Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.31, NO.2, Apr. 2021 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2021.31.2.221

키 재사용 공격을 통한 Ragnar Locker 랜섬웨어 감염 파일 복호화 및 활용 방안 연구*

강 수 진,[1†] 이 세 훈,[1] 김 소 람,[1] 김 대 운,[2] 김 기 문,[2] 김 종 성[3‡]
[1,3]국민대학교 (대학원생, 교수), [2]한국인터넷진흥원 (연구원)

A Study on Decryption of Files Infected by Ragnar Locker Ransomware through Key Reuse Attack and Its Applications*

Soojin Kang,[1†] Sehoon Lee,[1] Soram Kim,[1] Daeun Kim,[2] Kimoon Kim,[2] Jongsung Kim[3‡]
[1,3]Kookmin University (Graduate Student, Professor)
[2]Korea Internet & Security Agency (Researcher)

요 약

랜섬웨어는 악성 소프트웨어로 컴퓨터에 저장된 데이터를 암호화하여 접근을 제한하고, 사용자에게 접근 권한을 대가로 금전을 요구한다. 최근 랜섬웨어는 대칭키 또는 스트림 암호 알고리즘을 사용하여 피해자의 파일을 암호화하고, 해당 암호키를 공격자의 공개키로 암호화하는 하이브리드 암호 시스템을 사용한다. 대부분 랜섬웨어는 파일 암호화 시 안전하다고 알려진 AES 알고리즘을 사용하며, 일부 랜섬웨어에서 Salsa20과 같은 스트림 암호를 사용한 경우도 발견된다. 모두 안전성이 확인된 알고리즘이지만, 공격자가 잘못 설계하는 경우 암호학적 취약점이 발생한다. 2019년 하반기에 등장한 Ragnar Locker 랜섬웨어는 하이브리드 암호 시스템을 사용하지만, 파일 암호화 시 스트림 암호에 동일한 키를 반복 사용함으로 취약점이 발생했다. 본 논문에서는 키 재사용 공격을 활용하여 공격자의 개인키 없이 감염된 데이터의 복호화 방안과 해당 취약점을 활용할 수 있는 방안을 제시한다.

ABSTRACT

Ransomware is malicious software that restricts access by encrypting data stored in a computer, and demands money in return for access rights. Ransomware has recently been using a hybrid encryption scheme that combines both symmetric and asymmetric algorithms. The symmetric system is for the encryption of files of a target system and the asymmetric system is used to encrypt the symmetric key. Most ransomware uses the AES algorithm but some use a stream cipher such as the Salsa20. These algorithms are secure however the vulnerability is caused by cryptographic design flaws. Ragnar Locker Ransomware, which appeared in the second half of 2019, uses a hybrid cryptographic system, however, it is vulnerable by reusing the same key when encrypting the files. In this paper, we propose a method of decrypting infected data without a private key of the attacker by utilizing a key reuse attack and a way to apply for other applications.

Keywords: Ransomware, Ragnar Locker, Vulnerability, Cryptography, Key Reuse Attack

Received(01. 06. 2021), Modified(03. 02. 2021), Accepted(03. 02. 2021)

* 본 논문은 2020년도 과학기술정보통신부(암호이용활성화)의 재원으로 한국인터넷진흥원의 지원을 받아 수행된 연구사업임

† 주저자, szin31@kookmin.ac.kr
‡ 교신저자, jskim@kookmin.ac.kr(Corresponding author)

I. 서 론

랜섬웨어(Ransomware)는 컴퓨터의 시스템을 잠그는 방식이나 저장된 데이터를 암호화하는 방식으로 사용자의 데이터를 탈취하는 악성 소프트웨어이다. 이후 데이터의 복호화를 대가로 사용자에게 금전을 요구한다. 과거 랜섬웨어는 대칭키 암호 알고리즘을 사용하여 사용자 데이터를 암호화했다. 랜섬웨어 공격자 입장에서 저장된 데이터 암호화 시 대칭키 암호 알고리즘을 사용하면 암호화 속도는 빠르지만, 키 보관 문제가 존재한다. 따라서 키 보관 문제를 해결하기 위해 C&C (Command & Control) 서버를 통해 키를 받아오거나, 랜섬웨어 내부에 키를 보관한다. 하지만 C&C 서버를 통해 키를 받아와야 하는 경우, 네트워크에 연결되어 있지 않으면 랜섬웨어가 동작하지 않는 경우가 존재한다. 또한, 랜섬웨어가 내부의 고정키를 사용하는 경우, 역공학을 통해 랜섬웨어 내부나 메모리에 남아있는 키를 찾아내어 복호화가 가능하다[1]. 따라서 최신 랜섬웨어는 하이브리드 암호 시스템을 사용하여 빠른 암호화를 진행하고, 키를 효율적으로 관리한다. 하이브리드 암호 시스템은 데이터 암호화에 블록 암호나 스트림 암호 알고리즘을 사용하고, 사용된 암호키를 공격자의 공개키 암호 알고리즘으로 암호화해 감염된 시스템 내부에 보관한다. 이로 인해 공격자의 개인키를 획득하지 못하면 파일 암호화에 사용된 암호키를 복호화할 수 없으므로, 공격자 이외에는 암호화된 데이터의 복호화가 어렵다.

