사물인터넷융복합논문지 Vol. 9, No. 1, pp. 1-7, 2023 https://doi.org/10.20465/KIOTS.2023.9.1.001

악성 랜섬웨어 SW에 사용된 암호화 모듈에 대한 탐지 및 식별 메커니즘

이형우*
한신대학교 컴퓨터공학부 교수

Cryptography Module Detection and Identification Mechanism on Malicious Ransomware Software

Hyung-Woo Lee*
Professor, Division of Computer Engineering, Hanshin University

요 약 랜섬웨어에 의해 개인용 단말 또는 서버 등이 감염되는 사례가 급증하고 있다. 랜섬웨어는 자체 개발한 암호화 모듈을 이용하거나 기존의 대칭키/공개 키 암호화 모듈을 결합하여 공격자만이 알고 있는 키를 이용하여 피해 시스템 내에 저장된 파일을 불법적으로 암호화 하게 된다. 따라서 이를 복호화 하기 위해서는 사용된 키 값을 알아야만 하며, 복호화 키를 찾는 과정에 많은 시간이 걸리므로 결국 금전적인 비용을 지불하게 된다. 이때 랜섬웨어 악성코드는 대부분 바이너리 파일 내에 은닉된 형태로 포함되어 있어 프로그램 실행시 사용자도 모르게 악성코드에 감염된다. 그러므로 바이너리 파일 형태의 랜섬웨어 공격에 대응하기 위해서는 사용된 암호화 모듈에 대한 식별 과정이 필요하다. 이에 본 연구에서는 바이너리 파일 내 은닉된 악성코드에 적용 된 암호화 모듈을 역분석하여 탐지하고 식별할 수 있는 메커니즘을 연구하였다.

주제어 : 랜섬웨어, 은닉된 악성코드, 역공학 분석, 탐지 및 식별 메커니즘.

Abstract Cases in which personal terminals or servers are infected by ransomware are rapidly increasing. Ransomware uses a self-developed encryption module or combines existing symmetric key/public key encryption modules to illegally encrypt files stored in the victim system using a key known only to the attacker. Therefore, in order to decrypt it, it is necessary to know the value of the key used, and since the process of finding the decryption key takes a lot of time, financial costs are eventually paid. At this time, most of the ransomware malware is included in a hidden form in binary files, so when the program is executed, the user is infected with the malicious code without even knowing it. Therefore, in order to respond to ransomware attacks in the form of binary files, it is necessary to identify the encryption module used. Therefore, in this study, we developed a mechanism that can detect and identify by reverse analyzing the encryption module applied to the malicious code hidden in the binary file.

Key Words : Ransomware, Hidden Malicious Code, Reverse Engineering, Detection and identification Mechanism.

이 성과의 일부는 2022년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No 2021R1F1A1046954).
*교신저자 : 이형우(hwlee@hs.ac.kr)
접수일 2022년 10월 28일 수정일 2022년 12월 5일 심사완료일 2022년 12월 8일

1. 서론

CryptoAPI, OpenSSL와 같은 오픈 라이브러리를 이용하여 랜섬웨어 또는 각종 악성코드가 포함된 실행 파일이 개발 및 배포되고 있다. 하지만, 악성 바이너리 파일 내에 포함된 암호화 모듈에 대한 역분석 및 사용된 암호화 모듈을 자동 탐지하는 과정이 매우 어렵다[1,2]. 특히, 윈도우즈 운영체제 환경에서 암호화 모듈이 적용된 바이너리 형태의 실행파일(EXE) 및 암호화 라이브러리(DLL) 내부에 은닉된 악성코드를 탐지하고 식별하는데 어려움이 발생하고 있다. 윈도우즈 환경을 대상으로 한 랜섬웨어인 경우 변형된 CryptoAPI를 사용하고 있으며, SSL/TLS 등과 같은 암호 통신 서비스를 제공하기 위해서는 OpenSSL Architecture 기반 SSL/TLS 모듈을 이용하여 랜섬웨어 감염을 위한 제반 동작 방식이 이루어지고 있다[3,4].

