게임해킹

스토리로 이해하는 해킹의 원리

예제 소스 다운로드

http://www.roadbook.co.kr/138

질의 응답 사이트

http://roadbook.zerois.net/QnA

게임 해킹 스토리로 이해하는 해킹의 원리

지은이 한주성 **1판 1쇄 발행일** 2015년 2월 16일 **펴낸이** 임성춘 **펴낸곳** 로드북 **편집** 장미경
디자인 이호용(표지), 박진희(본문) **주소** 서울시 관악구 신림로 29길 8 101-901호
출판 등록 제 2011-21호(2011년 3월 22일) **전화** 02)874-7883 **팩스** 02)6280-6901
정가 22,000원 **ISBN** 978-89-97924-14-1 93000

이메일 chief@roadbook.co.kr **블로그** www.roadbook.co.kr

지은이의 글

"여러분은 왜 해킹을 배우려고 합니까?"

필자가 대강 추측해본 답변들이다.

"솔직히 얘기하면 어둠의 세계가 좀 궁금해요. 난 블랙 해커(크래커)가 되고 싶거든요."

(그럼 나빠요~~. 솔직히 얼마나 많을지 무척 궁금하다. 필자의 적이기도 하니…)

"보안이 중요하다는데, 보안 책은 따분하고 해킹을 배우다 보면 자연스레 보안을 배울 수 있어서요."

"보안전문가가 되고 싶은데, 여러 책들을 섭렵하고 있어요. 화이트해커라고들 하죠."

"솔직히 게임이라는 단어에 솔깃했어요. 게임 해킹은 어떻게 하는지 궁금했거든요."

"IT 전문가가 되려면 좀 다방면으로 알아야겠더라구요."

질문은 단순한데, 답변은 정말 수십 가지다. 그것은 해킹이 여러 얼굴을 갖고 있기 때문이다. 블랙 해커와 화이트 해커처럼, 뚫으려는 자와 막으려 하는 자가 공존하고 해킹이라는 영역 자체가 시스템 레벨의 난이도 높은 지식이 필요하여 IT 전문가의 지적 호기심의 대상이 되기도 하기 때문이다.

이 책을 기획할 당시에 과연 수많은 요구를 어떻게 충족시켜줄 수 있을까 많은 고민을 했다. 그에 대한 해답은 *"범위를 축소하고 집중하자"* 였다. 독자의 요구를 네 가지 정도로 축소했고 이것이 곧 학습목표이자 이 책의 컨셉이 되었다.

"스토리가 있으면서 좀 재밌게 배우고 싶어요."

"게임은 돈(?)이 된다는데, 그 돈을 어떻게 지키는지 알고 싶어요."

"간단하게라도 공격용 코드와 방어용 코드를 배우고 싶어요."

"해킹을 당하지 않으려면 어떻게 시스템을 구축해야 하는지 알고 싶어요."

그리고 대표적인 해킹 기법 네 가지에 집중하였다. 그냥 이론적인 설명만 하자면 짧게 끝낼 수 있는 내용이다. 하지만, 해킹이 어떤 경로로 이루어졌는지 실제 이야기를 통해 몰입도를 높이고 다양한 툴들을 이용해 원인을 파악하고 때론 직접 툴도 만들어보고 공격용과 방어용 소스도 만들어보는 실습까지 곁들였다. 그냥 읽어도 재미있게 읽을 수 있는 책이지만 공부해야 할 것도 빼놓지 않은 에듀테인먼트 같은 책이라 할 수 있다.

필자는 이 책이 두 번째다. 이렇게 서문을 다시 쓰고 있자니 감회가 새롭다. 분명 1년 전 〈리버싱 윈도우〉라는 책을 처음으로 집필하여 완성하였을 때, *"다른 책을 못쓸 것 같다"* 라고 생각했는데, 생각과 현실은 다른 것 같다. 아마도 시간이 지날수록 밀려오는 아쉬움과 그리고 집필을 끝마치고 책으로 나왔을 때의 희열, 가끔이지만 독자의 이메일이 다시 한번 용기를 주었던 것 같다.

이 책은 전작에서 개선하고자 했던 부분들과 저자가 경험했던 해킹 사건들을 간접적으로나마 함께 공유하고 점점 더 복잡해져 가는 IT 산업에서 좀더 안전하게 서비스를 구축하고 운용했으면 하는 바람으로 집필하였다.

쓰면서 가장 고민했던 부분은 *"어떻게 하면 보다 재미있게 해킹 관련 지식을 전달할 수 있을까"* 였고 이를 위해 저자의 부족한 유머 감각(?)을 최대한 발휘하였다. 이 책을 끝까지 읽고 웃는 분이 한 사람이라도 있으면 좋겠다.

이렇게 책을 쓰고 이를 통해 여러분들에게 간접 경험을 전달할 수 있어 기쁘다. 책 안에서 독자가 알지 못했던 부분들이 잘 전달되기를 소망한다. 집필을 위해 힘써준 임성춘 편집장과 디자이너 이호용, 박진희님, 사랑하는 아내 성희, 지안, 재인, 그리고 곧 세상에 나올 재택이에게 감사의 말을 전한다.

해킹이라는 분야는 알면 알수록 배워야 할 게 많은 분야이다. 필자는 불혹을 훌쩍 넘긴 지금도 화이트해커로 살고 있고 죽을 때까지도 이 일을 멈추고 싶지 않다. 그만큼 매력적인 직업이다.

