ISSN(Print) : 1976-5304, ISSN(Online) : 2713-931X
https://kdfs.jams.or.kr, DOI : 10.22798/kdfs.2022.16.2.74

비트코인 지갑의 아티팩트 분석 연구

안 정 언*, 이 상 진**
고려대학교 정보보호학과 (대학원생)*
고려대학교 정보보호대학원 (교수)**

The analysis of bitcoin wallet artifacts

Jung-Un Ahn*, Sang-Jin Lee**
Dept. of Information Security, Korea University (Graduate Student)*
School of Cybersecurity, Korea University (Professor)**

요 약

최근 가상화폐가 불법 약물 거래 및 개인 자금 세탁 등을 목적으로 하는 범죄 행위에도 빈번히 사용되고 있으나, 블록체인 기반 거래의 특성상 이에 대한 원점 추적이 어려우며 다크웹 등의 추가적인 은닉 수단을 활용하는 경우가 많아 범죄 수사에 제한이 생긴다. 본 논문에서는 가상화폐 지갑 서비스 아티팩트를 분석하였고, 지갑 사용자의 거래 행위와 관련한 데이터가 분석 대상 기기의 파일시스템 및 메모리에 저장되는 구조를 식별하고 추출하는 스크립트(WalletExtractor)를 제작하였다. 이를 활용하면 Windows 10 환경에서 별도로 설치하여 사용 가능한 4가지 지갑 서비스를 대상으로, 각 서비스 내에서 거래 주소의 생성 및 관리, 발생한 거래 내역 등에 관한 데이터를 자동으로 추출할 수 있다. 이러한 결과를 통해 수사 과정에서 확보한 디지털 기기에서 특정 가상화폐 거래가 실제로 발생하였음을 입증하고, 범죄 목적의 거래 주체를 특정할 수 있을 것이다.

주제어 : 비트코인, 가상화폐, 지갑, 클라이언트, 포렌식

ABSTRACT

Virtual currency is frequently used for criminal acts aimed at illegal drug transactions and personal money laundering, but due to the nature of blockchain-based transactions, it is difficult to trace the origin spot and additional concealment means such as the dark web are limiting criminal investigations. Therefore, we analyzed virtual currency wallet service artifacts. We identified the structure in which data related to the wallet transaction behavior is stored in the file system and memory of the device to be analyzed, and created the script named WalletExtractor to extraction of them. This was used to automatically extract data on the creation and management of transaction addresses and transaction details within each service for four wallet services that can be installed in Windows 10. Through these results, we can prove that a specific virtual currency transaction actually occurred on the digital device obtained during the investigation process and specify the subject of transaction for criminal purposes.

Key Words : Bitcoin, Virtual Currency, Wallet, Client, Forensics

※ 이 논문은 과학기술정보통신부·경찰청이 공동 지원한 '폴리스랩2.0 사업(www.kipot.or.kr)'의 지원을 받아 수행된 연구임.
〔과제명: 안티-포렌식 기술 대응을 위한 데이터 획득 및 분석 기술 연구 / 과제번호: 210121M07〕
▪ Received 07 March 2022, Revised 14 March 2022, Accepted 27 June 2022
▪ 제1저자(First Author) : Jungun Ahn (Email : wjddjs0330@korea.ac.kr)
▪ 교신저자(Corresponding Author) : Sangjin Lee (Email : sangjin@korea.ac.kr)

Ⅰ. 서 론

가상화폐의 개발 및 활용 기술이 발전하면서 실질적인 거래량이 증가하고, 개인 단위의 결제 수단으로 사용하는 등 그 활용과 범위가 넓어지고 있다. 하지만 가상화폐를 범죄 목적으로 사용하는 사례가 꾸준히 증가하고 있다. 사이버 공간에서 범죄 수단으로 접근하는 다크 웹에서 마약, 불법 약물 거래, 랜섬웨어 몸값 등의 결제 화폐로서 비트코인이 오랜 기간 통용되고 있다. 이로 인해 불법 거래가 발생하더라도 토르 브라우저 등 추가적인 은닉 수단을 활용할 경우, 실제 발생 주체에 대한 원점 추적이 어려운 상황이다. 하지만 특정 거래에 대한 실 주체를 식별하기 위해 IP 및 네트워크 노드 등의 경로 데이터를 통해 추적하는 것 외에, 클라이언트 차원에서 PC나 디지털 기기 분석을 통해 해당 거래의 발생을 입증할 수 있다. 검찰 또는 수사기관에서 압수수색 및 기타 경로로 디지털 기기를 획득하였을 때, 기기 내 설치된 지갑 서비스 프로그램과 그에 저장된 가상화폐 주소, 거래 내역 등의 정보로부터 거래 주체를 특정할 수 있으며, 이를 통해 추후 범죄 규모 및 관련 대상을 추적하는 데도 활용할 수 있을 것이다.

본 논문에서는 PC 내 설치된 4가지 가상화폐 지갑 서비스를 통한 사용자의 행위 데이터와 관련한 아티팩트를 분석하였다. 웹 서비스의 형태가 아닌 로컬 환경 내 실행 파일로 구동하는 방식의 지갑 서비스를 통해 실험을 진행하였으며, 분석에 활용한 가상화폐의 종류는 비트코인(BTC)으로 선정하였다. 최종적으로는 분석 결과를 반영하여 지갑 서비스별 각 아티팩트의 추출을 자동화하는 Python 기반의 스크립트(WalletExtractor)를 제작하였다.

분석 대상의 지갑 아티팩트는 Ⅲ절에 자세히 기술하였으며, 분석 환경 구축에 활용된 데이터 정보는 Ⅳ절, 각 서비스별 분석 결과는 Ⅴ절에 포함하였다. Ⅵ절에서는 본 논문의 결론과 향후 연구과제에 관련한 내용을 기술하였다.