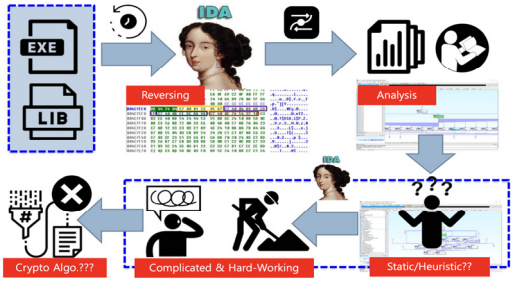
이에 본 연구에는 랜섬웨어 파일의 내부 구조/동작 방식을 분석하였고, CryptoAPI와 OpenSSL 등을 이용하여 개발된 바이너리 실행파일에 대한 역분석 과정을 통해 악성코드 내에 적용된 암호화 모듈에 대한 탐지 및 식별 메커니즘을 제시하였다. 이를 위해 윈도우즈 암호화 모듈 적용시 동작 방식 및 특징에 대한 분석하였으며, 바이너리 실행파일로부터 암호화 방식(블럭/공개키/해쉬/인증) 수행에 따른 특징 정보 추출 과정을 수행하였다. 그리고 기존 CryptoAPI와 OpenSSL 라이브러리를 적용한 랜섬웨어 실행파일 구동 과정에서 발견되는 바이너리 패턴에 대한 분석과정을 수행하였다. 이를 통해 바이너리 파일 실행시 메모리, CPU, I/O 등에 대한 정적/동적 역분석을 통해 행위 분석 및 API 호출 특성에 대한 분석과정[5]을 수행하였고, IDA Pro 도구를 이용하여 암호화 모듈 적용 바이너리 실행파일에 대한 특징 정보 추출/분석할 수 있는 플러그인 프로그램을 개발하였다. 본 연구를 통하여 윈도우즈 환경에서 개발된 랜섬웨어 형태의 암복호화 실행파일 및 DLL 라이브러리 등에 적용된 암복호화 알고리즘을 자동으로 식별할 수 있는 메커니즘을 제공하였다.

2. 랜섬웨어에 사용된 암호화 모듈 탐지 기술 조사 및 분석

2.1 암호학적 특성 자동 탐지 및 식별 방법

랜섬웨어 바이너리 실행파일 개발 과정에 사용된 암호 알고리즘과 매개 변수를 역분석하여 분석 대상 파일의 동작 과정에서 추출 가능한 암호학적 특성을 자동 탐지 및 판별하여 랜섬웨어 파일에 적용된 내부 암호화 모듈을 탐지 및 분석하는 방법(Automatic Identification of Cryptographic Primitives in Ransomware)을 적용할 수 있다. Dynamic Binary Instrumentation(DBI) 과정을 통해 바이너리 코드를 동적으로 삽입한 후 소스 코드를 수정하지 않고도 API Trace, 인자 조작 및 반환 값 획득 과정을 수행할 수 있다. 이를 위해 IDA-PRO를 이용하여 FindCrypt 플러그인을 사용할 경우 실행 파일 내에 포함된 암호화 모듈을 찾아내는 과정을 수행할 수 있다. 하지만, 이는 상수 기반 분석 방식이다 보니 변경된 암호화 모듈을 찾을 수는 없으며, 특정 상수 정보를 포함하지 않는 암호화 모듈에 대해서는 탐지률이 떨어진다는 문제점을 있다[5-8].

만일 PEiD – Krypto ANALyzer를 사용할 경우 실행파일에 포함된 PE 파일을 분석하여 사용된 패커를 탐지하는 기능을 제공한다. 또한 PE 기반으로 암호 알고리즘과 컴파일러를 감지하고 어떤 언어로 제작된 프로그램인지를 분석할 수 있는 툴이다. 하지만 이 또한 랜섬웨어 파일 내 은닉된 암호화 모듈 분석이 어렵다.

