Smart Media Journal / Vol.8, No.1 / ISSN:2287-1322
http://dx.doi.org/10.30693/SMJ.2019.8.1.27


# 랜섬웨어 탐지를 위한 효율적인 미끼 파일 배치 방법

## (An Efficient Decoy File Placement Method for Detecting Ransomware)

이진우*, 김용민*, 이정환*, 홍지만**

(Jinwoo Lee, Yongmin Kim, Jeonghwan Lee, Jiman Hong)

**요 약**

악성 코드의 일종인 랜섬웨어는 공격 방법이 다양해지고 복잡해지고 있다. 기존 랜섬웨어가 이메일 또는 특정 사이트를 통해 유포 및 감염시키는 것과 달리 WannaCryptor 같은 신종 랜섬웨어는 PC가 인터넷에 연결만 되어 있어도 데이터를 손상시킬 수 있다. 전 세계적으로 랜섬웨어 피해는 시시각각 발생하고 있고 이에 랜섬웨어를 탐지하고 차단하려는 많은 연구가 진행되고 있다. 기존 랜섬웨어 탐지 관련 연구는 프로세스의 특정 행위를 감시하거나 시그니처 데이터베이스를 활용하여 탐지하기 때문에 기존 랜섬웨어와 다른 동작을 보이는 신종 랜섬웨어가 실행되는 경우에는 탐지하고 차단하는 것이 어려울 수 있다. 따라서 본 논문에서는 기존의 랜섬웨어가 파일시스템에서 파일에 접근하고 동작하는 방식을 분석하여 미끼 파일을 배치하여 랜섬웨어를 탐지하는 방법을 제안한다. 또한, 제안하는 방법으로 랜섬웨어를 탐지하고 차단하는 실험을 진행한다.

■ 중심어 : 랜섬웨어 분석; 랜섬웨어 탐지; 미끼 파일

**Abstract**

Ransomware is a malicious program code evolved into various forms of attack. Unlike traditional Ransomware that is being spread out using email attachments or infected websites, a new type of Ransomware, such as WannaCryptor, may corrupt files just for being connected to the Internet. Due to global Ransomware damage, there are many studies conducted to detect and defense Ransomware. However, existing research on Ransomware detection only uses Ransomware signature database or monitors specific behavior of process. Additionally, existing Ransomware detection methods hardly detect and defense a new Ransomware that behaves differently from the traditional ones. In this paper, we propose a method to detect Ransomware by arranging decoy files and analyzing the method how Ransomware accesses and operates files in the file system. Also, we conduct experiments using proposed method and provide the results of detection and defense of Ransomware in this paper.

■ keywords : Ransomware Analysis; Ransomware Detection; Decoy File

## Ⅰ. 서 론

정보통신기술의 빠른 발전과 데이터 저장 비용이 감소하면서 컴퓨터가 저장하는 정보의 양도 급격하게 증가하고 있다. 그러나 최근 신종 악성코드인 랜섬웨어가 등장하여 관련 직종 종사자들이 많은 피해를 받고 있다[1]. 랜섬웨어는 몸값을 의미하는 'ransom'과 프로그램을 의미하는 'ware'가 합쳐진 단어로 사용자의 컴퓨터 시스템을 암호화하고, 이를 인질 삼아 사용자에게 복구비용을 요구하는 악성코드의 한 종류이다. 최초의 랜섬웨어는 1989년 '조셉 팝'에 의해 만들어졌다. 2013년에 출현한 CryptoLocker는 피해자에게 복구비용으로 비트코인을 요구한 최초의 랜섬웨어로 이때부터 해커들은 복구비용을 익명성을 보장받으며 요구할 수 있게 되었다[2].

대부분의 랜섬웨어는 블록 암호를 통해 파일을 손상시킨다[3]. 랜섬웨어로 인하여 파일이 손상되면 피해자는 랜섬웨어 개발자에게 돈을 지불하여 손상된 파일의 복구를 시도하거나 포기해야 한다. 그러나 랜섬웨어 개발자에게 돈을 지불하여 복구

* 학생회원, 숭실대학교 컴퓨터학과
** 종신회원, 숭실대학교 컴퓨터학부
이 논문은 2016년 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. NRF-2016R1D1A1B01016073, 가 상화 기반 오픈 게이트웨이 플랫폼과 오케스트레이션 보 안 서비스 프레임워크 설계 및 구현)

접수일자 : 2018년 02월 08일
수정일자 : 2019년 02월 21일
게재확정일 : 2019년 02월 26일
교신저자 : 홍지만 e-mail : jiman@ssu.ac.kr

를 진행하더라도 손상된 파일을 완전하게 복구할 수 없는 경우의 사례도 증가하고 있다[4]. 랜섬웨어로 인하여 발생하는 대부분의 피해는 피해자가 부주의하게 랜섬웨어가 포함된 실행 프로그램을 입수하여 직접 실행하거나 플래시(Flash) 취약점을 이용하는 랜섬웨어를 포함한 특정 웹 사이트에 접속하는 경우 발생한다[2]. 하지만 최근에는 Windows 운영체제의 SMB(Server Message Block) 취약점을 이용한 WannaCryptor 랜섬웨어와 같이 인터넷에 연결된 상태만으로도 감염시킬 수 있는 랜섬웨어가 존재한다[5]. 따라서, 랜섬웨어로 인한 피해는 사용자의 주의만으로 방지할 수 있는 것이 아니기 때문에 랜섬웨어를 탐지하고 차단하는 기술 개발이 필요하다.

