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.2015.25.6.1361

클래시 오브 클랜 오토 프로그램의 악성 행위 분석을 통한 모바일 게임 보안 위협에 관한 연구

허 건 일,[1†] 허 청 일,[2] 김 휘 강[1‡]
[1]고려대학교 정보호호대학원, [2]한국산업기술보호협회

A Study on Mobile Game Security Threats by Analyzing Malicious Behavior of Auto Program of Clash of Clans

Geon Il Heo,[1†] Cheong Il Heo,[2] Huy Kang Kim[1‡]
[1]Graduate School of Information Security, Korea University
[2]Korean Association for Industrial Technology Security

요 약

모바일 게임 시장 규모와 인구가 증가하고 동시에 모바일 게임의 생명주기가 크게 길어지면서 PC 기반 온라인 게임에서 나타났던 오토 프로그램 문제가 모바일 게임에서 그대로 재현되고 있다. 사용자들은 오토 프로그램 실행을 위해 안티 바이러스 프로그램의 경고를 무시하거나 심지어 안티 바이러스 프로그램을 삭제하기 때문에, 오토 프로그램이 게임 관련 기능 외에 악성 행위를 수행할 경우 게임 이용자들은 위협에 무방비로 노출될 수 있다.

본 논문에서는 장기간 대규모의 사용자를 확보하고 있는 대표적인 모바일 게임인 클래시 오브 클랜의 오토 프로그램 7종을 대상으로 악성 행위 유무를 분석하고, 이를 바탕으로 향후 발생 가능한 보안 위협과 대응방안을 제시하였다. 높은 인기를 가진 특정 모바일 게임의 오토 프로그램들을 일괄 분석함으로써 개발 플랫폼, 동작 방식 등 오토 프로그램의 최신 동향을 파악하였고, 향후 오토 프로그램의 변화 양상 예측 및 잠재적 위협을 사전에 차단할 수 있는 방안을 제시하였다.

ABSTRACT

Recently, the size of the mobile game market and the number of mobile game users are growing. Also, as the mobile game's life cycle is increasing at the same time, auto program issue reappears which has been appeared in PC online games. Gamers usually tend to ignore warning messages from antivirus programs and even worse they delete antivirus program to execute auto programs. Therefore, mobile game users are easily compromised if the auto program performs malicious behaviors not only for the original features.

In this paper, we analyze whether seven auto programs of "clash of clans" which has a lot more users for a long time perform malicious behaviors or not. We forecast the possible security threats in near future and proposed countermeasures based on this analysis. By analyzing auto programs of highly popular mobile game of today, we can acquire the knowledge on auto program's recent trend such as their development platform, operating mode, etc. This analysis will help security analysts predict auto program's evolving trends and block potential threats in advance.

Keywords: auto program, macro, mobile game, malicious behavior

Received(06 .24. 2015), Modified(1st: 09. 07. 2015 2nd: 10. 12. 2015), Accepted(10. 13. 2015)

† 주저자, aza837@korea.ac.kr
‡ 교신저자, cenda@korea.ac.kr(Corresponding author)

I. 서 론

스마트 기기의 보급 확대와 하드웨어 성능이 향상됨에 따라 모바일게임 시장 규모 및 게임 인구도 가파르게 증가하고 있다. 한국콘텐츠진흥원의 2014 대한민국 게임백서에 따르면, 2013년 모바일 게임 시장 규모는 2012년 대비 190.6% 증가하였고, 2013년 모바일게임 인구는 2012년 대비 107.7% 증가하였다[1].

아울러 모바일 게임의 생명주기도 매우 길어졌다. Table 1.은 2015년 6월 18일 기준 구글플레이와 애플 앱스토어를 종합한 매출순위를 나타낸다. 1위 레이븐을 제외한 모든 게임이 출시된 지 최소 1년이 경과되었고, 클래시 오브 클랜의 경우 출시된 지 무려 약 3년 가까이 되었다. 모바일 게임 흥행이 장기화되고 게임 이용자 수가 과거에 비해 크게 증가했다[2].

Table 1. Sales Ranking of Mobile Game

Sales Ranking	Game Title	Registration Date
1st	Raven	2015.02.20
2nd	Clash of Clans	2012.08.02
3rd	Seven Knights	2014.03.06
4th	Modoo Marble	2013.06.10
5th	Training Monster	2013.08.12
6th	Anipang2	2014.01.13
7th	Hero	2014.11.16
8th	Summoner's War	2014.04.18
9th	Hearth Stone	2014.04.16
10th	Blade	2014.04.20

이러한 인프라 형성으로 인해 PC 기반 온라인 게임에서 나타났던 보안 문제가 모바일 게임에서도 재현되고 있다. 전통적인 PC 기반 온라인 게임의 경우 오토 프로그램, 핵, 사설 서버, 그리고 아이템 현금거래, 계정 도용 등의 보안 문제가 나타났고[3,4,5], 모바일 게임에서도 이와 유사하게 계정 도용, 앱 리패키징, 결제 부정 등의 보안 문제가 나타나고 있다[6]. 그리고 모바일 게임의 생명 주기가 길어진 시점부터는 안드로이드 에뮬레이터와 오토 프로그램 등을 이용한 소위 개인 단위의 작업장 운영이 가능해지고, 클래시 오브 클랜의 경우 모바일 게임 최초의 사설 서버가 등장하는 등 점점 그 심각성이 커져 가고 있다[7,8].

이에 대한 대응방안으로서 PC 기반 온라인 게임의 보안 문제의 경우 클라이언트 사이드보다 서버 사이드에서의 방법론이 주로 연구되어 왔고[9,10], 모바일 게임의 보안 문제의 경우 주로 앱 리패키징 방지에 관한 다양한 기법들이 연구되어 왔다[11]. 그러나 최근 증가하고 있는 모바일 게임의 오토 프로그램에 대한 연구는 많이 부족한 상황이고 그에 따라 관련된 위협 여부 또한 거의 알려져 있지 않다.

하지만 사용자들이 모바일 게임 관련 오토 프로그램, 핵, 크랙 버전의 앱 등을 접하는 것은 매우 쉬운 상황이다. Fig. 1.에서 볼 수 있듯이 포털 사이트에서 모바일 게임 매크로라는 문자열로 검색한 결과, 연관 검색어와 함께 관련 게시물이 검색 결과 상단에 노출된다.

Fig. 1. Search Result of Mobile Game Macro

게임 크랙이라는 문자열로 검색했을 때에도 Fig. 2.와 같이 유명 모바일 게임들의 크랙 관련 게시물이 검색 결과 상단에 노출된다.

Fig. 2. Search Result of Game Crack

Fig. 3.은 android hack이라는 문자열의 이미지 검색 결과이다. 특정 게임 이름을 지칭하여 검색할 경우, 더욱 다양한 불법 프로그램이 검색된다.

Fig. 3. Image Search Result of android hack

공격자들은 이러한 높은 접근성을 악용하여 모바일 게임의 오토 프로그램, 핵, 크랙 버전의 앱에 악성코드를 은닉시켜 유포할 수 있다. 해당 프로그램을 다운받은 사용자는 대개 이런 프로그램들이 안티 바이러스 프로그램에서 진단될 수 있다고 생각하는 경향이 있기 때문에, 설치 및 사용 과정에서 안티 바이러스 프로그램이 제대로 된 진단을 하더라도 이를 무시하거나 심지어 프로그램을 삭제할 수 있다. 표면적으로는 프로그램의 기능이 정상적으로 동작하고 백그라운드에서 악성 행위를 수행하기 때문에 사용자는 이를 인지하기 어렵고 동시에 감염이 지속되는 기간이 증가하면서 피해수준도 커진다.

