Full text available at: http://dx.doi.org/10.1561/3300000044

Security Analysis and Formal Verification on Blockchain and its Applications

Full text available at: http://dx.doi.org/10.1561/3300000044

Other titles in Foundations and Trends® in Privacy and Security

Recommender Systems Meet Large Language Model Agents: A Survey
Xi Zhu, Yu Wang, Hang Gao, Wujiang Xu, Chen Wang, Zhiwei Liu, Kun Wang, Mingyu Jin, Linsey Pang, Qingsong Weng, Philip S. Yu and Yongfeng Zhang
ISBN: 978-1-63828-564-9

Trustworthy Machine Learning: From Data to Models
Bo Han, Jiangchao Yao, Tongliang Liu, Bo Li, Sanmi Koyejo and Feng Liu
ISBN: 978-1-63828-548-9

Advances in Secure IoT Data Sharing
Phu Nguyen, Arda Goknil, Gencer Erdogan, Shukun Tokas, Nicolas Ferry and Thanh Thao Thi Tran
ISBN: 978-1-63828-422-2

Navigating the Soundscape of Deception: A Comprehensive Survey on Audio Deepfake Generation, Detection, and Future Horizons
Taiba Majid Wani, Syed Asif Ahmad Qadri, Farooq Ahmad Wani and Irene Amerini
ISBN: 978-1-63828-492-5

Reverse Engineering of Deceptions on Machine- and Human-Centric Attacks
Yuguang Yao, Xiao Guo, Vishal Asnani, Yifan Gong, Jiancheng Liu, Xue Lin, Xiaoming Liu and Sijia Liu
ISBN: 978-1-63828-340-9

Identifying and Mitigating the Security Risks of Generative AI
Clark Barrett *et al.*
ISBN: 978-1-63828-312-6

Full text available at: http://dx.doi.org/10.1561/3300000044

Security Analysis and Formal Verification on Blockchain and its Applications

Kang Li
CertiK

Ronghui Gu
Columbia University

Jun Xu
University of Utah

Zhaofeng Chen
CertiK

Siwei Wu
Zhejiang University

Yajin Zhou
Zhejiang University

Mu Zhang
University of Utah

Xiapu Luo
Hong Kong PolyU

Yuzhe Tang
Syracuse University

Yi Li
NTU

Xiaokuan Zhang
George Mason University

Yibo Wang
Syracuse University

now
the essence of knowledge
Boston — Delft

Full text available at: http://dx.doi.org/10.1561/3300000044

Foundations and Trends® in Privacy and Security

Published, sold and distributed by:
now Publishers Inc.
PO Box 1024
Hanover, MA 02339
United States
Tel. +1-781-985-4510
www.nowpublishers.com
sales@nowpublishers.com

Outside North America:
now Publishers Inc.
PO Box 179
2600 AD Delft
The Netherlands
Tel. +31-6-51115274

The preferred citation for this publication is

K. Li *et al.*. *Security Analysis and Formal Verification on Blockchain and its Applications*. Foundations and Trends® in Privacy and Security, vol. 8, no. 1, pp. 1–121, 2025.

ISBN: 978-1-63828-569-4
© 2025 K. Li *et al.*

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, mechanical, photocopying, recording or otherwise, without prior written permission of the publishers.

Photocopying. In the USA: This journal is registered at the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923. Authorization to photocopy items for internal or personal use, or the internal or personal use of specific clients, is granted by now Publishers Inc for users registered with the Copyright Clearance Center (CCC). The 'services' for users can be found on the internet at: www.copyright.com

For those organizations that have been granted a photocopy license, a separate system of payment has been arranged. Authorization does not extend to other kinds of copying, such as that for general distribution, for advertising or promotional purposes, for creating new collective works, or for resale. In the rest of the world: Permission to photocopy must be obtained from the copyright owner. Please apply to now Publishers Inc., PO Box 1024, Hanover, MA 02339, USA; Tel. +1 781 871 0245; www.nowpublishers.com; sales@nowpublishers.com

now Publishers Inc. has an exclusive license to publish this material worldwide. Permission to use this content must be obtained from the copyright license holder. Please apply to now Publishers, PO Box 179, 2600 AD Delft, The Netherlands, www.nowpublishers.com; e-mail: sales@nowpublishers.com

Full text available at: http://dx.doi.org/10.1561/3300000044

Foundations and Trends® in Privacy and Security
Volume 8, Issue 1, 2025

Editorial Board

Editor-in-Chief

Jonathan Katz
University of Maryland, USA

Founding Editors

Anupam Datta
Carnegie Mellon University, USA

Jeannette Wing
Columbia University, USA

Editors

Martín Abadi
Google and University of California, Santa Cruz

Michael Backes
Saarland University

Dan Boneh
Stanford University

Véronique Cortier
LORIA, CNRS

Lorrie Cranor
Carnegie Mellon University

Cédric Fournet
Microsoft Research

Virgil Gligor
Carnegie Mellon University

Jean-Pierre Hubaux
EPFL

Deirdre Mulligan
University of California, Berkeley

Andrew Myers
Cornell University

Helen Nissenbaum
New York University

Michael Reiter
Duke University

Shankar Sastry
University of California, Berkeley

Dawn Song
University of California, Berkeley

Daniel Weitzner
Massachusetts Institute of Technology

Full text available at: http://dx.doi.org/10.1561/3300000044

Editorial Scope

Foundations and Trends® in Privacy and Security publishes survey and tutorial articles in the following topics:

- Access control
- Accountability
- Anonymity
- Application security
- Artifical intelligence methods in security and privacy
- Authentication
- Big data analytics and privacy
- Cloud security
- Cyber-physical systems security and privacy
- Distributed systems security and privacy
- Embedded systems security and privacy
- Forensics
- Hardware security
- Human factors in security and privacy
- Information flow
- Intrusion detection
- Malware
- Metrics
- Mobile security and privacy
- Language-based security and privacy
- Network security
- Privacy-preserving systems
- Protocol security
- Security and privacy policies
- Security architectures
- System security
- Web security and privacy

Information for Librarians

Foundations and Trends® in Privacy and Security, 2025, Volume 8, 4 issues. ISSN paper version 2474-1558. ISSN online version 2474-1566. Also available as a combined paper and online subscription.

Full text available at: http://dx.doi.org/10.1561/3300000044

Contents

Full text available at: http://dx.doi.org/10.1561/3300000044

Full text available at: http://dx.doi.org/10.1561/3300000044

Security Analysis and Formal Verification on Blockchain and its Applications

Kang Li[1], Ronghui Gu[2], Jun Xu[3], Zhaofeng Chen[1], Siwei Wu[4], Yajin Zhou[4,5], Mu Zhang[3], Xiapu Luo[6], Yuzhe Tang[7], Yi Li[8], Xiaokuan Zhang[9] and Yibo Wang[7]

[1] *CertiK, USA; kang.li@certik.com, zhaofeng.chen@certik.com*
[2] *Columbia University, USA; rg3123@columbia.edu*
[3] *The University of Utah, USA; junxzm@cs.utah.edu, muzhang@cs.utah.edu*
[4] *Zhejiang University, China; wusw1020@zju.edu.cn*
[5] *BlockSec, Hong Kong; yajin_zhou@zju.edu.cn*
[6] *The Hong Kong Polytechnic University, Hong Kong; csxluo@comp.polyu.edu.hk*
[7] *Syracuse University, USA; ytang100@syr.edu, ywang349@syr.edu*
[8] *Nanyang Technological University, Singapore; yi_li@ntu.edu.sg*
[9] *George Mason University, USA; xiaokuan@gmu.edu*

ABSTRACT

Blockchains have become an integrated part of our finance infrastructures. Being monetary yet fully automated, blockchains and their applications are unanimously deemed impracticable before undergoing necessary verification. This monograph reviews the previous attempts at verifying two fundamental properties of blockchains: correctness (where flaws lead to unintentional damages) and security (where vulnerabilities incur attacks and losses). First, it summarizes

Kang Li, Ronghui Gu, Jun Xu, Zhaofeng Chen, Siwei Wu, Yajin Zhou, Mu Zhang, Xiapu Luo, Yuzhe Tang, Yi Li, Xiaokuan Zhang and Yibo Wang (2025), "Security Analysis and Formal Verification on Blockchain and its Applications", Foundations and Trends® in Privacy and Security: Vol. 8, No. 1, pp 1–121. DOI: 10.1561/3300000044.
©2025 K. Li *et al.*

Full text available at: http://dx.doi.org/10.1561/3300000044

and categorizes the correctness and security flaws encountered by real-world blockchains. Second, it systematizes the development of formal verification to address the flaws in blockchains, covering the aspects of models, specifications, and techniques. Third, it unveils the progress of security analysis for mitigating the flaws, unveiling the analysis principles being followed, the flaw oracles being devised, and the detection methods being used. Finally, it summarizes the challenges remaining to be addressed, followed by our vision of the trend in the near future. Throughout this monograph, we anticipate shedding light on future blockchain verification advances, especially in expanding its applicability, making specification generation easier, and discovering previously unknown vulnerabilities. By identifying gaps such as missing tools for infrastructure-level components and the difficulty of writing formal specifications, this work aims to motivate the development of more automated, intelligent, and practical verification frameworks.

Full text available at: http://dx.doi.org/10.1561/3300000044

1

Introduction

1.1 Blockchain and its Applications

1.1.1 A Brief History

In 2008, Satoshi Nakamoto's white paper, "*Bitcoin: A Peer-to-Peer Electronic Cash System*" (Nakamoto, 2008), was released to the public, proposing a solution enabled by peer-to-peer network for electronic cash payments without needing a trusted third party. This was the first time the concept of blockchain and the technology underlying Bitcoin (the first decentralized cryptocurrency) came to our attention.

Since the advent of Bitcoin, blockchain technology has evolved significantly. Ethereum, introduced by Vitalik Buterin in 2013 (Buterin *et al.*, 2013), expanded the capabilities of blockchain by enabling smart contracts, which are self-executing programs with agreement terms written into code. This development opened up unprecedented possibilities for decentralized applications across various sectors.

