Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.28, NO.6, Dec. 2018 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2018.28.6.1319

기계학습을 활용한 이더리움 미확인 스마트 컨트랙트 자동 분류 방안*

이 동 건,† 권 태 경‡
연세대학교 정보보호연구실

Automated Classification of Unknown Smart Contracts of Ethereum Using Machine Learning*

Donggun Lee,† Taekyoung Kwon‡
Information Security Lab., Graduation School of Information, Yonsei University

요 약

암호화폐를 위해 개발된 블록체인 시스템은 탈중앙화, 분산원장 및 부분적 실명은닉성의 특징을 가지고 있어 최근 다양한 분야에서 적용이 시도되고 있다. 그 중 부분적 실명은닉성은 사용자 프라이버시를 강력히 보장하지만 범죄악용 등 부작용 또한 나타나고 있어 이를 공격하기 위한 방안들이 지속 연구되어 왔다. 본 연구에서는 2세대 암호화폐의 대표인 이더리움 블록체인 시스템에서의 사용자 행위 식별을 위해 기계학습을 활용한 미확인 스마트 컨트랙트 기능 및 디자인 패턴의 자동 분류 방안에 대하여 제안한다.

ABSTRACT

A blockchain system developed for crypto-currency has attractive characteristics, such as de-centralization, distributed ledger, and partial anonymity, making itself adopted in various fields. Among those characteristics, partial anonymity strongly assures privacy of users, but side effects such as abuse of crime are also appearing, and so countermeasures for circumventing such abuse have been studied continuously. In this paper, we propose a machine-learning based method for classifying smart contracts in Ethereum regarding their functions and design patterns and for identifying user behaviors according to them.

Keywords: Blockchain, Ethereum, Smart contract, De-anonymity, Forensics

I. 서 론

2008년 S. Nakamoto에 의해 개발된 블록체인(blockchain) 시스템[1]은 2009년 1월 최초의 암호화폐인 비트코인(bitcoin)의 발행을 시작으로 2015년 스마트 컨트랙트(smart contract)로 대표되는 2세대 암호화폐 이더리움(ethereum)[2]을 거쳐 최근에는 인증, 금융결제 및 공공분야까지 적용이 시도되는 등 4차 산업 혁명시대에 관심이 집중되고 있는 시스템이다.

블록체인 시스템은 최초 제안시 암호화폐를 위해 설계되었기에 탈중앙화(de-centralization), 분산원장

Received(08. 27. 2018), Modified(10. 18. 2018), Accepted(11. 09. 2018)
* 본 연구는 과학기술정보통신부 및 정보통신기술진흥센터의 대학ICT연구센터육성지원사업의 연구결과로 수행되었음(IITP-2018-2016-0-00304). 이 논문은 2018년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임(No. 2017-0-00380, 차세대 인증 기술 개발)
† 주저자, c15336@yonsei.ac.kr
‡ 교신저자, taekyoung@yonsei.ac.kr(Corresponding author)

(distributed ledger) 및 실명은닉성(anonymity)을 특징으로 가지고 있으며, 이중 실명은닉성은 공개키(public key)를 사용자 아이디로 사용하여 사용자에게 강력한 프라이버시(privacy)를 보장하지만 사기범죄, 마약거래 및 탈세 등에 활용되는 부작용 또한 나타나고 있다.

이와 관련하여 블록체인 시스템이 도입된 이후 실명은닉성에 대한 공격 방법이 지속적으로 연구되어 왔으며 특히, 최초의 암호화폐인 비트코인에 대해서는 시스템 내부적으로 휴리스틱(heuristics)한 분석을 통해 동일사용자의 주소들을 클러스터링(clustering)하는 방법과 시스템 외부적으로 비트코인 연동 결제 시스템에서 사용자 데이터를 유출시키는 방법 등이 연구되어 왔다.

하지만, 이오스(EOS) 및 스테이터스 네트워크 토큰(status network token) 등 많은 가상화폐들의 기반 블록체인 시스템이자 다양한 스마트 컨트랙트를 구현 할 수 있는 2세대 암호화폐인 이더리움에 대해서는 현재 암호화폐 시가총액 2위의 위상과 달리 실명은닉성 관련 연구가 부족한 실정이다.

본 연구에서는 이더리움 스마트 컨트랙트가 가지는 특성을 활용하여 확인되지 않은(unknown) 스마트 컨트랙트의 기능을 기계학습(machine learning)으로 자동 분류(classification)하는 방안을 제시하며, 이를 통해 확인된 스마트 컨트랙트와 거래한 특정한 사용자에 대하여 블록체인 내에서의 행위 판단이 가능토록 한다.

II. 블록체인 실명은닉성 공격 관련 연구

2.1 비트코인

E. Androulaki 등은 비트코인 UTXO(Unspent Transaction Output) 거래 시스템이 가지는 특징을 이용하여 휴리스틱한 방법으로 동일 사용자의 주소들을 클러스터링 하였다[3]. OTC(One-Time Change)는 사용자가 타인에게 송금이후 남은 잔액을 새로운 주소로 돌려받도록 시스템이 권장하는 것을 이용하여 송금한 주소와 직후 새로 생성된 주소를 동일사용자로 판단하는 것이다. CS(Common Spending)는 일반적인 거래에서 다수의 주소에서 거래 잔액이 없이 한 개의 주소로 송금하는 것은 흔치않으며, 이는 한 명의 사용자가 여러 주소에 분산 보관중인 비트코인을 한 곳으로 모으는 것으로 간주하고 동일사용자의 주소로 판단한다.