현재 대표적인 인기 모바일 게임 중의 하나는 클래시 오브 클랜이다. 일반적으로 생명주기가 몇 개월에 불과한 여타 모바일 게임과 달리 최초 출시일인 2012년 8월을 시작으로 현재까지 약 3년 동안 인기 차트 상위순위에 장기 집권하고 있는 게임이다. 오랜 기간 높은 인기를 누린 만큼 많은 게임 이용자 수를 보유하고 있고 그에 따라 다양한 오토 프로그램이 인터넷을 통해 거래되고 있다. 대부분 유료 형태이나 체험판 제공을 통해 사용자의 구매를 유도하고 있고 관련 포럼도 매우 활성화되어 있다.

이에 따라 본 논문에서는 클래시 오브 클랜 오토 프로그램을 대상으로 악성 행위 유무를 분석하고, 이를 바탕으로 잠재적으로 향후 발생 가능한 모바일 게임 보안 위협과 대응방안을 제시한다.

II. 관련 연구

본 장에서는 안드로이드 게임 부정행위 관련 프로그램의 현황과 오토 프로그램의 악성 행위 유무 분석 시 점검 항목 선정을 위한 선행 연구를 수행한다.

2.1 안드로이드 게임 부정행위 관련 프로그램 현황

2.1.1 안드로이드 에뮬레이터

Fig. 4. Attempt to Connect to Certain Game with Multiple Accounts by Executing Multiple Emulators

Table 2. List of Android Emulators

Application Name	Cost	Latest Version (Release Date)
AMIDuOS	Non-Free	1.0.15.6798 (2015.04)
Andy	Non-Free	43.1 (2015)
BlueStacks	Free	0.9.25.5401 (2015)
Genymotion	Free	2.4.0 (2015.03)
Windroy	Free	2.8.0b (2015)

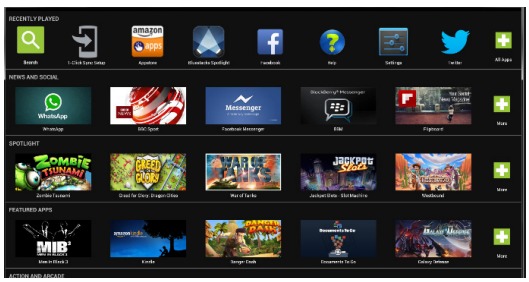


Fig. 5. BlueStacks' User Interface

안드로이드 에뮬레이터는 PC에서 안드로이드 앱을 구동해줄 수 있는 소프트웨어로서 원래는 안드로이드 운영 환경에서의 개발자들의 디버깅을 용이하게 하기 위한 목적이 컸다. 그러나 일부 이용자들이 이를 악용하여 디버깅을 통해 게임 앱의 취약점을 찾거

나, Fig. 4.와 같이 복수의 에뮬레이터를 동시에 실행하여 여러 개의 계정으로 접속, 그리고 매크로를 이용하여 자동으로 게임을 수행함으로써 PC 기반 온라인게임에서의 작업장과 같은 문제를 발생시키고 있다[7]. Table 2.는 안드로이드 에뮬레이터의 현황을 나타내고[12,13,14,15,16], Fig. 5.는 이 중 하나인 BlueStacks의 User Interface를 나타낸다.

Fig. 4.와 같이 복수의 에뮬레이터를 통한 다중 접속을 차단하기 위한 방안으로서 기기 인증 방법이 사용될 수 있으나 게임 이용자의 반발 및 게임 이탈이 발생할 확률이 높기 때문에 조심스러운 접근이 필요하다.

2.1.2 메모리 변조 앱

메모리 변조 앱은 게임 내 특정 값을 메모리 상에서 검색 및 변조하는 프로그램을 말한다. 그 종류를 정확히 모두 파악할 수는 없지만 대표적인 것은 Table 3.과 같다[17,18,19,20].

해당 앱이 실행되기 위해서는 반드시 루팅이 선행되어야 하고, 4개 모두 조작법이 아주 간단하다. 게임 앱을 실행한 후 골드, 특정 아이템 개수 등 변조하고자 하는 항목의 현재 값을 메모리 변조 앱에 입력하고, 이 후 해당 값을 계속 변경시키면서 검색 결과를 좁혀나가는 방식이다. Fig. 6.은 GameCIH를 이용한 메모리 변조 과정의 예시이다[17].

메모리 변조 앱을 통해 게임 상의 수치를 변경할 경우 대개 값의 전후 차이가 극명하기 때문에 이에 대한 탐지는 상대적으로 쉬운 편이다.

Table 3. List of Memory Modification Apps.

Application Name	Country	Cost	Latest Version (Release Date)
Game CIH	Taiwan	Free	3.0.0.0 (2012.02)
Game Guardian	China	Free	6.0.5.9 (2013.11)
Game Killer	China	Free	-
SB Game Hacker	China	Free	3.1 (2014.12)

Fig. 6. Attempt to Modify Memory using Game CIH

2.1.3 매크로

매크로는 게임 내 사용자의 특정 행위를 녹화하고 원할 때마다 반복적으로 수행하는 프로그램을 말하며, 크게 스마트기기에서 작동하는 것과 PC에서 작동하는 것으로 나뉜다. 2.1.1에서 소개한 에뮬레이터를 이용하여 PC에서 모바일 게임을 하는 이용자가 많기 때문에 PC용 매크로도 많이 사용되고 있다. Table 4.는 모바일 게임에서 사용되는 대표적인 매크로의 현황을 나타내고[21,22,23,24,25], Fig. 7.은 이 중 하나인 FRep의 User Interface를 나타낸다[23].

Table 4. List of Macro Applications

Application Name	Plat-form	Cost	Latest Version (Release Date)
AutoHot key	PC	Free	1.1.22.03 (2015.06)
AutoIt	PC	Free	3.3.14.1 (2015.07)
FRep	Smart Device	Free	3.8 (2015.08)
G Macro	PC	Free	2.0 (2003.11)
Hiro Macro	Smart Device	Free	2.0.1 (2015.06)

매크로의 사용이 무조건 불법이라고 판단하기에 애매모호한 측면이 존재하고, 매크로 사용 시 게임사에서는 이것이 사용자의 행위인지, 매크로로 인한 반복적인 행위인지 판단하기 쉽지 않다. 따라서 게임 앱 실행 전에 매크로 실행여부를 검사하여, 매크로가

실행되어 있을 경우 게임이 실행되지 않도록 조치를 취하는 것이 효율적인 대응방법이다.

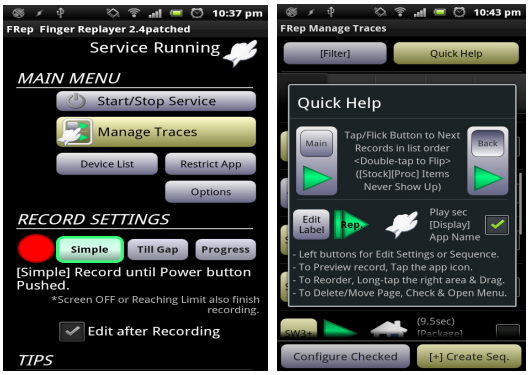


Fig. 7. FRep's User Interface

2.1.4 게임 앱 크랙 버전

게임 앱 크랙 버전은 기본적으로 마켓 구매인증과정을 우회하고 게임 내 골드, 아이템 개수 등 특정 항목의 수치를 고정시키거나 인앱결제 시 무료로 아이템을 구매하게 하는 등 제작자에 따라 다양한 형태를 가질 수 있다.