1.1.2 Blockchain Technology

Overview: At its core, blockchain is a distributed database that maintains a continuously growing list of records, called *blocks*, which are

Full text available at: http://dx.doi.org/10.1561/3300000044

securely linked using cryptographic techniques. Each block contains a cryptographic hash of the previous block, a timestamp, and transaction data, making it virtually tamper-proof and resistant to modification. This structure ensures that once a block is added to the chain, the information it contains is immutable and can be trusted by all participants in the network.

Most blockchains today are supported by networks following a peer-to-peer (P2P) model (Buford *et al.*, 2009), where each node acts as a peer and can perform operations independently of central servers or authorities. Data on blockchains, especially the transactions, are distributed across nodes, ensuring each node can access the entire of it.

Block Operations: The creation of blocks depends on the blockchain's *consensus mechanism* (Lashkari and Musilek, 2021). Blockchain consensus mechanisms are foundational protocols that allow network participants to agree on the current state of a distributed ledger, ensuring all transactions are accurate and preventing potential fraud. For illustrations, we introduce the two most popular consensus mechanisms and how blocks are created under them.

- **Proof of Work (PoW, Nakamoto, 2008):** In PoW blockchains like Bitcoin, a subset of nodes called *miners* compete to solve complex cryptographic puzzles (known as *mining*), and the first miner to solve the puzzle gets the right to add a new block consisting of transactions to the blockchain. *Full nodes*, a superset of miners, validate the new block and its transactions. If the block violates the rules of the blockchain network, it is rejected to prevent invalid transactions or fraudulent blocks from being added to the blockchain.

- **Proof of Stake (PoS, Smith, 2024):** In PoS blockchains like Tendermint-based Ethereum (Buchman, 2016), creating and validating blocks are handled by nodes called *validators*, which are required to stake a certain amount of their assets as collateral. Validators are selected to create a new block based on various factors, such as the size of their stake, random selection processes, and the length of time they have held the stake. Once chosen,

Full text available at: http://dx.doi.org/10.1561/3300000044

the validator can create a block with transactions and broadcast it to other nodes. If accepted by other validators, the block is added to the blockchain. To discourage validators from acting maliciously or negligently, PoS blockchains can penalize them by slashing a portion of their staked tokens if they attempt to approve fraudulent transactions or fail to remain online and functional.

User Interactions: Users in the wild can interact with blockchains through *transactions* (precisely, units of exchange on blockchain networks to enable the transfer of value and information between participants). To issue transactions, users usually need a *wallet*—a digital identity that allows users to manage their cryptocurrency or digital assets. With a wallet ready, a user can create a transaction by specifying the necessary information (e.g., transaction type, recipient's address, transfer amount, etc.) and sign it with the private key associated with their wallet. A signed transaction is first submitted to a blockchain node and propagated across the peer-to-peer network. Miners or validators then select pending transactions from the network for execution and inclusion in a new block. Once a block is created, it is broadcast to the network for validation. If a majority of nodes accept the block as valid according to the consensus protocol, the block is appended to the blockchain. At this point, the transaction is considered finalized and immutable. An overview of the transaction life cycle is presented in Figure 1.1.

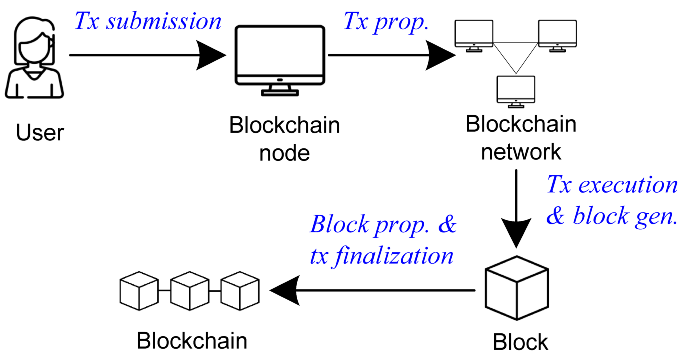

Figure 1.1: Transaction life cycle.

Full text available at: http://dx.doi.org/10.1561/3300000044

To incentivize the miners or validators to include a transaction, the users need to offer a *transaction fee*. Usually, higher fees can lead to faster processing, especially on congested networks. With different blockchains, transaction fees are collected using different mechanisms. On Bitcoin, transaction fees are calculated based on the transaction size in bytes, and users pay the fees implicitly with Bitcoins from their inputs to the transaction. On Ethereum, transaction fees are measured by the *gas* needed for completing the transaction. Specifically, each operation, from sending the transaction to executing the transaction, requires a predetermined amount of gas. The total gas, after completion of the transaction, will be paid from the user's wallet with Ether.

1.1.3 Blockchain Applications

Blockchain technology has enabled a wide range of applications across various industries. However, the diversity of applications escalated dramatically after smart contracts were invented.

Before Smart Contracts: Before the advent of smart contracts, blockchain technology was primarily known for its application as a decentralized ledger for cryptocurrencies, with Bitcoin being the pioneer. The most fundamental use of cryptocurrencies is to enable transfers between two user wallets without needing a trusted intermediary like a bank. A side use is to earn rewards for participating in the blockchain operations. For instance, Bitcoin miners receive rewards for each block mined, including a combination of newly minted bitcoins and transaction fees from all transactions included in the block.

Besides peer-to-peer transfers, early blockchains have enabled a few applications in other domains, including but not limited to:

— *proof of ownership of digital art and virtual properties;*

— *timestamping of documents to prove existence at a particular time;*

— *tracking the origin and journey of products in industries;*

— *voting systems with reduced fraud and enhanced transparency.*

Clarification: Some early applications, such as proof of ownership and product tracking, resemble later use cases like NFTs and supply chain

Full text available at: http://dx.doi.org/10.1561/3300000044

automation. However, smart contracts significantly enhanced these domains by enabling programmability and automation. For example, NFTs formalized ownership through standardized token interfaces, and supply chain systems now benefit from automatic updates and conditional payments via smart contracts. Thus, while the core ideas existed before, smart contracts expanded their scope, functionality, and adoption.

After Smart Contracts: Smart contracts, introduced by Ethereum, are self-executing programs with the terms of the agreement between participants being directly written into lines of code. These contracts automatically enforce and execute themselves when predefined conditions are met, providing a secure, transparent, and efficient way to facilitate and verify transactions without intermediaries.

Smart contracts are typically written in high-level programming languages crafted for blockchains, such as Solidity for Ethereum. Once finalized, a smart contract is usually compiled into bytecode, a low-level representation runnable in the blockchains' *virtual machine* (VM). A special transaction can be created to deploy the bytecode to a block. To execute a function defined in the smart contract, users send transactions directly to the smart contract's address with details such as function identifier and any parameters required by the function.

The programmability and automation provided by smart contracts have spurred a wide range of applications across various domains. Some representative categories include:

- **Decentralized Finance (DeFi):** Smart contracts have been used to create protocols that replicate existing financial services, enabling the DeFi ecosystem. In this ecosystem, we have witnessed *lending and borrowing platforms* where users loan cryptocurrencies and pay interest automatically, *stablecoins* where smart contracts maintain a peg to other assets like USD, *yield farming and liquidity mining* where users stake liquidity and earn rewards in the form of transaction fees or governance tokens.

- **Decentralized Exchanges (DEXs):** Smart contracts have enabled Automated Market Makers (AMMs) like Uniswap and SushiSwap to create liquidity pools that automatically execute

Full text available at: http://dx.doi.org/10.1561/3300000044

trades based on algorithms. Smart contracts have also created space where users can exchange tokens directly from their wallets, bypassing the need for centralized exchanges.

- **Non-Fungible Tokens (NFTs):** Smart contracts have been adopted to verify the authenticity and ownership of digital assets, allowing artists and creators to sell unique digital art pieces directly to consumers. Similarly, in games, smart contracts can manage in-game assets that players can own, trade, or use across different gaming platforms.
- **Supply Chain Management:** Smart contracts can record the journey of a product through its supply chain, automatically updating at each stage when conditions are met to ensure transparency and authenticity. With smart contracts, payments can be automatically triggered when goods are delivered or milestones are met, reducing delays and removing the need for manual processing.

1.2 The Need for Verification

Compared to conventional computation platforms, blockchains offer the following set of unique properties, leading to their rapid development and tremendous deployment.

- **Decentralization:** A blockchain is maintained by a network of nodes, each holding a copy of the entire ledger. This decentralization enhances security and reduces the risk of data manipulation.
- **Transparency:** Transactions on a blockchain are visible to all participants in the network, providing a high level of transparency. This feature is particularly valuable in applications requiring accountability and auditability.
- **Immutability:** Once recorded on a blockchain, data cannot be altered or deleted. This immutability ensures the integrity and reliability of the data, making blockchain an ideal solution for record-keeping and verification purposes.
- **Security:** Blockchain employs advanced cryptographic techniques to secure transactions and control the creation of new units. The

Full text available at: http://dx.doi.org/10.1561/3300000044

consensus mechanisms used in blockchain networks further enhance security by making it computationally infeasible for malicious actors to alter the blockchain.

However, the properties shall never be taken for granted. The designs and implementations of blockchains and their applications can involve high complexities and subtleness. If not properly handled, they can introduce various flaws compromising those properties. As we will systematize in Section 2, we have witnessed numerous flaws in real-world blockchains and applications, spanning all aspects and components.

When the flaws are triggered, especially when exploited by adversaries, the four properties above can break, and assumptions about the safety of assets no longer hold. This often leads to financial damage at an astonishing level. In 2023 alone, the top ten attacks against blockchain flaws have led to asset losses totaling around $1,146 million. These notorious attacks have shaken society's confidence in blockchains. For instance, the DAO hack against Ethereum in 2016 (Chen, 2019) not only incurred a $60 million theft but, more importantly, led to a hard fork of the blockchain. This sparked a significant debate over the immutability of blockchains, threatening the foundations of blockchain technology. Thus, ***verification of the core properties of blockchains and their applications is a must to ensure the development and sustainability of the entire ecosystem.***

1.3 Outline

In this monograph, we aim to present a review of the existing efforts on verification for mitigating flaws in blockchains and their applications. We differentiate these efforts into two big categories of *formal verification* and *security analysis* and discuss them separately. Unlike previous attempts that organize the literature based on how the methods work (modeling, specification, techniques, etc.), we take a problem-driven strategy: we organize the existing methods based on the flaws they focus on. Specifically, we elaborate on each family of major flaws, and under each flaw, we discuss the applicable formal verification and security analysis methods. This way, we deliver a clear understanding of what

Full text available at: http://dx.doi.org/10.1561/3300000044

problems have been addressed and what have not.

