|  |
| --- |
| **IOT기기 디지털 포렌식 도구 개발**  팀 이웃(IoT) - 김시환(2019102158)  정태규(2020105661)  **요 약**  디지털 포렌식은 현대 사회에서 중요한 역할을 수행하고 있는 분야 중 하나로, 디지털 기기 및 시스템에서 증거를 수집하고 분석하여 범죄나 인위적인 활동을 조사하는 핵심 기술이다. 이러한 디지털 포렌식 기술은 사회, 경제, 법 집행, 기업 보안 등 다양한 분야에서 큰 영향을 미치고 있으며, IoT 기기가 점차 증가함에 따라 이러한 기기에 대한 조사 필요성이 더욱 커지고 있다. |

**1. 서론**

**1.1. 연구배경**

과거에는 컴퓨터와 스마트폰과 같은 전통적인 디지털 기기가 디지털 포렌식의 주요 대상이었다. 그러나 현재의 사회에서는 IoT 기기가 우리 주변에 널리 보급되어 있다. 스마트 홈 시스템, 스마트 카메라, 스마트 냉장고 및 의료 기기와 같은 IoT 기기는 우리의 일상 생활에 녹아들어 있으며, 기업과 정부 기관에서도 널리 사용되고 있다. 이러한 IoT 기기는 다양한 데이터를 생성하고 저장하며, 이 데이터는 사건 및 사고의 중요한 증거로 사용될 수 있다.

IoT 기기는 기존의 디지털 포렌식 환경과 다른 독특한 특성을 가지고 있어 기존 도구 및 기술로는 충분히 대응하기 어려운 도전 과제를 제기한다. 그 중 하나는 IoT기업마다의 다양한 파일시스템의 사용이다. IoT 기기는 다양한 플랫폼과 운영 체제로 구성되어 있으며, 일반적인 컴퓨터나 스마트폰과는 다른 특성을 가지고 있다. 또한 메모리가 제한적이기 때문에 디지털 포렌식 도구의 성능 및 용량 제약이 중요한 고려 사항이다.

IoT 기기에 대한 적절한 디지털 포렌식 도구의 부재는 범죄 조사, 기업 보안 사고 대응, 사회 문제 해결 및 개인 정보 보호에 대한 심각한 리스크를 초래할 수 있다. 따라서 IoT 기기의 디지털 포렌식을 위한 효율적이고 특화된 도구가 필요하다.

따라서 이 프로젝트는 IoT 기기에 대한 디지털 포렌식을 수행하기 위한 고도로 특화된 도구를 개발하는 데에 중점을 두고 있으며, 이러한 도구는 현재의 디지털 포렌식 커뮤니티와 법 집행 기관에 큰 가치를 제공할 것으로 기대된다. 또한, 이 프로젝트는 보안 및 개인 정보 보호에 대한 새로운 관점을 제시하여 사회적으로 의미 있는 결과물을 얻을 것이다.

**1.2. 연구목표**

우리 연구의 주요 목표는 리눅스 기반의 IoT 기기에서 디지털 포렌식을 수행하기 위한 도구를 개발하는 것이다. IoT기기는 OpenWrt ,BusyBox등과 같은 각 기업마다의 플랫폼을 사용하여 동작하는데, 대부분의 기기는 리눅스 운영 체제를 기반으로 한다. 우리는 이러한 리눅스를 기반으로 하고 있는 기기에 대한 디지털 포렌식 전문가 및 수사관이 증거 수집 및 분석을 보다 효과적으로 수행할 수 있는 도구를 개발하고자 한다.

이 연구의 중점은 IoT 기기의 저장된 데이터를 복구하고 분석하는 능력을 개발하는 것이다. IoT 기기는 다양한 센서 및 로그 데이터를 생성하며, 이 데이터는 사건 조사 및 사고 분석에 중요한 역할을 한다. 따라서 이 도구는 파일 시스템, 로그 파일, 데이터베이스 등 다양한 데이터 소스에서 데이터를 추출하고 복구하는 데 사용될 것이다.

이를 위해서는 리눅스 기반으로 된 OpenWrt나 BusyBox와 같은 경량 리눅스 환경을 갖는 IoT 기기의 파일 시스템에 대한 연구가 선행되어야 할 것이다. 따라서 리눅스 파일 시스템과 리눅스를 기반으로 하는 OpenWrt와 BusyBox에 대한 파일시스템에 대한 연구를 진행할 예정이다.

**2. 관련연구**

**2.1. 디지털 포렌식**

정보기기에 내장된 디지털 자료를 근거로 삼아 그 정보기기를 매개체로 하여 발생한 어떤 행위의 사실 관계를 규명하고 증명하는 보안서비스 분야로서, 디지털 포렌식은 크게 증거 수집, 증거 분석, 증거 제출과 같은 절차로 이루어진다.