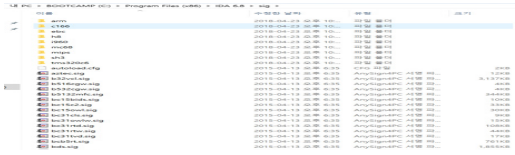


[Fig. 1] Difficulty in reverse analysis of basic ransomware encryption module

따라서 본 연구에서는 Hash & Crypto Detector를 사용하여 PE 파일에 사용된 해시 알고리즘과 암호 알고리즘 및 사용된 컴파일러를 탐지할 수 있다. 직관적인 GUI 인터페이스를 제공하며 탐지 속도가 상당히 빠르고 통합된 Shell과 Commend line 방식을 지원한다.

또한 IDA Pro FLIRT는 IDA Pro 분석 시 실행파일 내에 사용된 라이브러리에 포함된 함수들의 시그니처를 기반으로 해당 함수를 판별하고 구체적으로 함수의 이름 정보를 제공하는 기능을 포함하고 있다. 물론 대단위 라

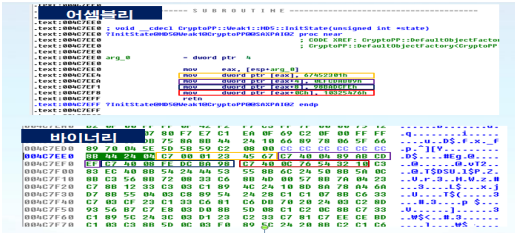
이브러리 내에서 유사한 시그니처가 검출될 수 있기 때문에 IDA Pro FLIRT에서는 시그니처 생성 시 충돌 예외처리 옵션 기능을 포함하고 있다[9,10].



[Fig. 2] Signature-based encryption module reverse analysis process

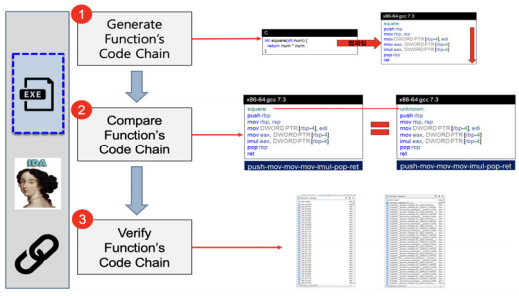
2.2 암호화 모듈 탐지 기법

랜섬웨어 등 바이터리 실행파일 형태의 SW 내에 포함된 암호 알고리즘을 자동 탐지하기 위해 사용하는 가장 간단한 방법은 상수 기반 탐지 방식이다. 암호 알고리즘 내에 사용되는 상수(배열)을 추적하여 소프트웨어 내에 적용된 암호 알고리즘을 탐지하는 방법이다.



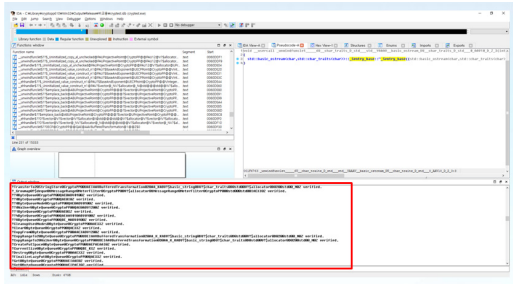
[Fig. 3] Constant-based cryptographic module detection

상수 기반 탐지 방식의 한계점을 개선하기 위해 제시된 방법으로 코드 체인(시퀀스) 기반 탐지 방식은 암호 알고리즘의 바이너리(어셈블리) 패턴을 분석하여 분석 대상 SW 바이너리 실행 파일 내에 사용된 함수를 탐지하는 방식이다[11-13].