Ⅱ. 관련 연구

전 세계적으로 가상화폐 시장의 규모가 점차 확대되고 다양한 서비스의 활용이 대중화됨에 따라 가상화폐와 관련한 선행 연구가 다양하게 진행되었다. 블록체인 기반의 거래 방식 및 거래 시스템의 안전성, 이에 활용되는 네트워크 프로토콜 등 다양한 영역에서의 접근이 이루어졌다. 이 중 가상화폐 거래의 특징 중 하나인 P2P(Peer-To-Peer) 방식을 기반으로 한 익명성과 관련하여, 이에 대한 탈익명화를 통해 엔드포인트 사용자를 식별하기 위한 연구도 진행되었다. 대표적으로 블록체인 데이터에 대한 직접 분석과 가상화폐 서비스의 특성과 관련 데이터를 분석하는 연구들이 진행되었다.

2.1 블록체인 분석

블록체인 데이터에서 확인할 수 있는 공개키를 기반으로, Meiklejohn 등[1]은 동일 사용자로부터 사용된 각 데이터(pubkey)를 연결하여 주소별 군집화를(Clustering) 시도하였다. 이를 통해 특정 주소가 포함된 그룹이 식별된다면, 해당 그룹 내 기타 주소에 대한 사용자 역시 특정할 수 있게 된다. 이와 유사하게 Spagnuolo 등[2]은 사용자 및 거래 주소에 대한 분류를 수행하는 도구를(Bitlodine) 제작하여, 실제 블랙마켓(Silk Road)에서 사용되는 주소들과 유명한 Mining pool에 대한 해킹 사례를 대상으로 성능 검증을 진행하였다.

그러나 범죄 목적으로 활용되는 다크웹 등의 은닉 수단이 사용될 경우, 사용자 추적을 위한 추가 과제가 존재한다. 또한 블록체인 데이터만을 활용할 경우, 사용자 식별 외에 거래 목적 및 주소별 관리 내역 등 범죄 수사 관점의 유의미한 추가 데이터를 확보하기 어렵다.

2.2 클라이언트 서비스 분석

가상화폐 서비스에 대한 연구는 대상 서비스의 종류, 운영체제 등 서비스 구동 환경, 파일시스템 또는 메모리 등 아티팩트 분석 영역에 따라 세분화되어 진행되었다. Gurkok 등[3]은 OS X 환경에서의 가상화폐 클라이언트 서비스(MultiBit)의 아티팩트 분석을 위한 Volatility 플러그인을 개발하였다. 이는 Python 기반의 정규식 패턴을 활용하여 거래 주소 등의 출력을 자동화한다.

모바일 기기에서 분석하기 위해 안드로이드 및 iOS 운영체제의 유사 연구도 진행되었으며, Haigh 등[4]은

안드로이드 환경의 서로 다른 7가지 서비스를 대상으로 서비스 아티팩트를 분석하였다. 이를 통해 중요 데이터의 저장 형태 및 사용자의 접근 권한 등을 식별하여 서비스별 보안성 검증에 기여하였다. Montanez[5]는 두 가지 운영체제 모두에서 분석하였다. 이는 서비스 구매 내역 및 서비스를 통해 수행한 행위별 시각 데이터를 추출하였으나, .log 파일 등의 관련 데이터에서 확인할 수 있는 readable text 데이터에만 초점을 맞추어 진행하였다. non-readable 형태의 지갑 서비스 데이터는 대표적으로 TimeStamp가 있으며, 이는 많은 경우에 Unix Time 포맷을 기반으로 한 4바이트 데이터 형태로 존재한다. 따라서 이를 포함한 기타 바이너리 데이터는 식별 불가능하다.

Windows 환경의 가상화폐 서비스 사용 행위에 대한 추적 연구로 Van Der Horst 등[6]은 2가지 지갑 서비스(Bitcoin Core 및 Electrum)를 대상으로 메모리 아티팩트를 분석하였다. 이후 Zollner 등[7]은 해당 두 가지 서비스를 포함한 다양한 서비스별 중요 데이터 파일의 경로를 식별하고, 이를 수집하는 도구를 개발하였다. 그러나 이러한 선행 연구에서는 Transaction ID, passpharase, address 등 다양한 아티팩트에 대하여 접근하였지만, Timestamp에 관한 데이터 분석은 진행하지 않았다. 따라서, 본 연구에서는 총 4가지 가상화폐 지갑 서비스에 대하여, 거래 주소 생성 및 거래 시각 등의 Timestamp를 포함한 아티팩트를 분석하였다.

마지막으로 Tyler Thomas 등[8]은 가상화폐 하드웨어 지갑에 대한 메모리 아티팩트 분석 및 시각화를 위한 Volatility 플러그인을 개발하였다. 메모리 내 식별 가능한 지갑 서비스 프로세스 정보 및 하드웨어 지갑 정보, 거래 내역 정보를 포함한 아티팩트 분석 상세 결과를 제공한다. 그러나 다양한 프로세스가 혼재하는 휘발성 메모리에서 특정 프로세스 데이터는 확보 시점에 따라 달라지지 때문에 파일시스템에 존재하는 지갑 서비스 아티팩트 및 관련 데이터의 저장 구조를 포함하여 추가적인 분석이 필요하다.

Ⅲ. 분석 아티팩트

본 논문에서 분석한 아티팩트는 지갑 서비스를 활용한 사용자 행위와 직접적인 관련이 있는 것으로, 블록체인 데이터와의 대조를 통해 해당 서비스로 실거래가 이루어졌음을 증명하는데 활용할 수 있다. 서비스 사용자가 가상화폐 거래를 발생시키는 과정에서 저장되는 사용자의 행위 흔적에 관한 데이터를 크게 3가지로 분류하여 분석하였다.

3.1 거래 주소 및 생성 시각

일반적으로 가상화폐 거래를 위해서 가장 먼저 지갑을 생성한다. 대부분의 서비스는 이 때 지갑 데이터와 해당 지갑을 통한 거래를 보호하기 위해 암호화 기능 옵션을 제공한다. 이를 통해 해당 지갑을 사용할 때에 패스워드 인증을 수행하고, 디스크에 저장되는 지갑 데이터 파일을 암호화하여 저장하게 된다.