**2.1.1. 증거 수집**

보통의 디지털 기기는 운영체제가 탑재되어 있으며, 운영체제는 휘발성 저장매체와 비휘발성 저장매체를 사용한다. 디지털 포렌식에서 증거 수집은 대상 매체의 운영체제 종료 여부에 따라 데드 시스템상에서의 증거 수집과 라이브 시스템상에서의 증거수집으로 나뉜다.

* **데드 시스템 상에서의 증거 수집**

운영체제가 종료된 컴퓨터나 핸드폰 같은 기기에 대한 증거 수집을 말하며, 주로 하드디스크나 플래시 메모리로부터 데이터를 얻는 것으로 이루어진다. 본 저장매체에 있는 데이터의 무결성을 보장할 수 있도록 원본 데이터를 복제하는 이미징 기술이 사용된다.

* **라이브 시스템상에서의 증거수집**

운영체제가 종료되지 않은 컴퓨터나 핸드폰 같은 기기에 대한 증거 수집을 말한다. 하드디스크와 같은 비휘발성 매체뿐만 아니라 컴퓨터 메모리와 같은 휘발성 저장매체로부터 데이터를 얻는 것으로 이루어진다. 라이브 시스템상에서의 디지털 포렌식을 수행하기 위해서는 운영체제가 사용중인 휘발성 메모리와 하드디스크를 접근할 필요가 있다. 하지만 윈도우와 같은 운영체제에서는 중요한 메모리 영역이나 하드디스크상의 파일에 대한 사용자 프로그램의 접근을 막고 있다. 따라서 라이브 시스템상에서의 포렌식을 위해서는 운영체제의 보호기능을 우회할 수 있는 기술이 필요하다.

**2.1.2. 증거 분석**

증거 수집에서 얻어진 데이터로부터 유용한 정보를 얻는 것을 말하며, 일반적으로 다음과 같은 증거 분석 기술들이 사용된다.

* **덤프 메모리 분석**

프로세스가 사용중인 가상 메모리의 덤프를 획득했을 경우에 사용자 ID나 패스워드와 같은 유용한 정보가 가상 메모리에 남아 있을 수 있다. 프로세스를 위한 가상 메모리는 보통 코드 영역, 데이터 영역, 스택 영역 등으로 나뉘어지며, 데이터 영역이나 스택 영역이 프로세스에서 필요한 여러 정보를 저장하고 있으므로 프로세스가 가상 메모리를 어떻게 사용하는지를 분석하는 덤프 메모리 분석 기술이 사용된다.

* **Windows 레지스트리 분석**

윈도우의 경우, 레지스트리(registry)에 프로그램이나 시스템에 관한 다양한 정보를 저장하고 있다. 레지스트리 Hive 파일들 중 SAM이라는 파일은 패스워드들의 해시 정보를 가지고 있으며, 운영체제에 의해서 암호화되어 보호되고 있는데 이를 복구하는 기술이다.

* **Timeline 분석**

파일 시스템들은 각각의 파일들이 만들어진 시간 정보와 마지막으로 접근된 시간 정보 그리고 마지막으로 수정된 시간 정보들을 가지고 있다. 이런 시간 정보를 가지고서 시간의 흐름에 따라 어떤 파일들이 생성되고 접근되었는지를 알기 쉽게 보여주어 증거 분석을 좀 더 수월하게 할 수 있게 하는 기술이다.

* **삭제된 파일 복구**

하나의 파일은 여러 클러스터들의 리스트로 이루어져 있으며, 이런 리스트 정보가 파일 시스템에 들어 있다. 일반적으로 하나의 파일을 삭제할 경우에 파일시스템은 클러스터들에 들어 있는 파일 내용을 지우는 것이 아니라 파일에 할당된 클러스터들을 프리시키는 것으로 파일을 지운다. 따라서 프리된 클러스터들이 다른 파일에 할당되지 않는 한 삭제된 파일을 복구할 가능성이 가짐으로써 사용되는 기술이다.

* **비정상적인 파일 찾기**

사용자가 중요한 데이터를 숨길 경우에 Windows에서 파일을 숨김 속성으로 놓거나 파일 확장자를 바꾸어서 데이터를 숨기려 할 수 있다. 따라서 숨김 속성을 가진 파일들이나 파일 확장자가 바뀐 파일들을 따로 찾아내어 분석에 많은 도움이 될 수 있는 기술이다.

* **로그 분석**

어떤 장치나 응용 프로그램을 사용하게 되면, 운영체제나 응용 프로그램이 로그를 남기는 경우가 있으며 이런 로그를 사건 분석에 중요한 정보로서 분석하는 기술이다.

* **슬랙 공간 분석**

파일 시스템은 하나의 큰 파일을 저장할 때 여러 클러스터들로 나누어 저장하게 된다. 이 때 가장 마지막 클러스터에는 파일의 가장 뒷부분을 저장한 다음 남게 되는 공간이 생길 수 있는데 이런 공간을 파일 슬랙 공간(slack space)이라고 한다. 사용자들이 이런 슬랙 공간에 데이터를 숨겨 놓을 수 있기 때문에 이런 슬랙 공간의 데이터를 분석하는 기술이다.