랜섬웨어의 공격은 꾸준히 발생하여 많은 피해를 생성하고 있다. 2019년 상반기 GandCrab 랜섬웨어는 약 50만 건의 피해를 발생시켰으며[2], 2019년 6월, Ryuk 랜섬웨어는 플로리다 시를 공격하여 2차례에 걸쳐 110만 달러를 받아냈다[3], 2020년 2월에는 네덜란드의 Maastricht 대학은 Clop 랜섬웨어에 감염된 데이터를 복호화하기 위해 공격자에게 24만 달러를 지불했다[4]. 같은 해 8월, 미국의 프린트 제조기업인 Xerox와 한국의 LG전자는 Maze 랜섬웨어에 감염되어 주요 기업 데이터가 유출되는 피해가 발생했다[5]. 위의 피해 사례와 같이 랜섬웨어의 공격이 발생하면 경제적 피해가 크게 발생하므로, 이를 해결하기 위해 랜섬웨어의 동작 과정과 사용된 암호 알고리즘을 분석하여 감염된 파일에 대한 복호화 방안 연구가 필요하다.

본 논문에서는 2020년에 주요한 영향을 끼친 Ragnar Locker를 분석하여 암호학적 취약점을 밝힌다. 이를 바탕으로 공격자의 개인키를 이용하지 않고 데이터 복호화 방안을 제시한다.

본 논문의 구성은 다음과 같다. 2장에서는 배경지식 및 관련 연구에 대해 서술하며, 3장에서는 Ragnar Locker의 전체 동작 과정에 대해 서술한다. 4장에서 데이터 암호화 과정에 대해 자세히 분석하며, 5장에서는 분석한 결과를 바탕으로 감염된 데이터의 복호화 방안을 제안하고, 실험을 통해 증명한다. 6장에서는 분석한 취약점의 활용 방안을 제시하며, 끝으로 7장에서 결론으로 마무리한다.

II. 배경지식 및 관련 연구

2.1 배경지식

2.1.1 키 재사용 공격

스트림 암호 알고리즘에서 동일한 키를 재사용하는 경우, 키 재사용 공격(Key reuse attack)이 가능한 취약점이 존재한다. 스트림 암호 알고리즘은 식(1)과 같이 암호키(Key)로부터 키 스트림(KS)을 생성한다. 이를 평문과 XOR 연산하여 식(2)와 같이 암호문을 생성한다.

$$KS = KeyScheduler(Key) \tag{1}$$

$$E(P) = P \oplus KS = C \tag{2}$$

동일한 키로 암호화된 암호문 C_A, C_B는 식(3)과 (4)로 표현이 가능하다.

$$E(A) = A \oplus KS = C_A \tag{3}$$

$$E(B) = B \oplus KS = C_B \tag{4}$$

이때, 암호문 C_A에 대응되는 평문 A를 획득할 수 있는 경우, 식(5)와 같이 암호문 C_A에 평문 A를 한 번 더 XOR 연산하여 키 스트림을 획득할 수 있다. 획득한 키 스트림을 사용하면 식(6)과 같이 다른 암호문 C_B를 복호화 할 수 있다.

$$C_A \oplus A = A \oplus KS \oplus A = KS \tag{5}$$

$$C_B \oplus KS = B \oplus KS \oplus KS = B \tag{6}$$

이와 같이 스트림 암호에 동일한 키를 반복적으로 사용하는 경우, 암호화에 사용된 키 스트림을 획득할 수 있다. 이를 통해 다른 암호문의 복호화가 가능하다. 이와 같은 키 재사용 공격을 활용하여 David Rupprecht 등은 LTE 프로토콜의 도청 공격이 가능함을 보이고, 프로토콜의 취약점을 발견했다[6].

2.2 관련 연구

랜섬웨어에 관한 연구는 탐지 및 예방, 랜섬웨어의 공격 회피 방안 연구 및 네트워크 단에서 감염 차단 등 다양한 방식으로 이루어지고 있다(Table 1).

랜섬웨어를 사전에 탐지하여 감염을 예방하고자 하는 연구가 다수 진행되어 왔다. Andrea Continella 등은 I/O (Input/Output)의 모니터링, 파일명이 변경된 파일의 비율 및 유형, 폴더 나열 작업의 빈도 등을 기반으로 랜섬웨어 탐지 방안을 제시했다[7]. D. Gonzalez와 T. Hayajneh는 일반적인 암호화 랜섬웨어의 페이로드 감염 방법 및 공격 방식에 대해 연구하고 이를 바탕으로 예방법을 제시했다[8]. 정상문 등은 암호화된 파일의 엔트로피 측정을 통해 랜섬웨어 탐지 모델을 구축하고 이를 탑재한 랜섬웨어 탐지 및 대응 시스템을 제안했다[9].

랜섬웨어의 공격을 회피하기 위한 연구도 진행됐다. Nolen Scaife 등은 사용자 데이터를 변조하는 프로세스를 탐지하고 이를 발견하면 강제 종료하는 방법을 사용하여 랜섬웨어 공격에 대한 조기 경고 시스템을 제안했다[10]. 이수현 등은 공격의 대상이 되는 시스템 자체의 구성을 변화시켜 공격을 어렵게 하는 moving target defense 개념을 이용한 랜섬웨어 공격 회피방안을 연구했다[11].

Table 1. Related research of ransomware

Research contents	Reference
Detection ransomware	[7]
	[8]
	[9]
Ransomware attack avoidance	[10]
	[11]
Network layer blocking	[12]
	[13]
Decryption and restore infected data	[14]
	[15]
	[16]

네트워크 단에서 랜섬웨어의 공격을 막기 위한 연구도 진행됐다. Krzysztof Cabaj 등은 C&C 서버와 통신하여 암호키를 받아오는 CryptoWall 랜섬웨어를 분석하여, 랜섬웨어가 사용하는 도메인, URL, IP 정보를 수집하고 이를 이용하여 랜섬웨어의 감염을 차단하는 시스템을 제안했다[12]. 김동현 등은 랜섬웨어가 주로 사용하는 DGA (Domain Generation Algorithm) 함수를 역공학으로 분석하여 랜섬웨어 공격을 탐지하고 잠재적인 악성 URL로의 접속을 차단하는 연구를 진행했다[13].