지갑을 생성한 이후에 실질적인 자금 입출금에 활용할 거래 주소(receiving address)를 생성해야 한다. 대부분의 서비스는 하나의 지갑 내에서 여러 개의 거래 주소를 생성하여 사용할 수 있고, 각 주소별 입출금 내역이 종합되어 지갑 자금으로 반영된다. 생성하는 거래 주소의 포맷은 가상화폐의 종류에 따라 다르며, 동일한 가상화폐일지라도 주소를 생성하는 방식에 따라 그 형태에 차이가 있다. 여러 가상화폐 중 대표적으로 비트코인을 예시로 살펴보면, 분석 대상 서비스들에서 사용한 실제 주소 포맷은 크게 P2PKH (Pay to Pubkey Hash), P2SH (Pay to Script Hash), P2WPKH (Pay_to_Witness_Public_Key_Hash)로 분류된다. 각 비트코인 주소는 거래에 활용되는 특정한 Public Key, Private Key 쌍을 가지며, Public Key의 해싱 값을 기반으로 몇 가지 추가 변환 과정을 통해 생성된다. 3가지 주소 포맷은 각각 서로 다른 특성을 가지는데, P2PKH와 P2SH 포맷은 비트코인의 표준 스크립트이다. 또한 이들의 비트코인 주소 값은 각각 “1”, “3” 으로 시작한다. P2WPKH는 가장 최근에 등장한 포맷으로 기존의 base58 인코딩을 활용한 것과 다르게 base32 인코딩 방식을 기반으로 하며, 주소의 가장 앞부분을 통해 메인넷(bc)과 테스트넷(tb) 환경을 구분할 수 있다.

이렇듯 가상화폐별 주소의 표현 가능한 포맷 및 각각의 특성을 기반으로 하여, 분석 대상 기기 및 지갑 서비스로부터 가상화폐 주소와 관련한 아티팩트를 올바르게 분석할 수 있다. 분석 대상 기기의 디스크 내 파일이 일부 훼손되었거나 메모리 덤프를 확보하여 분석해야 할 경우, 주소 포맷별 특징을 통해 식별한 주소값의 유효성을 일차적으로 확인할 수 있을 것이다. 추가로 지갑 내에서 동시에 여러 개의 주소를 관리하는 경우에는, 각 주소의 생성 시각이 지갑 데이터의 일부로 저장된다. 블록체인 데이터는 거래 자체에 관련된 데이터들만을 기록하며, 거래 해시값과 거래 주소, 거래 시각들을 관리한다. 실제로 거래를 위한 주소의 생성 및 관리는 각 클라이언트가 수행하기 때

문에 클라이언트 측 로컬 데이터의 분석 외에는 다른 방안으로 생성 시각을 추적하기 어렵다. 해당 아티팩트는 타임라인 구성에 중요한 추가 단서로 활용될 수 있다.

〈Table 1〉 Wallet Service Information

Wallet Service	Version	Hash(MD5)
Bitcoin Core	v22.0.0	192E9704618653FF8D63EB98AE206060
Bither	v1.4.8	FCE8475F1028A66CDED444D621826007
Bitpay	v12.10.2	D3CB5E32587E8115A1124967BD14382C
Electrum	v4.1.5	7FE70A446ED538CB748F561F56F4A89C

3.2 거래 내역

사용자는 거래 주소를 생성한 이후에 해당 주소로 가상화폐 입출금을 진행한다. 거래가 발생하면 활용된 지갑 및 거래량 등을 포함한 해당 서비스 내에서 관련 데이터들이 전부 로컬 디스크에 저장된다. 또한 클라이언트 내의 메타데이터뿐 아니라, 거래를 통해 발생한 블록체인 데이터가 전부 기록되어 이에 대한 간단한 조회는 웹 서비스를 통해 가능하다. 해당 페이지에 가상화폐의 종류 및 주소 등에 대한 정보를 입력하면 거래 해시값 및 거래 시각, 해당 주소(계좌)의 잔금 등 실제 발생한 거래 내역에 대한 상세 정보를 조회할 수 있다. 이는 분석 대상 아티팩트를 통해 추출한 데이터의 유효성 검증을 위해서도 활용할 수 있다.

사용자는 지갑 주소와 관련한 데이터 파일만 보유한다면, 여러 기기에서 동일 지갑 서비스를 통해 해당 주소를 활용할 수 있다. 따라서 암호화되지 않은 지갑 데이터가 유출되었을 시에, 특정 주소(계좌) 내의 자금이 불법적으로 출금될 수 있다. 이러한 경우에 실제로 거래를 발생시킨 기기 또는 개인을 특정할 필요가 있으며, 이를 위해 대상 기기 내의 거래 내역 아티팩트를 분석하여 범죄 행위에 해당 기기와 설치된 지갑 서비스가 활용되었음을 입증할 수 있다. 실제로 특정 거래에 대하여 거래 시각 및 거래에 사용된 입출금 주소, 거래 가상화폐 종류 등 많은 관련 데이터가 존재한다. 이 중 각 거래의 식별자로 사용할 수 있는 정보로써 거래 해시(Transaction Hash) 및 거래 발생 시각을 중점적인 아티팩트로 선정하여 분석하였다.

3.3 지갑명 및 주소 별칭

사용자가 생성한 지갑 및 거래 주소에 대한 메타데이터 역시 거래 주소별 생성 시각과 동일하게 온전한 클라이언트 측 서비스 데이터로 저장되며, 블록체인 기반 데이터를 통해 확인할 수 없다. 대다수의 서비스는 가상화폐의 종류 및 기타 옵션에 따라 여러 개의 지갑을 생성할 수 있으며, 생성한 지갑 내에서 동시에 여러 개의 주소를 관리할 수 있다. 이 경우 사용자는 생성 목적에 맞게 지갑 명을 설정하며, 지갑 내 각 주소를 구분하고자, 주소를 생성할 때에 별칭(Description)을 기록할 수 있다. 이는 사용자에게 편의성을 제공함과 동시에, 수사 관점에서는 해당 주소에 관한 활용 목적 또는 거래 대상 등을 특정할 수 있는 단서로 활용될 수 있다.