랜섬웨어로 인한 피해가 증가함에 따라 이를 탐지 및 차단하기 위한 다양한 기법들이 연구되고 있다. 랜섬웨어 탐지 및 차단 기법으로는 미끼 파일을 이용한 기법[6], 파일 확장자 모니터링을 이용한 탐지 기법[7], 랜섬웨어의 I/O 패턴을 모니터링을 이용한 기법[8] 등이 있다.

본 논문에서는 소스코드 분석을 통해 랜섬웨어의 동작 과정을 분석한다. 소스코드가 공개되지 않은 랜섬웨어의 경우 랜섬웨어 실행 파일을 역공학한 소스코드를 사용하였다.[9-10]. 또한, 랜섬웨어의 파일 및 디렉터리 암호화 패턴을 분석하여 파일 시스템에 영향을 크게 주지 않고 적절한 위치에 효율적으로 미끼 파일 배치 방법을 제안한다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 본 논문과 관련된 배경 지식과 기존 랜섬웨어 탐지 관련 연구를 소개한다. Ⅲ장에서는 기존 랜섬웨어를 파일 시스템 관점에서 분석하고, 파일 암호화 과정을 분석한다. Ⅳ장에서는 기존 랜섬웨어 프로그램의 소스코드를 분석한 결과와 향후 배포될 수 있는 신종 랜섬웨어의 행위 기반 미끼 파일 배치 방법을 제안한다. Ⅴ장에서는 Ⅳ장에서 제안하는 미끼 파일을 이용한 랜섬웨어 탐지를 실험하고 결과를 분석하여 성능을 평가한다. Ⅵ장에서는 결론에 대해 서술한다.

## Ⅱ. 관련 연구

### 1. 배경 지식

#### 가. Windows 이벤트 로그

Windows의 이벤트 로그는 Windows 운영체제에서 발생하는 로그인, 네트워크 이용, 파일 생성・접근・삭제 등의 I/O 이벤트, 레지스트리 변경 등의 로그가 실시간으로 저장된다. Windows 이벤트 로그에는 이벤트 ID, 프로세스 이름, 프로세스 ID, 개체 이름, 객체에 대한 행위 등에 대한 정보가 존재한다. 이러한 이벤트 로그는 BXML(Binary XML) 형태로 저장되며 Windows에서 제공하는 이벤트 뷰어를 사용하여 분석할 수 있다[11-13].


그림 1. 파일 삭제 이벤트 로그

[그림 1]은 파일을 삭제하였을 때 발생하는 이벤트를 'Windows 이벤트 뷰어'를 이용하여 나타낸 것이다. 해당 이벤트 로그에는 프로세스 ID가 '0x4f78'인 'file_create_test.exe' 프로세스가 'aaaaa.txt' 파일을 삭제한 정보가 기록된 것을 알 수 있다.

본 논문에서는 Windows 이벤트 로그를 사용하여 랜섬웨어가 미끼 파일의 암호화 여부를 실시간으로 검사한다. 이벤트 로그 파일(.evtx)을 실시간으로 모니터링하고 분석하여 특정 프로세스가 미끼 파일에 대해 쓰기 또는 삭제를 하면 해당 프로세스를 랜섬웨어로 판단한다.

### 2. 랜섬웨어 탐지 관련 연구

윤정무[6] 등은 랜섬웨어가 파일을 암호화할 때 파일의 확장자를 특정 확장자로 변경시키는 점에서 착안하여 파일 확장자 변화 모니터링을 통한 랜섬웨어 탐지 및 차단 방법을 연구하였다. 해당 연구는 기존에 발견된 랜섬웨어가 변경하는 확장자들을 블랙리스트로 등록하고 임의의 프로세스가 파일 확장자를 블랙리스트에 등록된 확장자로 변경하면 랜섬웨어로 판단하는 방법이다. 또한, 다수의 파일에 대한 확장자 변경이 사람이 수행할 수 없는 짧은 시간에 이루어진 경우 이를 수행한 프로세스를 랜섬웨어로 간주하여 탐지를 하였다. 그러나 해당 방식은 파일 확장자를 변경하지 않는 랜섬웨어는 탐지되지 않는다는 단점이 있다.

이후기[14] 등은 랜섬웨어 감염 시 네트워크 트래픽 내에 고유한 신호가 발생하는 점에서 착안하여 고유 신호 탐지를 통해 랜섬웨어를 탐지 및 차단하는 방법을 연구하였다. 네트워크 트래픽 내에 공통적으로 발생하는 문자열을 확인한 후 해당 문자

열을 통하여 패턴을 작성하였다. 위 방법으로 시그니처 기반 랜섬웨어 탐지를 수행하였다. 그러나 해당 방식은 신·변종 랜섬웨어가 등장할 경우 탐지가 힘들다는 단점이 있다.

