【서지사항】

【서류명】 특허출원서

【참조번호】 0055

【출원구분】 특허출원

【출원인】

【명칭】 단국대학교 산학협력단

【특허고객번호】 2-2005-016634-4

【대리인】

【명칭】 특허법인린

【대리인번호】 9-2023-100001-3

【지정된변리사】 장영태,박진철,박진태

【포괄위임등록번호】 2024-018865-7

【발명의 국문명칭】 네트워크 동기화 환경 기반 시간 조작 탐지 장치 및 방법

【발명의 영문명칭】 NETWORK SYNCHRONIZATION ENVIRONMENT BASED TIME MANIPULATION DETECTION DEVICE AND METHOD

【발명자】

【성명】 조성제

【성명의 영문표기】 CHO, Seong Je

【국적】 KR

【주민등록번호】 660209-1XXXXXX

【우편번호】 16832

【주소】 경기도 용인시 수지구 풍덕천로 161, 105동 302호 (풍덕천동, 수지마을동보아파트)

【거주국】 KR

【발명자】

【성명】 정지헌

【성명의 영문표기】 JUNG, Jee Heun

【국적】 KR

【주민등록번호】 980228-1XXXXXX

【우편번호】 06256

【주소】 서울특별시 강남구 논현로65길 38, 201호 (역삼동, 원형빌라)

【거주국】 KR

【발명자】

【성명】 조민혁

【성명의 영문표기】 CHO, Min Hyuk

【국적】 KR

【주민등록번호】 000714-3XXXXXX

【우편번호】 12955

【주소】 경기도 하남시 대청로116번길 59, 408동 503호 (창우동, 꿈동산신안아파트)

【거주국】 KR

【출원언어】 국어

【심사청구】 청구

【공지예외적용대상증명서류의 내용】

【공개형태】 학회(2024 한국정보과학회 정보보안 및 고신뢰컴퓨팅 하계 워크샵) 논문 발표

【공개일자】 2024.08.19

【공지예외적용대상증명서류의 내용】

【공개형태】 논문지(한국차세대컴퓨팅학회 논문지) 게재

【공개일자】 2024.08.23

【공지예외적용대상증명서류의 내용】

【공개형태】 학회(2024 한국차세대컴퓨팅학회 춘계학술대회) 논문 발표

【공개일자】 2024.04.25

【이 발명을 지원한 국가연구개발사업】

【과제고유번호】 2710008520

【과제번호】 II221022

【부처명】 과학기술정보통신부

【과제관리(전문)기관명】 정보통신기획평가원

【연구사업명】 정보보호핵심원천기술개발(R&D)

【연구과제명】 이벤트 기반 실험시스템 구축을 통한 자동차 내·외부 아티팩트 수집 및 통합 분석 기술 개발

【과제수행기관명】 단국대학교 산학협력단

【연구기간】 2024.01.01 ~ 2024.12.31

【이 발명을 지원한 국가연구개발사업】

【과제고유번호】 2710017608

【과제번호】 2021R1A2C2012574

【부처명】 과학기술정보통신부

【과제관리(전문)기관명】 한국연구재단

【연구사업명】 개인기초연구(과기정통부)

【연구과제명】 모바일 플랫폼 기반 차량 포렌식을 위한 효과적/지능적 프레임워크

【과제수행기관명】 단국대학교

【연구기간】 2024.03.01 ~ 2025.02.28

【취지】 위와 같이 특허청장에게 제출합니다.

대리인 특허법인린 (서명 또는 인)

【수수료】

【출원료】 0 면 46,000 원

【가산출원료】 46 면 0 원

【우선권주장료】 0 건 0 원

【심사청구료】 22 항 1,288,000 원

【합계】 1,334,000원

【감면사유】 전담조직(50%감면)[1]

【감면후 수수료】 667,000 원

【첨부서류】 1.공지예외적용대상(신규성상실의예외, 출원시의특례)규정을 적용받기 위한 증명서류[학회(2024 한국정보과학회 정보보안 및 고신뢰컴퓨팅 하계워크샵) 논문 발표

2024.08.19]_1통 2.공지예외적용대상(신규성상실의예외, 출원시의특례)규정을 적용받기 위한 증명서류[논문지(한국차세대컴퓨팅학회 논문지) 게재 2024.08.23]_1통 3.공지예외적용대상(신규성상실의예외, 출원시의특례)규정을 적용받기 위한 증명서류[학회(2024 한국차세대컴퓨팅학회 춘계학술대회) 논문 발표 2024.04.25]_1통

1 : 공지예외적용대상(신규성상실의예외,_출원시의특례)규정을_적용받기_위한_증명서류
PDF 파일 첨부

2 : 공지예외적용대상(신규성상실의예외,_출원시의특례)규정을_적용받기_위한_증명서류
PDF 파일 첨부

3 : 공지예외적용대상(신규성상실의예외,_출원시의특례)규정을_적용받기_위한_증명서류
PDF 파일 첨부

【발명의 설명】

【발명의 명칭】

네트워크 동기화 환경 기반 시간 조작 탐지 장치 및 방법{NETWORK SYNCHRONIZATION ENVIRONMENT BASED TIME MANIPULATION DETECTION DEVICE AND METHOD}

【기술분야】

【0001】 개시되는 실시예들은 네트워크 동기화 환경 기반 시간 조작 탐지 장치 및 방법과 관련된다.

【발명의 배경이 되는 기술】

【0002】 차량 인포테인먼트 시스템(vehicle infotainment system)은 운전과 길안내 등을 의미하는 인포메이션(information)과 다양한 오락거리와 인간 친화적인 기능을 의미하는 엔터테인먼트(entertainment)의 통합 시스템을 의미할 수 있다. 상기 차량 인포테인먼트 시스템은 블루투스, Wi-Fi, USB 등의 통신을 통해 차량 내 존재하는 다양한 기기(예를 들어, 스마트폰 등)와 연결되어, 다양한 서비스를 제공할 수 있다.

【0003】 이러한 통신을 통해 연결된 기기들은 조작자들에 의해서 악의적인 의도로 시간을 비롯한 다양한 정보들이 조작인 안티 포렌식(Anti-Forensics) 행위가 발생될 수 있다. 상기 안티 포렌식은 데이터를 위조, 변조, 은닉, 암호 사용, 파괴, 타임 스탬프 조작 등을 통해 디지털 증거의 존재, 양 및 품질을 저하시켜 포

렌식 수사를 어렵게 하거나 불가능하게 만드는 행위를 의미할 수 있다.

【0004】 이에, 관련 기술 운용자는 네트워크 동기화 환경에서 발생하는 타임 스탬프의 조작을 비롯한 각종 안티 포렌식 행위를 발견하기 위한 기술을 연구하고 있다.

【선행기술문헌】

【특허문헌】

【0006】 **(특허문헌 0001)** 대한민국 등록특허공보 제10-1665199호 (2016. 10. 05.)

【발명의 내용】

【해결하고자 하는 과제】

【0007】 개시된 실시예들은 네트워크 동기화 환경에서 다양한 운용체제 시스템의 시간 조작을 포함하는 안티 포렌식 행위를 효율적으로 탐지하기 위한 네트워크 동기화 환경 기반 시간 조작 탐지 장치 및 방법을 제공하고자 한다.

【과제의 해결 수단】

【0009】 일 실시예에 따른 시간 조작 탐지 장치는, 네트워크 동기화 환경에서 분석 대상으로부터 시스템 로그 및 근거리 무선통신 로그 중 적어도 하나 이상

을 포함하는 로그 데이터를 수집하는 데이터 수집부; 및 사전 설정된 제1 분석기준에 따라 상기 근거리 무선통신 로그 및 상기 시스템 로그 중 적어도 하나 이상을 분석하여 시간 조작 의심 흔적을 발견하고, 사전 설정된 제2 분석기준에 따라 상기 시스템 로그를 분석하여 시간 조작 탐지를 수행하는 로그 분석부를 포함한다.

【0010】 상기 분석 대상은, 차량 단말 및 사용자 단말 중 적어도 하나 이상을 포함하고, 상기 사용자 단말은, 이동통신 단말 및 유선 단말 중 적어도 하나 이상을 포함할 수 있다.

【0011】 상기 로그 분석부는, 상기 시스템 로그에서 사전 설정된 시간 조작 의심 로그 메시지를 발견하면 상기 시간 조작 의심 흔적으로 판단할 수 있다.

【0012】 상기 로그 분석부는, 상기 시스템 로그에서 네트워크 동기화 오프(off) 상태를 포함하는 시간 조작 의심 로그 메시지를 검출하면 상기 시간 조작 의심 흔적으로 판단할 수 있다.

【0013】 상기 로그 분석부는, 상기 시간 조작 의심 흔적을 발견한 후, 상기 시스템 로그에서 사전 설정된 시간 조작 로그 메시지를 통해 조작된 시간을 파악할 수 있다.

【0014】 상기 근거리 무선통신 로그는 제1-1 로그 및 제1-2 로그를 포함하고, 상기 로그 분석부는, 시간의 순방향 기준 이벤트 발생 시나리오의 상기 제1-1 로그 또는 상기 제1-2 로그에서, 시간의 역방향 기준 날짜 또는 시간으로 변경된 타임 스탬프(time stamp)를 발견하는 경우, 시간 조작 의심을 인지할 수

있다.

【0015】 상기 로그 분석부는, 상기 제2-2 로그 중 특정 로그 메시지로부터 사전 설정된 시간 조작 로그 메시지를 검출하여 시간 조작 탐지를 수행할 수 있다.

【0016】 상기 로그 분석부는, 상기 제2-1 로그로부터 시간 조작 시 생성되는 사전 설정된 시간 조작 파일을 검출하고, 상기 시간 조작 파일을 분석하여 시간 조작을 탐지할 수 있다.

【0017】 상기 로그 분석부는, 시간의 순방향 기준 로그 메시지 사이에 사전 설정된 시간 조작 의심 메시지가 발견되는 경우, 시간 조작 의심을 인지할 수 있다.

【0018】 상기 로그 분석부는, 사전 설정된 제3 분석기준에 따라 상기 시스템 로그 분석을 통해 시간 조작 전 실제 시간을 파악할 수 있다.

【0019】 상기 로그 분석부는, 상기 제2-1 로그 중 특정 로그 파일로부터 사전 설정된 그리니치 표준시(GMT)를 포함하는 로그 메시지를 확인하고, 확인된 상기 그리니치 표준시에 지역별 시간차를 반영하여 상기 실제 시간을 파악할 수 있다.

【0020】 다른 실시예에 따른 시간 조작 탐지 방법은, 시간 조작 탐지 장치에 의해 수행되는 방법에 있어서, 상기 시간 조작 탐지 장치가 분석 대상으로부터 시스템 로그 및 근거리 무선통신 로그 중 적어도 하나 이상을 포함하는 로그 데이터를 수집하는 단계; 사전 설정된 제1 분석기준에 따라 상기 근거리 무선통신 로그 및 상기 시스템 로그 중 적어도 하나 이상을 분석하여 시간 조작 의심 흔적을 발견

하는 단계; 및 사전 설정된 제2 분석기준에 따라 상기 시스템 로그를 분석하여 시간 조작 탐지를 수행하는 단계를 포함한다.

【0021】 상기 분석 대상은, 차량 단말 및 사용자 단말 중 적어도 하나 이상을 포함하고, 상기 사용자 단말은, 이동통신 단말 및 유선 단말 중 적어도 하나 이상을 포함할 수 있다.

【0022】 상기 시간 조작 의심 흔적을 발견하는 단계에서, 상기 시스템 로그에서 사전 설정된 시간 조작 의심 로그 메시지를 발견하면 상기 시간 조작 의심 흔적으로 판단할 수 있다.

【0023】 상기 시간 조작 탐지 방법은, 상기 시간 조작 의심 흔적을 발견하는 단계에서, 상기 시스템 로그에서 네트워크 동기화 오프(off) 상태를 포함하는 시간 조작 의심 로그 메시지를 검출하면 상기 시간 조작 의심 흔적으로 판단할 수 있다.

【0024】 상기 시간 조작 탐지 방법은, 상기 시간 조작 탐지를 수행하는 단계에서, 상기 시간 조작 의심 흔적을 발견한 후, 상기 시스템 로그에서 사전 설정된 시간 조작 로그 메시지를 통해 조작된 시간을 파악할 수 있다.

【0025】 상기 근거리 무선통신 로그는 제1-1 로그 및 제1-2 로그를 포함하고, 상기 시간 조작 의심 흔적을 발견하는 단계에서, 시간의 순방향 기준 이벤트 발생 시나리오의 상기 제1-1 로그 또는 상기 제1-2 로그에서, 시간의 역방향 기준 날짜 또는 시간으로 변경된 타임 스탬프(time stamp)를 발견하는 경우, 시간 조작 의심을 인지할 수 있다.

【0026】상기 시간 조작 탐지 방법은, 상기 시간 조작 탐지를 수행하는 단계에서, 상기 제2-2 로그 중 특정 로그 메시지로부터 사전 설정된 시간 조작 로그 메시지를 검출하여 시간 조작 탐지를 수행할 수 있다.

【0027】상기 시간 조작 탐지 방법은, 상기 시간 조작 탐지를 수행하는 단계에서, 상기 제2-1 로그로부터 시간 조작 시 생성되는 사전 설정된 시간 조작 파일을 검출하고, 상기 시간 조작 파일을 분석하여 시간 조작을 탐지할 수 있다.

【0028】상기 시간 조작 탐지 방법은, 상기 시간 조작 의심 흔적을 발견하는 단계에서, 시간의 순방향 기준 로그 메시지 사이에 사전 설정된 시간 조작 의심 메시지가 발견되는 경우, 시간 조작 의심을 인지할 수 있다.

【0029】상기 시간 조작 탐지 방법은, 사전 설정된 제3 분석기준에 따라 상기 시스템 로그 분석을 통해 시간 조작 전 실제 시간을 파악하는 단계를 더 포함할 수 있다.

【0030】상기 시간 조작 탐지 방법은, 상기 제2-1 로그 중 특정 로그 파일로부터 사전 설정된 그리니치 표준시(GMT)를 포함하는 로그 메시지를 확인하고, 확인된 상기 그리니치 표준시에 지역별 시간차를 반영하여 상기 실제 시간을 파악할 수 있다.

【0031】이 외에도, 개시되는 실시예를 구현하기 위한 방법을 실행하기 위한 컴퓨터 판독가능 저장매체에 저장된 컴퓨터 프로그램이 더 제공될 수 있다.

【0032】이 외에도, 개시되는 실시예를 구현하기 위한 방법을 실행하기 위한 컴퓨터 프로그램을 기록하는 컴퓨터 판독 가능한 기록 매체가 더 제공될 수 있다.

【발명의 효과】

【0034】개시되는 실시예들에 따르면, 근거리 무선통신을 통해 연결된 사용자 단말과 차량 단말 간의 타임 스탬프(time stamp) 조작을 비롯한 안티 포렌식 행위를 효율적으로 탐지할 수 있다.

【0035】또한, 개시되는 실시예들에 따르면, 네트워크 동기화 환경에서 안드로이드 및 리눅스 등의 다양한 운용체제 시스템의 시간 조작을 탐지할 수 있다.

【0036】또한, 개시되는 실시예들에 따르면, 차량 단말과 이동통신 단말 간에 근거리 무선통신을 통해 연결된 환경에서, 범죄자를 비롯한 조작자가 다양한 방식으로 시간을 조작하거나 시스템을 재부팅 하더라도, 이를 효과적으로 인지 및 탐지하고 조작 전 정확한 실제 시간을 파악할 수 있다.

【0037】또한, 개시되는 실시예들에 따르면, 안티 포렌식 행위 수사 과정에서 시간 조작을 탐지하고, 이를 통해 안티 포렌식 행위에 대한 디지털 포렌식 수사 효율성을 극대화 하여 수사 시간을 상대적으로 크게 단축할 수 있다.

【0038】또한, 개시되는 실시예들에 따르면, 해시값을 확인하여 증거의 무결성을 보장함으로써, 파악된 시간 조작 관련 정보가 법적 증거의 가치로 활용될 수 있다.

【도면의 간단한 설명】

【0040】 도 1은 일 실시예에 따른 시간 조작 탐지 장치를 설명하기 위한 블록도

도 2 내지 도 6은 일 실시예에 따른 시간 조작 탐지 방법을 설명하기 위한 실시예

도 7은 일 실시예에 따른 시간 조작 탐지 방법을 설명하기 위한 흐름도

도 8은 일 실시예에 따른 컴퓨팅 장치를 포함하는 컴퓨팅 환경을 예시하여 설명하기 위한 블록도

【발명을 실시하기 위한 구체적인 내용】

【0041】 이하, 도면을 참조하여 본 발명의 구체적인 실시형태를 설명하기로 한다. 이하의 상세한 설명은 본 명세서에서 기술된 방법, 장치 및/또는 시스템에 대한 포괄적인 이해를 돕기 위해 제공된다. 그러나 이는 예시에 불과하며 본 발명은 이에 제한되지 않는다.

【0042】 본 발명의 실시예들을 설명함에 있어서, 본 발명과 관련된 공지기술에 대한 구체적인 설명이 본 발명의 요지를 불필요하게 흐릴 수 있다고 판단되는 경우에는 그 상세한 설명을 생략하기로 한다. 그리고, 후술되는 용어들은 본 발명에서의 기능을 고려하여 정의된 용어들로서 이는 사용자, 운용자의 의도 또는 관례 등에 따라 달라질 수 있다. 그러므로 그 정의는 본 명세서 전반에 걸친 내용을 토

대로 내려져야 할 것이다. 상세한 설명에서 사용되는 용어는 단지 본 발명의 실시예들을 기술하기 위한 것이며, 결코 제한적이어서는 안 된다. 명확하게 달리 사용되지 않는 한, 단수 형태의 표현은 복수 형태의 의미를 포함한다. 본 설명에서, "포함" 또는 "구비"와 같은 표현은 어떤 특성들, 숫자들, 단계들, 동작들, 요소들, 이들의 일부 또는 조합을 가리키기 위한 것이며, 기술된 것 이외에 하나 또는 그 이상의 다른 특성, 숫자, 단계, 동작, 요소, 이들의 일부 또는 조합의 존재 또는 가능성을 배제하도록 해석되어서는 안 된다.

【0043】 도 1은 일 실시예에 따른 시간 조작 탐지 장치를 설명하기 위한 블록도이다.

【0044】 이하에서는, 일 실시예에 따른 시간 조작 탐지 방법을 설명하기 위한 실시예인 도 2 내지 도 6을 참조하여 설명하기로 한다.

【0045】 도 1을 참고하면, 시간 조작 탐지 장치(100)는 데이터 수집부(110), 로그 분석부(120) 및 평가부(130)를 포함한다. 도 1에 도시된 구성요소들은 본 개시에 따른 시간 조작 탐지 장치(100)를 구현하는데 있어서 필수적인 것은 아니어서, 본 명세서 상에서 설명되는 시간 조자 탐지 장치(100)는 위에서 열거된 구성요소들 보다 많거나, 또는 적은 구성요소들을 가질 수 있다.

【0046】 도 1에 도시된 구성요소들은 통신 네트워크(미도시)를 통해 서로 간에 통신 가능하게 연결될 수 있다. 몇몇 실시예들에서, 통신 네트워크는 인터넷, 하나 이상의 로컬 영역 네트워크(local area networks), 광역 네트워크(wire area networks), 셀룰러 네트워크, 모바일 네트워크, 그 밖에 다른 종류의 네트워크들,

또는 이러한 네트워크들의 조합을 포함할 수 있다.

【0047】 데이터 수집부(110)는 네트워크 동기화 환경에서 분석 대상으로부터 시스템 로그 및 근거리 무선통신 로그 중 적어도 하나 이상을 포함하는 로그 데이터를 수집할 수 있다.

【0048】 상기 분석 대상은 차량 단말 및 사용자 단말 중 적어도 하나 이상을 포함할 수 있다. 상기 사용자 단말은 이동통신 단말 및 유선 단말 중 적어도 하나 이상을 포함할 수 있다.

【0049】 상기 차량 단말은 차량 내 구비되는 차량 인포테인먼트 시스템(vehicle infotainment system)을 구현하는 단말을 의미하는 것으로, 내비게이션, 오디오 및 비디오 기반의 다양한 서비스를 차량 운전자 및 동승자에게 제공할 수 있다.

【0050】 상기 사용자 단말은 스마트폰, 휴대폰 및 태블릿 PC 등의 이동통신 단말 및 데스크톱과 같은 PC 등의 유선 단말을 모두 포함하는 의미일 수 있다.

【0051】 상술한 사용자 단말 및 차량 단말은 안드로이드 시스템 또는 리눅스 시스템 등 다양한 운영체제를 설치할 수 있는 단말을 포함할 수 있다. 개시되는 실시예에서는 안드로이드 시스템 및 리눅스 시스템을 예로 들어 설명하기로 한다.

【0052】 상기 근거리 무선통신 로그는 블루투스 로그를 의미할 수 있으며, 이에 한정되지 않는다. 구체적으로, 근거리 무선통신 로그는 근거리 무선통신(예를 들어, 블루투스 통신, 와이파이 통신 등)을 통해 연결된 차량 단말과 사용자 단말

간에 근거리 무선통신과 관련된 다양한 이벤트와 상태 변화를 기록한 로그 파일을 의미할 수 있다.

【0053】 상술한 시스템 로그는 안드로이드(Android) 기반 시스템 로그 및 리눅스(Linux) 기반 시스템 로그 중 적어도 어느 하나 이상을 포함할 수 있다. 구체적으로, 시스템 로그는 운영체제와 애플리케이션의 동작을 기록한 파일로, 시스템의 상태를 모니터링하고 문제를 디버깅하는데 중요한 정보를 제공할 수 있다. 시스템 로그는 다양한 이벤트, 오류, 경고, 정보 메시지를 포함하여 해당 단말의 전반적인 활동을 기록할 수 있다.

【0054】 일 예로, 사용자 단말이 안드로이드 시스템 기반 이동통신 단말인 경우, 데이터 수집부(110)는 ADB(Android Debug Bridge) 환경에서 추출된 로그 데이터를 수집할 수 있다. 이를 위해, 사용자 단말은 개발자 옵션에서 USB 디버깅 옵션을 활성화한 후, ADB에서 bugreport 명령어를 사용하여 로그를 추출할 수 있다.

【0055】 이때, 데이터 수집부(110)는 비휘발성 로그 수집용 명령어(예를 들어, bugreport)를 이용하여 비휘발성 로그 데이터를 수집할 수 있다. 이때, 비휘발성 로그 수집용 명령어는 비휘발성 로그 데이터 뿐만 아니라 휘발성 로그 수집 명령어(예를 들어, logcat)를 통해 수집되는 휘발성 로그 데이터를 포함한 시스템의 전체적인 상위 로그 데이터까지 수집할 수 있는 명령어를 의미할 수 있다. 상기 휘발성 로그 수집 명령어는 시스템이 재부팅 된 경우, 시스템 재부팅 시점 이전의 로그 데이터들을 확인할 수 없는 명령어이다.

【0056】 안드로이드 시스템 기반 차량 단말의 경우, 데이터 수집부(110)는

차량 단말의 숨겨진 기능(hidden feature)인 엔지니어링 모드(engineering mode) 또는 딜러 모드(dealer mode)에 진입한 후, Copy image to USB 기능(예를 들어, "Log Copy to USB"버튼 또는"Copy Logs to USB"버튼) 선택을 통해 로그 데이터를 추출할 수 있다. 상기 로그 데이터는 로그라고도 명명할 수 있다.

【0057】 다른 예로, 데이터 수집부(110)는 리눅스 시스템 기반의 사용자 단말로부터 로그 데이터(예를 들어, 시스템 로그)를 수집할 수 있다.

【0058】 예를 들어, 리눅스의 경우, 유선 단말(예를 들어, PC) 형태인 사용자 단말은 /var/log 하위 디렉터리에 다양한 로그들이 생성될 수 있다. 이때, 데이터 수집부(110)는 사용자 단말로부터 시스템에 직접 적인 이벤트 관련 로그가 저장된 /syslog를 수집할 수 있다. 특히, 데이터 수집부(110)는 네트워크 시간 동기화와 관련 있는 NTP(Network Time Protocol) 관련 로그 데이터를 수집할 수 있다.

