Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.27, NO.3, Jun. 2017 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2017.27.3.521

클라우드 기반 랜섬웨어 복구 시스템 설계 및 구현*

하 상 민,† 김 태 훈, 정 수 환‡
숭실대학교

Design and Implementation of a Cloud-Based Recovery System against Ransomware Attacks*

Sagnmin Ha,† Taehoon Kim, Souhwan Jung‡
Soongsil University

요 약

본 논문에서는 원본 파일뿐 아니라 외부 저장소의 백업파일까지 암호화하는 등 지능화 되어가는 랜섬웨어 공격에 대비하고자, 파일 생성 시점에 자동으로 클라우드 서버에 암호화하여 백업하고 클라이언트에서는 특정 프로세스가 원본 파일에 영향을 주게 되는 경우를 모니터링하여 차단하는 시스템을 설계하였다. 클라이언트에서는 파일 생성 혹은 저장 시에 프로세스 식별자, 부모 프로세스 식별자, 실행파일의 해쉬 값을 비교하여 whitelist에서 보호하고자 하는 파일 형식이 다른 프로세스에 의하여 변경이 발생하는 경우를 모니터링하여 차단함으로써 의심되는 행위에 대한 파일 변경을 방지하였다. 본 논문에서 제안하는 시스템을 적용하여 랜섬웨어에 의한 파일의 변경 혹은 삭제 시도로부터 안전하게 보호하여 발생가능 한 피해를 방지할 수 있도록 하였다.

ABSTRACT

In this paper, we propose a protection solution against intelligent Ransomware attacks by encrypting not only source files but also backup files of external storage. The system is designed to automatically back up to the cloud server at the time of file creation to perform monitoring and blocking in case a specific process affects the original file. When client creates or saves a file, both process identifiers, parent process identifiers, and executable file hash values are compared and protected by the whitelist. The file format that is changed by another process is monitored and blocked to prevent from suspicious behavior. By applying the system proposed in this paper, it is possible to protect against damage caused by the modification or deletion of files by Ransomware.

Keywords: Malware, Ransomware, Vaccine

I. 서 론

랜섬웨어는 몸값을 의미하는 Ransom 이라는 단어와 제품을 의미하는 Ware 라는 단어의 합성어로써 사용자 동의 없이 전자 기기 매체에 담겨져 있는 사용자 특정 파일들을 암호화 하거나 시스템 제어권 등을 획득한 후, 원래의 상태로 되돌리는 과정에서 금전적 이익을 요구하는 악성코드의 한 종류이다.랜섬웨어가 발견되는 플랫폼은 PC, 모바일 및 IoT 제품이며 Microsoft사의 Windows 운영체제를 대상으로 한 랜섬웨어가 가장 많은 피해를 유발하고 있다. 또한 최근 Apple사의 OSX와 모바일 플랫폼인

Received(02. 23. 2017), Modified(1st: 05. 08. 2017 2nd: 06. 07. 2017), Accepted(06. 08. 2017)
* 이 논문은 2017년도 정부(미래창조과학부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.2016-0-00078, 맞춤형 보안서비스 제공을 위한 클라우드 기반 지능형 보안 기술 개발, 한 본 연구는 미래창조과학부 및 정보통신기술진흥센터의 대학ICT연구센터육성 지원사업의 연구결과로 수행되었음 (IITP-2017-2012-0-00646)
† 주저자, hcjswo@nate.com
‡ 교신저자, souhwanj@ssu.ac.kr(Corresponding author)

Android, IoT를 목표로 한 랜섬웨어가 출현함에 따라 랜섬웨어에 완벽하게 안전한 플랫폼은 없는 것으로 보여진다[1].

랜섬웨어는 미국을 중심으로 활동하는 악성코드였으나 인터넷을 사용하는 시스템의 보급이 증가함에 따라, 전 세계적으로 급속하게 유포되었으며 2015년 4월에 국내를 목표로 한 CryptoLocker가 출현하였다[2]. 이후 발견된 CryptoLocker의 변종인 Nitol은 랜섬웨어 기능외에 DDoS 공격을 일으키며, 2016년에는 JavaScript 형태의 랜섬웨어도 출현하였다[3]. 2016년 3분기에는 랜섬웨어 피해를 경험한 인터넷 사용자가 전 분기 대비 두 배 이상 늘어난 82만명에 이르며, 미국에서만 2016년 한해 동안 랜섬웨어에 의한 피해가 10억 달러로 예상되고, 국내 피해도 지속적으로 증가될 전망이다. 또한, PC 외에도 다양한 모바일 단말기가 출시됨에 따라 각각의 플랫폼을 대상으로 악성행위를 수행할 수 있는 랜섬웨어의 특징 상 그 피해규모는 상당할 것으로 예상된다.

국내에서 랜섬웨어의 감염으로 인한 금전적인 피해가 증가함에 따라 많은 보안업체에서 다양한 랜섬웨어 방지 앱들이 출시되고 있다. 하지만 인터넷 브라우저의 취약점인 Drive By Download, 악성코드가 삽입된 문서 파일을 이메일에 전파 하는 등 여러 가지 감염 경로가 존재하기 때문에 사용자의 데이터를 백업하는 방법 외에는 완벽하게 랜섬웨어를 차단하기는 어렵다.

