A Taxonomy of Blockchain Incentive Vulnerabilities for Networked Intelligent Systems

Hector Roussille, Önder Gürcan, Fabien Michel

A Taxonomy of Blockchain Incentive Vulnerabilities for Networked Intelligent Systems

Hector Roussille, *Member, IEEE,* Önder Gürcan, *Member, IEEE,*
Fabien Michel, *Member, IEEE*

***Abstract*—This paper presents a taxonomy of incentive vulnerabilities that can affect public and consortium blockchain-based networked intelligent systems. The taxonomy aims to help researchers and developers better understand the related threats and design more secure systems. To this end, the proposed taxonomy is grounded in a generic multi-agent organizational model for blockchain systems (AGR4BS) and establishes a relationship between the vulnerabilities and the dedicated agent roles. We expressed the vulnerabilities as behavior deviations and classified them according to the roles and behaviors identified in AGR4BS to form the categories and refine the subcategories of the taxonomy. The proposed taxonomy is novel and distinctively different from other taxonomies found in the literature.**

***Index Terms*—Autonomous Systems, Blockchain, Vulnerabilities, Security, Incentives, Taxonomy, Roles**

I. Introduction

NETWORKED intelligent systems connect smart physical and software systems to the Internet, allowing them to interact and cooperate. They have created new opportunities for innovation in various industries, from smart homes and cities to supply chain management and healthcare. Blockchain technology, provides a secure and transparent way to store and manage data, has emerged as a critical enabling technology for interconnected intelligent systems and their applications.

Blockchain is an attractive technology since it maintains a public, append-only, immutable, and ordered log of transactions which guarantees an auditable ledger accessible by anyone. The rise in popularity of blockchain systems motivates the development of new applications with use cases from mere cryptocurrencies to smart contract-based decentralized financial systems attracting retail and professional investors. As blockchain systems and blockchain-based applications are gaining popularity, more participants are joining these systems since various mechanisms incentivize them. However, blockchain systems are vulnerable in multiple ways, with consequences ranging from a simple slowdown to theft estimated in hundreds of millions of dollars or simply halting the system.

In recent years, numerous reported exploits have been targeting blockchain systems [1]. Because of these, the security of blockchain systems has gained prominence and priority. Since blockchains are primarily open-source projects, attackers have a vector to exploit blockchain participants' honest/nominal behaviors as they can freely access the implementation source code, and also enter the system without any restrictions in the case of public blockchains.

Blockchains are also socio-economic systems [2], so malicious participants may exploit existing incentive vulnerabilities. Incentives refer to reward and punishment mechanisms that channel rationality towards the nominal behavior.

In the literature, a vulnerability is a defect that can produce undesired and incorrect behavior. Therefore, an *incentive vulnerability* can be defined as a misalignment between the agents' behavior, as expected by the protocol designers, and the rational one deliberately following a utility-based interpretation of the incentives. Such a vulnerability leads to public incentivization of the deviated behavior for every rational participant, making it a more serious and widespread issue than a bug exploitation for example. Besides, an exploit is a process by which one or more vulnerabilities are exploited to attack a system with malicious intent or optimize a selfish objective with similar consequences but without harmful intentions. Identifying such vulnerabilities is essential to prevent exploits, but not a trivial task due to the diversity and interrelationships of actors. Moreover, agents might exploit vulnerabilities that strictly follow the system's rationales or aim to impact the blockchain system without showing any rationality towards it. Understanding the relationships of such vulnerabilities with the incentives and how they are exploited appears crucial to secure blockchain systems. Besides, such an understanding can facilitate developing a framework, exposed, for example, as a set of blockchain environments, for deriving multi-agent strategies (*e.g.*, Reinforcement Learning) to assess blockchain system security and automate attack discovery.

Therefore, this paper presents a role-based taxonomy of incentive vulnerabilities in blockchain systems. We focus mainly on public and consortium blockchains as private ones do not necessarily rely on on-chain incentives. The taxonomy is grounded in a generic multi-agent organizational model for blockchain systems called AGR4BS [3], composed of three first-class abstractions: Agent, group, and role. AGR4BS identifies the behaviors of each role that, in this study, are subject to deviations for exploitation. The contributions of this article are as follows:

- We systematically explore the incentive vulnerabilities of blockchain technology, with an emphasis on the roles played by blockchain participants.
- We identify possible behavior deviations for each role and link them with known or possible vulnerabilities.

H. Roussille and Ö. Gürcan are with Université Paris-Saclay, CEA, List, F-91120, Palaiseau, France. Emails: {hector.roussille; onder.gurcan}@cea.fr
H. Roussille and F. Michel are with Université de Montpellier, LIRMM, CNRS, France. Emails: {hroussille; fmichel}@lirmm.fr

- We rank the identified deviations through their potential impacts and feasibilities.
- We compare existing reviews and surveys, with a focus on taxonomies concerning the security of blockchains.