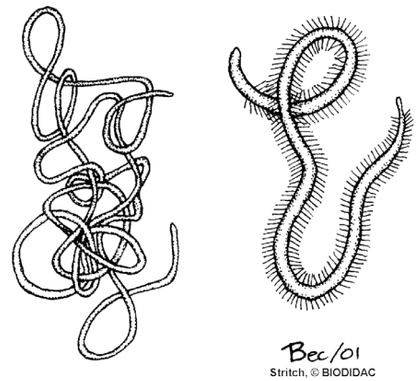
2015년 2월
한주성 드림

목차

story 1

APT 공격: 여치의 몸에 기생하는 연가시처럼

연가시는 곤충의 몸 안에 기생하여 곤충의 몸을 숙주 삼아 영양분을 빨아먹으면서 살아간다. 성충이 되면 숙주의 목이 마르도록 조종해서 물가로 유인한 후 숙주의 신체를 뚫고 나와 물 속에서 생활한다. 이때 숙주보다 더 길어진 연가시가 숙주의 몸을 뚫고 나오면서 숙주인 곤충은 큰 충격을 받게 되는데, 이 충격으로 대부분 죽거나 가사 상태가 된다. 이런 무서운 기생충이 인간에게 감염됐다는 보고도 있지만 일반적으로는 인간의 경우 감염되지 않는다는 게 참 다행이라 생각된다. 영화로도 만들어질 만큼 충격적인 곤충임에는 틀림없다.



[그림 1-1] 철사처럼 생긴 연가시

필자가 연가시 얘기를 들었을 때 소름이 끼치면서 생각난 해킹이 바로 APT(Advanced Persistent Threat, 지능형 지속적 위협) 공격이었다.

APT 공격은 특정 공격 대상을 정하고 목적이 달성될 때까지 지속적인 공격을 시도하고 직원이 이용하는 정상적인 접속 방법을 최대한 활용하는 방식이 특징이다. 특히 온라인 네트워크 환경을 이용해서 돈을 버는 게임 개발 회사 및 게임 퍼블리셔를 공략해서 속칭 금전적 빨대를 몰래 꼽을 수 있는 최고의 킬러 타이틀이라 할 수 있다.

APT 공격의 특징은 오랜 시간 숙주의 몸 속에서 생활하며 성충이 되기를 기다리는 연가시처럼 감염된 시스템을 통해서 개발중인 게임의 개발이 완료되기를 기다린다는 것이다. 그리고 개발이 완료되었음을 감지했을 때 백도어, 루트킷과 같은 부가적인 도구를 이용해 퍼블리셔의 게임 서버들을 조종하고 캐시 아이템이 관리되는 고퀄리티의 영양분인 데이터베이스를 점령해서 자신이 원하던 금전적 욕구를 채우기 시작한다. 그리고 시간이 지나 게임이 황폐해져 금전적 가치가 없어지면 유유히 빠져나가게 된다.

이처럼 게임은 해커가 쉽게 큰 돈을 벌 수 있는 수단인 만큼 특정 회사를 타깃으로 전용 바이러스를 만들어 백신에서도 검출되지 않는 경우가 많아 여러 게임 업체가 해킹을 당한지도 모르게 해커와 함께 일상을 보내고 있다고 할 수 있다.

그렇다면 이 무시무시한 APT 공격을 어떻게 하면 효과적으로 막을 수 있을까? 필자의 경험담을 통해 그 방법을 알아보도록 하자.

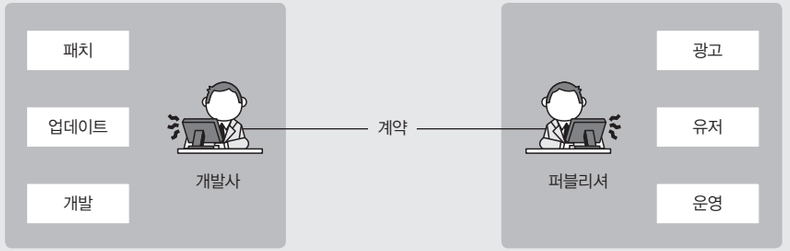
1.1 끝까지 기다린다

어느 여름에서 가을로 넘어가는 시기, 개발 실장이 조용히 나를 방으로 불렀다.

"OO 이슈 알고 있지? 우리가 좀 도와줘야 할 것 같다."

그건 다름 아닌 해외 퍼블리셔의 긴급 요청이었다.

>>> 참고 퍼블리셔란 고객에게 서비스를 진행하는 업체를 통칭하는 말로 해외에서 게임의 좋은 성과를 위해 각 나라별로 유명하고 고객 서비스를 잘하는 업체와 계약하여 게임 운영을 맡기고 수익을 나누어 갖는다. 국내 업체로는 NHN, 넷마블, 넥슨 등이 전문 퍼블리셔라 할 수 있다.



[그림 1-2] 게임 개발사와 퍼블리셔의 관계

이 상태라면 현지에서 더 이상 게임 서비스를 하지 못할 것 같다며, 해킹 사고를 해결할 수 있다면 모든 지원을 아끼지 않겠으니 지원해 달라는 요청이었다.