랜섬웨어로 인한 금전적인 손실 등의 2차 피해를 예방하기 위해 본 연구에서는 사용자의 데이터를 백업하는 클라우드 서버와 시스템을 모니터링하는 클라이언트로 구성된 랜섬웨어 복구 시스템을 구축하였다. 본 논문에서 제안하는 시스템을 이용하면 사용자의 파일들이 생성될 때 자동으로 클라우드 서버에 백업되고, 랜섬웨어가 사용자의 파일에 접근할 때 해당 프로세스를 차단하여 랜섬웨어에 의한 피해를 방지할 수 있다.

서론에 이어 본 논문은 2장에서 배경 지식으로 랜섬웨어에 대해 소개하고, 3장에서는 본 논문에서 제안하는 랜섬웨어 복구 시스템의 구성과 클라우드 서버 및 클라이언트의 동작 방식에 대해 설명한다. 4장에서는 랜섬웨어 복구 시스템의 분석 결과를 제시하며, 마지막으로 5장에서 결론을 맺는다.

II. 배경 지식

2.1 랜섬웨어의 배포

랜섬웨어는 주로 신뢰할 수 없는 사이트를 방문하여 브라우저를 통해 감염되거나 메일의 첨부파일 혹은 파일공유 사이트를 통해 주로 감염된다[6].

2.1.1 브라우저를 통한 감염

해커가 보안이 취약한 웹사이트를 변조하고 사용자가 브라우저를 통해 취약한 사이트 방문 시 취약점에 의해 사용자 모르게 악성코드를 설치하고 실행하게 하는 방법으로 불특정 다수를 대상으로 하는 악성코드 전파시 사용되는 방법을 Drive By Download 라고 한다. 주로 웹 브라우저 내에서 동작되는 JavaScript 언어를 이용하여 Adobe Flash Player, Java 같은 다수의 사용자가 사용하는 소프트웨어의 취약점을 이용하고 있다. 또한 보안장비인 WAF나 IPS의 룰을 회피하거나 분석가의 분석을 방해하고 악성코드의 생존력을 높이기 위해 취약점이 포함된 JavaScript 소스코드를 난독화 하기도 하는데, 사용하는 운영체제와 브라우저 및 관련 소프트웨어의 보안 패치를 항상 최신으로 업데이트 하여 대비를 하지만 Zero-day 형태의 공격에는 완벽히 막을 수 없어 보안 관리가 미흡한 사이트 방문을 이용 자제하여야 한다.

2.1.2 스팸 메일을 통한 감염

스팸 메일로 인한 랜섬웨어 배포 방식은 이메일에 문서안의 매크로나 JavaScript 파일 또는 실행파일들을 첨부하여 사용자의 실행을 유도해서 악성코드를 실행하거나 취약점이 포함된 사이트 링크를 이메일 내용에 포함하여 악성코드 감염을 유도하는 방법이다. 특정 타겟을 대상으로 전파되는 방식이며, 출처가 불분명한 이메일을 수신하였을 때 첨부파일을 바로 실행하지 말고, 해당 이메일을 삭제하는 것을 권고한다.

2.1.3 토렌트 및 P2P 사이트를 통한 감염

토렌트나 P2P사이트의 다운로드 런처 프로그램을 설치할 때 자동으로 같이 설치되는 광고프로그램 또

는 업데이트 프로그램 및 무료 게임 프로그램 등에 포함되어 악성코드를 배포 방식이다. 이러한 방식으로 감염되는 랜섬웨어는 불특정 다수를 대상으로 하기 때문에, 이와 관련 사이트 방문을 자제 하고 정식으로 유통되는 프로그램을 통해 사용해야 한다.

2.2 랜섬웨어의 공격 대상

랜섬웨어는 주로 디렉토리 내 사용자의 문서나 사진 등을 대상으로 암호화시키며, MBR을 오염시켜 부팅을 방해하거나 시스템 사용자의 중요파일이 저장된 디스크를 암호화시키는 비트로커를 사용한다[7].

하드디스크의 첫 번째 섹터에는 하드디스크에 할당된 파티션(볼륨)의 위치 정보와 시스템을 부팅하는데 있어 중요한 부트 코드가 포함된 MBR(Master Boot Record) 영역이 있다. 이 부분이 오염되면 해당 시스템은 정상적으로 부팅이 되지 않으며, PETYA[8]나 MISCHA 같은 악성코드들의 공격 대상이 Fig.1.은 시스템이 PETYA에 감염되어 부팅하였을 때 나타나는 화면이다. MBR이 오염된 시스템은 VBR(Volume Boot Record) 영역에 백업된 MBR을 토대로 복구가 가능하다.