This paper is organized as follows. Section II introduces the basic concepts used to define the taxonomy. Section III enumerates the deviations and vulnerabilities for each role of AGR4BS, thus defining the taxonomy. Section IV discusses how this taxonomy, paired with the AGR4BS model, can be used to help secure blockchain systems. Section VI concludes this paper and discusses future work perspectives.

II. Basic Concepts / Preliminaries

A. Blockchain Systems Overview

A blockchain system allows its participants to collectively build a distributed economic, social and technological system where participants perform verified transactions without needing to trust each other fully, neither relying on a trusted third party nor having a global view of the system [2]. They do so by looking for peers and connecting with them based on an implementation-dependent selection strategy. More precisely, while some participants use the blockchain as a transactional service, others are incentivized to contribute and provide this service by participating in the consensus mechanism.

We can differentiate between two leading blockchain families: Private and public. In private blockchains, participation and contribution are conditioned on a permission system, where contributors are not necessarily incentivized by system itself but rather by the structure (*i.e.,* company or consortium) owning the blockchain. Public blockchains do not have permission mechanisms. Thus contribution is accessible to anyone and incentivized through financial compensation inside the system itself (*i.e.,* block creation reward). In such systems, participants are indirectly interested in the system's long-term stability to maintain their stakes. Contribution to a blockchain is made through consensus participation. The consensus mechanism revolves around a predefined algorithm where participants consensually agree on the blockchain's state transition. Typically, in a Proof-of-Work (PoW) consensus, participants (*i.e.,* miners) compete in the block creation process through raw computing power. In a Proof-of-Stake (PoS) consensus, participants are deterministically selected to propose a block where increasing their respective stakes (*i.e.,* locked investments) increases their selection probability.

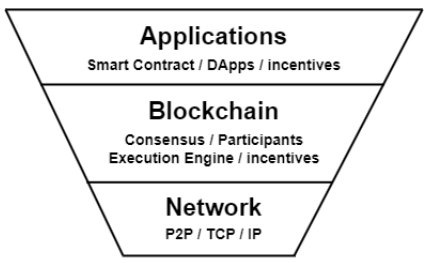

Fig. 1: Layered Architecture of Blockchain Systems

Blockchain systems can be represented as a hierarchy of layers, as presented in Fig. 1. The Network layer is the lowest one, holding communication primitives and low-level protocols required to build a blockchain network. The Blockchain layer manages the blockchain participants, their incentives and the execution environment, and provides the specific protocols and data structures as well as the consensus algorithm. The Applications layer holds the smart contracts, the DApps they form and the incentives related to them. We disagree with works separating the consensus from the incentives. Participating in the consensus leads to rewards or punishments in case of misbehavior; in that sense, they are both parts of the Blockchain Layer. When discussing attacks and vulnerabilities, a structural decomposition leads to cross-layer vulnerabilities; *e.g.,* selfish mining is often portrayed as related to the consensus, incentive and network. A more concise representation is achievable through a role-based approach.

B. The AGR4BS Model

According to the AGR model, Multi-Agent Systems are modeled using an organizational perspective thanks to three core concepts: Agent, Group and Role [3]. Agents are communicating entities playing roles within groups. Groups identify contexts for patterns of activities (*i.e.* roles) shared by several agents. Roles are abstract representations of the functional positions of agents in a group.

In the blockchain context, the AGR4BS model [3] identifies the generic agents, groups and roles, and specifies the attributes and behaviors necessary for each role's generic functionality (Fig. 2). So, a specific combination of these roles defines a logical entity in each concrete blockchain (*e.g.,* a Bitcoin Miner is composed of the Blockchain Maintainer, Block Proposer, Block Endorser and Investor roles). AGR4BS relies on a unified way of modeling blockchain systems through an agent-oriented view, thus providing a solid foundation for incentive vulnerability analysis, thanks to the concrete representation of the participants' roles and behaviors.

C. Incentive Vulnerability, Deviated Behavior and Countermeasure

A vulnerability can be formally defined as a weakness or flaw within a system that can be exploited by an external party to cause harm to the system. In this study, we focus on a specific type of blockchain system vulnerability: *incentive vulnerability*, which we define as a misalignment between (1) the behavior of an agent as expected by the protocol designers and (2) the behavior eventually obtained by following a utility based interpretation of the incentives. This misalignment incentivizes participants to deviate from their nominal behavior (*i.e.,* external fault). In that sense, a behavior is said to deviate when not strictly adhering to the official implementation (*i.e.,* the nominal behavior). If such a deviation harms the system or its participants, one or several countermeasures must be designed and implemented to mitigate the deviation feasibility and/or its impact. Strictly speaking, an incentive vulnerability is the root cause of a deviation.

