Article

Standalone Behaviour-Based Attack Detection Techniques for Distributed Software Systems via Blockchain

Hosam Aljihani 1,2,* , Fathy Eassa 2, Khalid Almarhabi 3, Abdullah Algarni 2 and Abdulaziz Attaallah 2

1 Department of Computer Science, College of Computer Science and Engineering, Taibah University, Madinah 42353, Saudi Arabia
2 Department of Computer Science, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia; feassa@kau.edu.sa (F.E.); amsalgarni@kau.edu.sa (A.A.); aattallah@kau.edu.sa (A.A.)
3 Department of Computer Science, College of Computing at Alqunfudah, Umm Al-Qura University, Makkah 21514, Saudi Arabia; kamarhabi@uqu.edu.sa

* Correspondence: hjihani@taibahu.edu.sa or haljihani0006@stu.kau.edu.sa

Abstract: With the rapid increase of cyberattacks that presently affect distributed software systems, cyberattacks and their consequences have become critical issues and have attracted the interest of research communities and companies to address them. Therefore, developing and improving attack detection techniques are prominent methods to defend against cyberattacks. One of the promising attack detection methods is behaviour-based attack detection methods. Practically, attack detection techniques are widely applied in distributed software systems that utilise network environments. However, there are some other challenges facing attack detection techniques, such as the immutability and reliability of the detection systems. These challenges can be overcome with promising technologies such as blockchain. Blockchain offers a concrete solution for ensuring data integrity against unauthorised modification. Hence, it improves the immutability for detection systems’ data and thus the reliability for the target systems. In this paper, we propose a design for standalone behaviour-based attack detection techniques that utilise blockchain’s functionalities to overcome the above-mentioned challenges. Additionally, we provide a validation experiment to prove our proposal in term of achieving its objectives. We argue that our proposal introduces a novel approach to develop and improve behaviour-based attack detection techniques to become more reliable for distributed software systems.

Keywords: attack detection techniques; blockchain; cyber-attack; distributed software systems; specification-based attack detection

1. Introduction

Distributed systems have become one of the cornerstones of the contemporary computing model due to their compatibility with accelerated technology evolution in both hardware and software capabilities and functionalities. This compatibility is manifested in the ability to integrate independent computing resources and make them appear as a single coherent system [1]. As a reflection of the nature of computing resources, distributed systems generally operate at two levels, the hardware level and the software level. While the responsibility for exploiting multi-hardware resources falls on the hardware level, the responsibility for making the software work in a distributed, integral, extendable and maintainable fashion, falls on the software level [2]. Hence, software systems that run in a distributed environment, for whatever purpose, can be called distributed software systems. Distributed software systems which are derived from the features of distributed systems, bring many clear benefits for the users who adopt them, such as increased performance, flexibility and reliability [3]. However, as a result of the natural complexity of distributed software systems, especially in terms of integration, they still face many outstanding chal-
Challenges and threats, such as their required transparency and the consequent difficulty of management and their potential heterogeneity [4].

In recent decades, it has become obvious that different kinds of cyberattacks using various methods constitute a real and rapidly increasing threat to distributed software systems. According to [5], annual damage from cybercrime amounting to a loss of USD 10.5 trillion is expected by 2025. In addition, a study by [6] indicated that an attack occurs every 39 s, which means that approximately 2244 attacks occur on average each day. Application vulnerabilities, insider attacks, and malware are still considered the most common forms of cyber-attack [7], and therefore pose a direct threat to distributed software systems. Clearly, there is a need to deal with cyberattacks effectively to mitigate their consequences. Based on [8], the cybersecurity framework should consist of five phases. If an attack has potentially occurred, the phase responsible for the initial attempt to counter the attack is the detection phase; therefore, more consideration for this phase is deserved.

Attack detection systems have many features and forms of classification. Initially, detection systems have three main functions, detecting the attacks, recording the detected attacks, and generating alerts [9]. Furthermore, detection systems aim to achieve three main security criteria, confidentiality, integrity, and availability [10]. In addition, detection systems employ two main analysis methods, data mining and pattern-matching techniques [11]. The data mining analysis method focuses on adopting artificial intelligence techniques to perform the detection, while the pattern-matching analysis method focuses on creating patterns for either the attack or the target systems to perform the detection. Attack detection systems also perform in two styles: standalone style, where the detection system performs its tasks without the need to directly interact with other detection systems’ participants; or collaborative style, where groups of detection systems’ participants interact with each other directly in order to improve their detection experiences [12]. Based on the target domain, attack detection techniques can be either Network-based systems or Host-based systems [9]. In Network-based systems, the detection will act on the network level, including packages and communication control, while in Host-based systems, the detection will be at the host level, including processes and software activities.

Importantly, attack detection systems have three approaches [13]. The first approach is the signature-based or misuse-based approach, where the main idea is to recognise the patterns of well-known attacks and then monitor the target system’s activities to detect such attacks based on their patterns. This approach has high performance but is inefficient at detecting zero-day attacks or any unknown attack strategies. The second approach is the anomaly-based or behaviour-based approach, where the main idea is to create target system profiles representing supposed “right behaviours”, where behaviour means a set of related events, with each event representing an interactive action that occurs in the system either internally or externally, and the ordered events collectively represent the system behaviour, then monitor the target system’s behaviour to detect attacks based on anomalies that may occur in conflict with the profiles’ details. This approach is more effective than the previous approach but assumes that the source codes for the target system will not be accessible when creating the profile. Therefore, profiles will be created based on several techniques, including some AI techniques, but these gained profiles do not provide all behaviours of the target system, including their correct but unusual behaviours. Consequently, high false-positive detection occurs by considering “correct but unusual” behaviour as an attack, which weakens the reliability of this approach. The third approach is the specification-based approach, which is considered in some studies as a sub-approach of the behaviour-based approach [14]. The main idea of this approach is to operate analysis techniques on the source codes of the target system in order to extract all trusted behaviours for the target system. Then, during the runtime, just as in the previous approach, the technique will monitor the target system’s behaviour to detect the attacks based on any anomalies that may occur in conflict with the extracted behaviours. The main distinction between the second and third approaches is the reliability aspect, as the last approach should achieve high reliability due to the absence of false-positive detection,
i.e., all truth behaviours are provided, so no false-positive detections are expected this not like the second approach [12]. Table 1 shows the difference between the three detection approaches. These detection systems, especially standalone-style systems where no direct communication between the detection systems takes place, are commonly used in various cybersecurity products.

Table 1. The difference between detection approaches.

| Detection Approach          | Detected Attacks’ Behaviours | False-Positive Detected Attacks’ Behaviours | Undetected Attacks’ Behaviours |
|----------------------------|------------------------------|-------------------------------------------|-------------------------------|
| Signature-based approach    | Well-known attacks’ behaviours | None                                      | Unknown/zero-day attacks’ behaviours |
| Anomaly-based approach      | Abnormal target system behaviours | Normal, but unusual target system behaviours | None |
| Specification-based approach| Any abnormal target system behaviours | None                                      | None |

Our solution, as explained in the evaluation section with different scenarios, will detect any attacks to software in a distributed system that affects behaviour. These attacks can occur on any part of the distributed system software and not limited to the blockchain itself, given that the use of blockchain in the proposal architecture was based on the immutability feature provided in this technology. Examples of attacks that change normal behaviour include malicious code attacks and Denial-of-Service (DoS) attacks. However, some attacks do not affect the behaviour of the software directly; therefore, they remain undetected, such as shoulder surfing attacks and social engineering attacks.

However, behaviour-based attack detection techniques are facing challenges that affect their role to preserve distributed software systems against attacks. One of these challenges is ensuring the immutability of the data used for detecting the attacks. This data including both of the predefined trusted behaviours and the observed behaviours during runtime, are supposed to be unchangeable. In fact, such behaviour-based attack detection techniques have strict mechanisms in place that require high accuracy for any data and/or behaviours dealt with. Without these mechanisms the results obtained by these detection techniques will be wrong and thus unreliable. However, the potential vulnerability of stored and exchanged data in distributed systems, especially in terms of integrity, is that storage and extraction behaviours can alter during runtime. Hence, any direct or indirect effects on the data used for detecting the attacks will affect the data’s immutability and the whole detection systems’ results and thus its trustworthiness.