이와 같은 연구들은 제안된 방안에 사용한 특징을 가진 랜섬웨어의 탐지에는 유용하지만, 탐지에 적용되지 않은 특징으로 동작하는 랜섬웨어의 경우 탐지 및 예방이 어렵다. 이에 대한 침해 대응의 일환으로서 감염된 데이터의 복구 및 복호화 방안에 대한 연구도 진행됐다. 이세훈 등은 메모리 분석을 통해 Donut 랜섬웨어에 감염된 파일을 복호화했다[14]. Kyungroul Lee 등은 랜섬웨어에 감염된 경우, 백업 시스템과 동기화된 랜섬웨어 감염 파일을 탐지하여 백업 시스템에서 원본 파일을 복구하는 방법을 제시했다[15]. 또한, 이세훈 등은 2019년 주요 신규 랜섬웨어 5종을 분석하여 그 중 메모리 포렌식 및 취약한 난수 발생기 재현을 통해 LooCipher의 복호화 방안을 제시했다[16]. 그러나 공격자가 감염된 파일의 복구 및 복호화 가능성을 제거하기 위해 감염 시 사용한 암호키와 관련 요소 및 시스템 백업 파일을 제거하는 경우, 기존 연구에서 제안된 방법을 사용하는데 어려움이 발생한다.

본 논문에서는 랜섬웨어가 파일 암호화 시 사용하는 스트림 암호 알고리즘의 동일한 키 재사용으로 발생하는 취약점을 활용하여 감염된 데이터를 복호화하는 방법을 제안한다. 제안된 방법을 사용하면 랜섬웨어 감염 시 사용한 암호키 및 관련 요소에 대한 정보 없이도 복호화가 가능하다.

III. Ragnar Locker 랜섬웨어

3.1 개요

Ragnar Locker 랜섬웨어는 2019년 말 발견되었으며, 가상머신을 이용해 악성 행위를 수행하는 특징이 있다. 내장된 가상화 프로그램을 사용하여 감염된 PC에 설치된 백신을 우회한다[17]. 이러한 백신 우회 기술을 사용하여 주로 기업을 대상으로 공격을

수행한다. 2020년 4월, 다국적 에너지 기업인 EDP를 공격하여 약 10TB의 기밀 정보를 탈취했으며, 이를 유포하지 않는 조건으로 약 1,000만 달러를 요구했다[18]. 같은 해 7월, 미국의 여행 서비스 회사인 CWT를 공격했고, 해당 기업은 파일 복호화를 위해 약 450만 달러를 지불했다[19]. 또한, 9월에는 컨테이너 운송회사를 공격하여 일시적으로 서비스를 마비시켰다[20]. 최근 11월에는 이탈리아 주류회사인 Campari와 일본의 게임 개발 업체인 Capcom을 공격하여 각 기업의 기밀 정보를 탈취하고, 서비스를 마비시켰다[21, 22].

본 논문에서 분석한 Ragnar Locker 랜섬웨어 샘플 파일의 해시값은 Table 2와 같다.

Table 2. Ragnar Locker ransomware hash value

Hash Value	
MD5	24B354B142B5046263E91170DB92790B
SHA1	BBB71391CA40BCEBFAADF8E1367412333457D771
SHA256	9706A97FFA43A0258571DEF8912DC2B8BF1EE207676052AD1B9C16CA9953FC2C

3.2 전체 동작 과정

Ragnar Locker의 전체 동작 과정은 다음 Fig.1과 같이 진행된다.

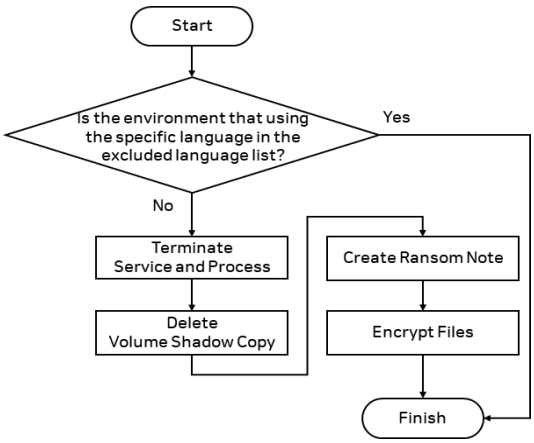

Fig. 1. Operation process of Ragnar Locker

감염이 시작되면 감염할 PC의 사용 언어를 확인한다. Table 3에 속하는 언어를 사용하는 환경의 경우, 감염을 진행하지 않는 특징이 있다.

Table 3. Languages to be excluded from infection

Languages to be excluded from infection		
Azerbaijani	Armenian	Belorussian
Tajik	Russian	Turkmen
Kazakh	Kyrgyz	Moldavian
Uzbek	Ukrainian	Georgian

Ragnar Locker는 악성 행위에 사용할 정보를 RC4 알고리즘으로 암호화하여 내부에 저장한다. 이를 Fig.2와 같이 0x40바이트 길이의 고정키로 복호화하여 사용한다. 고정키는 랜섬웨어 내부에 하드코딩 되어 저장되며, 서비스 목록, 랜섬노트 및 공개키 복호화 시 반복적으로 사용된다.