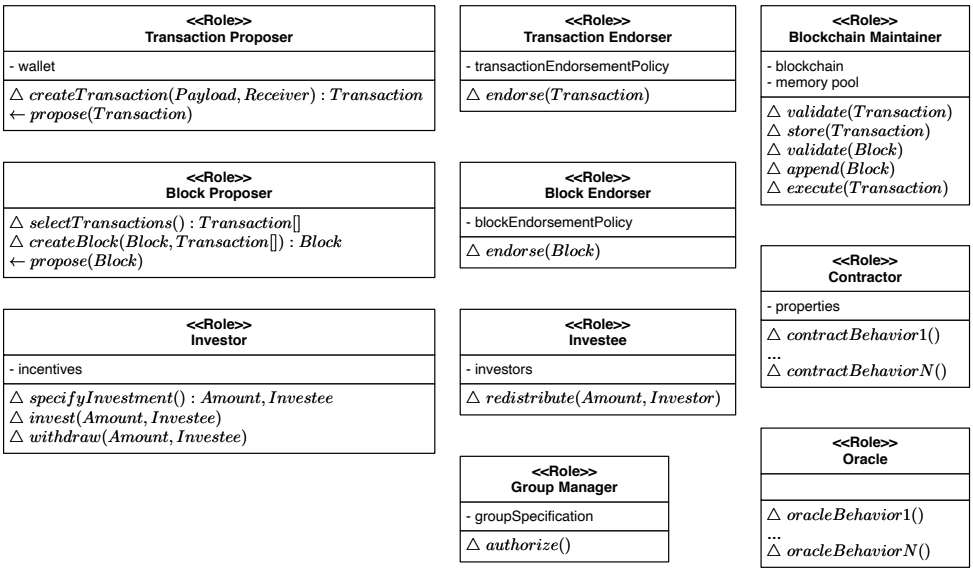

Fig. 2: The roles and their corresponding attributes and behaviors for blockchain systems [3].

D. Taxonomy Characteristics

To classify, categorize and measure vulnerabilities, we use the following concepts: impact family, severity, risk, scale, priority score and system.

Impact Family relates to the expected impact of vulnerability exploitation. Three possibilities are considered: Fairness, Economics and Security. A Fairness impact arises whenever discrimination between agents occurs for any reason that is not part of the protocol. Also, any imbalance between the proportionality of invested resources and the reward is included in this family. An Economic impact happens when the system's economy is disturbed, such as an artificial transaction fee increase. A Security impact relates to the blockchain system's partially or fully compromised core properties, such as Block Finality or Chain Integrity.

Severity defines the level of impact of a successful attack, and takes a value in Very High, High, Medium, Low, and Very Low, which are aliases for 1, $\frac{4}{5}$, $\frac{3}{5}$ $\frac{2}{5}$ and $\frac{1}{5}$ respectively. These levels are not based on a quantifiable notion of severity but are used to categorize vulnerabilities informally and thus help compute their respective priority scores.

'Very Low' implies that an agent or group of agents is mildly impacted but still functioning, with no quantifiable impact on groups or the system. 'Low' also implies that an agent, or a subgroup of agents, is impacted in a more meaningful way, possibly non-functional, while the group and blockchain system they are part of is still functional. A 'Medium' severity level impacts both agents and groups in a way not jeopardizing the system, but implying consequences in at least one of its core properties, such as Fairness, Security or Economics. A 'High' severity level implies a non-negligible impact on the system. Finally, 'Very High' refers to an immediate threat, such as a general unfairness issue or halting of the system.

Risk refers to the feasibility of an attack in terms of resources required to conduct it. The risk levels are similar to the ones defined for severity: 'Very High', 'High', 'Medium', 'Low', and 'Very Low'. Those risk levels are also mapped to values similar to severity levels. 'Very High' signifies that the associated vulnerability is relatively easy to set up as it only requires a few resources. A risk level of 'High' refers to an attack where some amount of resources must be committed to it, but still doable by most participants. A risk level of 'Medium' means that an attack requires a non-trivial amount of resources. Risk levels of 'Low' and 'Very Low' are used to describe attacks requiring overwhelmingly large resources.

An important note regarding the definition of risk and, more specifically, feasibility: the resource required to achieve a specific attack depends on the attack's type. For example, a mining-based attack requires computational power, while a network-related one requires many identities and bandwidth. Our resource definition is, therefore, fluid to accommodate various attack types.

Scale. As blockchains are decentralized, one must differentiate the risk and severity levels over the scale of the actual attack. In this paper, we consider both a low-scale attack and a large-scale one. Depending on the attack type, the scale might be related to the number of attackers (*i.e.,* sybil attack), the total required computing power (*i.e.,* mining attack) or the economic value (*i.e.,* staking attack) required for the attack.