윈도우 시스템 복원 기능은 마이크로소프트사의 Windows Me 이후에 추가된 기능으로 시스템 파일, 레지스트리 키, 설치된 프로그램 등을 복원 시점으로 되돌리는 기능을 가지고 있다. 하지만 최근에 출현하는 랜섬웨어에는 시스템 볼륨 정보를 삭제하여 랜섬웨어로 인해 시스템이 감염되었을 때 사용자가 시스템 복원 기능을 통해 이전 시스템으로 복원 할 수 없게 만든다[9].

특정 확장자를 대상으로 하는 랜섬웨어는 개발 과정에서의 소스코드 및 업무파일, 개인 사진 파일, 기업의 문서 파일 등을 대상으로 암호화가 진행되며 가장 많은 피해를 입히고 있다

Fig. 1. Boot messages after MBR contamination by PETYA

2.3 랜섬웨어의 동작 흐름

특정 파일을 대상으로 하는 랜섬웨어는 악성코드 실행 이후로 디렉토리 탐색, 파일 암호화, 암호화키 전송, 경고 및 지불 의 행위 순서를 가지고 있다.

랜섬웨어가 실행되면 파일 시스템 내의 디렉토리와 파일을 검색하게 되며 해당 파일의 확장자가 랜섬웨어가 타겟으로 하는 확장자 일 때 해당 파일을 열고 데이터 내용을 암호화 한다. 사용되는 암호화 방식은 대칭키 또는 비대칭키 암호화 방식이며, 암호키를 생성한 후 파일을 암호화하고 암호키를 외부의 서버로 전송하게 된다. 사용자나 악성코드 분석가가 랜섬웨어 분석을 통해 복호화를 시도하면, 암호키를 알 수 없어 원본파일을 복호화 하기 어렵다.

랜섬웨어가 파일을 암호화하는 과정에는 원본파일의 데이터 영역에 암호화 된 데이터를 덮어 쓰는 방식이나 기존 원본파일을 암호화 하여 새로 파일을 생성하고 원본파일을 지우는 방식이 있다.

2.3.1 파일 덮어쓰기

Fig. 2.는 파일시스템 내의 Priavate.data가 랜섬웨어의 대상이 되어 암호화 될 때 해당 파일이 파일시스템 내의 Private.data의 데이터가 담겨진 기존 영역에 암호화 된 데이터가 덮어 씌워지게 된다. 이러한 방식을 사용하는 랜섬웨어에 감염될 경우 암호화 된 파일이 파일시스템 내의 데이터 영역이 중복되어 기존 데이터에 접근할 수 없기 때문에 파일 복원이 어렵다.

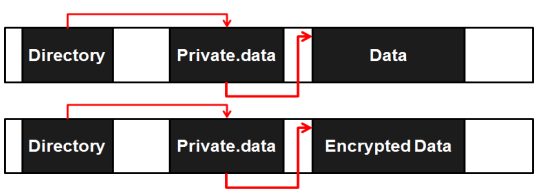


Fig. 2. File Overwrite

2.3.2 파일 생성 후 기존 파일 삭제

Fig. 3.은 원본 파일 데이터를 메모리에 로드 하여 메모리 내에서 암호화 한 후 파일시스템의 다른

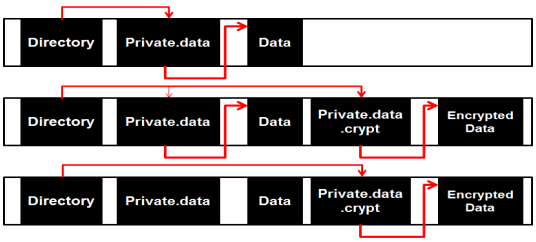


Fig. 3. File create after delete

영역에 암호화된 데이터를 저장하고 원본파일을 삭제하는 방식이다. 이러한 방식을 사용하는 랜섬웨어에 감염될 경우 원본 파일의 데이터 영역을 토대로 원본 파일을 복구할 가능성이 존재한다.

2.4 경고 및 지불

Fig.4.는 랜섬웨어에 감염된 사용자에게 악성코드 제작자는 암호화 결과를 알리는 경고창이며, 복호화 과정을 보여주며 금전적인 요청을 한다. 랜섬웨어 제작자는 금전 결제 방식을 익명성이 확보되어 수사기관의 계좌추적이 어려운 전자화폐인 비트코인(BitCoin)을 주로 이용한다.

Fig. 4. Warning screen due to Crypt0L0cker infection

2.5 랜섬웨어 탐지 방법.

국내 백신 제품을 가지고 있는 A사에서는 랜섬웨어를 탐지하기 위해 루트 디렉토리에 임의의 디렉토리를 생성 후 랜섬웨어의 타겟이 되는 doc, jpg, ppt 파일을 임의로 생성해서 해당 파일의 변화 시도를 탐지하는 행위 기반 진단 사용 기법을 이용하고 있다[10]. 하지만 루트 디렉토리의 하위에 있는 디렉토리의 파일을 대상으로 행위 기반 탐지를 하기에 '아이랜섬(iRansom)'과 같이 바탕화면에 있는 파일들만을 대상으로 한 랜섬웨어를 완벽히 막을 수 없다.