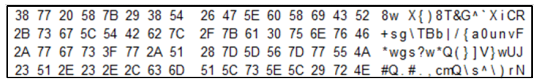

Fig. 2. Fixed key used in RC4 algorithm

감염 전 백업 및 원격 기능을 지원하는 서비스를 종료하며, 해당 서비스 목록은 랜섬웨어 내부에 RC4 알고리즘으로 암호화되어 있다. 이를 고정키로 복호화하여 사용하며, 복호화된 목록은 다음 Table 4와 같다.

Table 4. Termination service list

Termination service list		
vss	sql	memtas
backup	pulseway	logme
mepocs	sophos	veeam
logmein	connectwise	splashtop

서비스 중지 이후, 현재 실행 중인 프로세스의 설치 경로를 획득하여 Windows 폴더가 아닌 경우 해당 프로세스를 종료한다. 또한, Ragnar Locker는 윈도우의 백업 기능을 사용하지 못하도록 볼륨 섀도 복사본을 삭제한다. 랜섬노트는 랜섬웨어 내부에 RC4 알고리즘으로 암호화되어 있으며, 고정키로 복호화하여 생성한

다. 감염이 완료되면 랜섬노트를 모니터에 출력한다. 파일 암호화 시 스트림 암호인 Salsa20[23] 알고리즘을 사용한다. 랜섬웨어 실행과 운영체제 실행에 연관이 있는 파일 및 폴더를 제외한 모든 파일을 암호화하며, 암호화 제외 대상 목록은 Table 5와 같다.

Table 5. Encryption exclusion list

Extension	Folder	File
.db .msi .sys .drv .dll .exe .lnk	Windows Windows.old Tor browser Internet Explorer Google Opera Software $Recycle.Bin Mozilla Firefox ProgramData All Users	autorun.inf boot.ini bootfont.bin bootsect.bak desktop.ini iconcache.db ntldr ntuser.dat ntuser.dat.log ntuser.ini thumbs.db Ransom Note

IV. Ragnar Locker 암호화 과정

본 장에서는 Ragnar Locker의 상세 암호화 과정에 대해 서술한다. 전체 암호화 과정은 다음 Fig.3과 같다.

파일 암호화 시 스트림 암호인 Salsa20 알고리즘을 사용하며, 스트림 암호키 생성에 사용된 seed를 RSA-2048-OAEP로 암호화하여 보관된다.

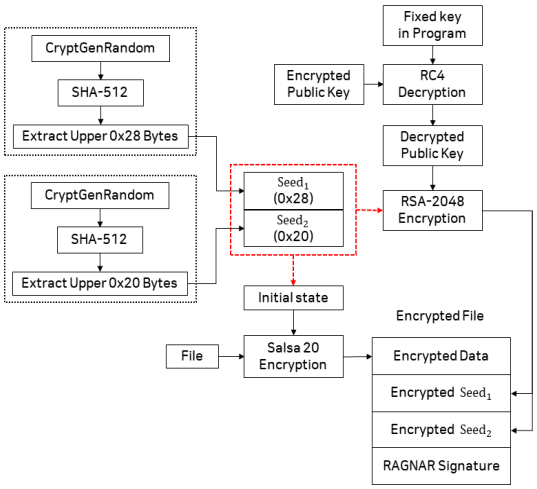

Fig. 3. Encryption process of Ragnar Locker

4.1 암호키 생성 및 파일 암호화

파일 암호화에 사용할 암호키는 두 개의 랜덤한 난수인 $seed_1$과 $seed_2$로 생성된다. 두 seed의 생성 과정은 다음과 같다.

① $r_1 = CryptGenRandom$
② $n_1 = SHA512(r_1)$
③ $seed_1$ = n_1의 상위 0x28바이트
④ $r_2 = CryptGenRandom$
⑤ $n_2 = SHA512(r_2)$
⑥ $seed_2$ = n_2의 상위 0x20바이트
r_1 : 0x28바이트, r_2 : 0x20바이트

윈도우에서 제공하는 난수 생성 함수인 CryptGenRandom 함수를 사용하여 난수를 생성한다. 생성된 난수를 SHA512로 해싱한 후 특정 길이의 상위 바이트를 seed로 사용한다. $seed_1$은 0x28바이트 길이이며, $seed_2$는 0x20바이트 길이이다.

생성된 $seed_1$과 $seed_2$를 산술연산자와 비트연산자를 사용하여 파일 암호화에 사용되는 Salsa20 알고리즘의 initial state를 구성한다. 이때 생성된 암호키는 단일키로써, 모든 파일 암호화 시 동일하게 사용된다.

4.2 Seed 암호화

파일 암호키 생성에 사용된 $seed_1$과 $seed_2$는 공격자의 공개키와 RSA-2048-OAEP 알고리즘으로 암호화 되며, 감염된 파일의 끝에 연접된다. 이후 Ragnar Locker 랜섬웨어의 시그니처인 "_RAGNAR_"문자열과 NULL(0x00)문자를 삽입한다. 암호화된 파일의 구조는 다음 Table 6과 같다.

Table 6. Encryption file format

Size(Byte)	Data
Size of original file	Encrypted file
0x100	Encrypted $seed_1$
0x100	Encrypted $seed_2$
0x9	Ragnar Signature

4.3 암호키 삭제

파일 암호키 생성에 필요한 seed들을 프로그램 단위에서 제로화 혹은 할당해제와 같은 키 파기 과정을 진행하지 않는다. 그러나, 랜섬웨어의 악성 행위가 끝나고 난 후, ExitProcess 함수를 이용하여 자신을 종료시키며, 이 경우 프로세스의 스레드가 모두 종료되고 모든 핸들이 닫힌다[24]. 따라서, 메모리 포렌식을 활용한 암호키 복원 가능성을 확인하기 위하여 실험을 진행했다. 실험환경 구성은 다음과 Table 7과 같다.