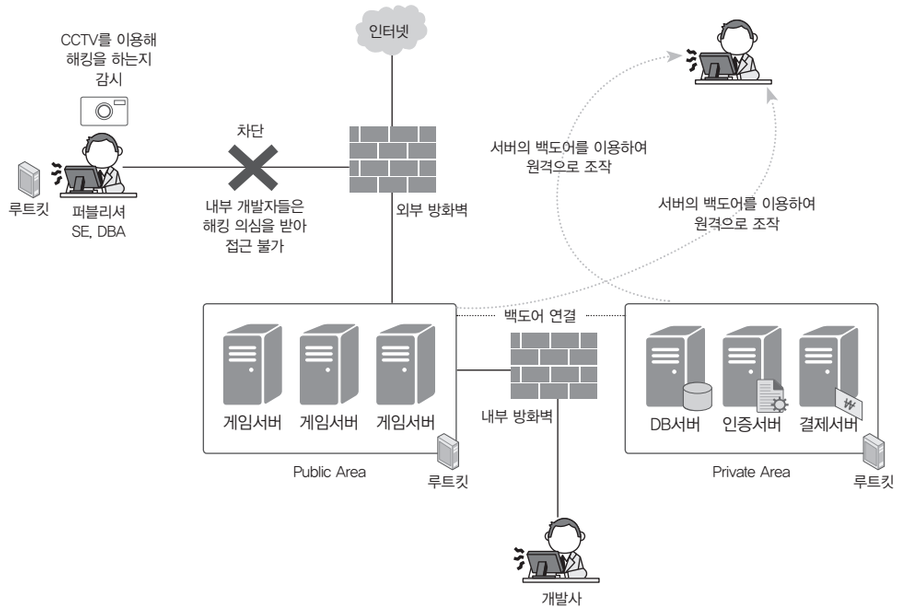
담당자에게 확인해본 결과 연초에 게임 내 해킹이 있음을 감지하고 해당 시스템을 외부 보안 업체에 의뢰하여 점검해보니 백도어는 물론 데이터베이스 조작 도구와 시간별로 실행되는 스케줄 작업 등 해킹 흔적이 서버에서 발견되었다는 것이다.

게임 시스템은 대부분 사설 망과 외부 망으로 구분하여, 중요도가 높고 외부 연결이 필요 없는 DB(Database, 데이터를 저장하고 관리하는 서버를 말한다) 및 빌링 서버들은 사설 망에 구축하고, 클라이언트와 통신이 필요한 웹 서버, 게임 서버들은 외부 망에 구축하여 보안을 위해 분리해 운영한다. 그런데, 내부에서만 접근할 수 있는 DB 서버에서 해킹 정황이 포착됐다는 것이다.

퍼블리셔는 이러한 해킹을 진행한 사람이 내부 소행일 수 있다고 생각해 먼저 데이터베이스를 조작할 수 있는 자사의 DBA(Database Administrator, 데이터베이스를 관리하는 인력)를 의심하며 DBA의 시스템 접근을 통제하고 CCTV를 통해 DBA 팀 전체를 감시하였다고 한다.

이렇게 감시를 하고 있는 와중에도 해킹은 계속되자 감시 범위를 확대해 게임을 관리하는 시스템 엔지니어들도 통제하여 퍼블리셔에서 게임 시스템의 접근을 차단하고 서버 접근 계정도 전부 차단하였지만 해킹은 계속되었고, 이제 게임 개발사인 우리도 의심을 하는 상황에 이르렀다.

외부 보안 업체에게 문제 해결을 부탁한 지 몇 개월이 지나도 해결되기는커녕 수 개월 째 더욱 더 깊은 나락으로 빠져 게임은 엉망이 되어 갔고 매출에도 큰 타격을 입은 상태로 대부분 포기하고 있는 상황이었다. 퍼블리셔의 게임 서버들은 이미 해커에 점령당해 관리자보다 상위 권한으로 서버를 관리하고 일반적인 방식으로 삭제할 수 없는 루트킷(관리자 권한으로 확인이 불가능한 백도어)이라는 도구가 전 서버에 퍼진 상태였다. 더욱이 서버를 관리하는 DBA는 물론 SE(시스템 엔지니어로 운영체제의 관리와 모니터링 등 인프라 관리를 담당한다)도 루트킷에 감염된 최악의 상황이었다.



[그림 1–3] 해외 퍼블리셔의 해킹 당시 상황

그 당시 담당자가 해킹 사고 분석을 통해 해커가 시스템에 남긴 흔적들과 해킹 당한 정황을 설명해 주는데, *"어, 이거 그거 아냐?!"*라는 생각이 머리를 스쳐 지나갔다. 번뜩하며 떠오른 것은 퍼블리셔의 지원 요청이 있기 바로 한달 전 내부에서 개발자가 *"누군가 컴퓨터를 사용한 것 같다"*라는 얘기를 듣고 내부 시스템 조사를 착수했던 기억을 떠올렸다.

당시 사고 분석을 위해 개발자 PC에 접근하여 가장 먼저 프로세스 익스플로러(5장 참고)를 이용해 이상한 프로세스가 있는지 확인했다. 기본적으로 설치된 프로세스와 회사에서 구입한 프로그램의 프로세스들은 제외한 결과, 무료 프로그램들이 몇개 나왔지만 특별한 프로세스는 찾을 수 없었다. 현 프로세스 현황을 캡처 하고서 네트워크 사용 현황을 확인하기 위해 TCPView(5장 참고)로 관찰하는데, 네트워크에서 이상한 점을 찾을 수 있었다. 외부 통신 시도를 할 이유가 없는 탐색기(Explorer.exe) 프로세스에서 메일 포트를 이용하여 외부 연결 시도를 하고 있었던 것이다.

[redacted]	8406	192.168.0.45	13000	TIME_WAIT
[redacted]	8421	chinacache.la.x...	smtp	SYN_SENT
[redacted]	8423	chinacache.la.x...	pop3	SYN_SENT
[redacted]	8425	chinacache.la.x...	pop3	SYN_SENT
[redacted]	8395	[redacted]	8080	TIME_WAIT
[redacted]	8396	[redacted]	8080	TIME_WAIT
[redacted]	8104	[redacted]	4027	ESTABLISHED
[redacted]	8120	[redacted]	https	ESTABLISHED

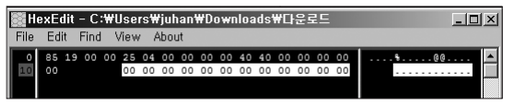
[그림 1–4] 수상한 네트워크 이용 현황 표착

해당 IP를 Whois(http://whois.kisa.or.kr/kor/)를 이용해 확인해본 결과 중국 상해에 위치한 IPS에서 관리하는 것으로 확인되었다..