한국 백신 회사 Ahnlab에서는 'Decoy 진단' 기술을 사용하여 랜섬웨어의 탐지를 진행한다[7]. 해당 기술은 미끼 파일을 담아두는 폴더를 각 드라이브의 루트(root) 경로에 배치하여 해당 폴더 내의 파일을 암호화하거나 파일 이름 변경을 시도하는 프로그램을 탐지하고 차단한다. 그러나 해당 방법은 랜섬웨어가 드라이브의 루트 경로를 탐색하지 않으면 탐지가 불가능하며 루트 경로의 탐색이 나중에 이루어질 경우 빠른 탐지가 불가능하다는 단점이 있다.

Kharaz[8] 등은 랜섬웨어가 파일을 암호화하는 과정에서 동일한 I/O 이벤트가 반복적으로 발생한다는 사실을 이용하여 랜섬웨어를 탐지한다. 해당 연구는 랜섬웨어의 동작 방식을 분석하여 파일을 암호화할 때 발생하는 I/O 이벤트 패턴을 추출하여 랜섬웨어의 특징으로 삼는다. 파일 시스템을 모니터링 하여 특정 프로세스가 특징으로 추출한 I/O 패턴을 발생시키면 해당 프로세스를 의심스러운 랜섬웨어라고 간주한다. 그러나 해당 연구에서 제안하는 방법은 신종 랜섬웨어가 파일을 암호화할 때 발생하는 I/O 이벤트 패턴이 보유한 랜섬웨어의 I/O 이벤트 패턴과 일치하지 않을 경우 탐지에 어려움이 존재한다.

옥정균[15] 등은 랜섬웨어를 탐지하는 과정에서 PE Imphash 값을 비교하여 정적 분석 정보의 문자열, 동적 분석 정보의 로그, 파일 관련 정보, 레지스트리 관련 정보를 활용하여 랜섬웨어를 탐지한다. 해당 연구는 Cukkoo Sandbox를 활용하여 랜섬웨어 의심 파일에 대한 정적 분석 정보 및 동적 분석 정보를 얻는다. 분석 정보 중 정적 분석 정보를 PE Imphash 값을 랜섬웨어 탐지 모델의 PE Imphash 데이터베이스와 비교하여 PE Imphash 값이 같다면 랜섬웨어로 판별하고 다르다면 정적 분석 정보의 문자열 정보와 동적 분석 정보의 파일 관련 정보, 레지스트리 관련 정보, 로그 정보들을 Random Forest 알고리즘에 맞게 벡터화한다. 최적화된 벡터값을 통해 랜섬웨어 여부를 판단한다. 그러나 해당 연구에서 제안하는 방법은 분석기법을 우회하는 방법을 사용하는 랜섬웨어 및 코드 인젝션과 같은 정적 분석 및 동적 분석으로 탐지가 불가능한 랜섬웨어 탐지에는 적용시키지 못한다는 단점이 있다.

## III. 랜섬웨어 분석

### 1. 랜섬웨어의 파일 암호화 과정

본 논문에서 분석한 랜섬웨어는 널리 배포되었거나 잘 알려진 Windows 운영체제 기반 랜섬웨어이다. 소스코드가 존재하지 않고 바이너리 실행 파일만 있는 랜섬웨어는 Hex-Ray 사에서 제작한 IDA 5.0 Freeware를 사용하여 소스코드를 추출하였다.

```
1  procedure TraverseDirectory(dirPath)
2    validExtensions <- list of file extensions
3    subDirs = getSubdirectory(dirPath)
4    fileList = getFilelist(dirPath)
5
6    for each file in fileList do :
7      extension = getExtension(file)
8      if extension in validExtensions then :
9        encryptFile(file)
10
11   for each directory in subDirs do :
12     TraverseDirectory(directory)
```

그림 2. 일반적인 랜섬웨어 동작 과정

분석한 Windows 기반의 랜섬웨어의 공통적인 행위는 특정 확장자의 파일을 탐색하고 해당 파일을 암호화하는 것이다. [그림 2]는 일반적인 랜섬웨어가 특정 파일을 탐색하고 암호화하는 과정을 보이고 있다. 랜섬웨어가 동작하는 방법은 크게 2가지가 존재한다. 첫 번째 방법은 파일시스템의 재귀적 탐색을 통해 '.txt', '.doc', '.docx', 등의 특정 확장자를 갖는 파일 찾고 전체 파일시스템의 탐색이 끝난 후 해당 파일을 일괄적으로 암호화 시키는 방법이다. 두 번째 방법은 파일의 특정 확장자와 동일한 확장자를 가지는 파일을 발견하였을 때 위치를 저장하지 않고 바로 암호화를 수행하는 방법이다.

랜섬웨어는 특정 파일 확장자와 일치되는 파일을 탐색하기 위해 다양한 디렉터리에서 탐색을 시작한다. 일부 랜섬웨어는 Windows 파일 시스템의 루트 디렉터리인 'C:/' 부터 탐색을 시작하기도 하며, 또 다른 랜섬웨어는 'C:/사용자/'나 그 하위 디렉터리인 'C:/사용자/[사용자계정이름]/즐겨찾기', 'C:/사용자/[사용자계정이름]/문서', 'C:/사용자/[사용자계정이름]/바탕화면' 의 경로에서 파일 시스템 탐색을 시작한다.