실험을 위해 가상머신 소프트웨어인 VMware Workstation을 사용하여 가상 분석환경을 구축하였으며, Ragnar Locker를 감염시킨 후 메모리 추출 도구인 FTK Imager와 DumpIt을 이용하여 메모리를 추출하고, 이를 Volatility를 활용하여 메모리 포렌식을 진행했다. 또한, 동적 분석을 위해 x64dbg를 사용하였고, 메모리 내 존재하는 seed들을 찾기 위해 010 Editor를 활용했다.

Table 7. Test environment

Host Computer Environment	
Memory (RAM)	32.0 GB
System	Window 10 x64
Processor	Intel(R) Core(TM) i5-6500 CPU @3.20GHz
Virtual Computer Environment	
Memory (RAM)	4.0 GB
System	Window 7 x86
Software	**Version**
VMware® Workstation 15 Pro	15.5.0
010 Editor (hex editor)	6.0.3
Volatility (memory forensics)	2.6
x64dbg	Apr 12 2020
FTK Imager	3.2.0.0
DumpIt	1.3.2.20110401

4.3.1 Volatility를 활용한 Ragnar process 검색

$seed_1$과 $seed_2$는 전역변수로써, 프로세스 내 데이터 영역에 존재한다. 따라서, 메모리 내 Ragnar

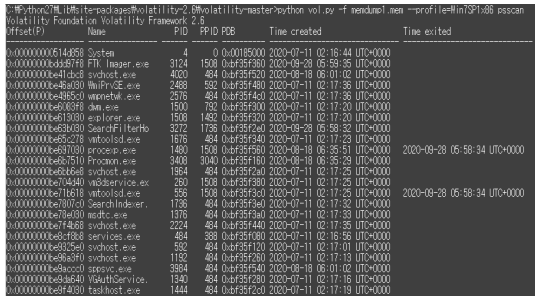

Fig. 4. Process search using Volatility

Locker 프로세스가 존재한다면 Ragnar Locker process를 분석하여 $seed_1$과 $seed_2$를 획득할 가능성이 존재한다. Ragnar Locker 랜섬웨어는 모든 악성행위가 끝나면 Exitprocess 함수를 통해 자신을 종료시키므로, 현재 실행중인 프로세스뿐만 아니라 종료된 프로세스도 검색하는 psscan 기능을 사용하여 Ragnar Locker 프로세스가 존재하는지 확인하였다(Fig.4). 그러나, 해당 실험을 7회 수행하였으나 Ragnar Locker 프로세스를 찾을 수 없었다.

4.3.2 동적 분석을 통한 seed 전수조사

감염 후 추출된 메모리 내 $seed_1$과 $seed_2$ 값이 존재한다면 메모리 전수 조사를 통해 seed를 추측하여 파일 암호화 시 사용된 키를 구현할 수 있다. 따라서 전수 조사 가능성을 확인하고자 메모리 내 $seed_1$과 $seed_2$의 값이 존재하는지 확인했다. 가상 환경에서 감염 전 Ragnar Locker 랜섬웨어를 동적 분석을 하여 파일 암호화에 사용된 $seed_1$과 $seed_2$ 값을 획득한 후, 파일 암호화를 계속 진행했다. 이후 악성행위가 종료되면, 메모리를 추출하고 010 Editor를 사용하여 디버깅 중 획득한 $seed_1$과 $seed_2$값을 검색하였으나, $seed_1$과 $seed_2$ 모두 존재하지 않았다. 따라서, 메모리 포렌식을 활용한 암호키 복원은 어렵다.

V. 파일 복호화 방안

Ragnar Locker 랜섬웨어는 메모리 분석을 통해 암호키 생성에 사용된 seed와 파일 암호키의 획득이 어렵다. 또한, 암호키를 추측하더라도 암호키에 대한 인증자가 없어 데이터 복호화를 진행하여 추측한 암호키를 검증해야 한다. Ragnar Locker는 암호키 생성 시 난수 생성 함수인 CryptGenRandom 함수

와 SHA512 해시 함수를 사용하여 $seed_1$과 $seed_2$를 생성한다. CryptGenRandom 함수는 암호학적으로 랜덤한 난수 발생기이며[25], SHA512 해시 함수 또한 암호학적으로 안전한 해시 알고리즘이다. 즉, $seed_1$과 $seed_2$를 찾는 것은 r_1과 r_2를 전수조사하는 것과 동일하며, 이는 40바이트의 $seed_1$과 32바이트의 $seed_2$를 랜덤하게 추측하는 것과 동일하다. 또한, Salsa20의 initial state는 64바이트이므로 이를 전수조사하는 방법도 존재한다. Ragnar Locker는 initial state의 key, nonce, 고정문자열을 $seed_1$과 $seed_2$를 통해 랜덤하게 생성하므로 0으로 고정되는 position 값을 제외하면, 56바이트를 전수조사해야 한다. 해당 계산량은 약 2^{448} 정도가 요구되므로 현실적으로 전수조사가 불가능하다. 하지만 Ragnar Locker 랜섬웨어는 모든 파일에 대해 동일한 암호키를 사용하여 스트림 암호 알고리즘으로 암호화한다. 따라서 키 재사용 공격을 활용하면 공격자의 개인키 없이도 감염된 파일을 복호화 할 수 있다.

키 재사용 공격을 활용하기 위해 Ragnar Locker에 감염된 파일에 대응되는 원본 파일이 필요하다. 해당 파일을 통해 파일 암호화에 사용된 키 스트림을 획득할 수 있다. 획득한 키 스트림을 이용하면 다른 감염된 파일의 복호화가 가능하다. 이때, 획득한 키 스트림의 길이가 복호화 대상 파일의 크기보다 작은 경우, 키 스트림의 크기만큼만 복호화가 가능하다. 분석한 내용을 바탕으로 감염된 파일로부터 원본 파일을 복호화하는 알고리즘은 다음 Fig.5와 같다.