[Fig. 4] Code chain (sequence) based encryption module detection

예를들어 cryptest.exe 파일을 대상으로 코드 체인 분석을 수행하게 되면 코드 체인 정보로 구성된 함수가 총 11,945개 중에서 동일한 코드 체인을 갖는 시퀀스(충돌) 1,945개를 확인할 수 있으며 이는 동일한 opcode(mov-lea-mov-xor-call-mov-jmp)인 것을 확인할 수 있다. 하지만 여전히 오탐률이 존재하며 일반적인 형태의 바이너리 실행파일에서는 더 높은 오탐률이 나타날 가능성이 있다[14,15]. 따라서 이를 해결하기 위해서 Fuzzy Hashing 기법을 적용할 수 있다. Fuzzy Hashing 기반 탐지는 사용되는 암호 알고리즘 함수에 대해 Fuzzy Hash 시그니처를 생성하고 그것을 기반으로 유사도를 측정하여 사용여부를 판별하는 방법이다.

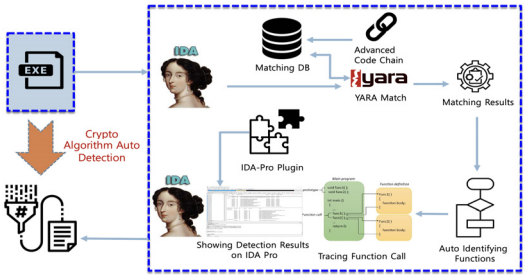


[Fig. 5] Analysis of cryptographic modules based on fuzzy hashing and measurement of similarity

3. 제안하는 암호화 모듈 탐지 및 식별 기법

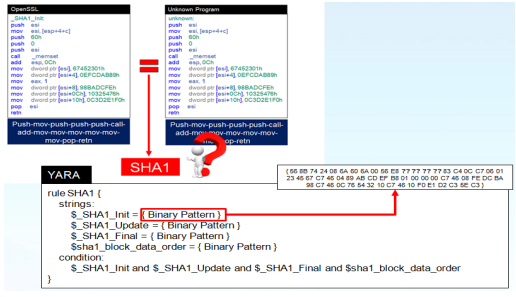
3.1 개선된 코드 체인 분석 방식

기존의 코드 체인 방법은 Operand를 제외한 상태로 Opcode만을 비교하므로 Operand의 영향을 받지 않아 탐지가 간단하고 단순 검색으로 탐지속도가 빠르다는 강력한 장점을 갖고 있다. 하지만 Operand 값에 상수가 들어가는 경우 특징값(시그니처 변수)으로 선별되지 않고, 또한 Operand는 다르나 Opcode가 같은 Code Chain의 경우 처리할 수 없다는 문제점이 발생한다. 따라서 이와 같은 문제점을 개선하기 위해 개선된 코드 체인 기반 탐지 방법을 제시한다.



[Fig. 6] Proposed cryptographic module detection procedure

Fuzzy Hashing 기반 탐지 방법을 적용할 경우 라이브러리 함수가 컴파일 될 때마다 매번 다른 주소에 할당되며, 코드 블록 전체를 대상으로 유사도 분석 과정을 진행하면 알고리즘 탐지 효율을 높일 수 있으며, 결과적으로 오탐률을 낮출 수 있다. 따라서 이와 같은 기능을 제공하기 위해 Yara 룰을 적용하여 자동화된 탐지 기능을 수행할 수 있도록 구현하였다. YARA를 사용하면 Binary Pattern 부분에 Code Chain을 그대로 사용할 수 있고, 하나의 Yara 룰을 이용하여 여러 개의 함수에 대한 매칭(유사도 측정)이 가능하다.