S. Meiklejohn 등은 사용자의 비의도 유출을 이용한 방법으로 커뮤니티 및 SNS 등 공개된 공간에 비트코인 사용자들이 개인 송금, 기록 및 기부 등의 목적으로 남긴 게시글을 수집하여 사용자의 실제 이름(real name)과 비트코인 주소를 연결시키는 공격 방법을 제시하였다[4].

M. Fleder 등은 OTC 및 CS를 이용한 사용자 주소 클러스터링과 커뮤니티 및 SNS의 사용자 비의도 유출 분석을 통해 2013년 체포된 마약 등 불법물품 비트코인 암거래 웹 사이트 실크로드(silk road)의 운영자 주소를 찾아내었다[5].

S. Goldfeder 등은 사용자가 웹 사이트에서 물품구매를 위해 비트코인을 결제한 정보가 담긴 쿠키들을 트래커(tracker) DB를 통해 수집하고 이와 연결된 비트코인 블록체인 시스템의 트랜잭션(transaction)을 분석함으로서 동일 사용자의 주소들을 확인하였으며[6], P. Koshy 등은 비트코인 블록체인 시스템에서 사용자가 거래시 연결된 노드(node)의 트래픽(traffic)을 모니터링 하여 30%의 정확도로 비트코인 주소와 IP 주소를 연결시켰다[7].

한편, 이러한 실명은닉성 공격 방안에 대응하여 사용자 프라이버시를 보장하기 위해 CoinJoin[8] 등과 같은 거래 입출금(transaction input / output) 내역을 섞는 방법(Peer-to-peer mixing protocols)과 MixCoin[9] 등과 같은 별도의 노드를 통해 거래 자체를 섞는 방법(Distributed mixing networks)이 제안되었다.

2.2 이더리움

이더리움은 비트코인과 달리 UTXO 거래 시스템을 사용하지 않아 OTC 및 CS를 이용한 주소 클러스터링이 불가능하며, 2015년에 발표되어 비트코인에 비해 상대적으로 역사가 짧아 실명은닉성 공격 관련 연구가 미흡한 실정이다.

J. Payette 등은 250,000개의 이더리움 사용자 주소에 대하여 Total Ether, Total Number of Transactions 및 Transaction per Month 등 일반적으로 수집할 수 있는 34종류의 자질(features) 세트를 K-means 알고리즘으로 클러스터링 하여 4개의 클러스터로 분류하였다[10]. 이더리움 주소가 클러스터링이 가능한 것을 보여 주었으나 동일 사용자 주소 수준의 클러스터링은 제한되었다.

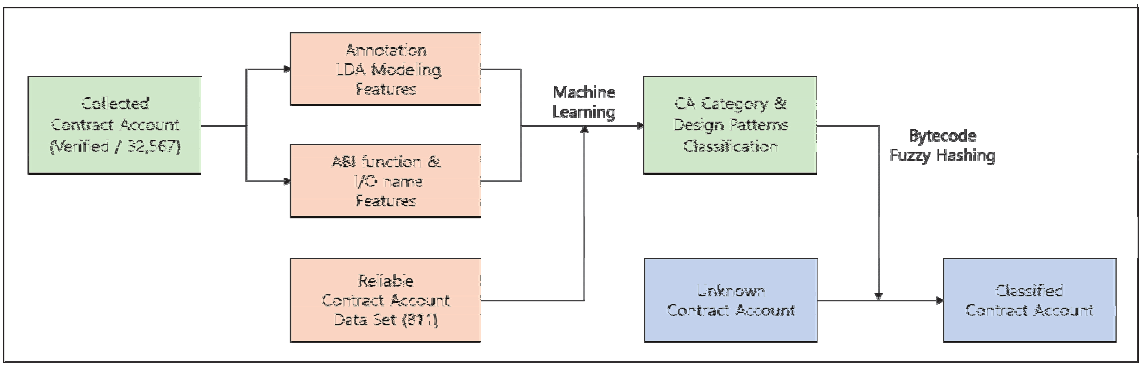


Fig. 1. Automated Classification Model

R. Norvill 등은 이더리움 스마트 컨트랙트를 분석하기 위해 936개의 CA(Contract Account)를 수집하고 CA의 이름을 토큰화(tokenize)한 1,803개의 네임워드(name words)를 Affinity Propagation 및 K-medoids 알고리즘으로 클러스터링 하여 분석하였으며, CA의 bytecode를 hashing 하여 확인되지 않은 CA를 특정한 클러스터에 포함시키는 방법을 제안하였다[11]. 하지만, 결과를 CA 이름에 의존하여 클러스터가 단순하게 구성되었고 각 클러스터에 속한 CA 기능의 명확한 확인이 제한되었다.

2.3 기존 연구와의 차이점

이더리움 블록체인 시스템에서의 사용자 행위는 사용자간 단순 이더리움 화폐 거래와 스마트 컨트랙트를 이용한 거래가 있으며, 공개된 이더리움 트랜잭션 확인을 통해 사용자가 어떤 행위를 하였는지 판단이 가능하다. 하지만, 소스코드를 공개하지 않은 스마트 컨트랙트를 이용한 거래는 기능 확인이 불가하여 사용자의 행위 판단이 불가능하다.