The follow-up sections of this review are organized as follows. In Section 2, we categorize and summarize the common flaws we have observed in the real world. In Section 3 and Section 4, we systematize the formal verification and security analysis methods to address those flaws. In Section 5, we discuss the remaining challenge faced by formal verification and security analysis, followed by sharing our opinions about the future in Section 6.

Full text available at: http://dx.doi.org/10.1561/3300000044

References

Abate, A., C. David, P. Kesseli, D. Kroening, and E. Polgreen. (2018). "Counterexample guided inductive synthesis modulo theories". In: *International Conference on Computer Aided Verification*. Springer. 270–288.

Abdelaziz, T. and A. Hobor. (2023). "Smart learning to find dumb contracts". In: *32nd USENIX Security Symposium (USENIX Security 23)*. 1775–1792.

Alchemy. (2022). "How to Send Private Transactions on Ethereum". URL: https://www.alchemy.com/overviews/ethereum-private-transactions.

Aleth. (2021). "Ethereum C++ client, tools and libraries". URL: https://github.com/ethereum/aleth.

Almakhour, M., L. Sliman, A. E. Samhat, and A. Mellouk. (2020). "Verification of smart contracts: A survey". *Pervasive and Mobile Computing*. 67: 101227.

Alt, L., M. Blicha, A. E. J. Hyvärinen, and N. Sharygina. (2022). "SolCMC: Solidity Compiler's Model Checker". In: *Computer Aided Verification - 34th International Conference, CAV 2022, Haifa, Israel, August 7-10, 2022, Proceedings, Part I*. Ed. by S. Shoham and Y. Vizel. Vol. 13371. *Lecture Notes in Computer Science*. Springer. 325–338. DOI: 10.1007/978-3-031-13185-1_16.

Full text available at: http://dx.doi.org/10.1561/3300000044

Alturki, M. A., J. Chen, V. Luchangco, B. Moore, K. Palmskog, L. Peña, and G. Roşu. (2020). "Towards a verified model of the algorand consensus protocol in coq". In: *Formal Methods. FM 2019 International Workshops: Porto, Portugal, October 7–11, 2019, Revised Selected Papers, Part I 3*. Springer. 362–367.

Alur, R. (1999). "Timed automata". In: *Computer Aided Verification: 11th International Conference, CAV'99 Trento, Italy, July 6–10, 1999 Proceedings 11*. Springer. 8–22.

Analog. (2022). "A Timeline and History of Recent Cross-Chain Bridge Attacks". URL: https://medium.com/@analogtime/a-timeline-and-history-of-recent-cross-chain-bridge-attacks-analog-insights-507349381a4b.

Ashraf, I., X. Ma, B. Jiang, and W. K. Chan. (2020). "GasFuzzer: Fuzzing Ethereum Smart Contract Binaries to Expose Gas-Oriented Exception Security Vulnerabilities". *IEEE Access*. 8: 99552–99564. DOI: 10.1109/ACCESS.2020.2995183.

Atzei, N., M. Bartoletti, and T. Cimoli. (2017). "A survey of attacks on ethereum smart contracts (sok)". In: *Principles of Security and Trust: 6th International Conference, POST 2017, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2017, Uppsala, Sweden, April 22-29, 2017, Proceedings 6*. Springer. 164–186.

Baldoni, R., E. Coppa, D. C. D'elia, C. Demetrescu, and I. Finocchi. (2018). "A survey of symbolic execution techniques". *ACM Computing Surveys (CSUR)*. 51(3): 1–39.

Basu, A., B. Bensalem, M. Bozga, J. Combaz, M. Jaber, T.-H. Nguyen, and J. Sifakis. (2011). "Rigorous component-based system design using the BIP framework". *IEEE software*. 28(3): 41–48.

Belchior, R., P. Somogyvari, J. Pfannschmidt, A. Vasconcelos, and M. Correia. (2024). "Hephaestus: Modeling, Analysis, and Performance Evaluation of Cross-Chain Transactions". *IEEE Trans. Reliab.* 73(2): 1132–1146. DOI: 10.1109/TR.2023.3336246.

Bertrand, N., P. Ghorpade, S. Rubin, B. Scholz, and P. Subotic. (2024). "Reusable Formal Verification of DAG-based Consensus Protocols". *arXiv preprint arXiv:2407.02167*.

Full text available at: http://dx.doi.org/10.1561/3300000044

Bertrand, N., V. Gramoli, I. Konnov, M. Lazić, P. Tholoniat, and J. Widder. (2022). "Holistic verification of blockchain consensus". *arXiv preprint arXiv:2206.04489*.

Blanchard, J. and A. Popowycz. (2023). "Analysis & Remediation of the Precompile Attack on the Hedera Network". URL: https://hedera.com/blog/analysis-remediation-of-the-precompile-attack-on-the-hedera-network.

Block, T. (2023). "BlockSec prevents $5 million from being stolen on Paraspace". URL: https://www.theblock.co/post/220761/blocksec-prevents-5-million-from-being-stolen-on-paraspace.

BlockSec. (2021a). "A Short Analysis of the Wild Exploitation of CVE-2021–39137". URL: https://blocksec.com/blog/a-short-analysis-of-the-wild-exploitation-of-cve-2021-39137.

BlockSec. (2021b). "New Integer Overflow Bug Discovered in Solana rBPF". URL: https://blocksec.com/blog/new-integer-overflow-bug-discovered-in-solana-r-bpf.

BlockSec. (2023a). "#10: ThirdWeb Incident: Incompatibility Between Trusted Modules Exposes Vulnerability". URL: https://blocksec.com/blog/10-third-web-incident-incompatibility-between-trusted-modules-exposes-vulnerability.

BlockSec. (2023b). "Systematic Approach to Maintaining EVM Compatibility and Security". URL: https://blocksecteam.medium.com/systematic-approach-to-maintaining-evm-compatibility-and-security-32d0f42846f5.

BlockSec. (2024a). "Reflecting on Reflection Tokens: A Security Perspective". URL: https://blocksec.com/blog/reflecting-on-reflection-tokens-a-security-perspective.

BlockSec. (2024b). "Security Check: Do EVM-Compatible Chains Hold Up?" URL: https://blocksec.com/blog/security-check-do-evm-compatible-chains-hold-up.

Bodell III, W. E., S. Meisami, and Y. Duan. (2023). "Proxy hunting: understanding and characterizing proxy-based upgradeable smart contracts in blockchains". In: *32nd USENIX Security Symposium (USENIX Security 23)*. 1829–1846.

Full text available at: http://dx.doi.org/10.1561/3300000044

Boxler, D. and K. R. Walcott. (2018). "Static taint analysis tools to detect information flows". In: *Proceedings of the International Conference on Software Engineering Research and Practice (SERP)*. The Steering Committee of The World Congress in Computer Science, Computer. 46–52.

Bradley, A. R. (2011). "SAT-based model checking without unrolling". In: *International Workshop on Verification, Model Checking, and Abstract Interpretation*. Springer. 70–87.

Brent, L., N. Grech, S. Lagouvardos, B. Scholz, and Y. Smaragdakis. (2020). "Ethainter: a smart contract security analyzer for composite vulnerabilities". In: *Proceedings of the 41st ACM SIGPLAN Conference on Programming Language Design and Implementation*. 454–469.

Brent, L., A. Jurisevic, M. Kong, E. Liu, F. Gauthier, V. Gramoli, R. Holz, and B. Scholz. (2018). "Vandal: A scalable security analysis framework for smart contracts". *arXiv preprint arXiv:1809.03981*.

Büchi, J. R. (1966). "Symposium on decision problems: On a decision method in restricted second order arithmetic". In: *Studies in Logic and the Foundations of Mathematics*. Vol. 44. Elsevier. 1–11.

Buchman, E. (2016). "Tendermint: Byzantine fault tolerance in the age of blockchains". *PhD thesis*. University of Guelph.

Buford, J., H. Yu, and E. K. Lua. (2009). *P2P networking and applications*. Morgan Kaufmann.

Buterin, V. *et al.* (2013). "Ethereum white paper". *GitHub repository*. 1: 22–23.

Caversaccio, P. (2024). "pcaversaccio/reentrancy-attacks: A chronological and (hopefully) complete list of reentrancy attacks to date." URL: https://github.com/pcaversaccio/reentrancy-attacks.

Cecchetti, E., S. Yao, H. Ni, and A. C. Myers. (2021). "Compositional security for reentrant applications". In: *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE. 1249–1267.

CertiK. (2023). "The HamsterWheel: An In-Depth Exploration of a Novel Attack Vector on the Sui Blockchain". URL: https://www.certik.com/resources/blog/the-hamsterwheel-an-in-depth-exploration-of-a-novel-attack-vector-on-the-sui.

Full text available at: http://dx.doi.org/10.1561/3300000044

Chain-Split, B. C. (2018). "Bitcoin Cash Chain-Split Bug (2018)". URL: https://www.bitcoinabc.org/2018-05-07-incident-report/.

Chainlink. (2023). "What Are Cross-Chain Smart Contracts?" URL: https://chain.link/education-hub/cross-chain-smart-contracts.

ChainWall. (2023). "Reentrancy Attack in Smart Contracts". URL: https://medium.com/chainwall-io/reentrancy-attack-in-smart-contracts-4837ed0f9d73.

Chandra, A. K. and D. Harel. (1985). "Horn clause queries and generalizations". *The Journal of Logic Programming*. 2(1): 1–15.

Chen, J., Y. Wang, Y. Zhou, W. Ding, Y. Tang, X. Wang, and K. Li. (2023a). "Understanding the Security Risks of Decentralized Exchanges by Uncovering Unfair Trades in the Wild". In: *8th IEEE European Symposium on Security and Privacy, EuroS&P 2023, Delft, Netherlands, July 3-7, 2023*. IEEE. 332–351. DOI: 10.1109/EUROSP57164.2023.00028.