대부분의 랜섬웨어는 파일의 종류와 상관없이 이름순으로 탐색을 수행한다. 그러나 일부 랜섬웨어는 'Windows-1252[16-17]' 문자 인코딩 순서로 파일을 탐색하거나 그것의 역순으로 탐색하기도 한다. 특정 랜섬웨어는 일반 파일과 폴더를 구분하여 특정 확장자를 가진 일반 파일을 먼저 검색하고 하부 폴더 내 일반 파일을 검색한다. 또한, 일반 파일부터 탐색을 시작하지 않고 하부 폴더부터 탐색하는 랜섬웨어도 존재한다.

### 2. 랜섬웨어 파일 암호화 방법

#### 가. ‘write-in-place’ 방법

[그림 3]은 랜섬웨어의 파일 암호화 방법 중 ‘write-in-place’ 방법을 서술한 알고리즘을 나타낸 것이다. 먼저, 임시파일을 생성하고 목표 파일을 암호화한 데이터를 임시파일에 저장한다. 그리고 임시파일의 내용을 기존 목표 파일에 덮어쓴다. ‘write-in-place’ 암호화 방법은 읽기 작업과 쓰기 작업이 대부분이다. 따라서, 해당 방법을 사용하는 랜섬웨어는 파일의 읽기 작업과 쓰기 작업만 감시하면 비교적 쉬운 방법으로 탐지할 수 있다.

```
1  procedure WriteInPlace(filePath)
2    tmpFile = CreateTmpFile()
3    encryptFile = Open(filePath)
4
5    while true do :
6      data = Read(encryptFile)
7      if data == null then :
8        break;
9      encryptedData = Encrypt(data)
10     Write(tmpFile, encryptedData);
11
12   while true do :
13     data = Read(tmpFile)
14     if data == null then :
15       break;
16   Write(encryptFile, data);
17
18   Close(tmpFile)
     Close(encryptFile)
```

그림 3. ‘write-in-place’ 암호화 방법

#### 나. ‘rename-and-encrypt’ 방법

[그림 4]는 파일 암호화 방법 중 ‘rename-and-encrypt’ 방법을 서술한 알고리즘을 나타낸 것이다. ‘rename-and-encrypt’ 방법은 암호화 대상 파일의 이름을 임시 이름으로 변경한다. 그리고 해당 파일을 ‘write-in-place’ 방법으로 암호화한 후, 다시 파일의 이름을 임시 이름에서 기존 이름으로 변경하는 방법이다. 해당 방법을 통하여 랜섬웨어는 암호화한 파일을 기존 이름으로 변경할 때 랜섬웨어만의 고유한 파일 확장자를 추가할 수도 있다.

```
1  procedure RenameAndEncrypt(filePath)
2    tmpFilePath = GetTmpFileName()
3    RenameFile(filePath, tmpFilePath)
4    tmpFile = CreateTmpFile()
5    encryptFile = Open(tmpFilePath)
6
7    while true do :
8      data = Read(encryptFile)
9      if data == null then :
10       break;
11     encryptedData = encrypt(data)
12     Write(tmpFile, encryptedData)
13
14   while true do :
15     data = Read(tmpFile)
16     if data == null then :
17       break;
18     Write(encryptFile, data)
19
20   Close(tmpFile)
21   Close(encryptFile)
22   RenameFile(tmpFilePath, filePath + rswareExt)
```

그림 4. ‘rename-and-encrypt’ 암호화 방법

#### 다. ‘create-encrypt-and-delete’ 방법

[그림 5]는 ‘create-encrypt-and-delete’ 방법을 서술한 알고리즘을 나타낸 것이다. ‘create-encrypt-and-delete’ 방법은 새로운 임시 파일을 생성하고 암호화 대상 파일의 데이터를 일정 크기만큼 읽고 암호화하여 생성한 임시 파일에 기록한다. 그리고 암호화 대상이었던 파일은 삭제한다. 해당 방법은 파일을 다른 폴더로 이동시킬 경우 발생하는 로그와 유사하다. 따라서 랜섬웨어가 해당 암호화 방법을 통하여 파일을 암호화하는 것인지, 사용자가 파일을 이동시키는 것인지의 구분에 어려움이 존재한다.

```
1  procedure CreateEncryptAndDelete(filePath)
2    newFile = CreateTmpFile(filePath+rswareExt)
3    originalFile = Open(filePath)
4
5    while true do :
6      data = Read(originalFile)
7      if data == null then :
8        break;
9      encryptedData = Encrypt(data)
10     Write(newFile, encryptedData)
11
12   Close(newFile)
13   Close(originalFile)
```

그림 5. ‘create-encrypt-and-delete’ 암호화 방법

## Ⅳ. 효율적인 미끼 파일 배치

### 1. 기존 랜섬웨어 대응 미끼 파일 배치

Ⅲ장에서 분석한 내용을 통하여 대부분의 랜섬웨어는

'Windows-1252' 문자 인코딩의 순서 또는 그것의 역순으로 디렉터리를 탐색한다는 특징이 존재한다. 해당 특징을 이용하여 파일 시스템 상의 모든 디렉터리에 'Windows-1252' 문자 인코딩의 순서 중 첫 번째 문자(' ', 0x0020)로 시작하는 이름의 미끼 파일과 마지막 문자('₩', 0xA3DC)로 시작하는 이름의 미끼 파일을 만들면 'Windows-1252' 문자 인코딩 순서 또는 그것의 역순으로 파일을 탐색하는 랜섬웨어를 탐지할 수 있다.