III. 제안하는 시스템

본 논문에서 제안하는 랜섬웨어 복구 시스템은 모니터링을 수행하는 클라이언트와 인증 및 키 관리를 수행하는 인증 서버, 해당 파일들을 암호화하여 백업을 수행하는 클라우드 서버로 구성되어 있다.

3.1 클라이언트

클라이언트 프로그램은 사용자 시스템에 설치하여 해당 시스템에서 각 프로세스에서 동작하는 파일의 쓰기 시도를 모니터링하고 정상·비정상 행위를 탐지하여 클라우드 서버에 백업하거나 랜섬웨어를 차단한다.

클라이언트 프로그램은 시스템 내에서 디렉토리 핸들의 변화가 있으면 해당 위치와 이벤트 종류까지의 제공함으로써 디스크 내의 파일 변화를 모니터링을 할 수가 있다. Fig.5.의 파이썬 코드는 Windows API 함수 중 CreateFile 함수로 C드라이브의 핸들을 얻어낸 후 ReadDirectoryChanges 함수로 대상 디스크 드라이브를 모니터링하고 C 드라이브와 하위 폴더 내에서 변화되는 파일의 이벤트(생성, 쓰기, 읽기 등)를 확인할 수 있다. 어떤

```
import win32con
import win32file
from ntsecuritycon import FILE_LIST_DIRECTORY

h_directory = win32file.CreateFile("c:\\",
                    FILE_LIST_DIRECTORY,
                    win32file.FILE_SHARE_READ |
                    win32file.FILE_SHARE_WRITE |
                    win32file.FILE_SHARE_DELETE,
                    None,
                    win32con.OPEN_EXISTING,
                    win32con.FILE_FLAG_BACKUP_SEMANTICS,
                    None
                    )
while True:
    results =  win32file.ReadDirectoryChangesW(h_directory,
                    1024,
                    True,
                    win32con.FILE_NOTIFY_CHANGE_FILE_NAME |
                    win32con.FILE_NOTIFY_CHANGE_DIR_NAME |
                    win32con.FILE_NOTIFY_CHANGE_ATTRIBUTES |
                    win32con.FILE_NOTIFY_CHANGE_SIZE |
                    win32con.FILE_NOTIFY_CHANGE_LAST_WRITE |
                    win32con.FILE_NOTIFY_CHANGE_SECURITY,
                    None,
                    None
                    )
    for action, filename in results:
        if action == 1:     print "[+] Created %s" % filename
        elif action == 2:   print "[-] Deleted %s" % filename
        elif action == 3:   print "[*] Modified %s" % filename
        elif action == 4:   print "[>] Renamed from: %s" % filename
        elif action == 5:   print "[<] Renamed to: %s" % filename
        else:               print "[???] Unknown: %s" % filename
```

Fig. 5. File change monitoring source code using ReadDirectoryChanges API

프로세스가 어느 파일을 핸들링하는지, 랜섬웨어에 감염되어 파일의 변화가 있는지 모니터링하기 위해 API 후킹 기술을 사용하였다.

시스템에는 여러 프로세스가 동작 중이며, 모니터링 과정에서 정상 프로세스가 파일을 핸들링 하는지 비정상 프로세스가 파일을 핸들링 하는지 여부를 확인하기 위해 각각의 프로세스를 핸들링 하는 과정이 필요하다. ReadDirectoryChange 함수를 통해 모니터링을 할 경우 어느 프로세스가 어느 파일을 핸들링 하는지 확인이 불가능하기 때문에, CreateProcess와 CreateFile API를 후킹하고 새로 생성되는 프로세스가 어느 파일을 사용하였는지 확인하였다. 이 과정에서 발생한 동작에 따라 해당 프로세스를 종료하고 모니터링 하는 디버거를 제작하였다. 모니터링 하고자 하는 프로그램을 실행하고 CPU의 Instruction Pointer 레지스터가 모니터링 하고자 하는 함수의 주소를 나타낼 때 해당 프로세스의 제어권을 디버거에게 넘겨서 스택 메모리 정보와 레지스터 정보를 조작하도록 하였다.

윈도우 시스템에서는 Explorer.exe 프로세스가 리눅스 계열의 Shell 역할을 하고 있다. Explorer.exe는 윈도우 운영체제의 설정과 정보가 저장되어 있는 레지스트리를 통해 확장자마다 실행되어야 하는 프로그램을 연결하여 프로그램을 실행시키며, 윈도우에서는 프로그램이 실행 될 때 Kernel32.dll을 통해 CreateProcess API를 호출한다. 해당 API를 후킹하여 새로 생성할 프로세스의 실행파일 이름인 첫 번째 파라미터와 해당 프로세스에 인자를 전달할 때 사용하는 두 번째 파라미터를 확인하면 윈도우 시스템에서 사용자에 의해 실행되는 프로세스의 모니터링이 가능하다.