【0059】 데이터 수집부(110)는 수집되는 로그 데이터에 대해 SHA-512와 같은 사전 설정된 해시 알고리즘을 이용하여 해시값을 생성하거나, 또는 수집 과정을 동영상 녹화 할 수 있다. 이때 해당 로그 데이터가 어떤 차량 또는 단말로부터 수집한 것인지에 대한 정보도 남긴다. 차량의 경우에는 차대번호(VIN: Vehicle Identification Number)를, 스마트폰의 경우에는 IMEI (International Mobile Equipment Identity), 가능하다면 차량과 스마트폰의 소유주 등의 정보도 함께 식별하여 관리한다.

【0060】 데이터 수집부(110)는 수집된 원본 로그 데이터를 복사하여, 복사본 로그 데이터를 기초로 이후 수행되는 분석 절차가 이루어질 수 있도록 할 수 있다. 이때, 데이터 수집부(110)는 원본 로그 데이터를 상대적으로 안전한 메모리에 보관하고, 해당 원본 로그 데이터의 어떤 데이터를 누가 언제 왜 어떻게 접근했는지에 대한 기록도 함께 매칭하여 저장 및 관리할 수 있다.

【0061】 개시되는 실시예의 로그 분석부(120)는 시간 조작 의심, 시간 조작 탐지 및 실제 시간 파악을 수행할 수 있으며, 이하에서 상세 설명하기로 한다. 이때, 시간 조작은 타임 스탬프(time stamp)를 조작했다는 것을 포함할 수 있으나, 이에 한정되지 않고, 시간(날짜 포함)과 관련된 각종 데이터를 조작한 모든 경우를 포함할 수 있다. 상기 실제 시간은 조작 시점에서의 실제 현재 시간을 의미할 수 있다. 상기 시간 조작 의심은 시간 조작이 발생했다는 것을 인지하는 것을 의미할 수 있다. 상기 시간 조작 탐지는 시간 조작이 발생했음을 입증할 수 있는 증거를 파악하는 것을 의미할 수 있다. 상기 증거는 포렌식 수사에서 시간 조작 증거로 활용될 수 있다.

【0062】 로그 분석부(120)는 사전 설정된 제1 분석기준에 따라 근거리 무선통신 로그 및 시스템 로그 중 적어도 하나 이상을 분석하여 시간 조작 의심 흔적을 발견할 수 있다.

【0063】 상기 사전 설정된 제1 분석기준은 근거리 무선통신 로그 또는 시스템 로그를 분석하여 시간 조작 의심 흔적을 발견하기 위한 기준으로, 시스템 환경(예를 들어, 안드로이드, 리눅스 등) 또는 분석 대상(예를 들어, 차량 단말, 사용

자 단말)에 따라 차별화 되게 설정된 기준을 포함할 수 있으며, 이는 운용자에 의해서 사전 설정될 수 있다.

【0064】일 예로, 로그 분석부(120)는 시스템 로그에서 사전 설정된 시간 조작 의심 로그 메시지를 발견하면 시간 조작 의심 흔적으로 판단할 수 있다.

【0065】리눅스 기반 사용자 단말의 경우, 로그 분석부(120)는 /var/log/syslog 아래에 생성된 로그를 통해 특정 메시지를 담고 있는 로그와 시스템 시간이 변경되는 로그를 기초로 분석할 수 있다.

【0066】예를 들어, 도 2를 참고하면, 로그 분석부(120)는 시스템 로그에서 네트워크 동기화 오프(off) 상태를 포함하는 시간 조작 의심 로그 메시지를 검출하면, 시간 조작 의심 흔적으로 판단할 수 있다. 이때, 시스템 로그는 리눅스 시스템 환경에서 수집된 시스템 로그일 수 있다. 리눅스 시스템 환경에서, 로그 분석부(120)는 Set NTP to disabled(201)와 같은 네트워크 동기화 오프 로그 메시지를 발견하는 경우, 시간 조작 의심을 할 수 있다. 이때, 로그 분석부(120)는 <date 명령어>를 통해 변경된 날짜와 시간 정보를 확인할 수 있다. 유선 단말 형태의 사용자 단말에서 대부분의 로그 메시지는 syslog 파일에 저장될 수 있다. 로그 분석부(120)는 <cat 명령어>를 통해 syslog를 분석한 결과, 2024-07-24 15:18에 NTP(Network Time Protocol)가 비활성화된 메시지를 발견할 수 있다.

【0067】리눅스 시스템 환경에서, Set NTP to disabled 로그 메시지는 서로 다른 단말들 간의 시간 동기화가 오프 되었을 때 발생되는 로그로, 시간 동기화 오프 시점의 시간(예를 들어, 2024-07-24 15:18)을 포함할 수 있다. 상기 동기화 오

프 시점의 시간은 개시되는 실시예에서의 실제 시간을 의미할 수 있다. 즉, 시간 조작 의심 로그 메시지가 실제 시간을 포함하는 경우, 로그 분석부(120)는 시간 조작 의심 로그 메시지에서 실제 시간을 획득할 수 있다.

【0068】 로그 분석부(120)는 시간 조작 의심 흔적을 발견한 후, 시스템 로그에서 사전 설정된 시간 조작 로그 메시지를 통해 조작된 시간을 파악할 수 있다. 상기 조작된 시간은 조작자에 의해서 실제 시간이 실제와 다른 시간으로 조작된 시간을 의미할 수 있다. 또한, 상기 조작된 시간은 조작 시점을 기준으로 과거 시간 또는 미래 시간으로 조작되는 경우를 모두 포함할 수 있다.

【0069】 도 2를 참고하면, 로그 분석부(120)는 리눅스 시스템 환경의 시스템 로그에서 Clock change detected(203) 또는 Changed local time to(205)와 같은 사전 설정된 시간 조작 로그 메시지를 발견하면, 이에 포함된 시간을 조작된 시간으로 파악할 수 있다.

【0070】 예를 들어, 로그 분석부(120)는 Change local time to ~ 라고 생성된 로그 메시지를 분석하여 바뀐 시간대로 로그가 저장되는 것을 확인할 수 있다. 로그 분석 결과, 네트워크 동기화를 비활성화한 시점인 NTP 비활성화 로그 메시지가 생성된 2024-07-24 15:18에 시간을 변경하였다는 것을 알 수 있다.

【0072】 후술하는 다른 예는 상술한 사전 설정된 시간 조작 의심 로그 메시지를 발견하는 일 예를 수행한 이후에 구현되거나, 또는 상술한 일 예와 별도로 구

현될 수 있다.

【0073】 다른 예로, 로그 분석부(120)는 수집한 로그 데이터의 분석을 수행할 수 있다. 로그 분석부(120)는 생성된 로그 파일에서 <수정한 날짜> 태그를 확인하여 해당 날짜에 타임 스탬프로 생성된 로그 파일 위주로 분석할 수 있다. 또한, 로그 분석부(120)는 차량 단말 및 사용자 단말 간에 근거리 무선통신 연결을 통한 통신을 하였기에 근거리 무선통신 로그 역시 분석할 수 있다. 이후, 로그 분석부(120)는 시간 조작 관련 키워드를 이용하여 분석할 수 있다. 예를 들어, 시간 관련 키워드는 TimeChange, TimeSet, DateChange, DateSet, UsetTime 및 UsetDate 등을 포함할 수 있으며, 이에 한정되지 않는다. 이때, 로그 분석부(120)는 시간 조작 관련 로그 메시지가 미 존재하는 경우, 변경된 타임스탬프를 기준으로 사전 설정된 전후 범위로 로그 데이터를 분석할 수 있다. 이에 대한 상세 설명은 후술하기로 한다.

【0074】 상술한 근거리 무선통신 로그는 차량 근거리 무선통신 로그인 제1-1 로그 및 이동통신 단말 근거리 무선통신 로그인 제1-2 로그를 포함할 수 있다. 상기 시스템 로그는 차량 시스템 로그인 제2-1 로그 및 이동통신 단말 시스템 로그인 제2-2 로그를 포함할 수 있다.

【0075】 구체적으로, 일 예로, 로그 분석부(120)는 시간의 순방향 기준 이벤트 발생 시나리오의 제1-1 로그 또는 제1-2 로그에서, 시간의 역방향 기준 날짜 또는 시간으로 변경된 타임 스탬프(time stamp)를 발견하는 경우, 시간 조작 의심을 인지할 수 있다. 상기 시간의 역방향 기준은 상기 시간의 순방향 기준 이벤트 발생

시나리오에서 개시하는 바와 같이 이벤트가 시간에 따라 순차적으로 생성되는 시나리오에서 순방향에 따르지 않는 방향으로 불규칙하게 생성되는 기준을 의미하는 것일 수 있다. 이에, 상기 시간의 역방향 기준은 조작된 시간이 과거 또는 미래인 경우에 대해 모두 검출할 수 있다.

【0076】 예를 들어, 상술한 시간의 순방향 기준 이벤트 발생 시나리오는 표 1과 같을 수 있다. 도시하지 않았지만, 하기 표 1의 각각의 이벤트는 매칭되는 로그 데이터(또는 로그 메시지)를 포함할 수 있다.

【0077】 **【표 1】**

시간	이벤트
17:13	대상 시스템 블루투스 연결
17:33	차량 시간 조작
17:33	스마트폰 시간 조작
17:34	전화 발신
17:34	전화 수신
17:34	음악 재생
17:35	차량 로그 덤프
17:59	스마트폰 로그 추출

【0078】 상술한 이벤트는 통화 이벤트, 음악 관련 이벤트, 내비게이션 이벤트 및 메시지 이벤트 등을 포함할 수 있으나, 이에 한정되지 않고, 근거리 무선통신 연결 환경에서 발생할 수 있는 모든 이벤트를 포함할 수 있다.

【0079】 도 3a를 참고하면, 로그 분석부(120)는 제1-2 로그에서, 시간의 역방향 기준 날짜가 변경(2024-07-19 -> 2024-07-13)된 타임 스탬프를 검출하고(301), 시간의 역방향 기준 시간이 변경(17:33:41 -> 12:00:00)된 타임 스탬프를

검출하여(303), 시간 조작을 의심할 수 있다.

【0080】 다른 예로, 로그 분석부(120)는 시간의 순방향 기준 로그 메시지 사이에 사전 설정된 시간 조작 의심 메시지가 발견되는 경우, 시간 조작 의심을 인지할 수 있다. 상기 사전 설정된 시간 조작 의심 메시지는 사전 설정된 규칙과 일치하지 않는 메시지이거나, 또는 사전 설정된 내용을 포함하는 메시지일 수 있다.

【0081】 예를 들어, 로그 분석부(120)는 하기와 같이 2022년도 10월 ~ 11월에 생성된 로그 메시지들 사이에, 2023년도 로그 메시지를 발견하는 경우, 해당 메시지를 시간 조작 의심 메시지로 간주하여, 시간 조작 의심을 인지할 수 있다.

【0082】 <로그 메시지>

【0083】 - 2022.10.22. 11:56:37 Log Message 1

【0084】 - 2022.10.23. 20:38:55 Log Message 2

【0085】 - 2023.01.22. 16:43:13 log message 3 -> 시간 조작 의심 메시지

【0086】 - 2022.11.24. 09.15:45 Log Message 4

【0087】 로그 분석부(120)는 제1-2 로그를 분석한 결과, 시스템 시간을 기준으로 로그 데이터가 생성되다가 이동통신 단말 형태의 사용자 단말(예를 들어, 스마트폰) 시간을 조작한 시점에서 도 3a와 같이 타임 스탬프가 변경되어 로그 데이터가 생성되는 것을 검출할 수 있다.

【0088】 만약 안티 포렌식 행위를 수행하는 조작자가 날짜 및 시간을 조작한다면, 미래 시간보다는 과거로 조작하여 포렌식 행위에 혼란을 줄 수 있다. 예를

들어, 사건 발생 시간대에 조작자 본인은 그 시간대에 다른 행위를 하고 있었다고 주장할 수 있다. 예를 들어, 차량 단말과 사용자 단말 모두의 시간을 과거로 조작하되, 서로 다른 과거 시간으로 조작하여 혼란을 증가 시킬 수 있다. 이에, 개시되는 로그 분석부(120)는 상술한 방식을 통해 시간 조작 의심 흔적을 파악할 수 있다. 이때, 조작되는 시간은 과거 시간에 한정되지 않고, 미래 시간도 포함할 수 있음은 당연하다 할 것이다.

【0089】 도 3b를 참고하면, 로그 분석부(120)는 차량 단말과 사용자 단말 간에 근거리 무선통신 연결(예를 들어, 블루투스 연결) 환경에서 수행된 이벤트(예를 들어, 전화 통화 이벤트 및 노래 재생 이벤트)가 조작된 시간에서 이루어진 것으로 제1-2 로그에서 확인할 수 있다.

【0090】 로그 분석부(120)는 도 3b와 같이, 시나리오에서 작성한 이벤트들이 발생된 로그 데이터들을 발견할 수 있다. 이에, 시간 조작된 상태에서 타임 스탬프는 변경된 시간을 기준으로 로그가 생성되고, 변경된 타임스탬프에서 이벤트가 수행되었음을 확인할 수 있다.

【0091】 도 4를 참고하면, 로그 분석부(120)는 제1-1 로그에서, 시간의 역방향 기준 날짜(2024-07-19 -> 2024-07-11) 및 시간(17:33:17 -> 14:00:00)이 변경된 타임 스탬프를 검출하여(401), 시간 조작을 의심할 수 있다. 이때, 로그 분석부(120)는 변경된 타임 스탬프에서 통화 이벤트 및 음악 이벤트가 수행된 로그(402)를 확인할 수 있다.

【0092】 로그 분석부(120)는 제1-1 로그를 분석한 결과, 제1-2 로그의 분석

결과와 유사하게 차량의 시스템 시간을 따르는 로그 데이터들이 생성되다가 도 4의 401과 같이, 시간 조작이 수행된 시점에 타임 스탬프가 변경되어 로그 데이터가 생성됨을 검출할 수 있다. 이후, 도 4의 402와 같이 변경된 타임 스탬프에서 통화 이벤트 및 음악 이벤트가 수행된 로그를 확인할 수 있다.

【0093】로그 분석부(120)는 사전 설정된 제2 분석기준에 따라 시스템 로그를 분석하여 시간 조작 탐지를 수행할 수 있다. 상기 사전 설정된 제2 분석기준은 시스템 로그를 분석하여 시간 조작 탐지를 수행하기 위한 기준으로, 시스템 환경(예를 들어, 안드로이드, 리눅스 등) 또는 대상 기기(예를 들어, 차량 단말, 사용자 단말)에 따라 차별화 되게 설정된 기준을 포함할 수 있으며, 이는 운용자에 의해서 사전 설정될 수 있다.

【0094】로그 분석부(120)는 이동통신 단말 시스템 로그인 제2-2 로그 중 특정 로그 메시지로부터 사전 설정된 시간 조작 로그 메시지를 검출하여 시간 조작 탐지를 수행할 수 있다.

【0095】일 예로, 안드로이드 시스템 기반 이동통신 단말의 경우, 로그 분석부(120)는 제2-2 로그 중 dumpstate.txt 파일을 분석하되, 해당 파일에서 data/misc/wifi_hostapd/dataUsage_log.txt 경로에 도 5와 같은 시간 조작 로그 메시지(예를 들어, date changed, User Changed Time to yyyy-MM-dd HH:mm)가 기록된 것을 검출할 수 있다.

【0096】도 5를 참조하면, 로그 분석부(120)는 7-19 17:32:30에서 7-13 17:33:32로 날짜 변경이 먼저 이루어졌고, 이후 07-13 12:00:00로 시간 변경이 이루어졌음을 확인할 수 있다. 이때, 로그 분석부(120)는 제2-2 로그에서, date changed, User Change Time to yyyy-MM-dd HHmmss 로그 메시지가 존재함을 검출할 수 있다. 이는, 안티 포렌식 행위가 수행되었음을 입증하는 자료로 활용될 수 있다.

【0097】다른 예로, 로그 분석부(120)는 차량 시스템 로그인 제2-1 로그로부터 시간 조작 시 생성되는 사전 설정된 시간 조작 파일을 검출하고, 시간 조작 파일을 분석하여 시간 조작을 탐지할 수 있다.

【0098】도 6을 참조하면, 제2-1 로그가 안드로이드 시스템 기반일 경우, 로그 분석부(120)는 "SET_USER_TIME@UNIX Epoch Time"로그 파일(600)과 같은 시간 조작 파일을 검출하고, 해당 파일에서 시간 조작 로그 메시지(601)를 확인 할 수 있다. 이때, 차량 인포테인먼트 로그 파일(예: SET_USER_TIME@1720674000081)에서 @ 뒤에 있는 숫자는 변경된 시간을 Unix Epoch Time으로 표현된 것일 수 있다. 로그 분석부(120)는 이러한 원리에 따라 해당 Unix Epoch Time을 사전 설정된 프로그램(예를 들어, DCode v5.6 프로그램)을 통해 시간 변환(Unix Epoch Time 시간 변환)하여 조작된 시간임을 파악할 수 있다.

【0099】이때, 로그 분석부(120)는 시간 변환된 텍스트 파일을 메모장을 통해 열어 도 6과 같은 시간 조작 정보가 기록된 파일(SET_USER_TIME@UNIX Epoch Time 파일)을 획득할 수 있다.

【0100】또한, 로그 분석부(120)는 시간 조작 파일에서 status 항목의 isUserTimeset 부분(601)이 true로 설정된 시간 조작 로그 메시지를 확인하면, 차량의 시간이 조작되었음을 확인할 수 있다. 또한, 로그 분석부(120)는 시간 조작 파일에서 offset항목의 new부분이 -703998606으로 설정 되어있음을 확인할 수 있다. 이를 통해, 로그 분석부(120)는 조작된 시간이 기존 이전 시간에서 얼마만큼 차이가 나는지를 파악할 수 있다. 로그 분석부(120)는 설정된 Unix Epoch Time과 -703998606을 기초로 시간 변환 계산을 진행하여 시간이 조작되었음을 탐지할 수 있다.

【0101】로그 분석부(120)는 에포크 타임 컨버터(epoch time converter) 기능을 이용하여 시간 조작 파일의 오프셋(offset)을 통해 조작 이전 시간을 확인할 수 있다. 이때, 조작 이전 시간은 조작된 시간 이전의 시간 즉, 현 조작된 시간의 바로 직전의 시간을 의미할 수 있다. 만약, 시간 조작이 2회 이상 발생한 경우, 상기 조작 이전 시간은 실제 시간과 차이가 있을 수 있다. 실제 시간을 파악하는 방법의 상세 설명은 후술하기로 한다.

【0102】로그 분석부(120)는 사전 설정된 제3 분석기준에 따라 시스템 로그 분석을 통해 시간 조작 전 실제 시간을 파악할 수 있다. 상기 사전 설정된 제3 분석기준은 시스템 로그를 분석하여 시간 조작 전 실제 시간을 파악하기 위한 기준으로, 시스템 환경(예를 들어, 안드로이드, 리눅스 등) 또는 분석 대상(예를 들어, 차량 단말, 사용자 단말)에 따라 차별화 되게 설정된 기준을 포함할 수 있으며, 이는 운용자에 의해서 사전 설정될 수 있다.

【0103】일 예로, 로그 분석부(120)는 차량 시스템 로그인 제2-1 로그 중 특정 로그 파일로부터 사전 설정된 그리니치 표준시(GMT)를 포함하는 로그 메시지를 확인하고, 확인된 그리니치 표준시에 지역별 시간차를 반영하여 실제 시간을 파악할 수 있다.

【0104】예를 들어, 로그 분석부(120)는 차량 단말의 시스템 로그 중 특정 로그 파일(예를 들어, mofgen2.log 파일)을 분석하면, 그리니치 표준시로 표현된 로그 메시지(예를 들어, Date Fri, 19 Jul 2024 08:33:31 GMT)를 확인할 수 있다. 한국 표준시는 그리니치 표준시보다 9시간 빠르므로, 이를 반영해 +9를 적용하면 실제 시간(예를 들어, Date Fri, 19 Jul 2024 17:33:31 GMT)을 도출할 수 있다.

【0105】개시되는 실시예의 로그 분석부(120)는 상술한 프로세스를 통해 조작자가 시스템을 재부팅하거나 다양한 방식으로 시간 조작을 시도하더라도, 시간 조작에 대한 의심과 증거를 발견할 수 있으며, 올바른 조작 당시 실제 시간(실제 현재 시간)을 파악할 수 있다.

【0106】평가부(130)는 로그 분석부(120)에서의 시간 조작 의심 흔적 발견, 시간 조작 탐지 및 실제 시간 파악을 비롯한 프로세스 진행 중 사전 설정된 평가기준에 따라 획득된 정보에 대한 평가를 수행할 수 있다.

【0107】시간 조작 탐지 서비스는 예를 들어, 웹 페이지 형태로 제공되거나 시간 조작 탐지 장치(100)에 설치된 애플리케이션을 통해 제공될 수 있다.

【0108】일 실시예에 따르면, 데이터 수집부(110), 로그 분석부(120) 및 평가부(130)는 각각 물리적으로 구분된 하나 이상의 장치를 이용하여 구현되거나, 하나 이상의 하드웨어 프로세서 또는 하나 이상의 하드웨어 프로세서 및 소프트웨어의 결합에 의해 구현될 수 있으며, 도시된 예와 달리 구체적 동작에 있어 명확히 구분되지 않을 수 있다.

【0110】도 7은 일 실시예에 따른 시간 조작 탐지 방법을 설명하기 위한 흐름도이다. 도 7에 도시된 방법은 예를 들어, 전술한 시간 조작 탐지 장치(100)에 의해 수행될 수 있다. 도시된 흐름도에서는 상기 방법을 복수 개의 단계로 나누어 기재하였으나, 적어도 일부의 단계들은 순서를 바꾸어 수행되거나, 다른 단계와 결합되어 함께 수행되거나, 생략되거나, 세부 단계들로 나뉘어 수행되거나, 또는 도시되지 않은 하나 이상의 단계가 부가되어 수행될 수 있다.

【0111】후술하는 시간 조작 탐지 방법은 상술한 도 1 내지 도 6을 참조하여 설명한 시간 조작 탐지 장치(100)의 동작을 동일하게 구현할 수 있으나, 설명의 편의를 위해 중복되는 상세 설명은 생략하기로 한다.

【0112】1100 단계에서, 시간 조작 탐지 장치(100)는 데이터 수집부(110)를 통해 분석 대상으로부터 시스템 로그 및 근거리 무선통신 로그 중 적어도 하나 이상을 포함하는 로그 데이터를 수집할 수 있다.

【0113】상기 분석 대상은 차량 단말 및 사용자 단말 중 적어도 하나 이상을 포함할 수 있다. 상기 사용자 단말은 이동통신 단말 및 유선통신 단말 중 적어도 하나 이상을 포함할 수 있다.

【0114】1200 단계에서, 시간 조작 탐지 장치(100)는 로그 분석부(120)를 통해 사전 설정된 제1 분석기준에 따라 상기 근거리 무선통신 로그 및 상기 시스템 로그 중 적어도 하나 이상을 분석하여 시간 조작 의심 흔적을 발견할 수 있다.

【0115】일 예로, 시간 조작 탐지 장치(100)는 시스템 로그에서 사전 설정된 시간 조작 의심 로그 메시지를 발견하면 시간 조작 의심 흔적으로 판단할 수 있다.

【0116】예를 들어, 도 2를 참고하면, 시간 조작 탐지 장치(100)는 시스템 로그에서 네트워크 동기화 오프(off) 상태를 포함하는 시간 조작 의심 로그 메시지를 검출하면 시간 조작 의심 흔적으로 판단할 수 있다. 이때, 시스템 로그는 리눅스 시스템 환경에서 수집된 시스템 로그일 수 있다.