In addition, ensuring the reliability of detection systems is another challenge facing this kind of detection technique [15]. The reliability challenge comes from the fact that there is a need to ensure that the detection results provided by the attack detection techniques are correct and accurate. If the target system is a distributed system, there is a possibility that the results may be subjected to change if they have not been stored and transmitted securely. At the same time, the detection results have to be easily accessible without any access restrictions caused by security requirements. Hence, achieving the reliability for attack detection techniques that work for distributed systems is essential and must be achieved by balancing data security and accessibility. Therefore, and as a motivation for this research, more investigations to address these challenges are required.

From another perspective, blockchain has emerged as a cryptographic distributed data structure named “blocks” that can be shared and replicated among the independent network participants in a reliable manner [16]. Accordingly, blockchain has several characteristics compared to other related technologies providing similar services, such as a distributed database. For instance, blockchain is a distributed peer-to-peer data structure and network with no trusted third parties between its participants. Thus, the concept of trusted third parties to serve as intermediaries in the blockchain is meaningless and
unnecessary [17]. In addition, blockchain is a shared, decentralised, and replicated ledger, where this ledger is readable, but it is append-only and cannot be removed or updated [18]. Nevertheless, blockchain can be considered an application layer on top of many other forms of networks to easily interact with other Internet services and technologies [19]. In Table 2 below, the most important differences between blockchain and distributed databases are summarised to demonstrate the significance of blockchain technology [20,21].

Table 2. Blockchain vs. distributed database [20,21].

| Feature            | Distributed Database                  | Public Blockchain                  | Private Blockchain                   |
|--------------------|---------------------------------------|------------------------------------|--------------------------------------|
| Data control       | Many control features are centralised  | Decentralised                      | Hybrid with few features centralised  |
| Architecture       | Client-server architecture             | Public peer-to-peer architecture    | Closed peer-to-peer architecture     |
| Performance        | Extremely fast                        | Slow                               | Fast to medium                       |
| Data persistence   | non-persistence                       | Immutable                          | Immutable                            |
| Access type        | Permissioned                          | Permissionless                     | Permissioned                         |
| Chance of failure  | Hard but possible                     | High, a trusted third party does not exist, any participant can verify the operations independently | High, a trusted third party does not exist, any participant can verify the operations independently |
| Reliability        | Low, a trusted third party must exist, cannot verify the operations independently | High, a trusted third party does not exist, any participant can verify the operations independently | High, a trusted third party does not exist, any participant can verify the operations independently |
| Transparency       | No, under administrators' decisions   | Achieved                           | No, under administrators' decisions  |
| Integrity          | Vulnerable to data-change attacks     | Supported                          | Supported                            |

From another perspective, blockchain networks can generally be categorised into two types: (1) Public-permissionless blockchain is an open network where any willing participant can join this network and view, read, and write blockchain. The availability of stored data and the full decentralisation from any form of authority are the main features of the public blockchain. However, confidentiality and privacy are not provided for public blockchain data. (2) A private-permissioned blockchain is a private network that allows predefined entries of verified participants, which private businesses prefer. While private blockchain does not achieve full decentralisation, it still achieve the blockchain principles for stored data in a distributed, private, and secure manner [22].

In fact, blockchain relies principally on consensus algorithms which are considered the backbone of the blockchain. Consensus algorithms are “protocol sets that provide a technique with the help of which the users or machines can coordinate in a distributed and decentralised setting” [23]. In the absence of the central governing authority in blockchain, consensus algorithms’ main role is to ensure the collective agreement by the majority of the blockchain entities on the blockchain about any action attempted to occur. Hence, consensus algorithms aim to achieve reliability, integrity, and security of the blockchain [22].

Practically, every blockchain has various application scenarios. Thus, adopted consensus algorithms by blockchain need to be compatible with the requirements for each blockchain application. Hence, there are many forms of consensus that help to fit these requirements. The proof-of-work (PoW), the proof-of-stake (PoS), and the proof-of-authority (PoA) are the most popular consensus algorithms [22,23].

The PoW consensus algorithm involves a set of mechanisms that cause a considerable amount of effort when performing most of the network processing, such as mining new blocks. The purpose is to protect the blockchain against computing power-based attacks such as DoS attacks. More computing power nodes are often more likely to perform the mining and similar operations on the blockchain network, thus achieving the rewards [23].
The PoS consensus algorithm applies a set of pseudorandom-based processes that determine the validator node for the subsequent blocks. The purpose is to distribute network tasks fairly among the network’s participants since the reward for the network process is not limited to the most computational power participants but instead is a right for everyone. Hence, PoS could provide a secure network against 51% of attacks given the fair distribution of tasks in the network [22].

The PoA consensus algorithm grants authority for selected nodes in validating the transactions and other network tasks. As such, authorised nodes are considered as trusted entities for this blockchain and responsible for their security. The other nodes benefit from the network services and play a prominent role in monitoring the blockchain registry of their transactions and verifying the authority of their executors. Consequently, the transactions and other network tasks are performed significantly faster, which positively reflects on the network throughput. However, the concept of full decentralisation will be affected in this consensus algorithm as long as the majority of nodes do not have full authority, similar to the other consensus algorithms. However, there is a cost of obtaining high throughput and scalability. The PoA is secure against DoS and attacks and 51% of other attacks to some extent [4]. Table 3 presents the main comparisons among the above-mentioned consensus algorithms.

| Consensus Algorithm        | Advantages                                                                 | Disadvantages                                                                 | Security                                      |
|----------------------------|---------------------------------------------------------------------------|-------------------------------------------------------------------------------|-----------------------------------------------|
| Proof-of-work (PoW)        | Reaches consensus quickly, rules out the possibility of spamming and is most tested over time. | More power is consumed and wasted in the mining process. Hardware dependency can lead to mining centralisation. | Open to 51% of attacks along with selfish mining and eclipse attack. |
| Proof-of-stake (PoS)       | It is advantageous over other protocols given reduced energy consumption, easy staking, and being environmentally friendly. | One basic disadvantage is that large stakeholders dominate the network.       | Despite being secure against 51% of attacks, it is vulnerable to nothing-at-stake, grinding, long-range attacks. |
| Proof-of-authority (PoA)   | PoA increases the speed of transaction validation and functions as a platform for the development and maintenance of Dapps. | Full decentralisation is not considered.                                       | Secure against DoS attacks and 51% of other attacks to some extent. |

Realistically, every distributed software system looking to leverage the blockchain must determine the blockchain type based on the nature and validity of the users’ access to the network. Additionally, based on the software scenario and requirements, the appropriate consensus algorithm should be determined. The appropriate blockchain network and consensus algorithm by each software are determined in light of the characteristics of each type, as mentioned above. For instance, if a distributed software system has been designed for a public business such as cryptocurrency, and is concerned about data immutability and availability and is not concerned about data privacy, where the throughput is not the ultimate goal for this system, thus a public blockchain that utilises PoW or PoS is a preferred choice for this software system.

Blockchain offers promising solutions for attack detection techniques to overcome their challenges. In terms of attack detection techniques and their immutability challenge, blockchain provides a solid solution through applying the immutable mechanisms of blocks mining and appending to the blockchain network, which can be adopted as a mechanism to store behavioural-based attack detection techniques’ data. Both trusted behaviours and the runtime behaviours detection techniques’ input data have a set of linked events that in total represent the whole behaviour. Once an event is generated, it can be mined in the same fashion as blocks mining, where each event depends on the previous event hash.
value as required data to mine the new event. Any attempt to make an unobserved change in an event block will require changing all previous blocks in the chain. Such an action would be costly computationally and easily noticeable by the other blockchain properties. Consequently, all of the behaviour’s events will be stored as linked blocks on a blockchain and once stored in this manner, any attempt to perform unobserved changes would be impossible. Therefore, the use of blockchain mining and storage mechanisms results in an increase of immutability for these detection systems.