Ⅳ. 분석 방안

4.1 분석 환경

본 논문에서는 Windows 10 64bit 운영체제에서 지갑 서비스 아티팩트 분석을 위한 실험 환경을 구성하였다. 각 서비스별 최신 버전은 공통적으로 Windows 10 환경에서 구동 가능하였으며, 모든 서비스를 웹 서비스 환경

〈Table 2〉 Wallets and Receiving Addresses Data

Wallet Service	WalletName (Encryption)	Receiving Address	Description	Creation Time
Bitcoin Core	Encrypted_Wallet (O)	3FsT1hKg5pnBHKPLsAe7NxBX4e7k9UTaBH	1st Receiving Address	2021-12-20 23:31:44
		bc1qe0nvsmlhdv7mqc3luraf2c59z55mkcf8mx7y0g	2st Receiving Address	2021-12-21 17:10:40
	Not_Encrypted_Wallet (X)	bc1qcrk8sjvy220qezmzmu93yqgpscqn3fzapjp57r	1st Receiving Address	2021-12-20 23:32:14
		37igGpwH1onE4WjNLuqPGMaajG4gUs45k9	2st Receiving Address	2021-12-20 23:32:27
Bither	- (O)	12AfhVnsd21wtPuVzGjpbnoZnjqNBS4qfY	-	2021-11-03 11:06:59
		1GHdQycX5Mx4pUaPTfUoXxzq4X9Eo1MZSa	-	2021-11-03 12:27:02
Bitpay	Encrypted_Wallet (O)	1HTfhV5vHeCiPG7BBpRQKJvhjsSBygNhoy	-	2022-01-01 16:41:12
Electrum	Encrypted_Wallet (O)	18HircSwPxZupbgHKmgt6GaYQDS6mBWa97	1st Receiving Address	2021-12-28 22:50:57
		1CW6iqX4WejyJMCrqhCgMdr7a182K4wBcS	2st Receiving Address	2021-12-28 22:52:41
	Not_Encrypted_Wallet (X)	19CodTs7LhpvKEvDR68saQxsN4jKdbQPo4	1st Receiving Address	2021-12-28 22:52:23
		1GRmAnEgq1YrStYCDsQQz5eFD1WWzMiBrL	2st Receiving Address	2021-12-28 22:53:05

〈Table 3〉 Transaction Data

Wallet Service	Transaction Hash	Hash(MD5)
Bitcoin Core	e0c00e04230b1ebf2c7a0f092a0bddd4118946a7387562b33d3dfcf7077cd1fe	2021-12-28 22:30:13
	733d188ab427182ef60ffc53a2453791bb02fed865c649f086d3dc0ae87c5258	2021-12-29 21:43:20
Bitpay	ba4a418f89775e09c64b5772a6b45f47deb5d8184962a24fa0914f76911ddb02	2022-01-01 17:30:09
Bither	26evXKWjZSix9MwECiuf81CSSru69fmoMmdRF88qznkR	2021-11-28 16:16:24
	EjPKQZDH7BvcSc7Eh8AwDsZdGsZrFhKEMvz4ZzEceszt	2021-11-28 16:44:10
	J9hppDqjn7sN62tCeU9iST8sbMAZsEC7KaxAzXQKBQt7	2021-12-28 22:19:22
	97VTjJ9kus86u2Ta3iZdYdBpKZSFarMXGUPyVh8M12Uo	2021-12-28 22:55:27
	6wmq9a4fWsQa5pYaRGK1Y5ESnbkFsQb4THTtRhDhvzrE	2021-12-29 21:43:39
	C9bvzDwEBgjPNFXxjEwKtjhswzfFc4FgWwkvv8K3rQH	2022-01-01 17:26:52
Electrum	79523e067285d73c0045ba299ec4c140d004e4b4285451c3963347caff3af858	2021-12-28 22:55:15
	102859154c818d0239b5754d1402f1e72798e560d09dc88430f7e28d3a298778	2021-12-28 23:16:24

〈Table 4〉 Non-Encrypted Data Area(Filesystem / Memory) for each Encryption Options used case

Wallet Service	Encryption	Non-Encrypted Data Presence Area		
		Receiving Address Info	Transaction Info	Description Info
Bitcoin core	O	F	M	F
	X	F	M	F
Bither	O	F	F	F
Bitpay	O	M	M	M
Electrum	O	M	M	M
	X	F	F	F

이 아닌 별도의 실행 파일을 통해 로컬 환경에 설치하여 활용하는 방식으로 진행하였다. 이는 지갑 서비스 프로세스의 행위 모니터링 및 관련 데이터 분석이 용이하다는 장점을 가진다.

분석에 활용한 총 4가지 종류의 지갑 서비스에 대한 상세 정보는 〈표 1〉과 같다.

4.2 분석 데이터

구축한 실험 환경 내에서 모든 지갑 서비스에 대하여 동일한 절차를 거쳐 데이터를 생성하였다. 서비스별 특징이 상이하여 일부 기능이 사용 불가능하였던 경우를 제외하고, 모두 같은 기능을 활용하여 샘플 데이터를 확보하였다. 가장 먼저 지갑을 생성할 때에, 암호화 옵션 존재 여부에 따라 1개 또는 2개의 지갑을 생성하였다. 암호화가 강제되었던 서비스의 경우에는 1개의 지갑을 생성하였고, 그 외의 서비스는 암호화 활성화/비활성화 속성으로 지갑을 각 1개씩 생성하였다. 추가로 지갑 생성 시 지갑명 설정이 가능한 경우 모두 구별 가능한 값으로 설정하였으며, 여러 가상화폐의 사용을 지원하는 경우에 모두 비트코인용 지갑만을 별도로 추가 생성하였다.

이후 각 서비스에서 생성된 지갑별로 주소를 최대 2개 생성하였으며, 주소 생성 시 별칭 부여의 기능이 존재하는 서비스의 경우 모두 부여하였다. 마지막으로 모든 지갑 서비스 내에서 테스트넷이 아닌 메인넷 환경으로 실거래를 발생시켰으며, 각 거래의 발생 여부를 분석하기 위한 목적이므로 입금과 출금 행위를 구분하지 않았다.