* **스트링 서치**

해시 검색(hashed search)처럼 조사 대상의 검색범위를 축소하기 위해서는 잘 알려진 파일은 검색 대상에서 제외하고, 주목해서 검색할 대상을 선정하여 검색 범위를 축소하고, 조사우선순위를 부여하는 기능을 제공하는 검색 기술이다.

**2.1.3. 증거 제출**

증거 자료의 신뢰성을 확보하기 위해서, 수집된 데이터가 변조 및 손상되지 않았음을 해시 및 오류 검증 알고리즘을 이용하여 증명하는 기술로서 디지털 증거 무결성 확보기술이 필요하다.

**2.2. 클라우드 기반 스마트 홈 IoT 포렌식 연구**

스마트홈 IoT와 IoT가 등록된 스마트 폰은 서비스 제공을 위해 클라우드 서버와 통신을 수행한다. 클라우드 서버와 통신 과정에서 스마트홈 IoT와 스마트폰, 클라우드 서버에는 사용자에 대한 다양한 정보가 저장될 가능성이 있다. 사용자에 대한 다양한 정보가 클라우드 서버로 전송되는 것은 개인정보 문제를 야기할 수 있지만, 포렌식 관점에서는 범죄를 해결하는 데 증거로 사용될 수 있다.

**2.2.1. 스마트 TV 포렌식**

스마트 TV는 네트워크에 연결되어 TV를 시청하는 것뿐만 아니라 인터넷 검색, OTT 스트리밍 서비스를 사용할 수 있다.

**(1) Boztas et al.의 연구**

Boztas et al.은 스마트 TV에서 데이터 추출을 위해 세 가지 방식을 사용하였다. 세 가지 방식은 NAND 플래시 메모리에 시그널 연결, MTK II (NFI MemoryToolkit II) 사용, 애플리케이션 사용이다. 세 가지 방식 중 MTK II를 사용하여 데이터를 획득하였다. MTKII는 메모리 칩에서 사용자 데이터를 추출해주는 도구이다. 해당 도구를 사용하여 스마트 TV의 사진, 연결된 장치, 웹사이트 방문 기록과 같은 데이터를 획득하였고 스마트TV의 칩오프 가능성을 보였다.

**(2) Nemayire et al.의 연구**

Nemayire et al.은 스마트 TV에서 포렌식을 위해 애플리케이션 레벨, 네트워크 레벨, 디바이스 레벨로 나누어 포렌식을 수행하였다. 이 중 디바이스레벨에서는 칩오프를 수행하여 데이터를 획득할 수 있었다. 칩오프를 수행하여 NAND 플래시 메모리 칩을 분리할 수 있었다. 본 연구에서는 데이터 분석을 위해 한컴GMD에서 개발된 MD-Reader와 MD-NEXT를 사용하여 데이터를 분석하였다. 해당 NAND 플래시 메모리는 VDFS(Vertically Deliberate improved performance FileSystem) 파일 시스템을 사용하였고, 파일 시스템 분석을 통해 이미지, 음성, 비디오 파일과 같은 미디어파일, 브라우저 히스토리, 구글 검색기록을 획득하였다.

**2.2.2. AI 스피커 포렌식**

AI 스피커는 가정에서 많이 사용되는 홈IoT 중 하나로 음성 명령을 통해 날씨 알림, 일정 추가와 같은 기능뿐만 아니라 음성 명령으로 타 IoT를 제어할 수 있다.

**(1) Boztas et al.의 연구**

Boztas et al.은 스마트 TV에서 데이터 추출을 위해 세 가지 방식을 사용하였다. 세 가지 방식은 NAND 플래시 메모리에 시그널 연결, MTK II (NFI Memory Toolkit II) 사용, 애플리케이션 사용이다. 세 가지 방식 중 MTK II를 사용하여 데이터를 획득하였다. MTK II는 메모리 칩에서 사용자 데이터를 추출해주는 도구이다. 해당 도구를 사용하여 스마트 TV의 사진, 연결된 장치, 웹사이트 방문 기록과 같은 데이터를 획득하였고 스마트 TV의 칩 오프가능성을 보였다.

**(2) Shin et al.의 연구**

Shin et al.은 AI 스피커와 클라우드 서버 사이의 암호화된 트래픽을 분석하기 위해 인증서를 AI 스피커에 주입하는 다섯 가지 방법을 제안했다. 다섯 가지 방법 중 AI 스피커 앱 취약점 분석과 플래시 메모리 재작업을 통해 인증서를 주입할 수 있었다. AI 스피커에 인증서를 주입하여 암호화된 트래픽을 분석할 수 있었고 사용자의 스케줄, 메모, 음성 명령어 등을 획득하였다. 해당 방법은 AI 스피커뿐만 아니라 칩오프가 가능한 타 스마트 홈 IoT 기기에도 적용할 수 있을 것으로 추정된다.