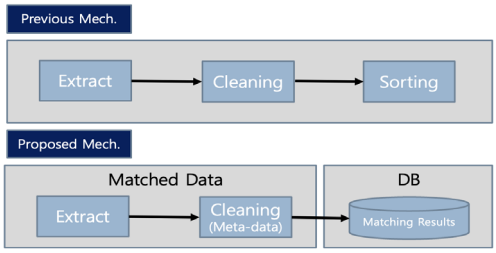


[Fig. 7] Improved code chain analysis method based on Yara rules

3.2 Yara 룰 기반 코드 체인 분석 메커니즘

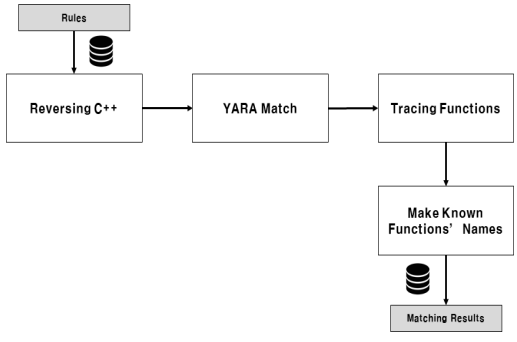
기존 방식의 경우 알고리즘 자동 검출을 위한 유사도 매칭 시 비교 대상이 되는 데이터를 추출, 정제 그리고 정렬하는 과정을 수행한다. 하지만 이와 같은 정제와 정렬 과정에 매번 다른 방식으로 코드를 구현해야 한다는 문제점이 있다.

따라서, 본 연구에서 제안된 방식의 경우 전체 매칭 데이터를 메타 데이터로 정제하여 데이터베이스에 저장한 후 데이터베이스에 질의어를 통해 필요한 메타 데이터를 원하는 형태의 데이터로 정렬/획득할 수 있다. 따라서 불필요한 코드를 만들 필요가 없어지며 최소화된 코드만으로도 탐지 및 검출 과정을 수행할 수 있다. 데이터베이스를 사용하여 이전 분석 결과를 저장하기 때문에 큰 프로그램의 경우 재분석을 할 필요가 없으며, 메타 데이터를 이용하여 매번 원하는 형태의 결과를 출력할 수 있으므로 데이터 재가공과 유연성을 제공한다.



[Fig. 8] Analysis Method Comparison

본 연구에서 제시한 암호 알고리즘 탐지/검출을 위한 플러그인의 전체 흐름도는 아래 그림과 같이 Yara 룰 파일을 입력받고 분석 전에 해당 프로그램에 대한 데이터베이스의 존재 여부를 확인한다. 만일 분석 과정을 수행할 경우 Reversing C++ 모듈에서 RTTI, Trace Virtual Functions, Detect This Pointers 모듈을 수행하게 된다. 룰과 분석된 결과 그리고 분석할 프로그램의 정보를 가지고 Yara Match 모듈을 수행하게 된다. 매칭된 결과를 가지고 IDA Pro에서 제공하는 API로 Tracing Function과 매칭된 함수에 대해서 이름을 변경한 후 해당 결과를 다시 DB에 저장하게 되며 최종적으로 IDA Pro에 GUI 형태로 탐지 결과를 함수의 시작주소, 룰 이름, 함수 이름, 참조 호출 타입, 참조 호출 주소, 참조 호출 정보를 표출해준다.