[그림 1–5] IP의 근원지는 중국

>>> 참고 IPS는 회선을 제공하는 사업자인데, 네트워크 전송 특성상 제공하는 IP의 네트워크 회선의 효율을 높이기 위해 네트워크 이용 수요가 많은 곳이라면 제공한 IP 인근에 위치한다.

역시나 해킹 공격에서 자주 목격하게 되는 중국 IP였다. 즉 외부로 접근할 수 없는 개발자의 PC는 좀비 PC가 되어 해커의 서버와 외부 교신을 하고 있음을 확인할 수 있었다. 아래는 해커의 서버로부터 받아오는 데이터를 확인한 내용인데, 코드 자체만으로는 어떤 코드인지 의미를 알 수 없다.



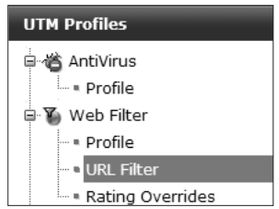
[그림 1-6] 알 수 없는 코드를 받고 있다.

정밀 분석 전에 보고를 위한 데이터를 만들기 위해서 현재 확인된 현상들을 기록해 놓고 개발자 PC의 네트워크 인터페이스를 비활성화하였다. 개발자에게는 PC를 재설치 하기를 권고하고 사내 네트워크의 접점에 위치한 방화벽에서 3개월 로그를 검색하여 해커의 서버 IP로 접근한 적이 있는 IP의 전체 리스트를 확인하였다. 내부 서버들이 해커의 서버와 교신한 것을 알 수 있었고, 개발망(폐쇄망, 외부 연결이 되지 않는 인트라넷 전용망)이 이미 본연의 기능을 상실한 것을 확인할 수 있었다.

이후 해당 IP의 사용자와 서버들에 대해서 네트워크 연결을 차단하고 감염된 PC를 이용하는 개발자들에게 현재 상황과 PC를 재설치 할 것을 요청한 후 정밀 분석을 진행하기 위해 감염된 것으로 확인된 서버 하나는 가상화 이미지로 생성하였다.

처음 분석을 진행하면서 가장 궁금한 점은 *"개발자가 사용하는 PC는 개발망인데, 어떻게 HTTP, DNS 등에 연결할 수 있는 걸까?"*라는 생각과 *"어떻게 감염된 것인지"*라는 것이었다. 이를 확인하기 위해 내부 시스템 구조의 히스토리를 확인하던 중 첫번째 물음은 쉽게 찾을 수 있었는데, 바로 개발자가 사용하던 인터넷 환경의 PC를 아무런 조치 없이 개발망으로 전환하여 사용했기 때문이었다. 즉 외부 인터넷 망을 사용하다 감염되었을 가능성이 성립되었다.

이외에도 방화벽 설정상의 문제점으로, 웹 접근의 부분적인 허용을 위해 사용하였던 웹 필터 부분에 문제가 있었다.



[그림 1-7] Fortinet 웹 필터링

웹 필터링은 도메인 주소를 이용해서 차단하거나 허용하는 방화벽의 보안 기능인데, IP로 요청하는 경우 이 웹 필터링이 동작을 하지 않아 연결이 가능하였다. 그리고 그 당시 DNS를 자체적으로 운영하지 않고 외부에 있는 KT의 DNS를 다수 이용하였는데, 개발망이라 하더라도 DNS 포트는 전체를 오픈하였다. 이런 관리적 실수가 더해져 해커는 이러한 회사의 네트워크 이용 지점을 치밀하게 분석하여 개발망에서도 외부 연결이 가능한 구조를 만들어 냈던 것이다.

알고 보면 단순하지만 분석 전에는 알 수 없는 것이 참으로 많다는 생각에 한숨을 쉬며 두 번째 답을 찾기 위해서 생성해 둔 서버 이미지를 이용해 정밀 분석을 진행하였다.

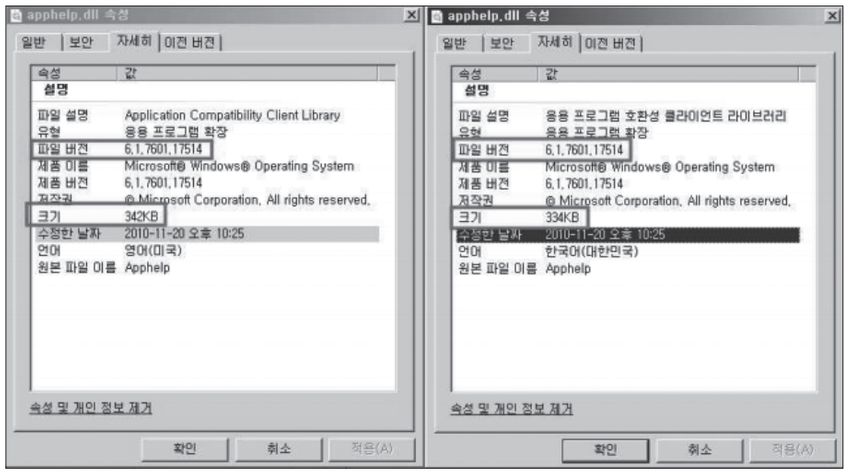
초기 분석을 통해 활동이 확인된 탐색기를 집중 분석하였는데, 탐색기에서 로드된 Dll의 네트워크 이용 상태를 확인하여 발생했던 패킷의 실행 스레드의 정체를 찾을 수 있었다.

범인은 바로 Apphelp.dll이라는 라이브러리. 파일을 숨김 속성으로 해놓았을 뿐만 아니라 라이브러리 설명도 정식 Apphelp.dll의 정보를 모두 넣어 놓는 치밀함에 혀를 내둘렀다(보통 바이러스 제작 시 Description까지 넣는 경우는 드물다).

더 당혹스럽게 만들었던 것은 파일의 생성 일자가 지금부터 2년 전에 만들어졌다는 것이다. 오랫동안 우리 회사를 지켜보며 활동한 것인지, 나보다 회사 내부 사정을 더 잘 알 수 있겠다는 생각까지 들었다.