Priority Score ranks the identified vulnerabilities loosely. It is based on severity and risk and is defined as the product of those variables. As we can compute low-scale and large-scale priority scores, we opt for a pessimistic approach and consider the attack to have an overall priority score equal to the maximum priority score across the different scales.

System describes the subset of blockchain systems vulnerable to a specific attack. Some systems may be independent, while others might be linked deeply to the underlying consensus mechanism : PoW (Proof of Work), PoS (Proof of Stake), PoA (Proof of Authority), PBFT (Practical Byzantine Fault Tolerant inspired systems), Explicit Block and/or Transaction Endorsement, All.

In the following, we present each role and its deviations. For each role, we summarize all of its known deviations, their

Role	Deviations Exploiting Incentive Vulnerabilities				Vulnerability Metrics						
	Deviation Name	Deviated Behavior	Impacted Roles	Reference	Impact Family	Low Scale		Large Scale		Priority Score	System
						Severity	Risk	Severity	Risk		
Block Proposer	Censure of Transaction	selectTransactions	Transaction Proposer	[2]	Fairness	●○○○○	●●●●○	●●●○○	●●●○○	0.16	All
	Selective Block Propagation	proposeBlock	Blockchain Maintainer Block Proposer	N/A		●○○○○	●○○○○	●●●○○	●●○○○	0.24	All
	Consensus delay	createBlock proposeBlock	All	N/A		●○○○○	●○○○○	●●●●●	●○○○○	0.80	PBFT
	Selfish / Stubborn Block Creation	createBlock proposeBlock	Blockchain Maintainer Block Proposer	[4]	Fairness Security	●●○○○	●●○○○	●●●●●	●○○○○	0.25	PoW
	Maximal Extractable Value	selectTransaction createBlock	Transaction Proposer	[5]	Fairness Economics	●●○○○	●●●●○	●●○○○	●●●●○	0.32	All
Block Endorser	Censure of Blocks	endorseBlock	Block Proposer Transaction Proposer	N/A	Fairness	●○○○○	●●●●○	●●●●○	●○○○○	0.16	Explicit Endorsement
Transaction Endorser	Censure of Transactions	endorseTransaction	Transaction Proposer	[6]		●○○○○	●●●●○	●●●○○	●○○○○	0.16	Explicit Endorsement
Transaction Proposer	Double Spending	createTransaction	All	[7]	Fairness Economics	●○○○○	●○○○○	●●●●●	●○○○○	0.25	All
	Front Running	createTransaction	Transaction Proposer	[8]		●●○○○	●●●●●	●●●○○	●●●●●	0.60	All
Blockchain Maintainer	Skip Transaction Validation	validateTransaction	None	[9]	Security	●○○○○	●●●○○	●●●●●	●●○○○	0.40	All
	Skip Block Validation	validateBlock	None			●○○○○	●●●○○	●●●●●	●●○○○	0.40	All
	Skip Transaction Execution	validateTransaction executeTransaction	None			●○○○○	●●●○○	●●●●●	●●○○○	0.40	All
	Skip Transaction Diffusion	diffuseTransaction	Blockchain Maintainer	[10]	Fairness	●○○○○	●●●●○	●●●●○	●●●●○	0.64	All
Oracle	Corrupted Oracle	Dedicated Oracle behavior	Contractor Investor Investee	[11]	Economics	●●○○○	●●●●○	●●●●●	●●○○○	0.40	All
Investee	No / Partial Redistribution	redistribute	Investor	N/A	Fairness Economics	●●○○○	●●●●○	●●●○○	●●○○○	0.32	All

TABLE I: The taxonomy of role-based incentive vulnerabilities.
Very Low: ●○○○○ , Low: ●●○○○ , Medium : ●●●○○ , High: ●●●●○ , Very High : ●●●●●

impact families, severities and risks in low and large scales, and their calculated priority scores (summarized in Table I). Each subsection starts with a nominal behavior definition of a role, followed by possible deviations.

III. Role-based Taxonomy

Here we present the role-based taxonomy of vulnerabilities (see Table I) that provides a classification of violable constraints and assumptions that are bound to the roles.

A. Block Proposer

Nominal behavior. Block Proposer selects a subset of the most relevant transactions, orders them, and tries to create a valid block, always extending the main chain according to the consensus protocol of the system and, if it succeeds, immediately proposes it to its neighbors.

Censure Transaction. Through a deviation of the *selectTransactions* behavior, a Block Proposer may censure some transactions and therefore impacts Fairness. This is the case when a Block Proposer purposely excludes from its selection mechanism specific transactions coming from Transaction Proposer, even though they are financially attractive. This is an identity/address-based censure whose purpose is to delay or even forbid transactions involving a specific sender or receiver. While several blacklisted addresses are already purposely excluded from the network, the same behavior applied to non-criminal addresses is an illegitimate censure. For this deviation to be impactful, a majority of block proposer must be willing to enforce the censure due to the complexity associated with having an overwhelming majority in blockchain systems. While significantly delayed, the agents or groups targeted by such censure can still rely on the remaining nominal participants or become a Block Proposer. However, a single block proposer may choose to censure any other participant; this requires few resources and has little to no impact.