【0117】시간 조작 탐지 장치(100)는 시간 조작 의심 흔적을 발견한 후, 시스템 로그에서 사전 설정된 시간 조작 로그 메시지를 통해 조작된 시간을 파악할 수 있다. 상기 조작된 시간은 조작자에 의해서 실제 시간이 실제와 다른 시간으로 조작된 시간을 의미할 수 있다.

【0118】상술한 근거리 무선통신 로그는 제1-1 로그 및 제1-2 로그를 포함할 수 있다. 성가 제1-1 로그는 차량 근거리 무선통신 로그를 의미할 수 있다. 상기 제1-2 로그는 이동통신 단말 근거리 무선통신 로그를 의미할 수 있다.

【0119】 다른 예로, 시간 조작 탐지 장치(100)는 시간의 순방향 기준 이벤트 발생 시나리오의 상기 제1-1 로그 또는 상기 제1-2 로그에서, 시간의 역방향 기준 날짜 또는 시간으로 변경된 타임 스탬프(time stamp)를 발견하는 경우, 시간 조작 의심을 인지할 수 있다.

【0120】 다른 예로, 로그 분석부(120)는 시간의 순방향 기준 로그 메시지 사이에 사전 설정된 시간 조작 의심 메시지가 발견되는 경우, 시간 조작 의심을 인지할 수 있다. 상기 사전 설정된 시간 조작 의심 메시지는 사전 설정된 규칙과 일치하지 않는 메시지이거나, 또는 사전 설정된 내용을 포함하는 메시지일 수 있다.

【0121】 예를 들어, 로그 분석부(120)는 하기와 같이 2022년도 10월 ~ 11월에 생성된 로그 메시지들 사이에, 2023년도 로그 메시지를 발견하는 경우, 해당 메시지를 시간 조작 의심 메시지로 간주하여, 시간 조작 의심을 인지할 수 있다.

【0122】 <로그 메시지>

【0123】 - 2022.10.22. 11:56:37 Log Message 1

【0124】 - 2022.10.23. 20:38:55 Log Message 2

【0125】 - 2023.01.22. 16:43:13 log message 3 -> 시간 조작 의심 메시지

【0126】 - 2022.11.24. 09.15:45 Log Message 4

【0127】 1200 단계에서, 시간 조작 의심 로그 메시지를 발견하는 방식의 상술한 일 예와 시간의 역방향 기준 날짜 또는 시간을 발견하는 방식의 다른 예는 각각 병합되어 구현되거나, 또는 단독으로 구현될 수 있다. 병합되어 구현되는 경우,

동작이 구현되는 순서는 운용자의 필요에 따라 임의로 결정할 수 있다.

【0128】 1300 단계에서, 시간 조작 탐지 장치(100)는 로그 분석부(120)를 통해 사전 설정된 제2 분석기준에 따라 시스템 로그를 분석하여 시간 조작 탐지를 수행할 수 있다.

【0129】 시간 조작 탐지 장치(100)는 제2-2 로그 중 특정 로그 메시지로부터 사전 설정된 시간 조작 로그 메시지를 검출하여 시간 조작 탐지를 수행할 수 있다.

【0130】 시간 조작 탐지 장치(100)는 제2-1 로그로부터 시간 조작 시 생성되는 사전 설정된 시간 조작 파일을 검출하고, 상기 시간 조작 파일을 분석하여 시간 조작을 탐지할 수 있다.

【0131】 1400 단계에서, 시간 조작 탐지 장치(100)는 로그 분석부(120)를 통해 사전 설정된 제3 분석기준에 따라 시스템 로그 분석을 통해 시간 조작 전 실제 시간을 파악할 수 있다.

【0132】 예를 들어, 시간 조작 탐지 장치(100)는 제2-1 로그 중 특정 로그 파일로부터 사전 설정된 그리니치 표준시(GMT)를 포함하는 로그 메시지를 확인하고, 확인된 그리니치 표준시에 지역별 시간차를 반영하여 실제 시간을 파악할 수 있다.

【0133】 1500 단계에서, 시간 조작 탐지 장치(100)는 평가부(130)를 통해 상술한 시간 조작 의심 흔적 발견, 시간 조작 탐지 및 실제 시간 파악을 비롯한 프로세스 진행 중 사전 설정된 평가기준에 따라 획득된 정보에 대한 평가를 수행할 수

있다.

【0135】 도 8은 일 실시예에 따른 컴퓨팅 장치를 포함하는 컴퓨팅 환경을 예시하여 설명하기 위한 블록도이다. 도시된 실시예에 서, 각 컴포넌트들은 이하에 기술된 것 이외에 상이한 기능 및 능력을 가질 수 있고, 이하에 기술된 것 이외에도 추가적인 컴포넌트를 포함할 수 있다.

【0136】 도시된 컴퓨팅 환경(10)은 컴퓨팅 장치(12)를 포함한다. 컴퓨팅 장치(12)는 일 실시예에 따른 시간 조작 탐지 장치(100)에 포함된 하나 이상의 컴포넌트일 수 있다.

【0137】 컴퓨팅 장치(12)는 적어도 하나의 프로세서(14), 컴퓨터 판독 가능 저장 매체(16) 및 통신 버스(18)를 포함한다. 프로세서(14)는 컴퓨팅 장치(12)로 하여금 앞서 언급된 예시적인 실시예에 따라 동작하도록 할 수 있다. 예컨대, 프로세서(14)는 컴퓨터 판독 가능 저장 매체(16)에 저장된 하나 이상의 프로그램들을 실행할 수 있다. 상기 하나 이상의 프로그램들은 하나 이상의 컴퓨터 실행 가능 명령어를 포함할 수 있으며, 상기 컴퓨터 실행 가능 명령어는 프로세서(14)에 의해 실행되는 경우 컴퓨팅 장치(12)로 하여금 예시적인 실시예에 따른 동작들을 수행하도록 구성될 수 있다.

【0138】 컴퓨터 판독 가능 저장 매체(16)는 컴퓨터 실행 가능 명령어 내지 프로그램 코드, 프로그램 데이터 및/또는 다른 적합한 형태의 정보를 저장하도록

구성된다. 컴퓨터 판독 가능 저장 매체(16)에 저장된 프로그램(20)은 프로세서(14)에 의해 실행 가능한 명령어의 집합을 포함한다. 일 실시예에서, 컴퓨터 판독 가능 저장 매체(16)는 메모리(랜덤 액세스 메모리와 같은 휘발성 메모리, 비휘발성 메모리, 또는 이들의 적절한 조합), 하나 이상의 자기 디스크 저장 디바이스들, 광학 디스크 저장 디바이스들, 플래시 메모리 디바이스들, 그 밖에 컴퓨팅 장치(12)에 의해 액세스되고 원하는 정보를 저장할 수 있는 다른 형태의 저장 매체, 또는 이들의 적합한 조합일 수 있다.

【0139】 통신 버스(18)는 프로세서(14), 컴퓨터 판독 가능 저장 매체(16)를 포함하여 컴퓨팅 장치(12)의 다른 다양한 컴포넌트들을 상호 연결한다.

【0140】 컴퓨팅 장치(12)는 또한 하나 이상의 입출력 장치(24)를 위한 인터페이스를 제공하는 하나 이상의 입출력 인터페이스(22) 및 하나 이상의 네트워크 통신 인터페이스(26)를 포함할 수 있다. 입출력 인터페이스(22) 및 네트워크 통신 인터페이스(26)는 통신 버스(18)에 연결된다. 입출력 장치(24)는 입출력 인터페이스(22)를 통해 컴퓨팅 장치(12)의 다른 컴포넌트들에 연결될 수 있다. 예시적인 입출력 장치(24)는 포인팅 장치(마우스 또는 트랙패드 등), 키보드, 터치 입력 장치(터치패드 또는 터치스크린 등), 음성 또는 소리 입력 장치, 다양한 종류의 센서 장치 및/또는 촬영 장치와 같은 입력 장치, 및/또는 디스플레이 장치, 프린터, 스피커 및/또는 네트워크 카드와 같은 출력 장치를 포함할 수 있다. 예시적인 입출력 장치(24)는 컴퓨팅 장치(12)를 구성하는 일 컴포넌트로서 컴퓨팅 장치(12)의 내부에 포함될 수도 있고, 컴퓨팅 장치(12)와는 구별되는 별개의 장치로 컴퓨팅 장치

(12)와 연결될 수도 있다.

【0141】일반적으로 로그 데이터는 시스템 재부팅 시 사라지는 휘발성 로그일 수 있다. 특히, 안드로이드 시스템에서 블루투스 통신을 통해 연결된 두 기기의 시간이 서로 다를 경우, 두 기기로부터 수집된 로그 데이터를 기초로 안티포렌식 행위 수사 시, 기준으로 고려해야 할 명확한 시간대가 존재하지 않을 수 있다. 또한, 양 기기의 시간이 모두 조작되었을 때, 올바른 조작 당시 실제 현재 시간대를 찾는 것이 어려울 수 있다. 이에, 개시되는 실시예는 상술한 기재를 통해 범죄자를 비롯한 조작자가 시간을 조작하거나 시스템을 재부팅 하더라도, 이를 효과적으로 인지하고 탐지하고, 정확한 실제 시간을 파악할 수 있다.

【0142】개시된 실시예들은 컴퓨터에 의해 실행 가능한 명령어를 저장하는 기록매체의 형태로 구현될 수 있다. 명령어는 프로그램 코드의 형태로 저장될 수 있으며, 프로세서에 의해 실행되었을 때, 프로그램 모듈을 생성하여 개시된 실시예들의 동작을 수행할 수 있다. 기록매체는 컴퓨터로 읽을 수 있는 기록매체로 구현될 수 있다.

【0143】이상에서 본 발명의 대표적인 실시예들을 상세하게 설명하였으나, 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자는 상술한 실시예에 대하여 본 발명의 범주에서 벗어나지 않는 한도 내에서 다양한 변형이 가능함을 이해할 것이다. 그러므로 본 발명의 권리범위는 설명된 실시예에 국한되어 정해져서는 안 되며, 후술하는 청구범위뿐만 아니라 이 청구범위와 균등한 것들에 의해 정해져야 한다.

【부호의 설명】

【0144】10: 컴퓨팅 환경

12: 컴퓨팅 장치

14: 프로세서

16: 컴퓨터 판독 가능 저장 매체

18: 통신 버스

20: 프로그램

22: 입출력 인터페이스

24: 입출력 장치

26: 네트워크 통신 인터페이스

100: 시간 조작 탐지 장치

110: 데이터 수집부

120: 로그 분석부

130: 평가부

【청구범위】

【청구항 1】

네트워크 동기화 환경에서 분석 대상으로부터 시스템 로그 및 근거리 무선통신 로그 중 적어도 하나 이상을 포함하는 로그 데이터를 수집하는 데이터 수집부; 및

사전 설정된 제1 분석기준에 따라 상기 근거리 무선통신 로그 및 상기 시스템 로그 중 적어도 하나 이상을 분석하여 시간 조작 의심 흔적을 발견하고, 사전 설정된 제2 분석기준에 따라 상기 시스템 로그를 분석하여 시간 조작 탐지를 수행하는 로그 분석부를 포함하는, 시간 조작 탐지 장치.

【청구항 2】

청구항 1에 있어서,

상기 분석 대상은, 차량 단말 및 사용자 단말 중 적어도 하나 이상을 포함하고,

상기 사용자 단말은, 이동통신 단말 및 유선 단말 중 적어도 하나 이상을 포함하는, 시간 조작 탐지 장치.

【청구항 3】

청구항 2에 있어서,

상기 로그 분석부는,

상기 시스템 로그에서 사전 설정된 시간 조작 의심 로그 메시지를 발견하면 상기 시간 조작 의심 흔적으로 판단하는, 시간 조작 탐지 장치.

【청구항 4】

청구항 3에 있어서,

상기 로그 분석부는,

상기 시스템 로그에서 네트워크 동기화 오프(off) 상태를 포함하는 시간 조작 의심 로그 메시지를 검출하면 상기 시간 조작 의심 흔적으로 판단하는, 시간 조작 탐지 장치.

【청구항 5】

청구항 4에 있어서,

상기 로그 분석부는,

상기 시간 조작 의심 흔적을 발견한 후, 상기 시스템 로그에서 사전 설정된 시간 조작 로그 메시지를 통해 조작된 시간을 파악하는, 시간 조작 탐지 장치.

【청구항 6】

청구항 1에 있어서,

상기 근거리 무선통신 로그는 제1-1 로그 및 제1-2 로그를 포함하고,

상기 로그 분석부는,

시간의 순방향 기준 이벤트 발생 시나리오의 상기 제1-1 로그 또는 상기 제1-2 로그에서, 시간의 역방향 기준 날짜 또는 시간으로 변경된 타임 스탬프(time stamp)를 발견하는 경우, 시간 조작 의심을 인지하는, 시간 조작 탐지 장치.

【청구항 7】

청구항 6에 있어서,

상기 로그 분석부는,

상기 제2-2 로그 중 특정 로그 메시지로부터 사전 설정된 시간 조작 로그 메시지를 검출하여 시간 조작 탐지를 수행하는, 시간 조작 탐지 장치.

【청구항 8】

청구항 7에 있어서,

상기 로그 분석부는,

상기 제2-1 로그로부터 시간 조작 시 생성되는 사전 설정된 시간 조작 파일을 검출하고, 상기 시간 조작 파일을 분석하여 시간 조작을 탐지하는, 시간 조작 탐지 장치.

【청구항 9】

청구항 1에 있어서,

상기 로그 분석부는,

시간의 순방향 기준 로그 메시지 사이에 사전 설정된 시간 조작 의심 메시지가 발견되는 경우, 시간 조작 의심을 인지하는, 시간 조작 탐지 장치.

【청구항 10】

청구항 1에 있어서,

상기 로그 분석부는,

사전 설정된 제3 분석기준에 따라 상기 시스템 로그 분석을 통해 시간 조작 전 실제 시간을 파악하는, 시간 조작 탐지 장치.

【청구항 11】

청구항 10에 있어서,

상기 로그 분석부는,

상기 제2-1 로그 중 특정 로그 파일로부터 사전 설정된 그리니치 표준시(GMT)를 포함하는 로그 메시지를 확인하고, 확인된 상기 그리니치 표준시에 지역별 시간차를 반영하여 상기 실제 시간을 파악하는, 시간 조작 탐지 장치.

【청구항 12】

시간 조작 탐지 장치에 의해 수행되는 방법에 있어서,

상기 시간 조작 탐지 장치가 분석 대상으로부터 시스템 로그 및 근거리 무선통신 로그 중 적어도 하나 이상을 포함하는 로그 데이터를 수집하는 단계;

사전 설정된 제1 분석기준에 따라 상기 근거리 무선통신 로그 및 상기 시스템 로그 중 적어도 하나 이상을 분석하여 시간 조작 의심 흔적을 발견하는 단계; 및

사전 설정된 제2 분석기준에 따라 상기 시스템 로그를 분석하여 시간 조작 탐지를 수행하는 단계를 포함하는, 시간 조작 탐지 방법.

【청구항 13】

청구항 12에 있어서,

상기 분석 대상은, 차량 단말 및 사용자 단말 중 적어도 하나 이상을 포함하고,

상기 사용자 단말은, 이동통신 단말 및 유선 단말 중 적어도 하나 이상을 포함하는, 시간 조작 탐지 방법.

【청구항 14】

청구항 13에 있어서,

상기 시간 조작 의심 흔적을 발견하는 단계에서,

상기 시스템 로그에서 사전 설정된 시간 조작 의심 로그 메시지를 발견하면 상기 시간 조작 의심 흔적으로 판단하는, 시간 조작 탐지 방법.

【청구항 15】

청구항 14에 있어서,

상기 시간 조작 의심 흔적을 발견하는 단계에서,

상기 시스템 로그에서 네트워크 동기화 오프(off) 상태를 포함하는 시간 조작 의심 로그 메시지를 검출하면 상기 시간 조작 의심 흔적으로 판단하는, 시간 조작 탐지 방법.

【청구항 16】

청구항 15에 있어서,

상기 시간 조작 탐지를 수행하는 단계에서,

상기 시간 조작 의심 흔적을 발견한 후, 상기 시스템 로그에서 사전 설정된 시간 조작 로그 메시지를 통해 조작된 시간을 파악하는, 시간 조작 탐지 방법.

【청구항 17】

청구항 12에 있어서,

상기 근거리 무선통신 로그는 제1-1 로그 및 제1-2 로그를 포함하고,

상기 시간 조작 의심 흔적을 발견하는 단계에서,

시간의 순방향 기준 이벤트 발생 시나리오의 상기 제1-1 로그 또는 상기 제1-2 로그에서, 시간의 역방향 기준 날짜 또는 시간으로 변경된 타임 스탬프(time stamp)를 발견하는 경우, 시간 조작 의심을 인지하는, 시간 조작 탐지 방법.

【청구항 18】

청구항 17에 있어서,

상기 시간 조작 탐지를 수행하는 단계에서,

상기 제2-2 로그 중 특정 로그 메시지로부터 사전 설정된 시간 조작 로그 메시지를 검출하여 시간 조작 탐지를 수행하는, 시간 조작 탐지 방법.

【청구항 19】

청구항 18에 있어서,

상기 시간 조작 탐지를 수행하는 단계에서,

상기 제2-1 로그로부터 시간 조작 시 생성되는 사전 설정된 시간 조작 파일을 검출하고, 상기 시간 조작 파일을 분석하여 시간 조작을 탐지하는, 시간 조작 탐지 방법.

【청구항 20】

청구항 12에 있어서,

상기 시간 조작 의심 흔적을 발견하는 단계에서,

시간의 순방향 기준 로그 메시지 사이에 사전 설정된 시간 조작 의심 메시지가 발견되는 경우, 시간 조작 의심을 인지하는, 시간 조작 탐지 방법.

【청구항 21】

청구항 12에 있어서,

사전 설정된 제3 분석기준에 따라 상기 시스템 로그 분석을 통해 시간 조작 전 실제 시간을 파악하는 단계를 더 포함하는, 시간 조작 탐지 방법.

【청구항 22】

청구항 21에 있어서,

상기 제2-1 로그 중 특정 로그 파일로부터 사전 설정된 그리니치 표준시(GMT)를 포함하는 로그 메시지를 확인하고, 확인된 상기 그리니치 표준시에 지역별 시간차를 반영하여 상기 실제 시간을 파악하는, 시간 조작 탐지 방법.

【요약서】

【요약】

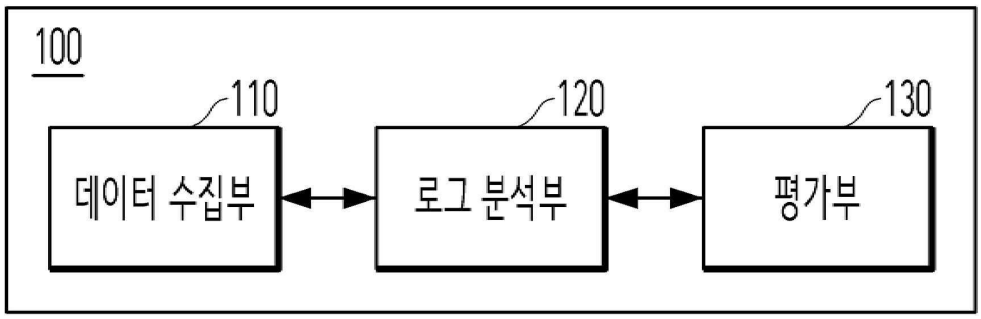
스마트폰과 연결된 차량 인포테인먼트 시스템에서의 시간 조작 탐지 장치 및 방법이 개시된다. 본 발명의 일 실시예에 따른 시간 조작 탐지 장치는 차량 단말 및 차량 단말과 통신 연결된 스마트폰으로부터 각각 시스템 로그 및 근거리 무선통신 로그 중 적어도 하나 이상을 포함하는 로그 데이터를 수집하는 데이터 수집부; 및 사전 설정된 제1 분석기준에 따라 근거리 무선통신 로그 및 시스템 로그 중 적어도 하나 이상을 분석하여 시간 조작 의심 흔적을 발견하고, 사전 설정된 제2 분석기준에 따라 시스템 로그를 분석하여 시간 조작 탐지를 수행하는 로그 분석부를 포함한다.

【대표도】

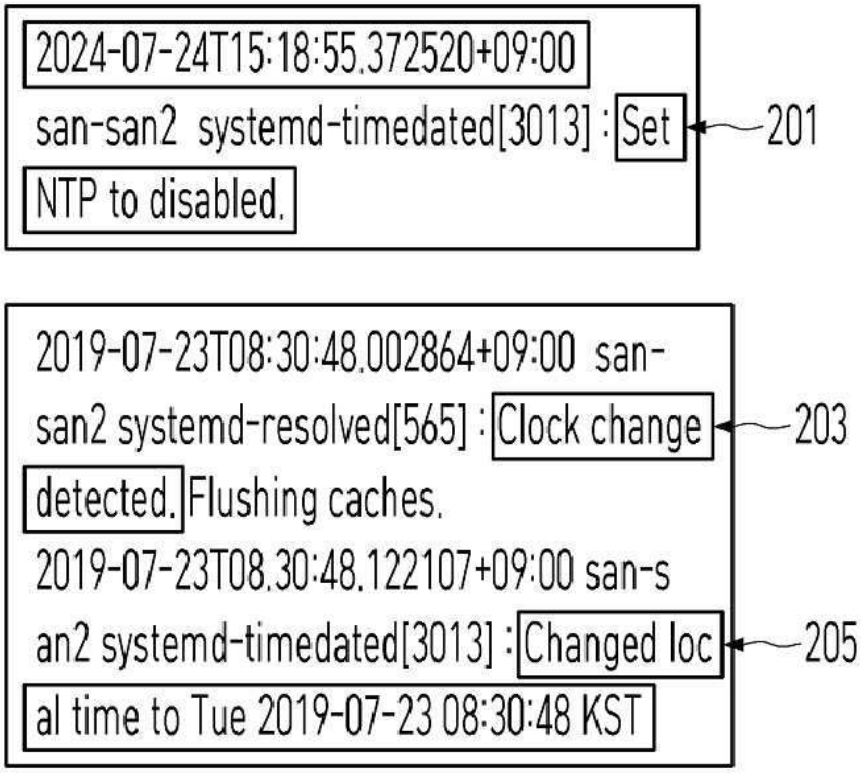
도 1

【도면】

【도 1】


100
110
120
130
데이터 수집부
로그 분석부
평가부

【도 2】


2024-07-24T15:18:55.372520+09:00
san-san2 systemd-timedated[3013] : Set
NTP to disabled.
201
2019-07-23T08:30:48.002864+09:00 san-
san2 systemd-resolved[565] : Clock change
detected. Flushing caches.
203
2019-07-23T08.30:48.122107+09:00 san-s
an2 systemd-timedated[3013] : Changed loc
al time to Tue 2019-07-23 08:30:48 KST
205

【도 3a】

2024-07-19 17:33:32.703606 controller
2024-07-13 17:33:32.722573 SamsungElect_85:e7:b7 (SOS◆◆◆ S21)
날짜 변경 (301)
host
HIC_EVT
8 Rcvd Number of Completed Packets
LGInnotek_8c:11:84 (Avante)
RTP
63 PT=MPEG Audio, SSRC=0x2, Seq=4018, Time=250880, Mark
2024-07-13 17:33:41.846439 controller
2024-07-13 12:00:00.307375 controller
시간 변경 (303)
host
HIC_EVT
14 Rcvd Sniff Subrating
host
HIC_EVT
60 Rcvd LE Meta (LE Extended Advertising Report)

【도 3b】

2024-07-13 12:00:22.412200 SamsungElect_85:e7:b7 SOS◆◆◆ S21) LGInnotek_8c:11:84 (Avante) HFP 52 Sent +CLCC: 1,0,2,1,0,"010492321988",129 -전화 통화 이벤트

2024-07-13 12:01:16.998617 SamsungElect_85:e7:b7 SOS◆◆◆ S21) LGInnotek_8c:11:84 (Avante) AVRCP 123 Sent Vendor dependent: Stable - GetElementAttributes - Title: "Over the Horizon -노래 재생 이벤트