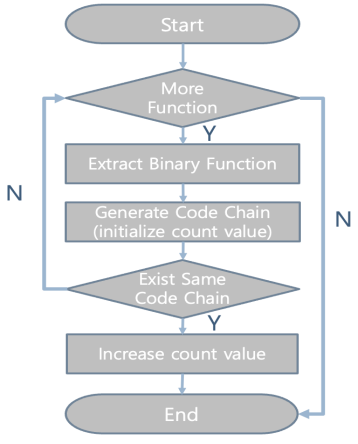


[Fig. 9] Encryption module identification process based on Yara rule matching

3.3 개선된 코드 체인 분석 모듈 설계 및 구현

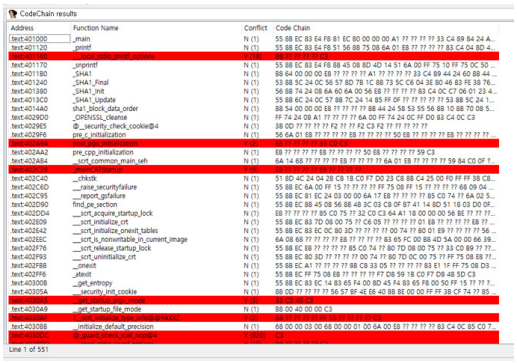
본 연구에서 제시한 개선된 코드 체인 기반 분석 모듈은 입력된 실행 파일 내 모든 함수를 순차 탐색하여 각각의 함수에 대한 바이너리 정보를 추출한다. 그리고 이를 코드 체인 리스트 값으로 생성하고 동일한 코드 체인 여부를 확인하기 위해 IDA Pro에서 제공하는 검색 기능을 이용해 프로그램 전체 블록을 대상으로 동일한 코드 체인의 개수를 카운트한다. 이와 같은 과정을 반복하며 모든 함수에 대해 추출과 생성 그리고 검색을 반복/탐색한 후 종료하게 된다.

아래 순서도에서 코드 체인 생성 과정은 앞부분에서 제시한 Yara 룰 기반 최적 매칭 기법이 적용된 개선된 코드 체인 기법을 적용하였다.



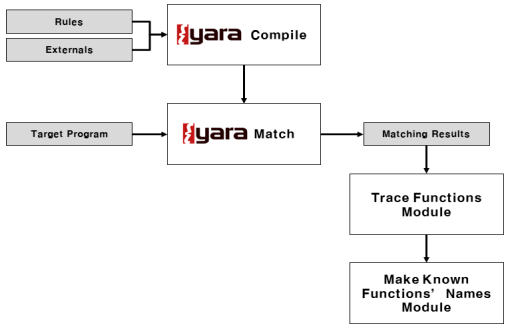
[Fig. 10] Cryptographic module identification algorith

OpenSSL 라이브러리를 이용한 SHA1 바이너리 실행 파일을 대상으로 IDA Pro에서 로딩후 개선된 코드 체인 모듈을 적용한 결과는 다음 그림과 같이 실행 파일 내에 모든 함수에 대해 코드 체인이 생성된 것을 확인할 수 있다.



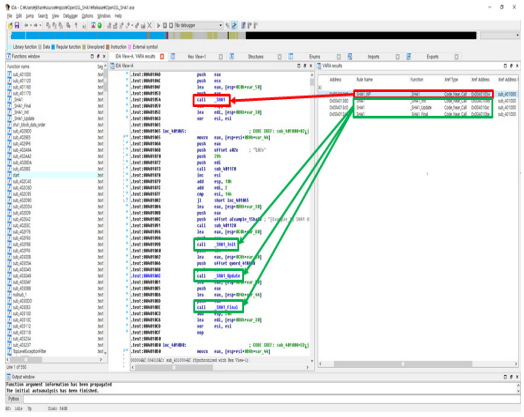
[Fig. 11] Result of code chain generation when applying the proposed technique

이와 같은 과정을 다시 전체 흐름 순으로 정리하면 다음과 같다. Yara 룰과 추가 정보(Extensions)를 이용하여 Yara compiler를 생성한 후 바이러니 실행파일을 대상으로 Yara match 과정을 수행하게 된다. 그리고 코드 체인 정보를 이용하여 매칭된 결과가 도출되면 해당 함수와 모듈을 상세 추적하는 과정을 수행하고 최종적으로 발견된 암호화 모듈 등에 대해 이름을 부여하게 된다.



[Fig. 12] Encryption module determination process based on Yara rule matching

본 연구에서 제안한 개선된 코드 체인 메커니즘을 이용하여 OpenSSL 라이브러리를 이용한 SHA1 바이너리 실행파일에 대한 역분석 결과는 다음과 같다. 아래 그림과 같이 바이너리 코드 내에 사용된 SHA1 알고리즘을 정확하게 판별/검출할 수 있었다.