**3. 프로젝트 내용**

**3.1. 시나리오**

**3.1.1. Live Forensic**

Live Forensic은 실시간으로 IoT 기기에서 디지털 증거를 수집, 분석하고 해석하는 과정을 말한다. 우리가 개발할 Live Forensic도구의 시나리오는 다음과 같다.

1. IoT기기에 우리가 개발한 포렌식 소프트웨어를 설치한다.
2. 소프트웨어는 선택한 디바이스 또는 시스템에서 데이터를 실시간으로 수집한다.
3. 네트워크 트래픽을 캡처하거나, 메모리 덤프를 수행하거나, 파일 시스템을 스캔하거나, 로그 파일을 수집한다.
4. 소프트웨어는 분석 중에 악성 활동, 해킹 시도, 비정상적인 동작 등을 탐지한다.
5. 탐지된 활동에 대한 증거를 확보하여 수사기관에 전송하고, 필요한 경우 보호 조치를 취한다.

**3.1.2. Offline Forensic**

Offline Forensic은 IoT 기기나 기기내의 저장매체를 가져와 디지털 증거를 수집, 분석하고 해석하는 과정을 말한다. 우리가 개발할 Offline Forensic도구의 시나리오는 다음과 같다.

1. 포렌식 대상 디바이스 또는 저장매체를 연결하거나 로딩한다.
2. 대상 디바이스 또는 저장매체의 이미지를 생성한다.
3. 저장매체 이미지를 기반으로 소프트웨어는 삭제된 파일, 파티션, 디렉터리 등을 복구하거나 추출한다.
4. 저장매체 이미지와 함께 시스템 메모리 덤프를 수행할 수 있다. 메모리 덤프에서 프로세스, 스레드, 열린 포트, 레지스트리 항목 등을 분석하여 악성 활동 또는 이상 징후를 확인할 수 있다.

**3.2. 고려사항**

**3.2.1. 디렉토리별 분석 고려사항**

\* Linux Forensic 핵심 대상 디렉토리

|  |  |
| --- | --- |
| / (최상위 경로) | \* 최상위 디렉토리 |
| /home | \* 사용자 홈 디렉토리  - 계정별로 관련된 모든 정보를 저장 |
| /dev | \* 장치 관련 디렉토리  - 일반적으로 실행 파일이 존재하지 않는 경로이므로, 실행 파일이 있을 시, 주의 깊게 확인이 필요 |
| /var | \* 대부분의 로그가 저장되는 경로 |
| /etc | \* 전반적인 시스템 설정 정보를 저장  - 악성 설정 정보에 대한 추적에 용이 |
| /usr | \* Default로 지정되는 Application 설치 경로 |
| /bin | \* 명령어가 저장되는 경로  - 보통 실행 파일을 덮어쓰는 등의 방식으로 악성 행위 수행 |
| /proc | \* Process 정보를 저장하는 경로 |

**3.2.2. 로그 파일별 분석 고려사항**

리눅스 시스템 로그 파일

- 리눅스의 로그들은 주로 /var/log 디렉토리에 저장하게 된다.

1. /var/log/massages

- 시스템에 문제가 생겼을때 가장 먼저 확인해보는 로그 파일.

- syslog.conf 설정파일에 로그를 남기지 않도록 지정된 내용을 제외한 모든 항목들이 기록

- 많은 항목들이 기록되기 때문에 일반적으로 grep 명령어를 통해서 확인

2. /var/log/secure

- 시스템에 접속한 사용자에 대한 기록이 남는 파일.

- .ssh 접속이나 텔넷 접속이 발생했을 경우 접속한 시간, 사용자명, 클라이언트 IP 등이 기록

- 시스템 해킹이 의심될 경우 이 파일을 열어서 접속 기록을 확인

3. /var/log/maillog

- 전송되거나 수신된 메일에 대한 내용들이 기록으로 남는다.

4. /var/log/cron

- crontab으로 설정한 작업들이 정상 수행되었는지 확인할 수 있는 로그

5. /var/log/dmesg

- 시스템 부팅시 로그가 기록된다 명령어를 . dmesg 통해 출력되는 내용이 이 파일에 기록된다.

6. /var/log/wtmp

- 최근 접속 사항이 기록되는 파일이다 명령을 . last 통해 출력되는 내용이 기록된다.

7. /var/log/lastlog

- 각 사용자의 마지막 로그인 내용이 기록되어 있다 명령어를 . lastlog 통해 관련 내용 확인 가능하다

8. xferlog

- FTP 데몬으로 파일을 전송한 내용이 기록된다.

9. /var/log/httpd/access\_log

- Apache 서비스 데몬의 로그 파일이다 웹. 서비스를 운영할 경우 해당 내용이 기록된다.