【도 4】

401

07-19 17:33:17.733 - - D BTA_BT_PLAYER_MANAGER sendMetaDataToHomeWidget , title:Over the Horizon , artist:Samsung ,device:SOS의 S21 ,Tune:null ,Browsing:false ,Cur:-1 ,Tot:-1 ,Play:0 ,Key:9466

07-11 14:00:00.104 - - D AvrcpControllerService [BTAD] BroadcastPlayPosition pos:-1 ,len:-1

402

07-11 14:00:48.212 - - D ==BTHFPManager=== onMobisBTCallInfo :: getContacPhoto callNum = 01049232198, callerName = 조민혁

07-11 14:01:02.806 - - D BTA_BT_PLAYER_MANAGER sendMmetaDataToHomeWidget ,title:Over the Horizon , artist:Samsung ,device:SOS의 S21 ,Tune:null ,Browsing:false ,Cur:-1 ,Tot:-1 ,Play:0 ,Key:9466

【도 5】

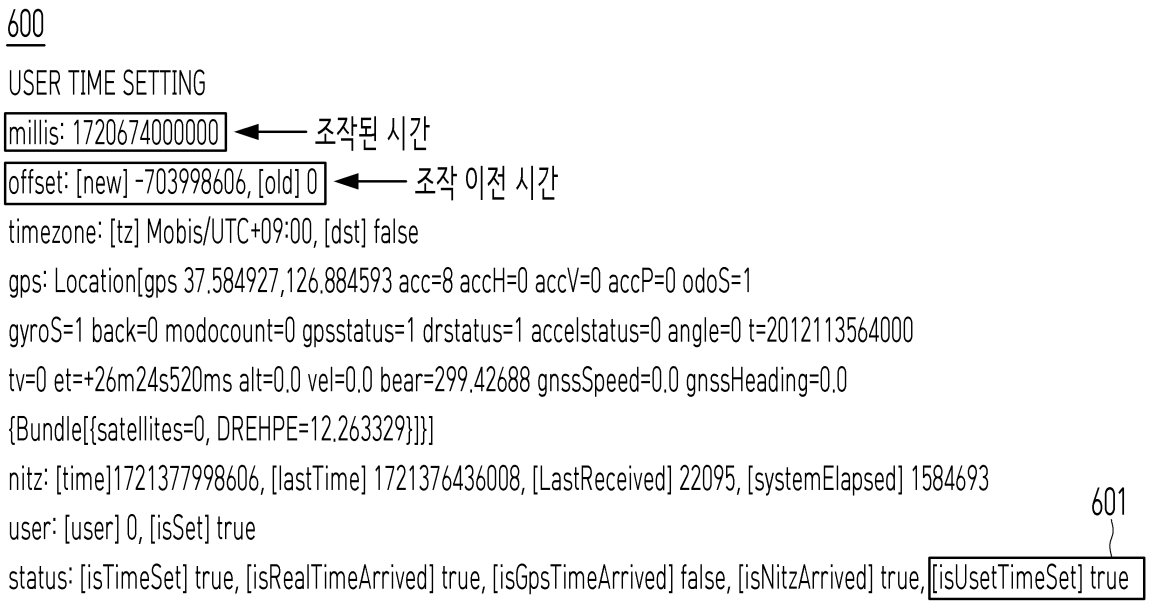
07-19 17:32:30.917 SemWifiDataUsage date changed:current date = 浚댑쉿?? 7??12, 2024 at 5:27 ?ㅻ쏇, new date = 浚댑쉿?? 7??19, 2024 at 5:32 ?ㅻ쏇

07-13 17:33:32.831 SemWifiDataUsage User Changed Time to yyyy-MM-dd HH:mm:ss = 2024-07-13 17:33:32

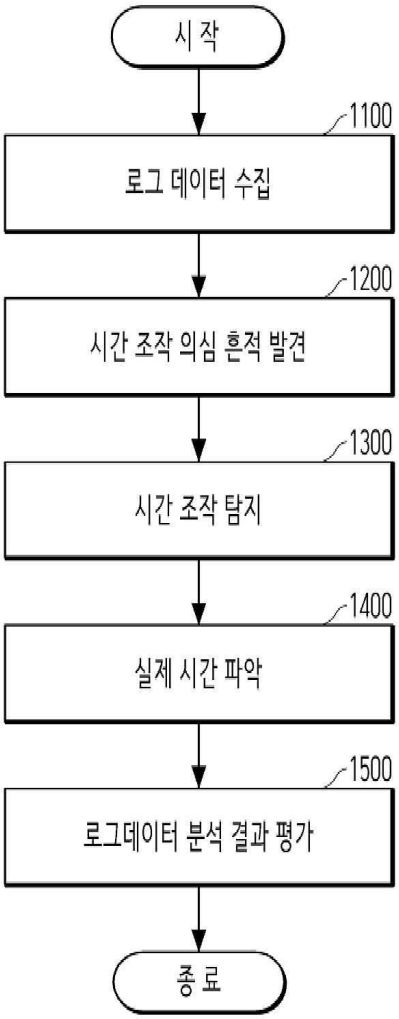
07-13 17:33:32.833 SemWifiDataUsage date changed:current date = 浚댑쉿?? 7??19, 2024 at 5:32 ?ㅻ쏇, new date = ?좎쉿?? 7??13, 2024 at 5:33 ?ㅻ쏇

07-13 12:00:00.110 SemWifiDataUsage User Changed Time to yyyy-MM-dd HH:mm:ss = 2024-07-13 12:00:00

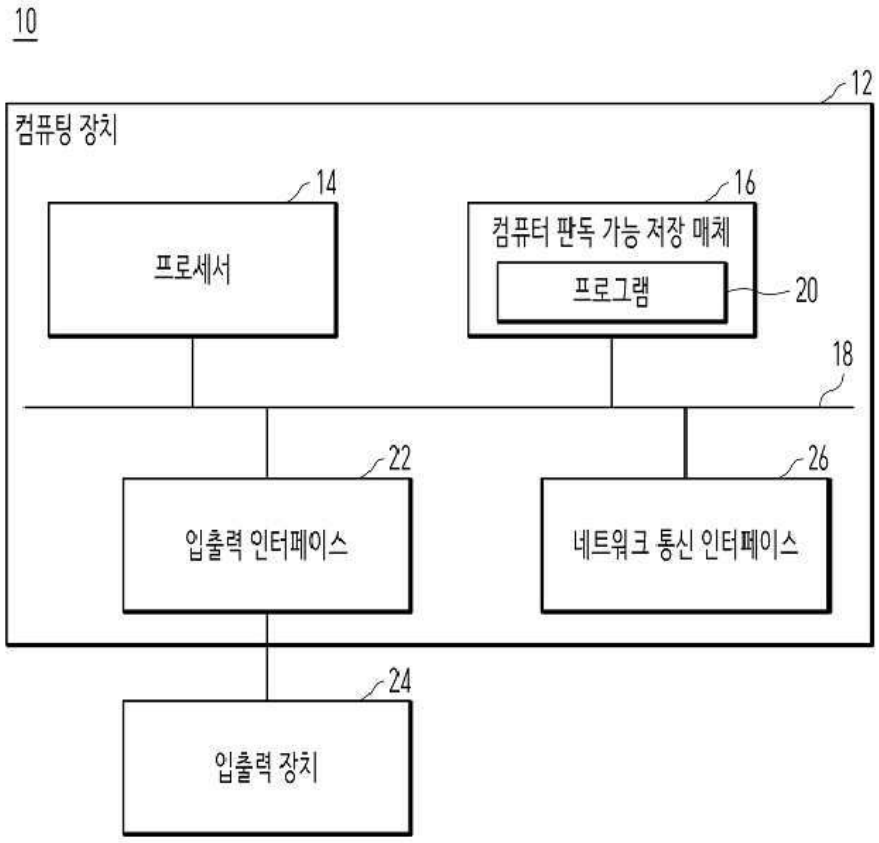
【도 6】


600
USER TIME SETTING
millis: 1720674000000
조작된 시간
offset: [new] -703998606, [old] 0
조작 이전 시간
timezone: [tz] Mobis/UTC+09:00, [dst] false
gps: Location[gps 37.584927,126.884593 acc=8 accH=0 accV=0 accP=0 odoS=1
gyroS=1 back=0 modocount=0 gpsstatus=1 drstatus=1 accelstatus=0 angle=0 t=2012113564000
tv=0 et=+26m24s520ms alt=0.0 vel=0.0 bear=299.42688 gnssSpeed=0.0 gnssHeading=0.0
{Bundle[{satellites=0, DREHPE=12.263329}]}]
nitz: [time]1721377998606, [lastTime] 1721376436008, [LastReceived] 22095, [systemElapsed] 1584693
user: [user] 0, [isSet] true
status: [isTimeSet] true, [isRealTimeArrived] true, [isGpsTimeArrived] false, [isNitzArrived] true, [isUsetTimeSet] true
601

【도 7】


시 작
1100
로그 데이터 수집
1200
시간 조작 의심 흔적 발견
1300
시간 조작 탐지
1400
실제 시간 파악
1500
로그데이터 분석 결과 평가
종 료

【도 8】


10
12
컴퓨팅 장치
14
프로세서
16
컴퓨터 판독 가능 저장 매체
프로그램
20
18
22
입출력 인터페이스
26
네트워크 통신 인터페이스
24
입출력 장치

WDSC2024

2024 Workshop on Dependable and Secure Computing

(2024 한국정보과학회 정보보안 및 고신뢰컴퓨팅 하계 워크숍)

일시: 2024년 8월 19일 (월) ~ 21일 (수)

장소: 제주 시리우스호텔 (제주시 연동)

주최: 한국정보과학회 정보보안및고신뢰컴퓨팅연구회

한국정보과학회

정보보안 및 고신뢰컴퓨팅연구회

2024 Workshop on Dependable and Secure Computing (WDSC2024)

(2024 한국정보과학회 정보보안 및 고신뢰컴퓨팅 하계워크샵 프로그램)

(워크샵 홈페이지: https://sites.google.com/view/wdsc2024)

일시: 2024년 8월 19일 (월) ~ 21일 (수)
장소: 제주 시리우스호텔 (제주시 연동)
주최: 한국정보과학회 정보보안및고신뢰컴퓨팅연구회

워크샵 조직위원장:
조성제 교수(단국대)

워크샵 프로그램위원장:
허준영 교수(한성대)
최윤호 교수(부산대)

정보보안 및 고신뢰컴퓨팅연구회 운영위원장
한환수 교수(성균관대)

워크샵 준비 위원

프로그램:	박문주 교수(인천대)
	임을규 교수(한양대)
	김동규 교수(한양대)
	박민규 교수(건국대)
현장:	민 홍 교수(가천대)
	김판구 교수(조선대)
홍보:	조진성 교수(경희대)
	정진만 교수(인하대)
	최종무 교수(단국대)
	김봉재 교수(충북대)
	이재홍 교수(대전대)
홍보(웹)/재무:	전광일 교수(한국공학대)

한국정보과학회 정보보안 및 고신뢰컴퓨팅연구회

한국정보과학회
정보보안 및 고신뢰컴퓨팅연구회

초대의 말씀

안녕하십니까?

한국정보과학회 정보보안 및 고신뢰컴퓨팅연구회는 정보보안과 고신뢰컴퓨팅의 전반적인 주제에 대해 연구, 토론하며, 특히 다음과 같은 연구 주제에 대해 많은 관심을 가지고 있습니다. 1) 컴퓨터시스템, 소프트웨어, 하드웨어를 위한 정보보안 및 고신뢰컴퓨팅 이론; 2) 정보보안 시스템, 신뢰컴퓨팅, 침입 및 오류의 탐지/방지/복구, 방화벽과 네트워크 등 정보보안과 고신뢰가 고려된 시스템 설계; 3. 정보보안과 고신뢰를 평가하기 위한 모델링 및 예측, 테스트 베드 설계, 모니터링, 벤치마킹, 서비스 품질 평가 등; 4) 정보보안과 고신뢰컴퓨팅이 필요한 실시간 시스템, 임베디드 시스템, 분산 시스템 등의 다양한 응용 프로그램; 5) 탐지 및 복구를 지원하는 운영체계, 자기 점검, 테스트 및 검증 등을 위한 소프트웨어 설계를 비롯한 정보보안 및 고신뢰컴퓨팅과 관련된 전반적인 연구.

한국정보과학회 정보보안 및 고신뢰컴퓨팅 하계워크샵(Workshop on Dependable and Secure Computing: WDSC)은 정보보안 및 고신뢰컴퓨팅 분야의 연구 결과를 교류하고 토의하는 목적으로 매년 개최되는 워크샵입니다. 2024년 정보보안 및 고신뢰컴퓨팅워크샵(WDSC2024)가 8월 19일부터 21일까지 제주시리우스호텔(제주시 연동)에서 개최됩니다. WDSC2024에서는 정보보안및고신뢰컴퓨팅 분야의 전문가 11분을 초청하여 정보보안 및 고신뢰컴퓨팅 이슈들에 대한 초청 발표가 있을 예정입니다.

WDSC2024가 정보보안 및 고신뢰컴퓨팅 연구 분야에 관심을 가진 분들의 학술 정보교환의 장소가 되기를 바라며, 많은 관심과 참여를 부탁드립니다.

감사합니다.

2024년 6월

2024년 한국정보과학회 정보보안 및 고신뢰컴퓨팅 하계워크샵 (WDSC2024)

조직위원장 조성제

프로그램위원장 허준영, 최윤호

WDSC2024 워크샵 프로그램

8월 19일	세션명	발표자	발표 주제
13:00~ 13:10	개회식 (사회: 성균관대 한환수 교수) <시리우스 홀>		
13:10 ~ 16:20	KISDI 정책과제발표 : 인공지능에 초점을 둔 디지털 위험 및 보안 <시리우스 홀>	좌장: 단국대 조성제 교수	
		김명주 교수 (40′) (서울여자대학교)	Keynote Speech : AI for Security, Security for AI
		권태경 교수 (40′) (연세대학교)	전문가 특강 : 챗 GPT 등 생성형 AI 활용 보안 가이드라인
		휴식 시간(20′)	
		최윤호 교수 (30′) (부산대학교)	o 1세부: 사이버 보안에서 인공지능을 활용하는 모범 사례 및 잠재적 부작용 식별 - 인공지능을 활용한 사이버 보안 강화 방안
		오준형 교수 (30′) (서울여자대학교)	o 2세부: 사이버 보안 위협에 대처하기 위한 보안 인공지능의 사회 전반 확산 방안 - 정책적 제언 및 지원을 중심으로
		자유 토론(30′)	
16:20 ~ 17:40	전문가 초청 강연1 <시리우스 홀>	좌장: 한국공학대 전광일 교수	
		박지혜 변리사 (부산대학교) (40′)	AI기반 발명의 특허출원 전략
		최종무 교수 (단국대학교) (40′)	학습된 인덱스(Learned Index) 최근 연구 동향
8월 20일	**세션명**	**발표자**	**발표 주제**
09:00 ~ 12:00	KISDI 정책과제 전문가 토론 - 사이버 보안에서 인공지능 활용과 부작용 (좌장: 단국대 조성제 교수) <베가 홀>		
09:00 ~ 10:30	일반 논문 발표 세션1 <시리우스 홀>	좌장: 부산대 최윤호 교수	
		제3자에게 데이터 교집합을 제공하는 MPSI에 관한 연구, 김기환, 김태훈, 김수현, 이임영(순천향대)	
		안전한 상호 인증을 제공하는 스마트카드를 이용한 ECC기반의 새로운 인증 기법, 윤성철, 장석준, 김수현, 이임영(순천향대)	
		MAVLink를 이용한 UAS 공격 시나리오 분석, 유영빈, 손지언, 조진성(경희대)	
		오픈소스 기반 무인이동체에 대한 보안성 강화, 손지언, 유영빈, 손희승, 김정활, 이준호, 조진성(경희대)	

		오픈소스 패키지 저장소 공격 유형과 대응 기술 조사, 이찬우, 허준영(한성대)	
		랜섬웨어 암호화 알고리즘 분석 연구, 전혜민, 최두섭, 임을규(한양대)	
10:30 ~ 12:00	일반 논문 발표 세션2 <시리우스 홀>	좌장: 한양대 임을규 교수	
		SQLite 활용 실태 조사 및 삭제된 레코드 복원 기법 분석, 이성현, 김나경, 최연규, 김민성(단국대), 이인수, 박수영(대검찰청), 최종무(단국대)	
		직렬화 기법 성능 분석을 위한 벤치마크 설계 및 구현, 이제연, 이용민, 신수환(단국대), 이인수, 박수영(대검찰청), 유시환, 최종무(단국대)	
		블루투스로 연결된 스마트폰과 자동차 인포테인먼트 시스템에서의 시간 조작 탐지, 조민혁, 이산, 정지헌, 김선재, 조성제(단국대)	
		안드로이드 멀웨어 탐지의 지속가능성을 위한 K-Means 기반 분류 모델, 정성원, 안석현, 조성제, 김동재(단국대), 황영섭(선문대)	
		계층적 클러스터링 하이브리드 모델의 활용을 통한 지속 가능한 안드로이드 악성 앱 탐지 기법, 이승민, 안석현, 조성제, 김동재(단국대), 황영섭(선문대)	
		윈도우 10 환경에서의 안티포렌식 기법 및 탐지 알고리즘 분석, 윤혜준(조선대), 조효빈(성신여대), 박창민(순천향대), 하유정(아주대), 최규연(동국대), 유승완(고려대), 이산(단국대)	
13:10 ~ 16:50	전문가 초청 강연2 <시리우스 홀>	좌장: 한성대 허준영 교수	
		이호준 교수 (성균관대학교) (40′)	RustSan: Rust를 위한 효율적인 메모리 오류 검출 기술
		박제만 교수 (경희대학교) (40′)	AI 및 소프트웨어에 대한 고급 사이버 포렌식 기법
		박승현 교수 (한성대학교) (40′)	소프트웨어 정의 스마트 자동차 보안 연구개발 최근 동향
	휴식 시간(20′)		
	전문가 초청 강연3 <시리우스 홀>	좌장: 건국대 박민규 교수	
		문재찬 연구원 (국가보안기술연구소) (40′)	Expert x LLM: 분석가가 바라보는 사이버 위협 분석에서의 LLM 활용
		박영웅 연구원 (국가보안기술연구소) (40′)	안드로이드 코드 분석과 AI: 분석가 눈으로 보는 안드로이드 코드분석 연구

8월 21일	세션명	발표자	발표 주제
09:00~ 11:00	KISDI 정책과제 전문가 토론 - 사이버 보안 위협 대처를 위한 인공지능 기술 확산 방안 (좌장: 조선대 김판구 교수) <시리우스 홀>		
11:00~ 11:30	폐회식 <시리우스 홀>		

등록안내

o 등록비

날짜	일반	학생
사전등록(2024년 8월 15일 오후 6시 이전)	350,000원	300,000원
일반등록(현장등록)	400,000원	350,000원

o WDSC2024에 등록하실 분들은 워크샵 홈페이지 (https://sites.google.com/view/wdsc2024)의 온라인 등록사이트를 이용하여 등록하여주시기 바랍니다.

행사장: 제주시리우스 호텔(064-743-1147) 제주시 도령로 133

제주 시리우스 호텔 찾아오시는 길: https://naver.me/ForHTuG1

목차

초청 발표

일반 논문

블루투스로 연결된 스마트폰과 자동차 인포테인먼트 시스템에서의 시간 조작 탐지

조민혁[O1] 이산[2] 정지헌[3] 김선재[4] 조성제[1]
단국대학교 모바일시스템공학과[01] 단국대학교 산업보안학과[2],
단국대학교 컴퓨터학과[3,4], 단국대학교 소프트웨어학과[1]
{cgumgek8, dltksdhkd915, wlgjsjames7224, rlatjswo0824, sjcho}@dankook.ac.kr

Detecting Timestamp Manipulation in Bluetooth-Connected Smartphone and Automotive Infotainment System

Min Hyuk Cho[O1], San Lee[2], Jiheun Jung[3], Sun Jae Kim[4], Seong-Je Cho[1]
Department of Mobile System Engineering, Dankook University[O1]
Department of Industrial Security, Dankook University[2]
Department of Computer Science and Engineering, Dankook University[3,4]
Department of Software Science, Dankook University[1]

요 약

디지털 포렌식은 컴퓨터나 스마트폰 등의 IT 기기에 저장된 디지털 데이터를 과학적으로 수집·분석하여 사건을 해결하는 것이다. 이때 IT 기기에서 추출한 타임스탬프는 이벤트들을 시간대별로 분석할 수 있는 근거이므로 디지털 포렌식 조사에서 매우 기본적이면서도 중요한 정보이다. 본 논문에서는 범죄자가 자신의 행위를 숨기기 위해, 자신 IT 기기의 시간을 조작했을 경우에, 시간 조작 여부를 탐지하는 기법을 제안한다. 구체적으로는 자동차의 인포테인먼트 시스템이 운전자의 스마트폰과 블루투스로 연결된 상황에서 발생한 이벤트들의 타임스탬프를 운전자가 조작했을 때, 이를 탐지하는 방법을 제안한다. 시간 조작 여부를 탐지하기 위해서, 우리는 스마트폰과 자동차 인포테인먼트 시스템으로부터, 블루투스 HCI 스눕 로그(HCI Snoop log), 블루투스 로그, 시스템 로그 등을 추출하고 분석한다. 시나리오 기반 실험을 통해, 제안 기법이 시간 조작 여부를 탐지할 수 있음을 보인다.

1. 서 론

디지털 포렌식(Digital Forensics)이란 디지털 데이터를 통해 디지털 증거의 보존, 수집, 검증, 식별, 분석, 해석, 문서화 및 제시를 위해 과학적으로 도출되고 입증된 방법을 사용하여 사건의 재구성을 촉진하거나 사건의 행위를 예측하는 데 도움을 주는 수사기법이다[1].

안티 포렌식(Anti-Forensics)이란 데이터를 위조, 변조, 은닉, 암호 사용, 파괴, 타임스탬프 조작 등을 통해 디지털 증거의 존재, 양 및 품질을 저하시켜 포렌식 수사를 어렵게 하거나 불가능하게 만드는 행위를 말한다[2].

타임스탬프는 이벤트들을 시간대별로 분석할 수 있는 근거가 되기 때문에 디지털 포렌식 분석에 있어서 가장 기본적이면서도 중요한 정보다. 범죄자는 타임스탬프를 조작하여 포렌식 수사를 어렵게 할 수 있다[3].

본 논문에서는 차량과 스마트폰 간에 블루투스로 연결된 환경에서 각 기기의 시간이 조작된 후 이벤트가 발생되었을 때 각 기기의 로그 데이터는 타임스탬프가 상이하게 기록되어 발생한다. 이와 같은 상황에서 본 논문은 블루투스 HCI 스눕 및 블루투스 로그 분석과 시스템 로그 분석을 통하여, 안티 포렌식 행위인 시스템 시간 조작 여부를 탐지하는 기법을 제안한다.

본 논문의 구성은 다음과 같다. 2장에서는 리눅스 PC와 안드로이드 시스템의 타임스탬프 변경 및 차량에서의 포렌식 관련 연구에 대해 설명하고, 3장에서는 딜러 모드 및 추출하고자 하는 로그 데이터 종류에 대해 설명한다. 4장에서는 '시간 조작 탐지 기법'이라는 방법을 간략하게 제시하며, 5장에서는 시나리오 기반 실험을 통해 생성된 로그 데이터를 수집하는 방법을 제시한다. 6장에서는 수집된 로그에 대해 분석하는 과정을 진행한다. 7장에서는 분석된 로그를 통해 결과를 도출하고, 본 논문에서 발생하는 한계점을 기술한다. 마지막으로 8장에서는 결론과 향후 연구를 기술한다.