지금까지의 이더리움 관련 연구는 이더리움 사용자의 계정을 클러스터링 하거나 공개된 일부 스마트 컨트랙트를 매우 단순한 자질로 클러스터링 하는 비지도 학습 방향으로 수행되어 확인된 스마트 컨트랙트를 특성별로 단순하게 군집시킬 수는 있었지만 소스코드가 공개되지 않은 미확인 스마트 컨트랙트에 대한 기능 식별은 제한되었다.

본 연구에서는 스마트 컨트랙트의 기능을 사전 분류한 신뢰할 수 있는 데이터 세트(811개)를 트레이닝 세트로 활용하는 지도 학습으로 새롭게 연구를 수행하며, 큰 데이터 세트 확보를 위해 지금까지 공개된 모든 스마트 컨트랙트(32,567개)를 기능별로 자동 분류하고 이를 활용하여 미확인 스마트 컨트랙트의 기능을 확인하는 방법을 제안한다.

III. 스마트 컨트랙트 자동 분류 모델

3.1 모델 설계

스마트 컨트랙트의 자동 분류 모델(automated classification model) 설계는 Fig. 1.과 같이 소스코드가 확인된 CA(32,567개)를 수집하고 이를 대상으로 정확한 분류를 위한 주석(annotation) LDA modeling features와 ABI function & I/O name features를 구성한다.

위의 두 features sets와 사전 분류가 되어있는 reliable CA data set(811개)으로 기계학습 분류 알고리즘을 적용하여 CA 분류기(classifier)를 만들고 스케일이 큰(large scale) 수집한 CA를 대상으로 카테고리(category)와 디자인 패턴(design patterns)을 분류 및 분석한다.

최종 분류된 CA의 bytecode를 fuzzy hashing 한 data set을 바탕으로 확인되지 않은 CA의 bytecode와 비교하여 동일하거나 유사한 소스코드를 가진 CA를 찾아 카테고리와 디자인 패턴을 분류함으로 알려지지 않은 CA의 기능을 확인한다.

3.2 데이터 수집

3.2.1 컨트랙트 계정 데이터

이더리움은 etherscan.io[12] 웹 사이트를 통해 이더리움 블록(block), 거래 및 주소 등에 대한 모든 정보를 공개하고 있다. 이더리움 블록체인 시스템에

등록된 많은 CA 중에 제작자가 사용자들이 자신의 CA를 신뢰하고 이용할 수 있도록 스마트 컨트랙트의 이름과 소스코드를 공개하여 확인된 CA를 verified CA라 말한다. 본 연구에서는 확인 서비스가 시작된 2016년 3월 24일 부터 2018년 6월 26일 까지 등록된 모든 32,567개의 verified CA를 수집하였다.

3.2.2 신뢰할 수 있는 컨트랙트 계정 데이터

M. Bartoletti 등은 2017년 1월 부터 811개의 이더리움 verified CA를 수집하고 소스코드를 분석하여 수작업(manually inspect)으로 분류하였다[13]. 분류는 카테고리와 디자인 패턴 두 종류로 실시하였으며, 카테고리는 Financial 및 Notary 등 6종류로 분류하였고 디자인 패턴은 Token, Authorization 및 Oracle 등 9종류로 분류하였다.

디자인 패턴의 경우 카테고리와 달리 CA에 중복으로 포함하도록 분류하였는데 이는 CA의 기능을 구현하기 위해 다수의 코드 디자인이 동시 적용되어야 하기 때문이다.

본 연구에서는 위에서 분류한 카테고리 및 디자인 패턴으로 스마트 컨트랙트의 기능을 설명하기에 충분한 것으로 판단하여 이를 트레이닝 세트로 활용하였으며, 각 분류는 Table 1.과 같다.

Table 1. Manually Inspected Classification of Smart Contracts

Classification		CAs
Category	Financial	373
	Notary	79
	Game	158
	Wallet	17
	Library	29
	Unclassified	155
	Total	811
Design patterns	Token(DPT)	173
	Authorization(DPA)	491
	Oracle	53
	Randomness(Random)	121
	Poll(Vote)	70
	Time constraint(Time)	267
	Termination(Kill)	178
	Math	32
	Fork check(Fork)	42

3.3 자질 세트

3.3.1 주석 LDA 모델링 자질 세트

LDA(Latent Dirichlet Allocation) modeling[14]은 토픽모델링(topic modeling)의 기법중 하나로 문서(document)를 대표하는 주제(topic)를 찾고 이를 구성하는 단어(word)가 나타날 확률을 알 수 있다.

등록된 verified CA의 경우 사용자가 CA를 신뢰하고 사용할 수 있도록 하기 위해 이름과 소스코드를 제작자가 직접 등록한 것이기 때문에 사용자가 소스코드를 이해하기 쉽게 하기 위해서 일반적으로 기능을 상세히 설명한 주석이 포함되어 있다. 이 주석은 CA의 전반적인 기능과 각 함수 부분에 대한 구성요소를 포함하여 이를 LDA Modeling을 할 경우 CA 기능을 대표하는 단어들을 자동화하여 찾을 수 있다.