In terms of the reliability challenges faced by attack detection techniques, blockchain can provide a solution through applying the distributed ledger recording mechanisms consensus protocol, to be adopted as a mechanism for the detection results announcement. In the blockchain, consensus protocols such as proof-of-work (PoW) and proof-of-stake (PoS) play a significant role in recording any mined block and/or any other transaction on the distributed ledgers in a non-repudiated manner, thus making the distributed ledgers more reliable. Applying consensus protocols such as these to record any detected attack will increase the detection techniques’ reliability at the level of exchanged data. In addition, abandoning the use of a trusted third party should increase the reliability at the level of participant communication. Therefore, it could be argued that blockchain can provide promising solutions for the challenges of immutability and reliability in attack detection techniques. However, it must be noted that the implementation of such mechanisms and protocols is not in itself a direct solution. There are many challenges facing the integration of blockchain with attack detection techniques, such as:

- The right analysis of the software’s behaviour and correct arrangement of its events.
- Determining the most appropriate data structure for the event to be properly stored as a block.
- Determining the type of blockchain network best suited to attack detection techniques, either public or private.
- Determining the most appropriate consensus protocol for these techniques.

This research seeks to explore these issues and to make the idea of integration possible. In this paper, and based on the above-mentioned challenges and opportunities, we aim to make the following contributions:

- To investigate the advantages of blockchain when it has been integrated with improved behaviour-based attack detection techniques.
- To clarify to what extent we can address the current challenges of detection techniques in terms of data immutability and system reliability.
- To address the detection techniques and blockchain integration challenges.
- To design and develop our proposal to be compatible with the specification-based approach that has not previously been integrated with blockchain [12]. Hence, we consider our proposal as the first novel proposal that integrates blockchain with a specification-based approach in one system.
- To perform an experiment of our proposal for validation purposes.

It is important to mention that this article is a part of ongoing research aimed at investigating and analysing possible methods for improving the accuracy and the reliability of attack detection techniques through blockchain by exploiting the advantages of blockchain and through developing some of its characteristics, to achieve the research goals. The research objective is to provide two detection techniques styles, firstly, standalone detection techniques and secondly, collaborative detection techniques. From a research point of view, the purpose of developing the first detection style is to emphasise the significance of the research goals. Hence, the provided development and implementation at this stage was performed for conceptual validation purposes. On the other hand, the purpose of developing the second detection style is to demonstrate the full picture of the research outcomes in both conceptual and practical aspects. The following sections are an attempt to describe how to achieve this objective.
The paper is structured as follows: Section 1 is the introduction to clarify the research domain, problem statement and research objectives. Section 2 lays out the research methodology, focusing on how related studies have been collected and classified. Section 3 consists of related works to highlight state-of-the-art studies as well as the field’s directions and limitations. Section 4 is the proposal framework to illustrate our proposal, as well as its components and their functionalities, with more detailed explanations and diagrams. Section 5 is the proposed experimental and attack models, which explains the environmental tools and how the experiment was performed. Section 6 details our findings and discussion to explain the results and reflect on our research goal. Section 7 is the conclusion, which summarises the project and outlines future work.

2. Methodology

Regarding our research methodology in this study, we first collected a set of relevant studies gathered from several academic databases. The search for collected studies specified that they should all be based on related terms, such as ‘detect’, ‘blockchain’ amongst others and that these related terms must be found in the studies’ titles. This condition is set to ensure that the gathered studies are directly related to the subject of study.

Secondly, we filtered the gathered studies based on their metadata and abstracts, according to a set of predefined rules, in order to exclude unrelated studies, categorise the remaining studies based on the extent of their relationship to the subject of this study, and obtain the relevant studies. The filtering rules, and thus the obtained classifications, depend on three criteria: detection deployment approaches, detection method approaches and blockchain utilisation level, as shown in Figure 1. In fact, these classification criteria are proposed based on either previous detection classification forms or the observed detection system requirement level as described below. The first and second criteria were explained in Section 1. The third criterion refers to whether the blockchain utilisation is at the technique level or the participant level. Blockchain utilisation in the technique level is the detection technique workflow itself, such as storing/retrieving data to/from the blockchain. Blockchain utilisation at the participant level is the task of the detection systems’ participants and includes sharing data among them and performing voting mechanisms.

![Figure 1. Classification criteria.](image-url)

After that, we went through the in-domain studies to analyse the extent of improvement in the detection techniques based on blockchain. Therefore, the studies highlighted in this paper will be either non-attack blockchain-based detection techniques, in order to explore some details of the target domains and the purpose of utilising blockchain, or attack blockchain-based detection techniques, to explore the applied detection deployments and methods as well as the applied blockchain mechanisms.
3. Related Works

In this section, we will initially explicate the relationship between detection mechanisms and blockchain technology in order to facilitate understanding of the research domain. After that, we will demonstrate some studies’ attempts to utilise blockchain for their non-attack detection systems by highlighting their domains and the purpose of blockchain utilisation. Next, we will discuss some related studies that seek to utilise blockchain for their attack detection systems, wherein we will show their main contributions, the purpose of utilising blockchain, and how blockchain has been applied in those proposals. In the end, we will evaluate the related studies’ achievements, along with their observed limitations, to identify the research gap.

3.1. Blockchain and Detection Relationship

Research concerning the interaction between detection techniques and blockchain technology has become a very popular trend in the last 3 years. According to the EBSCO academic search engine, the research studies that involve ‘detection’ and ‘blockchain’ as keywords in their titles exceeded 188 studies in the past 3 years. That shows the significance of this research field and suggests that researchers think the interaction between detection techniques and blockchain will make a promising enhancement in the domain.

By comparative screening and looking up related studies, we find that the relationship between detection and blockchain in terms of dependency is a reciprocal relationship; hence, the studies in this field can be divided into two main categories. The first category is the blockchain-based systems or blockchain technology itself, which need to enhance their own detection techniques [24]; thus, the work in these studies is to propose and develop appropriate detection techniques for these blockchain systems [25,26]. The second category is the detection techniques, which still need to overcome some of the challenges facing them, especially in a distributed environment in which integrating blockchain with these detection techniques could result in a promising solution for them and their target domains [27]. Hence, it could be argued that the objectives of each category are significantly different from the other. In fact, to the best of our knowledge, no study has presented a categorisation such as this, as described above and shown in Figure 2. We argue that the proposed categorisation would help researchers facilitate their investigations in this field. Additionally, we indicate that our proposal design is classified under the second category.

![Figure 2. The relationship between blockchain and detection techniques.](image)

3.2. Non-Attack Detection Techniques

For studies that seek to utilise blockchain for non-attack detection, we can observe that their detected non-attack and target domains are varied, including video platforms [28], Internet of Vehicles (IoV) [29], social media distributed databases [30], Mobile Crowdsensing (MCS) [31], Software-Defined Networking (SDN) [32] and Water Control Systems [33]. In [28], a fraud video detection system is proposed in which blockchain is utilised as a secure data storage system. In [29], driving law violations are sought through a proposal in order to reduce vehicle accidents, and blockchain acts as immutable data storage in
this proposal. Preserving sensitive user data in social media databases is the purpose of the proposed algorithm in [30], where blockchain is utilised as privacy-preserving data storage. Detecting the fake sensing activities in MCS is the main objective of [31], in order to protect MCS environment data, while ensuring collected data validation is the proposal’s primary blockchain task. Blockchain has been proposed to securely share the flow table rules and to validate and trace for the flow table rules in SDN controller, as an alternative to a trusted third party for SDN environments [32]. In [33], a wastewater recycling control system is proposed to manage wastewater efficiently, wherein blockchain is adapted to act as a wastewater reuse encouragement incentive method. Clearly, all of these proposals were not designed for cyber-attack detection purposes, but they used blockchain for several related purposes, as mentioned above. In addition, their blockchain usage has occurred in distributed environment target domains, so it could be argued that blockchain can be employed in the attack detection systems field, as it has points in common with these proposals.

3.3. Related Studies—Attack Detection Techniques

Moving on to some state-of-the-art studies, Suranjan et al. [34] propose a blockchain-based database management framework for malware detection systems. Based on the proposed system flow chart, it is obvious that the proposal utilises the signature-based detection technique for verification purposes. Additionally, the proposed system performs a monitoring technique to detect any new malware through spreading the detection alarm on the network and then recording the malware’s hash value on the blockchain, which is known to the other participants via the signature detection technique. Avoiding single-point-of-failure, as well as increasing the end-user’s security, are the main benefits of the proposal, according to the authors. The main challenge with this proposal is the lack of ability to detect zero-day malware due to the nature of signature-based detection techniques, which do not support detecting these kinds of attacks.