그러나 미끼 파일을 모든 디렉터리에 다수 생성할 경우 디스크 저장 공간의 낭비가 발생할 수 있다. 또한, 새로운 디렉터리를 생성할 때마다 미끼 파일을 배치해야 하므로 추가적인 저장 공간이 필연적이다. 따라서 랜섬웨어를 효율적으로 탐지하기 위해서는 기존 랜섬웨어가 디렉터리를 탐색하며 파일을 암호화하는 특징을 통하여 미끼 파일을 모든 디렉터리에 생성하지 않고 랜섬웨어가 탐색하는 디렉터리에만 적절하게 생성해야 한다.

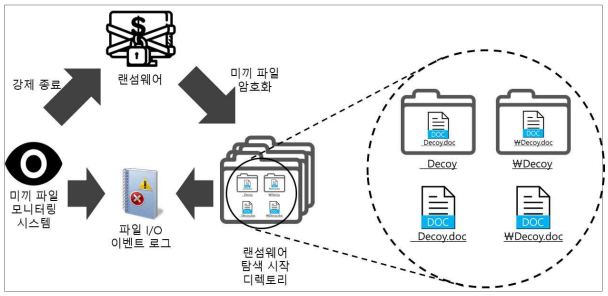


그림 6. 제안하는 미끼 파일 배치 방법

[그림 6]은 Ⅲ장에서 분석한 랜섬웨어의 특징을 기반으로 랜섬웨어 탐지를 위해 미끼 파일과 디렉터리를 효율적으로 생성하는 방법을 보인 것이다. 랜섬웨어가 탐색을 시작하는 디렉터리들은 차이가 존재하나 기본적으로 'C:/', 'C:/사용자,' 'C:/사용자/[사용자계정이름]/즐겨찾기', 'C:/사용자/[사용자계정이름]/문서', 'C:/사용자/[사용자계정이름]/바탕화면'에서 탐색을 시작한다. 이와 같이 랜섬웨어가 탐색을 시작하는 해당 디렉터리에만 미끼 파일을 배치한다면 저장 공간의 낭비를 줄일 수 있다. 이와 같이 미끼 파일을 배치하고, 파일 I/O 이벤트 로그와 미끼 파일을 모니터링 하는 시스템을 구축한다면 랜섬웨어 차단이 가능할 것이다.

## 2. 신종 랜섬웨어 대응 미끼 파일 배치

현존하는 랜섬웨어 이외에도 신종 랜섬웨어가 출현할 수 있기 때문에 랜섬웨어 개발자 입장에서 구현 가능한 랜섬웨어의 행위를 예측하고 이를 미끼 파일 배치에 적용해야 한다. 개발자 입장을 고려하였을 때 향후 등장 가능한 랜섬웨어는 다음과 같은 특징을 가질 수 있다.

신종 랜섬웨어는 디렉터리에 존재하는 파일의 크기순으로 파일을 접근하여 암호화할 수 있다. 작은 크기를 갖는 파일을 먼저 암호화 시킬 경우 많은 파일을 암호화할 수 있다는 특징이 있다. 반대로, 파일 크기가 큰 순서로 암호화할 수 있다.

다른 방식으로는 파일의 접근 시간 순서로 디렉터리를 탐색하여 암호화할 수 있다. 파일에 접근한 시간이 최근일수록 사용자가 자주 사용하거나 중요한 파일일 가능성이 높다. 따라서 신종 랜섬웨어가 파일에 접근한 시간이 최근인 순서대로 암호화를 수행한다면 사용자에게 더 큰 피해를 줄 수 있다.

마지막으로 기존 랜섬웨어의 패턴 분석을 통한 탐지를 회피하기 위하여 신종 랜섬웨어는 일정한 패턴을 통한 행위가 아닌 난수 생성 함수를 이용하여 암호화할 파일을 일정하지 않은 패턴으로 선택하여 암호화할 가능성도 있다. 향후 출몰 가능한 신종 랜섬웨어의 예상 행위에 대한 대응 방법은 다음과 같다.

파일 크기를 기준으로 가장 작거나 가장 큰 파일을 우선적으로 암호화하는 랜섬웨어를 탐지하기 위하여 파일 크기가 작은 미끼 파일 n개와 파일 크기가 큰 미끼 파일 한 개를 각 디렉터리마다 배치한다. 크기가 큰 미끼 파일의 크기는 랜섬웨어가 탐지를 시작하는 디렉터리의 가장 큰 파일보다 크게 설정하였다. 크기가 작은 미끼 파일이 충분한 개수가 배치되지 않는 경우, 랜섬웨어가 차단되는 시간 동안 사용자의 중요 파일을 암호화할 수도 있다. 그러나 n이 무한대에 가까워진다면 랜섬웨어가 사용자의 중요 파일을 암호화하기 전에 탐지하여 차단할 수 있는 가능성이 높으나, 배치되는 미끼 파일로 인하여 파일 시스템에 영향을 줄 수 있다. n은 아래 수식을 만족하는 값으로 설정하였다. $S_d$, $S_b$, $N_d$는 각각 디스크 사이트, 파일시스템 블록 사이즈, 랜섬웨어 탐지 디렉터리 수(본 논문에서는 5)이다.