Chen, J. and S. Micali. (2016). "Algorand". *arXiv preprint arXiv:1607.01341*.

Chen, T., R. Cao, T. Li, X. Luo, G. Gu, Y. Zhang, Z. Liao, H. Zhu, G. Chen, Z. He, *et al.* (2020). "SODA: A Generic Online Detection Framework for Smart Contracts." In: *NDSS*.

Chen, Y., F. Ma, Y. Zhou, Y. Jiang, T. Chen, and J. Sun. (2023b). "Tyr: Finding consensus failure bugs in blockchain system with behaviour divergent model". In: *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2517–2532.

Chen, Z. (2019). "Smart Contract Reentrancy: TheDAO". URL: https://medium.com/@zhongqiangc/smart-contract-reentrancy-thedao-f2da1d25180c.

Choi, J., D. Kim, S. Kim, G. Grieco, A. Groce, and S. K. Cha. (2021). "Smartian: Enhancing smart contract fuzzing with static and dynamic data-flow analyses". In: *2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE. 227–239.

Clarke, E. M. (1997). "Model checking". In: *Foundations of Software Technology and Theoretical Computer Science: 17th Conference Kharagpur, India, December 18–20, 1997 Proceedings 17*. Springer. 54–56.

Full text available at: http://dx.doi.org/10.1561/3300000044

Colby, C. and P. Lee. (1996). "Trace-based program analysis". In: *Proceedings of the 23rd ACM SIGPLAN-SIGACT symposium on Principles of programming languages*. 195–207.

Consensus-Bug, G. (2016). "Ethereum Geth Consensus Bug (2016)". URL: https:///2016/11/25/security-alert-11242016-consensus-bug-geth-v1-4-19-v1-5-2/.

Cosmos. (2023). "ICS04: Something confusing about function timeoutPacket and timeoutOnClose". URL: https://github.com/cosmos/ibc/issues/965.

Cosmos. (2024). "Interchain Standards (ICS) for the Cosmos network & interchain ecosystem." URL: https://github.com/cosmos/ibc.

Cosmos Hub Forum. (2024). "EVM Precompiled contract bug allowing unlimited token mint". URL: https://forum.cosmos.network/t/critical-evm-precompiled-contract-bug-allowing-unlimited-token-mint/14060.

Cousineau, D., D. Doligez, L. Lamport, S. Merz, D. Ricketts, and H. Vanzetto. (2012). "TLA + Proofs". In: *FM 2012: Formal Methods*. Ed. by D. Giannakopoulou and D. Méry. Berlin, Heidelberg: Springer Berlin Heidelberg. 147–154.

Crain, T., C. Natoli, and V. Gramoli. (2021). "Red belly: A secure, fair and scalable open blockchain". In: *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE. 466–483.

Cui, S., G. Zhao, Y. Gao, T. Tavu, and J. Huang. (2022). "VRust: Automated vulnerability detection for solana smart contracts". In: *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. 639–652.

CVE. (2018). "NVD - CVE-2018-11329". URL: https://nvd.nist.gov/vuln/detail/CVE-2018-11329.

CWA-2023-004. (2023). "CWA-2023-004". URL: https://github.com/CosmWasm/advisories/blob/main/CWAs/CWA-2023-004.md.

Danila, R. (2020). "Responsible Vulnerability Disclosure". URL: https://medium.com/nexus-mutual/responsible-vulnerability-disclosure-ece3fe3bcefa.

Das, S. and D. L. Dill. (2001). "Successive approximation of abstract transition relations". In: *Proceedings 16th Annual IEEE Symposium on Logic in Computer Science*. IEEE. 51–58.

Full text available at: http://dx.doi.org/10.1561/3300000044

Desharnais, J. and P. Panangaden. (2003). "Continuous stochastic logic characterizes bisimulation of continuous-time Markov processes". *The Journal of Logic and Algebraic Programming*. 56(1-2): 99–115.

Ding, W., Y. Tang, and Y. Wang. (2025). " Asymmetric Mempool DoS Security: Formal Definitions and Provable Secure Designs ". In: *2025 IEEE Symposium on Security and Privacy (SP)*. Los Alamitos, CA, USA: IEEE Computer Society. 61–61. DOI: 10.1109/SP61157.2025.00061.

Doshi, V. (2023). "Types of attacks on smart contracts". URL: https://varunx.hashnode.dev/smart-contract-attack-vectors.

Durieux, T., J. F. Ferreira, R. Abreu, and P. Cruz. (2020). "Empirical review of automated analysis tools on 47,587 ethereum smart contracts". In: *Proceedings of the ACM/IEEE 42nd International conference on software engineering*. 530–541.

Ethereum. (2016). "Exception on overflow". URL: https://github.com/ethereum/solidity/issues/796%5C#issuecomment-253578925.

Ethereum. (2018). "Utilities for interacting with the Ethereum virtual machine". URL: https://github.com/ethereum/evmlab.

Ethereum. (2020). "Go Ethereum: Official Go implementation of the Ethereum protocol". URL: https://github.com/ethereum/go-ethereum.

Ethereum. (2024a). "Common tests for all Ethereum implementations". URL: https://github.com/ethereum/tests.

Ethereum. (2024b). "Contract ABI Specification — Solidity 0.8.27 documentation". URL: https://docs.soliditylang.org/en/latest/abi-spec.html.

Ethereum. (2024c). "ERC-20 Token Standard". URL: https://ethereum.org/en/developers/docs/standards/tokens/erc-20/.

Ethereum. (2024d). "Solidity 0.8.27 documentation". URL: https://docs.soliditylang.org/en/v0.8.27/.

Ethereum. (2024e). "Solidity, the Smart Contract Programming Language". URL: https://github.com/ethereum/solidity.

Ethereum. (2024f). "theRun Contract". URL: https://etherscan.io/address/0xcac337492149bdb66b088bf5914bedfbf78ccc18%5C#code.

Full text available at: http://dx.doi.org/10.1561/3300000044

Etherscan. (2024). "Ethereum (ETH) Blockchain Explorer". URL: https://etherscan.io/.

Py-EVM. (2024). "A Python implementation of the Ethereum Virtual Machine". URL: https://github.com/ethereum/py-evm.

EVM. (2024). "EVM Codes - Precompiled Contracts". URL: https://www.evm.codes/precompiled.

JS-EVM. (2024). "TypeScript implementation of the Ethereum EVM." URL: https://www.npmjs.com/package/@ethereumjs/evm.

Feist, J., G. Grieco, and A. Groce. (2019). "Slither: a static analysis framework for smart contracts". In: *2019 IEEE/ACM 2nd International Workshop on Emerging Trends in Software Engineering for Blockchain (WETSEB)*. IEEE. 8–15.

Felleisen, M., R. B. Findler, M. Flatt, S. Krishnamurthi, E. Barzilay, J. McCarthy, and S. Tobin-Hochstadt. (2015). "The racket manifesto". In: *1st Summit on Advances in Programming Languages (SNAPL 2015)*. Schloss-Dagstuhl-Leibniz Zentrum für Informatik.

Feng, Y., E. Torlak, and R. Bodík. (2020). "Summary-based symbolic evaluation for smart contracts". In: *Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering*. 1141–1152.

Filliâtre, J.-C. and A. Paskevich. (2013). "Why3—where programs meet provers". In: *Programming Languages and Systems: 22nd European Symposium on Programming, ESOP 2013, Held As Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2013, Rome, Italy, March 16-24, 2013. Proceedings 22*. Springer. 125–128.

Finance, B. (2021). "Economic Exploit Attack on Bunny Protocol". URL: https://pancakebunny.medium.com/hello-bunny-fam-a7bf0c7a07ba.

Fioraldi, A., A. Mantovani, D. Maier, and D. Balzarotti. (2023). "Dissecting american fuzzy lop: a fuzzbench evaluation". *ACM transactions on software engineering and methodology*. 32(2): 1–26.

Flashbots. (2024). "Flashbots Docs". URL: https://docs.flashbots.net/flashbots-auction/overview/.

Flaws, S. (2024). "Solidity compiler bugs". URL: https://github.com/ethereum/solidity/blob/develop/docs/bugs.json.

Full text available at: http://dx.doi.org/10.1561/3300000044

Fu, Y., M. Ren, F. Ma, H. Shi, X. Yang, Y. Jiang, H. Li, and X. Shi. (2019). "Evmfuzzer: detect evm vulnerabilities via fuzz testing". In: *Proceedings of the 2019 27th ACM joint meeting on european software engineering conference and symposium on the foundations of software engineering*. 1110–1114.

Gorrieri, R. and R. Gorrieri. (2017). "Labeled transition systems". *Process Algebras for Petri Nets: The Alphabetization of Distributed Systems*: 15–34.

Graves, A. and A. Graves. (2012). "Long short-term memory". *Supervised sequence labelling with recurrent neural networks*: 37–45.

Grech, N., M. Kong, A. Jurisevic, L. Brent, B. Scholz, and Y. Smaragdakis. (2018). "MadMax: surviving out-of-gas conditions in Ethereum smart contracts". *Proc. ACM Program. Lang.* 2(OOPSLA): 116:1–116:27. DOI: 10.1145/3276486.

Grieco, G., W. Song, A. Cygan, J. Feist, and A. Groce. (2020). "Echidna: effective, usable, and fast fuzzing for smart contracts". In: *Proceedings of the 29th ACM SIGSOFT international symposium on software testing and analysis*. 557–560.

Grishchenko, I., M. Maffei, and C. Schneidewind. (2018). "A semantic framework for the security analysis of ethereum smart contracts". In: *Principles of Security and Trust: 7th International Conference, POST 2018, Held As Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2018, Thessaloniki, Greece, April 14-20, 2018, Proceedings 7*. Springer. 243–269.

Gritti, F., N. Ruaro, R. McLaughlin, P. Bose, D. Das, I. Grishchenko, C. Kruegel, and G. Vigna. (2023). "Confusum contractum: confused deputy vulnerabilities in ethereum smart contracts". In: *32nd USENIX Security Symposium (USENIX Security 23)*. 1793–1810.