2. 관련 연구

이산 등[4]은 리눅스 PC와 안드로이드 기기에서 각각 시간 조작을 하였을 때, 리눅스의 syslog 파일을 분석하여 시간 조작의 흔적을 찾았다. 그 다음, 루팅된 안드로이드 기기에서 logcat 명령어로 로그를 수집하여 분석해 안드로이드 기기의 시간 조작 흔적을 찾아 시간 변경의 시점을 파악하였다.

윤지수 등[5]은 안티 포렌식 기법들과 형사 처벌 방안을 제시하였다. 또한 타임스탬프를 손상시키는 행위는 파일 및 시스템 관리에 문제를 야기하며 포렌식 수사 지연시킬 수 있다고 하였다.

Whelan 등[6]과 Le-Khac 등[7]은 Volkswagen, Dodge Dart 및 Toyota Highlander Limited의 인포테인먼트 시스템을 대상으로 포렌식 분석을 수행했다. 이 연구를 통해 내비게이션 사용 기록, Wi-Fi 및 블루투스 연결 정보, 최근 차량 위치, 트랙 로그(track log), 운전자의 개인 데이터(예: 통화 기록, 연락처 목록, SMS 메시지, 사진, 비

디오)와 같은 다양한 디지털 정보를 추출할 수 있음을 확인했다.

기존 연구에서는 안드로이드 모바일 기기와 리눅스 PC의 타임스탬프를 조작하여 실험을 진행해 로그를 수집한 후 분석하여 시간 조작 증거를 찾았다. 또한 차량 포렌식을 통하여 사용자의 데이터를 추출하였다. 그러나 블루투스 연결 환경에서 시간 조작을 통해 양 기기의 시간 정보가 다를 때 시스템 및 로그에 어떤 영향을 미치는 지에 대한 연구, 그리고 포렌식 수사에서 어떤 시간대로 결정하여 수사를 진행해야 하는지에 대한 연구가 수행된 적이 없다. 따라서 본 논문에서는 블루투스 HCI 스눕 로그 및 블루투스 로그, 시스템 로그 분석 과정을 통해 시간 조작 탐지 기법을 제시한다.

3. 배경 지식

3.1 딜러 모드

딜러 모드란 차량의 AVN 시스템의 히든 메뉴로 특정 방법을 통해 접근 가능하다. 딜러 모드에 접근하여 소프트웨어 업데이트, 차량의 시스템 진단 정보, 그리고 'Copy to USB'를 통해 로그 수집이 가능하다.

3.2 블루투스 로그

블루투스 기능은 서로 다른 종류의 기기여도 통신이 가능하게 해준다. 만약 서로 다른 종류의 기기가 양 기기 간 통신이 필요한 상황이라면 블루투스 기능을 이용해볼 수 있다. 각 기기가 블루투스 기능을 지원한다면 블루투스 기능을 활성화함으로써 서로 다른 기기 간에 통신이 가능해진다[8]. '블루투스 로그'란 블루투스 통신을 한 기기 간에 블루투스 통신과 관련된 다양한 이벤트와 상태 변화를 기록한 로그 파일이다.

3.3 시스템 로그

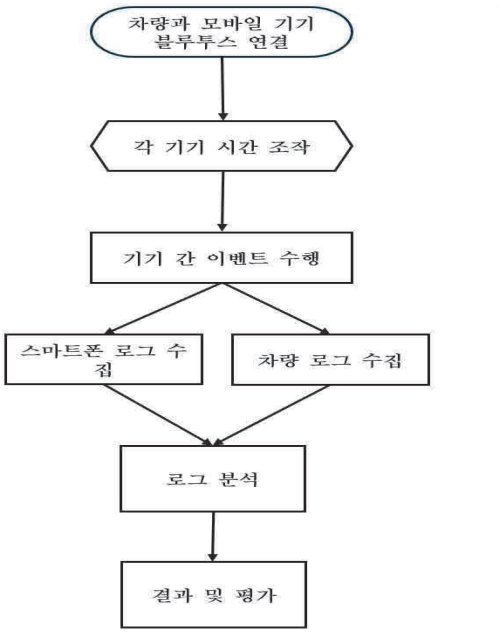
시스템 로그는 운영체제와 어플리케이션의 동작을 기록한 파일로써, 시스템의 상태를 모니터링하고 문제를 디버깅하는데 중요한 정보를 제공한다. 시스템 로그는 다양한 이벤트, 오류, 경고, 정보 메시지를 포함해 기기의 전반적인 활동을 기록한다[9].

3.4 블루투스 HCI 스눕 로그

블루투스 HCI 스눕 로그는 블루투스 통신을 모니터링하고 디버깅하기 위해 생성되는 로그다. 해당 로그는 블루투스 프로필을 통하여 로그의 특성을 보여준다[10]. 또한 HCI 이벤트 메시지를 통하여 현재 블루투스 연결된 기기들의 상태를 분석 가능하게 해준다[11]. 블루투스 HCI 스눕 로그를 사용하기 위해서는 Android 4.4(KitKat) 이상의 기기에서 수동으로 개발자 옵션을 통한 활성화를 해야 한다. 이를 통해 블루투스 호스트와 컨트롤러 간의 모든 HCI 로그를 캡처하여 블루투스 연결 상태 및 데이터 전송 상태를 분석할 수 있다.

4. 시간 조작 탐지 기법

본 논문에서 제안하는 시간 조작 탐지 기법에 대한 프로세스가 그림 1에 나타나 있다.



[그림 1]. 시간 조작 탐지 기법의 프로세스

먼저 기기 간 블루투스 연결을 진행을 한다. 이후, 각 기기의 시간을 조작한다. 시간 조작 같은 경우 표 1과 같이 여러 가지 경우의 수가 존재한다.

[표 1]. 시간 조작 경우의 수

기기	차량 AVN	모바일 기기
시간 조작	과거	과거
		현재
		미래
	현재	과거
		현재
		미래
	미래	과거
		현재
		미래

시간 조작 이후, 각 기기 간 이벤트를 수행한다. 블루투스 연결 환경에서 가능한 이벤트의 종류는 표 2와 같다.

[표 2]. 블루투스 연결 환경 이벤트 목록

수행 가능한 이벤트 종류
통화 이벤트
음악 이벤트
내비게이션 이벤트
메시지 이벤트

이벤트 수행 이후, 각 기기의 로그를 수집한다. 스마트폰의 경우, ADB(Android Debug Bridge) 환경에서 로그를 수집한다. 로그를 추출하는 명령어는 표 3과 같다.

[표 3]. ADB 로그 추출 명령어

명령어
logcat
bugreport

logcat의 명령어는 휘발성 로그에 대해 로그 추출이 가능하다. 만약 모바일 기기의 전원이 꺼진다면 로그가 모두 사라진다. bugreport 명령어의 경우 logcat의 로그 데이터를 포함하여 시스템의 전체적인 로그 데이터를 추출 가능하므로 bugreport를 통하여 모바일 기기의 로그 데이터를 추출한다[12]. 차량의 경우 로그를 추출할 수 있는 방식은 표 4와 같다.

[표 4]. 차량 로그 추출 방법

로그 추출 방법
Chip-off
Copy image to USB

Chip-off 방식의 경우 디지털 장치에서 메모리칩을 물리적으로 제거하여 로그 데이터를 추출하는 방식이다. 이 경우 메모리칩 손상 가능성이 있기에 Chip-off 방식은 본 논문에서는 사용하지 않는다. 따라서 차량 AVN의 딜러 모드에 접근하여 Copy image to USB를 통하여 로그 데이터를 추출한다.

로그를 수집한 후 로그 분석을 진행한다. 생성된 로그 파일에서 '수정한 날짜' 태그를 확인하여 해당 실험 날짜 타임스탬프로 생성된 로그 파일 위주로 분석한다. 또한 기기 간에 블루투스 연결을 통한 통신을 하였기에 블루투스 로그 역시 분석한다. 이후, 시간 조작 관련 키워드를 이용해 분석을 한다. 시간 관련 키워드는 표 5와 같다. 만약 시간 조작 관련 로그 메시지가 존재하지 않는다면 변경된 타임스탬프 전후로 로그 데이터를 분석하도록 한다. 마지막으로 분석 결과를 평가한다.

[표 5]. 시간 관련 키워드

시간 관련 키워드	
TimeChange	DateSet
TimeSet	UsetTime
DateChange	UsetDate

5. 실험 환경 및 실험 방법

5.1 실험 대상 시스템

본 논문의 실험 대상 시스템은 국내 차량에 탑재되는 Android 4.4.2(KitKat) 버전의 AVN과 Android 14버전을 탑재한 Galaxy S21이다. 본 논문에서 사용한 AVN 시스템과 모바일 기기의 사양은 표 6과 같다.

[표 6]. 대상 AVN 시스템 및 모바일 기기

AVN	
차종	Hyundai Avante
AVN 제조사	Hyundai Mobis
운영체제	Android 4.4.2 (Kitkat)
커널 버전	Linux 3.18.24-tcc
모바일 기기	
기기명	Galaxy S21
제조사	Samsung
운영체제	Android 14
커널 버전	Linux 5.4.242-27760517-abG991NKSU4FWK7

AVN 내부에서 운전자 행위와 관련된 로그 데이터를 생성하기 위해서 시나리오 기반의 실험을 했다. 주행 중 운전자가 수행할 수 있는 이벤트를 선별하고 표 7과 같은 시나리오를 작성했다.

[표 7]. 주요 이벤트 시나리오

시간	시나리오
17:13	대상 시스템 블루투스 연결
17:33	차량 시간 조작
17:33	스마트폰 시간 조작
17:34	전화 발신
17:34	전화 수신
17:34	음악 재생
17:35	차량 로그 덤프
17:59	스마트폰 로그 추출

차량 시간 조작 당시 시스템 시간은 2024.07.19.17:33이었으며, 스마트폰 시간 조작 당시 시스템 시간은 2024.07.19.17:33이었다.

만약 안티 포렌식 행위를 수행하는 범죄자가 날짜 및 시간을 조작한다면 미래 시간보다는 과거로 조작하여 포렌식 행위에 혼란을 줄 수 있다. 예를 들어, 사건 발생 시간대에 본인은 그 시간대에 다른 행위를 하고 있었다고 주장할 수 있다. 따라서 표 8과 같이 두 기기 모두 과거로 조작하되 서로 다른 과거로 조작하여 혼란을 증가시켰다.

[표 8]. 날짜 및 시간 조작 시나리오

	Hyundai Avante	Samsung Galaxy S21
현재 시간	2024.07.19.17:33	2024.07.19.17:33
변경 시간	2024.07.11.14:00	2024.07.13.12:00

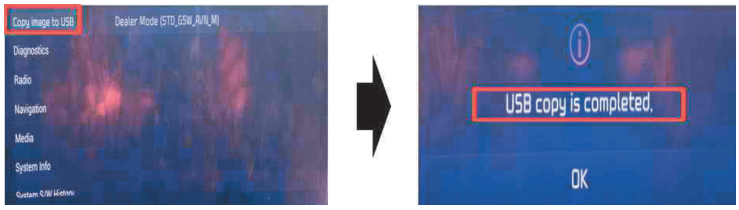
5.2 로그 수집

시나리오 기반으로 실험을 진행한 후 로그 데이터를 수집하였다. 먼저, 스마트폰의 경우 4장에서 언급한 방식으로 bugreport 명령어를 이용하여 로그 데이터를 수집

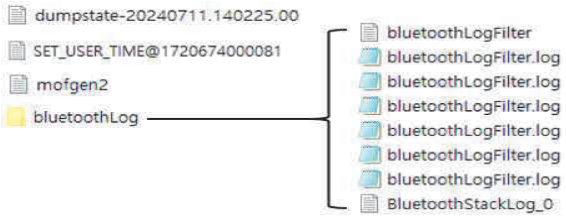
한다. 로그 데이터를 추출한 후 분석 대상 로그 파일은 그림 2와 같다.

[그림 2]. 추출된 스마트폰의 로그 중 분석 대상 파일

차량의 로그는 그림 3과 같이 '딜러 모드'에 접근 후 'Copy image to USB' 버튼을 클릭하여 로그 덤프를 하였다. 로그 데이터를 추출한 후 분석 대상 로그 파일은 그림 4와 같다.



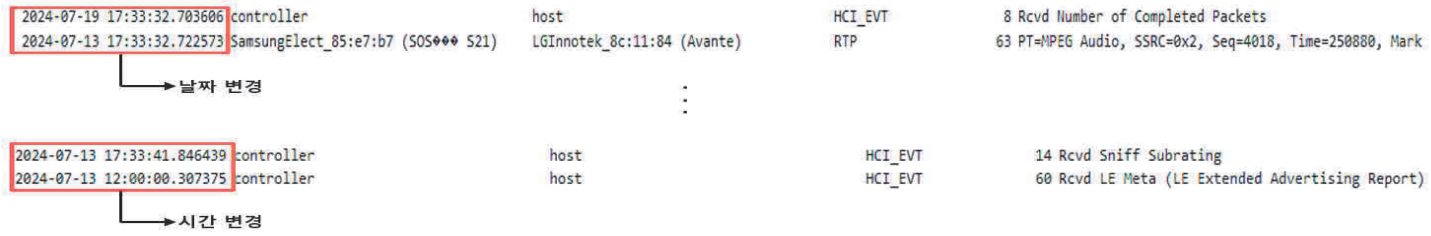
[그림 3]. 차량의 로그 덤프 과정



[그림 4]. 추출된 차량의 로그

6. 로그 분석

본 장에서는 5장에서 획득한 로그 데이터를 분석한다. 수집한 로그 데이터는 작성한 시나리오를 기반으로 시간 조작 후 수행한 이벤트 위주로 분석하였다. 블루투스 패킷의 경우 와이어샤크로 분석을 하였고, 그 외 로그 데이터는 메모장, VSCode를 이용하여 분석하였다.



[그림 5]. 스마트폰의 변경된 타임스탬프

[그림 6]. 스마트폰의 변경된 시간 정보에서의 이벤트

6.1 스마트폰 블루투스 HCI 스눕 로그 분석

스마트폰 블루투스 HCI 스눕 로그를 분석한 결과, 시스템 시간을 기준으로 로그가 생성되다가 스마트폰 시간을 조작한 시점에서 그림 5와 같이 타임스탬프가 변경되어 로그가 발생했다.

이후, 그림 6과 같이 시나리오에서 작성한 이벤트들이 발생된 로그들을 발견할 수 있었다. 따라서 시간 조작된 상태에서 타임스탬프는 변경된 시간을 기준으로 로그가 생성되고, 변경된 타임스탬프에서 이벤트가 수행되었음을 알 수 있었다.

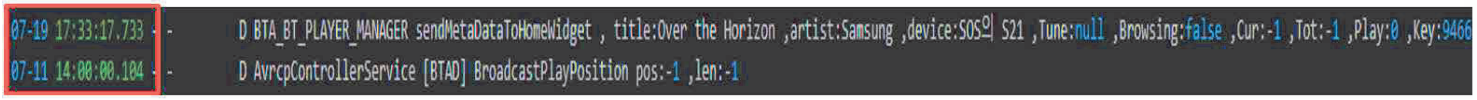
6.2 차량 블루투스 로그 분석

차량의 블루투스 로그를 분석한 결과 스마트폰 블루투스 HCI 스눕 로그 분석 결과와 유사하게 차량의 시스템 시간을 따르는 로그들이 생성되다가 그림 7과 같이 시간 조작이 수행된 시점에 타임스탬프가 변경되어 로그가 생성되었다. 이후 그림 8과 같이 변경된 타임스탬프에서 통화 이벤트, 음악 이벤트가 수행된 로그를 확인할 수 있었다.

6.1절에서와 같이 스마트폰 블루투스 HCI 스눕 로그 분석한 결과와 차량 블루투스 로그 분석한 결과를 정리해보았을 때, 타임스탬프가 변경되었음을 통해 시간 조작에 대한 안티 포렌식 행위를 의심해볼 수 있었다.

또한 변경된 타임스탬프에서 이벤트에 대한 로그들이 발생되는 것을 알 수 있었다. 그러나 타임스탬프가 변경되었음을 알 수 있음을 제외하고는 시간 조작에 대한 증거를 블루투스 로그에서 발견할 수 없었다.

따라서 시스템 로그 분석을 통해 시간 정보 조작에 대한 안티 포렌식 행위를 찾는 과정이 필요하다.



[그림 7]. 차량의 변경된 타임스탬프

[그림 8]. 차량의 변경된 타임스탬프에서의 이벤트

6.3 스마트폰 시스템 로그 분석

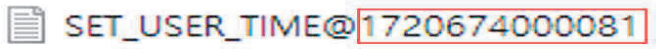
스마트폰 시스템 로그 분석은 'dumpstate-2024-07-19-17-59-48.txt'파일을 통해 분석 가능하다. 해당 파일의 로그를 분석해본 결과 'data/misc/wifi_hostapd/dataUsage_log.txt' 경로에 그림 9와 같은 로그들이 존재하였다.

먼저 '7-19 17:32:30'에서 '7-13 17:33:32'로 날짜 변경이 먼저 이루어졌고, 이후 '07-13 12:00:00'로 시간 변경이 이루어졌다. 또한 'date changed', 'User Change Time to yyyy-MM-dd HHmmss' 메시지가 존재하였다. 따라서 해당 로그를 통해 안티 포렌식 행위가 수행되었음을 알 수 있었다.

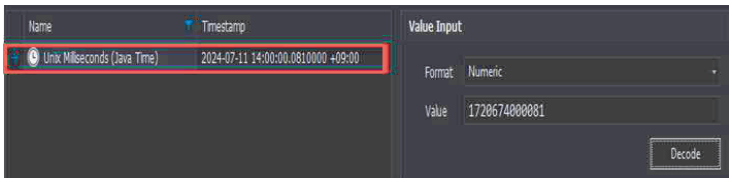
[그림 9]. 시간 조작 시스템 로그

6.4 차량 시스템 로그 분석

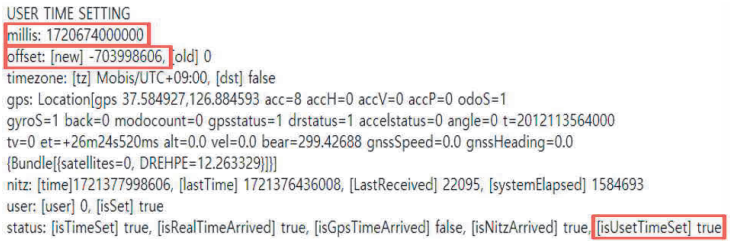
차량의 시간을 조작한다면 그림 10과 같이 'SET_USER_TIME@Unix Epoch Time.txt' 파일이 생성된다. @뒤에 있는 숫자는 변경된 시간을 Unix Epoch Time으로 표현된 것이다. 따라서 이에 대해 해당 Unix Epoch Time를 'DCode v5.6' 프로그램을 이용하여 시간 변환하면 그림 11과 같이 표현됨을 통해 조작된 시간임을 알 수 있다. 해당 텍스트 파일을 메모장을 통해 열어보면 그림 12와 같은 화면을 볼 수 있다.



[그림 10]. 'SET_USER_TIME' 파일



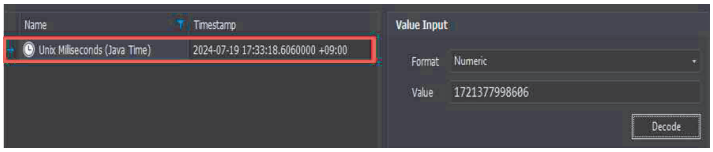
[그림 11]. Unix Epoch Time 시간 변환



[그림 12]. SET_USER_TIME@Unix Epoch Time 파일

해당 파일의 내용을 분석해보면 'status' 항목의 'isUsetTimeSet' 부분이 true로 설정되어있다. 이를 통해 사용자가 설정한 시간임을 알 수 있고, 'offset'항목의 'new'부분이 -703998606로 설정되어있다. 이를 통해 설정된 시간이 기존 시간에서 얼마만큼 차이가 났는지를 알 수 있다.

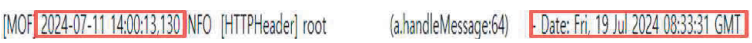
설정된 Unix Epoch Time과 -703998606를 계산하여 시간 변환을 진행하면 그림 13과 같은 결과를 통해 시간이 조작되었다고 알 수 있다.



[그림 13]. Unix Epoch Time 시간 변환

차량의 'SET_USER_TIME@Unix Epoch Time.txt' 파일 분석을 통해 시간이 조작되었음을 알 수 있었다. 또한 이전의 시간은 어떤 시간이었는지 어느 시간으로 시간 조작을 시도했는지 알 수 있었다. 그러나 이전의 시간 역시 조작된 시간일 수 있기에 진짜 시간을 찾는 과정이 필요하다. 이를 위해 'mofgen2.txt' 파일을 분석하였다.

해당 파일을 열어서 분석을 해본 결과, 그림 14와 같은 로그 메시지를 확인할 수 있었다.



[그림 14]. mofgen2.txt 파일의 로그 메시지

해당 파일을 분석해보면 '2024-07-11-14:00:13' 으로 조작된 시간이 타임스탬프로 존재한다. 그리고 로그 메시지는 'Date Fri, 19 Jul 2024 08:33:31 GMT'로 그리니치 시간대로 존재한다. 그리니치 시간대로 존재하기에 오른쪽 메시지는 시스템 시간인 것을 알 수 있다. 따라서 한

국의 시간이 그리니치 시간대를 기준으로 9시간 빠르기에 +9를 해주면 'Date Fri, 19 Jul 2024 17:33:31'로 정확한 시간을 알 수 있다.

7. 논의 및 한계점

본 논문에서는 Hyundai Avante AVN 시스템의 로그 데이터를 '딜러 모드'에 접속하여 'Copy image to USB'를 통해 로그 데이터를 획득했다. 또한 Samsung Galaxy S21, Android 14 모바일 기기를 'bugreport' 명령을 통해서 로그 데이터 및 패킷을 획득할 수 있었다. 6절에서는 획득한 로그 데이터 및 패킷을 분석을 진행하였다.

먼저, 스마트폰 블루투스 HCI 스눕 로그 및 차량의 블루투스 로그에서는 타임스탬프가 변경된 로그를 통해 시간 조작이 이루어졌단 걸 의심해볼 수 있었다. 그러나 사용자가 안티 포렌식 행위를 했다는 메시지는 발견할 수 없었다.

따라서 안티 포렌식에 대한 명확한 증거를 찾기 위해 스마트폰 시스템 로그와 차량의 시스템 로그를 분석해본 결과 시간 조작이 이루어졌다는 걸 알 수 있었다. 또한 기존의 시스템 시간이 무엇인지 알 수 있었다. 분석된 결과를 정리하면 표 9와 같다.

본 논문에서는 '블루투스 HCI 스눕 로그' 기능을 활성화를 하여 블루투스 로그 수집하였다. 그러나 해당 기능은 기본적으로 활성화 되어있는 기능이 아니기 때문에 수동으로 ON/OFF를 해줘야한다. 만약 해당 기능이 OFF 상태라면 블루투스 로그가 생성되지 않기 때문에 블루투스 로그 수집할 수 없다는 한계점이 존재한다.

[표 9]. 각 로그 별 아티팩트 확인 가능 여부

	블루투스 로그	시스템 로그
타임스탬프 변경	O	O
시간 정보 조작	X	O
기존의 시스템 시간	X	O

8. 결론 및 향후 연구

본 논문에서는 차량 및 모바일 기기 간 블루투스 연결이 된 환경에서 각 기기의 시간 정보가 조작되었고, 시간 정보가 조작된 상황에서 이벤트가 발생했을 때 시간 정보 조작을 탐지할 수 있는 기법을 제시했다. 먼저 블루투스 로그를 수집 후 분석하여 시간 조작을 의심하고, 시스템 로그를 분석하여 시간 조작에 대한 아티팩트를 찾음으로써 시간 조작 행위가 이루어졌다는 걸 알 수 있었다.

마지막으로 시스템 로그를 추가 분석하여 정확한 시간을 찾을 수 있었다. 범죄자가 시간을 조작한 후 이벤트를 수행하여도 위와 같은 과정을 통해 정확한 시스템 시간을 알아내어 사건에 대한 조사를 효율적으로 진행할 수 있을 것이다.