$$S_d \times 0.005 / S_b / N_d \leq n \leq S_d \times 0.005 / S_b / N_d \quad (1)$$

다음으로 파일의 최근 접근 시간을 기준으로 파일을 접근하여 암호화하는 랜섬웨어를 탐지하기 위하여 미끼 파일의 정보 중 접근 시간을 주기적으로 갱신한다. 미끼 파일이 암호화되는 동안 랜섬웨어를 탐지하면 되기 때문에 미끼 파일이 포함된 하나의 폴더만 주기적으로 갱신한다.

마지막으로 일반적인 프로그램은 랜섬웨어 프로세스처럼 랜덤함수를 사용하고 파일 탐색과 입출력을 빠르게 진행하는 경우가 드물다. 따라서 랜덤함수를 호출하고 파일 I/O 작업이 빈번한 프로세스를 랜섬웨어로 탐지 및 차단한다.

```
1   procedure MakeDecoy(smDecoy, lgDecoy, n)
2     userNames = GetSystemUserName(ALLUSER)
3     sw = GetStartOfWord();
4     ew = GetEndOfWord();
5     startFolder = {"C:/", "C:/Users/"}
6
7     for each name in userNames do :
8       startFolder+="C:/Users/"+name+"/Desktop/"
9       startFolder+="C:/Users/"+name+"/Documents/"
10      startFolder+="C:/Users/"+name+"/Favorites/"
11
12    for each path in startFolder do :
13      MakeFolder(path+sw)
14      MakeFolder(path+ew)
15
16      for i = 1 to n do :
17        MakeDecoy(path+sw+i+".doc",smDecoy)
18        MakeDecoy(path+ew+i+".doc",smDecoy)
19        MakeDecoy(path+sw+"/"+sw+i+".doc",smDecoy)
20        MakeDecoy(path+ew+"/"+ew+i+".doc",smDecoy)
21
22      MakeDecoy(path+sw+"large"+i+".doc", lgDecoy)
23
24    MakeDecoy("C:/"+sw+"/Recent.doc", lgDecoy)
25    new Thread(RegularlyUpdateDecoyFile,
26      "C:/"+sw+"/Recent.doc")
27
28    new Thread(RandomizeMonitoring)
```

그림 7. 제안하는 미끼 파일 배치 방법

[그림 7]은 기존 랜섬웨어와 신종 랜섬웨어 탐지를 위해 효율적으로 미끼 파일을 생성하는 방법을 알고리즘으로 서술한 것이다. 먼저, 시스템 사용자들의 계정 이름을 얻어 각 사용자의 바탕화면, 문서, 즐겨찾기 등의 디렉터리에 미끼 파일을 생성한다. 그리고, 파일 이름이 'Windows-1252' 문자 인코딩에서 가장 앞선 순서의 문자와 마지막 순서 문자로 시작하는 미끼 파일과 및 디렉터리를 만든다. 랜섬웨어가 파일을 크기순 또는 역순으로 암호화할 수 있기 때문에 미끼 파일은 n개의 작은 미끼 파일들과 1개의 큰 미끼 파일을 생성한다. 추가로, 랜섬웨어가 최신 파일을 먼저 접근하는 방식으로 동작할 수 있기 때문에 미끼 파일의 접근 시간을 주기적으로 갱신한다.

## V. 실험 결과

본 장에서는 제안한 랜섬웨어 탐지 및 차단 기법과 기존 기법을 다양한 랜섬웨어에 대하여 실험 및 비교한다. 실험을 진행한 호스트, 게스트, 가상머신의 실험 환경은 [표 1] 과 같다. 가상머신은 호스트의 SSD에 설치하고, 게스트의 데이터는 호스트의 HDD에 위치한다.

표 1. 연구에 사용한 실험 환경

| 구분 | | 설명 |
|---|---|---|
| 호스트 | CPU | Intel(R) Core(TM) i5-6600 CPU @ 3.30GHz |
| | 메모리 | DDR4 16GB |
| | HDD | SSD 256GB, HDD 1TB |
| | 운영체제 | Windows 10 Home 64bit |
| 게스트 | CPU | 1 Core |
| | 메모리 | 2GB |
| | HDD | 30GB |
| | 운영체제 | Windows 7 ultimate 64bit |
| 가상머신 | 종류 | VMware Workstation |
| | 버전 | 12.5.7 |

본 논문에서는 제안하는 기법이 랜섬웨어를 피해 없이 차단하는데 얼마나 효과적인지 실험한다. 제안하는 기법의 효용성을 알아보기 위하여 랜섬웨어가 차단될 때까지 암호화된 파일 수 및 총 용량을 측정한다. 또한, 랜섬웨어가 파일 암호화를 시작한 후 차단될 때까지 걸린 시간을 측정한다.