Groce, A. (2021). "A Year in the Life of a Compiler Fuzzing Campaign". URL: https://blog.trailofbits.com/2021/03/23/a-year-in-the-life-of-a-compiler-fuzzing-campaign/.

Groce, A. (2024). "Variation of american fuzzy lop for testing compilers". URL: https://github.com/agroce/afl-compiler-fuzzer.

Full text available at: http://dx.doi.org/10.1561/3300000044

Grossman, S., I. Abraham, G. Golan-Gueta, Y. Michalevsky, N. Rinetzky, M. Sagiv, and Y. Zohar. (2017). "Online detection of effectively callback free objects with applications to smart contracts". *Proceedings of the ACM on Programming Languages*. 2(POPL): 1–28.

Gurfinkel, A. and N. Bjørner. (2019). "The science, art, and magic of constrained horn clauses". In: *2019 21st International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*. IEEE. 6–10.

Hafer, T. and W. Thomas. (1987). "Computation tree logic CTL* and path quantifiers in the monadic theory of the binary tree". In: *International Colloquium on Automata, Languages, and Programming*. Springer. 269–279.

Hajdu, Á. and D. Jovanovic. (2019). "solc-verify: A Modular Verifier for Solidity Smart Contracts". In: *Verified Software. Theories, Tools, and Experiments - 11th International Conference, VSTTE 2019, New York City, NY, USA, July 13-14, 2019, Revised Selected Papers*. Ed. by S. Chakraborty and J. A. Navas. Vol. 12031. *Lecture Notes in Computer Science*. Springer. 161–179. DOI: 10.1007/978-3-030-41600-3_11.

Hallberg, B. J. (2012). *Return to First Principles*. AuthorHouse.

Harbor. (2023). "Edition of series on security audits: Risk of precompiled contracts". URL: https://medium.com/coinmonks/edition-of-series-on-security-audits-risk-of-precompiled-contracts-c99b64006cef.

Harper, C. (2020). "Ethereum's 'Unannounced Hard Fork' Was Trying to Prevent the Very Disruption It Caused". URL: https://www.coindesk.com/tech/2020/11/11/ethereums-unannounced-hard-fork-was-trying-to-prevent-the-very-disruption-it-caused.

Hart, S. and M. Sharir. (1984). "Probabilistic temporal logics for finite and bounded models". In: *Proceedings of the sixteenth annual ACM symposium on Theory of computing*. 1–13.

Harz, D. and W. Knottenbelt. (2018). "Towards safer smart contracts: A survey of languages and verification methods". *arXiv preprint arXiv:1809.09805*.

Full text available at: http://dx.doi.org/10.1561/3300000044

He, J., M. Balunović, N. Ambroladze, P. Tsankov, and M. Vechev. (2019). "Learning to fuzz from symbolic execution with application to smart contracts". In: *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*. 531–548.

He, N., R. Zhang, H. Wang, L. Wu, X. Luo, Y. Guo, T. Yu, and X. Jiang. (2021). "{EOSAFE}: security analysis of {EOSIO} smart contracts". In: *30th USENIX security symposium (USENIX Security 21)*. 1271–1288.

Hildenbrandt, E., M. Saxena, N. Rodrigues, X. Zhu, P. Daian, D. Guth, B. Moore, D. Park, Y. Zhang, A. Stefanescu, *et al.* (2018). "Kevm: A complete formal semantics of the ethereum virtual machine". In: *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*. IEEE. 204–217.

Hirai, Y. (2017). "Defining the ethereum virtual machine for interactive theorem provers". In: *Financial Cryptography and Data Security: FC 2017 International Workshops, WAHC, BITCOIN, VOTING, WTSC, and TA, Sliema, Malta, April 7, 2017, Revised Selected Papers 21*. Springer. 520–535.

Hoare, C. A. R. (1978). "Communicating sequential processes". *Communications of the ACM*. 21(8): 666–677.

Huang, M., J. Chen, Z. Jiang, and Z. Zheng. (2024). "Revealing Hidden Threats: An Empirical Study of Library Misuse in Smart Contracts". In: *Proceedings of the 46th IEEE/ACM International Conference on Software Engineering*. 1–12.

Huet, G., G. Kahn, and C. Paulin-Mohring. (1997). "The coq proof assistant a tutorial". *Rapport Technique*. 178.

Imeri, A., N. Agoulmine, and D. Khadraoui. (2020). "Smart contract modeling and verification techniques: A survey". In: *8th International Workshop on ADVANCEs in ICT Infrastructures and Services (ADVANCE 2020)*. 1–8.

Immunefi. (2023). "Common Cross-Chain Bridge Vulnerabilities". URL: https://medium.com/immunefi/common-cross-chain-bridge-vulnerabilities-d8c161ffaf8f.

Inspex. (2022). "Cross-Contract Reentrancy Attack". URL: https://inspexco.medium.com/cross-contract-reentrancy-attack-402d27a02a15.

Full text available at: http://dx.doi.org/10.1561/3300000044

Iosiro. (2024). "Geth Out-of-Order EIP Application Denial-of-Service". URL: https://iosiro.com/blog/geth-out-of-order-eip-application-denial-of-service.

Jeltsch, W. (2019). "A process calculus for formally verifying blockchain consensus protocols". In: *International Conference on Applications of Declarative Programming and Knowledge Management*. Springer. 24–39.

Jiang, B., Y. Liu, and W. K. Chan. (2018). "Contractfuzzer: Fuzzing smart contracts for vulnerability detection". In: *Proceedings of the 33rd ACM/IEEE international conference on automated software engineering*. 259–269.

Jiao, J., S. Kan, S.-W. Lin, D. Sanan, Y. Liu, and J. Sun. (2020). "Semantic understanding of smart contracts: Executable operational semantics of solidity". In: *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE. 1695–1712.

Jiao, T., Z. Xu, M. Qi, S. Wen, Y. Xiang, and G. Nan. (2024). "A Survey of Ethereum Smart Contract Security: Attacks and Detection". *Distributed Ledger Technologies: Research and Practice*.

Jones, E. and D. Marmsoler. (2024). "Towards Mechanised Consensus in Isabelle". In: *5th International Workshop on Formal Methods for Blockchains, FMBC 2024, April 7, 2024, Luxembourg City, Luxembourg*. Ed. by B. Bernardo and D. Marmsoler. Vol. 118. *OASIcs*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik. 4:1–4:22. DOI: 10.4230/OASICS.FMBC.2024.4.

Jordan, H., B. Scholz, and P. Subotić. (2016). "Soufflé: On synthesis of program analyzers". In: *Computer Aided Verification: 28th International Conference, CAV 2016, Toronto, ON, Canada, July 17-23, 2016, Proceedings, Part II 28*. Springer. 422–430.

Jurafsky, D. and J. H. Martin. (2024). "Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition".

Full text available at: http://dx.doi.org/10.1561/3300000044

Kado, C., N. Yanai, J. P. Cruz, and S. Okamura. (2023). "An Empirical Study of Impact of Solidity Compiler Updates on Vulnerabilities". In: *IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events, PerCom Workshops 2023, Atlanta, GA, USA, March 13-17, 2023*. IEEE. 92–97. DOI: 10.1109/PERCOMWORKSHOPS56833.2023.10150389.

Kalra, S., S. Goel, M. Dhawan, and S. Sharma. (2018). "Zeus: analyzing safety of smart contracts." In: *Ndss*. 1–12.

Ke, C.-S. and Y.-R. Chen. (2020). "Instruction Verification of Ethereum Virtual Machine by Formal Method". In: *2020 Indo–Taiwan 2nd International Conference on Computing, Analytics and Networks (Indo-Taiwan ICAN)*. IEEE. 69–74.

Keidar, I., E. Kokoris-Kogias, O. Naor, and A. Spiegelman. (2021). "All you need is dag". In: *Proceedings of the 2021 ACM Symposium on Principles of Distributed Computing*. 165–175.

Komuravelli, A., A. Gurfinkel, and S. Chaki. (2016). "SMT-based model checking for recursive programs". *Formal Methods in System Design*. 48: 175–205.

Kong, Q., J. Chen, Y. Wang, Z. Jiang, and Z. Zheng. (2023). "Defitainter: Detecting price manipulation vulnerabilities in defi protocols". In: *Proceedings of the 32nd ACM SIGSOFT International Symposium on Software Testing and Analysis*. 1144–1156.

Konnov, I., M. Lazić, H. Veith, and J. Widder. (2017). "A short counterexample property for safety and liveness verification of fault-tolerant distributed algorithms". In: *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages*. 719–734.

Kremer, S. and R. Künnemann. (2016). "Automated analysis of security protocols with global state". *Journal of Computer Security*. 24(5): 583–616.

Krishna, A. (2024). "Blockchain Security Issues: A Complete Guide". URL: https://www.getastra.com/blog/knowledge-base/blockchain-security-issues/.

Full text available at: http://dx.doi.org/10.1561/3300000044

Kwiatkowska, M., G. Norman, and D. Parker. (2004). "Probabilistic symbolic model checking with PRISM: A hybrid approach". *International journal on software tools for technology transfer*. 6: 128–142.

Kwiatkowska, M., G. Norman, and D. Parker. (2010). "Advances and challenges of probabilistic model checking". In: *2010 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE. 1691–1698.

Lahiri, S. K., K. Vaswani, and C. A. Hoare. (2010). "Differential static analysis: Opportunities, applications, and challenges". In: *Proceedings of the FSE/SDP workshop on Future of software engineering research*. 201–204.

Lamport, L. (1983). "What good is temporal logic?" In: *IFIP congress*. Vol. 83. 657–668.

Lamport, L. (2021). "A High-Level View of TLA+". URL: https://lamport.azurewebsites.net/tla/high-level-view.html.

Laneve, C. and A. Veschetti. (2020). "A Formal Analysis of Blockchain Consensus". *AFABC. pdf*.

Larsen, K. G., P. Pettersson, and W. Yi. (1997). "UPPAAL in a nutshell". *International journal on software tools for technology transfer*. 1: 134–152.

Lashkari, B. and P. Musilek. (2021). "A comprehensive review of blockchain consensus mechanisms". *IEEE access*. 9: 43620–43652.