각 지갑 서비스에서 설정한 지갑명과 주소 생성 시 부여한 별칭은 모두 동일한 값으로 설정하였다. 전체적으로 생성된 지갑 서비스별 데이터는 〈표 2〉, 〈표 3〉과 같다.

4.3 아티팩트 분석 방안

본 논문에서는 대상 기기 내의 지갑 서비스 아티팩트 분석을 위해 크게 파일시스템과 메모리 영역에 대한 분석을 진행하였다. 지갑 서비스는 거래 주소 생성 및 거래 발생 등의 이벤트에 관련한 데이터를 %APPDATA% 경로 하위 디렉토리에 저장한다. 서비스를 실행시켜 UI를 통해 확인 가능한 주소 생성 이력과 시각, 거래 내역 등은 모두 해당 서비스가 설치된 기기의 파일 시스템에 저장되어, 이후 서비스 프로세스에 의해 해석되어 보여진다. 따라서 분석하고자 하는 아티팩트 데이터가 저장되는 지갑 데이터 파일 경로를 가장 먼저 식별하고, 이후 프로세스가 종료된 시점에 해당 파일을 분석할 수 있다. 그러나 지갑을 암호화하여 생성하면, 해당 경로에 지갑 데이터가 암호화된 형태로 저장되는 경우가 있으며, 서비스별로 암호화되는 지갑 데이터의 범위 또한 상이하다.

현재 대부분의 가상화폐 지갑 서비스는 사용자에게 지갑 암호화 옵션을 권장하는데, 이는 해당 기기(관련 데이터 포함)가 탈취되었을 때에 비인가자에 의한 가상화폐 송금을 방지하기 위함이다. 정상적인 거래를 위해서는 Transaction Data에 송금 측 거래주소에 매핑되는 개인키로 서명을 해야 한다. 그러나 지갑의 암호화는 사용자가 등록한 패스워드를 통해 개인키를 암호화하여 저장하므로 올바른 인증 없이는 개인키의 복호화에 실패하여 송금이 제한된다. 이러한 목적으로 지갑의 암호화 시에 개인키의 암호화는 필수적이지만, 일부 서비스는 이를 포함해 생성 주소 및 시각 등의 지갑 데이터 파일 전체를 암호화하여 저장한다. 이러한 경우에는 파일시스템에서의 아티팩트 분석이 불가능하고, 메모리 덤프를 활용하는 방안으로 접근해야 한다.

파일시스템상에서 암호화 된 지갑 데이터 파일은 프로세스가 해당 데이터를 참조하는 시점에 복호화된 평문의 형태로 메모리(RAM)에 존재하게 된다. 따라서 지갑 서비스 프로세스를 실행시켜 파일시스템상의 지갑 데이터를 읽어들이는 것을 확인한 이후에, .vmem 파일을 확보하여 분석한다. 지갑의 암호화가 사용자에게 선택적인 서비스의 경우엔, 암호화되지 않은 지갑을 생성하여 파일시스템에 저장되는 평문 데이터 구조를 확인한다. 이후 이를 활용하여 메모리 덤프 내에서의 패턴 기반 탐색을 진행한다. 그러나 파일시스템상에서 아티팩트 데이터의 평문 구조를 확인할 수 없는 경우에는 확보한 메모리 덤프 내에서 관련 데이터의 저장 패턴을 직접 분석하였다.

각 서비스별 아티팩트 데이터가 분석 대상 기기 내 저장되는 경로와 구조를 확인한 이후, 최종적으로는 이에 대한 추출을 자동화 하는 스크립트를(WalletExtractor) 작성하였다.

〈Table 5〉 Path of Wallet Files

Wallet Service	Path	File Format
Bitcoin Core	%Appdata%\Bitcoin\wallets\[wallet_name]\wallet.dat	Hex(raw)
Bither	%Appdata%\Bither\wallets\address.db	SQLite3
	%Appdata%\Bither\wallets\bither.db	
Bitpay	%userprofile%\.bitpay\app\Local Storage\leveldb\[index].log	Log
Electrum	%Appdata%\Electrum\wallets\[wallet_name]	JSON

Ⅴ. 지갑 서비스별 아티팩트 분석 결과

파일시스템에서는 지갑의 생성 및 이후의 모든 사용에 대한 데이터가 저장되는 로컬 디스크상의 경로를 식별하였다. 일부 서비스에서는 지갑의 암호화 여부에 따라 데이터의 저장 방식이 상이한 경우가 있다. 암호화된 지갑인 경우에도 분석 대상 아티팩트 데이터가 모두 평문으로 파일시스템에 존재하는 경우가 있고, 그렇지 않은 경우에는 메모리 덤프상에서만 평문으로 확인할 수 있다. 각 서비스별 지갑의 암호화 여부에 따라 아티팩트 데이터가 평문으로 존재하는 영역에 대한 분류 결과는 〈표 4〉와 같다. Bitcoin Core 및 Electrum 서비스는 사용자에게 지갑 암호화 선택 옵션을 제공하지만, Bither, Bitpay 서비스는 모든 경우에 지갑 암호화를 강제한다.

지갑 데이터를 암호화하여 디스크에 저장하는 대다수의 경우엔 사용자가 설정한 인증 패스워드에 기반하여 특정 암호(AES-256-CBC 등)에 활용할 KEY, IV(Initial Vector)를 생성한다. 이후 사용자가 지갑에 접근할 때마다 입력하는 패스워드를 통해 KEY를 생성하여 암호화/복호화의 유효성을 검증하기 때문에 별도의 인증 정보 없이 암호화된 파일을 임의로 복호화하는 것은 제한되었다. 하지만 암호화가 되지 않은 지갑을 생성한 후에 지갑 데이터의 평문 저장 패턴을 분석하여 메모리 덤프상에서 검색한 결과, 암호화된 지갑의 데이터를 동일한 구조의 평문 형태로 모두 식별할 수 있었다.

서비스별 지갑 데이터가 저장되는 파일의 경로는 〈표 5〉와 같다. 각 파일 포맷은 데이터베이스(.db), json, log 파일 등으로 다양하게 존재하였다.