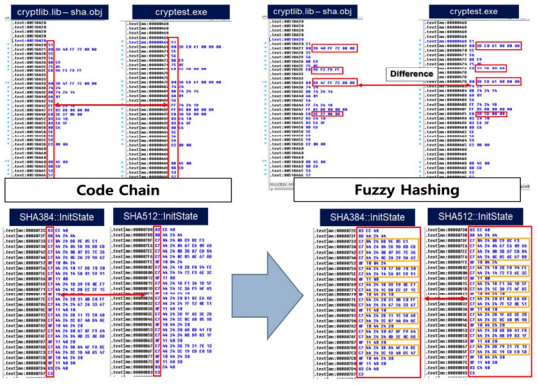


[Fig. 13] Improved code chain analysis result applying the proposed technique

4. 랜섬웨어 내 암호화 모듈 탐지 및 식별

4.1 바이너리 실행파일 역분석 결과

본 연구에서 제시한 방법을 적용하였을 경우 아래 그림과 같이 기존의 코드 체인 방법을 적용하였을 때 보다 퍼지 해싱 기법을 적용할 경우 바이너리 특징 패턴에 대한 분석 정확도가 향상되었음을 확인할 수 있었으며, 최근 큰 피해를 유발하고 있는 랜섬웨어 내에 적용된 해시 알고리즘을 판별해 낼 수 있었다.



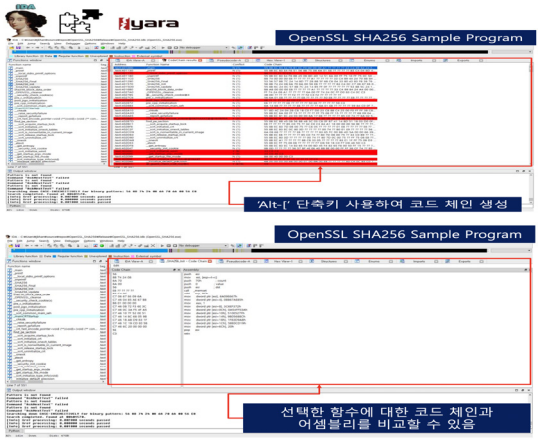
[Fig. 14] Comparison of discrimination results through the existing technique and the proposed technique

4.2 바이너리 실행파일 내 암호화 모듈 식별

퍼지 해싱 기법을 적용하였을 경우 기존의 방식보다 45.2% 이상 역공학 분석 과정에서 어셈블 코드에 대한 특정화 과정이 개선되었음을 확인할 수 있었으며, 실험 대상 파일을 대상으로 검토한 결과 역분석 과정에서 발생하는 오탐율 역시 37.8% 이상 개선된 것을 확인할 수 있었다. 본 연구에서 제시한 암호화 모듈 탐지 및 식별 메커니즘을 적용할 경우 기본 방법 보다 41.5% 이상의 랜섬웨어 역분석 과정의 효율성 및 정확도 향상이 있었음을 확인할 수 있었다.

5. 결론

최근 랜섬웨어로 인한 피해가 급증하고 있다. 이에 본 연구에서는 랜섬웨어 바이너리 파일 내에 사용된 암호화 모듈을 분석 하고 이를 자동탐지하기 위해 개선된 역공학 분석 메커니즘에 대해 제시하였다. 기존 역분석 방법의 특징과 구조에 대한 분석을 통해 Yara 룰 자동 매칭 기반 형태의 개선된 코드 체인 기반 역분석 방법을 도출할 수 있었다. IDA 기반 역분석 과정에서 추출되는 어셈블리 코드 정보를 토대로 각각의 암호화 모듈에 대한 코드 체인을 생성하였으며, 퍼지 해싱 기법이 적용된 Yara 룰 기반 자동 매칭 과정을 통해 랜섬웨어 파일 내에 은닉된 암호화 모듈을 판별하는 과정에 적용하였다. 탐지 결과의 정확도를 측정하기 위해 PDB 기반으로 함수 매칭 결과를 비교하였으며 분석 결과 본 연구에서 구현한 YARA Match 모듈인 경우 랜섬웨어 파일 내에 포함된 AES 및 SHA 암호화 모듈 등을 정확하게 분석/탐지하는 것을 확인할 수 있었다.