Selective Block Propagation. The block proposal to the network might be intentionally skewed through a deviation of the *proposeBlock* behavior. For example, suppose an agent wishes to delay a competing Block Proposers and Blockchain Maintainer. In that case, it might propose its new block to all its peers except that competing one, thus slightly delaying its competitor's knowledge update. On a large scale, the targeted agent(s) may have a significant delay with the rest of the network, thus lowering their potential for valid block creation.

Consensus Delay. Consensus delay, or halting, is mainly related to PBFT consensus-inspired blockchain systems, where block proposers either propose conflicting blocks or do not propose through a combination of deviations from the *createBlock* and *proposeBlock* behaviors. A consensus-level attack impacts every participant. In such a configuration, consensus participants, often called *validators*, may collude to reach the 33% threshold of malicious nodes in the committee.

Selfish / Stubborn Block Creation A Block Proposer might not mine on the head of the public main-chain but rather on a private adversarial fork. This is done with another combination of deviations from the *createBlock* and *proposeBlock* behaviors. Other Blockchain Maintainers and Block Proposers are the primary victims of such a deviation. Such a deviation is mainly linked to Proof-of-Work (PoW) blockchains and has been studied extensively [4].

Maximal Extractable Value Another vulnerability targeting the economics of public blockchains is the possibility of

reordering transactions for the highest financial gain[1] [5]. This optimization results from a deviation in *selectTransaction* and *createBlock* directly impacting the Transaction Proposers. While it is rational for a miner to do so, Miner / Maximal Extractable Value (MEV) takes advantage of front-runners mostly looking for profitable arbitrage opportunities. This dynamic eventually raises the blockchain fees and reduces accessibility. While MEV and front-runners serve Decentralized Finance (DeFi) economic equilibrium, they hamper the overall economy. They may create blocks with a such attractive rewards that other miners might be incentivized to attempt to create a fork and capture the reward for themselves. Additionally, the impact on fairness is evident as the order of transactions is purposely modified. Maximizing the reward from a block creation is rational individually or in a group, such as a mining pool. Because of this, the system becomes less accessible due to higher fees but it is still usable.

B. Block Endorser

Nominal behavior. Block Endorser vouches for blocks to be included in the chain following a block endorsement policy.

Censure Block. Block Proposers might purposely refuse to endorse blocks with specific characteristics through a deviation from the *endorseBlock* nominal behavior, directly impacting the Block Proposer who created the block and, indirectly, the Transaction Proposers whose transactions are included in it. Depending on the endorsement policy, such actions prevent the block from proceeding into the blockchain for non-consensual reasons and therefore impact the Fairness of the system. If such agents were to misbehave, they could impact or even stop the production of blocks. This censure is relatively easy to set up for an individual agent. However, it has little to no impact as Block Proposers can and should always submit their proposals to several endorsers. Conducting this attack at a large scale requires most of the Block Endorsers to deviate from the nominal behavior, which is unlikely to happen thanks to the decentralized nature of the system.

C. Transaction Endorser

Nominal behavior. Transaction Endorser vouches for the inclusion of a transaction following an endorsement policy.

Censure Transaction. Similarly to the *Endorse Block* behavior, *Endorse Transaction* is subject to a malicious deviation from the *endorseTransaction* behavior, leading to censorship of one or several Transactions Proposers. Any endorser could refuse to endorse specific transactions. Such actions could forbid the transaction to proceed any further in explicit endorsement schemes, such as in the Hyperledger Fabric blockchain[2]However, at a larger scale, its impact is more severe as it is possible to lock participants out of the system. Still, this requires a majority of endorsers to deviate from the nominal behavior, reducing the risk.

[1]Quantifying Blockchain Extractable Value: How dark is the forest? - https://arxiv.org/abs/2101.05511 last accessed on : 10-28-2022

[2]Hyperledger Fabric, https://www.hyperledger.org/use/fabric, accessed on 09/12/2022.

D. Transaction Proposer

Nominal behavior. Transaction Proposer creates a valid transaction with a payload and the right fee for its inclusion in a block and then proposes it to the system.

Double Spending. In the context of a blockchain that allows forking in its protocol, an agent might deviate from the nominal *createTransaction* behavior for a double-spending attempt [7], usually paired with a form of forking attack such as selfish mining by knowingly proposing two conflicting transactions on different candidate chains. Such a deviation, if successful, impacts every participant of the blockchain system.