본 논문에서는 제안하는 미끼 파일 배치 기법을 기반으로 현존하는 랜섬웨어를 탐지하고 차단하는 실험을 진행한다. [표 2]는 본 논문에서 제안하는 미끼 파일 기반 탐지 시스템 실험에 사용할 65개의 랜섬웨어 중 일부를 나타낸 것이다. [표 2]에 나타낸 랜섬웨어 및 실험에 사용한 65개의 랜섬웨어는 이미 널리 배포되었거나 잘 알려진 Windows 운영체제 기반의 랜섬웨어이다.

표 2. 제안하는 탐지 방법 실험에 사용된 랜섬웨어

| 번호 | 해시 값 | 랜섬웨어 종류 |
|---|---|---|
| 1 | 032dab57b4db94120176ae4b18106816a4663a6573b02e72b22702cfd6f85e8f | Cerber |
| 2 | 0ea994b577d1d76b944fc299a23b9292303d0b69fe29f7e5cfba64281dc3cf8e | Cerber |
| 3 | 4ec69ca41129435f3ecc19402b95fd329593c7dfabee7842214bc38aac75b9e7 | Cerber |
| 4 | 1898f2cae1e3824cb0f7fd5368171a33aba179e63501e480b4da9ea05ebf0423 | nsb ransomware |
| 5 | 0e77563d1d4585c2d71916e1c109abf0b8373fd8eae90d40ac685f844528a405 | TeslaCrypt |
| 6 | 198f1dcdf3466ff81aa884ff5a399bb7742e88e01355559b6fba266fe78be569 | Unknown |
| 7 | e6996ecfa47bd3ff69225675dc903bdb419bc83557a15dd3937dabb58b915fda | Unknown |
| 8 | f51f4a15a0edac0d8631b021b868097f4dd48c83c7f09692d85411f842139017 | Unknown |
| 9 | 09a46b3e1be080745a6d8d88d6b5bd351b1c7586ae0dc94d0c238ee36421cafa | WannaCryptor |
| 10 | 149601e15002f78866ab73033eb8577f11bd489a4cea87b10c52a70fdf78d9ff | WannaCryptor |
| 11 | 14c49894872 4d72311aa6fea90a3a6473927daaa3836051d5ca97fad377459e3 | WannaCryptor |
| 12 | ed43c819cc24b76ac1c48a94680ae1e85f834d6ca128f6e1c0635ede13cb3dc1 | WannaCryptor |
| 13 | f0ae40aaec29c4fb88f81b62854c7fe21d16b528c1e2bb30b87ceb71f39e0ce2 | WannaCryptor |

제안하는 미끼 파일 배치를 이용한 랜섬웨어 탐지 기법과 기존의 미끼 파일 배치를 이용한 랜섬웨어 탐지 기법을 비교하여 실험하였다. [그림 8] 은 기존 기법과 본 논문에서 제안하는 기법의 실험 결과를 보여준다. 기존 기법과 제안한 기법이 탐지한 랜섬웨어의 수는 비슷하였으나 랜섬웨어가 파일을 암호화 하는 순간 탐지된 랜섬웨어의 수는 제안한 기법이 많았다. 기존 기법이 탐지하지 못한 랜섬웨어는 C 드라이브의 루트 경로를 탐색 경로에서 제외하였기 때문에 탐지되지 않았다. 또한 빠르게 탐지되지 못한 랜섬웨어는 바탕화면이나 C:/Users 폴더를 먼저 탐색 후 C 드라이브의 루트 경로를 탐색하였기 때문에 빠른 탐지가 불가능했다.

표 3. 랜섬웨어가 차단될 때까지의 행위 분석 결과

| | 소요된 시간(초) | 암호화된 파일 수 | 암호화된 총 용량 (KB) |
|---|---|---|---|
| 최댓값 | 7169 | 53 | 11431 |
| 최솟값 | 144 | 3 | 66 |
| 평균 | 772 | 17 | 4131 |

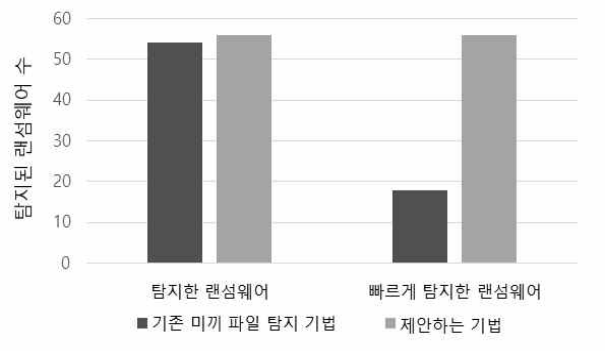


그림 8. 기존 미끼 파일 기법과 제안하는 기법 비교

## VI. 결 론

본 논문에서는 기존 랜섬웨어와 미래에 새롭게 나타날 수 있는 신종 랜섬웨어를 탐지하기 위해 효율적으로 미끼 파일을 배치하는 방법을 제안하였다. Windows 운영체제를 대상으로 배포된 65개의 랜섬웨어를 분석하였다. 이를 통해 랜섬웨어가 미끼 파일을 먼저 암호화 하도록 미끼 파일을 배치한다. 또한 신종 랜섬웨어가 보일 수 있는 행위 패턴을 예측하여 미끼 파일을 배치한다. 미끼 파일은 사용자의 파일보다 먼저 암호화되기 때문에 사용자 파일이 암호화 되기 전에 랜섬웨어를 탐지 및 차단할 수 있다.