5.1 거래 주소 및 생성 시각

각 서비스별로 비트코인 지갑을 생성한 후에 관리하는 거래 주소 포맷이 정해져 있으며, 대부분은 사용자의 지갑 생성 시 속성값으로 설정할 수 있다. 하나의 서비스 내에서는 주소 포맷별로 매번 고정된 길이의 주소가 생성되며, 거래 주소 아티팩트 데이터를 추출한 이후에 일차적인 검증 수단으로 활용할 수 있었다. 또한 모든 서비스에서 지갑 내 거래 주소 데이터를 식별하기 위한 특정 문자열들이 존재한다. (Bither 서비스 내 데이터베이스 형식의 지갑 데이터 파일의 경우에는 주소에 대한 특정 컬럼명을 확인하였다). 지갑 데이터 파일 또는 메모리 덤프에서 서비스별 주소 식별자를 검색하여, 연이어 저장되는 평문 형태의 거래 주소를 확인할 수 있다. 본 연구에서는 실제 생성된 거래 주소에 초점을 두어 분석하였으나, Electrum 서비스의 경우에는 사용자에게 순차적으로 발급될 주소 30개를 미리 생성하여 관리하기 때문에, 향후 추가적으로 활용 가능한 주소 정보도 획득할 수 있다.

각 서비스를 통해 실제 사용자가 생성한 주소를 식별하면, 이와 동일한 데이터 블록에서 주소별 생성 시각을 확인할 수 있다. 모든 서비스에서 생성 시각은 Unix Time 포맷으로 저장되어 있다. 이와 더불어, Bither 및 Electrum 서비스의 경우에는 거래 주소의 경우와 유사하게, 해당 데이터 블록에서 생성 시각을 식별하기 위한 다른 특정 문자열들이 존재한다. 반면 Bitcoin Core 서비스는, 별도의 문자열 없이 지갑 데이터 파일에서 거래 주소의 위치보다 4바이트 이전에 16진수의 Little Endian 방식으로 시각 데이터(4바이트)를 저장하며, Bitpay 서비스는 특정 주소에 대한 생성 시각을 별도로 관리하지 않는다.

서비스별 거래 주소 및 생성 시각 아티팩트에 대한 분석을 종합한 결과는 〈표 6〉과 같다.

5.2 거래 내역

특정 지갑 내 거래 주소를 기반으로 발생시킨 입출금 거래에 관한 데이터는, 해당 지갑 데이터 파일에 함께 저장되는 경우와 별도의 거래 데이터 파일로 존재하는 경우로 나뉘었다. Bitpay 및 Electrum 서비스의 경우에는,

〈Table 6〉 Receiving Address and Creation Time Data Format for Each Service

Wallet Service	Artifact	Data Stored Format or Path (BTC Address Format)		
Bitcoin Core	Receiving Address	**purpose*** + {Address} (P2WPKH)		
		purpose" + {Address} (P2SH)		
	Creation Time	Null(4 bytes) + {Time} + "01 00 00 00" + {Address}		
Bither	Receiving Address	**address.db** (database file)	**addresses** (table name)	**address** (P2PKH)
	Creation Time			**sort_time**
Bitpay	Receiving Address	**txsHistory**-{Wallet_ID}…**"address"**:"{Address}" (P2WPKH, P2PKH)		
	Creation Time	-		
Electrum	Receiving Address	**"payment_requests"**: {…"{Address}": {…}, (P2WPKH, P2PKH)		
	Creation Time	**"payment_requests"**: {…"{Address}": {…**"time"**:{Time}		

지갑 데이터 파일 내에서 거래 정보를 식별할 수 있는 특정 문자열과 함께 저장되었다. Bither 서비스는 별도 파일(.db)에 거래 내역 및 시각 등의 정보를 저장하며, Bitcoin Core 서비스는 블록체인 데이터를 블록 단위(blk.dat)로 저장 및 관리한다. 또한 발생한 거래와 관련하여 저장되는 정보 유형은 모든 서비스에서 거의 동일하였으며, 거래 해시 및 거래 시각은 공통적으로 포함되어 있다. 이 중 거래별 구분자로 활용되는 거래 해시는 서비스에 따라 해시 또는 거래 아이디(Transaction ID)로 표현되었다.

Bitcoin Core를 제외한 모든 서비스는 파일시스템 내에서 거래 해시 또는 시각을 식별하기 위한 특수 문자열이 존재하였으나, 해당 서비스가 관리하는 블록 데이터 단위에서는 이를 확인할 수 없었다. 하지만 메모리 덤프에서 거래 해시와 시각 정보가 각각 "Transaction ID", "Date" 의 문자열과 함께 존재하였고, 일반 텍스트가 아닌 유니코드 문자열의 형태로 저장되었는데, 이는 서비스 UI를 통해 해당 정보가 드러나는 시점에서의 데이터로 확인하였다. 따라서 이 경우에 거래 시각은 UTC Time 포맷으로 메모리에 존재하며, 기타 모든 서비스는 거래 시각이 모두 Unix Time 포맷으로 파일 시스템에 존재한다.

또한 Electrum 서비스는 기타 서비스들과 다르게, 거래 해시를 위한 식별자(문자열)만 존재할 뿐 거래 시각을 위한 별도 식별자는 존재하지 않는다. 지갑 데이터 파일의 특성(json)에 따라, 거래 해시값을 키(Key)로 하여 이에 따른 내부 데이터가 존재하는데, 해당 데이터(list)의 2번째 원소가 거래 시각을 나타낸다.

서비스별 거래 내역 아티팩트에 대한 분석을 종합한 결과는 〈표 7〉과 같다.