Algorithm 1: Decryption method of infected file of Ragnar Locker

Function: *Decrypt*($File_1, File_2, OriginalFile$)
Input: *encrypted file* $File_1$, *encrypted file* $File_2$, *original file of* $File_1$, *OriginalFile*
Output: *File of decrypted* $File_2$

$KeyStreamSize \leftarrow sizeof(OriginalFile)$
$KeyStream = \{0,\}$
//Finding keystream
for $i \in 0, ..., KeyStreamSize$ **do**
  $KeyStream[i] = File_1[i] \oplus OriginalFile[i]$
end for
$len = \min(sizeof(File_2), KeyStreamSize)$
// Decrypting $File_2$
for $i \in 0, ..., len$ **do**
  $File[i] = File_2[i] \oplus KeyStream[i]$
end for
return *File*

Fig. 5. Decryption method of infected files of Ragnar Locker

감염된 파일을 복호화하기 위한 알고리즘은 감염된 두 개의 파일 $File_1$, $File_2$와 그중 $File_1$의 원본 파일인 *Original File*을 입력으로 한다. 입력받은 $File_1$과 *Original File*을 통해 키 스트림을 획득하고, 해당 키 스트림으로 $File_2$를 복호화하는 방식으로 동작한다.

감염 파일 복호화 알고리즘에 따라 실제 감염 파일을 복호화하는 과정은 다음과 같이 진행된다. 감염 파일 복호화 알고리즘의 3번 과정과 같이 감염 파일 $File_1$과 그에 대응되는 원본 파일인 *Original File*을 XOR 연산하여 키 스트림을 획득한다(Fig. 6).

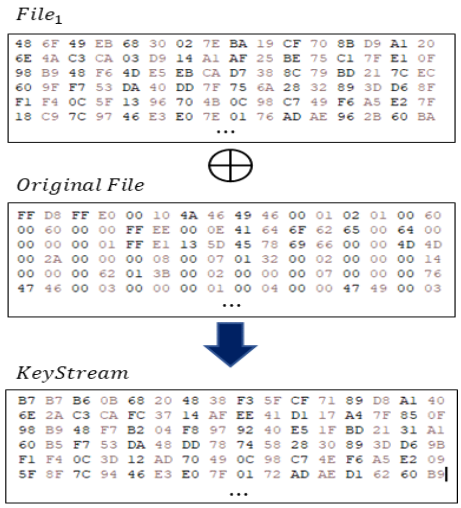

Fig. 6. Acquiring a key stream using XOR operation

감염 파일 복호화 알고리즘의 5번 과정에서 복호화 가능한 파일의 길이가 결정된다. 획득한 키 스트림의 길이가 감염된 파일의 길이보다 큰 경우에는 해당 파일 전체를 복호화할 수 있다. 하지만 키 스트림의 길이보다 감염된 파일의 길이가 더 큰 경우, 키 스트림 길이만큼만 복호화가 가능하다. 이후, 6번 과정과 같이 획득한 키 스트림과 암호화된 파일을 XOR 연산하여 원본 파일을 획득할 수 있다. 실제 위의 과정을 통해 원본 파일을 획득한 모습은 다음 Fig. 7과 같다.

Windows 7의 경우, "C:\Users\Public\Pictures\Sample Pictures" 경로에 항상 기본적으로 설치되는 사진이 존재한다. 해당 사진 파일과 동일한 파일을 인터넷에서 다운받을 수 있다. 또한, 이메일이나 메신저를

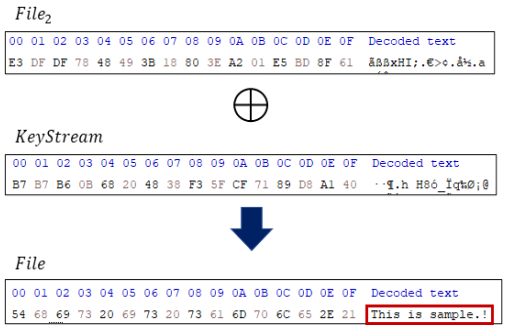

Fig. 7. Obtaining original data using XOR operation

통해 주고받은 파일을 저장해놓은 경우, 해당 파일을 다시 저장하는 방법으로 원본 파일을 획득할 수 있다. 이와 같이 다양한 방법을 통해 감염된 PC의 암호화 된 파일의 원본 파일을 획득하여 복호화 실험을 진행했다. 실험 결과 Fig.8과 같이 감염된 파일이 올바르게 복호화되었다.

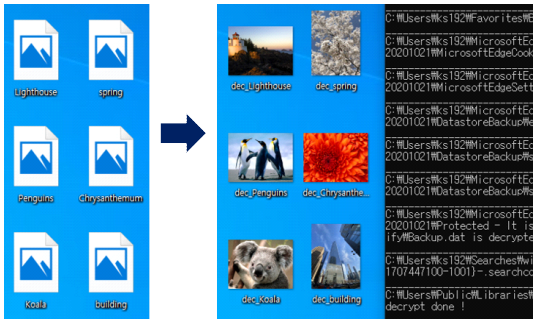

Fig. 8. Decryption test

VI. 활용 방안

스트림 암호 알고리즘의 키 재사용 공격을 활용하면 암호화 시 사용된 암호키 없이도 데이터의 복호화가 가능하다. 실제 Ragnar Locker 랜섬웨어의 경우, 하이브리드 암호 시스템을 사용했지만, 스트림 암호에 동일한 키를 사용하여 키 재사용 공격이 가능했다. 해당 취약점은 블록 암호에도 적용될 수 있다. 블록 암호의 운용모드 중 CTR 모드와 OFB 모드는 스트림 암호와 유사하게 암호키를 통해 키 스트림을 생성하고 이를 평문과 XOR 하는 방식으로 동작한다. 따라서 블록 암호 알고리즘에 동일한 암호키와 IV (Initial Vector)를 사용하면 키 재사용 공격이 가능하다.