Laszka, A., S. Eisele, A. Dubey, G. Karsai, and K. Kvaternik. (2018). "TRANSAX: A blockchain-based decentralized forward-trading energy exchanged for transactive microgrids". In: *2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS)*. IEEE. 918–927.

Li, B., Z. Pan, and T. Hu. (2022). "ReDefender: Detecting Reentrancy Vulnerabilities in Smart Contracts Automatically". *IEEE Trans. Reliab.* 71(2): 984–999. DOI: 10.1109/TR.2022.3161634.

Li, H., Y. Chen, X. Shi, X. Bai, N. Mo, W. Li, R. Guo, Z. Wang, and Y. Sun. (2023). "Fisco-bcos: An enterprise-grade permissioned blockchain system with high-performance". In: *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. 1–17.

Full text available at: http://dx.doi.org/10.1561/3300000044

Li, K., J. Chen, X. Liu, Y. R. Tang, X. Wang, and X. Luo. (2021a). "As Strong As Its Weakest Link: How to Break Blockchain DApps at RPC Service". In: *28th Annual Network and Distributed System Security Symposium, NDSS 2021, virtually, February 21-25, 2021*. The Internet Society. URL: https://www.ndss-symposium.org/ndss-paper/as-strong-as-its-weakest-link-how-to-break-blockchain-dapps-at-rpc-service/.

Li, K., Y. Tang, J. Chen, Y. Wang, and X. Liu. (2021b). "TopoShot: uncovering Ethereum's network topology leveraging replacement transactions". In: *IMC '21: ACM Internet Measurement Conference, Virtual Event, USA, November 2-4, 2021*. Ed. by D. Levin, A. Mislove, J. Amann, and M. Luckie. ACM. 302–319. DOI: 10.1145/3487552.3487814.

Li, K., Y. Wang, and Y. Tang. (2021c). "DETER: Denial of Ethereum Txpool sERvices". In: *CCS '21: 2021 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, Republic of Korea, November 15 - 19, 2021*. Ed. by Y. Kim, J. Kim, G. Vigna, and E. Shi. ACM. 1645–1667. DOI: 10.1145/3460120.3485369.

Li, X., J. Yang, J. Chen, Y. Tang, and X. Gao. (2024). "Characterizing Ethereum Upgradable Smart Contracts and Their Security Implications". In: *Proceedings of the ACM on Web Conference 2024, WWW 2024, Singapore, May 13-17, 2024*. Ed. by T. Chua, C. Ngo, R. Kumar, H. W. Lauw, and R. K. Lee. ACM. 1847–1858. DOI: 10.1145/3589334.3645640.

Li, X., C. Su, Y. Xiong, W. Huang, and W. Wang. (2019). "Formal verification of BNB smart contract". In: *2019 5th International Conference on Big Data Computing and Communications (BIGCOM)*. IEEE. 74–78.

Liao, Z., Y. Nan, H. Liang, S. Hao, J. Zhai, J. Wu, and Z. Zheng. (2024). "SmartAxe: Detecting Cross-Chain Vulnerabilities in Bridge Smart Contracts via Fine-Grained Static Analysis". *Proceedings of the ACM on Software Engineering*. 1(FSE): 249–270.

Liu, C., H. Liu, Z. Cao, Z. Chen, B. Chen, and B. Roscoe. (2018). "Reguard: finding reentrancy bugs in smart contracts". In: *Proceedings of the 40th International Conference on Software Engineering: Companion Proceeedings*. 65–68.

Full text available at: http://dx.doi.org/10.1561/3300000044

Luo, Y., W. Xu, K. Andersson, M. S. Hossain, and D. Xu. (2024). "FELLMVP: An Ensemble LLM Framework for Classifying Smart Contract Vulnerabilities". In: *2024 IEEE International Conference on Blockchain (Blockchain)*. IEEE. 89–96.

Lutz, O., H. Chen, H. Fereidooni, C. Sendner, A. Dmitrienko, A. R. Sadeghi, and F. Koushanfar. (2021). "ESCORT: Ethereum Smart COntRacTs Vulnerability Detection using Deep Neural Network and Transfer Learning". URL: https://arxiv.org/abs/2103.12607.

Luu, L., D.-H. Chu, H. Olickel, P. Saxena, and A. Hobor. (2016). "Making smart contracts smarter". In: *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. 254–269.

Ma, F., Y. Chen, M. Ren, Y. Zhou, Y. Jiang, T. Chen, H. Li, and J. Sun. (2023). "LOKI: State-Aware Fuzzing Framework for the Implementation of Blockchain Consensus Protocols". In: *30th Annual Network and Distributed System Security Symposium, NDSS 2023, San Diego, California, USA, February 27 - March 3, 2023*. The Internet Society. URL: https://www.ndss-symposium.org/ndss-paper/loki-state-aware-fuzzing-framework-for-the-implementation-of-blockchain-consensus-protocols/.

Ma, H., W. Zhang, Q. Shen, Y. Tian, J. Chen, and S. Cheung. (2024). "Towards Understanding the Bugs in Solidity Compiler". In: *Proceedings of the 33rd ACM SIGSOFT International Symposium on Software Testing and Analysis, ISSTA 2024, Vienna, Austria, September 16-20, 2024*. Ed. by M. Christakis and M. Pradel. ACM. 1312–1324. DOI: 10.1145/3650212.3680362.

Maier, D., F. Fäßler, and J.-P. Seifert. (2021). "Uncovering Smart Contract VM Bugs Via Differential Fuzzing". In: *Reversing and Offensive-oriented Trends Symposium*. 11–22.

Mavridou, A. and A. Laszka. (2018). "Designing secure ethereum smart contracts: A finite state machine based approach". In: *Financial Cryptography and Data Security: 22nd International Conference, FC 2018, Nieuwpoort, Curaçao, February 26–March 2, 2018, Revised Selected Papers 22*. Springer. 523–540.

Full text available at: http://dx.doi.org/10.1561/3300000044

Mavridou, A., A. Laszka, E. Stachtiari, and A. Dubey. (2019). "VeriSolid: Correct-by-design smart contracts for Ethereum". In: *Financial Cryptography and Data Security: 23rd International Conference, FC 2019, Frigate Bay, St. Kitts and Nevis, February 18–22, 2019, Revised Selected Papers 23*. Springer. 446–465.

Mazieres, D. (2015). "The stellar consensus protocol: A federated model for internet-level consensus". *Stellar Development Foundation*. 32: 1–45.

McKeeman, W. M. (1998). "Differential testing for software". *Digital Technical Journal*. 10(1): 100–107.

Meier, S., B. Schmidt, C. Cremers, and D. Basin. (2013). "The TAMARIN prover for the symbolic analysis of security protocols". In: *Computer Aided Verification: 25th International Conference, CAV 2013, Saint Petersburg, Russia, July 13-19, 2013. Proceedings 25*. Springer. 696–701.

Meisami, S., H. Dabadie, S. Li, Y. Tang, and Y. Duan. (2025). "SigScope: Detecting and Understanding Off-Chain Message Signing-related Vulnerabilities in Decentralized Applications". In: *THE WEB CONFERENCE 2025*. URL: https://openreview.net/forum?id=8OIqXq455O.

Miller, C., Z. N. Peterson, *et al.* (2007). "Analysis of mutation and generation-based fuzzing". *Independent Security Evaluators, Tech. Rep.* 4.

Monarch. (2017). "King of the Ether". URL: https://www.kingoftheether.com/thrones/kingoftheether/index.html.

Mossberg, M., F. Manzano, E. Hennenfent, A. Groce, G. Grieco, J. Feist, T. Brunson, and A. Dinaburg. (2019). "Manticore: A user-friendly symbolic execution framework for binaries and smart contracts". In: *2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE. 1186–1189.

Mueller, B. (2017). "Mythril: security analysis tool for evm bytecode".

Mulligan, D. P., S. Owens, K. E. Gray, T. Ridge, and P. Sewell. (2014). "Lem: reusable engineering of real-world semantics". *ACM SIGPLAN Notices*. 49(9): 175–188.

Full text available at: http://dx.doi.org/10.1561/3300000044

Murray, Y. and D. A. Anisi. (2019). "Survey of formal verification methods for smart contracts on blockchain". In: *2019 10th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*. IEEE. 1–6.

Nakamoto, S. (2008). "Bitcoin: A Peer-to-Peer Electronic Cash System". URL: https://bitcoin.org/bitcoin.pdf.

Nehaí, Z., F. Bobot, S. Tucci-Piergiovanni, C. Delporte-Gallet, and H. Fauconnier. (2022). "A tla+ formal proof of a cross-chain swap". In: *Proceedings of the 23rd International Conference on Distributed Computing and Networking*. 148–159.

Neo. (2024). "NeoVM - NEO Developer Resource". URL: https://developers.neo.org/docs/n3/foundation/neovm.

Nervos. (2024). "CKB Consensus". URL: https://docs-old.nervos.org/docs/basics/concepts/consensus.

Nguyen, T. D., L. H. Pham, and J. Sun. (2021). "SGUARD: towards fixing vulnerable smart contracts automatically". In: *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE. 1215–1229.

Nguyen, T. D., L. H. Pham, J. Sun, Y. Lin, and Q. T. Minh. (2020). "sfuzz: An efficient adaptive fuzzer for solidity smart contracts". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. 778–788.

Nielsen, J. B. and B. Spitters. (2019). "Smart contract interactions in Coq". In: *International Symposium on Formal Methods*. Springer. 380–391.

Nielson, F., H. R. Nielson, and C. Hankin. (2015). *Principles of program analysis*. springer.

Nikolic, I., A. Kolluri, I. Sergey, P. Saxena, and A. Hobor. (2018). "Finding The Greedy, Prodigal, and Suicidal Contracts at Scale". In: *Proceedings of the 34th Annual Computer Security Applications Conference, ACSAC 2018, San Juan, PR, USA, December 03-07, 2018*. ACM. 653–663. DOI: 10.1145/3274694.3274743.

NIST. (2018). "CVE-2018-10376 Detail". URL: https://nvd.nist.gov/vuln/detail/CVE-2018-10376.

NIST. (2020a). "NVD - CVE-2020-26241". URL: https://nvd.nist.gov/vuln/detail/CVE-2020-26241.