따라서 Fig. 2.의 LDA Modeling 결과는 CA 기능을 대표하는 단어와 확률 값으로 CA를 카테고리로 분류하기 위한 feature set으로 사용이 가능하다. 분류 알고리즘에 사용하기 위해 총 5,339개의 features로 처리하였으며, 각 feature의 값은 확률 값을 적용하였다.

	0	1	2	3	4
0	('date', 0.1)	('must', 0.1)	('call', 0.06666667)	('end', 0.06666667)	('name', 0.06666667)
1	('owner', 0.07692307)	('throw', 0.076923065)	('assert', 0.0576923)	('transfer', 0.0576923)	('newowner', 0.0576923)
2	('owner', 0.07692307)	('throw', 0.076923065)	('assert', 0.0576923)	('transfer', 0.0576923)	('newowner', 0.0576923)
3	('throw', 0.06086957)	('owner', 0.04347826)	('number', 0.04347826)	('newowner', 0.026086956)	('interface', 0.026086956)
4	('owner', 0.07692307)	('throw', 0.076923065)	('assert', 0.0576923)	('transfer', 0.0576923)	('newowner', 0.0576923)
5	('bet', 0.025358329)	('price', 0.01874311)	('owner', 0.015435502)	('use', 0.014332968)	('payout', 0.012127895)
6	('bet', 0.025358329)	('price', 0.01874311)	('owner', 0.015435502)	('use', 0.014332968)	('payout', 0.012127895)
7	('bet', 0.025358329)	('price', 0.01874311)	('owner', 0.015435502)	('use', 0.014332968)	('payout', 0.012127895)
8	('owner', 0.07692307)	('throw', 0.076923065)	('assert', 0.0576923)	('transfer', 0.0576923)	('newowner', 0.0576923)
9	('account', 0.073446326)	('transfer', 0.062146887)	('owner', 0.050847456)	('token', 0.03954802)	('spender', 0.033898305)

Fig. 2. Annotation LDA Modeling Result

3.3.2 ABI 함수 & 입출력 이름 자질 세트

등록된 verified CA의 소스코드를 분석을 통해 ABI(Application Binary Interface)를 구성하고 있는 함수(function) 이름과 각 함수에 포함되어 있는 입출력 변수(input / output variables)

	0	1	2	3	4
0	isActive	endTime	beneficiary	tokenReward	startTime
1	end_time	beneficiary	fifty_two_weeks	start_time	release
2	end_time	beneficiary	fifty_two_weeks	start_time	release
3	allowanceForContract	batchTransferToken	owner	balanceOfContract	batchTransferEther
4	end_time	beneficiary	fifty_two_weeks	start_time	release
5	winnerDeterminedDate	determineWinner	BETTING_OPENS	collectionFees	_callback
6	winnerDeterminedDate	determineWinner	BETTING_OPENS	collectionFees	_callback
7	winnerDeterminedDate	determineWinner	BETTING_OPENS	collectionFees	_callback
8	end_time	beneficiary	fifty_two_weeks	start_time	release
9	name	approve	totalSupply	transferFrom	decimals

Fig. 3. ABI Function & I/O Name

의 이름을 추출 할 수 있다.

일반적으로 프로그래머는 작업 효율성과 편의를 위해 소스코드 작성 시 함수 및 변수의 이름은 기능과 역할을 나타내도록 사용하며, 그 형태는 프로그래머의 습관에 따라 일정한 경향성을 가진다.

따라서 CA의 기능을 구현하기 위한 디자인 패턴은 함수들과 연계되어 있기에 이를 대표할 수 있는 Fig. 3.의 함수 및 변수의 이름은 디자인 패턴으로 분류하기 위한 feature set에 사용이 가능하다. 또한, 프로그래머 습관에 따른 경향성으로 이름의 반복 사용 확률이 높아 분류 성능을 향상시킬 수 있다. 분류 알고리즘에 사용하기 위해 총 13,265개의 features로 처리하였으며, 각 feature의 값은 반복 카운트로 적용하였다.

3.4 분류 알고리즘

3.4.1 랜덤 포레스트

랜덤포레스트(random forest)[15] 알고리즘은 기계학습에서 분류, 회기분석 등에 사용되는 앙상블 학습 방법의 일종으로 여러 개의 결정 트리(decision tree)를 임의적으로 학습하는 방식이다. 랜덤포레스트는 결정 트리의 과대적합 단점을 해결하며, 각 feature의 특성이 상이하여 스케일링(scaling)이 제한되는 경우에도 좋은 성능을 보인다. 모의 실험결과에서도 일반 결정 트리 및 가우시안 NB(Gaussian Naive Bayes) 등 다른 알고리즘에 비해 우수한 성능을 보였다.

본 연구에서 Reliable Contract Account Data Set을 분류하는 분류기를 만들기 위해 랜덤포레스트 알고리즘을 사용하였으며, 사용 툴은 파이썬(python3)의 사이킷-런(scikit-learn) 기계학습 라이브러리를 활용하였다.

3.4.2 퍼지 해싱 프로그램(Ssdeep)

Ssdeep[16]은 fuzzy hashing을 구현하는 프로그램으로 CTPH(Context-Trigger Piecewise Hash) 알고리즘을 바탕으로 한다. CTPH 알고리즘은 바이너리 데이터를 블록화 하여 데이터의 일부가 변경되어도 부분적으로 해쉬(hash) 값이 달라져 데이터 간의 유사도를 신속하게 판별할 수 있다.