Fig.6.은 모니터링 프로그램을 통해 시스템에서 관리하는 시스템 프로세스을 예외처리하고, 랜섬웨어의 대상이 되는 확장자마다 실행되어야 하는 프로그램들을 정의한 테이블을 참조하여 프로세스를 후킹하는 과정을 흐름도로 나타낸 것이다. 이후에 Explorer.exe 후킹과 기존에 실행되고 있는 프로그램 정보 획득을 통해 사용자에 의해 실행되는 프로그램들을 테이블에 등록하여 관리하도록 하였다. Table 1.은 레지스트리를 참고하여 랜섬웨어의 공격 대상이 되는 확장자들이 실행되는 프로그램들을 정리한 테이블이다.

본 시스템은 Fig.7.과 같이 후킹된 프로세스가 실행되어 쓰기 모드로 보호되어야 할 확장자를 가진 파일에 접근하고자 할 때, 프로세스 정보를 담은 테이블을 참조하여 정상인 경우와 비정상인 경우로 나누어 결과를 도출하였다.

사용자 윈도우 시스템의 바탕화면에서 1.hwp를 사용자가 마우스로 더블 클릭을 하게 되면, 해당 파일의 확장자 hwp를 통해 레지스트리의 .hwp 컴포넌트를 참조하여 해당 hwp의 연결프로그램의 경로인 "c:\program files\hnc\hwp80\hwp.exe"을

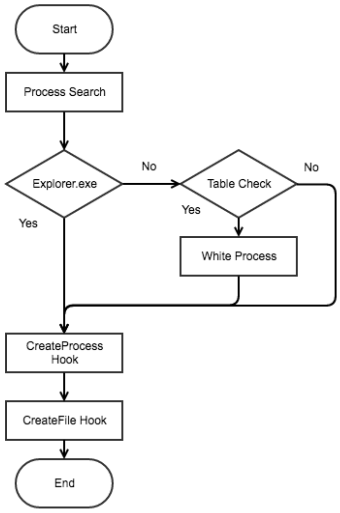


Fig. 6. Process hooking flow diagram

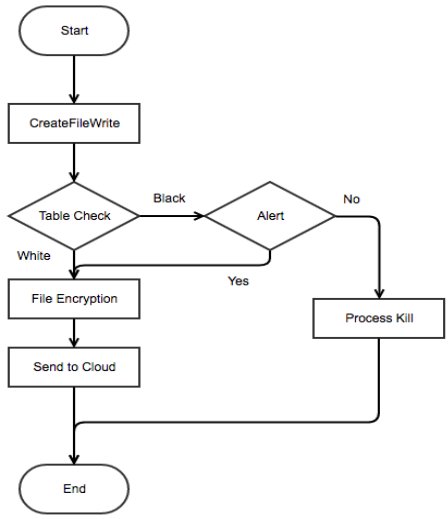


Fig. 7. The flowchart that is executed when the hooked process is in the write mode

Table 1. Whitelist table referencing registry

Ext	Program
txt	c:\windows\system32\notepad.exe
jpg	rundll32.exe c:\windows\system32\shimgvw.dll,imageview_fullscreen
jpeg	rundll32.exe c:\windows\system32\shimgvw.dll, ImageView_Fullscreen
hwp	c:\program files\hnc\hwp80\hwp.exe
doc	c:\program files\windows nt\accessories\wordpad.exe
pdf	c:\program files\adobe\reader 11.0\reader\acrord32.exe
avi	c:\program files\windows media player\wmplayer.exe /prefetch:8 /open
zip	rundll32.exe zipfldr.dll, RouteTheCall
png	rundll32.exe c:\windows\system32\shimgvw.dll,imageview_fullscreen

얻게 된다. 이 과정에서 hwp 확장자를 핸들링하는 "c:\program files\hnc\hwp80\hwp.exe"를 WhiteProcess Table에 등록한다. hwp.exe가 실행되고 hwp확장자를 가진 문서파일들이 쓰기 모드로 핸들링 되어 수정될 때, 클라우드 서버에 암호화한 후 저장한다. 만약 랜섬웨어에 감염되어 사용자의 파일이 암호화되었을 때, 클라우드 서버에 복원 요청을 하여 원본파일을 복구할 수 있다.

3.2 인증서버

3.2.1 데이터 베이스

클라이언트로부터 요청받는 데이터를 안전하게 저장하기 위해 Table.2와 같이 정리하였다. User의 가입 정보와 Login 시점에서의 정보, File의 정보를 기반으로 사용자에게 키를 제공하여 파일의 존재 유무를 확인하여 저장소에 새로운 파일 등록 여부를 결정하도록 하는 기능을 제공한다.

