 많은 오진 사례가 발생할 수 있어 치팅 이슈 및 치팅 대응 기능 개발에 대해 소극적으로 진행하고 있다. 이에 최근 서버 로그 분석을 통해 치팅 유저를 탐지하는 로그 분석 기법이 주목받고 있으나, 저 사양 서버 환경의 특징을 가지는 FPS 게임에 취합된 로그의 정보만으로는 탐지 패턴을 찾기 어려워, FPS 게임의 서버 로그 분석을 통한 탐지는 많이 활용 되지 못하고 있다.

상기한 종래의 클라이언트 단 탐지 문제점을 해결하기 위해, 본 논문에서는 FPS 게임 서버 로그를 이용하여 게임보안 솔루션에서 제공하는 클라이언트 탐지 기능의 오 탐지 문제점을 검증하는 복합형 탐지 모델을 설계하였으며, 전 세계 70여 개국 1억 명의 유저를 보유한 Zepetto[10]사의 포인트블랭크(pointblank)라는 FPS 게임의 데이터 셋을 이용하여 이를 검증하였다.

II. 문헌 연구

FPS 게임의 기존 치팅 탐지 연구 중 Chen 등[4]은 유저의 이동경로를 Isomap을 이용해 데이터의 차원을 축소하고 SVM과 KNN 알고리즘을 이용하는 탐지 방법을 제안하였다. Alayed 등[5]은 치팅 유저 탐지를 위한 최적의 피처를 찾기 위해 다양한 피처의 조합으로 실험을 진행하여 탐지 알고리즘과 피처들의 조합에 따라 높은 순위를 제시하였다.

Yu등[6]은 AimBot 유저를 탐지하기 위한 두 가지 피처를 소개하였다. Han 등[7]의 연구에서는 통계적 분석을 통해 일반 유저와 치팅 유저의 차이를 확인하였고 룰(rule) 기반의 치팅 유저 탐지 방법을 제안하였다. Galli 등[8]은 치팅 유저 탐지를 위한 프레임워크를 제안하였으며 5 가지 머신러닝(Machine Learning) 알고리즘을 적용하여 치팅 유저 탐지 실험을 수행하였으나, Galli 등[8]에 의해 제안된 프레임 워크는 게임 내 캐릭터 변화 수치를 상세히 기록하는 방식으로 게임 서버의 많은 부하를 발생시킬 뿐 아니라 실제 게임 서비스에 적용하지 못한다는 단점이 존재하였다. Park 등[9]은 인공신경망 알고리즘인 MultiLayer Perception을 적용해 치팅 유저 탐지 모델을 생성하여 기존 클라이언트 단 치팅 탐지 솔루션 대비 높은 탐지 율을 확인할 수 있었으나, 치터로 분류된 유저 중 높은 실력을 가

Table 2. Pros and Cons of client side detection model and server log analysis

	Pros	Cons
Client side detection	- File based detection - Anti-reversing - Manipulated memory detection	- Easily bypass the security modules - Limited protection area due to high false positive ratio - Hard to detect new cheating programs - Third party dependency
Server log analysis	- Hard to bypass and analyze the rule of detection - High detection ratio - Real time detection	- Hard to find valid algorithms to tell cheaters from benign users due to lack of information caused by the game characteristic,

진 일반 유저들 역시 치터로 분류 될 수 있는 단점이 존재한다.

위와 같이 오진 사례가 많이 발생할 수 있는 FPS 장르의 특성 상, 기존 서버 로그 분석만으로는 한계가 오진 사례를 보완 할 수 있는 한계가 존재하기에 본 논문에서는 기존 게임사에서 사용하는 기능 및 로그를 활용한 클라이언트 의심 유저 탐지 모델과 이를 서버 로그 분석을 이용하여 검증하는 복합형 탐지 모델을 제안한다.

III. 제안하는 방법론

본 논문에서는 위와 같은 종례의 탐지 모델의 문제점인 오 탐지 사례로 인해 적극적으로 활용하지 못하였던 클라이언트 단 탐지 기능을 개선하고 게임 내 보호 영역의 확장이 가능하도록 클라이언트 로그와 서버 로그의 분석을 융합한 Fig 1과 같은 복합형 탐지 모델을 제안한다.

Fig 1은 게임 보안 솔루션의 클라이언트 탐지 기능으로 의심 유저를 추출한 후, 서버의 누적된 로그를 분석하여 Benign유저의 그룹을 추출하여 오 탐지 사례를 최소화 하고 치팅 유저에 대해서만 서버에서 제재를 진행하는 복합형 프레임워크의 동작 순서도이다.

