EasyChair Preprint

№ 3697

Implementation of Network Sniffing Tool with Carving

Seo-jeong Kim, Yun-Hee Kim, Yu-jin Jeon and Eun-jung Choi

June 29, 2020

카빙을 지원하는 네트워크 스니핑 툴 개발

김서정*, 김윤희*, 전유진*, 최은정*

*서울여자대학교 정보보호학과

Implementation of Network Sniffing Tool with Carving

Seo-jeong Kim, Yun-hee Kim, Yu-jin Jeon, Eun-Jung Choi

*Department of Information Security, Seoul Women's University

요 약

본 논문에서는 카빙 모듈을 탑재한 네트워크 트래픽 모니터링 툴을 소개한다., 네트워크 트래픽을 실시간 모니터링을 통해 수집하여, 변조되기 이전의 파일 데이터를 저장 후 파일을 복구한다. 따라서, 원본 파일로 복구할 수 있는 카빙 모듈을 개발하여 소규모 네트워크 혹은 모니터링 및 포렌식 기능이 필요한 집단에서 간단하게 사용할 수 있도록 한다.

여러 개의 파일 시그니처를 구분하여 서로 다른 확장자 파일까지 네트워크 트래픽 스니핑 만을 통해 실시간으로 복구하며, 푸터 시그니처가 없어 파일의 끝을 명확히 알 수 없는 파일들을 복구하는 방안도 함께 제안한다.

I. 서론

1. 연구선정 배경 및 필요성

최근 인터넷 통신 증가로 파일 송수신 통신의 양 또한 증가하고 있다. 파일 카빙의 경우 시스템 로그 혹은 하드 디스크 외에 필요에 따라 패킷 파일과 같은 저장 매체에 저장되는 원본 데이터를 파일로 다시 복구해야 하는 경우가 발생한다. 실시간으로 수집되는 패킷의 송수신 데이터가 조작되거나 변조되기 이전에 실시간으로 데이터를 수집하여 복구해야 하는 경우가 발생하기 때문에 파일 카빙 기능을 제공하는 네트워크 스니핑 툴을 제안하고자 한다.

II. 본론

1. 스니핑(Sniffing) 구현

파이썬 scapy 라이브러리를 사용[1]하여 패킷 스니핑에 활용하였다. scapy 라이브러리는 간단한 몇 개의 명령어 만으로 쉽게 패킷 스니핑, 스캐닝, 스푸핑, 네트워크 커넥션, DDoS 공격을 할 수 있는 도구이자 강력한 보안 솔루션 공개 도구로, 전체적인 틀은 Github에 공개된 오픈소스[2]를 활용하였다.

네트워크 인터페이스는 기존 소스를 활용하였으며, 추가로 파일 카빙 기능과, TCP 프로토콜 선택 후 하위 프로토콜인 HTTP 와 HTTPS 프로토콜까지 필터링할 수 있도록 기능을 추가하였다[3].

또한 패킷 전송 MTU 사이즈가 1500 byte이기 때문에 패킷의 크기가 크면 한번의 통신이 패킷 여러 개로 나뉘어 전송된다[4]. 따라서 이후 카빙 기능을 구현할 때 나뉘어 전송되는 패킷의 데이터를 추출하는 방법에 대해 소개하도록한다.

2. 카빙(Carving)

2.1 카빙 개념

파일 자체의 바이너리 데이터를 이용해 디스

크의 비할당 영역에서 파일을 복구하는 방식을 의미하며, 쉽게 말해 바이너리 데이터에서 정보 획득 후 할당 되지 않은 영역에서 파일을 복구하는 것이다[5].

시그니처 기반 카빙[6]은 메타 데이터를 이용하지 않기 때문에 고유의 특성으로 복구해야 한다. 각 파일의 포맷 별로 존재하는 파일 시그니처를 이용하는 방법으로 Header 시그니처와 Footer 시그니처가 존재한다.

다음 표는 주요 확장자의 시그니처이다.

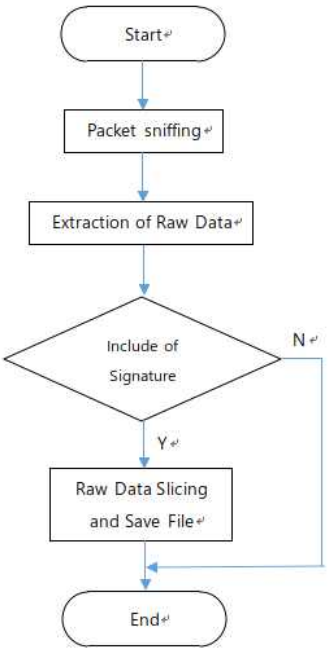
File format	Signature(Hex)	
	Header	Footer
JPEG	FF D8	FF D9
PNG	89 50 4E 47 0D 0A 1A 0A	49 45 4E 44 AE 42 60 82
PDF	25 50 44 46 2 D 31 2E	25 25 45 4F 46
HWP	D0 CF 22 E0 A1 B1 1A E1	-
EXE	48 57 50 20 4 4 6F 63 75 6 D 65 6E 74 2 0 46 69 6C 65 / D0 CF 11 E 0 A1 B1 1A E1	-
DOCX	50 4B 03 04 1 4 00 06 00	-

[표 1]

2.2 카빙 구현 방법

가공되지 않은 raw data가 패킷을 통해 그대로 전송되기 때문에, 나뉘어 전송되는 패킷을 추합하기 위해 syn number가 같은 패킷끼리 모아서 데이터를 추출해야 한다.