[Fig. 15] Encryption module identification result through the proposed technique

REFERENCES

[1] 'Ransomware Latest Trend Analysis and Implications', Digital & Security Policy, KISA Insight, Vol.02, 2021, Korea Internet & Security Agency, https://www.kisa.or.kr/20301/form?postSeq=4&page=1

[2] C. Beaman, A. Barkworth, T. D. Akande, S. Hakak, M. H. Khan, "Ransomware: Recent advances, analysis, challenges and future research directions," Computer & Security, Vol.111, 2021, 102490

[3] H. Alshaikh NR, Nagy, H. Hefny, "Ransomware prevention and mitigation techniques." International Journal of Computer Applications, Vol.117, No.40, pp.31-39, 2020.

[4] P. Bajpai, R. Enbody, "An empirical study of API calls in ransomware," IEEE International Conference on Electro Information Technology (EIT); 2020, pp. 443-448.

[5] B. Qin, Y. Wang, C. Ma, "API call based ransomware dynamic detection approach using textCNN," 2020 International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE); 2020. pp.162-166.

[6] J. Y. Moon and Y. H. Chang, "Ransomware analysis method for minimize the damage," The Journal of the Convergence on Culture Technology, Vol.2, No.1, pp.79-85, 2016.

[7] J. Y. Kim, "A study on the recovery of ransomware infected file through real-time file behavior analysis," Master's Thesis, Korea University, May. 2017.

[8] S. W. Yoon, M. S. Jun, "A Study on a Method of Identifying a Block Cipher Algorithm to increase Ransomware Detection Rate," Journal of The Korea Institute of Information Security & Cryptology, Vol.28, No.2, Apr. 2018.

[9] "Wannacry report," https://www.pandasecurity.com/mediacenter/src/uploads/2017/05/WannaCry_Report-en.pdf

[10] "SimpleLocker Ransomware Encryption Function Analysis Report," July. 2019, Korea Internet & Security Agency,

[11] "immuni Ransomware Encryption Function Analysis Report," Dec. 2020, Korea Internet & Security Agency,

[12] Hassannataj Joloudari, J., Haderbadi, M., Mashmool, A., GhasemiGol, M., Shahab, S., and Mosavi, A., "Early detection of the advanced persistent threat attack using performance analysis of deep learning", arXiv e-prints, 2020.

[13] Md Sahrom Abu, Siti Rahayu Selamat, Aswami Ariffin, Robiah Yusof, "Cyber Threat Intelligence - Issue and Challenges," Indonesian Journal of Electrical Emgineering and Computer Science, Vol.10, No.1, pp.371-379, 2018.

[14] H.Lee, "Intrusion Artifact Acquisition Method based on IoT Botnet Malware," Journal of The Korea Internet of Things Society, Vol.7, No.3, pp.1-8, 2021.

[15] Maria Stoyanova, Yannis Nikoloudakis, Spyridon Panagiotakis, Evangelos Pallis, and Evangelos K. Markakis, "A Survey on the Internet of Things (IoT) Forensics: Challenges, Approaches, and Open Issues," IEEE COMMUNICATIONS SURVEYS & TUTORIALS, Vol.22, No.2, pp.1191-1221, 2020.

이 형 우(Hyung-Woo Lee) [종신회원]

- 1994년 2월 : 고려대학교 컴퓨터학과 (학사)
- 1996년 2월 : 고려대학교 컴퓨터학과 (석사)
- 1999년 2월 : 고려대학교 컴퓨터학과 (박사)
- 1999년 3월 ~ 2003년 2월 : 백석대학교 정보통신학부 교수
- 2003년 3월 ~ 현재 : 한신대학교 컴퓨터공학부 교수

〈관심분야〉
사물인터넷, 정보보호, 모바일 보안 및 디지털 포렌식