하지만 본 논문은 기존에 존재하는 랜섬웨어 중 대표적인 것들을 뽑아 동작 방식을 분석하여 미끼 파일 배치 방법을 제안하였기 때문에 모든 랜섬웨어가 빠르게 탐지된다는 보장이 없다. 또한, 본 논문에서 제안한 미끼 파일 배치 방법을 우회한 방식의 신종 랜섬웨어가 나타날 경우 이를 탐지하지 못할 가능성이 높다. 따라서, 본 논문에서 제안한 방법 외에 다양한 탐지 방법을 종합적으로 사용하여야 새로 출현하는 랜섬웨어를 효율적으로 탐지할 수 있을 것이다.

## REFERENCES

[1] Wikipedia, “Malware”(2017), https://en.wikipedia.org/wiki/Malware. (accessed July 7, 2017)
[2] Wikipedia, “Ransomware”(2017), https://en.wikipedia.org/wiki/Ransomware. (accessed July 7, 2017)
[3] 윤기하, 박성모, “128비트 LEA 암호화 블록 하드웨어 구현 연구,” *스마트미디어저널*, 제4권, 제4호, 39-46쪽, 2015년 12월
[4] “랜섬웨어 해커에게 13억원 줬는데 데이터 완전복구가 안 된다고 한다”(2018), https://www.huffingtonpost.kr/2017/06/30/story_n_17341906.html.(11월 30일, 2018년)
[5] Wikipedia, “WannaCry ransomware attack”(2017), https://en.wikipedia.org/wiki/WannaCry_ransomware_attack. (accessed July 27, 2017)
[6] “랜섬웨어 대응도 한발 앞서다”(2016), http://www.ahnlab.com/kr/site/securityinfo/secunews/secuNewsView.do?seq=25436. (11월 30일, 2018년)
[7] 윤정무, 류재철. “MacOS 에서 파일확장자 관리를 통한 랜섬웨어 탐지 및 차단 방법”. 정보보호학회 논문지, 27권, 2호, 251-258쪽. 2017년 4월
[8] Kharraz, A., Arshad, S., Mulliner, C., Robertson, W. K., & Kirda, E, UNVEIL: A Large-Scale, Automated Approach to Detecting Ransomware, USENIX Security Symposium, pp. 757-772, AUSTIN, TX, 2016, August
[9] Utku Sen, “ransomware open-sources”(2015), https://github.com/goliate/hidden-tear. (accessed June 20, 2017)
[10] SanJay, “Various codes related to Ransomware Developement”(2017), https://github.com/roothaxor/Ransom. (accessed June 20, 2017)
[11] 신용학, 전준영, 김종성. “삭제되거나 손상된 이벤트 로그(EVTX) 파일 복구 기술에 대한 연구”. *정보보호학회논문지*, 26권, 2호, 387-396쪽. 2016년 4월
[12] “Audipol을 이용한 Windows 2008 이벤트오류 disable enable”(2014), http://www.duck.pe.kr/281 (11월 30일, 2018년)
[13] joachimmetz, “Windows XML Event Log (EVTX) format”(2016),

https://github.com/libyal/libevtx/blob/master/documentation/Windows%20XML%20Event%20Log%20(EVTX).asciidoc (accessed June 25, 2017)
[14] 이후기, 성종혁, 김유천, 김종배, 김관용. “랜섬웨어 분석 및 탐지패턴 자동화 모델에 관한 연구”. *한국정보통신학회논문지*, 21권, 8호, 1581-1588쪽. 2017년 8월
[15] 옥정균, 임을규. “정적 분석 정보와 동적 분석 정보를 이용한 랜섬웨어 탐지 모델 제안”. *한국컴퓨터종합학술대회논문집*, 24권, 2호, 1180-1182쪽, ICC JEJU, Korea, 2018년 6월
[16] “Code Page 1252 Windows Latin 1 (ANSI)”(2018),https://msdn. microsoft.com/en-us/library/cc195054.aspx.(12월 2일, 2018년)
[17] Wikipedia, “Windows-1252”(2017), https://en.wikipedia.org/wiki/Windows-1252. (accessed July 10, 2017)

## 저 자 소 개

**이진우**

2017년 숭실대학교 컴퓨터학부 학사 졸업.
2019년 숭실대학교 컴퓨터학과 석사 졸업

<주관심분야 : 시스템 소프트웨어, 운영체제>

**김용민**

2018년 숭실대학교 컴퓨터학부 학사 졸업.
2018년~현재 숭실대학교 컴퓨터학과 석사 재학

<주관심분야 : 시스템 소프트웨어, 운영체제>

**이정환**

2016년 숭실대학교 컴퓨터학부 학사 졸업.
2018년 숭실대학교 컴퓨터학과 석사 졸업

<주관심분야 : 시스템 소프트웨어, 운영체제>

**홍지만(종신회원)**

2003년 서울대학교 컴퓨터공학과 박사 졸업
2004년~2007년 광운대학교 컴퓨터공학과 교수.
2007년~현재 숭실대학교 컴퓨터학부 교수

<주관심분야 : 시스템 소프트웨어, 운영체제>