이더리움 블록체인 시스템에서 CA는 bytecode 형식으로만 존재하고 있어 미확인 CA와 분류된 CA data set의 유사성을 신속하게 비교하기 위한 방법으로 바이트코드 fuzzy hashing이 사용 가능하다.

	Bytecode	Ssdeep
0	6060604052346200000057 5b5b6040805180820190915 2...	192:53yktZviFxDqDmRDXU0ISGaO0IMvaL6N9gH9iFmiU1...
1	6060604052346200000057 5b5b6040805180820190915 2...	192:53yktZdFxDqDmGDwo5cC0ISS1O0IIranAs9G/9rLmi...
2	6060604052346200000057 5b5b6040805180820190915 2...	192:53yktZVFxDqDmGDwo5c40ISS1O0IIranoXF9WH9qim...
3	6060604052346200000057 5b5b6040805180820190915 2...	192:53yktZOt9T1VvnZV0isolc+amSxr+aXbQ3S98qWj7mi...
4	6060604052346100005 75b5b6040805180820190915 260...	192:0kA8bktBZb+7rIIIdC8wC8XS7Qq9L0+4ImiF6xHICE...
5	606060405261021 02c1806100126000396000f3606060405 2...	24:1JptNDMuWQLUTK3CvjvZVwt/wNBzVx35F7ONIT9fy7o...
6	6060604052600180546c0100000000000000000000000000...	384:s+MOr9HmnrrIX1ItXSjjejHtL4nUcrgLhcqTI7N40...
7	6060604052610427806100126000396000f3606060405 2...	48:A8HSJuVzWsu+NdFzVEQPVTFdIT9K7HcXV9UhdOysAc...
8	6060604052600080546c0100000000000000000000000000...	48:9UCcT/uIT9K7fKn4OZOCIT9K7HFFCmcOQZrhkLTO0c...
9	6060604052600080546c0100000000000000000000000000...	96:O+6+K7HRbZTmWylgCezCJ9rD+Yoap7y3nsZ09BXwHfN...

Fig. 4. Bytecode Ssdeep Result

본 연구에서는 미확인 CA 분류를 위해 Fig. 4.와 같이 CA의 bytecode 전체를 Ssdeep으로 fuzzy hashing하였으며, 사용 툴은 파이썬의 python-ssdeep 라이브러리를 활용하였다.

IV. 결과 분석

4.1 컨트랙트 계정 분석

Reliable Contract Account Data Set와 두 Feature sets를 이용한 랜덤포레스트 알고리즘 CA 분류기는 성능평가 결과 Table 2.와 같이 정확도(accuracy), 정밀도(precision) 및 F1-score에서 결과 값이 모두 0.90을 초과하는 좋은 성능을 보여주었다. Annotation LDA Modeling Features를 이용한 카테고리 분류기는 리콜(recall) 점수가 상대적으로 낮은 부분이 있지만 높은 정확도와 정밀도로 CA의 카테고리를 분류 할 수 있었으며, ABI Function & I/O name Features를 이용한 디

Table 2. Classifier Accuracy Score

Classification		Acc.	Pre.	Rec.	F1-s.
Category		0.9223	0.9694	0.8780	0.9152
Design Patterns	DPT	0.9874	0.9924	0.9655	0.9783
	DPA	0.9371	0.9314	0.9397	0.9349
	Oracle	1.0000	1.0000	1.0000	1.0000
	Random	1.0000	1.0000	1.0000	1.0000
	Vote	0.9937	0.9968	0.9967	0.9599
	Time	0.9434	0.9968	0.9967	0.9959
	Kill	0.9497	0.9156	0.9156	0.9156
	Fork	1.0000	1.0000	1.0000	1.0000
	Math	1.0000	1.0000	1.0000	1.0000

자인 패턴 분류기는 모두 높은 정확도로 CA의 디자인 패턴을 분류 할 수 있었다. 특히, 분류속도에서 0.025ms/CA로 매우 빠른 속도를 보여 대량의 CA에 대하여 신속한 자동분류가 가능하였다.

4.1.1 카테고리 분류

확인된 CA 32,567개에 대하여 주석이 없는 일부 CA를 제외한 27,321개의 CA를 카테고리 분류하여 Table 3.과 같은 결과를 얻었다. 암호화폐인 이더리움의 특성상 금융(financial) CA가 22,901개로 대부분을 차지하였으며, 게임(game) CA가 1,279개로 두 번째로 많았으나 기타 카테고리 CA는 극소수만 있는 것으로 확인되었다.

Table 3. Category Classification Result

Category	Contracts	Ratio
Financial	22,901	0.8382
Game	1,279	0.0468
Notary	310	0.0114
Library	248	0.0091
Wallet	148	0.0054
Unknown	2,435	0.0891

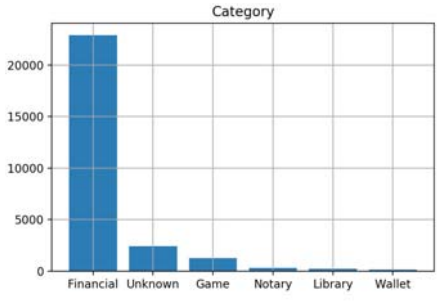


Fig. 5. Category Classification Graph

카테고리 분류결과에 대해 Fig. 6.의 시계열 분석을 보면 이더리움 발행 초기에는 금융 CA가 특별하게 많은 비중을 차지하지 않고 일정한 분포를 보이지만 2017년 6월 이후 급등하는 현상을 확인 할 수 있다. 이는 2017년 가상화폐 광풍에 따라 가격이 급등하여 많은 사용자들이 유입되고 금융 CA를 집중적으로 생성한 결과인 것을 확인 할 수 있으며, 특히 가격 하락시 CA 생성 증가가 보합을 보임을 알 수 있다.