이 과정에서 수동으로 패킷의 raw data를 하나 씩 클릭하여 추출했었으나 비효율적이고 데이터 손실이 일어나 제대로 복구가 되지 않는 문제가 발생하여 sniffing 후 데이터가 들어있는 패킷에 한해서 자동으로 raw를 모두 추출한 후 binary 형태로 변환하여 파일 형태로 저장한다.



[순서도 1]

다음 과정에서 binary 파일을 불러오고 파일 확장자 별 헤더 시그니처를 정규화[7] 한 후 매칭 시키는 과정을 거친다.

```
#SOF_list is File signature for carving
If SOF_list exist then
Do
        for SOF to SOF_list do
                for _ to len(SOF_list) do
                        set subdata to data[SOF: ]
                        carve_obj generated
                        open carve_obj file
                        write subdata to carve_obj
                        close carve_obj file

                        break
                delete SOF_list
                break
```

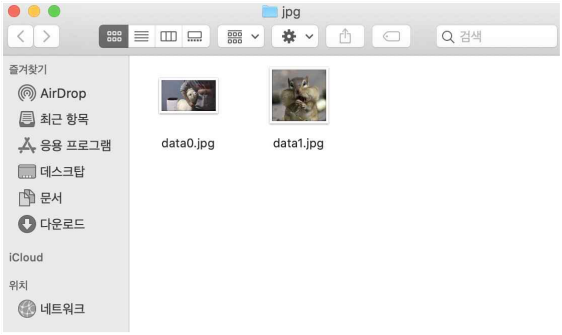
[그림 1]

매칭에 성공하면 해당 binary 데이터 위치를 리스트에 추출하여 저장하고, 확장자 별로 저장하여 파일을 추출하고 다음 확장자 리스트를 구축하기 전 리스트를 초기화하도록 한다.

기존 jpg 파일로 테스트할 경우에는 푸터 시그니처가 존재하여 헤더부터 푸터까지 슬라이

싱하는 과정으로 복구하였으나, docx 등과 같은 푸터 시그니처가 명확하지 않은 파일은 새로운 슬라이싱 기준이 필요하다.

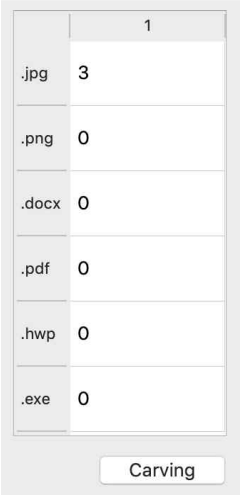
따라서 헤더 시그니처(jpg, png, pdf, hwp, docx, exe)를 담은 리스트에서, 다음 헤더 시그니처가 나오기 전까지 추출[8]하여 복구하였고,



[그림 2]

추출할 때마다 파일명을 변경하며 자동으로 확장자 별 폴더에 파일을 복구[9]하도록 하였다.

또한 코드 효율성을 위하여 파일 시그니처를 딕셔너리화 하여 사용하였으며, 이를 활용하여 스니핑 된 패킷들의 raw 를 검사하여 확장자 개수를 카운트하고 메인 윈도우에 개수를 출력하도록 한다.



[그림 3]

카빙 자체를 옵션으로 주어, 사용자가 원할 시에만 복구하도록 하여 툴의 효율성을 높였다.

III. 결론 및 향후 계획

본 논문에서 제안하는 카빙 모듈을 탑재한 네트워크 트래픽 모니터링 툴은, 네트워크 트래픽을 실시간으로 수집하며 모니터링하는 동시에 실시간으로 변조되기 이전의 파일 데이터를 저장하고 원본 파일로 복구할 수 있는 카빙 모듈을 활용하여 소규모 네트워크 혹은 모니터링 및 포렌식 기능이 필요한 집단에서 간단하게 사용할 수 있다.

추후 복구되는 파일의 악성파일 유무, DoS 와 같은 트래픽 기반 서비스 공격을 탐지[10]할 수 있는 기능을 추가하여 모니터링 툴의 기능도를 향상시킬 예정이다.

감사의 글

본 연구는 과학기술정보통신부 및 정보통신기술진흥센터의 SW 중심대학지원사업의 연구결과로 수행되었음.(2016-0-00022)

[참고문헌]

[1] https://scapy.readthedocs.io/en/latest/

[2] Github(https://github.com/PENGZhaoqing/SimpleSniff)

[3] Spitznagel EW. High performance packet classification. [Order No. 3207251]. Washington University in St. Louis; 2005.

[4] 정성모, 송재구, 김석수, 박길철. (2009). 패킷 스니핑을 활용한 효율적인 모니터링 시스템에 관한 연구. 한국정보기술학회 종합학술발표논문집, (), 587-590.

[5] 박세환, 최용수, 박종규.「」20146디지털 포렌식(Digital Forensics) 기술 동향

[6] 김진국, 이승봉, 박정흠, 방제완, 이상진. (2008). 파일카빙 기법에 관한 연구. 디지털포렌식연구, (2), 19-31.

[7] http://www.thehexninja.com/2018/01/practical-exercise-image-carving-ii.html?m=1

[8] 박민수, 박정흠, 이상진, (2013), Record File Carving Technique for Efficient File Recovery in Digital Forensic Investigation, (), 2-3

[9] 장지원, 방승규, 한재혁, 이상진. (2017.1). PDF 파일의 페이지 단위 복구 기법. 정보처리학회논문지/컴퓨터 통신 시스템 제6권 제1호, () , 3-6

[10] 홍성혁, 서유정, (2016), 효율적인 Sniffing 공격 대응 방안 연구,(),31-36