Front Running. As transactions are public and broadcast through the network before their inclusion in a block, every participant is aware of future events before they occur. For example, this allows a front-runner to take advantage of incoming large buy/sell orders on decentralized exchanges to front-run [8] such transaction through another deviation of the *createTransaction* behavior, impacting other Transaction Proposers. Front-runners can get priority through the fee mechanism that is the primary selection criteria of Block Creator when selecting which transactions to include in a potential new block. Note that front running is not a deviation. This behavior is rational and allowed by the protocol, but it is obviously harmful and can be therefore considered as an incentive vulnerability. Front running is deeply linked to MEV as any front-runner is theoretically willing to give up to 99.99% of its profit as a fee to the Block Creator, thus increasing its power and influence in public blockchain Systems. However, the impacts vary depending on the scale of the attack (*i.e.,* the number of participants involved in front-running transactions, looking for opportunities). While a few front-runners may only have a mild impact on the overall system, when this strategy is widely adopted, there are consequences for the front-run users and the global economy as it fuels artificial fee growth. Front running is a serious issue regarding both the Economics and Fairness of blockchain systems.

E. Blockchain Maintainer

Nominal behavior. Blockchain Maintainer validates all newly received blocks and transactions. Valid transactions are stored in the memory pool, valid blocks are appended to the local blockchain, and all its transactions are executed.

Skip Transaction Validation. As transaction validation is not rewarded, rational agents may be incentivized to skip it by deviation from the *validateTransaction* behavior, potentially sacrificing the overall security and correctness of the system to gain an advantage in both time and computing resources. Such a deviation has no impact on other participants as long as it is local. However, if it were widespread, all participants would be impacted as the ledger coherency is no longer ensured. This vulnerability is known as the Verifier's Dilemma [9]. The potential inclusion of invalid data into the blockchain is hazardous as it threatens the system's stability. As stated in Section II-A, participants that maintain the blockchain are interested in maintaining the system's stability but also in keeping the unrewarded amount of work to a minimum.

Skip Transaction Execution. In the nominal case, when an agent receives a transaction linked to a smart contract invocation, it should execute them using this behavior. The agent might not know in advance if the contract contains faulty logic, such as a hack of the execution environment to produce a potentially harmful result or simply invalid actions. The transaction execution time is also unknown and may be costly for the agent validating it. As smart contracts execution are linked to events and transactions, the primary vulnerability of this behavior is similar to that of the *validate(Transaction)* where an agent would skip the execution by deviating from both the *validateTransaction* and *executeTransaction* behaviors.

Skip Block Validation. Validating a block may be costly for a Blockchain Maintainer. So, to gain a slight advantage, it may simply skip this step and append/propagate invalid blocks by deviation from the *validateBlock* behavior. Validating a block implies validating its structure and the embedded transactions, which eventually requires executing them.

Skip Transaction Diffusion. When transaction diffusion is not incentivized, rational agents may be skewed toward selfish behavior. This involves not sharing a new transaction with its peers by deviating from the *diffuseTransaction* nominal behavior. This deviation may even be profitable for *Block Creators* in open PoW systems as it reduces competition on the memory pool.

Current blockchain systems do not explicitly reward transaction diffusion. Instead, they implicitly rely on the stake that contributors (*i.e.,* Block Proposers and Blockchain Maintainers) have in the system. No transaction diffusion would hamper the system's usability by its users and possibly lead to centralization. However, for the reasons mentioned above, such an attack is improbable as it is against the interest of every rational contributor.

F. Oracle

Nominal behavior. The *Oracle* role holds the behavior collection responsible for Oracle functionalities, that is, bridging outside information to the blockchain.

Corrupted Oracle. An oracle node might be corrupted and transmit erroneous data on purpose by deviating from one of its dedicated behaviors or simply due to faulty logic. This would lead the blockchain system to make decisions based on incorrect information. Additionally, the data source might be corrupted while Oracle is working nominally. Both cases are nearly indistinguishable from one another and can lead to serious consequences. Trusting external oracle data is known as the Oracle problem. It poses a paradox between the necessity of oracles for real-world usage of the blockchain and the trustless nature of blockchain systems as described in [11].

G. Investee

Nominal behavior. Investee receives investments from investors, provides a service, and redistributes the rewards to investors proportionally to their respective contributions.

No, Partial Redistribution. If an investee does not properly redistribute wealth earned thanks to its investors because of a deviation from the *redistribute* behavior, it may gain a financial advantage. However, this would come at the cost of a loss of reputation in the open blockchain system and hurt both the Fairness and Economics of the system due to its impact on Investors, Contractors, and other Investees. Such a behavior could be easily monitored and blacklisted. This has already been observed in the Tezos blockchain[3].