Fig. 8.은 국내의 대표적인 안드로이드 앱 커뮤니티 "앱짱닷컴(appzzang.ca)"의 크랙 게시판이다. 2015년 10월 8일 기준 761개의 크랙 버전 게임 앱이 등록되어 있고 누구나 무료로 다운받을 수 있다[26].

Fig. 8. Sharing Bulletin Board of Android Game Crack Version

2.1.5 특정 게임 전용 오토 프로그램

앞서 설명했던 2.1.2의 메모리 변조 앱과 2.1.3의 매크로는 임의의 모바일 게임에 범용적으로 사용 가능한 프로그램이고, 본 절에서 설명할 것은 특정 게임에 특화시켜 제작된 전용 오토 프로그램이다. 이러한 전용 오토 프로그램은 대부분 윈도우 운영체제에서 작동되고 2.1.1에서 소개한 안드로이드 에뮬레이터인 BlueStacks의 실행을 요구한다. Fig. 9.는 클래시 오브 클랜의 전용 오토 프로그램 중 하나인 Lazy Pressing의 사용자 인터페이스를 나타낸다. 대부분의 프로그램들이 모두 유료 형태를 띠고 있지만 일정 기능이 제한된 데모 버전을 동시에 제공하고, 과금방식은 월/분기/반기/연간 등으로 다양하며 결제 방식 또한 페이팔, 비트코인 등으로 다양하다[27].

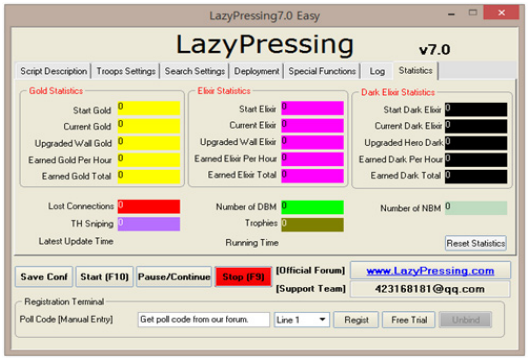


Fig. 9. LazyPressing's User Interface

2.2 오토 프로그램의 악성 행위 유무 분석 시 점검 항목 선정을 위한 선행 연구

본 논문에서는 오토 프로그램이 트로이목마 형태의 악성코드일 것이라 전제하고 있기 때문에 이에 대한 분석 방법은 기존의 악성코드 분석 방법과 매우 유사한 관점을 가진다. 이에 따라 이 절에서는 정보보호 관련 정부 기관 및 전문 기업에서 발간된 악성코드 분석 지침 또는 분석 보고서를 일차적으로 검토하여 기존 점검 항목의 당위성을 판단하고, 최근 악성코드의 특징을 이차적으로 분석하여 추가적으로 반영해야 할 점검 항목을 검토한다. 이를 통해 다음 3장에서 오토 프로그램의 악성 행위 유무 분석 시 점검 항목을 최종 도출한다.

2.2.1 정보보호 관련 정부 기관 및 전문 기업의 악성코드 분석 지침, 분석 보고서 검토

악성코드 분석 시 기존 점검 항목의 적절성을 검

토하기 위해 정보보호 관련 정부 기관 2개, 전문 기업 2개에서 발간된 악성코드 분석 지침 또는 분석 보고서를 검토했다. 선정된 정부 기관 및 전문 기업과 관련 문헌은 Table 5.와 같다[28,29,30,31].

KISA(Korea Internet & Security Agency)는 한국의 정보보호 진흥기관로서 인터넷침해대응센터(www.krcert.or.kr)를 운영하고 있다. SEI(Software Engineering Institute)는 미국 국방부에 의해 설립된 비영리 연구개발 센터로서 CERT(www.cert.org)를 운영하고 있다. Lastline의 Anubis와 Google의 VirusTotal은 온라인 상에서 샌드박스를 통한 악성코드 분석 서비스를 제공한다[28,29,30,31].

Table 5. Literature List

Name	Literature
KISA	Handbook of Incident Analysis Procedure
SEI	A New Approach to Prioritizing Malware Analysis
Anubis	Fiesta EK(CVE-2013-2551) Analysis Report
VirusTotal	Fiesta EK(CVE-2013-2551) Analysis Report

각각의 지침과 분석 보고서를 검토한 결과 세부 점검 항목은 약간의 차이가 존재했지만 모든 문헌에서 File, Registry, Process/Service, 이 3개의 항목을 큰 축으로 하여 해당 항목들의 변화를 중점적으로 분석했다. 일반적으로 악성코드가 파일 형태로 존재하고 실행 시 프로세스로서 메모리에 상주하는 점, 지속적인 실행을 위한 레지스트리에 관련 키를 생성하거나 서비스에 등록하는 점 등을 고려했을 때 기존 3개의 항목은 적절한 점검 항목이라 판단된다.

2.2.2 최근 악성코드 행위의 특징 분석

2.2.2.1 정보 유출형 악성코드의 득세

공격자의 호기심이나 실력 과시를 위해 해킹을 하던 과거와 달리 최근에는 금전적 목적을 가지고 개인의 민감한 정보를 노린 해킹이 대부분이다. Table 6.의 2015년 7월 월간 악성코드 은닉사이트 탐지 동향 보고서의 악성코드 유형별 비율을 살펴보면, 정보유출형 53%, 키로깅 7%, 파밍 3%, 총 63%의 악성코드가 직간접적으로 정보유출을 목적으로 하고 있다[32]. 이에 따라 동적/정적 분석을 통해 악성코드 내 IP 주소 또는 URL의 존재 유무를 파악하고 직접 접근하여 이것들의 유해성을 점검해야 할 필요가 있다.

Table 6. Classification of Malware by Percent

Malware's Type	Percent
Financial Information Leakage	42%
Dropper	16%
PC Information Leakage	11%
Key-logging	7%
Downloader	7%
Remote Control	4%
Adware	4%
Abnormal Files	4%
Pharming	3%
Etc.	2%
Total	100%

2.2.2.2 정상 프로그램의 업데이트를 악용한 악성코드 배포

감염 대상 시스템에 악성코드를 설치하는 것이 점점 어려워짐에 따라 공격자들은 정상적인 프로그램의 업데이트 체계를 악용하여 악성코드를 배포하기 시작했다. 2014년 KISA에서 발간한 "최근 피싱, 파밍 기법을 이용한 금융정보탈취 동향"에 따르면 공격자들은 웹하드 업데이트 서버 내 파일 자체를 변조하거나 감염 대상 시스템의 업데이트 설정 파일을 변조하였고[33], 동년 안랩에서 발간한 ASEC(AhnLab Security Emergency Response Center) Report에 따르면 프리웨어 등의 소프트웨어 설치 시 부가적으로 설치될 수 있는 PUP(Potentially Unwanted Program)의 업데이트를 통해 시스템을 감염시켰다[34]. 최초 설치된 프로그램은 정상일지라도 향후 업데이트를 통해 악성코드에 감염될 가능성이 존재하므로 자체 업데이트 기능의 존재 유무와 업데이트 메커니즘에 대한 점검이 필요하다.