해커가 만들어낸 Apphelp.dll은 C:\Windows\에 위치하고(정상 파일은 C:\Windows\System32에 위치), 파일 설명과 언어 속성이 영어라는 외관상 차이점이 있었다.

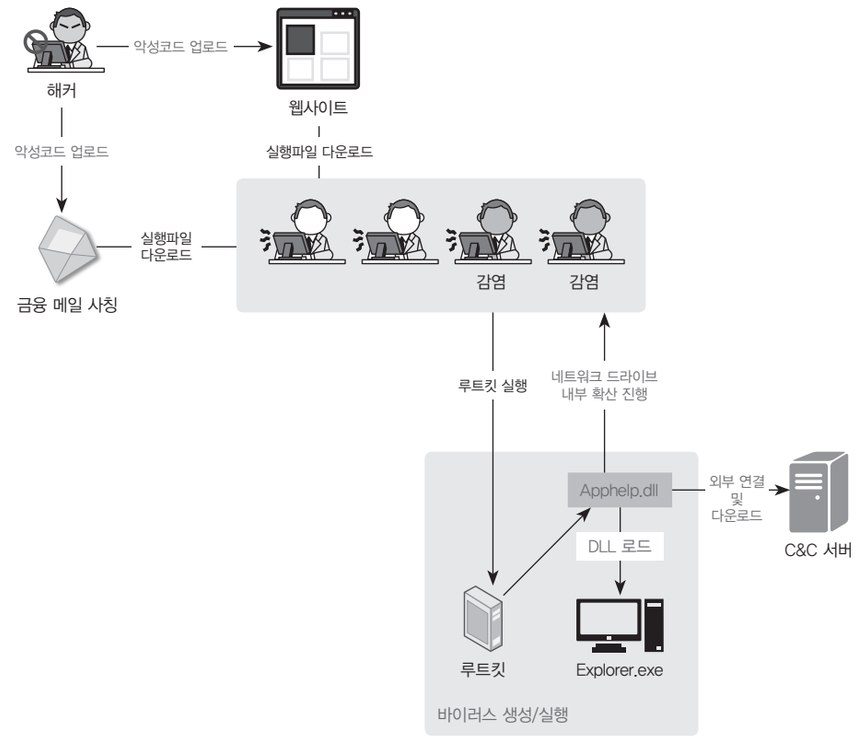
확인을 위해 파일의 이름을 바꾸고 탐색기를 재시작하자 더 이상 네트워크 연결 시도를 하지 않는 것을 확인하고 해당 라이브러리가 문제임을 확신하였다.



[그림 1-8] 해커가 생성한 Apphelp.dll(왼쪽)

즉 Apphelp.dll이 C&C 서버(Command & Control, 좀비 PC에게 명령을 내리고 제어하기 위한 서버)와 통신하며 해커로부터 명령을 전달 받고 실행하는 주체라는 것이다.

이 말은 회사 내 이용자가 외부 웹사이트나 메일의 악성 자료를 다운로드하여 감염되었을 것이며, 감염된 PC는 루트킷을 생성하고 이를 이용해 C&C 서버에 연결할 수 있는 Apphelp.dll을 시스템 프로세스인 Explorer.exe 실행시 로드하도록 했던 것으로 추측할 수 있다.



[그림 1-9] 사고 경위 추측

Apphelp.dll 이외에는 추가로 발견된 악성코드가 없어, 이미 루트킷은 자체적으로 삭제되었을 가능성이 높았다. 그리고 Apphelp.dll이 백신에도 잡히지 않는 것으로 보아 일반적인 악성코드가 아니며, 회사 내에서 사용하는 방화벽 구조를 이용해 통신하고 있는 점을 미루어 보아 우리 회사를 노리고 만든 APT 공격이라는 판단이 들었다.

APT 해킹 사고의 대응이 어려운 이유는 특정 업체를 노리고 기존 바이러스들을 변조하여 전용 바이러스를 만들다 보니, 바이러스 샘플을 통해서 엔진을 업데이트하는 백신 업체에서는 특정 업체를 공격하는 바이러스를 구하기 어려워 대부분의 백신 프

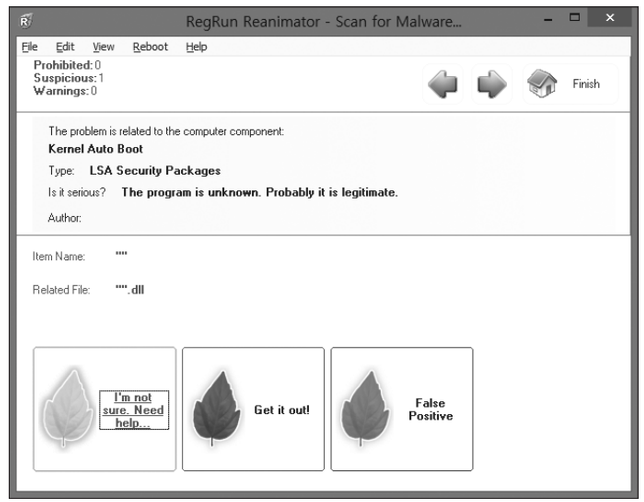
로그램이 전혀 이를 감지하지 못한다는 것이다. 상황이 이러한데, 대부분의 기업 보안 엔지니어들이 해킹 사고 시 백신만을 실행하고 별 다른 조치를 하지 않고 있는 현실은 매우 안타까운 부분이라 할 수 있다.

[그림 1-10] 바이러스에 걸린 상태이지만 백신은 안전하다는 결과를 보여주고 있다.