Jingjing et al. [35] proposed a framework for detecting and classifying android-based mobile device malware, named Consortium Blockchain for Malware Detection and Evidence Extraction. The main idea in this proposal is to involve two interacted blockchains: public blockchain and consortium blockchain. In the public blockchain, a multifaceted detection model is proposed to be used by consumers to detect and classify malware; this relies on some AI techniques and then stores the detected malware data on the public blockchain. In consortium blockchain, based on the detected malware data on the public blockchain, fact-based knowledge is created and stored by the members of anti-malware vendors in order to update the malware features. Based on the outcome of the experiment, in which multiple features have been used for malware detection, 92.5% detecting accuracy and a 94.6% recall rate can be achieved.

Mustafa and Adem [36] proposed a blockchain-based web attack detection model using a signature-based detection method. The use of blockchain is to store and share the blacklist of some common and updatable web-based attacks. Consequently, two levels of signature-based detection are performed: the first level is performed locally, based on the local copy of the blacklist, while the second level is performed based on the last copy of the blacklist found on the blockchain. Based on comparison results, the decision will be made and the blacklist on the blockchain will be updated by adding a new block if the detected attack is new. This experiment shows that web-based attack signatures can be detected and that blockchain can be updated by employing the proposal.

Ryusei et al. [37] proposed a blockchain-based malware file detection method. Their proposal seeks to exploit certain blockchain functionalities, specifically the immutability of stored data and the robustness of consensus management. Furthermore, this proposal attempts to combine two types of detection techniques: signature-based techniques, which utilise the blockchain benefit directly, and anomaly-based techniques, which work locally on each user system. Technically, the proposed method aims to detect suspected malware files by calculating the hash value of the downloaded executable file to check whether the
hash has been stored on the blockchain or not, which reflects whether the file is suspected to be malware or not, respectively. That is where the signature-based technique appears, along with one of the behaviour-based detection techniques performed locally to determine whether the file is malicious or not. Then, the results will be sent over the blockchain network to inform the other participants and vote about updated record information. Finally, based on the malicious degree, a decision whether to eliminate the detected file or not will be made by behaviour-based detection. Based on their experiment, the proposed method has improved the False Negative Rate and the False-Positive Rate. However, when the percentage of the total file hash the user will inspect decreases, the proposed system’s effectiveness will also be decreased. Furthermore, the need to carefully determine the threshold for total votes is essential to get enough votes for this parameter.

By making a critical comprehensive review of the previous four studies, we can summarise the deficiencies in the following points. Firstly, and most significantly from the detection method perspective, while the related studies have adopted Misuse-based, Anomaly-based detection method or both to be integrated with blockchain to enhance their functionality, no study, to the best of our knowledge, has attempted to utilise the specification-based detection method, though it appears to be the more promising detection method. Secondly, most of the relevant studies have not adequately determined the mechanism of detection of attacks; one of the related studies used artificial intelligence techniques to detect attacks, which are promising but costly, and require an exercise that may differ from one environment to another.

In addition, the related studies have discussed the detection of attacks in specific distributed environments, such as Android-based systems or web applications, which means that these solutions are not designed for generic distributed system environments. Additionally, in most of the relevant studies, the focus was on detecting attacks so that the malicious file can be discovered before it starts its work, but this is not the only attack detection mechanism. In other words, the focus was on discovering the behaviour of the attack “tool”, based on the hypothesis that it could be identified, but not on the behaviour of the target system that could be affected by the attacker in numerous and restricted ways. Most of the relevant studies have not been keen on optimising the beneficial services provided by the blockchain in detecting attacks; instead, they have limited blockchain to its secure data storage and sharing services.

Therefore, we can argue that our proposal, in contrast, could be considered as a novel proposal in terms of blockchain-based detection methods because our detection method can be classified as a specification-based detection method; no other blockchain-based attack detection system applies this mechanism [12]. Nevertheless, unlike many previous studies, we focus on detecting attacks based on a behaviour change that has occurred in the target systems with wilful negligence to determine the source of the attack (so long as we will be able to detect the attack before any damage to the system occurs), thus covering this important-but-neglected part of the attack detection technique in conventional techniques. From another novel perspective, our proposal has achieved a high level of blockchain utilisation in ways that have not been mentioned in previous studies, as we will explain in the following sections.

4. Proposal Framework

In this section, we will demonstrate our proposal’s design and its components in detail. By highlighting the proposal and its breakdown in a hierarchical structure and then demonstrating the proposal software system architecture by explaining its component functionalities, we will provide a detailed design for our proposal. In addition, the proposal’s implementation will be provided in the next section.

First of all, we would like to highlight our proposal’s five main features and their benefits in comparison to proposals from related studies.
Our proposal, to the best of our knowledge, is the first specification-based attack detection technique integrated with blockchain, which opens a new research direction in both detection techniques and blockchain fields.

2. Our proposal was designed based on generic distributed systems principles, thus it is compatible with and customisable for any type of distributed software system.

3. Our proposal is focused on extraction of the target system behaviours to provide a solid baseline for the detection technique and, thus, increase the detection techniques efficiency, instead of being distracted in identifying the various and renewable attack software behaviours, which is more computationally taxing and time consuming with less detection accuracy and efficacy.

4. Our proposal is producing a novel methodology to extract the runtime behaviours of the target system in such a manner that they will be stored in an immutable way.

5. Our proposal has leveraged extensively on the features of blockchain, going beyond its usual use as a secure medium to store data, into enhancing the immutability and reliability of detection techniques.

In the following subsections, we will highlight how our proposal exhibits all these features and benefits.

4.1. The Proposal and Its Breakdowns in a Hierarchical Structure

To begin with, Figure 3 shows the proposal and its breakdown in a hierarchical structure. The top level in the hierarchical structure represents the proposal from a technical point of view, which could be named Behavioural-based Attack Detection System via Blockchain for Distributed Software Systems.

![Figure 3. Overview of the proposal and its breakdowns in a hierarchical structure.](image)

The second level states that the proposed system consists of three integrated subsystems: Attack Detection Preparation (ADP) Subsystem, Runtime Attack Detection (RAD) Subsystem, and Blockchain Networks (BNs) Subsystem. The purpose of the ADP subsystem is to prepare both of the proposed attack detection systems by providing them with the required data and the target distributed software system by making it able to interact correctly and effectively with the runtime attack detection subsystem. The main objective of the RAD subsystem is to perform the proposed attack detection system on the target distributed software system, to report the detected attacks and to make suitable reactions in some cases. The BNs subsystem is responsible for managing the storage and retrieving behaviour and detected attack reports for both the ADP and RAD subsystems in an immutable secure manner.
The third level aims to state that each subsystem consists of a set of integrated techniques, each seeking to achieve its subsystem’s goal. For instance, one of the proposed techniques is Trusted Behaviour Immutable Storage Manager, which is located in the ADP subsystem; it is designed to store the extracted trusted behaviour in the blockchain in an immutable retrievable manner. We have already proposed, designed and implemented the majority of techniques in our proposal, especially where the techniques are relative to our research objectives.

The last two layers show two optional breakdowns: Components and Phases. In some contexts, we need to divide the proposed technique into several components in order to increase the readability of the proposal and to make the tracking and tracing easier and faster. Moreover, in some cases, we use the term ‘phase’ when the procedures of some techniques and/or components need to be explained in an ordered fashion.

4.2. The Proposal Deployment Architecture

The second feature of the proposal was designed to be deployed in any generic distributed software system. The key aim of the proposed deployment method is preparing the target system source codes through the ADP subsystem and then making the system run under the proposal supervision. Consequently, any target distributed software systems can be compatible with our proposal if the target system source codes are provided. Practically, the target system will include a part of the proposal in its structure as an output of the ADP subsystem processes. Additionally, during runtime, the target system will be monitored and controlled by the proposal to the extent of attack detection processes. The details of the proposed processes will be explained in the following subsection.

4.3. The Proposal Software System Architecture

Next, as shown in the following diagrams, we have a software system architecture for our proposal that involves three integrated subsystems and several interconnected techniques. The first subsystem we propose is the Attack Detection Preparation (ADP) Subsystem, which appears in the green L-shape border shown in Figure 4. The main purposes of this subsystem are (1) to extract the trusted behaviours of the target distributed software system, e.g., website, to be stored in an immutable fashion through blockchain; (2) to generate an enhanced target system that is able to feed the Runtime Attack Detection (RAD) Subsystem real-time events that represent the runtime behaviours of the target system.