2.2.2.3 분석 지연을 위한 코드 난독화 및 패킹 기법 사용

악성코드의 제작 의도를 달성하고 분석가가 악성

코드의 동작 방식을 파악하는 데 걸리는 시간을 최대한 지연시키기 위해, 일반적으로 공격자들은 악성코드 제작 시 코드 난독화 및 패킹 등의 보호 기법을 적용한다[35].

Table 7. Classification of Obfuscation

Obfuscation	Contents
Layout	Scrambling The Program around at The Source-Level
Control-Flow	Twisting The Typical Downward-Flow of A Program into Spaghetti Code
Data	Masking Data in Program by Any Means

Table 8. Monthly Packer Statistics

Packer	Count	Percent
Allaple_Polymorphic_Packer vna	1,870,224	67.32%
UPX All_Versions	452,728	16.30%
NullSoft_PiMP_SFX vna	203,211	7.31%
Inno_Setup v2.0.1	84,715	3.05%
FSG V1.0-1.2	40,941	1.47%
NullSoft_NSIS Generic	33,504	1.21%
UPX V2.9-3.X	32,179	1.16%
Unknown_33	10,849	0.39%
InstallShield v2000	6,925	0.25%
ASPack vna	5,434	0.20%

불특정 다수가 아닌 특정 그룹이나 개인을 지속적으로 공격하는 APT(Advanced Persistent Threat)가 현재 사이버 공격 트렌드의 주류로 떠오르는 가운데, 다단계로 존재하는 보안 솔루션을 최대한 우회하고 악성코드의 활동 시간을 최대한 지속하기 위해서 이러한 보호 기법은 악성코드 제작 시 더더욱 필수불가결한 요소가 되어 가고 있다. 코드 난독화 및 패킹 기법이 적용되었다고 해서 무조건 악성코드라고 단정할 순 없지만 그 확률이 매우 높으므로 의심 파일에 대한 코드 난독화 및 패킹 기법 적용 유무의 점검이 필요하다.

Table 7.은 대표적인 코드 난독화 기법을 나타내고[36], Table 8.은 Shadowserver Foundation에서 제공된 7월 17일부터 8월 16일 사이에 유입된 악성코드에 적용된 패커들의 현황을 나타낸다[37].

III. 분석 방법

3.1 점검 항목

2.2의 결과에 근거하여 총 5개 점검 항목으로 구성하며 상세 내용은 Table 9.와 같다.

Table 9. Inspection Checklist

Division	Contents
Item 1	File Creation/Modification/Deletion
1-a	File Creation/Modification/ Deletion by Execute Files
1-b	File Download by Connecting Network
Item 2	Registry Key Creation /Modification/Deletion
Item 3	Malicious Activity Related to Process/Service
3-a	Antivirus Process Pause/Kill
3-b	Backdoor Process Execution
3-c	Etc.
Item 4	Analysis of Possibility of Potential Threat
4-a	Existence of Update Function
4-b	Existence of Malicious IP Address/Domain
Item 5	Code Protection
5-a	Code Obfuscation
5-b	Packing

3.2 점검 도구

점검 도구는 에뮬레이터, 동적 분석, 정적 분석, 이력 조회 서비스와 같이 크게 4개로 구분되며 상세 내용은 Table 10.과 같다.

3.3 점검 방법

점검 항목과 관계없이 기본적인 분석 절차는 오토 프로그램의 import 함수의 목록, 문자열 및 패킹

적용 유무 분석, 모니터링 도구를 이용한 시스템 변화 분석 등과 같은 비교적 간단한 정적/동적 분석을 수행하고, 해당 결과를 바탕으로 상세 분석이 필요한 부분을 도출한다. 그리고 해당 부분에 한해 디스어셈블링 및 디버깅을 함으로써 상세 분석을 수행한다. 다음은 점검 항목별 주요 내용을 서술한다.

Table 10. Inspection Tools

Division	Inspection Tools	Reference
Emulator	VMware Workstation v10	38
	BlueStacks v0.9.25.5401	14
Dynamic Analysis	Process Monitor v3.2	39
	Process Explorer v16.05	40
	Regshot v1.9.0	41
	Wireshark v1.12.5	42
Static Analysis	OllyDbg v1.10	43
	PE Explorer v1.99	44
	PEBrowse Professional v10.1.5.0	45
	Dependency Walker v.2.2	46
	.NET Reflector v8.5.0.179	47
	BinText v3.0.3	48
	Stud_PE v2.6.1.0	49
	Exeinfo PE v.0.0.3.7	50
	HxD v1.7.7.0	51
History Inquiry Service	www.virustotal.com	52
	www.malwares.com	53
	Google's Safe Browsing	54
	KISA's WHOIS	55

3.3.1 파일 생성/변경/삭제

본 항목에서는 Process Monitor 및 Wireshark와 같은 도구를 이용해 시스템 내 파일들의 가시적인 변화를 분석하고, CreateFile, ReadFile, WriteFile, DeleteFile, Create FileMapping, MapViewOfFile과 같은 관련 함수의 존재 유무 및 해당 함수에 전달되는 파라미터를 분석한다.

3.3.2 레지스트리 키 생성/변경/삭제

본 항목에서는 Process Monitor 및 Regshot과 같은 도구를 이용해 레지스트리의 가시적인 변화를 분석하고 RegCreateKey, RegDeleteKey, RegOpenKey, RegSetValue, RegGetValue와 같은 관련 함수의 존재 유무 및 해당 함수에 전달되는 파라미터를 분석한다.

3.3.3 프로세스 및 서비스와 관련된 악성 행위

본 항목에서는 Process Monitor 및 Process Explorer와 같은 도구를 이용해 프로세스 및 서비스의 가시적인 변화를 분석하고, CreateProcess, CreateThread와 같은 관련 함수의 존재 유무 및 해당 함수에 전달되는 파라미터를 분석한다.

3.3.4 잠재적 위협 발생 가능성 분석

본 항목에서는 업데이트 기능의 존재 유무와 소스코드 내 악성 IP 주소 및 도메인 주소의 존재 유무를 분석한다.

업데이트 기능의 존재 유무 분석의 경우, 우선 오토 프로그램의 User Interface 상에 업데이트 메뉴 존재 유무를 확인하고 해당 메뉴가 존재할 경우 업데이트 관련 설정 파일 등의 분석을 통해 업데이트 시 접속하는 원격지 서버 주소의 변조 가능성을 확인한다. 그리고 socket, bind, listen, accept, connect, recv, send와 같은 관련 함수의 존재 유무 및 해당 함수에 전달되는 파라미터를 분석하여 사용자 동의 없이 백그라운드에서 업데이트가 수행될 가능성을 확인한다.

소스코드 내 악성 IP 주소 및 도메인 주소 분석의 경우 문자열 분석 및 위에서 언급한 함수들의 파라미터 분석을 수행하고, 추출된 IP 주소 및 도메인 주소에 직접 접속하거나 이력 조회 서비스를 이용하여 유해성 유무를 분석한다.

3.3.5 코드 보호 기법 존재 유무 분석

본 항목에서는 코드 난독화 및 패킹 기법의 적용 유무를 분석한다. Exeinfo PE, PEBrowse와 같은 도구를 이용하여 패킹 적용 유무 및 적용된 패커의 종류를 파악하고 그 종류에 따라 언패킹 도구를

이용하거나 매뉴얼 언패킹을 시도한다. 코드 난독화 여부는 디스어셈블링 및 언패킹의 결과물을 통해 확인한다.