만약 해킹 사고가 발생하였는데 제대로 분석과 조치를 할 수 없는 상태라면, 침해사고 발생 시 해당 머신을 비롯한 동일 네트워크에 존재하는 전체 장비에 대해서는 전부 재설치를 진행하는 것이 가장 바람직하다. 해당 네트워크에 감염된 머신이 남아 있다면 다시 해커가 침입할 수 있는 가능성이 있기 때문이다. 이러한 단순해 보이는 작업이 빠르고 확실하게 치료할 수 있는 방법이다.

하지만 전부 재설치 진행할 수 없는 상태거나 공격 범위가 애매한 경우가 많다. 이때는 공격에 대한 명확한 분석을 통해서만 문제점을 해결할 수 있으므로, 해킹 사고에 대한 분석이 반드시 필요하다.

저자 역시 이 해킹 사고의 범위가 워낙 넓어 전체를 재설치한다는 것은 불가능에 가깝다고 느끼고, 여러 도구를 테스트해본 결과 치료가 가능한 도구를 찾을 수 있었는데, 바로 Unhackme(5장 참고)라는 루트킷 전문 백신도구였다. Unhackme를 이용해 해커가 숨겨놓은 루트킷를 찾아내고 이를 제거할 수 있었다. 실제 감염된 리스트의 PC에서 실행해본 결과 동일한 루트킷이 존재함을 확인하고 손쉽게 제거하는 큰 성과를 얻었다. Unhackme의 특징은 루트킷을 포함하여 시스템 시작 시 로드되는 개체를 확인해 해당 개체가 문제가 있는지 사용자에게 묻는 방식이다.



[그림 1-11] 의심되는 파일에 대해서는 "Get it out"을 선택한다.

Unhackme를 이용해서 이번 해킹에 이용된 루트킷을 제거하는 것이 가능함을 분석 PC(바이러스와 같은 악성코드를 분석하는 머신으로 분석으로 인한 피해가 외부로 확산되지 않도록 구성한 PC를 말한다. 주로 가상화 솔루션을 이용해 구성한다)에서 확인하고, 모든 서버들에 대해서 전체 검사를 진행하였다. 이를 통해 만에 하나 남아 있을지 모를 해커의 잔해를 속출해냈다.

그리고 추가적인 샘플 분석을 통해 확인된 감염 시스템은 다음과 같은 특성을 지니고 있었다.

1. DNS와 메일포트를 이용한 외부 통신
2. 작업 스케줄러에 작업 스크립트 등록
3. 네트워크 드라이브 연결이 있는 경우 네트워크 드라이브의 C:₩Windows₩System32에 파일 복사 시도

1. DNS와 메일포트로는 해커가 필요한 명령 혹은 현재 상태를 해커에게 전달하는 등 에이전트 역할을 수행하며,
2. 작업 스케줄러의 T-SQL 구문을 스크립트로 등록하여 게임 데이터베이스 연결 및 업데이트로 본인의 이익을 챙기는 구조였다.
3. 그리고 원격 시스템에서 네트워크 드라이브를 통한 파일 공유를 이용할 때 연결한 PC에 바이러스를 감염시켜 빠르게 확장 가능한 구조였던 것이다. 즉 네트워크 드라이브를 주로 이용하는 우리 회사에 알맞은 방식이었다.

퍼블리셔의 해킹 분석 내용도 확인해본 결과 나타나는 해킹 증상이 동일함을 알 수 있었으며, 동일한 해커가 해외 퍼블리셔까지 공격을 하고 있는 상황이라는 결론을 얻을 수 있었다.

이렇게 된 이상 해커가 날뛰는 것을 묵과할 수 없었다. 그리하여 퍼블리셔와 게임을 살리기 위해 2주간의 출장을 나가게 되었다.

현재의 게임 시스템은 운영이 불가능하다는 판단 하에 진행된 출장이다 보니 출장의 목적은 게임 시스템을 새로 구축하고 더 이상 해킹 사고가 재발하지 않도록 조치하는 데 초점이 맞춰져 있었다. 이러한 사명감을 가지고 현지에 도착하여 가장 먼저 진행한 것은 바로 연결 지점의 단일화와 2중화된 인증방식 구성이다(이 내용은 이 장의 뒷부분에서 자세히 언급한다).

먼저 현재 엔지니어에게 게임 시스템 접근에 이용하는 VPN(Virtual Private Network, 공유된 인터넷망을 특정 회사나 단체로 연결하기 위해서 사용하는 기술로 연결 시 해당 데이터는 암호화 하여 전송하여 보호된다)을 우리만 사용할 수 있는 별도의 IP를 따로 지정해 달라고 요청했다. 그리고 게임 서버에서는 지정된 VPN의 IP만으로 로그인이 가능하도록 서버 방화벽을 구성하여 관리 연결 지점을 최소화하였다. 이 작업과 동시에 기존 계정도 초기화하고 담당자를 지정해 개인별로 지급하여 최소의 인원만이 게임 서버에 접근할 수 있도록 함으로써 계정의 외부 노출 가능을 최소화하고 책임 구분을 명확히 했다.

여기에 자체 개발한 시스템 모니터링 프로그램도 설치하였다. 이는 시스템 로그인 시도를 메일로 알려주도록 하여 사전에 시스템 로그인 시도를 탐지하고 어떤 계정에 문제가 있는지, 어디에서 로그인 시도를 하는지를 확인할 수 있도록 하였다. 다음은 저자가 개발한 로그 서버 감시 프로그램이다. http://asecurity.so에서 확인할 수 있다.