The ADP Subsystem consists of five main techniques that work in two directions as a reflection of the subsystem’s objectives. The first technique is Trusted Behaviours Extractor, which aims to analyse the source codes of the target system in the same manner as extracting the sequence diagrams from any other source codes, but in a text-based layout. In fact, this technique works similarly to reverse engineering technique principles, but it is more lightweight, as it simply reverses the source codes to text-based design descriptions. The output of this technique is lists of ordered trusted behaviour in text-based sequence diagram events linked with the behaviour form of the source codes, e.g., sequential, parallel, etc., in order to help RAD techniques work accordingly during runtime. The second technique is Trusted Behaviours Immutable Storage Manager, which is designed to store the trusted behaviour events on a static blockchain that has been configured to be updated only by authorised administrators and to exist as a read-only blockchain for the rest of the blockchain network participants. The third technique is Detected Attack Analyser, an optional technique that can access the reports of detected attacks on demand, in order to perform an offline analysis for the detected attacks as well as to perform some sort of long-term auditing to enhance the detection system’s functionalities. The above-mentioned techniques work in the direction of extracting and storing the trusted behaviours of the target system.

The fourth ADP technique is Enhanced Source Codes Generator, which takes the target system source codes, along with the extracted trusted behaviour events, as an input. The goal of this technique is to generate enhanced source codes that include new statements
responsible for giving real-time events during the runtime. This technique works in a similar way as the Instrumentor applied in software testing systems, which analyses the source codes to add new statements that, when executed, will generate a set of test cases and their results [38]. The fifth ADP technique is Source Code Compiler and EXE Generator, which easily compiles the enhanced source codes and makes executable files for the target system. These executable files will work exactly as the original target system should work, but they have the additional feature of generating real-time events to be fetched by RAD techniques. The last two ADP techniques work in the direction of generating an enhanced target system. In fact, ADP subsystem techniques functionalities show how the first and third features of our proposal are achieved.

Figure 4. Software system architecture for Attack Detection Preparation (ADP) Subsystem.

The second subsystem proposed is the Runtime Attack Detection (RAD) Subsystem, located in the red rectangular border as shown in Figure 5 below. The main objectives of this subsystem are (1) to monitor and fetch the target system’s real-time events during the runtime; (2) to perform the proposed detection mechanism based on the user/admin selection; (3) to store the extracted real-time events in an immutable fashion via blockchain. We should mention that the target system is run under the proposal supervisor, so the target system cannot work without running the attack detection system. Additionally, we proposed three detection modes and two detection mechanisms. Each detection mode, as will be described later, will determine the appropriate performance and performing order of detection mechanisms, as well as the way of storing the real-time events, either event-by-event or all events at once. The purpose of these detection modes is to achieve the optimal target and detection systems performance based on the sensitivity of the target system, i.e., higher sensitivity requires a stricter detection mode. The purpose of this detection mechanism is to determine the detection comparison level, which can be either at the level of event details implemented by the Each-To-Each Events Comparison technique, or at the level of the whole events sequence, number, and layout implemented by the All-To-All-Events Comparison technique.

The RAD Subsystem consists mainly of five techniques. The first technique is Runtime Subsystem Initiator, which is responsible for initialising the detection system, identifying
the referred detection mode by the user, and importing the behaviour form of the target system from the static blockchain. The output of this technique is a package containing the detection mode selected by users, the behaviour form imported from static blockchain, and the location where the real-time events will be posted. Based on the result of the first technique, one of the three detection modes will be run. If the detection mode selected by the user was A (which means that no runtime attack detection is required), the Mode A Manager technique proposed for low-data-sensitive target systems will work as shown in Figure 5a. Mode A Manager will simply execute the target system via EXE Executor technique and then fetch the real-time events through the All-Events Collector technique. Once the target system running the session is terminated, the Mode A Manager will not perform any sort of runtime detection mechanisms and will send all events, along with its behaviour mode and detection mode, to the Runtime Behaviours Immutable Storage Manager technique, which is responsible for storing gain data on the Runtime Behaviour Events Dynamic Blockchain. This blockchain is configured to be updated and accessed by any blockchain network participants for storing the real-time events, while the retrieval of their data is assigned to administrators and certain predefined participants.

If the selected detection mode was B (which means that runtime attack detection at the end of the session is required), the Mode B Manager technique proposed for moderately-data-sensitive target systems will work as shown in Figure 5b. The Mode B Manager will execute the target system via EXE Executor technique and then fetch the real-time events through the All-Events Collector technique. Once the target system running session is terminated, the Mode B Manager will perform both detection mechanisms, the All-To-All-Events Comparison technique first, followed by the Each-To-Each Events Comparison. Both of these comparison techniques will call the corresponded trusted behaviours from the static blockchain as needed in order to perform the detection comparison. The results of these comparisons, either detected attacks or not, will be reported to the system user. This report will be stored on the Detected Attack Reports Dynamic Blockchain, which has the same configuration as the Runtime Behaviour Events Dynamic Blockchain but in a different data structure for the stored data. Simultaneously, all obtained real-time events, along with their behaviour mode and detection mode, will be sent to the Runtime Behaviours Immutable Storage Manager technique, which is responsible for storing the events on the Runtime Behaviour Events Dynamic Blockchain.

![Figure 5. Cont.](image-url)
Figure 5. Runtime Attack Detection (RAD) Subsystem software system architectures for (a) Mode A; (b) Mode B; (c) Mode C.

If the selected detection mode was C (which means that real-time attack detection is required), the Mode C Manager technique proposed for high-data-sensitive target systems will work. The Mode C Manager will execute the target system via EXE Executor technique and then fetch the real-time events through the Specific Event Collector technique. While the target system is running and the real-time event is fetched, the Each-To-Each Events Comparison technique and then the Runtime Behaviour Immutable Storage Manager technique will work, as described before, but in concurrency mode. The stored real-time events, in this case, will not be sorted for the whole session as is the case for the other mode managers; rather, it will be an event-by-event process, where the allocated details of each stored event can help related techniques to collect all related events in one unit, as desired. In actual fact, RAD subsystem technique functionalities show how the fourth feature of our proposal has been achieved.
The third subsystem proposed is the Blockchain Networks (BNs) Subsystem, located in the blue rectangular border on the right side of Figure 6. The main objective of this subsystem is to provide an immutable, secure, transparent, and tractable storing and retrieving mechanism for the desired data that involve trusted events, real-time events, and detected attack reports to their corresponding blockchain. BNs consist of three private (permission-based) blockchains that have been mentioned and explained above. We chose to adopt the private blockchain given its compatibility with the majority of attack detection techniques and requirements, such as data privacy. However, it can also be a public blockchain if a particular detection system requirement is more conveniently compatible with public blockchain functionality in light of what has been demonstrated in Section 1. Each of the proposed blockchains has a specific configuration and data structure for its data blocks’ body, based on their use scenario. The block is the unit of storage in the blockchain, which represents one event or detected report in our proposal.

Based on the Trusted Behaviours Extractor, the event header will be the same as any mined block, including timestamp, block ID, previous block hash, etc., wherein the event body structure looks like text-based sequence diagram data, as shown in Table 4. Any potential attack will affect one or more of the event header or/and event body details and is thus easily detected. The report has a different data structure that is readable by human users and simultaneously readable by the related techniques. Consequently, the write (transaction)/read (call) from/to the blockchain could occur for one or several blocks, based on this technique’s mechanism. Furthermore, the consensus protocol adopted in our proposal is Proof-of-Authority (POA) since it is compatible with private blockchain functionalities. Additionally, because attack detection techniques require low latency and the high throughput of the transactions. That is because the response speed in terms of reading/writing data from/to the blockchain is an important requirement for attack detection techniques; otherwise, the reliability of these techniques will be affected. However,
any consensus protocol can be adopted based on the desired system requirements, as demonstrated in Section 1. In fact, BNs subsystem blockchains functionalities show how the fifth feature of our proposal has been achieved.