5.3 지갑명 및 주소 별칭

〈표 5〉에서 확인할 수 있듯이, 일부 서비스에서 지갑 데이터 파일은 해당 지갑명을 포함한 절대 경로에 생성된다. 이러한 경우에 대상 기기의 파일 시스템에 존재하는 파일 또는 디렉토리의 경로를 확인하는 것으로 지갑명을 식별할 수 있다. 그 외 Bitpay 서비스는 지갑명을 특정 경로를 통해 확인할 수 없지만, 지갑 데이터 파일 내에 이를 식별하기 위한 특수 문자열과 함께 저장된다. 또한 이는 해당 파일 내에서 지갑별 구분을 위한 지갑 ID 값과 함께 연이어 위치하며, 거래 내역에 지갑 ID가 포함되는 점과 연계하여 지갑명을 특정 거래 정보에 대응시킬 수 있다. 마지막으로 Bither 서비스는 별도의 지갑명 또는 주소 별칭을 관리하지 않으며, 지갑명을 대신하여 지갑 주소로 HD_Address 값을 사용한다.

특정 지갑 내 여러 개의 주소 관리를 위한 목적으로 사용되는 주소 별칭 기능은 Bitcoin Core 및 Electrum 서비스에서만 사용 가능하다. Electrum 서비스는 지갑 데이터 파일 내 주소별 별칭을 관리하는 데이터 영역을 가리키는 특수 문자열이 존재한다. 해당 파일의 특성(json)에 따라, 식별자("labels")를 키(Key)로 하여 이에 대한 내부 데이터가 존재하며, 해당 데이터(Dictionary)의 새로운 키(Key)인 거래 주소에 대한 별칭(Data)이 저장되는 구조를 가진다.

〈Table 7〉 Transaction History Data Format for Each Service

Wallet Service	Artifact	Data Stored Format or Path		
Bitcoin Core	TX Hash	Transaction ID: {Hash}(Transaction Total Size: …		
	TX Time	Date: {Time}(From: …		
Bither	TX Hash	bither.db (database)	txs (table)	tx_hash
	TX Time			tx_time
Bitpay	TX Hash	txsHistory-{Wallet_ID}…(37 bytes)"txid":"{Hash}"		
	TX Time	txsHistory-{Wallet_ID}…(164 bytes)"time":{Time}		
Electrum	TX Hash	"verified_tx3" : {…"{Hash}": […"		
	TX Time	"verified_tx3" : {…"{Hash}": […, {Time}, …]}		

〈Table 8〉 Wallet Name and Description for Each Service

Wallet Service	Artifact	Data Stored Format or Path
Bitcoin Core	Wallet Name	%Appdata%\Bitcoin\wallets\[Wallet_Name]\wallet.dat
	Description	{Time} + {Address} + "\x15" + {Description}
Bither	Wallet Name	-
	Description	
Bitpay	Wallet Name	"walletId":"{Wallet_ID}","walletName":"{Wallet_Name}"
	Description	-
Electrum	Wallet Name	%Appdata%\Electrum\wallets\[Wallet_Name]
	Description	"labels": {…"{Address}": {Description},… }

Bitcoin Core 서비스는 이와 달리, 식별자로 활용되는 별도의 문자열 없이 지갑 데이터 파일 내에서 거래 주소 및 생성 시각과 함께 연이어 저장되는 구조를 가진다. 4바이트 형태의 생성 시각 데이터와 거래 주소를 먼저 식별한 이후에, 해당 주소에 대한 별칭을 확인할 수 있다.

서비스별 지갑명 및 주소 별칭 아티팩트에 대한 분석을 종합한 결과는 〈표 8〉과 같다.

Ⅵ. WalletExtractor 개발 및 활용 방안

본 연구에서는 수사 과정에서 압수한 대상 디지털 기기를 분석하는 상황을 가정하여, 대상 내 파일시스템 및 지갑 서비스의 메모리 영역을 분석하였다. 구체적으로 특정 거래가 해당 기기 내에서 발생하였음을 입증하기 위한 3가지 아티팩트에 대하여, 각 지갑 서비스가 생성하고 관리하는 관련 데이터를 분석하였다. 이후 해당 결과를 기반으로, 분석 대상 기기의 파일시스템 및 메모리 덤프에 적용 가능한 지갑 아티팩트 추출 자동화 스크립트를 개발하였다.

6.1 WalletExtractor 개발 결과

각 지갑 서비스가 데이터를 생성하는 경로 및 저장하는 구조를 파악하여, 이에 대한 서비스별 검색 패턴을 구성하였다. 이후, 이를 정규식 형태로 변환하여 Python 스크립트에 반영하였다. 본 연구에서 대상으로 한 대표적인 지갑 서비스 4가지 이외에, 향후 기타 서비스들의 데이터 저장 패턴을 모듈로 추가하며 범용성을 확대할 수 있다. 이를 통해 파일시스템을 분석할 경우, 대상 기기 내에서 서비스명을 인자로 하여 직접 실행할 수 있다. 또한 메모리 분석을 진행할 경우는, 확보한 메모리 덤프 파일 경로를 서비스명과 함께 인자로 전달하여 별도의 분석 PC에서 실행할 수 있다.

본 연구에서 WalletExtractor를 적용한 결과, Bitcoin Core 서비스 내 한 가지 아티팩트 데이터(거래 내역)를 제외하고 모든 지갑 서비스의 아티팩트 추출을 자동화하였다. 실패 경우에 대한 원인은 파일시스템 내 해당 데이터의 경로를 식별하였지만, 이에 대한 분석이 아닌 메모리 덤프 내 프로세스의 관련 흔적을 추적하는 방안을 활용하였기 때문이다. UI를 통해 보이는 데이터가 메모리에 존재하는 경우에는, UI에서의 화면 또는 기능이 바뀔 시에 이전 데이터가 덮어지며 데이터의 추출 성공률이 낮아진다. 이에 대한 개선으로 Bitcoin Core 서비스의 블록체인 데이터 파일을 분석하여 내부 데이터 구조를 이해하고 서비스 아티팩트를 해당 영역에서 분석할 수 있다면, WalletExtractor를 개선하여 더 넓은 방면에 활용 가능할 것이다.