[그림 1-12] Asecurity 에이전트의 관리 도구 화면

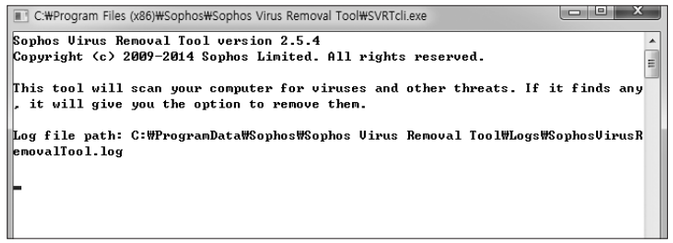
작업에 있어 가장 어려웠던 부분이 각 서버별 방화벽을 전부 구성한 점이라 할 수 있다. 이 부분에 가장 많은 시간을 투자하였는데, 이를 구성함으로써 연결 지점 단일화와 같은 네트워크 보안을 완성할 수 있었다. 아래 화면은 윈도우의 방화벽 정책을 구성하기 위해 사전에 만들어 놓은 CLI 명령 룰 셋이다(방화벽 CLI 명령 생성 방법은 링크 참고. http://support.microsoft.com/kb/947709). 이렇게 필요한 룰 셋을 미리 만들어 두면 작업 시 빠르게 적용이 가능하다.

```
netsh advfirewall firewall add rule name="[redacted]" action=allow protocol=TCP dir=in localport=80 profile=p
netsh advfirewall firewall add rule name="[redacted]" action=allow protocol=TCP dir=in localport=3389 profi
netsh advfirewall firewall add rule name="[redacted]" 510,8600" action=allow protocol=TCP dir=in localport=
netsh advfirewall firewall add rule name="[redacted]" action=allow protocol=TCP dir=in localport=139,445 prof
netsh advfirewall firewall add rule name="[redacted]" action=allow protocol=UDP dir=in localport=137,138 prof
netsh advfirewall firewall add rule name="[redacted]" action=allow protocol=TCP dir=in localport=25 profile=publi
netsh advfirewall firewall add rule name="[redacted]" action=drop protocol=TCP dir=out remoteport=1-1024 profi
netsh advfirewall firewall add rule name="[redacted]" action=allow protocol=TCP dir=out remoteport=1-1024 prof
netsh advfirewall firewall add rule name="[redacted] dir=out service=wuauserv

//Login
netsh advfirewall firewall add rule name="[redacted]" action=allow protocol=TCP dir=in localport=3389 profi
netsh advfirewall firewall add rule name="[redacted]" action=allow protocol=TCP dir=in localport=17000 profil
netsh advfirewall firewall add rule name="[redacted]4300" action=allow protocol=TCP dir=in localport=14300
netsh advfirewall firewall add rule name="[redacted]" action=drop protocol=TCP dir=out remoteport=1-1024 profi
netsh advfirewall firewall add rule name="[redacted]" action=allow protocol=TCP dir=out remoteport=1-1024 prof
netsh advfirewall firewall add rule name="[redacted] dir=out service=wuauserv
```

[그림 1-13] 사전에 구성할 방화벽 정책을 텍스트로 작성해 두었다.

이외 추가로 루트킷 전문 탐지 도구인 Sophos Virus Removal Tool(http://goo.gl/ddBpa)을 백신과 함께 설치하여 사용자가 로그인할 때마다 자동으로 전체 검사를 진행하도록 구성하였다.



[그림 1-14] CLI 명령을 이용해 예약 작업을 생성할 수 있다.

이렇게 시스템 재구성을 마치고 서비스를 오픈하였는데, 뜻하지 않은 문제점이 발견되었다. 바로 간헐적인 네트워크 지연이 발생하는 것이다. 기존에 없었던 문제점으로 게임에 로그인한 유저가 일정 수 이상으로 많아지면 게임 플레이를 하기 어려울 정도의 네트워크 순단(순간적으로 네트워크의 연결이 끊어지는 현상) 현상이 반복적으로 나타나 게임 서비스가 매우 불안정하였다.

개발자 및 엔지니어 등 많은 인원이 대기하며 문제점을 찾아보았지만 답을 찾을 수 없었다. 깊은 정적과 함께 시간은 점점 새벽으로 넘어가고 있었다. 시간이 의미 없이 흘러가자 한 개발자가 이야기를 했다.

"게임에는 문제가 없는 거 같아요, 윈도우에서 리소스를 많이 사용하는 거 아니에요?"

개발자가 할 수 있는 조치가 없다고 하여 시스템 리소스 쪽에 무게가 실리고 혹시나 하는 기대에 백신도 지워보았지만 나아지는 것은 없었다.

다음으로 의심되는 부분은 방화벽이었는데, 방화벽을 비활성화하면 시스템을 재구축한 의미가 없어지므로 방화벽 내 구성한 정책 수를 줄이는 방향으로 진행하였다. 실제로 방화벽을 구성할 때 윈도우 서버 2008 초기 버전의 방화벽 구성 방식이 매우 비효율적이었는데, 포트에 대한 범위 지정이 되지 않아 개별적으로 분리해 허용 정책

을 만들어 줘야 했고 게임 서버의 경우에 사용되는 포트가 많아 방화벽 정책이 800개 가량 들어갔다. 따라서 이 부분에서 문제가 발생할 가능성도 충분했다(이 문제점은 윈도우 서버 2008 서비스 팩에서 개선되었다).

방화벽을 경량화하기 위해 개별 IP로 지정하여 여러 개로 생성한 방화벽 정책을 게임이 사용하는 네트워크 서브넷(게이트웨이를 거치지 않고 서로 통신이 가능한 범위)으로 확장하여 구성을 변경하였다. 이렇게 변경이 가능했던 이유는 하나의 게임이 사용하는 네트워크를 분리하였기 때문에 방화벽 정책을 서브넷으로 확장하여도 보안상 위험은 크게 증가하지 않았다. 구성을 변경하자 방화벽 정책을 60% 가량 경량화할 수 있었고, 다시 서비스를 실행하였는데, 퍼블리셔의 PM(Project Manager, 프로젝트 담당자)이 환호를 외쳤다.

"랙이 없어졌어요!!"