카테고리별 이더리움 잔액(balance)과 거래 횟수의 관계는 Fig. 7.과 같이 금융 CA가 최대 잔액과 거래 횟수를 보이고 있으나 평균 잔액에서는 이더리움 지갑(wallet)과 게임 CA가 높았다. 특히, 평균 거래 횟수의 경우 게임과 금융이 유사하여 게임 CA의 비중은 전체에서 매우 적지만 활발하게 이더리움이 유통되고 있다는 사실을 알 수 있다.

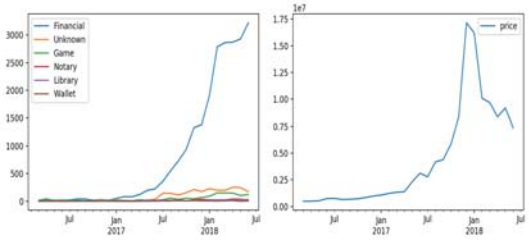


Fig. 6. Category-ETH Price Time Series Graph(corr. score : 0.7728)

category	Mean_balance	Max_balance	Mean_tx_cnt	Max_tx_cnt
Financial	1.163433	902.068460	2577.603249	9971581
Unknown	0.432233	212.416823	2103.822587	808184
Game	3.990021	850.269492	2491.302580	753289
Notary	0.416027	45.744688	1784.700000	130666
Library	0.161813	22.880100	580.028226	25707
Wallet	10.247289	715.320343	704.655405	44194

Fig. 7. Category-Balance & Transaction Count

4.1.2 디자인 패턴 분류

확인된 CA 32,567개에 대한 디자인 패턴 분류 결과 Fig. 8.과 같이 카테고리 분류 결과에서 대다수를 차지하고 있는 금융 CA와 관련이 깊은 토큰(DPT)과 사용자 인증(DPA)이 많은 부분을 차지하고 있다.

트레이닝 세트로 사용한 CA 디자인 패턴 구성에

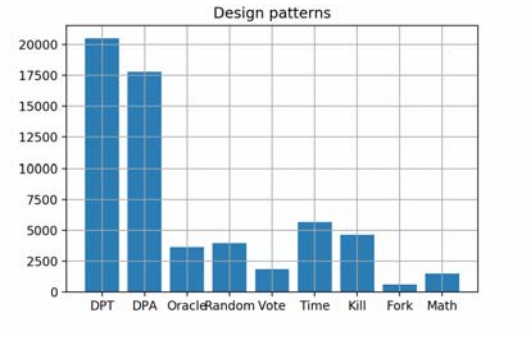


Fig. 8. Design Patterns Classification Graph

비해 토큰(DPT)의 비중이 훨씬 많아진 것을 확인할 수 있는데 이는 이오스, 스테이터스 네트워크 토큰 등 암호화폐 열풍과 함께 이더리움을 기반으로 하는 많은 토큰들이 생성된 결과로 판단된다.

Fig. 9.의 카테고리와 디자인 패턴의 관계를 보면 금융 CA의 경우 기능 발휘를 위해 다양한 함수가 필요하여 대부분의 패턴이 포함되어 있으며, 특히 수학적 연산(math)이 다른 카테고리에 비해 절대적으로 많았다. 게임 CA의 경우 타 카테고리에 비해 임의 값 생성(random)이 많아 다이스 또는 룰렛 등의 게임 형태가 다수 포함된 것임을 예상할 수 있다. 또한, 이더리움 지갑 CA은 이더리움의 안전한 보관을 위해 사용자 인증 패턴을 다수 포함하고 있다.

디자인 패턴에 대해 Fig. 10.의 시계열 분석을 보면 카테고리 분석과 유사한 양상을 보이는 것을 알 수 있다. 특히, 임의 값 생성 패턴을 가진 CA가 2018년 초반기에 급격히 많아지는 것을 알 수 있는데 이는 게임 카테고리 CA가 급격히 늘어난 것과 연계되어 있으며, 가격 상승 속도보다 상대적으로 늦은 임의 값 생성 패턴 CA 증가는 2017년 말 이더리움에 사용자들이 다수 유입되고 활발히 거래가 되면서 자연스럽게 스마트 컨트랙트를 이용한 이더리움 게임에 관심이 이어진 것으로 판단 할 수 있다.