단, '블루투스 HCI 스눕 로그' 기능은 기본적으로 활성화 되어 있는 기능이 아니기에 향후 해당 기능이 비활성화 되어 있는 환경에서도 블루투스 로그를 수집할 수 있는 방법을 연구하고, 본 논문에서는 하나의 시나리오만으로 실험을 진행하였지만 여러 시나리오를 두어 실험을 함으로써 해당 기법에 대해 일반화를 할 계획이다.

ACKNOWLEDGEMENT

이 연구는 2021년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(no. 2021R1A2C2012574), 또한 2022년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임(No.2022-0-01022, 이벤트 기반 실험 시스템 구축을 통한 자동차 내·외부 아티팩트 수집 및 통합 분석 기술 개발).

참고 문헌

[1] John Wiley & Sons Inc, Hoboken, NJ. "Digital Forensics: An Academic Introduction." 2018.

[2] Gary C. Kessler. "Anti-Forensics and the Digital Investigator." 5th Australian Digital Forensics Conference, 3, 12, 2007.

[3] Cho, Gyu-Sang. "Digital Forensic Analysis of Timestamp Change Tools: An Anti-Forensics Perspective." Korean Society of Computer Information Conference, 07a, Pages.391-392, 2019.

[4] 이산, 정지헌, 안석현, 조성제. "리눅스와 안드로이드에서 시간 정보 조작이 로깅에 미치는 영향 분석." 한국차세대컴퓨팅학회, 2024.

[5] 윤지수, 이경렬. "안티 포렌식 신종기법에 대한 형사법적 대응방안." 한국형사정책학회 논문지, 32(4), 65-99. 2021.

[6] Whelan, C. J., Sammons, J., McManus, B., and Fenger, T. W., "Retrieval of infotainment system artifacts from vehicles using iVe," Journal of Applied Digital Evidence, 1(1), 30, 2018.

[7] Le-Khac, N.-A., Jacobs, D., Nijhoff, J., Bertens, K., and Choo, K.-K. R.. "Smart vehicle forensics: Challenges and case study." Future Generation Computer Systems, 109: 500-510, 2020.

[8] R. Nusser and R. M. Pelz, "Bluetooth-based wireless connectivity in an automotive environment," Vehicular Technology Conference Fall 2000. IEEE VTS Fall VTC2000. 52nd Vehicular Technology Conference (Cat. No. 00CH37152), Vol. 4, pp.1935-1942, 2000.

[9] 강해인, 박민수, 조성제, 정지헌. "차량 디지털 포렌식에서 타임라인 분석을 위한 안드로이드 기반 오디오 비디오 내비게이션 시스템의 로그 활용." 한국정보과학회 2023 한국컴퓨터종합학술대회 논문집, 1,259-1,261. 2023

[10] https://source.android.com/docs/core/connect/bluetooth/verifying_debugging?hl=ko

[11] https://source.android.com/docs/core/connect/bluetooth/hci_requirements?hl=ko

[12] https://source.android.com/docs/core/tests/debug/read-bug-reports?hl=ko#event-log

리눅스와 안드로이드 시스템에서 타임스탬프 조작 식별을 위한 로그 분석 기법

Log Analysis Techniques for Identifying Timestamp Manipulation in Linux and Android Systems

이산, 조민혁, 정지헌, 조성제[1)]
San Lee, Min Hyuk Cho, Jiheun Jung, Seong-je Cho

(16890) 경기도 용인시 수지구 죽전로 152 단국대학교
산업보안학과, 모바일시스템공학과, 컴퓨터학과, 소프트웨어학과
{dltkshdkd915, cgumgek8, wlgjsjames7224, sjcho}@dankook.ac.kr

요 약

디지털 포렌식 분석에서 안티-포렌식 기법에 대응해 디지털 증거의 무결성을 유지하는 것이 중요하다. 타임스탬프가 변조되면 증거 신뢰성이 저하되고 사건 타임라인 구성이 어려워진다. 본 논문은 리눅스와 안드로이드 시스템에서 타임스탬프 변조가 의심될 때 로그 데이터를 수집·분석해 이를 탐지하는 효과적인 기법을 제안한다. 재부팅 시점에서 타임스탬프 변조 여부를 확인하는 새로운 방법도 제시하여 각 운영체제별 타임스탬프 조작 시점을 효과적으로 파악한다. Ubuntu와 CentOS 리눅스, 안드로이드 9와 14를 대상으로 실험하여 제안 기법의 유효성을 검증하였다. 본 기법은 여러 운영체제에서 발생할 수 있는 타임스탬프 조작 공격에 대응하고 디지털 증거의 정확성과 신뢰성을 높이는 데 기여할 수 있다.

Abstract

In digital forensic analysis, maintaining the integrity of digital evidence against anti-forensic techniques is crucial. If timestamps are tampered with, the reliability of the evidence diminishes, complicating the construction of an accurate incident timeline. This paper proposes an effective technique to detect timestamp manipulation by collecting and analyzing log data in cases of suspected tampering in Linux and Android systems. Additionally, we present a new method to verify timestamp manipulation at reboot, identifying tampering times for each operating system. Experiments on Ubuntu and CentOS Linux desktops, as well as Android 9 and 14 phones, demonstrate the technique's validity. This proposed method enhances responses to timestamp manipulation across various OSes, improving the accuracy and reliability of digital evidence.

1) 교신 저자

논문투고일자 : 2024. 08. 02 심사일자 : 2024. 08. 05 게재확정일자 : 2024. 08. 23 http://doi.org/10.23019/kingpc.20.4.202408.003

키워드: 디지털 포렌식, 안티 포렌식, 타임스탬프 조작, 로그 분석, 리눅스, 안드로이드
Keyword: Digital forensics, Anti forensics, Timestamp manipulation, Log analysis, Linux, Android

1. 서론

디지털 포렌식은 법적 수사 과정에서 디지털 기기에 저장된 데이터를 수집, 복구, 분석 및 보고하는 과학적 절차로, 법집행기관이 컴퓨터 기기를 압수 및 수색하여 잠재적 증거를 확보하는 중요한 역할을 한다[1][2]. 디지털 포렌식의 목표는 증거의 무결성과 신뢰성을 유지하면서 정확한 정보를 제공하는 것이다. 그러나 디지털 포렌식 분석을 방해하는 다양한 안티 포렌식(anti-forensic) 기법들이 존재한다. 잘 알려진 안티 포렌식 기법으로, 데이터를 암호화하거나 삭제하고, 타임스탬프(timestamp)를 조작하여 증거의 신뢰성을 손상시키는 행위 등이 있다[3]. 특히 타임스탬프 조작은 컴퓨팅 시스템의 시간을 왜곡하여 실제 사건 발생 시점을 혼란스럽게 만들어 포렌식 분석을 어렵게 한다[4].

기존 연구들은 다양한 운영체제와 환경에서의 디지털 포렌식과 안티 포렌식 기법에 대해 다루어 왔다. 예를 들어, 리눅스 기반 시스템에서의 타임스탬프 조작과 그에 따른 로그 파일 분석 방법에 대한 연구들이 진행되었으며[5], 안드로이드 시스템에서의 포렌식 분석 기법도 연구되어 왔다[6]. 본 연구진은 이전에 Ubuntu 리눅스 시스템과 안드로이드 9 시스템에서, 타임스탬프 조작이 디지털 포렌식에 미치는 영향을 분석했다[7].

본 논문은 기존 연구[7]를 확장한 버전이다. 즉, [7]에서 실험한 시스템들 외에, 추가로 CentOS 리눅스를 탑재한 데스크톱 컴퓨터와 안드로이드 14를 탑재한 스마트폰에서 타임스탬프 조작이 로깅 시스템에 미치는 영향을 분석하고, 타임스탬프 조작 여부를 탐지하는 기법을 제안한다. 특히, 안드로이드 기반 스마트폰의 재부팅 시점에서 시간 조작 여부를 확인할 수 있는 새로운 아티팩트(artifact)를 발견하고 이를 분석하였다. 이 새로운 아티팩트는 로그 데이터를 통해 시간 조작 시점을 보다 정확하게 파악할 수 있게 해 준다.

본 논문에서 제시한 기법은 디지털 포렌식 분석의 신뢰성을 높이고, 다양한 운영체제에서 발생할 수 있는 시간 조작 공격에 대한 대응력을 강화하는 데 기여할 수 있다. 제안 기법은 타임스탬프가 조작된 경우에도 디지털 포렌식 수사가 효과적으로 수행될 수 있게 하며, 이를 통해 디지털 증거 분석의 정확성과 신뢰성을 한층 높일 수 있을 것이다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구에 대해 설명하고, 3장에서는 모델 구성에 대해 기술한다. 4장에서는 실험 결과를 보이면서 제안 기법을 평가하고, 5장에서는 결론을 맺는다.

2. 관련 연구

Singh et al. [8]은 WSL(Windows Subsystem for Linux) 환경에서 NTFS 파일을 수정, 접근 등을 수행하는 다양한 유틸리티를 적용할 때의 타임스탬프 패턴을 분석하여 타임스탬프 위조를 감지하는 방법을 제시하였다. 파일 생성, 접근 수정, 이름 변경 및 복사에 대한 타임스탬프 규칙과 NTFS와 Ext4 간 파일 전송, 그리고 타임스탬핑 도구의 분석을 통해 타임스탬프의 변화를 체계적으로 연구하였다.

Song et al. [9]은 NTFS에서 저장장치의 성능을 활용하여 타임스탬프의 변조를 확인하는 기법을 설계하였다. 파일의 변경된 타임스탬프 시간 차이와 파일크기를 통해 초당 기록된 데이터양과 저장장치가 가진 최대 읽기, 쓰기 속도와 비교하여 타임스탬프의 변조 여부를 탐지하였다.

Gübel et al. [10]은 ext4 파일시스템에서 타임스탬프를 이용한 스테가노그래피(steganography) 채널로 데이터를 숨기는 방법을 제시하였으며 해당 기법의 비밀성과 사용성을 평가하였다. 이들은 Ext4 파일 시스템의 타임스탬프 구조와 사용법을

분석하여 데이터 은닉 가능성을 확인하였고 데이터 숨김 과정에서 암호화와 오류 수정 코드를 포함한 데이터 삽입 단계와 숨겨진 데이터를 복원하는 방법에 대해 설명하였다.

Oh et al. [11]은 기존의 타임스탬프 조작 탐지 방법의 특징과 한계를 비교 분석하고 새로운 탐지 알고리즘을 제안하였다. 제안된 탐지 알고리즘은 기존 NTFS 저널 기반 방법의 한계를 개선하여 탐지 정확도를 높였고, 실제 APT(Advanced Persistent Threat)공격에서 타임스탬프 조작을 탐지하는 사례를 보여주었다.

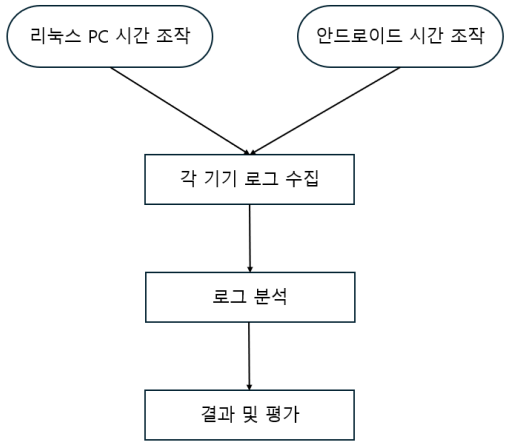
이상현. [12]은 IoS 운영체제를 기반으로 하는 디바이스 내에서 타임스탬프 위 · 변조 여부를 이벤트를 발생시켜 아티팩트 흔적이 남는 SMS, CallHistory, Photos.sqlite등을 확인하여 조작된 시스템 시간을 탐지하는 방법을 제안하였다.

기존 연구들[8, 9, 10, 11]에서는 파일의 메타 데이터인 타임스탬프의 조작을 탐지하였다. 기존 연구는 다양한 유틸리티를 통해 타임스탬프 변조 여부를 파악하거나 또는 저장장치의 성능을 분석하여 타임스탬프 조작 여부를 탐지하였다. 이러한 기존 연구는, 공격자에 의해 시스템의 시간이 조작된 이후에 생성된 파일과 파일 타임스탬프를 탐지하기 어렵다는 한계가 있다. 또한 기존 연구[7]에서는 안드로이드 9와 Ubuntu 리눅스를 대상으로 실험을 진행하였고, 본 논문에서는 [7]에서 실험한 시스템 뿐만 아니라 CentOS 리눅스와 안드로이드 14도 대상 시스템에 포함하여 실험한다.

본 논문에서는 파일의 메타 데이터 분석이 아닌 시스템의 시간 자체가 조작되었는지를 탐지하는 기법을 제안한다. 제안한 기법은 시스템의 시간 조작 이후에 생성된 파일 타임스탬프도 판별할 수 있다.

3. 타임스탬프 조작 및 로그 기반 탐지 기법

본 논문은 컴퓨팅 시스템의 타임스탬프가 조작된 경우에, 조작된 타임스탬프를 효과적으로 탐지하는 기법을 제안한다. 제안한 기법에 대한 전반적인 절차가 (그림 1)에 나타나 있다.



(그림 1) 타임스탬프 조작 및 탐지 프로세스

네트워크에 연결된 컴퓨팅 기기는 네트워크 시간을 동기화해서 타임스탬프를 관리한다. 본 논문에서는 제안 기법을 효율적으로 검증하기 위해, 네트워크 동기화를 일시적으로 해제하여 각 컴퓨팅 기기의 타임스탬프(시간)를 조작한다. 컴퓨팅 기기의 시간을 조작할 수 있는 경우의 수는 다양하며, 본 논문에서 고려하고 있는 타임스탬프 조작의 예가 <표 1>에 나타나 있다.

〈표 1〉 타임스탬프 조작 경우의 수

기기	리눅스		안드로이드	
시간 조작	현재	과거	현재	과거
		현재		현재
		미래		미래

타임스탬프 조작이 의심되는 경우, 타임스탬프 조작 여부를 판단하기 위해 시스템 로그를 수집하여 분석한다. 본 논문에서 고려하고 있는 시스템은 리눅스와 안드로이드이므로, 이들 로깅 기법을 분석한다.

리눅스의 경우 PC에서 /var/log 하위 디렉터리에 다양한 로그들이 생성된다. 그중 시스템에 직접

적인 이벤트 관련 로그가 저장된 /syslog를 수집한다. 특히 네트워크 시간 동기화와 관련 있는 NTP (Network Time Protocol) 위주로 로그를 수집한다. 안드로이드의 경우에는 ADB (Android Debug Bridge)를 통하여 로그를 수집한다. 로그 추출 명령어는 <표 2>와 같다.

〈표 2〉 안드로이드 로그 추출 명령어

명령어
logcat
bugreport

로그를 수집한 후 수집한 로그 데이터를 분석한다. 리눅스 기반 PC의 경우, /var/log/syslog 아래에 생성된 로그를 통해 특정 메시지를 담고 있는 로그와 시스템 시간이 변경되는 로그 위주로 분석한다. 안드로이드 스마트폰의 경우, 먼저, logcat을 통해 시간 조작 관련 로그 메시지를 확인한다. 기존 연구[7]에서는 logcat을 통해 시간 조작 시점을 확인 가능하였으나 휘발성 로그이기 때문에 재부팅시 확인할 수 없다는 한계점이 존재했다. 본 논문에서는 bugreport를 이용하여 추가로 분석한다. 마지막 단계에서는, 분석된 결과를 바탕으로 시스템의 타임스탬프가 조작되었는지 여부를 판단한다.

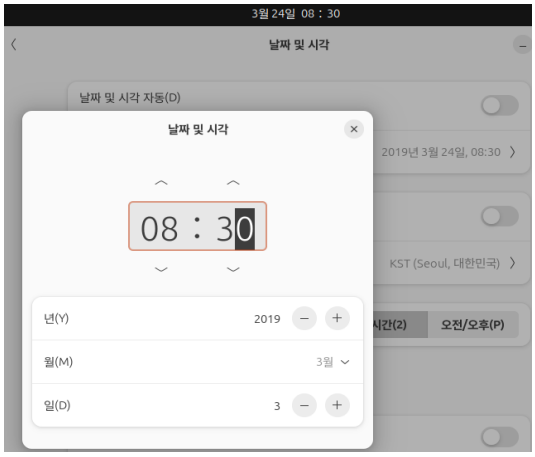
4. 실험 방법

4.1 대상 시스템

본 절에서는 데스크톱 리눅스(Ubuntu 24.04, CentOS Stream 9)와 안드로이드 스마트폰(안드로이드 버전 9, 14)을 대상으로 시간 정보를 변경한 후 로그 분석을 통해 시간 변경이 이루어진 시점을 파악한다.

4.2 PC용 리눅스

리눅스의 경우 설정 - 날짜 및 시각 자동(D) – 비활성화를 하고 (그림 2)와 같이 시간을 조작한다.



(그림 2) 네트워크 동기화 비활성화

4.2.1 Ubuntu 24.04

Ubuntu 24.04에 대해 <표 3>의 상태로 설정하여 시스템 시간을 조작한 후 /var/log/syslog 에 생성된 로그를 수집한다.

〈표 3〉 리눅스(Ubuntu) 실험 설정

Ubuntu 24.04 실험		
시간	과거	미래
실험 일시	2024.07.24, 15:18	2024.07.24, 15:29
변경 일시	2019.03.03, 08:30	2025.02.24, 08:30
실험 방법	시간 조작 후 syslog 확인	

터미널에서 date 명령어를 통해 변경된 날짜와 시간 정보를 확인할 수 있다(<표 4> 참조).

〈표 4〉 터미널에서 date 명령어로 확인

```
root@san-san2:/home/san# date
2019.03.24.(일) 08:31:15 KST
```

대부분의 로그 메시지는 syslog 파일에 저장된다. <표 5>와 같이 cat 명령어를 통해 syslog를 분석한 결과, 2024-07-24 15:18에 NTP(Network Time Protocol)가 비활성화된 메시지를 발견할 수

있었다.

〈표 5〉 /var/log/syslog 중 NTP 비활성화 내역

2024-07-24T15:18:55.372520+09:00 san-san2 systemd-timedated[3013] : **Set NTP to disabled.**

이후 생성된 로그 메시지는 "Change local time to-"라는 메시지가 생성되며, 바뀐 시간대로 로그가 저장되는 것을 확인할 수 있다. 로그 분석 결과, 네트워크 동기화를 비활성화한 시점인 NTP 비활성화 로그 메시지가 생성된 2024-07-24 15:18에 시간을 변경하였다는 것을 알 수 있다(<표 6> 참조).

〈표 6〉 syslog 중 시간 조작 내역

2019-07-23T08:30:48.002864+09:00 san-san2 systemd-resolved[565] : **Clock change detected.** Flushing caches. 2019-07-23T08.30:48.122107+09:00 san-san2 systemd-timedated[3013] : **Changed local time to** Tue 2019-07-23 08:30:48 KST

미래시간은 <표 7>의 date 명령어를 통해 변경된 날짜와 시간 정보를 확인할 수 있다.

〈표 7〉 터미널에서 date 명령어로 확인

root@san-san2:/home/san/바탕화면# **date** 2025. 02. 24. (월) 08:30:37 KST

<표 8>의 cat 명령어를 통해 syslog를 분석한 결과, 2024-07-24 15:29에 NTP가 비활성화된 메시지를 발견할 수 있었다.

〈표 8〉 /var/log/syslog 중 NTP 비활성화 내역

2024-07-24T15:29:26.589297+09:00 san-san2 systemd-timedated[3214] : **Set NTP to disabled.**

<표 9>를 통해 이후 생성된 로그 메시지는 "Change local time to-"라는 메시지가 생성되며, 바뀐 시간대로 로그가 저장되는 것을 확인할 수 있다. 로그 분석 결과, 네트워크 동기화를 비활성화한 시점인 NTP 비활성화 로그 메시지가 생성된 2024-07-24 15:29에 시간을 변경하였다는 것을 알 수 있다.

〈표 9〉 syslog 중 시간 조작 내역

2025-07-24T08:30:23.004872+09:00 san-san2 systemd-resolved[541] : **Clock change detected.** Flushing caches. 2025-07-24T08:30:23.306704+09:00 san-san2 systemd-timedated[3214] : **Changed local time to** Thu 2025-07-24 08:30:23 KST

4.2.2 CentOS stream 9

CentOS stream 9에 대해 <표 10>의 정보로 설정하여 시스템 시간을 조작한 후, /var/log/syslog에 생성된 로그를 수집한다.

〈표 10〉 리눅스(CentOS) 실험 설정

CentOS stream 9 실험		
시간	과거	미래
실험 일시	2024.06.24, 16:57	2024.06.24, 16:36
변경 일시	2019.03.03, 08:33	2025.02.01, 05:33
실험 방법	시간 조작 후 syslog 확인	

과거 시간은 <표 11>의 date 명령어를 통해 변경된 날짜와 시간 정보를 확인할 수 있다.

〈표 11〉 터미널에서 date 명령어로 확인

[leesan@localhost ~]$ **date** 2019.03.03.(일) 08:33:24 KST

<표 12>를 통해 6월 24일 16:57에 NTP가 비활성화되었다는 로그 메시지가 생성되었고, 이후

"Changed local time to-"라는 로그 메시지가 순차적으로 생성되는 것을 확인할 수 있다. 로그 분석 결과, 네트워크 동기화를 비활성화한 시점인 NTP 비활성화 로그 메시지가 생성된 2024-06-24 16:57에 시간을 변경하였다는 것을 알 수 있다.

〈표 12〉 CentOS의 /var/log/syslog 중 일부

6월 24일 16:57:11 localhost.localdomain systemd-timedated[8082] : chronyd.service :Disabling unit. 6월 24일 16:57:11 localhost.localdomain systemd-timedated[8082] : **Set NTP to disabled.** 6월 03 16:57:00 localhost.localdomain systemd-timedated[8082] : **Changed local time to** Sun 2018-06-03 16:57:00 KST 6월 03 16:57:00 localhost.localdomain systemd-timedated[8082] : **Changed local time to** Sun 201906-03 16:57:00 KST

<표 13>에 나타난 바와 같이 date 명령어를 통해 변경된 날짜와 시간 정보를 확인할 수 있다.

〈표 13〉 터미널에서 date 명령어로 확인

[root@localhost leesan]# **date** 2025.02.01.(토) 05:33:32 KST

<표 14>를 통해 6월 24일 16:36에 NTP가 비활성화 되었다는 로그 메시지가 생성되었고, 이후 "Changed local time to-"라는 로그 메시지가 순차적으로 생성되는 것을 확인할 수 있다. 로그 분석 결과, 네트워크 동기화를 비활성화한 시점인 NTP 비활성화 로그 메시지가 생성된 2024-06-24 16:36에 시간을 변경하였다는 것을 알 수 있다.

리눅스 PC의 경우 재부팅 시에도 syslog가 저장되어 있어 로그 분석을 통해 변경 시점을 확인할 수 있다(<표 15> 참조).

〈표 14〉 CentOS의 /var/log/syslog 중 일부

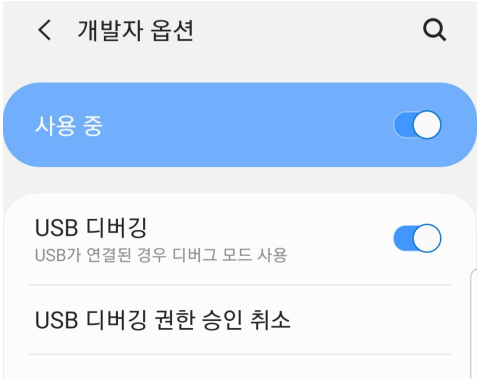
6월 24 16:35:00 localhost.localdomain systemd[1] : systemd-timedated.service : Deactivated successfully. 6월 24 16:36:11 localhost.localdomain systemd[1] : systemd-timedated.service : Deactivated successfully. 6월 24 16:36:15 localhost.localdomain systemd-timedate[6577] : chronyd.service : Disabling unit. 6월 24 16:36:15 localhost.localdomain systemd-timedate[6577] : **Set NTP to disabled.** 6월 24 16:36:00 localhost.localdomain systemd-timedate[6577] : **Changed local time to** Sat 2023-06-24 16:36:00 KST 6월 24 16:36:00 localhost.localdomain systemd-timedate[6577] : **Changed local time to** Sat 2022-06-24 16:36:00 KST

〈표 15〉 syslog 파일들

syslog syslog.1 syslog.2.gz syslog.3.gz syslog.4.gz

4.3 안드로이드

안드로이드의 경우 (그림 3)과 같이 설정 - 개발자 옵션 - USB 디버깅을 활성화시킨 뒤 리눅스와 같은 방법으로 시간을 조작한다.