위와 같은 탐지 방식은 오 탐지 사례로 인해 한정된 영역만을 보호하던 보안 솔루션 기능의 개선 및 적극적으로 보호 영역을 확장할 수 있을 뿐 아니라, 최종 서버 단에서 치팅 유저를 탐지하기 때문에 종례의 클라이언트 탐지의 단점인 분석 및 공격 지점을 최소화 할 수 있는 긍정적인 효과를 볼 수 있다.

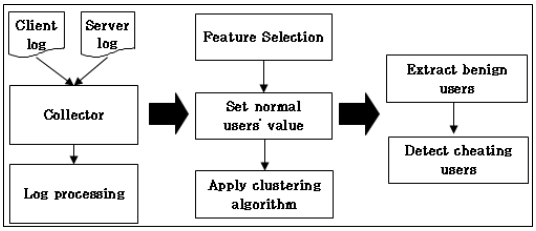

Fig 1. Hybrid framework for FPS games

3.1 치팅 프로그램 동작 방식 분석

포인트블랭크에서 사용하는 클라이언트 단 게임 보안 솔루션의 기능 개선을 위해 이를 분석 한 결과 Table 3과 같이 보안 솔루션에서 제공하는 대표 기능은 다음과 같다.

게임 프로세스의 중요 메모리 위/변조를 감지하는 Memory manipulation 탐지 기능, 그래픽 함수 및 시스템 함수의 후킹 여/부를 검사하는 API hooking 탐지 기능, 게임 실행 시 인젝션 된 파일의 정보(파일 hash값 혹은 메모리 패턴)를 확인하여 탐지하는 File/Memory pattern 탐지 기능이 있으며 위 기능들은 모두 이상 행위 탐지 후 바로 게임 프로세스가 종료 되어 추가적인 치팅 행위를 진행하지 못하도록 하는 특징을 가지고 있다.

위 기능들과는 다르게 탐지 후 모니터링 시스템으로 탐지 로그만을 전송하는 기능인 Monitoring blacklist file은 게임 실행 전 실행된 목록을 레지스트리 및 프리패치파일을 통해 확인하여 치팅 프로그램의 사용 여부를 확인하는 기능이다.

위와 같은 다양한 게임 보안 솔루션의 기능 중 효과적인 치팅 탐지 기능을 활용하기 위해 포인트블랭크의 대표 치팅 프로그램 7종[11]을 분석하였으며, Table 4와 같이 인젝션 방식에 따른 2가지 패턴인 Dll 및 Code 인젝션과 치팅 기능 구현 방식에 따른 2가지 패턴인 D3d Inline Hook(D3d-IH), D3d Table Hook(D3d-TH)을 확인할 수 있었다.

Dll 인젝션[14]이란 가장 많이 사용하는 치팅 프로그램 인젝션 방식으로, 인젝터를 이용하여, 게임 프로세스에 Dll을 삽입하는 방식이다. 삽입된 Dll은 게임 내의 중요 데이터 변조 혹은 코드 실행 흐름을

Table 3. Overview of online game security solution in PointBlank

Feature	Description
Memory manipulation detection	It detects unknown memory manipulation in a game process **Action:** Process termination
D3d pattern detection engine	It protects system and graphic APIs from being hooked **Action:** Process termination
File/Memory (Signature) pattern detection	It is the blacklist file and memory pattern detection within running game process **Action:** Process termination
Registry pattern detection engine	It figures blacklist tools which ran before a game starts by searching registry and prefetch files **Action:** Only recording detection log information

Table 4. A list of cheating programs

Name Region	Injection type	Attack type	Release date
DCN-vip Thailand	DLL	D3d-IH	15/03/06
SnapzV Singapore	DLL	D3d-TH	15/02/16
NewPKL Indonesia	DLL	D3d-IH	15/05/27
WC-load Brazil	DLL	D3d-IH	14/03/19
MPGH Brazil	Code	D3d-TH	14/09/12
Haxreborn Brazil	DLL	D3d-IH	14/07/27
PointBad Brazil	Code &DLL	D3d-IH	14/10/01

변경하여, 유저가 원하는 방식으로 게임을 동작하게 한다. 일반적으로 보안 솔루션 실행 전 종료 되는 인젝터는 감지가 어려우나, 삽입 된 Dll프로그램은 로드된 파일 리스트를 검사하여, 시그니처(signature)방식의 빠른 파일 탐지가 가능하다.

Code 인젝션[15]은 실행 가능한 코드를 게임 프로세스의 힙(heap)영역 등에 복사하여, 게임 실행 코드의 흐름을 변경한다는 특징이 있다.

일반적으로, Code인젝션의 경우 로더(loader)라는 실행파일이 최신 공격 코드를 서버로부터 내려 받아, 게임 프로세스 내의 힙 영역에 바로 실행 가능한 코드를 덮어씌우는 방식으로 구현된다.

로더 파일 역시, 보안 솔루션 실행 전 종료 되어, 기존 게임 보안 솔루션에서는 탐지가 어렵다는 단점이 존재한다.