추가로 프로세스 메모리에 대한 분석의 경우, 대상 기기의 메모리 덤프를 확보하는 시점에 따라 그 데이터가 상이하다. 또한 지갑 서비스 프로세스가 종료된 이후 시점에서는, 메모리 내 해당 프로세스 영역에서의 잔존 데이터가 희박할 것이다. 이렇게 다양한 압수 시점에서의 메모리 데이터에 대한 분석 성공률을 높이기 위해서, 각 서비스별 아티팩트 데이터 검색 패턴을 더욱 정교하게 구성해야 한다. 가령 온전한 문자열 형태의 거래 주소나 주소 별칭이 존재하지 않고 일부 훼손된 경우에, 여러 데이터 블록에서의 아티팩트 추출 결과를 부분적으로 종합하여 분석하는 기능도 고려할 수 있다.

6.2 WalletExtractor 활용 방안

다양한 지갑 서비스의 아티팩트 검색 패턴이 반영된 WalletExtractor는 수사 과정에서 활용되어 범죄행위 추적에 기여할 수 있다. 압수 현장에 출동하여 수사관은 일차적으로 현장에 위치한 디지털 기기의 디스크 또는 메모리 덤프를 확보한다. 이후 별도의 분석 환경에 덤프 데이터를 마운트 혹은 업로드하여 WalletExtractor를 동일 환경 내에서 실행한다. 이때, WalletExtractor 내 등록된 서비스별 패턴을 기반으로 지갑 서비스 사용 여부를 검사하고, 특정 지갑을 활용한 사용자의 행위 데이터를 자동 추출한다. 그 결과로 확보한 거래 해시 또는 거래 주소 등의 데이터를 블록체인 데이터 웹 서비스에 조회하여 유효성을 검증하고, 추출 결과에 따른 사용자의 가상화폐 사용 내역을 작성한다. 또한 지갑 데이터 파일의 MAC(Modify, Access, Create) Time을 분석하여 수집한 증거에 대한 무결성을 검증할 수 있다. 서비스 이용에 따라 지갑 데이터 파일의 수정 시각은 계속 갱신되지만, 생성 시각은 최초 앱을 설치하여 지갑을 생성한 최초의 시각으로 고정된다는 점을 활용할 수 있다. 따라서 Windows 이벤트 로그 및 레지스트리에 대한 추가 분석을 통해 서비스 설치/사용 일시 등에 대한 정보를 수집하여, 사용 내역에 대한 근거를 확보할 수 있다.

최종적으로 클라이언트 기기에서만 식별 가능한 주소 생성 시각, 주소 별칭 등의 정보를 용의자의 동선, 통신 이력 등의 기타 정보와 결합하여 범죄 타임라인을 재구성하고, 필요시에 수사 범위를 확대할 수 있을 것이다.

Ⅶ. 결 론

가상화폐의 종류와 그 활용 범위는 지속적으로 확대될 것이며, 이에 따라 필연적으로 가상화폐별 특성을 고려한 다양한 서비스에 대한 분석 연구가 필요하다. 특히 사용자의 입장에서 활용 가능한 지갑 서비스의 종류와 방식이 다양해짐에 따라, 사용자 행위 추적을 위한 클라이언트 서비스 아티팩트 분석은 그 중요성이 더욱 높아질 것이다.

본 연구에서 진행한 아티팩트 분석은 대상 서비스의 특성에 의존하지 않고 다양한 클라이언트 서비스 분석에 전반적으로 활용 가능하다. 이를 활용한 유사 연구들이 다수 진행된다면, 범죄 목적의 가상화폐 활용을 수사하는 과정에서 확실한 증거 수집 절차로의 정립으로 발전할 수 있을 것이다.

참 고 문 헌 (References)

[1] S. Meiklejohn, M. Pomarole, K. Levchenko, D. McCoy, G. M. Voelker, and S. Savage, ''A fistful of Bitcoins: Characterizing payments among men with no names,'' in Proc. Conf. Internet Meas. Conf. (IMC), New York, NY, USA, 2013, pp. 127-140.

[2] M. Spagnuolo, F. Maggi, and S. Zanero, ''BitIodine: Extracting intelli- gence from the Bitcoin network,'' in Financial Cryptography and Data Security. New York, NY, USA: Springer, 2014, pp. 457-468.

[3] Gurkok, C. 'bitcoin.py'. https://github.com/volatilityfoundation/community/ blob/b4d65bd01870299c8c08e1e11b7b70dfe96cd1dd/CemGurkok/bitcoin.py, 2015

[4] Haigh, T., Breitinger, F., Baggili, I. If I Had a Million Cryptos: Cryptowallet Application Analysis and a Trojan Proof-Of-Concept, pp. 45-65, 2019.

[5] A. Montanez, ''Investigation of cryptocurrency wallets on iOS and Android mobile devices for potential forensic artifacts,'' Dept. Forensic Sci., Marshall Univ., Huntington, WV, USA, Tech. Rep., 2014.

[6] Van Der Horst, L., Choo, K.-K.R., Le-Khac, N.-A. Process Memory Investigation of the Bitcoin Clients Electrum and Bitcoin Core, 2017.

[7] Zollner, S., Choo, K.-K.R., Le-Khac, N.-A. An automated live forensic and postmortem analysis tool for bitcoin on windows systems. IEEE Access 7, 158250-158263, 2019.

[8] Tyler Thomas, Mathew Piscitelli, Ilya Shavrov, and Ibrahim Baggili. Memory FORESHADOW: Memory FOREnSics of HArDware CryptOcurrency Wallets - a tool andvisualization framework. Forensic Science International: Digital Investigation, page301002, 2020.

[9] AhnJungUn, WalletExtractor. https://github.com/AhnJungUn/WalletExtractor

저 자 소 개

안 정 언 (Jungun Ahn)
준회원

2018년 2월 : 고려대학교 정보보호학부 졸업
2019년 9월 - 현재 : 고려대학교 정보보호학과 석박사통합과정
관심분야 : 디지털 포렌식, 정보보호 등

이 상 진 (Sangjin Lee)
평생회원

1989년 10월 - 1999년 2월 : 한국 전자통신연구원 선임연구원
1999년 3월 ~ 2001년 8월 : 고려대학교 자연과학대학 조교수
2001년 9월 ~ 현재 : 고려대학교 정보보호대학원 교수
2017년 3월 ~ 현재 : 고려대학교 정보보호대학원 원장
관심분야 : 대칭키 암호, 정보은닉 이론, 디지털 포렌식