Table 2. Data requested from client

Table	Column	Contents
User	user_id	user index
	user	user ID
	pw	user password
	joinDateTime	sign up time
Login	login_id	login index
	user_id	user index
	loginDateTime	login time
	key	Keys to be created upon login
File	file_id	file index
	user_id	user index
	login_id	login index
	file_location	source file path
	data_hash	source file hash

3.2.2 인증 및 저장 과정

클라이언트 프로그램을 처음 실행 하게 되면 인증서버에 SSL 통신을 이용하여 ID, Password를 User 테이블에 저장함과 동시에 가입 시점이 저장이 된다. 이후 Fig.8.과 같이 로그인을 시도하면 ID, PW를 기반으로 인증을 완료하게 된다. 이후 파일의 저장을 위하여 Login 테이블에 로그인 시기 정보와 User 테이블의 가입 시기 정보인 joinDateTime을 조합해서 1차 암호키를 생성하여 사용자에게 전달하게 된다. 키를 전달 받은 사용자는 파일 저장 이벤트가 발생할 시 파일 경로와 원본 해쉬를 통해 해당 사용자의 기존 저장 여부를 확인 한 후 클라이언트는 시스템 고유정보인 시리얼 넘버와 uuid[13]를 이용하여 인증서버로부터 받은 키를 조합하여 새로운 암호키를 생성하게 된다. 이 키를 통해 파일 암호화 과정을 진행 한 후 클라우드 저장소에

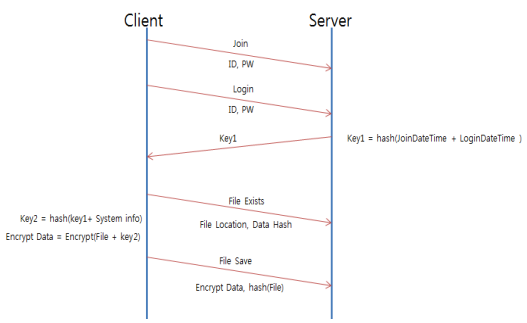


Fig. 8. File Store Flow

파일을 저장하게 되며 인증서버는 해당 파일의 경로와 원본 파일의 해쉬를 저장하는 과정을 거치게 된다.

3.3 클라우드 저장소와 파일 복원

기존 디스크에서 데이터를 관리할 때의 문제점은 하나의 디렉토리 안에 동일한 파일명을 여러 개를 가질 수 없으며, 하나의 파일에 대해 랜섬웨어 또는 사용자의 실수로 변경 되었거나 중첩 저장으로 인해 파일이 덮어 씌워졌을 경우 기존 내용을 얻을 수 없다는 점이다. 이러한 문제를 극복하기 위해 본 시스템에서는 Fig. 9과 같이 클라이언트부터 암호화 되어 클라우드 서버로 전송된 파일들을 "/사용자 ID/파일 패스 Hash/파일 데이터 Hash"와 같은 구조로 정의하여 저장한다. 사용자가 "c:\Temp.hwp" 파일을 내용이 다르게 4번 저장을 했을 때, 클라우드 서버에 각각의 데이터 내용에 따른 Hash값으로 저장하고 저장된 시점과 파일 이름을 DB화하여 관리하게 된다.

따라서 클라우드 서버에 하나의 파일이 같은 디렉토리 경로에 중복으로 저장되더라도 데이터의 Hash값으로 파일이름이 저장된다. 다양한 파일의 원본을 저장할 경우, 저장소의 용량 문제 및 효율성 저하로 인한 문제가 발생할 수 있어, 최대 용량 제한 및 최대 저장 횟수를 지정하여 초기 저장 파일부터 순차적으로 지우고 최근 파일을 저장하는 방식을 적용하였다. 사용자는 원본파일이 필요한 경우 클라우드 서버에 복원 요청을 하여 저장된 시점을 토대로 원본 파일을 얻게 되는데, 그 절차는 Fig. 8과 같다. Fig.7에서 설명한 바와 같이 클라이언트는 인증키를 전달

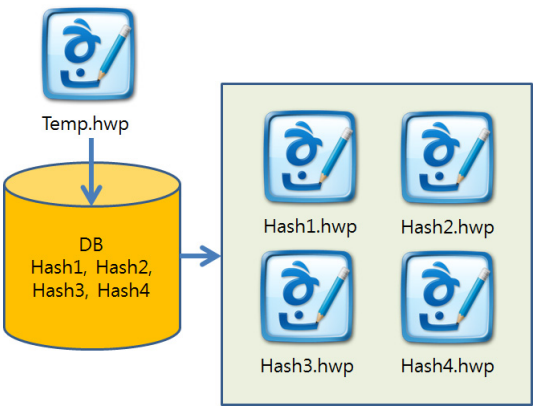


Fig. 9. The method to manage multiple files with the same file name in the cloud

받고 프로세스 모니터링을 통해 파일들을 저장해야할 이벤트가 받게 되면 파일의 위치와 파일 데이터를 전달받게 된 키와 시스템만의 고유값을 통해 암호화하여 클라우드 저장소에 전송을 하게 된다. 이후, Fig.8에서 설명된 복호화 과정은 사용자로부터 어느 파일이 존재하는지 리스트를 요청하게 되면 서버는 어느 시점에 몇 개의 파일이 있는지를 돌려주게 되고, 사용자는 특정 파일을 선택하여 복구를 요청하게 된다. 이때 서버에서는 요청받은 파일의 Login테이블과 User테이블을 참조하여 서버에 저장하였을 당시의 1차 암호키를 생성하여 암호화된 파일 데이터와 함께 사용자에게 돌려주게 되고, 사용자는 1차 암호키와 시스템 고유정보를 혼합하여 복호화 키를 생성한다. 이 복호화 키를 이용하여 암호화된 데이터를 복호화하여 원본파일을 얻을 수 있게 된다.