치팅 기능 구현 방식에 따른 방식인 Direct3d Inline Hook과 Direct3d Table Hook은 변조 대상이 코드인지 함수 테이블(V-table)인지에 따라 서로 다를 뿐 동작 방식은 동일하다.

위와 같이 총 7종의 치팅 프로그램을 통해 추출한 패턴을 통해 Table 3의 게임 보안 솔루션의 기능 중 이를 적절히 탐지 할 수 있는 2가지 탐지 기능인 Registry Pattern Engine(RPE)과 Direct3d Pattern Engine(DPE)를 선택하였다.

3.1.1 Registry Pattern Engine

RPE란 치팅 프로그램이 Code나 Dll인젝션을 수행하기 위해 사용하는 인젝터나 로더와 같은 실행 파일을 탐지하기 위한 기능으로 게임 실행 전 종료되어 사용 흔적을 찾기 어려웠던 기존의 문제점을 보완할 수 있는 기능이다.

Fig 2의 기능 블록도와 같이 RPE란 임계시간 전까지의 레지스트리와 프리패치 파일에서 실행 프로세스 목록을 취합하는 Registry pattern engine과 취합한 정보와 치팅 프로그램을 비교하는 Blacklist 그리고 탐지 정보를 전송받는 모니터링 시스템(monitoring system)으로 구성되어 있다.

RPE 기능은 게임 실행 전 치팅 프로그램을 실행한 유저를 탐지하는 기능으로 의심 유저 분류에는 효과적이나, 게임 실행 전 치팅 프로그램 실행 목록만으로 수많은 오진 사례가 발생할 수 있어 탐지 정보만을 모니터링 시스템으로 전송만 할 뿐 이를 활용하여 적극적인 제재를 진행하지 못할 뿐 아니라, 기존 다른 기능과 같이 탐지 후 바로 게임 종료를 진행할 경우, 치팅 제작자가 손쉽게 분석 지점을 확인하여 기능 분석을 통해 레지스트리 및 프리패치 파일의 삭제 등의 간단한 작업을 통해 쉽게 우회가 될 수 있다는 단점이 존재한다.

Fig 3과 Fig 4는 2014년 02월 14일부터 02월 27일까지 13일간의 클라이언트 보안 탐지 기능의 시그니처 탐지 로그와 RPE 탐지 로그를 비교한 자료이다.

Fig 3에서 보여 지듯이 기존 시그니처 탐지 방식

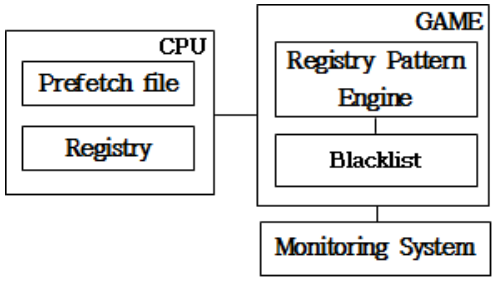

Fig. 2. Structural diagram of RPE

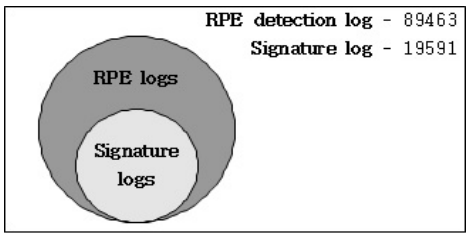

Fig. 3. RPE logs VS Signature's one

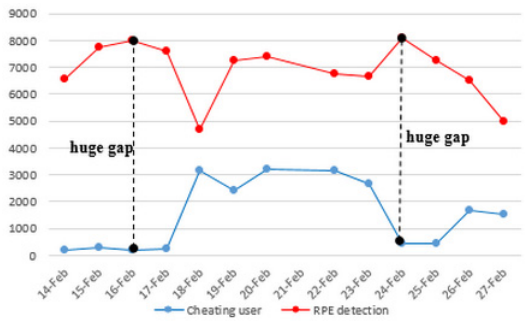

Fig. 4. Detection logs per day

대비 RPE 기능의 높은 탐지 율을 확인 할 수 있었다.

Fig 4는 새로운 버전의 치팅 프로그램이 배포되었거나 시그니처 탐지 방식에서 치팅 프로그램을 효율적으로 탐지하지 못한 날에도 RPE 탐지 기능에서는 높은 탐지 율을 보여 치팅 프로그램에 적절한 대응이 가능함을 확인할 수 있다.

3.1.2 Direct3d Pattern Engine

DPE는 게임 내의 중요 그래픽 함수 및 시스템 함수에 후킹 여부를 확인하여 탐지하는 Direct3d Pattern Engine과 이를 전송 받는 모니터링 시스템으로 구성된다.