IV. 실 험

4.1 실험 대상

실험 대상은 총 7종의 클래시 오브 클랜 전용 오토 프로그램으로서 각각의 공식 사이트를 통해 확보했다. BlueStacks에서의 실행을 요구하며 총 7종 중 5종이 유료, 2종이 무료로 배포되고 있다. 이를 요약한 것이 Table 11.이다.

Table 11. List of Auto Program of Clash of Clans

Program Name	Cost	Source Code Disclosure
A	free	X
B	free	O
C	6$/M	X
D	10$/M	X
E	11$/M	X
F	20$/M	X
G	10$/M	X

4.2 실험 환경

실험은 변수 통제 및 반복적인 실험의 용이성을 확보하기 위해 가상환경에서 실시한다. VMware Workstation v10에서 guest OS를 Windows 7로 구성하고 guest OS 내 BlueStacks를 설치한다.

4.3 실험 결과

4.3.1 요약

실험 결과 7개 프로그램 모두, 악성 행위를 직접적으로 수행하는 부분은 존재하지 않았다. 파일 및 레지스트리 키를 생성/변경/삭제하는 행위는 존재하지 않았고, 안티 바이러스 프로그램을 중지하거나 백도어 프로세스를 생성하는 등 프로세스 및 서비스와 관련된 악성 행위도 존재하지 않았다.

그러나 프로그램 A에 자동 업데이트 기능이 존재하고, 프로그램 D, E, F에서 악성 IP 주소 또는 URL이 검출되었다. 그리고 프로그램 D에 소스코드 난독화가 적용되었고, 프로그램 A, C, F, G에 패킹이 적용된 것이 확인됨에 따라 이러한 요소들이 향후 잠재적 보안 위협이 될 수 있음을 추정할 수 있었다. 이를 요약한 것은 Table 12.이다.

Table 12. Inspection Result

Division	A	B	C	D	E	F	G
Item 1							
1-a	-	-	-	-	-	-	-
1-b	-	-	-	-	-	-	-
Item 2	-	-	-	-	-	-	-
Item 3							
3-a	-	-	-	-	-	-	-
3-b	-	-	-	-	-	-	-
3-c	-	-	-	-	-	-	-
Item 4							
4-a	O	-	-	-	-	-	-
4-b	-	-	-	O	-	O	O
Item 5							
5-a	-	-	-	O	-	-	-
5-b	O	-	O	-	-	O	O

Table 13. Script Language used and Packer applied by Program

Program Name	Script Language	Packer Name
A	AutoIt	UPX
B	AutoIt	-
C	AutoIt	NSIS
D	AutoIt	-
E	-	-
F	Quick Macro	Themida
G	Quick Macro	Themida

그리고 7개 프로그램 중 6개가 AutoIt 또는 Quick Macro 라는 스크립트 언어로 개발된 것임이 드러났다. 우선 스크립트 언어로 개발하고 코드 보호 및 배포를 위해 exe 형식으로 변환한 것으로 추정된다. 나머지 1개는 .NET Framework를 이용해 개발되었다. 이를 요약한 것은 Table 13.이다.

4.3.2 상세 내용

4.3.2.1 자동 업데이트 존재 유무

A 프로그램의 경우, 자동 업데이트 기능이 존재했다. 기본적으로는 해당 기능이 비활성화되어 있으나, 활성화시키고 신규 업데이트가 존재할 경우 OOO.zip 파일을 다운로드한다. 이후 해당 파일의 압축을 해제하면 XXX.bat 파일이 생성되고, 이를 실행함으로써 본격적인 업데이트가 시작된다. XXX.bat 실행 이전에 업데이트 관련 파일이 포함된 디렉터리에 대한 삭제 명령을 추가함으로써 업데이트가 종료된 이후에는 관련된 모든 파일이 삭제된다.

현재 작성된 코드에서는 이상이 없으나, 최초 OOO.zip 파일을 받기 위해 접근하는 URL을 악성코드 유포 서버로 변경하여 임의의 악성코드를 유포할 수 있으므로 향후 보안 위협으로 작용할 수 있다.

4.3.2.2 악성 IP 주소/도메인 존재 유무

프로그램 D에서 www.neemedia.com 이라는 도메인 주소가 발견되었다. 해당 도메인으로 직접 접속을 시도하였고 이후 ww2.neemedia.com/?folio=9POR7JU99로 request를 보내는 과정에서 Fig. 10.과 같이 안티 바이러스 프로그램으로 인해 접근이 차단되었다. 안티 바이러스 프로그램 해제 후 재접속을 시도하였으나 "This domain has recently been listed in the marketplace" 메시지가 확인되었고, user-agent를 다양하게 변경하여 접속을 시도해도 동일한 메시지가 확인됨에 따라 악성코드 유포지가 변경되었음을 추정할 수 있었다.

프로그램 F, G의 경우 동일한 이메일 주소(hi@vrbrothers.com) 및 도메인 주소(www.anjian.com, www.xiaojl.com)가 확인되었는데 이것 모두 Quick Macro 개발사를 나타냈다. 악성코드 유포/경유 이력을 조회한 결과, 개인 사용자가 제작한 스크립트를 공유하는 게시판 때문에 탐지된 것으로 추정되는 일부를 제외하고 특이사항이 없었다.

Fig. 10. Block to connect to ww2.neemedia.com/?folio=9POR7JU99

4.3.2.3 코드 난독화 적용 유무

AutoIt으로 제작된 오토 프로그램은 일반적으로 Exe2Aut 라는 도구를 통해 소스코드를 복원하여 분석이 가능하다[56]. 그러나 그 중 프로그램 D는 복원된 소스코드를 확인한 결과 Fig. 11.과 같이 변수명과 변수 값, 사용자정의 함수 명에 의미를 알 수 없는 임의의 문자열이 할당되어 있었다. Win32API Wrapper 함수명은 그대로 나타나기 때문에 대략의 행위만 짐작할 수 있었으나 정확한 것은 알 수 없었다. 사용되는 파라미터에 따라 동일한 함수가 악성행위를 수행할 수도 있고 정상행위를 수행할 수도 있으므로, signature 기반의 정적 분석을 이용한 탐지는 정확한 결과를 보장할 수 없다. 동적 분석을 수행한다 하더라도 특정 조건을 만족시키지 않을 경우 악성 행위가 발생하지 않을 수 있기 때문에 이 또한 우회 가능성이 존재하므로 이러한 난독화 기법은 잠재적 보안 위협 요소가 될 수 있다.

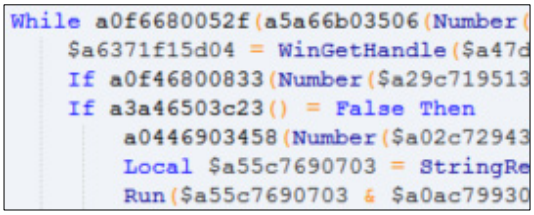


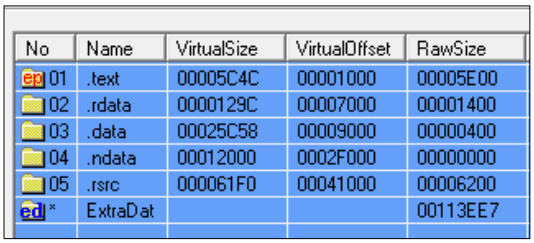
Fig. 11. Obfuscated Source Code

4.3.2.4 패킹 유무

프로그램 C의 경우 Fig. 12.와 같이 ExtraDat라는 별도의 섹션이 존재했고 다른 섹션에 비해 상대적으로 그 크기가 매우 컸다. 해당 섹션을 추출하여 확인한 결과 NSIS(Nullsoft Scriptable Install System)를 이용해 생성된 것임을 알 수 있었고[57], 7-zip으로 압축 해제 시 다시 exe 파일이 추출되었다. 해당 exe 파일은 resource 분석 결과 AutoIt으로 개발된 것임이 드러났고 Exe2Aut 도

구를 이용하여 소스코드를 복원, 결과적으로 악성 행위가 존재하지 않음을 확인했다.