3.4 관리 테이블

프로세스 관리 테이블에 등록되지 않은 프로그램에 의해 파일 접근 이벤트가 발생하면 사용자에게 경고창을 띄운 후 접근을 허가할 것인지 물어본다. 특정 확장자가 다른 프로그램에 의해 접근되어야 할 필요가 있을 경우 사용자는 프로세스 관리 테이블을 수정하여 다른 프로그램이 특정 확장자에 접근할 수 있도록 하거나 새로운 확장자와 정상 프로그램의 관계를 등록하여 사용할 수 있다.

또한 파일에 접근하는 프로세스의 Hash값과 부모 프로세스를 비교하는 부분을 추가적으로 구현하여 악의적인 프로세스가 관리 테이블에 등록된 프로그램 이름으로 변경하여 파일에 접근하는 경우를 방지하였다. 예를 들어, hwp 문서는 whitelist 테이블 안에 있는 "c:\program files\hnc\hwp80\hwp.exe" 프로세스가 접근하도록 허용하는데, 악성코드가 해당 프로그램을 대체하여 파일에 접근할 수 있다. 이 때 접근하는 프로세스를 whitelist 테이블에 등록된 프로그램의 이름과 Hash값, 부모 프로세스를 비교하여 일치하지 않는다면 경고창을 띄운다. Whitelist 테이블에는 모니터링 할 프로세스와 해당 프로세스 모니터링을 통한 확장자 보호 기능을 이용하여 랜섬웨어의 경로와 이름을 변경하여 공격하는 정보에 대하여 해당 프로세스의 부모 프로세스와 실행파일의 해시값 확인을 통하여 정상 실행되었는지 비교하기 때문에 실행파일의 변경에도 대응할 수 있다.

IV. 실 험

랜섬웨어의 공격 대상이 되는 hwp 확장자를 가지는 문서 파일은 한글과 컴퓨터사의 한글 워드프로세스 프로그램에 의하여 사용되고 있으며, “c:\program files\hnc\hwp80\hwp.exe” 프로세스가 hwp 문서 파일을 핸들링 하도록 레지스트리에 등록되어 있다. 본 논문에서 제안한 랜섬웨어 복구 시스템을 구현하여 레지스트리에 등록된 “c:\program files\hnc\hwp80\hwp.exe”가 아닌 다른 프로세스에 의해 hwp 확장자를 가지는 파일이 쓰기모드로 핸들링 되는 시나리오를 가정하여 1차 실험을 진행하였다. Fig. 10과 같이 “c:\proram files\hnc\hnc80\”의 디렉토리에 “hwp.exe”를 “fake.exe”로 변경한 후 파일 변경을 시도하였을 때, “fake.exe”를 Suspend 상태로 변경한 후 사용자에게 경고창을 띄워 사용자에게 해당 프로세스의 종료 여부와 프로세스를 계속 진행할 수 있는 선택창을 제공하였다. 경고창을 생성함과 동시에 핸들링 대상 파일은 클라우드 서버에 암호화되어 저장된다. 따라서 관리 테이블을 기반으로 프로세스 접근을 탐지하여 파일을 보호함과 동시에 클라우드 서버에 저장함으로써 랜섬웨어 감염으로 인한 피해를 방지하고 사용자 파일이 감염되어도 클라우드 서버에서 원래 파일을 복구할 수 있다.

2차 실험은 구현한 시스템을 통해 암호화 과정과 원격 클라우드에 파일을 저장하는 과정에서 리소스에 따른 소비되는 시간을 측정하는 실험을 진행하였다. 디버깅하는 시스템은 Windows-XP 32bit, i5-2145m cpu와 4G의 메모리를 사용하는 환경이

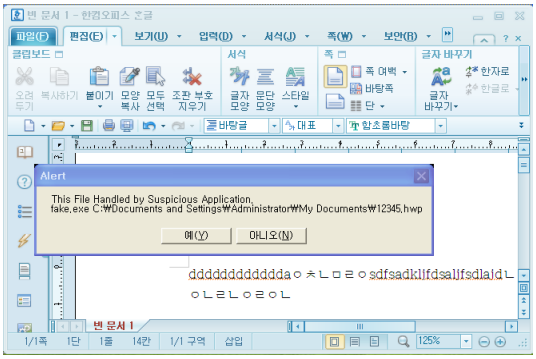


Fig. 10. Alert window that will be shown to the user when hwp file is used by abnormal process

Table 3. Performance measurement results

Index	528kbytes	27,664kbytes
1	0.156sec	2.117sec
2	0.172sec	2.023sec
3	0.187sec	2.007sec
4	0.172sec	2.060sec
5	0.172sec	2.138sec
6	0.171sec	2.306sec
7	0.188sec	2.400sec
8	0.157sec	2.589sec
9	0.172sec	2.761sec
10	0.172sec	2.495sec
11	0.188sec	2.416sec
12	0.172sec	2.572sec
13	0.172sec	2.138sec
Average	0.173sec	2.286sec

며, 528kbytes의 파일과 27,664kbytes의 파일을 모니터링하며 아래 Table. 3과 같은 결과를 도출해 냈다.