Fig 5의 DPE는 RPE와 달리 탐지 후 게임을 바로 종료하는 탐지 방식이나 DPE 역시 오 탐지 사례로 인해 매우 한정된 영역(일부 D3d 함수 및 System 함수)만을 보호하는 것을 확인하였으며, 포인트 블랭크 치팅 프로그램은 클라이언트 단 보안 솔루션에서 보호하지 않는 영역의 코드 변조 및 D3d 테이블을 후킹 하여 여러 가지 치팅 기능을 구현하여 효과적으로 치팅 프로그램에 대해 대응을 하지 못하고 있다.

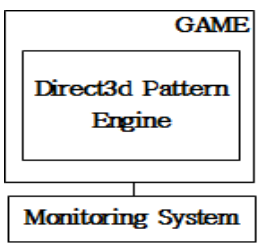

Fig 5. Structural diagram of DPE

3.2 실험 제약 사항

상기한 부분과 같이 분석한 7종의 치팅 프로그램의 공격 방식은 클라이언트 보안 솔루션만으로는 오 탐지 문제 및 기능 우회로 인해 적절한 대응을 하지 못하는 것을 확인할 수 있다.

이에 본 논문에서 명시한 클라이언트 단 보안 기능의 오진 여부를 게임 서버 로그 분석을 통해 검증하는 방식의 복합형 프레임워크를 제안하며, DPE 기능은 탐지 후 바로 게임이 종료 되어 게임 서버에 올바른 누적 수치가 기록 되지 않는 다는 제약사항으로 기존에 효율적으로 사용하지 못하였던 RPE 탐지 기능의 개선에 초점을 맞춘 실험만을 진행하였다.

실험에 사용한 RPE 로그는 Table 5와 같은 형태로 이루어지며, 유저 정보를 확인할 수 있는 ID 항목, 탐지한 유저의 IP와 Mac 주소를 확인할 수 있는 Address 항목, 보안 솔루션의 세부 탐지 코드 번호를 보여주는 Code 항목, 탐지된 파일의 시스템 경로를 기록해주는 IMG Path와 탐지된 시간을 확인할 수 있는 Time 항목으로 구성 되어 있다.

Table 5. Log format of RPE

Attribute	Description
ID	Account information of detected user.
Address	IP & MAC address.
Code	Specific codes for each detection feature.
IMG Path	File path information.
Time	Detection date information.

3.3 RPE 탐지 오진 사례 분석

RPE 기능에 탐지되어 모니터링 시스템에 기록된 로그의 오진 사례를 확인한 결과, 상기한 7종의 대표 치팅 프로그램은 Third party 보안 제품의 업데이트 여부를 확인하여 치팅 프로그램 사용 유저에게 Fig 6와 같은 메시지를 보여준다.

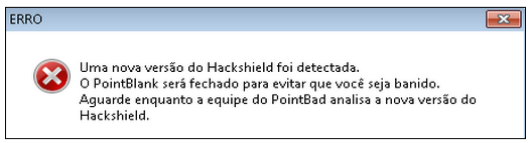

Fig. 6. notification of the current status for cheat programs

이를 통해 치팅 유저는 현재 치팅 프로그램의 In-game에서의 탐지 여부를 확인할 수 있으며,

로더를 실행하였으나, 치팅을 하지 않는 명백한 유저의 그룹이 존재하게 된다.

기존 RPE 탐지 로그는 이를 적절하게 탐지해 낼 수 없어, 오진 사례가 존재하였으며, 이에 게임사에서는 기존 시그니처 탐지보다 높은 탐지 율을 가지고 있는 RPE 탐지 기능을 적절히 활용하지 못하고 있었다.

본 논문에서는 위 오 탐지 사례에 포함하는 그룹을 추출하기 위해 RPE 탐지 기능에 의해 의심 유저로 분류된 유저들의 서버 누적 로그를 활용하여 실험을 진행하였다.

3.4 로그 수집 및 피처 선택(feature selection)

실험에 사용한 데이터는 2014년 02월 14일부터 02월 27일까지 13일간의 RPE탐지로그와 서버 로그를 사용하였으며, 유저마다 게임 시간이 서로 상이하기 때문에 하루 단위의 집계 데이터를 생성하여 실험을 진행하였다.

상기한 RPE 탐지 로그와 일치하는 13일간의 서버 로그를 확인한 결과 Table 6와 같은 데이터 셋을 취합할 수 있었다.

올바른 피처 선택을 위해, Table 1에서 명시한 기존 치팅 프로그램의 기능에 따른 게임 내 효과를 Table 7과 같이 분류하였으며 Kill, Headshot, Fast kill, Win, Short play와 같이 치팅 기능 사용 시 일반 유저에 비해 높은 수치를 갖는 항목을 선정하여 총 5가지 피처를 생성하였다.