패킹이 적용된 파일은 내부 파일이 정상적인 코드임에도 불구하고 안티 바이러스 프로그램의 종류에 따라 악성코드로 진단하는 경우도 있고 아닌 경우도 있기 때문에 신뢰성 있는 진단 결과를 기대하기 어렵다. 즉, 악성코드를 은닉한 뒤 패킹할 경우 미탐이 발생할 가능성이 존재하므로 이러한 패킹 기법은 잠재적 보안 위협 요소가 될 수 있다.



No	Name	VirtualSize	VirtualOffset	RawSize
01	.text	00005C4C	00001000	00005E00
02	.rdata	0000129C	00007000	00001400
03	.data	00025C58	00009000	00000400
04	.ndata	00012000	0002F000	00000000
05	.rsrc	000061F0	00041000	00006200
*	ExtraDat			00113EE7

Fig. 12. Section Information of Program C

프로그램 A는 AutoIt에서 기본적으로 제공하는 무료 패커 UPX, 프로그램 F, G는 상용 패커 Themida로 패킹되었다[58,59]. Themida는 악성코드의 분석을 지연시키기 위해 악성코드 제작자들이 많이 사용하기 때문에 해당 프로그램이 악성 행위를 할 것으로 예상되었으나 분석 결과 악성 행위는 존재하지 않았다.

V. 평 가

5.1 스크립트 언어의 보안 위협 확인

총 7개의 오토 프로그램 중 6개의 오토 프로그램이 스크립트 언어를 이용하여 개발한 뒤 배포를 위해 exe 형식으로 변환되었고, 코드 보호를 위해 필요에 따라 코드 난독화 및 패킹 기법이 적용된 것으로 확인되었다. 스크립트 언어를 이용해 개발된 총 6개의 오토 프로그램 중 4개가 AutoIt, 나머지 2개가 Quick Macro가 사용되었는데 이 스크립트 언어들은 본연의 단순 반복 작업이 아닌 시스템에 영향을 미치는 다양한 작업을 수행할 수 있다. 다음 Table 14.는 악성 행위 시 사용될 수 있는 AutoIt의 주요 함수를 나타낸다[60].

직접적으로 파일을 생성하는 것을 제외하고, 외부 네트워크 접속을 통한 파일 다운로드, 프로세스 삭제, DLL 내 함수 호출, 레지스트리 키 생성 등 악성코드 제작에 필요한 기본적인 행위가 거의 가능하다. Quick Macro와 AutoHotkey에서도 비슷한 역할을 수행하는 함수들이 제공된다[61,62]. 해외의 경우 AutoIt을 이용한 악성코드 이슈가 몇몇 존재하는데, 이 중 대표적인 것은 Table 15.와 같다[63,64].

Table 14. AutoIt's Main Function

Division	Function
File	ShellExecute, Run, RunAs, FileDelete, FileCopy, FileRead, FileMove
Directory	DirCreate, DirRemove
Registry	RegRead, RegWrite, RegDelete
Process	ProcessClose, ProcessList, ProcessExists, Shutdown
DLL	DllOpen, DllCall, DllClose
Network	Ping, InetGet, TCPSend, TCPAccept, TCPCloseSocket, TCPListen, TCPRecv, TCPStartup, TCPTimeout

Table 15. Foreign Cases of AutoIt Malware

Year	Contents
2012	MITB attck using Malware Named Boleto in Brazil - Injection of binary code into a system process that will search for browser processes in order to inject malicious code inside of them
2014	Sending spam mail of Ebola virus fashioned in those days with impersonate WHO - Concealment of obfuscated Dark Comet Remote Access Trojan in attached file using AutoIt script

이번에 분석한 오토 프로그램에서도 악성 행위를 위한 목적으로 사용되지는 않았지만 Table 14.의 함수들 중 일부가 사용되었고 이러한 부분은 향후 얼마든지 위험 요소로 작용할 수 있다.

그리고 AutoIt으로 개발된 프로그램은 Exe2Aut 도구를 이용하여 쉽게 디컴파일할 수 있기 때문에 소스코드 복원이 아주 용이하다. 정상 프로그램에 악성코드를 추가할 수도 있고 기존 악성코드의 다양한 변

종 제작도 가능하다. 이러한 점은 안티 바이러스 프로그램의 즉각적인 대응을 어렵게 한다[65].

5.2 대응 방안 고찰

가장 효율적인 대응 방안은 BlueStacks에서 게임 구동 자체를 차단하거나, BlueStacks에서의 게임 구동을 허용하되 상용 오토 프로그램의 실행 유무를 사전에 검사하고 오토 프로그램이 실행되어 있을 경우 게임 실행을 차단하는 것이다. 하지만 이러한 극단적인 정책은 정상적인 게임 이용자의 반발 및 이탈을 발생시킬 확률이 높으므로 게임 회사 입장에서는 감수해야 할 위험이 크다. 게다가 국내의 경우 오토 프로그램의 제작 및 배포에 관한 법적 제재 근거는 존재하지만[66,67] 사용 행위에 대한 적법성 여부는 게임 계정 복구 소송 등과 같이 여전히 논란의 불씨가 남아 있기 때문에 이러한 정책을 적용하기 쉽지 않다[68].

이에 따라 BlueStacks에서의 게임 실행 및 오토 프로그램 실행을 허용한 상황에서 가능한 대응방안을 고려해야 한다.

Fig. 13.에서 볼 수 있듯이 스크립트 언어 사용 여부는 패킹여부와 상관없이 확인할 수 있다. 하지만 어떠한 Win32API Wrapper 함수를 사용하고 그 함수에 어떠한 값을 전달하는지는 확인할 수 없다. 패킹하지 않은 경우 Fig. 14.와 같이 함수명이 평문으로 노출되는 것을 확인할 수 있지만, 이것은 호출된 함수가 아닌 AutoIt에서 사용가능한 모든 함수를 가리키므로 무의미한 정보이다. 즉, 파일을 실행하지 않고 오직 정적 분석을 통한 signature 기반의 탐지 기법만으로는 스크립트 언어를 악용한 악성 코드를 탐지하기 어렵다. 실행하는 과정을 지속적으로 관찰하여 악성 행위 발생 시 탐지하는 동적 분석이 반드시 요구된다. 따라서 개인 사용자 입장에서는 안티 바이러스 프로그램이 유일한 대안이다.