Table 4. Event’s body data structure.

| Event Order | Class Name (Where Data Will Be Reserved) | Method Name | Method Returned Data Type | Method Parameters Count | Method Parameters Data Types |
|-------------|------------------------------------------|-------------|---------------------------|-------------------------|-----------------------------|
| 1           | School Registration                      | main()      | Void                      | 1                       | String [],                  |
| 2           | Actor                                    | println()   | Void                      | 1                       | String                      |
| 3           | Actor                                    | println()   | Void                      | 1                       | String                      |
| 4           | School Registration                      | StringToDate() | Date | 1 | String |
| 5           | Actor                                    | println()   | Void                      | 1                       | String                      |
| 6           | Actor                                    | println()   | Void                      | 1                       | String                      |
| 7           | Checker                                  | checker()   | Boolean                   | 3                       | String | Date | int |
| 8           | Storage                                  | storage()   | Boolean                   | 3                       | String | Date | int |

To sum up, Figure 7 shows the proposal software system architecture, including the integration between all proposal subsystems as well as the full interactions between the subsystems’ techniques. In addition, it explains the symbols used in the diagrams, as well as their meanings.

Figure 7. The proposal software system architecture.

5. Proposal Experiment and Attack Models

In this section, we will present the experiment we developed in order to make an implementation for our proposal. Specifically, we will first demonstrate the environment tools we used and then the overall steps we followed to perform our experiment. Finally, we will present two attack models that we critically made to check the validation of our proposal by testing the detection system designed for this experiment.

5.1. Environment Tools Used in the Experiment

We should mention that to be realistic our proposal must have three main parts: the distributed target system, the attack detection system, and a blockchain environment. We developed a client-server target system in Java named ‘School Registration’, which emulates a student registration system in schools, and involves a set of conditions and communications between the client and the server to verify and implement services. We
chose Java as the programming language for the distributed target system, as it is one of the common languages and is supported by many lexical analyser techniques, which should help us gain even better analysis of the source codes as a part of our proposal. We also implemented our attack detection system in Python, as it is a promising programming language with a high level of abstraction and sufficient compatibility with blockchain environments. As mentioned before, the detection system has developed in the manner that supervises the target system and corresponds with it in the distribution form. Last, we chose Ethereum as the blockchain environment, as it is a stable environment and provides a premier (private) blockchain option. As we work on proofing the concept, we chose the Ganache platform, which is a personal Ethereum platform that helps validate the use of the blockchain and its smart contracts without the need for direct interaction with public Ethereum networks. Communication between the attack detection system and the blockchain is done through smart contracts written in the Solidity language. This is the language of smart contracts in Ethereum and was chosen with the knowledge that the contract between the attack detection system and smart contracts is performed through the Web3 library. Figure 8 shows some screenshots of the tools used in the experiment for the ADP and RAD subsystems, respectively.

5.2. Experiment Progress

This section presents the design science validation. Structurally, we started the experiment by developing the target system, while making sure to achieve the principles of distributed systems. After that, the first proposal subsystem techniques ADP were either developed or imported, if they had previously been built; then, all techniques were integrated into one subsystem and the required tests were performed. The second proposal subsystem techniques RAD were developed mode-by-mode; then all techniques were integrated into one subsystem and the required tests were performed. In the third proposal, subsystem blockchains BNs were created in conjunction with the first and second proposal subsystems.
Practically, the proposal works in three stages. Stage one works firstly offline at a software house or at the system administrator’s location by running ADP and BNs subsystems first, in order to initialise the system environment; then create the trusted behaviour events, store them on blockchain, and create an enhancement target system source code, as described before. Stage two works during runtime by running the attack detection system first and then the user chooses the attack detection mode; at that point the detection will automatically run, and the detection system will work based on the selected detection mode, where the final output is the immutable stored detected reports and dynamic events, as described in the previous section. Stage three works during runtime to evaluate the proposal by measuring the performance among the three detection modes and performing a set of test cases to validate the ability to detect the various types of attacks through the detection modes and detection models, as explained in Section 6.

5.3. Attack Models

Currently, cyberattacks are considered one of the Internet’s key concerns. Different attack tools are grouped into the Internet by computer virus categories and target the computer device linked to the Internet. A growing danger of attack may be caused by a criminal enterprise focused on connecting illegal objectives such as distributing spam messages or collecting sensitive data that an unauthorised individual cannot otherwise access [39]. However, there is a growing diversity of malware that affects protection techniques as well as classic security systems, such as antivirus scanners which deliver ineffective results. Consequently, hundreds of Internet hosts may be affected due to malware [40].

The article discusses the claims that malware exploration is a coherent discussion of the characteristics that separate malware into various categories and is considered an efficient and effective malware study. Malicious software is divided into complex and diverse actions based on various modifications to advanced network connections. Malware connections between different and the same families share similar behaviour, and their patterns, which are also the same, are based on certain modifications that may capture the specific machine [40]. Our task is to explore this malware’s sharing pattern as it is an automated analysis, and its basic goal is to provide a reliable structure that will help identify the malware community based on their actions.
Besides attack style, the most complicated malware analysis is an event-based model based on the simple technique of representing the information about discrete event systems. Several attacks may be analysed into basic components based on complex scalable event sequences. There is also another justification to use the event-based model that some researchers use in the methodology design evaluation stage, as it offers a working environment for the fundamental research and assessment goals used to follow the research process [41].

For example, the internet’s surface level could be attacked by the user as if it were a drive-based attack. Drive-by-download is considered an easy way to spread malware. Hackers are constantly searching for vulnerable websites and considering the creation of malicious HTTP or PHP code scripts that could be used in these websites to capture data. These kinds of states could be installed from any website by a machine, and it could take the form of an IFRAME that redirects the victim to download a particular site that is managed by criminals [42]. The event driver download arrangement is attacked; Figure 9 could reflect this situation. After exploring malicious software code, it could be executed as a resistance subscription, ending up with the static scan’s single object. However, a web browser may analyse the behaviour detection component and provide information about blocks and malicious activity.

Figure 9. Typical sequence of events in a drive-by-download attack [43].

We created two attack models in order to measure our proposal’s efficiency in achieving its main goal of attack detection, as well as to test the validity of having immutable data in these kinds of systems. The first attack model is the event-based model, which represents the attacks that may affect the existing event or event details. The second attack model is the behaviour-based model, which represents the attacks that affect the system behaviour layout.

Therefore, we injected the proposal with both of these attack models during runtime to perform our validation testing. The preliminary results show that the proposal was able to detect both attack models based on the selected detection modes related to the detection mechanisms. As mentioned in the previous section, we proposed three detection modes that utilise two proposed detection mechanisms. Hence, the Each-To-Each Events Comparison technique is the first detection mechanism designed to detect event-based attacks, and it is utilised in Modes B and C. The All-To-All-Events Comparison technique is the second detection mechanism, and was designed to detect behaviour-based attacks; it is utilised in Mode B. Additionally, due to the immutability of stored trusted events, the accuracy of the detection mechanisms was high.
Based on the above-mentioned attacks’ background information, we also propose two attack models in order to measure our proposal’s efficiency in achieving its main goal of attack detection, as well as test the validity of having immutable data in these kinds of systems. The first attack model is the event-based model, which represents the attacks that may affect the existing event or event details. For example, if the 10th event occurring in system A indicates that Method A sent two values to Method B, then the event-based model attacks could affect the event by deleting it, inserting a different event, or changing all/some of the event(s) details, such as method names or number/types of variables, etc. The second attack model is the behaviour-based model, which represents the attacks that affect the system behaviour layout. For example, if System B has 20 events occurring in one sequence, then the behaviour-based model attacks could affect the number of events or the structure of the event sequence.

Therefore, we injected the proposal with both of these attack models during runtime to perform our validation testing. The preliminary results show that the proposal was able to detect both attack models based on the selected detection modes related to the detection mechanisms. As mentioned in the previous section, we proposed three detection modes that utilise two proposed detection mechanisms. Hence, the Each-To-Each Events Comparison technique is the first detection mechanism designed to detect event-based attacks, and it is utilised in Modes B and C. The All-To-All-Events Comparison technique is the second detection mechanism, and has been designed to detect behaviour-based attacks; it is utilised in Mode B. Additionally, due to the immutability of stored trusted events, the accuracy of the detection mechanisms was high.