본 논문에서 제안한 파일 복호화 방안과 유사하게 데이터 암호화 애플리케이션인 “LockMyPix” 및 “Privary”의 미디어 파일 복호화 연구 및 진행된 바 있다[26, 27]. 이와 같이 랜섬웨어뿐 아니라 다양한 환경에서 키를 반복 사용하는 경우, 본 논문에서 분석한 복호화 방안을 적용할 수 있다. 따라서 암호키 없이 암호화된 데이터의 복호화 가능성을 높일 수 있다.

VII. 결론 및 논의

최신 랜섬웨어는 공개키 암호와 블록 암호 혹은 스트림 암호를 기반으로 한 하이브리드 암호 시스템을 사용한다. 스트림 암호 알고리즘의 경우, 동일한 키를 반복 사용하면 암호학적 취약점이 발생한다. Ragnar Locker 랜섬웨어는 하이브리드 암호 시스템을 사용했지만, 파일 암호화에 사용한 스트림 암호 알고리즘에서 동일한 키를 반복하여 사용했다. 결과적으로 키 재사용 공격이 가능하여 공격자의 개인키 없이도 암호화된 파일의 복호화가 가능했으며, 이를 실험을 통해 증명했다. 키 재사용 공격은 원본 파일의 획득이 어렵거나, 소유한 원본 파일의 길이가 충분하지 못하다면, 감염된 모든 파일의 복호화가 어렵다는 한계점이 존재한다. 그러나, 각 운영체제마다 저장된 고유한 파일을 획득하여 복호화할 수 있다. 따라서 해당 방식으로 데이터가 암호화된 경우, 공격자의 개인키나 파일 암호키가 없더라도 파일을 복호화할 수 있는 가능성이 존재한다. 이와 같이, 안전한 암호 알고리즘을 사용하더라도, 잘못 설계하여 사용하면 취약점이 발생한다. 향후 암호키의 획득이 불가능한 상황에서 동일한 키를 재사용한 경우, 본 논문의 연구 결과를 바탕으로 데이터를 복호화할 수 있을 것으로 기대한다.

References

[1] Bajpai, Pranshu, Aditya K. Sood, and Richard Enbody. "A key-management-based taxonomy for ransomware." 2018 APWG Symposium on Electronic Crime Research (eCrime). IEEE, pp. 1-12, May. 2018.

[2] AhnLab, “2019 ransomware trends”, https://asec.ahnlab.com/1241, Jul. 2019.

[3] SecureWorld, “Special Security Advisor

y: 'Ryuk Ransomware Targeting Organizations Globally'", https://www.securewworldexpo.com/industry-news/how-ryuk-ransomware-works, Sep. 2019

[4] Securityweek, "Netherlands University Pays $240,000 After Targeted Ransomware Attack", https://www.securityweek.com/netherlands-university-pays-240000-after-targeted-ransomware-attack, Feb. 2020.

[5] ZDNet, "Ransomware gang publishes tens of GBs of internal data from LG and Xerox", https://www.zdnet.com/article/ransomware-gang-publishes-tens-of-gbs-of-internal-data-from-lg-and-xerox/, Aug. 2020.

[6] Rupprecht, David, et al. "Call Me Maybe: Eavesdropping Encrypted {LTE} Calls With ReVoLTE." 29th {USENIX} Security Symposium ({USENIX} Security 20), pp. 73-88, Aug. 2020.

[7] Continella, Andrea, et al. "ShieldFS: a self-healing, ransomware-aware filesystem." Proceedings of the 32nd Annual Conference on Computer Security Applications, pp. 336-347, Dec. 2016.

[8] Gonzalez, Daniel, and Thaier Hayajneh. "Detection and prevention of crypto-ransomware." 2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON), pp. 472-478, Oct. 2017.

[9] Jung, S., Won, Y. "Ransomware detection method based on context-aware entropy analysis". Soft Computing 22(20), pp. 6731 - 6740. 2018.

[10] Scaife, Nolen, et al. "Cryptolock (and drop it): stopping ransomware attacks on user data." 2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS), pp. 303-312, Jun. 2016.

[11] Suhyeon Lee et al., "Ransomware protection using the moving target defense perspective," Computers & Electrical Engineering, Volume 78, pp. 288-299, Sep. 2019.

[12] K. Cabaj and W. Mazurczyk, "Using Software-Defined Networking for Ransomware Mitigation: The Case of CryptoWall," in IEEE Network, vol. 30, no. 6, pp. 14-20, Nov. 2016.

[13] Kim, Donghyeon, and Kangseok Kim. "DGA-DNS Similarity Analysis and APT Attack Detection Using N-gram." Journal of the Korea Institute of Information Security & Cryptology 28(5), pp. 1141-1151, Oct. 2018

[14] Sehoon Lee, Soram Kim, Giyoon Kim, Daeun Kim, Haeryong Park, Joungsung Kim, "A Study on the Decryption of Donut Ransomware through Memory Analysis", Journal of Digital Forensics, 13(1), pp. 13-22, Mar. 2019.