하지만 기업에서 업무용 목적으로 스크립트 언어를 이용해 프로그램을 개발하는 경우가 많고 이로 인해 오탐 발생 시 사용자 불만이 크게 제기되기 때문에 안티 바이러스 개발사마다 다소 차이는 있지만 기본적으로 스크립트 언어로 개발된 프로그램에 대해서 다소 느슨한 탐지 정책을 적용하는 경향이 있다[69,70]. Table 16.은 이번에 분석한 오토 프로그램 중 스크립트 언어로 개발된 6개의 오토 프로그램을 VirusTotal에서 분석한 결과로서, 평균적으로 약 28%의 안티 바이러스 프로그램만이 trojan 또는 possible threat, unclassified malware 등으로 진단하는 등 전반적으로 진단율이 매우 낮음을 알 수 있다. 스크립트 언어로 개발된 프로그램에 대한 탐지 정책의 민감도 제고가 고려되어야 한다.

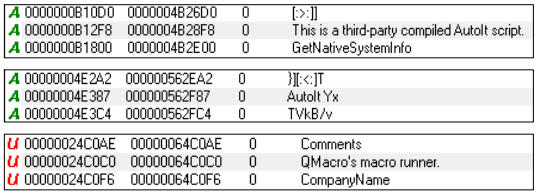


Fig. 13. Script Language's Name Exposed

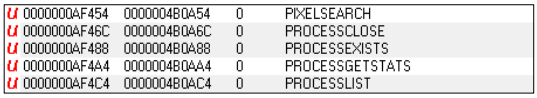


Fig. 14. AutoIt's Win32API Wrapper Function List Exposed

Table 16. Antivirus Software's Ratio which Detects Auto Program Developed by Script Language

Program Name	Ratio(%)	Major Diagnosis
A	3.6	Trojan Possible Threat Unclassified Malware
B	32.7	
C	38.2	
D	52.7	
F	25.5	
G	12.7	
Average	27.6	-

VI. 결 론

PC 기반의 온라인 게임에서 나타났던 오토 프로그램 문제가 모바일 게임에서 재현되고 이를 이용한 악성 행위 수행 가능성이 높아짐에 따라, 본 논문에서는 모바일계의 장수 게임인 클래시 오브 클랜의 오토 프로그램을 대상으로 악성 행위 유무를 분석하고 이를 바탕으로 향후 발생 가능한 모바일 게임 보안 위협을 예측 및 분석하였다.

분석한 모든 오토 프로그램은 자체 공식 홈페이지를 통해 확보하였고. 해당 게임이 BlueStacks 에뮬레이터 상에서 작동하는 경우를 가정하여 개발되었다. 총 7개의 오토 프로그램을 대상으로 파일/레지스트리 키의 생성/변경/삭제, 프로세스/서비스의 실행/중지, 자동 업데이트 실행, 악성 URL/IP 주소 접근, 소스코드 난독화 및 패킹 적용 유무 등 총 10개 항목을 기준으로 분석을 진행했고 결과적으로 직접적인 악성 행위는 존재하지 않는 것으로 확인되었다. 하지만 자동 업데이트 기능이 존재하거나 소스코드 난독화, 패킹 기법이 적용된 프로그램의 경우 개발자의 의도에 따라 악성 행위가 가능할 수 있음을 확인했다.

그리고 총 7개의 프로그램 중 6개가 AutoIt, Quick Macro와 같은 스크립트 언어를 이용하여 제작된 후 컴파일된 것으로 확인되었다. 이 스크립트 언어들은 단순한 반복적인 작업만 구현할 수 있는 것이 아니라 외부 네트워크에서의 파일 다운로드, 파일 실행 및 삭제, 외부 DLL 내 함수 호출 등 악성 행위에 필요한 대부분의 행위를 구현할 수 있고, 특히 AutoIt로 컴파일된 프로그램의 경우 Exe2Aut 도구를 이용하여 소스코드를 쉽게 복원할 수 있기 때문에 정상 프로그램에 악성코드를 삽입하거나 기존 악성코드의 새로운 변종을 제작하는 것이 매우 쉽다. 즉, 현존하는 모바일 게임 오토 프로그램에 악성코드를 추가하는 것은 아주 쉬운 작업이고, 현재 많은 게임 이용자들이 오토 프로그램을 사용하기 때문에, 오토 프로그램에 악성코드를 은닉시켜 재유포할 경우 많은 감염자를 양산할 수 있다.

가장 이상적인 대응방안은 BlueStacks에서의 게임 실행 차단 또는 게임 실행까지는 허용하되 오토 프로그램의 실행을 차단하는 것이다. 하지만 이것은 게임 이용자의 반발 및 이탈이 우려되므로 현실적으로 적용하기 쉽지 않다. 현실적인 대응방안은 안티 바이러스 프로그램을 중지하거나 삭제하지 않은 채 계속 상주시키는 것이다. 하지만 기업에서 업무용 목적으로 스크립트 언어를 이용해 프로그램을 개발하는 경우가 많고 오탐 발생 시 사용자의 불만이 높기 때문에 안티 바이러스 프로그램 개발사에서는 스크립트 언어로 개발된 프로그램에 대한 탐지 정책을 상대적으로 느슨하게 적용하는 경향이 있다. 그러므로 스크립트 언어로 개발된 프로그램에 대한 탐지 정책의 민감도를 높이면서 오탐률을 최소화하기 위해 AutoIt과 같은 스크립트 언어 기반의 악성코드에 대한 지속적인 연구가 필요할 것이라 판단된다.

References

〔1〕 Korea Creative Content Agency, 2014 White paper on Korean games, Korea Creative Content Agency, 35, Gyoyuk-gil, Naju-si, Jeollanam-do, Korea, 2014.

〔2〕 Google Play, https://play.google.com/store/apps/category/GAME/collection/topselling_free

〔3〕 Woo, Jiyoung and Huy Kang Kim. "Survey and research direction on online game security," Proceeding WASA '12 Proceedings of the Workshop at SIGGRAPH Asia, pp. 19-25, Nov, 2012.

〔4〕 Woo, Jiyoung, Hwa Jae Choi, and Huy Kang Kim. "An automatic and proactive identity theft detection model in MMORPGs." Applied Mathematics & Information Sciences, Vol. 6, No. 1S, pp. 291S-302S, Jan, 2012.

〔5〕 Hana Kim, Byung Il Kwak, and Huy Kang Kim, "A study on the identity theft detection model in MMORPGs." Journal of The Korea Institute of Information Security & Cryptology, vol.25, no.3, pp. 627-637, Jun. 2015.

〔6〕 Huy Kang Kim and Young Jun Kum, "Mobile game security issue in android." Review of The Korea Institute of Information Security & Cryptology, vol.23, no.2, pp. 35-42, Apr. 2013.

〔7〕 Il-bum Ahn, Shock! mobile games mouse auto prevalent 'fake ranked advisory', http://news.heraldcorp.com/view.php?ud=20130610000563&md=20130613004402_BL, Herald, Jun. 2013.