10. /var/log/httpd/error\_log

- Apache 서비스 데몬의 에러와 관련된 내용이 기록된다

**3.2.3. 휘발성 데이터 수집 고려사항**

|  |  |
| --- | --- |
| 분류 | 명령어/경로 및 내용 |
| 호스트 식별 | \* hostname - 침해 사고 조사 시(특히 다수 대상), 대상 식별을 위해 사용 |
| 시간 정보 | \* date - 현재 시스템에 대한 시간 정보 확인  \* date + %s - Unix Time으로 변경하여 시간 정보 출력 |
| 시스템 Timezone | \* cat /etc/timezone - 시스템 Timezone 설정 상태 확인 (date 명령어로도 확인 가능) |
| 부팅 기준 시간 | \* uptime - 시스템 부팅 후, 작동한 시간 확인 - 현재 로그인한 사용자 수 - 현재 시간 - 특정 시간동안의 시스템 평균 메트릭 등 |
| IP 설정 정보 | \* ip ad 또는 ifconfig - 호스트 식별을 위해 사용하며, eth, ens등의 NIC 설정 정보 확인 가능 - 네트워크 패킷 덤프를 위해 확인이 필수적임  \* 명령어 출력 정보 분류 - lo, ens33 = NIC - UNKNOWN = NIC가 작동하나, 연결이 없는 상태 - UP = NIC 작동 및 연결이 있는 상태 |
| 무차별 수집 | \* ifconfig eth숫자 promisc - 수사관 성향에 따라 사용 여부가 다름 - 보통 네트워크 차단, 전체 수집이 필요한 상황 등에 따라 사용 |
| Network Dump | \* tcpdump -i eth숫자 -vv -w 파일명.pcap - 지정한 eth 대상으로 파일명.pcap이란 패킷 덤프 파일을 기록  \* wireshark 파일명.pcap - tcpdump로 생성된 파일의 네트워크 통신 이력 추적 진행  \* tcpdump -i any - 모든 인터페이스에 대한 네트워크 통신 Dump |
| 소켓 정보 | \* netstat -ss - 활성화 된 소켓 정보에 대한 수집 명령어  \* netstat -rn - 라우팅 테이블 정보 수집  \* netstat -anop - 최적 옵션 설정으로, 출력 결과 전체를 볼 필요는 없으며, 명령어 상단에 tcp, udp. raw 등의 프로토콜 정보를 한정하여 선별 확인을 권장 |
| 포트 정보 | \* nmap -sT 또는 -sU 대상 - netstat으로 수집하지 못한 열린 포트에 대해 추가 정보 수집이 가능  \* lsof -i -P -n - nmap과 동일 목적으로 사용 가능한 명령어 (list open file) |
| 실행중인  파일 정보 | \*lsof -u 유저명 - 유저 이름을 별도로 지정하여 실행중인 파일에 대한 정보를 출력 |
| 마운트 정보 출력 | \* mount 또는 df -TH - 마운트 된 디스크 정보 출력 명령어 |
| 커널 모듈 정보 | \* lsmod - 일종의 라이브러리 정보로 로드 된 커널 모듈을 탐색하는 명령어 - 어떤 모듈이 의심스러운지 찾기 위한 전체 리스팅이 목적  \* modinfo 커널모듈이름 - 분석할 커널 모듈 이름을 입력하여 상세 정보를 출력 - 기본 설치 모듈이 아닐경우 의심해야함 |
| 사용자 이벤트 | \* ausearch -ui 숫자 -i - uuid 기반으로 사용자 이벤트를 검색하며 일종의 감사 목적의 로그 (별도 설치가 필요한 명령어) - 해당 감사 로그를 설정하지 않았다면, /var/log/audit/audit.log 정보(auditd)가 없음  \* auditd 설정 방법 1) auditd 설치 2) cd /etc/audit/plugins.d 3) vi 또는 mousepad를 통해 syslog.conf 파일 열기 4) 파일 내용 중 active 정보를 yes로 변경 후 저장하여 로깅 활성화 (active = yes) 5) service auditd restart  \* ausearch -ui 숫자 -i | grep -i session 및 ausearch -ui 숫자 -i | grep service - auditd가 설정되어 적재되는 로그들이 존재할 시, grep -i를 통해 추적 - 특히 session과 service로 필터링하여 서비스 유형, 시간 정보, uid, service\_start/stop, session\_open/close 등의 정보를 상세히 보길 권장 |
| ELF 파일 정보 | \* readelf -a 파일명 - 악성코드 초동 분석 시, 가장 많이 사용하는 명령어 - 재부팅 여부에 따라서 해당 정보가 달라질 수 있기 때문에 휘발성 정보로 분류  \* .rela.plt section - 해당 섹션에 나온 정보는 elf 파일에서 사용했던 함수의 정보를 확인할 수 있음 - 보통 해당 부분에서 socket 함수(외부 통신)를 가장 많이 찾음 |
| 프로세스 상세 정보 | \* pstree - 프로세스들을 상호 연관된 경우 트리 형태로 출력해주는 명령어 - 프로세스 정보 파악 시, 우선적으로 실행하여 연관 프로세스에 대한 파악 후 추가 확인 진행  \* top - CPU ,Memory 점유 정보를 출력하며, 리소스 과다 점유 프로세스는 채굴기인 경우가 다수  \* ps -aux - 시스템 / 유저 프로세스에 대한 구분을 하며 확인 진행을 권장 - [대괄호]로 묶인 프로세스의 경우 커널의 자체 데몬 프로세스로 굳이 볼 이유는 없음 (유저 프로세스를 중점으로 확인하길 권장) |
| SWAP 영역 정보 | \* cat /proc/swaps - 메모리 스왑 영역에 대한 정보 출력 |
| Disk 파티션 정보 | \* cat /proc/partitions - 디스크 파티션 정보 출력 |
| Kernel 메세지 로그 | \* dmesg - 커널 로그를 출력해주며 연결된 장치 정보 및 오류 정보 등의 중요 정보가 다수 저장 |
| 실행중인 서비스 정보 | \* systemctl --state=active --type=service - Live Forensic 환경에서 실행중인 서비스를 확인할 수 있는 명령어 |
| Memory Dump | \* 해당 Live 시스템의 Memory가 저장한 모든 정보를 파일로 저장하는 것 - volatility 툴을 많이 사용하며, 분석에 Profile 또는 ISF 정보가 필요하기 때문에 과정이 복잡함 |