Table 7의 KillRatio는 하루 평균 플레이 시간 중 유저가 적군을 죽인 비율이며, 치팅 프로그램을 사용한 유저는 일반 유저보다 높은 수치를 가진다. Headshot ratio는 하루 평균 플레이 시간 중 유저가 헤드 샷으로 적군을 죽인 비율이며, 치팅 기능 중 Aimbot을 사용한 유저는 일반 유저보다 해당 수치가 높게 나온다. Winratio는 총 플레이 횟수 중 승리한 비율을 보여주며, 치팅 프로그램을 사용한 유저

Table 6. Total game data set

Type	Count
Benign user	2,234,441
RPE detection user	89,463
Total	2,323,904

Table 7. Expected effectiveness of cheat feature.

	Kill	Head shot	Fast kill	Win	Short Play Time
Wallhack	●			●	
ESP	●			●	●
Aimbot	●	●	●	●	●
Flyhack	●			●	
Teleport	●		●	●	●
Infinite Ammo	●			●	
Infinite health	●			●	

는 일반 유저보다 높은 수치를 갖는다. Fastkill ratio는 총 플레이 시간 중 적군 사살에 성공한 비율로, 치팅 프로그램을 사용한 유저는 일반 유저보다 높은 수치를 갖는다.

생성한 5가지 피처의 검증을 위해 Random sampling selection 방식을 이용하여 50:50의 비율로 일반 유저(Benign user)의 로그를 취합 하였으며, 두 그룹의 차이를 수치 화 하기 위해 Information Gain Attributes를 이용하여 비교해 보았다. Information Gain Attribute란 Entropy 측정을 이용한 방식으로 Entropy란 임의의 표본 집합으로부터 그 특성을 나타내기 위해 사용한다. 임의의 서로 다른 값을 포함한 집합 Y의 Entropy 계산은 다음과 같다.

$$H(Y) = -\sum_{y \in Y} p(y)\log_2(p(y))$$

여기서, p(y)는 랜덤 변수 Y에 대한 주변 확률 밀도 함수(marginal probability density function)이며, Y의 엔트로피 값이 이전 X 피처에 의해 계산된 Y엔트로피 값 보다 작다면 X, Y간에는 서로 관계가 존재함을 알 수 있다. X에 의해 관측된 엔트로피 Y는 다음과 같다.

$$H(Y/X) = -\sum_{x \in X} p(x) \sum_{y \in Y} p(y/x)\log_2(p(y/x))$$

여기서 p(y/x)는 주어진 x의 조건부 확률 y로, 데이터 집합 S의 엔트로피 값을 기준으로 X에 대한 Y엔트로피 값의 측정을 정의 할 수 있으며, 이를

IG(Information Gain)라 한다.

$$IG = H(Y) - H(Y/X) = H(X) - H(X/Y)$$

IG를 이용하여, [Table 7]과 같이 두 그룹의 각 피처에 대한 측정 값(Attribute evaluation)을 확인하였으며, 값이 0에 가까울수록 두 그룹의 차이가 불명확하다는 것이다.

Table 8의 KillRatio는 하루 평균 플레이 시간 중 유저가 적군을 죽인 비율이며, 치팅 프로그램을 사용한 유저는 일반 유저보다 높은 수치를 가진다. Headshot ratio는 하루 평균 플레이 시간 중 유저가 헤드 샷으로 적군을 죽인 비율이며, 치팅 기능 중 Aimbot을 사용한 유저는 일반 유저보다 해당 수치가 높게 나온다. Winratio는 총 플레이 횟수 중 승리한 비율을 보여주며, 치팅 프로그램을 사용한 유저는 일반 유저보다 높은 수치를 갖는다. Fastkill ratio는 총 플레이 시간 중 적군 사살에 성공한 비율로, 치팅 프로그램을 사용한 유저는 일반 유저보다 높은 수치를 갖는다.

Table 8의 Attribute evaluation 결과, 두 그룹을 분류하는데 가장 작은 차이를 보이는 Average of Game time은 두 그룹을 분류하는데 왜곡된 결과를 초래할 수 있어 이를 제외한 4가지의 피처만을 선택하였다.

Fig 7은 RPE 기능에 탐지된 그룹과 일반 유저의 박스플롯(box plot)으로, RPE 기능에 의해 탐지된 유저는 Headshot ratio, Kill ratio, Win ratio와 Fast kill ratio 피처에서 상대적으로 일반 유저 보다 높은 것을 확인할 수 있었다.