528kbytes와 같은 데이터 사이즈가 작은 파일의 암호화 과정과 파일 전송과정은 평균 0.173초가 소비 되었으며, 27,664kbytes와 같은 사이즈가 큰 파일에 대해서는 평균 2.286초가 소비되는 것으로 보아 약간의 부하는 발생하지만 사용면에서 크게 문제가 없을 것으로 판단된다.

V. 결 론

국내에서도 랜섬웨어로 인한 피해가 증가함에 따라 보안업체에서 다양한 랜섬웨어 탐지 솔루션을 출시하고 이에 따른 연구가 많이 진행되고 있다. 하지만 기존 연구에서는 대부분 랜섬웨어에 의해 암호화된 파일을 암호키를 이용하여 사용자의 시스템에서 복원하고자 하는 방식으로 암호키를 찾지 못하는 경우 원본파일을 복원할 수 없다는 문제가 있다.

본 논문에서는 사용자의 파일이 저장되는 시점에 자동으로 클라우드 서버에 해당 파일을 암호화하여 저장함으로써, 랜섬웨어에 의한 피해를 방지하고 랜섬웨어에 감염되었을 때 원본파일을 복구할 수 있는 랜섬웨어 복구 시스템을 제안하였다. 앞선 실험결과와 같이 저장/복원에 약간의 시간이 소요되기는 하지만 암호 알고리즘과 필터 드라이버를 이용한 모니터

링 같은 프로그램의 개선을 통해 리소스에 따른 시간을 줄이고자 노력 할 예정이다. 본 논문에서 제안하는 시스템을 적용하여 랜섬웨어에 의해 파일이 오염되거나 삭제되어도 사용자의 파일을 안전하게 복원하여 랜섬웨어 감염으로 인한 피해를 줄일 수 있을 것이다.

References

[1] "Threat analysis report for the first half of 2016" TrendMicro, 2016.

[2] Ward, Mark. "Cryptolocker victims to get files back for free." *BBC News*, 2014.

[3] Pathak, P. B., and Yeshwant Mahavidyalaya Nanded, "A dangerous trend of cybercrime: Ransomware growing challenge," *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)* 2016

[4] Jaeyeon Moon and Younghyun Chang, "Ransomware Analysis and Method for Minimize the Damage," *The Journal of the Convergence on Culture Technology*, 2016, p79-85

[5] Oh, Joo-Hyung, Im, Chae-Tae and Jeong, Hyun-Cheol. "Technical Trends and Response Methods of Drive-by Download," *Communications of the Korean Institute of Information Scientists and Engineers*, 28.

[6] Richet and Jean-Loup, "Extortion on the Internet: the Rise of Crypto-Ransomware." *Harvard University. Retrieved October*, 2015

[7] Nolen Scaife, Henry Carter, Patrick Traynor and Kevin R.B. Butler, "CryptoLock (and Drop It): Stopping Ransomware Attacks on User Data," *International Conference on Distributed Computing Systems*, 2016.

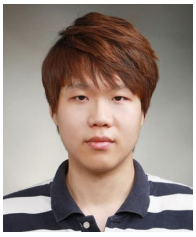
[8] Richardson Ronny and Max North, "Ransomware: Evolution, Mitigation and Prevention." *International Management Review 13.1*, 2017

[9] Miss. Harshada U. Salvi, and Mr. Ravidra V. Kerkar, "Ransomware: A Cyber Extortion," *Asian Journal of Cenvergence in Technology*, 2015.

[10] Moore, Chris. "Detecting Ransomware with Honeypot Techniques." *Cybersecurity and Cyberforensics Conference (CCC)*, IEEE, 2016, p77-81

〈저자소개〉

하 상 민 (Sangmin Ha) 학생회원
2015년 8월~현재: 숭실대학교 융합소프트웨어학과 석사과정
〈관심분야〉 정보보호, 모바일 보안, 클라우드 보안



김 태 훈 (Taehoon Kim) 학생회원
2016년 2월: 숭실대학교 정보통신전자공학부 졸업
2016년 3월~현재: 숭실대학교 정보통신공학과 석사과정
〈관심분야〉 시스템 보안, 모바일 보안, 클라우드 보안

정 수 환 (Souhwan Jung) 종신회원
1985년 2월: 서울대학교 전자공학과 졸업
1987년 2월: 서울대학교 전자공학과 석사
1996년 6월: University of Washington 박사
1988년~1991년: 한국통신 전임 연구원
1997년~현재: 숭실대학교 전자정보공학부 교수
〈관심분야〉 클라우드 보안, 모바일 보안, 네트워크 보안