6. Findings and Discussion

To start with, we should mention that this paper uses design science (DS) methodology in general. Iivari and Venable [44] define design science research as an approach activity that builds novel artefacts for addressing issues or achieving improvement and establishing new means to achieve research goals. These methods create a new reality instead of explaining existing issues or trying to understand the existing reality. The methodology was chosen due to the need for innovation and reliable solutions for the problems. DS involves four main phases, namely problem identification, design, implementation, and evaluation that provide feedback to enhance the design again. This research achieves these phases as explained in the next sections.

Based on our literature review, detailed design, and implementation-based validation, we can state the obtained findings. Firstly, there is a significant benefit to utilising the specification-based detection method that has been applied in the ADP subsystem through extracting the trusted behaviours as well as generating an enhanced source code. In fact, the importance of this method is to reduce the false-positive rate in detecting system attacks to the lowest possible level—0%—due to its robustness characteristic, by identifying all the trusted behaviours of the target system, even unusual or rare behaviours, which makes attack detection systems more accurate.

In contrast, it could be stated that the optimal-automated implementation of the specification-based detection method to extract all trusted behaviours is a hard task. This is due to the high computational cost of determining and extracting all the trusted behaviours with a sufficient distinction between the expected overlapped behaviours. Additionally, the high cost of comparing the trusted behaviours with real-time behaviours during the runtime may require the application of some predictions and probability mechanisms during the fetching of the real-time behaviours to avoid any false-positive detection; this is another challenge facing this optimal-automated implementation. Hence, it could be argued that more investigations and improvement for specification-based detection automation is an essential area of this research field.

In our experiment, we applied two attack models to check the validity of our proposal. Clearly, as a consequence of applying the specification-based detection method, the proposal has detected both of the designed attacks. However, each proposed detection mode
has a different rate of target system performance, as well as a different rate of detection response speed. That is because of the trade-off between the target system’s performance and detection speed, as a consequence of the required sensitivity level of the target system, as explained before. In Figure 10, we present the observed difference between target system performance speed and detection response speed among the proposed detection modes.

As Figure 10 highlights, Mode A has a high target system performance speed, while the detection response speed is slow. This is due to the low sensitivity of the target system. Thus, no high detection response speed is required in this mode, so no detection response will affect the target system performance. Mode B has a convergent performance between its target system speed and detection response speed. That is because of the moderate sensitivity of the target system’s data and hence the required moderate detection response for this mode, so the performance in the target system and the detection system will be moderate. Mode C, however, has opposite results to Mode A due to the high sensitivity of the target system, which requires a high detection response and subsequently affects the target system performance. Overall, the proposal has provided three detection modes in order to handle the trade-off between target system and detection system performance; the target system administrator can decide which detection mode is suitable for their needs.

Another finding of this research is the high efficiency of blockchain usage in attack detection systems. While some believe that blockchain can just be used as a secure data storage and sharing medium for this type of application, blockchain offers many other services and benefits. The immutability of stored data is one of the key features of blockchain that improve the robustness and accuracy of the detection techniques and thus the detection system’s overall reliability. In our proposal, the dependency and trustworthiness of the immutably stored behaviours are very high.

In addition, the transparency and non-repudiation of the stored data is another feature of utilising blockchain in detection systems; this feature comes from the fact that any behaviours will be recorded, along with its owner, whether they are trusted or real-time events, and will be accessible by all participants in the network. This feature is utilised implicitly in our proposal via the detected attack analyser technique, where it will be better exploited in the coming enhanced proposals. Traceability is another feature gained from blockchain, which allows for the tracing of a set of related events and then memorisation of the whole behaviour structure to be utilised as a trusted input of a detection mechanism designed for this level of detection. Overall, it could be argued that our proposal has achieved its objectives and overcome the limitations mentioned in previous proposals.

Figure 10. Target system performance speed vs. detection system response speed among detection modes.
Based on our experiment, we can generally indicate that adopting blockchain in any detection system will increase the above-mentioned beneficial features in comparison to the detection systems that apply any other distributed method of data storage and exchange. However, it could be argued that as a consequence of the costly mining for each behaviour’s event, the overall performance may be affected, especially when data is written to the blockchain, where the effect is based on the applied consensus protocol such as PoW or PoA and network throughput. In fact, it should be said that these effects should be tolerated in order to obtain the required advantages of the blockchain. Therefore, it is observed that there is another trade-off between immutability and performance by adopting blockchain or not, respectively.

7. Conclusions

In this paper, we proposed a design for standalone behaviour-based attack detection techniques for distributed software systems. We also provided detailed design and implementation-based validations for our proposal, including two proposed attack models. Our preliminary results show that the proposal can provide a promising solution for attack detection systems via optimal utilisation of blockchain functionalities, as well as novel utilisation of the specification-based detection method in this field. In the near future, we will utilise the blockchain to an even greater extent than in this proposal since it is also compatible with collaborative attack detection systems. Collaborative attack detection techniques have become a significant type of attack system, hence utilising blockchain for this type of detection system represents a substantial integration. Additionally, we intend to perform this proposal in a more realistic experimental environment and apply various evaluation metrics to evaluate the proposal even more accurately.

**Author Contributions:** Conceptualisation, F.E., H.A. and K.A.; methodology, H.A. and K.A.; software, H.A.; validation, H.A., F.E. and K.A.; writing—original draft preparation, H.A., A.A. (Abdullah Algarni) and A.A. (Abdulaziz Attaallah); writing—review and editing, H.A., K.A. and F.E.; visualisation, H.A., F.E., A.A. (Abdullah Algarni) and A.A. (Abdulaziz Attaallah); supervision, F.E.; funding acquisition, A.A. (Abdullah Algarni) and A.A. (Abdulaziz Attaallah). All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was funded by the Deanship of Scientific Research (DSR) at King Abdulaziz University, Jeddah, under grant no. (RG-21-611-38). The authors, therefore, acknowledge with thanks the DSR technical and financial support.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Not applicable.

**Conflicts of Interest:** The authors declare no conflict of interest.

**References**

1. Van Steen, M.; Tanenbaum, A.S. *Distributed Systems*; Prentice Hall: Leiden, The Netherlands, 2017; ISBN 1543057381.
2. Jin, W.; Liu, T.; Qu, Y.; Zheng, Q.; Cui, D.; Chi, J. Dynamic structure measurement for distributed software. *Softw. Qual. J.* 2018, 26, 1119–1145. [CrossRef]
3. Fleischmann, A. *Distributed Systems: Software Design and Implementation*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2012; ISBN 364278612X.
4. Arshad, N.; Heimbigner, D.; Wolf, A.L. Deployment and dynamic reconfiguration planning for distributed software systems. *Softw. Qual. J.* 2007, 15, 265–281. [CrossRef]
5. Morgan, S. Cybercrime To Cost The World $10.5 Trillion Annually By 2025. Available online: https://cybersecurityventures.com/hackerpocalypse-cybercrime-report-2016/ (accessed on 17 April 2021).
6. Cukier, M. Study: Hackers Attack Every 39 Seconds. Available online: https://eng.umd.edu/news/story/study-hackers-attack-every-39-seconds (accessed on 17 April 2021).
7. 29 Must-know Cybersecurity Statistics for 2020—Cyber Observer. Available online: https://www.cyber-observer.com/cyber-news-29-statistics-for-2020-cyber-observer/ (accessed on 17 April 2021).
8. Stouffer, K.; Zimmerman, T.; Tang, C.; Pease, M.; Lubell, J.; Cichonski, J.; McCarthy, J. Cybersecurity Framework Version 1.1 Manufacturing Profile; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2020.

9. Alkadi, O.; Moustafa, N.; Turnbull, B. A review of intrusion detection and blockchain applications in the cloud: Approaches, challenges and solutions. *IEEE Access* 2020, 8, 104983–104917. [CrossRef]

10. Benaddi, H.; Ibrahimi, K. A review: Collaborative intrusion detection for IoT integrating the blockchain technologies. In Proceedings of the 2020 8th International Conference on Wireless Networks and Mobile Communications (WINCOM), Reims, France, 27–29 October 2020; pp. 1–6.

11. Krishna, A.M.; Tyagi, A.K. Intrusion detection in intelligent transportation system and its applications using blockchain technology. In Proceedings of the International Conference on Emerging Trends in Information Technology and Engineering, ic-ETITE, Vellore, India, 24–25 February 2020.