Category	DPT	DPA	Oracle	Random	Vote	Time	Kill	Fork	Math
Financial	14692	13487	2906	3301	1198	3552	3462	386	1411
Unknown	2009	1068	188	84	55	605	247	66	5
Game	376	815	207	278	126	193	346	29	9
Notary	210	202	42	18	25	82	77	10	2
Library	167	136	25	26	8	101	32	6	5
Wallet	19	134	3	10	2	52	54	9	4

Fig. 9. Design Patterns Classification Result

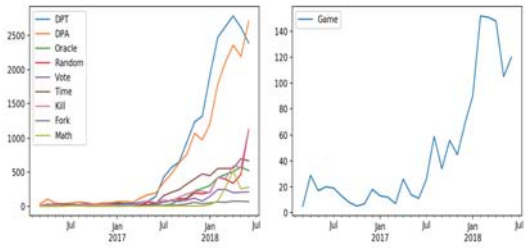


Fig. 10. Design Patterns - Game CA Time Series Graph (corr. score : 0.7732)

4.1.3 기존 연구와 결과 비교

이더리움 스마트 컨트랙트 관련 R. Norvill 등의 기존 연구에서는 스마트 컨트랙트를 분석하기 위해 936개의 CA를 사용하였고 트레이닝을 위한 데이터 세트가 없는 비지도 학습 방식으로 CA의 이름을 토큰화한 1,803개의 네임워드만을 feature로 사용하였다. 그 결과 936개의 CA를 기능이 명확하지 않은 5개의 클러스터 수준으로 분류하여 미확인 스마트 컨트랙트의 기능 확인을 위한 data set로는 부족한 부분이 있었다.

본 연구에서는 신뢰할 수 있는 data set를 사용하고 다양한 feature를 적용한 지도 학습 방식으로 지금까지 공개된 모든 CA 분류를 높은 정확성으로 자동화하여 수행 할 수 있었으며, 그 결과 확인된 CA의 분류 결과로 이더리움 스마트 컨트랙트의 최근 동향을 확인할 수 있었으며, 연구의 목적인 미확인 스마트 컨트랙트의 분류를 위한 충분한 크기의 기능이 확인된 CA data set를 확보할 수 있었다.

4.2 미확인 컨트랙트 계정 분류

Reliable CA data set의 811개 CA만을 사용하여 블라인드 처리 후 Ssdeep으로 분류 한 결과 Table 4.와 같이 0.3932의 낮은 확률로 미확인 CA를 분류 할 수 있었으나 분류된 미확인 CA의 실제 카테고리 및 디자인 패턴 정확도는 0.9486로 높았다.

랜덤포레스트 알고리즘 분류기를 이용하여 확인된 CA로 부터 스케일이 큰 CA data set 생성하였고 이를 바탕으로 전체 27,321개 CA를 블라인드 처리하여 분류를 시도한 결과 0.6374의 확률로 CA를 분류할 수 있었으며, 정확도 역시 0.9484로 높아

Table 4. Bytecode Fuzzy Hashing Compare Result

Contracts	Compare Rate	Accuracy Score	Mean Comp. Score
Reliable CA (811)	0.3932	0.9486	74.28
Verified CA (27,321)	0.6373	0.9484	88.72
Recent CA (1,000)	0.7630	0.9790	93.12

data set의 크기가 미확인 CA 분류 성능 향상에 중요한 영향을 미침을 알 수 있었다.

특히, 최근 1,000개의 CA를 대상으로 블라인드 처리하여 실험 하였을 때는 0.7630로 분류 확률이 더 높아졌으며, 정확도 또한 0.9790으로 향상되었다. 이는 이더리움 초기에 비해 최근에는 CA의 소스코드가 주요 커뮤니티를 통해 공유되면서 일정한 패턴으로 최적화 되었고 이를 활용한 CA가 증가함에 따라 분류 성능이 향상된 것으로 판단된다.

결과적으로 확인된 CA를 자동화하여 분류한 data set을 바탕으로 Ssdeep을 통해 미확인 CA의 bytecode를 hashing하여 비교 분석하면 data set의 크기에 비례하여 높은 확률로 미확인 CA의 카테고리와 디자인 패턴을 분류 할 수 있다.

4.3 제한사항

분류기를 위한 Reliable Contract Account Data Set은 811개의 CA로 충분한 수량은 아니었으나 Classifier Accuracy Score에서 좋은 성능이 나와 신뢰도 있게 분류가 가능하였다. 하지만, data set의 최초 구성이 금융 CA 373개, 미확인(Unknown) CA 155개로 과반 이상을 차지하여 소수 카테고리의 분류 결과에 영향을 미쳤을 것으로 판단된다. data set에서 크기가 작은 카테고리의 CA를 추가로 확보하고 미확인 CA에 대하여 새로운 기준을 적용한다면 더욱 정확한 분류 결과를 얻을 수 있을 것이다.

미확인 CA 분류 결과 낮은 분류 확률(compare rate)에 비해 상대적으로 매우 높은 정확도를 얻을 수 있었다. 분류 확률을 높이고 정확도를 합리적으로 조정하려 했지만 Ssdeep fuzzy hashing 툴에서 유사도 측정 기준의 Threshold를 변경하지 못하여 분류 확률을 인위적으로 올릴 수 없었다. 차후 연구에서는 이더리움 CA 바이트코드의 구조를 분석하여 fuzzy hashing을 핵심 코드 위주로 적용 하거나 유사도 측정 기준 Threshold를 합리적으로 조정한다면 분류 확률을 높일 수 있을 것이다.

V. 결 론

본 연구에서는 이더리움 블록체인 시스템의 실명 은닉성을 공격하는 방안의 일환으로 기계학습을 활용한 미확인 스마트 컨트랙트의 자동 분류 방안을 제안하였다.