(그림 3) USB 디버깅 활성화

시간 조작 여부를 파악하기 위해, 안드로이드에서는 'ADB(Android Debug Bridge)' 를 이용하여 로그를 추출한다. 로그를 추출하는 명령어는 <표 2>에 나타나 있다.

먼저, logcat의 경우에는 부팅 시점 이후로 log들이 누적되는 특성이 있다. 따라서 만약 범죄자가 시간을 조작한 후 현재시간으로 되돌려 재부팅을 한다면 logcat에서는 시간 조작에 대한 증거를 찾을 수 없다. 그러나 bugreport의 경우에는 logcat을 포함하여 상위 로그 파일들까지 추출이 가능하다. 따라서 기존 휘발성 로그를 누적하지 못하는 logcat의 단점을 bugreport를 통하여 극복 가능하다. 안드로이드의 경우의 실험에 대한 정보는 <표 16>과 같다.

〈표 16〉 안드로이드에서 시간 조작 실험 정보

안드로이드 실험 설정		
대상 기기	Galaxy S8, 안드로이드 9	Galaxy S21, 안드로이드 14
실험 일시	2024.07.21. 14:41	2024.07.21. 15:08
변경 일시	2024.07.14	2024.07.28
실험 방법	시간 조작 후 현재 시간으로 되돌려 재부팅 후 로그 확인	

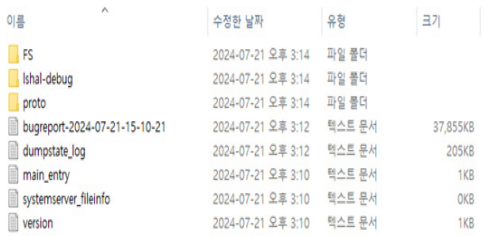
4.3.1 안드로이드 9

안드로이드 9에 대한 logcat 결과가 <표 17>에 나타나 있다. <표 17>을 분석해 보면 재부팅 시점 이후로 로그 데이터가 누적되는 걸 확인할 수 있다. 따라서 범죄자가 시간을 조작하고 현재시간으로 되돌린 후 재부팅을 하면 이전 로그들에 대해 확인이 불가능하다.

(그림 4)는 안드로이드 9에 대한 bugreport 결과이다. 안드로이드 9의 경우 bugreport 명령어 사용 시 'bugreport-2024-07-21-15-10-21.txt' 로 시스템 로그가 생성된다. 해당 파일을 분석한 결과는 (그림 4)와 같다.

〈표 17〉 안드로이드 9의 logcat 화면

```
PS C:\Users\cgumg> adb logcat
---------beginning of main
07-21 15:08:42.512 3212 3212 I SeLinux : SELinux : Loaded service_contexts from :
07-21 15:08:42.513 3212 3212 I SeLinux : /vndservice_contexts
07-21 15:08:42.602 3211 3211 I hwservicemanager : hwservicemanager is ready now.
07-21 15:08:42.605 3210 3210 I SeLinux : SeLinux : Loaded service_contexts from :
07-21 15:08:42.606 3210 3210 I SeLinux : /plat_service_contexts
07-21 15:08:42.606 3210 3210 I SeLinux : /vendor_service_contexts
07-21 15:08:42.619 3217 3217 D scs : bODMorProduct : false, bOmcSupport : true
07-21 15:08:42.619 3217 3217 D scs : OMC_B28_FILE : /system/omc/omc_b2b_list
07-21 15:08:42.619 3217 3217 D scs : getValue FromFile : Luc
```



(그림 4) bugreport 로그 파일

<표 18>에서 밑줄이 그어진 이탤릭체로 표시된 부분을 통해 시간 조작이 이루어졌고, 재부팅이 이루어졌다는 걸 알 수 있다.

〈표 18〉 안드로이드 bugerport 파일 분석 결과

+9d22h00m43s027ms(14) TIME : 2024-07-27-15-08-19
+9d22h00m43s160ms(2) c4190022 +fg=u0a53: "com.lguplus.appstore"
+9d22h00m43s235ms(2)023 c4190022 + tmpwhitelist =u0a187:"fg-service-launch"
+9d22h00m43s387ms(2)023 c4190022 +fg=u0a187:"com.samsung.android.net.wifi.wifiguider"
+9d22h00m45s252ms(2)023 c4190022 -fg=u0a53: "com.lguplus.appstore"
+9d22h00m46s762ms(3023 c4190022 current=281 ao temp=33 pst_temp=32 bat_temp=28 chg_temp =31 pa_temp=29
+9d22h00m48s460ms(3)023 c4190022
+9d22h00m48s469ms(14) *TIME : 2024-07-28-06-00-00*
+9d22h00m51s004ms(14) *TIME : 2024-07-21-15-08-26*
+9d22h00m58s571ms(12) SHUTDOWN
+9d22h00m58s582ms(12) START

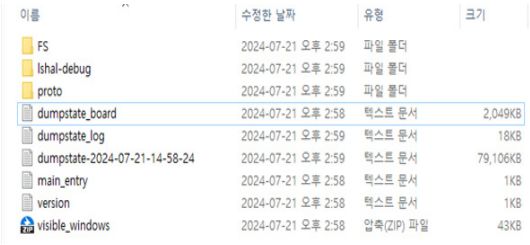
4.3.2 안드로이드 14

<표 19>는 안드로이드 14에 대한 logcat 결과이다. <표 19>를 분석한 결과, 안드로이드 9와 같이 재부팅 시점 이후로 로그 데이터가 누적되는 걸 확인할 수 있었다.

〈표 19〉 안드로이드 14의 logcat 화면

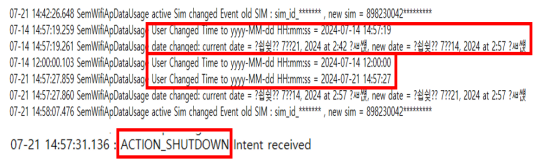
PS C:\Users\cgumg〉 **adb logcat**
---------beginning of main
07-27 14:41:37.880 626 626 W linker : Warining : failed to find generated linker configuration from "/linkerconfig/ ld.config.txt"
07-27 14:41:37.881 625 625 W linker :Warining : failed to find generated linker configuration from "/linkerconfig/ ld.config.txt"
07-27 14:41:37.884 624 624 D lmkd : fetech chimera enabled flag : [0] from property
07-27 14:41:37.884 624 624 I lmkd : Using psi monitors for memory pressure detection
07-27 14:41:37.884 624 624 I lmkd : lmkd enable_killbooster_all
07-27 14:41:37.884 624 624 I lmkd : set custom_sw_limit : 500
07-27 14:41:37.884 624 624 I lmkd : set upgrade_pressure : 80
07-27 14:41:37.884 624 624 I lmkd : set lmkd_freelimit_val : 13
07-27 14:41:37.884 624 624 I lmkd : enable_upgrade_ criadj : 0
07-27 14:41:37.884 624 624 I lmkd : Process polling is supported
07-27 14:41:37.885 624 624 I lmkd : Process reaping is not supported
07-27 14:41:37.886 624 629 W libprocessgroup : Controlier io is not found
07-27 14:41:37.887 624 629 W libprocessgroup : Controlier io is not found
07-27 14:41:37.887 624 629 W libprocessgroup : Controlier io is not found

(그림 5)는 안드로이드 14에 대한 bugreport 결과를 보여준다.



(그림 5) 안드로이드 14의 bugreport 결과

(그림 5)를 보면, 안드로이드 14는 안드로이드 9와는 달리 'dumpstate-2024-07-21-14-58-24.txt' 라는 로그 파일이 생성되는 것을 확인할 수 있다. 해당 파일을 분석한 결과를 (그림 6)에 제시하였다.



(그림 6) 안드로이드 bugreport 파일 분석 결과

(그림 6)을 분석하면 박스로 표시된 부분을 통해 먼저 시간 조작이 이루어지고 현재 시간으로 되돌린 후, 재부팅을 수행한 것을 알 수 있다.

따라서, 안드로이드 9와 안드로이드 14 두 버전 모두 logcat을 통하여 시간 조작 및 재부팅에 대한 로그 메시지를 확인할 수 없었지만, bugreport를 통하여 확인할 수 있었다. 이에 대한 결과를 정리하면 <표 20>와 같다.

〈표 20〉 안드로이드 실험 결과 정리

	안드로이드 9	안드로이드 14
logcat을 통한 시간 조작 증거 확인	X	X
bugreport를 통한 시간 조작 증거 확인	O	O

5. 논의

본 논문에서는 리눅스 기반 PC와 안드로이드 스마트폰에서 시간 정보를 조작한 후, 로그 분석을 통해 그 시점을 파악하는 방법을 제시하였다. 특히, 안티 포렌식 기법으로 타임스탬프를 조작하여 디지털 포렌식 분석을 방해하려고 할 때, 타임스탬프 조작 여부를 탐지하는 연구에 초점을 맞추었다.

리눅스 기반 OS(Ubuntu 24.04, CentOS)의 경우, 시간 조작 후 syslog를 통해 변경된 시점을 정확히 파악할 수 있었다. 실험 결과, NTP(Network Time Protocol)가 비활성화 되었다는 로그 메시지가 생성된 이후에 "Change local time to-"라는 메시지가 순차적으로 기록되는 것을 확인하였다. 이는 네트워크 동기화가 비활성화된 상태에서 사용자가 수동으로 날짜와 시간을 조작할 때 NTP 프로토콜이 비활성화 된다는 것을 의미한다. 본 논문은 선행연구[7]에 CentOS 리눅스와 안드로이드 14를 추가하여 실험하고 결과를 분석하였다. 또한, 시간 변경 관련 메시지 기반의 분석을 포함하여, NTP 프로토콜과의 연관성을 분석하여 타임스탬프 조작 시점을 명확히 파악할 수 있었다.

〈표 21〉 관련 연구와 본 논문과의 비교 분석

	[11]	[12]	[7]	본 논문
운영체제	Windows 10	iOS (7, 8)	안드로이드 9, Ubuntu 22.04	안드로이드 (9, 11), Ubuntu 24.04, CentOS stream 9
파일 시스템	NTFS (New Technology File System)	APFS (Apple File System)	Ext4	Ext4
탐지하려는 분석 행위	파일의 타임스탬프를 조작하는 행위	파일의 타임스탬프를 조작하는 행위	시스템의 시간을 조작하는 행위	시스템의 시간을 조작하는 행위
분석 파일	*$MFT, $LogFile, $UsnJrnl, $Prefetch, Registry, LNK Files, Event Log, Volume Shadow Copy*	*sms.db, Callhistory.storedata, Photos.sqlite, CurrentPowerLog.PLSQL, CurrentPowerLog.powerlog, general.log*	**안드로이드 9**	**안드로이드 9**
			logcat으로 추출한 파일	*bugreport-YYYY-MM-DD-HH-MM-SS.txt*
			Ubuntu 22.04	**안드로이드 11**
			/var/log/syslog	*dumpstate-YYYY-MM-DD-HH-MM-SS.txt*
				Ubuntu 24.04, CenOS stream 9
				/var/log/syslog

<표 21>은 본 논문과 기존 연구들과의 차이점을 보여준다. Oh et al.[11]은 Windows 10을 탑재한 데스크톱 컴퓨터에서 파일의 타임스탬프를 조작하는 행위를 탐지하였다. $MFT, $LogFile, $UsnJrnl, Prefetch, Registry, LNK 파일, 이벤트 로그, 볼륨 새도우 복사본과 같은 다양한 아티팩트들을 수집하고 분석하였다. 이상현 등[12]은 iOS를 탑재한 아이폰과 아이패드에서 파일의 타임스탬프를 조작하는 행위에 관한 흔적을 조사하였다. 이를 위해, 그들은 sms.db, Callhistory.storedata, Photos.sqlite, CurrentPowerLog.PLSQL,CurrentPowerLog.powerlog,general.log 등을 분석하였다.

본 연구진의 선행연구[7]는 Ubuntu 22.04와 안드로이드 9에서 시스템의 시간을 조작하는 행위가 로깅시스템에 미치는 영향을 분석하고, 시스템 시간 조작 여부를 탐지하기 위해 로그를 분석했다. 안드로이드 9에서는 logcat 명령어로 추출한 로그를 분석하였다. 본 논문은 [7]을 확장한 것으로, 추가적인 대상 OS로 안드로이드 14와 CentOS를 포함하여 실험하였고, 시간 동기화와 관련이 있는 NTP 프로토콜에 관한 로그와 비휘발성 로그인 bugreport를 통해 새로운 아티팩트를 수집 및 분석하였다.

안드로이드 기반 OS에서는 logcat 명령어를 사용하여 로그를 분석할 수 있다. 그러나 logcat을 이용해서 휘발성 로그만을 수집할 수 있어, 기기를 재부팅한 후에는 시간이 조작된 시점을 확인하기 어려웠다. 본 논문에서는 bugreport 명령어를 통해 비휘발성 파일을 수집하여 분석함으로써 시간이 조작된 시점을 효과적으로 파악할 수 있었다.

6. 결과

본 논문에서는 리눅스 기반 PC와 안드로이드 스마트폰에서 시간 조작을 통한 안티 포렌식 기법을 분석하고, 이러한 조작 행위를 탐지할 수 있는 방법을 제시하고 실험하였다. 논문에서 파악한 주요 결과는 다음과 같다. 첫째, 리눅스 시스템에서 NTP 비활성화 로그 메시지와 "Change local time to-" 메시지를 통해 시간 조작의 시점을 파악할 수 있다. 둘째, 안드로이드 시스템에서는 logcat과 bugreport를 활용하여 시간 조작의 시점을 파악할 수 있다. 셋째, 재부팅 후에도 syslog와 bugreport를 통해 시간 조작의 시점을 파악할 수 있다.

이러한 결과물들은 타임스탬프 조작을 통한 안티 포렌식 공격에 대응하기 위한 중요한 자료를 제공한다. 예로, 로그 파일을 통해 시간 조작의 시점을 파악하여 특정 사건과 관련된 아티팩트들에 대해 신뢰할 수 있는 타임라인을 구성할 수 있다. 향후에는 더 다양한 OS와 더 많은 안티 포렌식 기법을 고려하여 포괄적인 포렌식 분석을 수행할 필요가 있다.

Acknowledgements

이 연구는 2021년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(no. 2021R1A2C2012574). 이 연구는 2022년도 정부(과학기술 정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임(No.1711170476, 이벤트 기반 실험시 스템구축을 통한 자동차 내·외부 아티팩트 수집 및 통합 분석 기술 개발). 이 연구는 2024년도 산업통상자원부 및 한국산업기술진흥원(KIAT) 연구비 지원에 의한 연구임('P0023522')

참고문헌

[1] Michael G. Nobleet, Mark M. Pollitt, and Lawrence A, Presley, "Recovering and Examining Computer Forensic Evidence", Forensic Science Communication Vol.2, No. 4, Oct. 2000.

[2] Carrie Morgan Whitcomb, "An Historical Perspective of Digital Evidence : A Forensic Scientist's View", International Journal of Digital Evidence, Vol. 1, No. 1, pp. 7-15, Spring 2002.

[3] Simon L. Garfinkel, "Anti-forensics: Techniques,

detection and countermeasures", INTERNATIONAL CONFERENCE ON INFORMATION WAREFARE AND SECURITY, Volume 2, No. 1, pp 77-84, 2007.

〔4〕 R. González Arias, J. Bermejo Higuera, J. J. Rainer Granados, J. R. Bermejo Higuera, and J. A. Sicilia Montalvo,"Systematic Review: Anti-Forensic Computer Techniques", Applied Sciences, Vol. 14, No. 12, 2024.

〔5〕 Byeongyeong Yoo, "Analysis of File Time Change by File Manipulation of Linux System", The Journal of the Institute of Internet, Broadcasting and Communication, Vol. 16, No. 3, pp. 21-28, 2016.

〔6〕 Murtaza Ahmed. & M.N.A Khan "A Review Of Forensic Analysis Techniques For Android Phones", Journal of Independent Studies and Research - Computing, Vol. 15, NO. 1, 23-30, 2017.

〔7〕 S. Lee, J. Jung, S. Ahn and S. Cho, "Analysis of the Impact of Time Information Manipulation on Logging in Linux and Android", Spring Conference of Korean Institute of Next Generation Computing, April 2024.

〔8〕 Bhupendra Singh and Gaurav Gupta, "Analyzing windows subsystem for linux metadata to detect timestamp forgery", Advances in Digital Forensics XV: 15th IFIP WG 11.9 International Conference, Orlando, FL, USA, January 28 - 29, 2019, Revised Selected Papers 15. Springer International Publishing, 2019.

〔9〕 Jong-Hwa Song and Hyun-Seob Lee, "A Design of Timestamp Manipulation Detection Method using Storage Performance in NTFS", Journal of Internet of Things and Convergence, Vol. 9, No. 6, pp.23-28, 2023.

〔10〕 Thomas Göbel and Harald Baier. "Anti-forensics in ext4: On secrecy and usability of timestamp-based data hiding", Digital Investigation, Vol. 24, pp. S111-S120, 2018.

〔11〕 J. Oh, S. Lee and H. Hwang, "Forensic Detection of Timestamp Manipulation for Digital Forensic Investigation", IEEE Access, vol. 12, pp. 72544-72565, 2024.

〔12〕 Sanghyun Lee, Yunho Lee, and Sangjin Lee. "A Study on the Evidence Investigation of Forged/Modulated Time-Stamp at iOS (iPhone, iPad)", KIPS Tr. Comp. and Comm. Sys, Vol.5, No.7, pp.173-180, 2016.

저자소개

◆ 이산

- 2019년 3월~현재 단국대학교 산업보안학과 학사과정
- 관심분야: 디지털 포렌식, 안티 포렌식, 정보보안

◆ 조민혁

- 2020년 3월~현재 단국대학교 모바일시스템공학과 학사과정
- 관심분야: 디지털 포렌식, 안드로이드, 시스템 보안

◆ 정지헌

- 2018년 3월~2024년 2월 단국대학교 소프트웨어학과 학사과정
- 2024년 2월~현재 단국대학교 컴퓨터학과 석사과정
- 관심분야: 디지털 포렌식, 정보보안, 차량 내부 네트워크 보안

◆ 조성제

- 1989년 2월 서울대학교 컴퓨터공학과 공학사
- 1991년 2월 서울대학교 컴퓨터공학과 공학석사
- 1996년 8월 서울대학교 컴퓨터공학과 공학박사
- 1997년 3월~현재 단국대학교 컴퓨터학과/소프트웨어학과 교수
- 관심분야: 디지털 포렌식, 시스템 보안 및 악성코드 분석, 인공지능 보안, 시스템 소프트웨어 등

2024 한국차세대컴퓨팅학회 춘계학술대회 개최

□ 행사개요

- ㅇ 행 사 명 : 2024 한국차세대컴퓨팅학회 춘계학술대회
- ㅇ 일 시 : 2024. 4. 25.(금) ~ 4. 27(토)
- ㅇ 장 소 : 한국교통대학교(충북 충주)
- ㅇ 주최 · 주관 : 한국차세대컴퓨팅학회
- ㅇ 후 원 : 쌍용정보통신, 올포랜드, 태진티엔에스, 오스코, SK브로드밴드, 세림TSG, 세오, 트라콤
- ㅇ 프로그램

날짜	시간		내용	
1일차				
4/25 (목)	13:00~14:50	110分	학술대회 운영위원회	
	14:50~15:00	10分	Break	
	15:00~17:00	120分	학술대회 참석자 모임	
2일차				
4/26 (금)	09:30~10:00	30分	학술대회 등록	
	10:00~10:10	10分	개회식 축사 : 교통대 총장, 개회사 : 이상웅 회장	
	10:10~10:45	35分	Keynote Speech Ⅰ 사례에서 배우는 클라우드 플랫폼을 안전하게 사용하기 (이권왕 선임_안랩A-FIRST팀)	
	10:45~11:20	35分	Keynote Speech Ⅱ VR과 AR을 위한 3D 비디오 코딩기술 특허 분석 (박상철 수석심사관_특허청 전기통신심사국 방송미디어심사팀)	
	11:20~11:25	5分	Break	
	11:25~12:00	35分	Keynote Speech Ⅲ 정부 ICT R&D예산 현황 및 중점 투자 방향 (박상원 사무관_과학기술정보통신부 정보통신방송기술정책과)	
	12:00~12:30	30分	신임 교수 소개(3분)	
	12:30~12:35	5分	기념품 추첨	
	12:35~14:00	85分	Lunch	
	14:00~15:10	70分	Oral Session Ⅰ 인공지능 및 기계학습	Oral Session Ⅱ 의료, 멀티미디어 콘텐츠
	15:10~16:15	65分	**Poster Session 차세대컴퓨팅 기술 전 분야**	
	16:15~17:25	70分	Oral Session Ⅲ 인공지능 및 기계학습	Oral Session Ⅳ 정보보호, IoT
	17:25~18:00	35分	우수논문 시상 / 기념품 추첨 이벤트	
3일차				
4/27 (토)	10:00~11:00	60分	Oral Session Ⅴ 차세대 컴퓨팅 기술 전 분야	
	11:00~11:10	10分	Break	
	11:10~12:30	90分	산학협력 워크샵	

o 세부 프로그램

날짜	시간		내용	
1일차				
4/25 (목)	13:00~14:50	110分	학술대회 운영위원회	
	14:50~15:00	10分	Break	
	15:00~17:00	120分	학술대회 참석자 모임	
2일차				
4/26 (금)	09:30~10:00	30分	학술대회 등록	
	10:00~10:10	10分	개회식 축사 : 교통대 총장, 개회사 : 이상웅 회장	
	10:10~10:45	35分	Keynote Speech Ⅰ 사례에서 배우는 클라우드 플랫폼을 안전하게 사용하기 (이권왕 선임_안랩A-FIRST팀)	
	10:45~11:20	35分	Keynote Speech Ⅱ VR과 AR을 위한 3D 비디오 코딩기술 특허 분석 (박상철 수석심사관_특허청 전기통신심사국 방송미디어심사팀)	
	11:20~11:25	5分	Break	
	11:25~12:00	35分	Keynote Speech Ⅲ 정부 ICT R&D예산 현황 및 중점 투자 방향 (박상원 사무관_과학기술정보통신부 정보통신방송기술정책과)	
	12:00~12:30	30分	신임 교수 소개(3분)	
	12:30~12:35	5分	기념품 추첨	
	12:35~14:00	85分	Lunch(개별 식사)	
	14:00~15:10	70分	Oral Session Ⅰ 인공지능 및 기계학습	Oral Session Ⅱ 의료, 멀티미디어 콘텐츠
			An Improved DenseNet-Based Accurate Identification of Counterfeit QR Codes	Semantic Diffusion Model을 이용한 고화질 대장 내시경 영상 생성 및 과정 분석
			Study on automatic assessment of the possibility of inferior alveolar nerve injury	Diffusion을 이용한 Brain Tumor MRI데이터 생성 기술
			단안 3D 자세 추정을 위한 글로벌 및 시공간 트랜스포머	착용형 청각기기 적용을 위한 음성 향상 신경망 연구
			프로토타입 기반의 의료 지식 그래프 대조 표현 학습	전기동차용 공기스프링 정비 훈련용 실감형 콘텐츠 개발
			패혈증을 위한 적응적 다중-헤드 셀프-어탠션 기반 동적 치료 추천 학습	Emotion Recognition in Speech Signals through Graph Neural Networks Integration:A GCN-GAT Hybrid Model
			자폐 스펙트럼 장애 진단을 위한 시계열 적대적 생성 신경망 기반의 동적 연결 패턴 생성 연구	MediaPipe, GRU, TensorFlow를 활용한 수어 영상 인식
			Surveillance Abnormal Activity Recognition Using Residual Deep	Exploring the Potential of Mediapipe Hand Landmarks for Word-level Sign