**3.2.4. 비휘발성 데이터 수집 고려사항**

|  |  |
| --- | --- |
| 분류 | 명령어/경로 및 내용 |
| 시스템 정보 | \* cat /proc/cpuinfo - cpu 정보를 출력하나, cpu의 취약점 발현이 쉽지 않아서, 필수적이진 않음  \* cat /proc/self/mounts - 명령어가 실행 된 마운트 지점에 대한 출력 |
| 커널 정보 | \* cat /etc/os-release - Linux OS 정보가 상세히 담겨 있는 텍스트 파일 내용 출력  \* uname -r - Live 상태로 확인되는 커널 버전 정보  \* cat /proc/version - File 상태로 확인되는 커널 정보  \* cat /etc/hostname - hostname 명령어와 동일한 출력이지만, 공격자가 변조시키는 Case도 존재  \*\* 커널 정보의 비교는, 동일 OS를 구축하여, 동일 경로에 대한 비교로 빠르게 선별할 수 있음 |
| 사용자 계정 정보 | \* cat /etc/passwd (사용자 계정 정보 파일) - 사용자 이름 - shadow password - UID - GID - Comment - Home Directory - 사용자 절대 경로 로그인 쉘 (단, nologin으로 입력되어 있을 시, 사용자가 접근할 수 없는 계정) ex) root:x:0:0:root:/root:/usr/bin/zsh - 해당 정보는 특히, root 권한으로 사용되는 계정 여부 확인이 필수적  \* cat /etc/shadow (사용자 암호 정보 파일) - 계정명 - Hash + Salt를 조합한 Password (첫 시작 부분의 "$문자$" 사이의 문자 값에 따라 어떠한 암호 알고리즘을 사용하는지 식별 가능하며, 해당 정보는 암호를 Crack할 수 있는지 여부를 판단하는데 사용) - john the ripper 툴과 pass 사전 파일을 통해서 복호화를 수행하는 경우 다수 존재  \* cat /etc/group (사용자가 속할 수 있는 Group에 대한 정보 파일) - root 권한을 행사 할 수 있는 그룹에 속한 계정이 어떤 것들인지 식별하는 것이 중요 - root, lxd, wheel(su), sudo 그룹이 대표적으로 권한 상승이 가능한 그룹  \* cat /etc/sudoers (root 권한으로 상승이 가능하도록 설정 사항을 기록한 파일) |
| 현재 로그인 된 사용자, 로그인 History | \* w - wtmp Log 파일에 대한 정보 중 로그인한 사용자가 어떤 방식으로 로그인 했는지 등을 확인 가능 - tty (컴퓨터와 직접 연결된 통신 포트) 및 pts (원격지 터미널 정보 or 가상 터미널 연결) 등 확인 가능  \* last - /var/log/wtmp Binary 로그 파일 내용을 평문으로 출력 - sudo 명령 사용 이력, 인증, 원격 사용자 로그인 등을 확인 가능 (사용자 계정과 시간 정보를 Check 하는것이 핵심)   \* lastb - /var/log/btmp를 참조하여 마지막 로그인 정보 또는 과거에 잘못 접속한 계정의 이력을 확인 가능 |
| 자동 실행, 스케줄링 정보 | \*/home/계정명/.bash\_profile - root를 제외한 다른 계정의 홈 디렉토리에 존재하는 사용자 별 자동 실행 프로그램에 대한 설정  \* /etc/profile - 전체 사용자를 대상으로 적용되는 자동 실행 프로그램 설정 - 실행 권한(+x)이 존재하는 파일 이름을 삽입하여 무한 실행하는 등의 공격 수행 - 해당 File 악용 Case가 상당히 많은편  \* /etc/init.d - 부팅 시, 자동으로 실행되는 Binary 정보  \* /etc/service - 지정된 포트, 프로세스 등을 자동으로 허용하는 파일  \* cron - 스케줄러로 시간, 요일, 조건에 의해 실행 (= Windows 작업 스케줄러) - Linux 버전에 따라 이름이 상이하며 cron도 경로, 종류가 다양함 1) /var/log/cron 2) /etc/cron.d 3) /etc/cron 4) /etc/crontab 5) /var/spool/cron/crontabs - crontab 또는 crontabs 파일 내용을 설정해야 cron 관련 경로에 배치 된 실행 권한이 존재하는 파일이 실행 - 단순히 경로에 파일만 넣는 경우 실행이 되지 않음 / 단, 공격자가 사용 후 crontab 내용 삭제는 가능 |
| Linux Log File | \* /var/log - cron, web, system, application 등의 다수 로그 확인 가능 - 각각의 로그 디렉토리는 access.log (성공), error.log (실패) 정보가 존재 - 특히 error.log를 우선적으로 보며, 실패한 최초 시간을 확인하여 선별 추적 진행이 가능  \* /var/log/secure(구 버전) or /var/log/auth.log - 원격에서 접속한 사용자의 로그 확인이 가능 - 악성 서비스로 추측되는 로그가 존재할 시, syslog도 참조하여 해당 서비스가 어떤 명령어를 실행 했는지 확인할 때도 유용  \* /var/run/utmp 또는 /var/adm/utmpx - 현재 로그인한 사용자 상태 정보를 담고 있는 파일 - w, who, finger 명령어로 확인 가능  \* /var/log/wtmp 또는 /var/adm/wtmpx - 성공한 로그인/아웃 정보 및 System Boot/Shutdown 히스토리를 담은 파일 - last 명령어로 확인 가능  \* /var/log/btmp 또는 /var/adm/loginlog - 실패한 로그인 정보를 담은 파일 - lastb 명령어로 확인 가능 (솔라리스는 text 파일)  \* /var/log/lastlog 또는 /var/adm/lastlog - 마지막으로 성공한 로그인 정보를 담은 파일 - lastlog 명령어로 확인 가능 (솔라리스 finger) |