Full text available at: http://dx.doi.org/10.1561/3300000044

NIST. (2020b). "NVD - CVE-2020-26265". URL: https://nvd.nist.gov/vuln/detail/CVE-2020-26265.

NIST. (2021). "NVD - CVE-2021-39137". URL: https://nvd.nist.gov/vuln/detail/CVE-2021-39137.

NIST. (2022). "NVD - CVE-2022-26534". URL: https://nvd.nist.gov/vuln/detail/CVE-2022-26534.

OpenZeppelin. (2024). "Math - OpenZeppelin Docs". URL: https://docs.openzeppelin.com/contracts/2.x/api/math.

Palladino, S. (2017). "The Parity Wallet Hack Explained". URL: https://blog.openzeppelin.com/on-the-parity-wallet-multisig-hack-405a8c12e8f7.

Pcaversaccio. (2024). "A chronological and (hopefully) complete list of reentrancy attacks to date." URL: https://github.com/pcaversaccio/reentrancy-attacks?tab=readme-ov-file.

Permenev, A., D. Dimitrov, P. Tsankov, D. Drachsler-Cohen, and M. Vechev. (2020). "Verx: Safety verification of smart contracts". In: *2020 IEEE symposium on security and privacy (SP)*. IEEE. 1661–1677.

Pillai, B., Z. Hóu, K. Biswas, and V. Muthukkumarasamy. (2023). "Formal Verification of the Burn-to-Claim Blockchain Interoperable Protocol". In: *International Conference on Formal Engineering Methods*. Springer. 249–254.

Plotkin, G. D. (1981). "A structural approach to operational semantics".

Popescu, M.-C., V. E. Balas, L. Perescu-Popescu, and N. Mastorakis. (2009). "Multilayer perceptron and neural networks". *WSEAS Transactions on Circuits and Systems*. 8(7): 579–588.

Praitheeshan, P., L. Pan, J. Yu, J. Liu, and R. Doss. (2019). "Security analysis methods on ethereum smart contract vulnerabilities: a survey". *arXiv preprint arXiv:1908.08605*.

Priami, C. (1995). "Stochastic π-calculus". *The Computer Journal*. 38(7): 578–589.

Full text available at: http://dx.doi.org/10.1561/3300000044

Pusceddu, D. and M. Bartoletti. (2024). "Formalizing Automated Market Makers in the Lean 4 Theorem Prover". In: *5th International Workshop on Formal Methods for Blockchains, FMBC 2024, April 7, 2024, Luxembourg City, Luxembourg*. Ed. by B. Bernardo and D. Marmsoler. Vol. 118. *OASIcs*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik. 5:1–5:13. DOI: 10.4230/OASICS.FMBC.2024.5.

Puterman, M. L. (1990). "Markov decision processes". *Handbooks in operations research and management science*. 2: 331–434.

Qian, P., Z. Liu, Q. He, B. Huang, D. Tian, and X. Wang. (2022). "Smart contract vulnerability detection technique: A survey". *arXiv preprint arXiv:2209.05872*.

Qin, K., S. Chaliasos, L. Zhou, B. Livshits, D. Song, and A. Gervais. (2023). "The blockchain imitation game". In: *32nd USENIX Security Symposium (USENIX Security 23)*. 3961–3978.

Qin, K., L. Zhou, B. Livshits, and A. Gervais. (2021). "Attacking the DeFi Ecosystem with Flash Loans for Fun and Profit". In: *Financial Cryptography and Data Security - 25th International Conference, FC 2021, Virtual Event, March 1-5, 2021, Revised Selected Papers, Part I*. Ed. by N. Borisov and C. Díaz. Vol. 12674. *Lecture Notes in Computer Science*. Springer. 3–32. DOI: 10.1007/978-3-662-64322-8_1.

Rodler, M., W. Li, G. O. Karame, and L. Davi. (2019). "Sereum: Protecting Existing Smart Contracts Against Re-Entrancy Attacks". In: *Proceedings 2019 Network and Distributed System Security Symposium*. Internet Society.

Rodler, M., W. Li, G. O. Karame, and L. Davi. (2021). "{EVMPatch}: Timely and automated patching of ethereum smart contracts". In: *30th usenix security symposium (USENIX Security 21)*. 1289–1306.

Roșu, G. and T. F. Șerbănută. (2010). "An overview of the K semantic framework". *The Journal of Logic and Algebraic Programming*. 79(6): 397–434.

Ruaro, N., F. Gritti, R. McLaughlin, I. Grishchenko, C. Kruegel, and G. Vigna. (2024a). "Not your Type! Detecting Storage Collision Vulnerabilities in Ethereum Smart Contracts". In: *Proceedings of the Network and Distributed System Security (NDSS) Symposium*.

Full text available at: http://dx.doi.org/10.1561/3300000044

Ruaro, N., F. Gritti, R. McLaughlin, I. Grishchenko, C. Kruegel, and G. Vigna. (2024b). "Not your Type! Detecting Storage Collision Vulnerabilities in Ethereum Smart Contracts". In: *Netw. Distrib. Syst. Security Symp.*

Samreen, N. F. and M. H. Alalfi. (2020). "A survey of security vulnerabilities in ethereum smart contracts". In: *Proceedings of the 30th Annual International Conference on Computer Science and Software Engineering*. 73–82.

Samreen, N. F. and M. H. Alalfi. (2021). "A survey of security vulnerabilities in ethereum smart contracts". *arXiv preprint arXiv:2105.06974*.

Schneidewind, C., I. Grishchenko, M. Scherer, and M. Maffei. (2020). "ethor: Practical and provably sound static analysis of ethereum smart contracts". In: *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. 621–640.

Secure3. (2023). "Analysis of the Blockchain Security Chain Events Caused by Vyper Compiler Vulnerabilities". URL: https://medium.com/@Secure3/analysis-of-the-blockchain-security-chain-events-caused-by-vyper-compiler-vulnerabilities-37b66ad8aa45.

Sen, K., M. Viswanathan, and G. Agha. (2004). "Statistical model checking of black-box probabilistic systems". In: *Computer Aided Verification: 16th International Conference, CAV 2004, Boston, MA, USA, July 13-17, 2004. Proceedings 16*. Springer. 202–215.

Sendner, C., H. Chen, H. Fereidooni, L. Petzi, J. König, J. Stang, A. Dmitrienko, A.-R. Sadeghi, and F. Koushanfar. (2023). "Smarter Contracts: Detecting Vulnerabilities in Smart Contracts with Deep Transfer Learning." In: *NDSS*.

Serah, O. (2023). "Mastering Delegatecall in Solidity: A Comprehensive Guide with EVM Walkthrough". URL: https://medium.com/@ajaotosinserah/mastering-delegatecall-in-solidity-a-comprehensive-guide-with-evm-walkthrough-6ddf027175c7.

Serebryany, K. (2016). "Continuous fuzzing with libfuzzer and addresssanitizer". In: *2016 IEEE Cybersecurity Development (SecDev)*. IEEE. 157–157.

SlowMist. (2021). "The Analysis and Q&A Of Poly Network Being Hacked". URL: https://slowmist.medium.com/the-analysis-and-q-a-of-poly-network-being-hacked-8112a35beb39.

Full text available at: http://dx.doi.org/10.1561/3300000044

Smith, C. (2024). "Proof-of-stake (PoS)". URL: https://ethereum.org/en/developers/docs/consensus-mechanisms/pos/.

Smolka, S., J.-R. Giesen, P. Winkler, O. Draissi, L. Davi, G. Karame, and K. Pohl. (2023). "Fuzz on the beach: Fuzzing solana smart contracts". In: *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*. 1197–1211.

So, S., S. Hong, and H. Oh. (2021). "{SmarTest}: Effectively hunting vulnerable transaction sequences in smart contracts through language {Model-Guided} symbolic execution". In: *30th USENIX Security Symposium (USENIX Security 21)*. 1361–1378.

So, S., M. Lee, J. Park, H. Lee, and H. Oh. (2020). "Verismart: A highly precise safety verifier for ethereum smart contracts". In: *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE. 1678–1694.

SOL-2020-5. (2024). "Solidity compiler flaw SOL-2020-5". URL: https://github.com/ethereum/solidity/blob/develop/docs/bugs.json/%5C#L224.

Solidity. (2024a). "Function Modifiers". URL: https://docs.soliditylang.org/en/latest/contracts.html%5C#function-modifiers.

Solidity. (2024b). "Sending Ether (transfer, send, call)". URL: https://solidity-by-example.org/sending-ether/.

Solidity. (2024c). "Solidity, the Smart Contract Programming Language". URL: https://github.com/ethereum/solidity.

Stephens, J., K. Ferles, B. Mariano, S. Lahiri, and I. Dillig. (2021). "SmartPulse: automated checking of temporal properties in smart contracts". In: *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE. 555–571.

Su, L., X. Shen, X. Du, X. Liao, X. Wang, L. Xing, and B. Liu. (2021). "Evil under the sun: Understanding and discovering attacks on ethereum decentralized applications". In: *30th USENIX Security Symposium (USENIX Security 21)*. 1307–1324.

Sun, J., Y. Liu, J. S. Dong, and C. Chen. (2009a). "Integrating specification and programs for system modeling and verification". In: *2009 Third IEEE International Symposium on Theoretical Aspects of Software Engineering*. IEEE. 127–135.

Full text available at: http://dx.doi.org/10.1561/3300000044

Sun, J., Y. Liu, J. S. Dong, and J. Pang. (2009b). "PAT: Towards flexible verification under fairness". In: *Computer Aided Verification: 21st International Conference, CAV 2009, Grenoble, France, June 26-July 2, 2009. Proceedings 21*. Springer. 709–714.

Sun, M., Y. Lu, Y. Feng, Q. Zhang, and S. Liu. (2021). "Modeling and verifying the CKB blockchain consensus protocol". *Mathematics*. 9(22): 2954.

Sun, T. and W. Yu. (2020). "A formal verification framework for security issues of blockchain smart contracts". *Electronics*. 9(2): 255.

Sun, Y., D. Wu, Y. Xue, H. Liu, H. Wang, Z. Xu, X. Xie, and Y. Liu. (2024). "Gptscan: Detecting logic vulnerabilities in smart contracts by combining gpt with program analysis". In: *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*. 1–13.