IV. Discussion

The taxonomy presented in this paper, alongside its base model, AGR4BS [3], can be used for systematic incentive-level security assessments in blockchain systems. First, we must create an AGR4BS model of the system to achieve the role granularity required for the taxonomy. Most existing systems could expand one of the already defined AGR4BS modelizations. Still, some specific blockchains may require defining a new model with the needed roles and behaviors.

Many blockchain systems share standard functionalities and logic, which are abstracted through roles, meaning that a given role may be subject to the same vulnerability across various systems. An incentive security assessment would follow a top-down approach where known vulnerabilities are tested first in the context of that specific system. Then, both the model and the existing taxonomy can serve to identify critical roles (*i.e.,* often Block Proposer and Blockchain Maintainer).

Given the scale and complexity of blockchain systems and their participants' autonomy, the approach best suited to incentivize vulnerability exploration and discovery is Multi-Agent Reinforcement Learning (MARL). This approach allows for the study of participants with rational objectives (*i.e.,* profit) or non-rational ones (*i.e.,* impact). By defining the proposed taxonomy, we aim to provide a systematic framework for deriving multi-agent strategies to assess blockchain system security and automated attack discovery. Therefore, the AGR4BS model suits blockchains and MARL well since it uses role abstraction, which facilitates modeling MARL agents for strategy search.

MARL can be applied to ensure a secure update process if the incentive mechanism undergoes modifications. Additionally, the multi-agent interactions could be represented and learned to discover realistic behavior shedding light on previously unknown vulnerabilities, which could then be studied using more interpretable methods.

V. Related Works

There are exhaustive reviews and surveys reported in the literature [1], [12]–[15] (see Table II for a comparison). Saad et al. [1] define an attack taxonomy over the following three main categories: Structure attacks, Peer-to-Peer attacks and Application attacks. They list the known attacks, and discuss the existing or potential countermeasures. Hameed et al. [12] define several taxonomies with a strong focus on industrial application of blockchain systems. Those taxonomies relate to design, security, privacy requirements, and security. They expose several attacks on a per-layer basis, with known or proposed countermeasures. Sayeed et al. [13] propose a study

[3]Tezos, https://tezos.com/, last accessed on 10-12-2022

Caracteristics \ References	Saad et al. [1]	Hameed et al. [12]	Sayeed et al. [13]	Alkhalifah et al. [14]	Li et al. [15]	Ours
Layer of interest	- Application - Blockchain - Network	- Application - Blockchain - Network	- Application	- Application - Blockchain - Network	- Application - Blockchain - Network	- Blockchain
Proposes Countermeasures	Yes	Yes	Yes	Yes	Yes	Yes
Classification	Layer Based	Layer Based	Attack Type	Layer Based	Risk & Vulnerability	Role-Based
Incentives Focused	No	No	No	No	No	Yes

TABLE II: Comparison of studies with a focus on taxonomies concerning the security of blockchains.

focused on the Ethereum smart contracts / application layer. They implicitly provide a taxonomy through a categorization of the main types of attacks and discuss existing tools and techniques enabling some level of protection. Alkhalifah et al. [14] define a taxonomy of blockchain threats and vulnerabilities over the following categories : Client's Vulnerabilities, Consensus Mechanisms Vulnerabilities, Mining Pool Vulnerabilities, Network Vulnerabilities and Smart Contract Vulnerabilities (Ethereum and EVM focuses). Li et al. [15] survey the security of blockchain systems and propose a succinct taxonomy of blockchain risks covering encryption, consensus and transactions. They also propose a taxonomy of Ethereum's smart contracts vulnerabilities.

Exhaustive studies focus on the "How" and "Where" of an attack, defining how the attack is impacting the system as well as in which layer it is taking place. In this perspective, countermeasures are often restrained to only treat the problem's consequences (*i.e.,* detection systems, increased resilience). Precise studies almost only focus on the "Why", explaining the reasons and incentives motivating the attack. The proposed countermeasures modify the system so that the attack is no longer incentivized, treating the root cause of the problem. Our taxonomy aims to merge both approaches, covering most deviations with a focus on incentives, and a role-based classification for a natural use with reinforcement learning as shown in Table II.

VI. Conclusion

We introduced a taxonomy of blockchains incentive vulnerabilities for networked intelligent systems. This can help researchers and developers better understand the different types of blockchains available and make informed decisions when designing these systems. The presented taxonomy is based on a dedicated generic multi-agent organizational model [3] and calculates the priority scores for each incentive vulnerability. The taxonomy characterizes vulnerabilities as role deviations concerning nominal behavior and incentives. The taxonomy then lists and ranks several known vulnerabilities but provides a way to quantify and classify newly found ones. This taxonomy provides the foundation for characterizing incentive vulnerabilities and supports a role-based classification scheme. We suggest researchers to start studying vulnerabilities with the highest priority scores (Table I), such as *Consensus Delay* and *Skip Transaction Diffusion* linked to the *Block Proposer* and *Blockchain Maintainer* roles, respectively.