| 숨겨진 파일 및 디렉토리 | \* find /경로명 -name ".\*" 2>/dev/null - 지정 경로에 대해 숨김 파일을 출력하는 명령어 - 단, /(최상위 경로) 기준으로 설정한 find 명령어는 부하가 크고 시간이 오래걸려 권장하지 않음  \* ls -al - 디스크가 Mount 된 경우는 해당 명령어로 숨김 파일 정보의 확인이 수월 - 이미징 확인 툴에서도 ".파일명" 형태로 즉시 숨김 파일 인지가 가능 |
| File 서명 (일종의 File Header) | \* xxd 파일명 - 파일의 처음 10 ~ 20byte에 존재하는 헤더 정보에서 확장자를 비교하여 서명 상태 확인 - hexa 값과 ASCII 값을 구분하여 확인하는 등의 방법이 존재 (비교적 번거로운 과정) |
| File 확장자 | \* file 파일명 또는 strings 파일명 - xxd 명령 대신 file 명령어로 쉽게 확장자 비교가 가능 - 파일이 특정되면 strings 명령어로 문자열을 뽑아내서 추가 분석 진행  \*zip, jpg, png, pdf, xls, doc, exe, elf정도의 Header는 어떤식으로 ASCII, Hexa가 구성되어 있는지 숙지하는게 분석에 매우 도움 |
| 쓰기 가능 File | \* find 경로 -writable -type f 2>/dev/null - 쓰기 권한(+x)이 존재하는 파일 확인 |
| 시간 정보 | \* touch -t 시작시간 start 후, touch -t 종료시간 end - 시작 시간을 지정한 파일명과, 종료 시간을 지정한 파일명을 통해 범위를 지정하여 시간 범위 내에서 수정된 파일의 검색이 가능  \* find 경로 -newer start -a ! -newer end - 지정한 파일보다 더 최근에 생성되거나, 변경된 파일 검색 가능  \* 시간으로 파일명 생성 시, yyyymmddhhmm.ss 값으로 설정이 가능 ex) touch -t 202305011123.00 start 및 touch -t 202305102359.00 end  \* MAC (Modify, Access, Change) - Modify : 파일 내용에 대한 변경 시간 - Access : 파일 접근에 대한 시간 - Change : 파일 속성값의 변경에 대한 시간 (단, Access 시, Change도 동일하게 변경)  \* stat 파일명 - live 상태에서 stat 명령어를 통해 파일 생성 시간(Birth)값 까지 추가적으로 확인 가능 (정적 상태에서 분석 시, Disk 이미지 분석 Tool에서 created time 정보를 토대로 타임라인 구성 가능 |
| 악성 의심 정보 | \* rkhunter --check --rwo - /dev 경로 위주로 탐색을 수행하는 악성 정보 탐색 도구 (별도 설치 필요)  \* chkrootkit - 시스템 Binary 파일을 검사하여, 루트킷이 만든 수정 사항을 감지하는 셸 스크립트 도구 (최근 침해 사고 동향 상, 루트킷 방식으로 숨기기 보다는 대놓고 퍼트려 놓는 방식이 다수) |
| 패키지 설치 정보 | \* dpkg -l - 설치된 모든 패키지의 이름, 버전 정보 등 출력  \* dpkg -L 패키지명 - 해당 패키지가 설치된 모든 경로를 출력 |
| 사용자 홈 디렉토리 숨김 파일 | \* /home/계정명/.local - 해당 디렉토리는 일종의 휴지통 역할로 최근 문서 열람, 파일 및 폴더 열람 정보 등을 확인 가능 - .local/share/recently-used.xbel 파일에 최근 열람한 문서 정보를 확인 가능 (visited, bookmark, application name 항목을 통해 시간, 파일명, 접근에 사용된 앱 이름 등 기록)  \* /home/계정명/.mozilla - 웹 브라우저 전용 로그 디렉토리로 사용자 별 해시값으로 이름을 가진 로그 디렉토리가 존재 (.mozilla/firefox) - places.sqlite : 방문 정보 - cookies.sqlite : Web Cookie 정보 - extension.json : 확장 프로그램 정보  \* /home/계정명/.Thuderbird - 메일 프로그램 관련 정보  \* /home/계정명/.cache - 웹 이용 내역을 남기는 썸네일, Cache 정보  \* /home/계정명/.vim - vim을 이용한 기록 (단, vi만 사용하는 경우 존재하지 않는 경우도 있음)  \* /home/계정명/.\*\_history - 해당 사용자의 명령어 사용 기록 - 정상적으로 터미널 창이 종료된 경우에만 history 파일에 저장 (지울 시 복구가 불가능하며, 시간 정보가 존재하지 않음)  \* /home/계정명/.profile - 해당 사용자 대상으로 자동 실행되는 Program 정보 - 실행 권한(+x)이 존재하는 파일 이름을 넣어 놓는 형태로 무한 실행하는 등의 방식 |