Table 8. Attribute evaluation

Feature	Formula	Attribute evaluation
Kill ratio	$\frac{Kill}{Kill + Death}$	0.163
Head shot ratio	$\frac{Headshot}{Kill}$	0.072
Win ratio	$\frac{Win}{Win + Lose}$	0.046
Fast kill ratio	$\frac{Total\ Kill}{Total\ playtime}$	0.137
Game time avg.	$\frac{Game\ playtime}{Nm\ of\ Game\ play}$	0.027

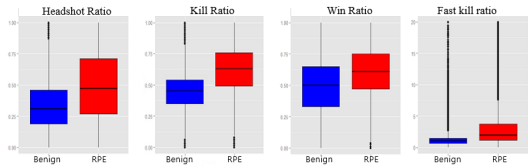

Fig. 7. Boxplot of each feature

그러나 위 4개의 피처는 오직 유저의 누적 수치만을 고려한 탐지로 상기한 RPE 기능의 오 탐지 사례만을 올바르게 추출 할 수 없다는 단점이 있다. 이에 RPE 오진 사례를 올바르게 탐지 할 수 있는 Detection ratio를 키(key)피처로 추가하여 실험을 진행하였다.

$$Detection\ ratio = \frac{Signature\ \mathrm{detection}}{RPE\ \mathrm{detection} + Signature\ \mathrm{detection}}$$

IV. 실 험

위 5가지 피처를 이용하여 13일간의 RPE 탐지 로그 중 일반 유저로 분류되는 그룹을 추출하기 위하여, Unsupervised learning 방식 중 가장 활발히 사용하는 K-means 알고리즘을 사용하였다.

K-means 알고리즘이란 n개의 d-차원 데이터 객체 (x1, x2, …, xn) 집합이 주어졌을 때, n 개의데이터 객체들을 각 집합 내 객체 간 응집도를 최대로 하는 k 개의 집합 S로 분할한다. 다시 말해, μi가 집합 Si의 중심점이라 할 때, 다음과 같은 수식으로 표현 가능하며, 각 집합 내 객체 간 거리의 제곱 합을 최소로 하는 집합 S를 찾는 것이 이 알고리즘의 목표다

$$V = \sum_{i=1}^{k} \sum_{j \in S_i} \| x_j - \mu_i \|^2$$

K-means알고리즘은 최초 파라미터 K의 값을 선택하며, K값에 따라 데이터의 클러스터 결과 값이 크게 달라질 수 있어 정확한 K의 값을 찾아내는 것이 매우 중요하다.

본 논문에서는 Euclidean distance를 이용한 K-means 알고리즘의 파라미터 K를 2부터 10까지 변경하여 실험을 진행하였으며, Euclidean distance는 다음과 같이 정의 된다.

$P=(p_1,p_2,p_3.....p_n)$, $Q=(q_1,q_2,q_3.....q_n)$ 일 때,

$$\sqrt{(p_1-q_1)^2+(p_2-q_2)^2+...+(p_n-q_n)^2}$$

$$Distance(p,q)=\sqrt{\sum_{i=1}^{n}(p_i-q_i)^2}$$

Table 9과 같이 파라미터 K의 값이 4일 때, RPE 오진 사례에서 발생할 수 있는 유저의 그룹으로 추정 된 C1을 추출할 수 있었다.

C1은 전체 RPE 탐지 유저 중 5.7%의 Instance ratio를 가지며, 각 피처의 중앙값이 다른 그룹과 달리 B(Benign)의 평균과 매우 유사한 것을 확인할 수 있었다. 해당 유저들은 게임 보안 솔루션에 치팅 프로그램이 많이 탐지한 날에 속한 그룹으로, RPE오진 사례와 같이 실제 치팅 프로그램을 사용하지 않은 그룹과도 일치하는 결과인 것을 알 수 있다.

위와 같이 치팅 유저로 분류된 클라이언트 탐지 로그에서, 클러스터링 알고리즘을 적용하여 오진 사례와 일치하며 게임 내 균형을 무너뜨리지 않는 그룹을 추출할 수 있음을 증명하였다. 다만 이와 같이 K-means 클러스터링 알고리즘을 실제 게임 환경에서 사용하기 위해서는 올바른 K 파라미터 값을 지정하여야 하며, Fig 8과 같이 파라미터 K의 값이 4, 7, 8, 9, 10일 때에만 나타나는 그룹 군인 C1은 K 값이 커질수록 최적화의 결과가 전역 최적값(global optimum)이 아닌 지역 최적값(local optimum)으로 빠질 가능성이 있기 때문에 가장 작은 K 값을 최고 성능(best performance)파라미터 값으로 선정하여 사용할 수 있다.