〔8〕 Seung-Jin Choi, Clash of clans, the server pre-emergence 'shock', http://news.tf.co.kr/read/economy/1477785.htm, THE FACT, Jan. 2015

〔9〕 Kang, A. R., Woo, J. Y., and Kim, H. K., "Data and text mining of communication

patterns for game bot detection," Proceedings of the 3th international conference on Internet, pp. 495-500, Dec. 2011.
[10] Kang, A. R., Kim, H. K., and Woo, J.. "Chatting pattern based game BOT detection: do they talk like us?," KSII Transactions on Internet and Information Systems (TIIS), Vol.6, No.11, pp. 2866-2879, Nov, 2012.
[11] Lee, Gi Seong and Huy Kang Kim. "Android game repackaging detection technique using shortened instruction sequence," Journal of Korea Game Society, Vol. 13, No. 6, pp. 85-94, Dec, 2012.
[12] AMIDuOS, http://www.amiduos.com/
[13] Andy, http://www.andyroid.net/
[14] BlueStacks, http://www.bluestacks.com/local/kor/home-kor.html
[15] Genymotion, https://www.genymotion.com/#!/
[16] Windroy, http://www.windroye.com/
[17] GameCIH, http://www.cih.com.tw/gamecih.html
[18] Game Guardian, https://gameguardian.net/forum/
[19] GameKiller, http://game-killer.com/
[20] SB Game Hacker, http://sbgamehacker.com/
[21] AutoHotkey, http://www.autohotkey.com/
[22] AutoIt, https://www.autoitscript.com/site/autoit/
[23] FRep, http://strai.x0.com/frep/
[24] G Macro, http://rhyshan.com/147
[25] Hiro Macro, http://prohiro.com/
[26] Appzzang.com, http://appzzang.ca/bbs/board.php?bo_table=Game
[27] Lazypressing, https://www.lazypressing.com/
[28] Hacking Response Team, Handbook of incident analysis procedure, Korea Internet & Security Agency, 135, Jungdae-ro, Songpa-gu, Seoul, Korea, 2010.
[29] Jose Morales, A new approach to prioritizing malware analysis. http://insights.sei.cmu.edu/sei_blog/2014/04/a-new-approach-to-prioritizing-malware-analysis.html, Apr. 2014.
[30] Fiesta EK(CVE-2013-2551) Analysis Report, http://anubis.iseclab.org/?action=result&task_id=125ae9e1cdf70696411250b649e954117&format=pdf
[31] Fiesta EK(CVE-2013-2551) Analysis Report, https://www.virustotal.com/ko/file/f7ea603361599bed0b24f771da5b1b01126423d438dab2a1bfc7c7e4f6a1abec/analysis/
[32] Incident Response Corps, Monthly report on detecting sites concealing malware, Korea Internet & Security Agency, 135, Jungdae-ro, Songpa-gu, Seoul, Korea, 2015.
[33] Kim Moo Yeol, Ryu So Joon, Financial information leakage by the latest phishing and pharming technique, Korea Internet & Security Agency, 135, Jungdae-ro, Songpa-gu, Seoul, Korea, 2014.
[34] AhnLab Security Emergency Response Center, ASEC report vol. 56, AhnLab, 220, Pangyoyeok-ro, Bundang-gu, Seongnam-si, Gyeonggi-do, Korea, 2014.
[35] Joshua Cannell, Obfuscation: malware's best friend, Malwarebytes, https://www.malwarebytes.org/, 2014
[36] Sean Taylor, Binary obfuscation from the top down, DEF CON 17, https://www.defcon.org/html/defcon-17/dc-17-speakers.html#Taylor, 2009
[37] Shadowserver Foundation, Packer Statistics, https://www.shadowserver.org/wiki/pmwiki.php/Stats/PackerStatistics
[38] VMware Workstation, http://www.vmware.com/kr/products/workstation
[39] Process Monitor, https://technet.microsoft.com/ko-kr/sysinternals/bb896645

[40] Process Explorer, https://technet.microsoft.com/en-us/sysinternals/bb896653.aspx
[41] Regshot, http://sourceforge.net/projects/regshot/
[42] Wireshark, https://www.wireshark.org/download.html
[43] OllyDbg, http://www.ollydbg.de/
[44] PE Explorer, http://www.heaventools.com/
[45] PEBrowse Professinal, http://www.smidgeonsoft.prohosting.com/pebrowse-pro-file-viewer.html
[46] Dependency Walker, http://www.dependencywalker.com/
[47] .NET Reflector, http://www.red-gate.com/products/dotnet-development/reflector/
[48] BinText, http://www.mcafee.com/kr/downloads/free-tools/bintext.aspx
[49] Stud_PE, http://www.cgsoftlabs.ro/studpe.html
[50] Exeinfo PE, http://www.softpedia.com/get/Programming/Packers-Crypters-Protectors/ExEinfo-PE.shtml
[51] HxD, http://mh-nexus.de/en/hxd/
[52] Virustotal, https://www.virustotal.com/
[53] Malwares.com, https://www.malwares.com/
[54] Google Safe Browsing, https://www.google.com/safebrowsing/diagnostic?site=Google.com
[55] Korea Internet & Security Agency WHOIS, http://whois.kisa.or.kr/kor/
[56] Exe2Aut, https://exe2aut.com/
[57] NSIS, http://nsis.sourceforge.net/Download
[58] UPX, http://upx.sourceforge.net/
[59] Themida, http://www.oreans.com/themida.php
[60] AutoIt functions, https://www.autoitscript.com/autoit3/docs/functions/
[61] Quick Macro, http://www.quickmacrs.com/help/QM_Help/IDH_FUNCTION.html
[62] AutoHotkey, https://www.autohotkey.com/docs/Functions.htm
[63] Violet Blue, RSA: Brazil's 'Boleto Malware' stole nearly $4 billion in two years, http://www.zdnet.com/article/rsa-brazils-boleto-malware-stole-nearly-4-billion-in-two-years/, ZDNet, July. 2014.
[64] Lee Bell, Hackers use Ebola outbreak to trick users into downloading malware, http://www.theinquirer.net/inquirer/news/2377496/hackers-use-ebola-outbreak-to-trick-users-into-downloading-malware, the INQUIRER, Oct. 2014.
[65] Vinoo Thomas and Prashanth Ramagopal, and Rahul Mohandas, The rise of autorun-based malware, McAfee, http://www.mcafee.com/us/, 2009.
[66] Prohibiting distribution of illegal game, Paragraph 8 of Article 32 of Act on Game Industry Promotion, Nov. 2014.
[67] Lee Kang Kook, Verification of constitutional violation of clause 2 paragraph 3 of article 46 of Act on Game Industry Promotion, Heonjae 2012.6.27. 2011 Heonma288, Jun. 2012.
[68] Yang Chang Soo, Withdraw of blocking game account, Supreme Court 2010.10. 28. Sentence 2010Da 9153 Judgement, Oct. 2010.
[69] AutoIt and malware, https://www.autoitscript.com/wiki/AutoIt_and_Malware, Jun, 2014.
[70] Kyle Wilhoit, AutoIt used to spread malware and toolsets, http://blog.trendmicro.com/trendlabs-security-intelligence/autoit-used-to-spread-malware-and-toolsets/, May, 2013.

〈저 자 소 개〉

허 건 일 (Geon Il Heo) 정회원
2012년 2월: 서울과학기술대학교 산업정보시스템공학과 학사
2013년 9월~현재: 고려대학교 정보보호학과 석사과정
2011년 10월~2012년 11월: SK인포섹 침해사고대응팀
2012년 12월~2014년 9월: ETRI부설연구소
2014년 10월~현재: 안랩 보안관제팀
〈관심분야〉 온라인게임 보안, 네트워크 포렌식, 데이터 시각화

허 청 일 (Cheong Il Heo) 정회원
2013년 2월: 한국산업기술대학교 컴퓨터공학과 학사
2015년 2월: 한국산업기술대학교 기술정보보호학과 석사
2013년 2월~현재: 한국산업기술보호협회 중소기업기술지킴센터 관제운영팀
〈관심분야〉 네트워크 포렌식, 빅데이터 분석, 리버스 엔지니어링

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