**4. 향후 일정 및 역할 분담**

먼저 공유기를 주 대상으로 연구를 진행할 것이다. 우선적으로 공유기에 OpenWrt와 BusyBox를 설치하여 해당 시스템에 접속해보고, 파일 시스템을 직접 확인함과 동시에 로그 파일이 무엇이 남는지 조사해보며 분석해보려 한다. 공유기에 대한 충분한 분석이 이루어지면, IoT 기기 중 IP 카메라에 대한 연구와 분석을 진행할 예정이다.

이후 분석한 내용을 기반으로 live및 offline포렌식 도구 개발을 위한 구체적인 계획을 수립하고, 포렌식 프로그램 초안을 만들어 볼 예정이다. 추가적으로 여건이 된다면 해당 프로그램에 AI를 적용하는 등 부가 기능을 구현하고자 한다.

우선 포렌식 도구를 개발하기에 앞서 공유기와 IP 카메라를 연구와 분석은 공동으로 진행할 예정이며, 포렌식 도구 개발에 대한 역할은 추후 상황에 따라 분담하려 한다.

**5. 결론 및 기대효과**

본 연구는 리눅스 기반의 OpenWrt 또는 BusyBox로 구성된 IoT 기기에 대한 디지털 포렌식 도구를 개발하고자 하는 데 주요 목표를 두고 진행된다. 이러한 디지털 포렌식 도구의 개발은 다음과 같은 결론을 도출한다.

첫번째로 IoT 기기의 보안 강화이다. IoT 기기는 점점 더 중요한 역할을 하고 있으며, 이에 따라 보안 문제가 증가하고 있다. 우리가 개발한 도구는 IoT 기기의 보안 취약점을 식별하고 보안 강화에 도움을 줄 것으로 기대된다.

두번째로는 수사기관의 디지털 포렌식 능력 향상이다. 디지털 포렌식 전문가와 수사기관은 이 도구를 활용하여 IoT 기기에서 증거 수집 및 분석을 더욱 효과적으로 수행할 수 있을 것이다. 이를 통해 범죄 조사, 기업 보안 사고 대응, 사회 문제 해결에 대한 능력이 향상될 것이다.

세번째로는 사회적 안전망 구축이다. 이 연구를 통해 범죄 예방 및 조사, 개인 정보 보호, 기업 보안, 사회적 문제 해결과 같은 다양한 분야에서 기여할 수 있다. 이 도구의 개발을 통해 이러한 도전 과제를 해결하고 기술적인 능력을 향상시킬 수 있다.

**6. 참고문헌**

[1] 정익래, 홍도원, 정교일, “디지털 포렌식 기술 및 동향", 전자통신동향분석 제 22권 제 1호

[2] 정규식, 김정길, 곽후근, 장훈, “유무선공유기를 이용한 임베디드 리눅스 시스템 구축 및 응용”