[15] K. Lee, S. Lee and K. Yim, "Machine Learning Based File Entropy Analysis for Ransomware Detection in Backup Systems," in IEEE Access, vol. 7, pp. 110205-110215, Jul. 2019.

[16] Sehoon Lee, Byungchul Youn, Soram Kim, Giyoon Kim, Yeongju Lee, Daeun Kim, Haeryong Park, Jongsung Kim, "A Study on Encryption Process and Decryption of Ransomware in 2019", Journal of The Korea Institute of Information Security & Cryptology, 29(6), pp.1339-1350, Dec. 2019.

[17] Tech Target, "Ragnar Locker ransomware attack hides inside virtual machine", https://searchsecurity.techtarget.com/news/252483581/Ragnar-Locker-ransomware-attack-hides-inside-virtual-machine, May. 2020.

[18] ZDNet, "Energy company EDP confirms cyberattack, Ragnar Locker ransomware blamed", https://www.zdnet.com/article/edp-energy-confirms-cyberattack-ragnar-locker-ransomware-blamed/, Jul. 2020.

[19] Binary Defense, "Travel Company CWT Pays $4.5 Million USD Ransom to Ragnar Locker Operators", https://www.binarydefense.com/threat_watch/travel-company-cwt-pays-4-5-million-usd-ransom-to-ragnar-locker-operators/, Aug. 2020.

[20] PortandTerminal, "CMA CGM up and ru

nning again following ransomware attack", https://www.portandterminal.com/cma-cgm-up-and-running-again-following-ransomware-attack/, Sep. 2020.
[21] Bleeping Computer, "Campari hit by Ragnar Locker Ransomware, $15 million demanded", https://www.bleepingcomputer.com/news/security/campari-hit-by-ragnar-locker-ransomware-15-million-demanded/, Nov. 2020.
[22] Bleeping Computer, "Capcom hit by Ragnar Locker ransomware, 1TB allegedly stolen", https://www.bleepingcomputer.com/news/security/capcom-hit-by-ragnar-locker-ransomware-1tb-allegedly-stolen/, Nov. 2020.
[23] Bernstein, Daniel J. "Salsa20 specification." eSTREAM Project algorithm description, http://www. ecrypt. eu. org/stream/salsa20pf. html, 2005.
[24] Microsoft Docs, "ExitProcess function (processthreadsapi.h)",https://docs.microsoft.com/en-us/windows/win32/api/processthreadsapi/nf-processthreadsapi-exitprocess, Dec. 2020.
[25] Microsoft Docs, "CryptGenRandom function (wincrypt.h)", https://docs.microsoft.com/en-us/windows/win32/api/wincrypt/nf-wincrypt-cryptgenrandom, Dec. 2020.
[26] Jinseong Park, Seunghee Seo, Yeog Kim, Changhoon Lee, "A Study of the Decryption Method of LockMyPix's Media Files for Forensic Analysis", Journal of Digital Forensics, 14(3), pp. 269-278, Sep. 2020.
[27] Jinseong Park, Seunghee Seo, Byoungjin Seok, Changhoon Lee, "A Research on App Data Decryption Using Encryption Key Reuse Vulnerability in Digital Forensic Perspective", CISC-W'20, pp. 185-188, Nov. 2020

〈저자소개〉

강 수 진 (Soojin Kang) 학생회원
2018년 2월: 국민대학교 정보보안암호수학과 졸업
2020년 3월~현재: 국민대학교 금융정보보안학과 석사과정
〈관심분야〉 디지털 포렌식, 정보보호

이 세 훈 (Sehoon Lee) 학생회원
2019년 2월: 경북대학교 전자공학부 졸업
2019년 3월~현재: 국민대학교 금융정보보안학과 석사과정
〈관심분야〉 디지털 포렌식, 정보보호

김 소 람 (Soram Kim) 학생회원
2016년 2월: 국민대학교 수학과 졸업
2018년 2월: 국민대학교 금융정보보안학과 석사
2018년 3월~현재: 국민대학교 금융정보보안학과 박사과정
〈관심분야〉 디지털 포렌식, 정보보호

김 대 운 (Daeun Kim) 정회원
2015년 2월: 전남대학교 컴퓨터공학과 졸업
2017년 2월: 전남대학교 정보보안협동과정 석사
2017년 3월~현재: 한국인터넷진흥원(KISA)
〈관심분야〉 악성코드, 디지털 포렌식, 빅데이터 분석

김 기 문 (Kimoon Kim) 정회원
2017년 2월: 고려대학교 정보보호대학원(공학석사)
2011년~현재: 한국인터넷진흥원(KISA) 책임연구원
〈관심분야〉 정보보호, 암호 알고리즘

김 종 성 (Jongsung Kim) 종신회원
2000년 8월/2002년 8월: 고려대학교 수학 학사/이학석사
2006년 11월: K.U.Leuven, ESAT/SCD-COSIC 정보보호 공학박사
2007년 2월: 고려대학교 정보보호대학원 공학박사
2007년 3월~2009년 8월: 고려대학교 정보보호기술연구센터 연구교수
2009년 9월~2013년 2월: 경남대학교 e-비즈니스학과 조교수
2013년 3월~2015년 8월: 국민대학교 수학과/일반대학원 금융정보보안학과 조교수
2015년 9월~2017년 2월: 국민대학교 수학과 부교수
2015년 9월~2020년 8월: 국민대학교 일반대학원 금융정보보안학과 부교수
2017년 3월~2020년 8월: 국민대학교 정보보안암호수학과 부교수
2020년 9월~현재: 국민대학교 정보보안암호수학과/일반대학원 금융정보보안학과 교수
〈관심분야〉 정보보호, 암호 알고리즘, 디지털 포렌식