Table 9. Result with the K parameter 4

	C0	C1	C2	C3	B
Win ratio	0.60	0.47	0.62	0.62	0.47
Kill ratio	0.62	0.47	0.65	0.64	0.43
Headshot ratio	0.49	0.36	0.52	0.52	0.33
Fast kill ratio	2.99	1.59	3.18	3.42	1.24
Detection ratio	23.16	40.24	3.72	30.0	-
Instance ratio	20.3 (%)	5.7 (%)	49.3 (%)	24.6 (%)	-

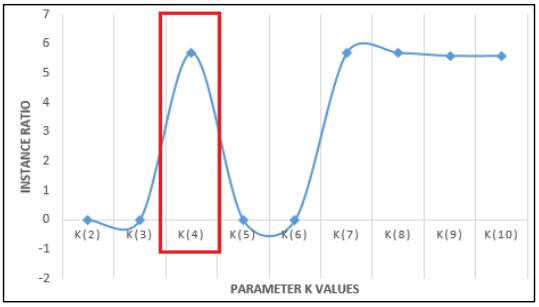

Fig. 8. Best performance value for K

V. 실험 결과

본 논문에서 제안한 복합형 프레임 워크를 사용하여 치팅 유저 탐지 시, Table 10과 같이 기존 시그니처 탐지 대비 64,772(약 330%)명을 추가 탐지할 수 있을 뿐 아니라, 기존 오 탐지 제약사항으로 인해 활용하지 못하였던 RPE 탐지 로그 중 Benign 유저로 분류된 5.7%의 오 탐지 유저를 분류할 수 있어 기존 Signature 탐지 대비 정밀도와 정확도의 높은 향상을 보이는 것을 확인할 수 있다.

본 논문에서 사용한 데이터 셋은 모두 실제 게임 환경을 통해 취합한 데이터 셋으로 치팅 사용자로 분류 된 올바른 정답지가 존재하지 않아, Supervised learning의 Classification 알고리즘이 아닌 Unsupervised learning의 Clustering 알고리즘을 사용하여 탐지 모델의 이진 정확도 수치인 Precision(정밀도), Recall(재현율), Accuracy(정확도)와 같은 평가 지표를 확인할 수 없다는 제약사항이 존재한다.

Table 10. Evaluation of cheater detection

Type	Benign user	Cheater
RPE detection	Not classifiable	89,463
Signature detection	Not classifiable	19,591
Proposed detection	5,100	84,363

VI. 결 론

온라인 게임의 치팅 탐지 방식은 크게 클라이언트 단 탐지 모델과 서버 단 탐지 모델로 분류 되며, 클라이언트 단 보안 솔루션은 다양한 환경에 탐지 모듈

이 배포되어 오 탐지 사례가 발생 할 수 있어 한정된 영역만을 보호 하거나 탐지 기능을 소극적으로 운용하여 끊임없이 진화하는 치팅 프로그램에 대해서 적절한 대응을 할 수 없으며, FPS서버 단 탐지 방식은 저 사양 서버 환경의 문제로 인해 접근성이 용이한 누적 수치만을 기록하여, 올바른 탐지 알고리즘을 적용하기 힘들 뿐 아니라, 누적 수치만을 고려한 탐지 방식에는 높은 실력을 가진 일반 유저 역시 치팅 유저로 분류될 수 있다는 단점이 존재한다.

이에 우리는 FPS게임의 효율적인 치팅 프로그램 탐지를 위해 실험에 사용한 게임의 대표 치팅 프로그램 7종을 선정하여 탐지 가능한 패턴을 추출하였고, 해당 게임사의 게임 보안 솔루션에서 제공하는 기능들 중 이를 탐지 할 수 있는 효율적인 기능 2가지(RPE와 DPE)를 선정하였으며, 기존 클라이언트 탐지 모델의 기능 개선 및 보호 영역 확장을 위해, 클라이언트 탐지 모듈의 탐지 로그와 서버 로그 분석을 융합하여 오 탐지 사례를 검증하는 복합형 프레임워크를 제안하였다.

논문에서 제안하는 복합형 탐지 방식은 클라이언트의 프로그램화 된 기능을 통해 이상 행위 유저를 1차로 분류하고, 이 중 오진 사례만을 게임 서버의 누적 수치를 분석하여 검증 및 추출하는 2차 작업을 거쳐 최적의 탐지 방식을 구현 할 수 있으며, 기존 클라이언트 탐지 모델의 단점과 FPS 서버 로그 분석의 단점을 효과적으로 대응 할 수 있음을 보였다.

실험에는 일반 유저와 RPE 기능에 의해 치팅 유저로 의심 되는 유저의 그룹을 적절하게 분류하기 위해 IG알고리즘을 사용하였으며 이를 기반으로 실험에 적합한 네 가지 피처 선정과, 오진사례를 분석하여 정의한 하나의 키 피처를 이용하여 K-means 클러스터링 알고리즘 적용하였다.