References

[1] M. Saad, J. Spaulding, L. Njilla, C. Kamhoua, S. Shetty, D. H. Nyang, and D. Mohaisen, "Exploring the Attack Surface of Blockchain: A Comprehensive Survey," *IEEE Communications Surveys and Tutorials*, vol. 22, no. 3, pp. 1977–2008, 2020.

[2] Ö. Gürcan, A. Del Pozzo, and S. Tucci-Piergiovanni, "On the bitcoin limitations to deliver fairness to users," in *On the Move to Meaningful Internet Systems. OTM 2017 Conferences*, H. Panetto, C. Debruyne, W. Gaaloul, M. Papazoglou, A. Paschke, C.-A. Ardagna, and R. Meersman, Eds. Cham: Springer International Publishing, 2017, pp. 589–606.

[3] H. Roussille, O. Gürcan, and F. Michel, "Agr4bs: A generic multi-agent organizational model for blockchain systems," *Big Data and Cognitive Computing*, vol. 6, no. 1, p. 41p, 2022. [Online]. Available: https://www.mdpi.com/2504-2289/6/1/1

[4] I. Eyal and E. G. Sirer, "Majority is not enough: Bitcoin mining is vulnerable," *Commun. ACM*, vol. 61, no. 7, p. 95–102, jun 2018. [Online]. Available: https://doi.org/10.1145/3212998

[5] P. Daian, S. Goldfeder, T. Kell, Y. Li, X. Zhao, I. Bentov, L. Breidenbach, and A. Juels, "Flash boys 2.0: Frontrunning in decentralized exchanges, miner extractable value, and consensus instability," in *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2020, pp. 910–927.

[6] P.-Y. Piriou, O. Boudeville, G. Deleuze, S. Tucci-Piergiovanni, and O. Gürcan, "Justifying the dependability and security of business-critical blockchain-based applications," in *2021 Third Inter. Conf. on Blockchain Computing and Applications (BCCA)*. IEEE, 2021, pp. 97–104.

[7] U. W. Chohan, "The Double Spending Problem and Cryptocurrencies," *SSRN Electronic Journal*, vol. n/a, no. n/a, p. 11p, 2018.

[8] S. Eskandari, S. Moosavi, and J. Clark, "Sok: Transparent dishonesty: Front-running attacks on blockchain," in *Financial Cryptography and Data Security*, A. Bracciali, J. Clark, F. Pintore, P. B. Rønne, and M. Sala, Eds. Cham: Springer Inter. Publishing, 2020, pp. 170–189.

[9] L. Luu, J. Teutsch, R. Kulkarni, and P. Saxena, "Demystifying incentives in the consensus computer," *Proc. of the ACM Conference on Computer and Communications Security*, vol. 2015-Octob, pp. 706–719, 2015.

[10] O. Ersoy, Z. Ren, Z. Erkin, and R. L. Lagendijk, "Transaction propagation on permissionless blockchains: Incentive and routing mechanisms," in *2018 Crypto Valley Conference on Blockchain Technology (CVCBT)*. IEEE, 2018, pp. 20–30.

[11] G. Caldarelli, "Understanding the blockchain oracle problem: A call for action," *Information (Switzerland)*, vol. 11, no. 11, pp. 1–19, 2020.

[12] K. Hameed, M. Barika, S. Garg, M. B. Amin, and B. Kang, "A taxonomy study on securing blockchain-based industrial applications: An overview, application perspectives, requirements, attacks, countermeasures, and open issues," *J Ind Inf Integr*, vol. 26, p. 100312, 2022.

[13] S. Sayeed, H. Marco-Gisbert, and T. Caira, "Smart Contract: Attacks and Protections," *IEEE Access*, vol. 8, pp. 24 416–24 427, 2020.

[14] A. Alkhalifah, A. Ng, A. Kayes, J. Chowdhury, M. Alazab, and P. Watters, *A Taxonomy of Blockchain Threats and Vulnerabilities*, 1st ed. United States: CRC Press, Aug. 2020, pp. 3–25.

[15] X. Li, P. Jiang, T. Chen, X. Luo, and Q. Wen, "A survey on the security of blockchain systems," *Future Generation Computer Systems*, vol. 107, pp. 841–853, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X17318332

Hector Roussille is currently a Ph.D candidate at Montpellier University, France in the CEA LIST, Paris-Saclay University, France since November 2020. Contact him at hector.roussille@cea.fr.

Önder Gürcan is an expert research engineer at CEA LIST, Paris-Saclay University, France. Contact him at onder.gurcan@gmail.com

Fabien Michel is an Associate Professor (HDR) at the Multi-Agent Systems research team (SMILE) at the LIRMM (Montpellier, France). Contact him at fmichel@lirmm.fr.