드디어 보안과 게임 서비스상 문제가 없는 구성을 완료하였다며 서로 고생했다는 말과 함께 주말을 편안하게 쉴 생각에 들뜬 마음으로 호텔에서 잠을 청할 수 있었다.

다음날 늦게 일어나 호텔 로비에서 쉬고 있는데, 사업 담당자가 담배를 연달아 피우면서 고민에 빠진 표정으로 얼굴이 망가져 있는 게 아닌가. 걱정스러운 마음에 무슨 일 있냐고 묻자 고민을 덜어 놓듯이,

"동시 접속 자 수가 일정 수준 이상이면, 다시 랙이 발생한다고 하네요. 후~ 이거 해결 못하면 우리 집에 안 보내 줄 것 같은데, 어떡하죠?"

DBA에게 보다 자세한 내용을 들을 수 있었는데, 특정 시점에 패킷이 튄다는 것이다. 튄다는 의미는 일정 주기로 네트워크 연결이 지연된다는 것인데, 이제 더 이상 지체할 시간이 없다고 생각한 우리는 주말이지만 해결 방안을 찾기 시작했다.

방화벽 정책이 얼마 되지 않는 데이터베이스 서버에서 발생하는 문제로 네트워크 지연 문제가 확실하므로, Netmon을 이용하여 각 서버의 네트워크 지연을 확인해 보았다. 그 결과 게임 서버에는 문제가 없는데 데이터베이스 서버에서만 10개의 패킷 중

한두 개는 상당한 지연시간을 보이는 것을 확인했다. 이 결과를 토대로 윈도우 2008 서버에서 설치한 데이터베이스와 관련된 문제점을 인터넷을 이용해 검색해본 결과 취할 수 있는 네트워크 구성 몇 가지를 생각해 볼 수 있었다.

더 이상 게임에 문제가 장기화되어서는 안 된다는 생각에 주말이지만 다시 IDC (International Data Corporation, 전문적으로 서버와 네트워크 회선 등을 제공하는 시설)로 달려갔다. 그리고 점검시간을 이용해 진행할 작업을 정리하였는데, 적용할 내용은 다음과 같다.

명령 프롬프트 실행 내용

netsh int tcp set global chimney=disabled
데이터를 전송하는 작업을 CPU가 아닌 네트워크 어댑터를 이용하도록 하는 기술, 하지만 불안정 가능성이 높아 비활성화 추천

netsh int tcp set global rss=disabled
수신 데이터를 여러 CPU를 이용해 분산하여 처리하는 기술, 비활성화 추천

netsh int tcp set global autotuninglevel=disabled
자동적으로 네트워크 구성을 최적화하는 기능이지만, 이로 인한 지연 가능성이 있고, 기본적으로 구성된 수동 값이 더 효율적인 경우가 많아 비활성화 추천

레지스트리 수정 내용

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\Tcpip\Parameters]
"MaxUserPort"=dword:0000ffff ← 이용 가능 포트 수를 최대로 지정
"TcpTimedWaitDelay"=dword:0000001e ← 연결 지연시 유지 시간을 30초로 지정
"EnableTCPA"=dword:00000000 ← 네트워크에서 직접 메모리에접근하는 기술, 비활성화 추천
```

윈도우 2008에서 새로 추가된 기능들을 비활성화하는 작업인데 위 기능을 사용할 경우 네트워크를 사용할 때 준비 작업을 위해 네트워크 지연이 발생할 가능성이 높아 비활성화를 진행하면 보다 안정적으로 서비스를 할 수 있다.

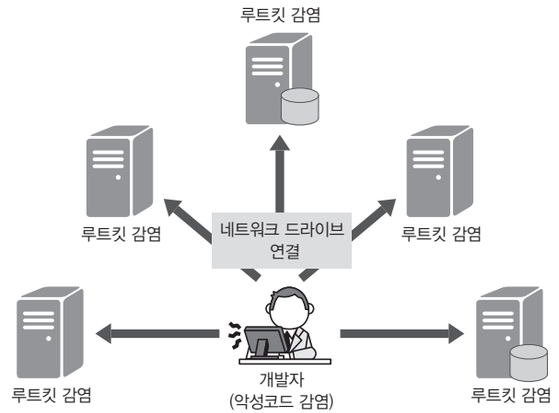
이렇게 만반의 준비를 하고 기다리던 점검시간이 다가와 준비한 설정을 적용하고 다시금 서버를 재기동 하였고 더 이상 게임에서 랙은 발생하지 않았다.

이후 사업담당자는 귀국할 때까지 웃는 얼굴이었으며, 우리는 퍼블리셔에게 우리가 적용한 모든 기술과 히스토리를 공유한 뒤 복귀하게 되었다.

지금도 가끔 이 해킹 사고를 생각하면 우리나라가 IT 강국이라는 것을 해외 업체에 보여줄 수 있었던 좋은 계기였던 것 같고 해커가 얼마나 치밀하고 끈질긴지 보안에 대해 다시 한번 생각할 수 있었던 계기가 되었다.

1.2 원인

저자가 경험한 APT 해킹 사고에서 이용된 루트킷 및 바이러스를 완전이 제거하기가 상당히 어려웠는데 그 이유는 쉽게 확장할 수 있는 시스템 구조에 있었다. 그림으로 확인해 보자.



[그림 1-15] 네트워크 드라이브 사용으로 인한 감염

위 그림과 같이 네트워크 드라이브를 연결하면 연결한 컴퓨터를 감염시키는 구조로, 감염된 서버를 이용하거나 감염된 PC가 서버에 연결할 경우 네트워크 드라이브를 이용해 쉽게 감염을 확산시키는 부분이 제일 큰 원인이라 할 수 있다. 원격 컴퓨터의 네트워크 드라이브를 연결하면 원격 컴퓨터에서 내컴퓨터의 드라이브 읽기/쓰기 모두