12. Dawit, N.A.; Mathew, S.S.; Hayawi, K. Suitability of blockchain for collaborative intrusion detection systems. In Proceedings of the 12th Annual Undergraduate Research Conference on Applied Computing, URC, Dubai, United Arab Emirates, 15–16 April 2020.

13. Kumar, M.; Singh, A.K. Distributed intrusion detection system using blockchain and cloud computing infrastructure. In Proceedings of the 4th International Conference on Trends in Electronics and Informatics, ICOMEI, Tirunelveli, India, 15–17 June 2020; pp. 248–252.

14. Meng, W.; Tischhauser, E.W.; Wang, Q.; Wang, Y.; Han, J. When intrusion detection meets blockchain technology: A review. *IEEE Access* 2018, 6, 10179–10188. [CrossRef]

15. Tian-yang, G.; Yin-Sheng, S.; You-yuan, F. Research on software security testing. *World Acad. Sci. Eng. Technol. 2010*, 70, 647–651.

16. Drescher, D. Blockchain Basics–A Non-Technical Introduction in 25 Steps; Apress: New York, NY, USA, 2017. [CrossRef]

17. Singhal, B.; Dhameja, G.; Panda, P.S. Beginning Blockchain: A Beginner’s Guide to Building Blockchain Solutions; Springer: Berlin/Heidelberg, Germany, 2018; ISBN 1484234448.

18. Li, X.; Jiang, P.; Chen, T.; Luo, X.; Wen, Q. A survey on the security of blockchain systems. *Future Gener. Comput. Syst.* 2020, 107, 841–853. [CrossRef]

19. Niranjamanurthy, M.; Nithya, B.N.; Jagannatha, S. Analysis of blockchain technology: Pros, cons and SWOT. *Clust. Comput.* 2019, 22, 14743–14757. [CrossRef]

20. Iredale, G. Blockchain vs. Database: Understanding the Difference. Available online: https://101blockchains.com/blockchain-vs-database-the-difference/ (accessed on 17 April 2021).

21. Ruan, P.; Dinh, T.T.A.; Loghin, D.; Zhang, M.; Chen, G.; Lin, Q.; Ooi, B.C. Blockchains vs. Distributed databases: Dichotomy and fusion. In Proceedings of the 2021 ACM SIGMOD International Conference on Management of Data, Xi’an, China, 20–25 June 2021; pp. 543–557.

22. Nguyen, G.-T.; Kim, K. A survey about consensus algorithms used in blockchain. *J. Inf. Process. Syst.* 2018, 14, 101–128.

23. Kaur, S.; Chaturvedi, S.; Sharma, A.; Kar, J. A research survey on applications of consensus protocols in blockchain. *Secur. Commun. Netw.* 2021, 2021. [CrossRef]

24. Putz, B.; Pernul, G. Detecting blockchain security threats. In Proceedings of the 2020 IEEE International Conference on Blockchain, Rhodes Island, Greece, 2–6 November 2020; pp. 313–320.

25. Chen, W.; Zheng, Z.; Cui, J.; Ngai, E.; Zheng, P.; Zhou, Y. Detecting Ponzi schemes on ethereum: Towards healthier blockchain technology. In Proceedings of the World Wide Web Conference, WWW, Lyon, France, 23–27 April 2018; pp. 1409–1418. [CrossRef]

26. Farrugia, S.; Ellul, J.; Azzopardi, G. Detection of illicit accounts over the ethereum blockchain. *Expert Syst. Appl.* 2020, 150. [CrossRef]

27. Mishra, S.; Tyagi, A.K. Intrusion detection in Internet of Things (IoTs) based applications using blockchain technology. In Proceedings of the 2019 Third International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC), Palladam, India, 12–14 December 2019; pp. 123–128. [CrossRef]

28. Dhiran, A.; Kumar, D.; Abhishek; Arora, A. Video fraud detection using blockchain. In Proceedings of the Second International Conference on Inventive Research in Computing Applications (ICIRCA), Coimbatore, India, 15–17 July 2020; pp. 102–107. [CrossRef]

29. Davydov, V.; Bezzateev, S. Accident detection in internet of vehicles using blockchain technology. *Int. Conf. Inf. Netw.* 2020, 766–771. [CrossRef]

30. Chen, Y.; Xie, H.; Lv, K.; Wei, S.; Hu, C. DEPLEST: A blockchain-based privacy-preserving distributed database toward user behaviors in social networks. *Inf. Sci. 2019*, 501, 100–117. [CrossRef]

31. Arafeh, M.; El Barachi, M.; Mourad, A.; Belqasmi, F. A blockchain based architecture for the detection of fake sensing in mobile crowdsensing. In Proceedings of the 4th International Conference on Smart and Sustainable Technologies, Brno, Czech Republic, 18–21 June 2019; pp. 1–6. [CrossRef]

32. Guan, Z.; Lyu, H.; Zheng, H.; Li, D.; Liu, J. Distributed audit system of SDN controller based on blockchain. *Lect. Notes Comput. Sci.* 2019, 11911, 21–31.

33. Iyer, S.; Thakur, S.; Dixit, M.; Katkam, R.; Agrawal, A.; Kazi, F. Blockchain and anomaly detection based monitoring system for enforcing wastewater reuse. In Proceedings of the 2019 10th International Conference on Computing, Communication and Networking Technologies, ICCCN, Kanpur, India, 6–8 July 2019.
34. Talukder, S.; Roy, S.; Al Mahmud, T. An approach for a distributed anti-malware system based on blockchain technology. In Proceedings of the 11th International Conference on Communication Systems & Networks (COMSNETS), Bengaluru, India, 7–11 January 2019; pp. 1–6. [CrossRef]

35. Gu, J.; Sun, B.; Du, X.; Wang, J.; Zhuang, Y.; Wang, Z. Consortium blockchain-based malware detection in mobile devices. IEEE Access 2018, 6, 12118–12128. [CrossRef]

36. Tanriverdi, M.; Tekerek, A. Implementation of blockchain based distributed web attack detection application. In Proceedings of the 1st International Informatics and Software Engineering Conference: Innovative Technologies for Digital Transformation, IIEC, Ankara, Turkey, 6–7 November 2019; pp. 1–6.

37. Fuji, R.; Usuzaki, S.; Aburada, K.; Yamaba, H.; Katayama, T.; Park, M.; Shiratori, N.; Okazaki, N. Blockchain-based malware detection method using shared signatures of suspected malware files. Adv. Intell. Syst. Comput. 2020, 1036, 305–316.

38. Eytani, Y.; Havelund, K.; Stoller, S.D.; Ur, S. Towards a framework and a benchmark for testing tools for multi-threaded programs. Concurr. Comput. Pract. Exp. 2007, 19, 267–279. [CrossRef]

39. Holz, T.; Engelberth, M.; Freiling, F. Learning more about the underground economy: A case-study of keyloggers and dropzones. In Proceedings of the European Symposium on Research in Computer Security, Luxembourg, 23–27 September 2019; pp. 1–18.

40. Rieck, K.; Trinius, P.; Willems, C.; Holz, T. Automatic analysis of malware behavior using machine learning. J. Comput. Secur. 2011, 19, 639–668. [CrossRef]

41. Yan, H.; Wang, J.; Zhang, H.; Shen, H.; Zhan, X. Event-based security control for stochastic networked systems subject to attacks. IEEE Trans. Syst. Man Cybern. Syst. 2018, 50, 4643–4654. [CrossRef]

42. Ibrahim, S.; Al Herami, N.; Al Naqbi, E.; Aldwairi, M. Detection and analysis of drive-by downloads and malicious websites. In Proceedings of the International Symposium on Security in Computing and Communication, Trivandrum, India, 18–21 December 2019; pp. 72–86.

43. See Everything, Fear Nothing Threat Solution Series. Available online: https://www.rsa.com/content/dam/en/solution-brief/asoc-malicious-protocols-gh0st-rat.pdf (accessed on 17 April 2021).

44. Iivari, J.; Venable, J.R. Action Research and Design Science Research—Seemingly Similar but Decisively Dissimilar. ECIS 2009 Proceedings. 2009. Available online: https://aisel.aisnet.org/ecis2009/73 (accessed on 17 April 2021).