실험 결과 etherscan.io를 통해 수집한 27,321개의 CA를 6개의 카테고리와 9개의 디자인 패턴으로 분류 할 수 있었으며, 이를 통해 이더리움 스마트 컨트랙트의 전반적인 동향을 분석 할 수 있었다.

그리고 분류를 통해 만든 CA data set과 Ssdeep을 이용하여 미확인 CA를 성공적으로 카테고리와 디자인 패턴으로 분류 할 수 있었다. 이러한 미확인 CA의 분류를 통해 CA 기능을 판단 할 수 있으며, 이는 특정사용자의 이더리움 주소로부터 거래한 CA를 분석하면 사용자가 이더리움 시스템을 통해 어떤 행위를 한 것인지, 즉 사용자 행위를 추정할 수 있어 포렌식의 관점에서도 활용성이 있음을 알 수 있다.

앞으로의 연구에서는 이더리움 블록체인 시스템에 등록된 모든 CA를 분류하고 분석하여 보다 정확한 이더리움 스마트 컨트랙트의 동향을 판단해야 할 것이며, 이를 통해 만들어진 큰 스케일의 CA data set으로 미확인 CA의 분류 확률을 향상시켜야 하겠다. 또한, 범죄 및 사고 등 실제 사례를 확보하여 이더리움 사용자의 행위가 미확인 CA 분류 결과와 연관되어 있음을 실질적으로 확인해야 할 것이다.

References

[1] S. Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System,” Oct. 2008.

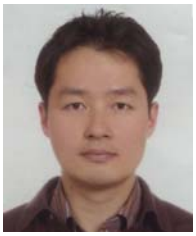
[2] Wood, Gavin, “Ethereum: A secure decentralised generalised transaction ledger,” Ethereum project yellow paper 151, pp. 1-32, Aug. 2014.

[3] Androulaki, Elli, et al, “Evaluating user privacy in bitcoin,” International Conference on Financial Cryptography and Data Security, pp. 34-51, Apr. 2013.

[4] Meiklejohn, Sarah, et al, "A fistful of bitcoins: characterizing payments among men with no names," Proceedings of the 2013 conference on Internet measurement conference, ACM, pp. 127-140, Oct. 2013.
[5] Fleder, Michael, Michael S. Kester, and Sudeep Pillai, "Bitcoin transaction graph analysis," arXiv preprint arXiv:1502.01657, Jan. 2015.
[6] Goldfeder, Steven, et al, "When the cookie meets the blockchain: Privacy risks of web payments via cryptocurrencies," arXiv preprint arXiv: 1708.04748, Aug. 2017.
[7] Koshy, Philip, Diana Koshy, and Patrick McDaniel, "An analysis of anonymity in bitcoin using p2p network traffic," International Conference on Financial Cryptography and Data Security, pp. 469-485, Mar. 2014.
[8] Maxwell, Greg, "CoinJoin: Bitcoin privacy for the real world," Post on Bitcoin forum, Aug. 2013.
[9] Bonneau, Joseph, et al, "Mixcoin: Anonymity for Bitcoin with accountable mixes," International Conference on Financial Cryptography and Data Security, pp. 486-504, Mar. 2014.
[10] Payette, James, Samuel Schwager, and Joseph Murphy, "Characterizing the ethereum address space," Dec. 2017.
[11] Norvill, Robert, et al, "Automated labeling of unknown contracts in Ethereum," The 26th International Conference on Computer Communications and Networks, Jul. 2017.
[12] Etherscan, "Etherscan," https://etherscan.io/, Sep. 2018.
[13] Bartoletti, Massimo, and Livio Pompianu, "An empirical analysis of smart contracts: platforms, applications, and design patterns," International Conference on Financial Cryptography and Data Security, pp. 494-509, Apr. 2017.
[14] Blei, David M., Andrew Y. Ng, and Michael I. Jordan, "Latent dirichlet allocation," Journal of machine Learning research, pp. 993-1022, Jan. 2003.
[15] Liaw, Andy, and Matthew Wiener, "Classification and regression by randomForest," R news, vol2, no. 3, pp. 18-22, Nov. 2001.
[16] Namanya, Anitta Patience, et al, "Detection of malicious portable executables using evidence combinational theory with fuzzy hashing," 2016 IEEE 4th International Conference on Future Internet of Things and Cloud, pp. 91-98, Aug. 2016.

〈저자소개〉

이 동 건 (Dong-gun Lee) 학생회원
2007년 2월: 육군사관학교 전산학과 학사
2017년 3월~현재: 연세대학교 정보대학원 석사과정
〈관심분야〉 정보보호, 디지털 포렌식 등



권 태 경 (Taekyoung Kwon) 종신회원
1992년 2월: 연세대학교 컴퓨터과학과 학사
1995년 2월: 연세대학교 컴퓨터과학과 석사
1999년 8월: 연세대학교 컴퓨터과학과 박사
1999년~2000년: U.C. Berkely Post-Doc.
2001년~2013년 8월: 세종대학교 컴퓨터공학과 교수
2007년~2008년: Univ. Maryland at College Park 교환교수
2013년 9월~현재: 연세대학교 정보대학원 교수
〈관심분야〉 암호프로토콜, Usable Security, 소프트웨어/시스템보안, 기계학습과보안 등