SWC. (2024). "SWC-115 - Smart Contract Weakness Classification (SWC)". URL: https://swcregistry.io/docs/SWC-115/.

Thin, W. Y. M. M., N. Dong, G. Bai, and J. S. Dong. (2018). "Formal analysis of a proof-of-stake blockchain". In: *2018 23rd International Conference on Engineering of Complex Computer Systems (ICECCS)*. IEEE. 197–200.

Thummavet, P. (2022). "Solidity Security By Example #04: Cross-Function Reentrancy". URL: https://medium.com/valixconsulting/solidity-smart-contract-security-by-example-04-cross-function-reentrancy-de9cbce0558e.

Tian, Z., F. Wang, Y. Chen, and L. Chen. (2024). "Differential testing solidity compiler through deep contract manipulation and mutation". *Software Quality Journal*: 1–26.

Tolmach, P., Y. Li, S.-W. Lin, Y. Liu, and Z. Li. (2021). "A survey of smart contract formal specification and verification". *ACM Computing Surveys (CSUR)*. 54(7): 1–38.

Tsankov, P., A. Dan, D. Drachsler-Cohen, A. Gervais, F. Buenzli, and M. Vechev. (2018). "Securify: Practical security analysis of smart contracts". In: *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*. 67–82.

Ullman Jeffrey, D. and W. Freeman. (1988). "Principles of database and knowledge-base systems".

Full text available at: http://dx.doi.org/10.1561/3300000044

Verge. (2018). "Verge (XVG) Time-Warp Exploit (2018)". URL: https://www.reddit.com/r/CryptoCurrency/comments/avvmb9/verge_xvg_has_lost_94_percent_of_its_value_after/.

Verma, S., D. Yadav, and G. Chandra. (2022). "Introduction of formal methods in blockchain consensus mechanism and its associated protocols". *IEEE Access*. 10: 66611–66624.

Veschetti, A. (2022). "Source Code of PRISM+". URL: https://github.com/adeleveschetti/bitcoin-analysis/tree/master.

Veschetti, A. (2023). "A formal analysis of blockchain consensus". URL: http://amsdottorato.unibo.it/10835/.

Vyper. (2024). "Compiling a Contract". URL: https://docs.vyperlang.org/en/stable/compiling-a-contract.html.

Wan, Z., D. Lo, X. Xia, and L. Cai. (2017). "Bug characteristics in blockchain systems: a large-scale empirical study". In: *Proceedings of the 14th International Conference on Mining Software Repositories, MSR 2017, Buenos Aires, Argentina, May 20-28, 2017*. Ed. by J. M. González-Barahona, A. Hindle, and L. Tan. IEEE Computer Society. 413–424. DOI: 10.1109/MSR.2017.59.

Wang, D., S. Wu, Z. Lin, L. Wu, X. Yuan, Y. Zhou, H. Wang, and K. Ren. (2021a). "Towards a first step to understand flash loan and its applications in defi ecosystem". In: *Proceedings of the Ninth International Workshop on Security in Blockchain and Cloud Computing*. 23–28.

Wang, P. *et al.* (2019a). "Type system for resource bounds with type-preserving compilation". *PhD thesis*. Massachusetts Institute of Technology.

Wang, S., C. Zhang, and Z. Su. (2019b). "Detecting nondeterministic payment bugs in ethereum smart contracts". *Proceedings of the ACM on Programming Languages*. 3(OOPSLA): 1–29.

Wang, W., W. Huang, Z. Meng, Y. Xiong, F. Miao, X. Fang, C. Tu, and R. Ji. (2023). "Automated inference on financial security of Ethereum smart contracts". In: *32nd USENIX Security Symposium (USENIX Security 23)*. 3367–3383.

Full text available at: http://dx.doi.org/10.1561/3300000044

Wang, Y., Y. Tang, K. Li, W. Ding, and Z. Yang. (2024). "Understanding Ethereum Mempool Security under Asymmetric DoS by Symbolized Stateful Fuzzing". In: *33rd USENIX Security Symposium, USENIX Security 2024, Philadelphia, PA, USA, August 14-16, 2024*. Ed. by D. Balzarotti and W. Xu. USENIX Association. URL: https://www.usenix.org/conference/usenixsecurity24/presentation/wang-yibo.

Wang, Y., Q. Zhang, K. Li, Y. Tang, J. Chen, X. Luo, and T. Chen. (2021b). "iBatch: saving Ethereum fees via secure and cost-effective batching of smart-contract invocations". In: *ESEC/FSE '21: 29th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, Athens, Greece, August 23-28, 2021*. Ed. by D. Spinellis, G. Gousios, M. Chechik, and M. D. Penta. ACM. 566–577. DOI: 10.1145/3468264.3468568.

Wei, Q., X. Zhao, X.-Y. Zhu, and W. Zhang. (2023). "Formal Analysis of IBC Protocol". In: *2023 IEEE 31st International Conference on Network Protocols (ICNP)*. IEEE. 1–11.

Wolper, P. (1985). "The tableau method for temporal logic: An overview". *Logique et Analyse*: 119–136.

Wood, G. *et al.* (2014). "Ethereum: A secure decentralised generalised transaction ledger". *Ethereum project yellow paper*. 151(2014): 1–32.

Wu, S., D. Wang, J. He, Y. Zhou, L. Wu, X. Yuan, Q. He, and K. Ren. (2021). "Defiranger: Detecting price manipulation attacks on defi applications". *arXiv preprint arXiv:2104.15068*.

Wu, S., L. Wu, Y. Zhou, R. Li, Z. Wang, X. Luo, C. Wang, and K. Ren. (2022). "Time-travel investigation: Toward building a scalable attack detection framework on Ethereum". *ACM Transactions on Software Engineering and Methodology (TOSEM)*. 31(3): 1–33.

Wu, S., Z. Yu, D. Wang, Y. Zhou, L. Wu, H. Wang, and X. Yuan. (2023). "DeFiRanger: Detecting DeFi Price Manipulation Attacks". *IEEE Transactions on Dependable and Secure Computing*.

Wüstholz, V. and M. Christakis. (2020). "Harvey: a greybox fuzzer for smart contracts". In: *ESEC/FSE '20: 28th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, Virtual Event, USA, November 8-13, 2020*. Ed. by P. Devanbu, M. B. Cohen, and T. Zimmermann. ACM. 1398–1409. DOI: 10.1145/3368089.3417064.

Full text available at: http://dx.doi.org/10.1561/3300000044

Xi, R., Z. Wang, and K. Pattabiraman. (2024). "POMABuster: Detecting Price Oracle Manipulation Attacks in Decentralized Finance". In: *2024 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society. 240–240.

Xscope. (2022). "Results: Xscope: Hunting for Cross-Chain Bridge Attacks". URL: https://github.com/Xscope-Tool/Results?tab=readme-ov-file.

Xue, Y., M. Ma, Y. Lin, Y. Sui, J. Ye, and T. Peng. (2020). "Cross-contract static analysis for detecting practical reentrancy vulnerabilities in smart contracts". In: *Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering*. 1029–1040.

Yang, X., Y. Chen, E. Eide, and J. Regehr. (2011). "Finding and understanding bugs in C compilers". In: *Proceedings of the 32nd ACM SIGPLAN conference on Programming language design and implementation*. 283–294.

Yang, Y., T. Kim, and B.-G. Chun. (2021). "Finding consensus bugs in ethereum via multi-transaction differential fuzzing". In: *15th USENIX Symposium on Operating Systems Design and Implementation (OSDI 21)*. 349–365.

Yoo, J., Y. Jung, D. Shin, M. Bae, and E. Jee. (2019). "Formal modeling and verification of a federated byzantine agreement algorithm for blockchain platforms". In: *2019 IEEE International Workshop on Blockchain Oriented Software Engineering (IWBOSE)*. IEEE. 11–21.

Zhang, J., J. Gao, Y. Li, Z. Chen, Z. Guan, and Z. Chen. (2022). "Xscope: Hunting for cross-chain bridge attacks". In: *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering*. 1–4.

Zhang, M., X. Zhang, J. Barbee, Y. Zhang, and Z. Lin. (2023a). "SoK: Security of Cross-chain Bridges: Attack Surfaces, Defenses, and Open Problems". *arXiv preprint arXiv:2312.12573*.

Zhang, M., X. Zhang, Y. Zhang, and Z. Lin. (2020a). "{TXSPECTOR}: Uncovering attacks in ethereum from transactions". In: *29th USENIX Security Symposium (USENIX Security 20)*. 2775–2792.

Full text available at: http://dx.doi.org/10.1561/3300000044

Zhang, X., Y. Li, and M. Sun. (2020b). "Towards a formally verified EVM in production environment". In: *Coordination Models and Languages: 22nd IFIP WG 6.1 International Conference, COORDINATION 2020, Held as Part of the 15th International Federated Conference on Distributed Computing Techniques, DisCoTec 2020, Valletta, Malta, June 15–19, 2020, Proceedings 22*. Springer. 341–349.

Zhang, Z., Z. Lin, M. Morales, X. Zhang, and K. Zhang. (2023b). "Your exploit is mine: instantly synthesizing counterattack smart contract". In: *32nd USENIX Security Symposium (USENIX Security 23)*. 1757–1774.

Zhang, Z., B. Zhang, W. Xu, and Z. Lin. (2023c). "Demystifying Exploitable Bugs in Smart Contracts". In: *45th IEEE/ACM International Conference on Software Engineering, ICSE 2023, Melbourne, Australia, May 14-20, 2023*. IEEE. 615–627. DOI: 10.1109/ICSE48619.2023.00061.

Zhou, L., X. Xiong, J. Ernstberger, S. Chaliasos, Z. Wang, Y. Wang, K. Qin, R. Wattenhofer, D. Song, and A. Gervais. (2023). "SoK: Decentralized Finance (DeFi) Attacks". In: *44th IEEE Symposium on Security and Privacy, SP 2023, San Francisco, CA, USA, May 21-25, 2023*. IEEE. 2444–2461. DOI: 10.1109/SP46215.2023.10179435.