			Bidirectional LSTM Network	Language Recognition through Masked-GRU Deep Learning
	15:10~16:15	65分	**Poster Session 차세대컴퓨팅 기술 전 분야**	
		클라우드 외	독성 단백질 분류를 위한 컴퓨터 라벨링 데이터의 효용성 입증	
			차량 궤적 예측을 위한 시계열 딥러닝 모델의 성능 평가	
			스마트 파이프 센싱 데이터 분석 운영환경 구축을 위한 쿠버네티스 아키텍처	
			해석 가능한 잠재변수를 활용한 다음 프레임 예측 기법	
			5G통신 광 전송 다중화 시스템의 OTDR 기능 구현	
			IoT 플랫폼 연동 데이터 시각화의 오픈 소스 구현 구조	
			자율주행 환경 돌발상황 관리를 위한 첨단 정보제공 서비스 연구	
			AI 기반 SNS 사용자 활동성 및 비활동성 진단 모델 개발	
			디지털 트윈을 활용한 커넥티드 카의 실시간 통신과 패킷 복구 메커니즘 설계 및 실험	
		정보보호 기술	미션 크리티컬 서비스 인증 프로토콜에 대한 보안성 분석	
			동형암호 기반 프라이버시 보존 NFT 거래 시스템	
			Deep Learning-based Cryptanalysis on Lightweight Block Ciphers	
			PAC 기반 CFI 동향 분석	
			허니팟과 윈도우 이벤트 로그 분석을 통한 랜섬웨어 탐지 및 방어 기법	
			리눅스와 안드로이드에서 시간 정보 조작이 로깅에 미치는 영향 분석	
			오픈소스를 활용한 정적 분석 도구 성능 비교	
			TLS 1.3 0-RTT Handshake의 Replay Attack 취약점과 대응 방안	
			산업제어시스템 대상 훈련용 사이버공격 시나리오 품질 평가 방법론 제안	
			원격 접속 취약점 관련 ICS 사이버공격 사례 분석 기반 SMR 보안 설계 요구사항 도출	
			그래프 신경망 기반 하드웨어 트로이목마 검출 연구 동향	
		인공지능 기계학습	Optimizing Drone Detection based on large-scale dataset using Yolov8	
			강화학습에 대한 정책 전이 기법 조사	
			DKN을 이용한 개인 맞춤형 뉴스 추천 시스템	
			테이블형 데이터셋의 이미지화 기법 설계	
			테이블형 데이터셋의 이미지화 방법론 연구	
			A Modified Vision Transformer-based Anomaly Recognition using Audio Data	
			기계학습 기법을 활용한 간편 신용평가 연구	
			스마트 파이프 센싱 데이터의 오류 검출 모듈 연구 및 검증	
			심층 강화학습을 활용한 암호화폐 트레이딩	
			Enhancing Intrusion Detection with Optimized RNNs for Network Traffic Analysis	
			Advancing Car Crash Detection: ConvNext Large and Active Learning Framework	
			히트맵 데이터를 사용한 PoseConv3D모델 기반의 한국 수어 인식 학습	
			고유값을 사용한 인체 포즈 인식 방법	
		기술전	어두운 환경이 Object Detection에 미치는 영향	

		분야	철도 구성요소 데이터셋을 활용한 YOLO V9과 RT-DETR의 객체 탐지 성능 분석	
			Source Location Privacy: Phantom-routing a Brief Survey	
			도로 및 지하철 공사 구간 정보를 활용한 내비게이션 시스템 개발	
			PRPD 유형 분류를 위한 Vision Transformer 기반 모델 간 성능 비교	
			가상현실 기술을 활용한 몰입형 정맥채혈 시뮬레이터 개발	
			몰입형 VR 기반 심장초음파 검사 시뮬레이터 개발	
			디지털 격차 해소를 위한 GPT 기반 대화형 키오스크	
			CNN을 활용한 M-PSK 변조 기반 무선 광통신 시스템에서의 대기 난류 채널 분류 성능 비교	
			Performance Analysis of Prefetching Techniques on Processing-in-Memory System	
	16:15~17:25	70分	Oral Session Ⅲ 인공지능 및 기계학습	Oral Session Ⅳ 정보보호, IoT
			발달성 고관절 이형성증 진단을 위한 딥러닝 기반 절구지수 자동 측정	SLAM에 의존하지 않는 강화학습 기반 Navigation시스템 제안
			Enhancing Crowd Counting Efficiency Using Spatial Attention based Multi-column CNN	계층형 주요정보통신기반시설 보안 관제 시스템
			A Self-Supervised Learning Approach For Aerial Image Segmentation	자연어 처리를 이용한 금융 위험 관리
			3D Reconstruction of Stomach Anatomy from Monocular Endoscope Video	블록체인을 활용한 자율적 사용자 동의 기반 개인 데이터 처리 시스템 설계
			IMU 센서와 RGB 영상을 이용한 인간행동인식 딥러닝 모델 연구	이상 탐지를 위한 딥러닝 모델의 특징 정보량 조절에 따른 이미지 데이터 재구성 결과 분석
			임베디드 시스템에서의 INT8 양자화 인식 학습을 통한 비용 효율적인 합성곱 신경망 구현	IoT 네트워크에서의 MTD 기반 사이버 기만기술 연구 동향 분석
			생성형 AI 기술을 활용한 기업 수요 맞춤형 MVP 효용성 연구	라디오믹스 기반 머신 러닝을 활용한 척추 압박 골절의 양성 및 악성 분류
	17:25~18:00	35分	우수논문 시상 / 기념품 추첨 이벤트	
3일차				
4/27 (토)	10:00~11:00	60分	Oral Session Ⅴ 차세대 컴퓨팅 기술 전 분야	
			군집화 기법을 활용한 과다 허용 방화벽 정책 최적화 프로세스에 대한 연구	
			독성 단백질 분류를 위한 컴퓨터 라벨링 데이터의 효용성 입증	
			다중 모달리티 활용을 위한 심층적 프롬프트 엔지니어링 분석	
			Automatic License Plate Recognition with YOLOv7: Observing Between Single-Line and Double-Line Layouts	
			최신 비전 트랜스포머 기반 객체 탐지 모델 연구	
			합성곱 신경망에서의 추론시간 감소를 위한 효율적인 동적 가지치기 기법	
	11:00~11:10	10分	Break	
	11:10~12:30	90分	산학협력 워크샵	

리눅스와 안드로이드에서 시간 정보 조작이 로깅에 미치는 영향 분석

Analysis of the Impact of Time Information Manipulation on Logging in Linux and Android

이산
산업보안학과
단국대학교
경기도 용인시
dltkshdkd915@dankook.ac.kr

정지헌
컴퓨터학과
단국대학교
경기도 용인시
wlgjsjames7224@dankook.ac.kr

안석현, 조성제
소프트웨어학과
단국대학교
경기도 용인시
{seokhyun, sjcho}@dankook.ac.kr

요 약

디지털 포렌식은 전자적으로 저장된 데이터를 식별·수집·처리·분석·보고하는 과정에 초점을 둔 포렌식 과학의 한 분야이다. 안티-디지털 포렌식은 디지털 포렌식 분석을 방해하는 기법으로, 데이터 암호화, 아티팩트 삭제, 타임스탬프 변경(자국 혼동, trail obfuscation) 등이 있다. 본 논문에서는 타임스탬프를 조작하여 디지털 포렌식 분석을 방해하는 기법이 로깅 시스템에 어떠한 영향을 미치는 지를 분석한다. 즉, 리눅스 기반의 PC와 안드로이드 기반의 스마트폰에서 의도적 시간 정보 조작이 로깅에 미치는 영향을 분석하고, 로그 분석을 통해 시간을 조작한 시점을 파악하는 방법을 제시한다.

키워드 : 디지털 포렌식, 안티 포렌식, 시간 조작, 로깅, 리눅스, 안드로이드

1. 서론

디지털 포렌식은 디지털 기기에 저장된 데이터를 수집·복구·분석·보고하는 과정을 포함한 포렌식 과학의 한 분야로, 법집행기관에서 컴퓨터 기기를 압수 및 수색하는 단계를 포함하여 압수된 기기로부터 잠재적 증거를 확보하는 수사기법을 말한다[1][2]. 안티 포렌식 기법(anti-forensic technique)이란, 디지털 포렌식 분석을 방해하는 것으로, 디지털 데이터를 조작, 삭제, 또는 난독화하거나 타임스탬프를 조작하여, 용의자(또는 범죄자)자 자신에게 불리하게 작용할 수 있는 증거물을 차단하려는 일련의 행위를 일컫는다[3]. 타임스탬프 변경은, 컴퓨팅 시스템의 시간을 특정 실제 사건 발생 전이나 후로 조작하는 것이다 [4]. 본 논문에서는 리눅스 기반의 PC와 안드로이드 기반의 스마트폰의 시간을 조작하여, 로깅 시스템에 미치는 영향에 대해 분석한다. 또한 시간 정보가 조작된 로그 파일들을 분석하여 타임스탬프가 조작된 시점을 파악하는 연구도 수행한다.

2. 관련 연구

윤지수 등[5]은 안티 디지털 포렌식 신종 기법과 이에 대응하는 형사처벌에 관한 내용을 제시하였다. 디지털 수사를 방해하는 안티 디지털 포렌식 기법들은 다양하며, 전통적인 안티 포렌식 기법 중 '데이터 위변조'의 타임스탬프 조작은 파일과 시스템 관리에 문제를 유발할 수 있으며 특정 사건 수사를 지연시킬 수 있다.

안드로이드 시간 조작이 디지털 포렌식에 미치는 영향을 조사한 연구도 있다. 조건희 등[6]은 안드로이드 스마트폰 시간 조작의 취약성이 앱 보안에 미치는 영향을 분석하였다. 스마트폰 설정의 변화가 JVM과 앱에 미치는 영향을 분석하여 사용자의 활동 조작, 앱 기능 중단, 증거 조작이라는 세 가지 유형의 공격을 식별했다. 이처럼, 안드로이드에서 시간 조작은 특정 앱들의 기능과 이벤트 타임라인 분석에 나쁜 영향을 미치며, 결과적으로 디지털 포렌식 수사를 지연시킬 수 있다.

또한 장대일 등[7]은 윈도우 NTFS 파일 시스템에서 타임스탬프 조작을 활용하는 안티포렌식 기법을 탐지하기 위한 포렌식 분석 방법을 제시하고 있다.

기존 연구에서는 안드로이드 시스템의 시간 조작이 앱 보안에 미치는 위험성과 윈도우 시스템에서 시간 조작을 탐지하는 연구가 수행되었다.

하지만 리눅스와 안드로이드의 시간 조작이 시스템 로그에 어떤 영향을 미치는지에 관한 연구가 거의 수행된 적이 없다. 본 논문에서는, 공격자가 기기의 시간을 임의로 조작할 때 로깅에 어떤 영

향이 있는지를 분석하고, 또한 시간이 조작된 시점을 파악하는데 중점을 둔다.

3. 실험 방법

본 절에서는 데스크톱 리눅스(Ubuntu 22.04)와 안드로이드 스마트폰(Samsung Galaxy S8, Pie)을 대상으로 시간 정보를 변경할 때, 로깅에서 어떤 영향이 발생하는지, 그리고 로그 분석을 통해 어떤 시점에서 시간 변경이 이루어졌는지 파악한다.

3.1 리눅스에서 시간 조작 및 로그 분석

그림 1은 리눅스 설정 탭에서 Automatic Date & Time을 비활성화 시켜준 후 Date & Time에서 과거/미래 시간으로 조작할 수 있음을 보여준다.

o 과거 시간대로 타임스탬프 변경

- 현재 일시 : 2024.3.10.15:00
- 변경 일시 : 2019.3.3. 08:30

그림 2에서, date 명령어를 통해, 바뀐 날짜와 시간 정보를 확인할 수 있다.

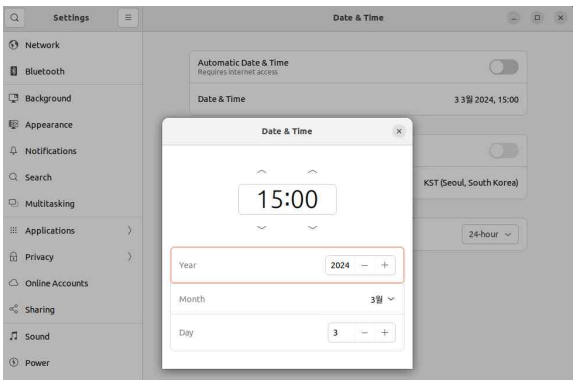


그림 1 리눅스 PC의 Setting 설정 탭

```
root@san-virtual-machine:~# date
2019. 03. 03. (일) 08:30:06 KST
```

그림 2 터미널에서 date 명령어로 확인

대부분의 로그 메시지는 syslog 파일에 저장된다. 그림 3과 같이 cat 명령어를 통해 syslog를 분석했다. 'WARNING' 문구가 포함된 타임스탬프 변경 로그가 기록된 메시지를 발견할 수 있었다. 1710082800의 UTC 시간대에 타임스탬프가 변경되었음을 알 수 있다.

```
Mar  3 08:30:07 san-virtual-machine apt.systemd.daily[5464]: WARNING: file /var/lib/apt/periodic/upgrade-stamp has a timestamp in the future: 1710082800
```

그림 3 /var/log/syslog 내용 중 WARNING 메시지

```
root@san-virtual-machine:/home/san# date -u -d @1710082800
2024. 03. 10. (일) 15:00:00 UTC
```

그림 4 utc 시간을 한국 표준시간으로 출력

로그를 보면 2024.03.10 15:00 시간대에 타임스탬프가 변경되었음을 알 수 있다(그림 4 참조).

o 미래 시간대로 타임스탬프 변경

- 현재 일시 : 2024。3。11 14:20
- 변경 일시 : 2025。2。1 05:30

그림 5에서 date 명령어로, 바뀐 날짜와 시간 정보를 확인할 수 있다.

```
root@san-virtual-machine:/home/san# date
2025. 02. 01. (토) 05:30:22 KST
```

그림 5 터미널에서 date 명령어로 확인

syslog 파일 분석 결과 현재 날짜로 로그들이 쌓이다가 특정 시점에 "Change local time to -" 라는 메시지가 생성되며 바뀐 시간대로 로그가 저장되는 것을 알 수 있다(그림 6 참조).

```
san@san-virtual-machine:~$ cat /var/log/syslog
Mar 11 14:20:40 san-virtual-machine systemd[1]: rsyslog.service: Sent signal SIGHUP to main process 831 (rsyslogd) on client request.
Mar 11 14:20:40 san-virtual-machine systemd[1]: logrotate.service: Deactivated successfully.
Mar 11 14:20:40 san-virtual-machine systemd[1]: Finished Rotate log files.
Feb 11 14:20:40 san-virtual-machine systemd-resolved[618]: Clock change detected. Flushing caches.
Feb 11 14:20:40 san-virtual-machine systemd-timedated[3980]: Changed local time to Tue 2025-02-11 14:20:40 KST
Feb 11 14:20:41 san-virtual-machine systemd[1]: apt-daily-upgrade.service: Deactivated successfully.
Feb 11 14:20:41 san-virtual-machine systemd[1]: Finished Daily apt upgrade and clean activities.
Feb 11 14:20:41 san-virtual-machine systemd[1]: apt-daily-upgrade.service: Consumed 2.288s CPU time.
Feb  1 14:20:40 san-virtual-machine systemd-timedated[3980]: Changed local time to Sat 2025-02-01 14:20:40 KST
Feb  1 14:20:40 san-virtual-machine systemd-resolved[618]: Clock change detected. Flushing caches.
Feb  1 04:20:40 san-virtual-machine systemd-timedated[3980]: Changed local time to Sat 2025-02-01 04:20:40 KST
Feb  1 04:20:40 san-virtual-machine systemd-resolved[618]: Clock change detected. Flushing caches.
Feb  1 05:20:40 san-virtual-machine systemd-resolved[618]: Clock change detected. Flushing caches.
Feb  1 05:20:40 san-virtual-machine systemd-timedated[3980]: Changed local time to Sat 2025-02-01 05:20:40 KST
Feb  1 05:30:40 san-virtual-machine systemd-resolved[618]: Clock change detected. Flushing caches.
Feb  1 05:30:40 san-virtual-machine systemd-timedated[3980]: Changed local time to Sat 2025-02-01 05:30:40 KST
Feb  1 05:30:40 san-virtual-machine systemd-resolved[618]: Clock change detected. Flushing caches.
Feb  1 05:30:40 san-virtual-machine systemd-timedated[3980]: Changed local time to Sat 2025-02-01 05:30:40 KST
```

그림 6 /var/log/syslog의 출력 결과

로그가 정상으로 쌓이다가 특정 시점에 "Changed local time to Tue 2025-02-11"로 변경된 것을 확인했다. 시간 변경 순서는 연도-날짜-시간 순으로 변경된다. 즉, 바뀐 연도 정보부터 출력되고 날짜, 시간까지 바뀌면서 바뀐 시간대로 로그 메시지들이 생성된다. 로그 분석 결과, 현재 시간대와 상이하게 출력되는 시점이 시간 변경한 시

점이다. 위 로그에서 3월 11일에 시간을 변경하였다는 것을 알 수 있다.

3.2 안드로이드에서 시간 조작 및 로그 분석.

그림 7은 안드로이드 스마트폰의 설정 탭이다. 리눅스와 유사하게 날짜 및 시간 자동 설정을 비활성화한 후 과거/미래 시간대로 변경할 수 있다.

그림 7 안드로이드폰의 설정 탭

안드로이드폰은 ADB(Android Debug Bridge)를 사용하여 시간을 조작할 수 있어, 개발자 옵션에서 그림 8과 같이 USB 디버깅을 활성화해야 한다.

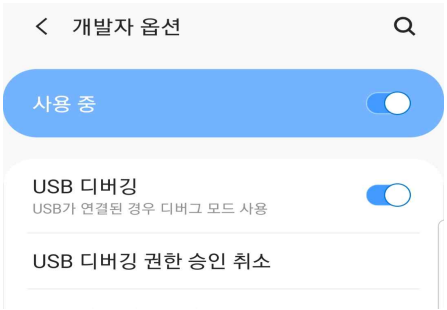


그림 8 USB 디버깅 활성화

o 과거 시간대로 타임스탬프 변경

- 현재 일시 : 2024。3。11 18:17
- 변경 일시 : 2025。3。3 08:30

그림 9는 adb shell의 date 명령어로 시간 정보가 변경된 것을 확인할 수 있다.

그림 9 adb shell date 명령어 실행 결과

그림 10은 logcat 명령어를 사용하여 시스템이 생성된 로그를 확인한 결과를 보여준다. 여기서 "AlarmMangerService: Setting time to day to sec=(UTC time)" 메시지는 UTC time으로 시간을 변경한 것을 나타낸다. 두 번째 메시지가 생성된 시간은 정상적인 시간이고 세 번째 메시지부터 변경된 시간으로 적용된 것을 확인했다.

그림 10 logcat 명령어 실행시

o 미래 시간대로 타임스탬프 변경

- 현재 일시: 2024。3。11 16:38
- 변경 일시: 2025。2。1 05:30

그림 11은 date 명령어로 바뀐 시간 정보를 확인한 것을 보여준다.

그림 11 adb shell date 명령어 실행 결과

그림 12는 logcat 명령어을 사용하여, 로그 메시지들 중에서 시간 조작과 관련된 시점을 찾는 과정을 보여준다.

그림 12 logcat 명령어 수행결과

로그 메시지들이 정상적으로 쌓이다가 특정 시점에 "AlarmMangerService: Setting time to day to sec=(UTC time)"이라는 메시지가 생성된 것을 확인하였다. 이후 시간이 변경되었음을 알 수 있었다. 리눅스와 마찬가지로 연도-날짜-시간 순으로 변경되기 때문에, 계속해서 로그 메시지를 조사하여 시간이 변경되는 것을 관측할 수 있었다.

리눅스와 안드로이드 기기에서, 과거 시간대 미

래 시간대 변경 시, 시간을 의도적으로 조작한 시점을 찾을 수 있었다.

4. 실험 결과

실험을 통해 현재 기기의 시간을 과거 및 미래 시간대로 조작할 수 있음을 확인했다(표 1 참고).

표 1. 리눅스 PC와 안드로이드 스마트폰의 시간 조작 후 로그 결과

	과거시간대	미래시간대
리눅스 기반 PC 버전 Ubuntu 22.04	'WARNING' 타임스탬프 경고문구로 변경된 시점 확인 가능.	정상적인 로그들이 쌓이다가 특정 시점에 "Change local time~" 문구 생성되며 바뀐 시간대로 로그 생성됨
안드로이드 스마트폰 삼성갤럭시 S8, Pie	'AlarmManger Service: Setting time to day to sec=(UTC time)' UTC time으로 변경된 시점 확인 가능	'AlarmMangerService: Setting time to day to sec=(UTC time)' UTC time으로 변경된 시점 확인 가능

안드로이드 logcat 명령어로 정상적인 로그들이 생성되다가 특정 시점에 시간 조작이 발생하였음을 확인할 수 있었다. 또한 경고문구 메시지를 분석하여 시간 조작을 시도한 시점의 utc 시간도 알아낼 수 있었다. 즉 용의자(범죄자)가 언제 시간을 조작하였는지 파악할 수 있었다.

5. 결론

본 논문에서는 리눅스 기반 PC와 안드로이드 기반 스마트폰을 대상으로 시간 정보 변경이 로그 정보에 미치는 영향을 분석하였다. 사용자(용의자, 범죄자)는 의도적으로 시간을 변경할 수 있었다. 또한, 수사관은 해당 기기의 로그 분석을 통해 시간을 변경한 시점을 파악할 수 있다.

하지만, 시간 변경이 이루어진 시점에 생성된 로그 메시지들이 시스템의 용량을 초과할 경우, 관련 로그 메시지가 사라질 수 있다는 한계점이 존재한다.

추후 다양한 운영체제에 대해서, 시스템 시간을 임의로 조작한 상황에서 시스템 로깅 기법이 어떤 영향을 받는지에 대해 연구할 계획이다.

Acknowledgement

이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(no. 2021R1A2C2012574). 또한, 2022년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임(No. 2022-0-01022, 이벤트 기반 실험시스템 구축을 통한 자동차 내·외부 아티팩트 수집 및 통합 분석 기술 개발).

참고문헌

[1] Michael G.Nobleet, Mark M. Pollitt, Lawrence A, Presley, "Recovering and Examining Computer Forensic Evidence", Forensic Communication Vol.2, Num. 4, Oct. 2000

[2] Carrie Morgan Whitcomb, "An Historical Perspective of Digital Evidence : A Forensic Scientist's View". International Journal of Digital Evidence, Vol 1., Spring 2002.

[3] 이정남, 사이버 범죄론, 사이버포렌식협회, 2005

[4] 조규상, "안티포렌식을 위한 타임스탬프 변경 도구들에 대한 디지털포렌식 관점에서의 기능의 분석", 한국컴퓨터정보학회 하계학술대회 논문집 제27권 제2호, 2019

[5] 윤지수, 이경렬 "안티 포렌식 신종기법에 대한 형사법적 대응방안" 한국형사정책학회 논문지, 제32권 제4호, pp. 65-99, 2021

[6] 조건희, 이연준 "안드로이드 시간 조작 취약점과 보안 문제" 한국정보처리학회 논문지, 제30권 제1호, pp. 183-184, 2023.05

[7] Jang, Dae-il, et al, "Understanding anti-forensic techniques with timestamp manipulation.", 2016 IEEE 17th International Conference on Information Reuse and Intergration(IRI), pp. 609-614, 2016