이에 오 탐지 사례로 추정되는 5.7%의 비율을 가지는 유저의 그룹 군을 추출할 수 있었으며, 실험에 사용한 데이터 셋인 2014년 02월 14일부터 02월 27일까지 13일간의 탐지 목록 중 Benign 유저로 분류된 5.7%를 제외하여 탐지 시, 기존 13일간의 시그니처 탐지 대비 64,772명의 유저를 추가로 제재 가능하며, 이는 기존 시그니처 탐지 대비 약 330%의 높은 탐지가 가능할 뿐 아니라, Table 2에서 명시한 종례의 클라이언트 단 문제점과 FPS 서버 로그 분석의 단점을 보완할 수 있다.

다만 실험에 사용한 게임 보안 솔루션의 기능인 RPE 탐지 기능은 블랙리스트를 활용한 탐지 방식으로 알려진 치팅 프로그램에 대해서만 감지가 가능하며, 레지스트리 삭제 등과 같은 간단한 작업만으로도 기능의 우회가 가능하다. 이에 향후 견고한 복합형 프레임워크를 구현하기 위해서는 치팅 프로그램 기능 구현 방식에 따라 탐지가 가능한 DPE의 탐지 영역을 확장한 후, 모니터링 시스템으로 탐지 로그를 전송하여 취합한 로그를 활용하여 치팅 유저 및 일반 유저를 검증하는 연구가 필요하다.

References

〔1〕 Woo, Jiyoung, and Huy Kang Kim, “Survey and research direction on online game security.” Proceedings of the Workshop at SIGGRAPH Asia, pp. 19-25, Nov. 2012

〔2〕 Ah Reum Kang, Jiyong Woo, Juyong Park and Huy Kang Kim, “Online game bot detection based on party-play log analysis.” Proceedings of Computers & Mathematics with Applications, Vol.65, Issue.9, pp. 1384-1395, May. 2013

〔3〕 M.van Kesteren, L.Jurriaan and G.Franc, “A step in the right direction: Botdetection in MMORPGs using movement analysis.” Proceedings of the 21st Belgian-Dutch Conference on Artificial Intelligence(BNAIC 2009). 2009

〔4〕 Chen, Kuan-Ta, Hsing-Kuo Kenneth Pao and Chang Hong-Chung, “Game bot identification based on manifold learning.” Proceedings of the 7th ACM SIGCOMM Workshop on Network and System Support for Games. ACM, pp. 21-26, Oct. 2008

〔5〕 H.Alayed, F.Frangoudes and C.Neuman “Behavioral-based cheating detection in online first person shooters using machine learning techniques.” Proceedings of Computational Intelligence in Games (CIG), 2013 IEEE Conference on. IEEE, pp. 1-8, Aug. 2013

〔6〕 Yu, Su-Yang, N.Hammerla, J.Yan and P.Andras, “A statistical aimbot detection

method for online FPS games." Proceedings of Neural Networks (IJCNN), The 2012 International Joint Conference on. IEEE, pp. 1-8, June. 2012
〔7〕 Han, Mee Lan, Jung Kyu Park and Huy Kang Kim, "Online Game Bot Detection in FPS Game." Proceedings of the 18th Asia Pacific Symposium on Intelligent and Evolutionary Systems-Volume 2. Springer International Publishing, pp. 479-491, 2015
〔8〕 L.Galli, D.Lioacono, L.Cardamone and P.L. Lanzi, "A cheating detection framework for unreal tournament III: a machine learning approach," Proceedings of Computational Intelligence and Games (CIG), 2011 IEEE, pp, 266-272, Aug. 2011
〔9〕 Han, Mee Lan, Jung Kyu Park and Huy Kang Kim, "A study of Cheater Detection in FPS Game by using User Log Analysis." Proceedings of Jonornal of Korea Game Society, 15(3), pp.177-188, June. 2015
〔10〕 Main website of Zepetto,http://www.zepetto.com/main.do
〔11〕 Cheating program sites,
1) http://www.dcnmode.com/
2) http://www.nyit-nyit.net/
3) http://www.komunitas-pekalongan.blogspot.com
4) http://www.webcheats.com.br/
5) http://www.mpgh.net/
6) http://www.haxreborn.com
7) http://www.pointbad.net/
〔12〕 Main web site of GameGuard. http://www.nprotect.com/index.html
〔13〕 Main web site of HackSheild. http://hackshield.ahnlab.com/hs/site/ko/main/main.do
〔14〕 https://en.wikipedia.org/wiki/DLLinjection
〔15〕 https://en.wikipedia.org/wiki/Code_injection

〈저자소개〉

김 선 민 (Seon Min Kim) 정회원
2010년 08월: 한남대학교 컴퓨터공학과 학사 졸업
2013년 9월~현재: 고려대학교 정보보호학과 석사과정
2010년 03월~현재: AhnLab 게임 보안 제품 Security Architect
〈관심분야〉 온라인게임 보안, 빅 데이터 분석, 어플리케이션 취약점 분석

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