SmartRetro: Blockchain-based Incentives for Distributed IoT Retrospective Detection

(Invited Paper)

Bo Wu*, Qi Li*†, Ke Xu*, Ruoyu Li‡, Zhuotao Liu§
*Department of Computer Science and Technology, Tsinghua University, Beijing, China
†Graduate School at Shenzhen, Tsinghua University, Shenzhen, China
‡Department of Computer Science, Columbia University, New York, USA
§Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Illinois, USA
Emails: {wub14@mails., qi.li@sz., xuke@}tsinghua.edu.cn, rl2929@columbia.edu, zliu48@illinois.edu

***Abstract*—Internet of Things (IoT) has already been in the period of rapid development and widespread deployment, while it is still vulnerable to various malicious attacks. Security detection before system installation is not enough to ensure that IoT devices are always secure, because newly emerging vulnerabilities can still be exploited to launch attacks. To address this issue, retrospective detection is often required to trace the security status of IoT systems. Unfortunately, existing centralized detection mechanisms cannot easily provide a comprehensive security analysis. In particular, consumers cannot automatically receive security notification whenever a new vulnerability is uncovered. In this paper, we propose a novel blockchain-powered incentive platform, called SmartRetro, that can incentivize and attract more distributed detectors to participate in retrospective vulnerability detection and contribute their detection results. Leveraging smart contracts, consumers in SmartRetro receive automatic security feedback about their installed IoT systems. We perform the security and theoretical analysis to demonstrate that SmartRetro achieves our desirable security goals. We further implement SmartRetro prototype on Ethereum to evaluate its performance. Our experimental results show SmartRetro is technically feasible and economically beneficial.**

***Index Terms*—Blockchain, Incentives, Retrospective Detection**

I. Introduction

Internet of Things (IoT) has achieved rapid development and widespread deployment in recent years: IoT connected devices will reach almost 31 billion by 2020 [1]. However, IoT devices are usually vulnerable to various attacks due to insecure design and implementation. Although anti-virus software is available for identifying security issues before an IoT device is installed, other vulnerabilities that cannot be identified with pre-installation scanning would remain undiscovered, and could be exploited to launch attacks after installation. For instance, the Mirai botnet [2] infests unsecured IoT devices (*i.e.,* cameras with default login credentials) to launch large-scale DDoS attacks against the Internet services [3] [4]. Given the foreseeable growth in IoT device deployment, it is crucial to build a secure IoT ecosystem to minimize its social impact. Retrospective security detection for IoT devices, as a necessary complement to pre-installation scanning, is a critical building block towards that goal.

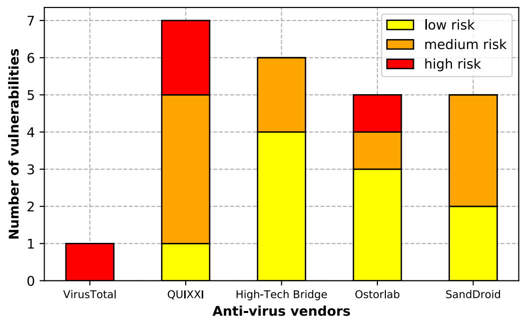

Fig. 1. The detection results for an IoT App (`Nest`).

Existing mechanisms for retrospective detection have several limitations. On one hand, retrospective detection relying on centralized services, e.g., cloud computing [5] [6] [7] and third-party authorities [8] [9] [10] [11] [12] often produce inconsistent and incomplete detection results. For example, as shown in Fig. 1, different third-party detection services share a very limited similarity in their detection results for one same IoT App (`Nest`[1]) in Google Play. Additionally, centralized detection services that release vulnerabilities in a batch typically incur longer delay from the time that a vulnerability is detected to the time that the vulnerability is publicly known by all stakeholders. On the other hand, although existing decentralization-based schemes (e.g., CloudAV [14] and Vigilante [15]) could produce more comprehensive retrospective detection, they fail to automatically provide detection feedback to consumers whenever a new vulnerability is uncovered on their installed devices. Further, they lack a well-designed incentive mechanism to attract more distributed detectors to participate in the detection.

In this paper, we propose a blockchain-based incentive platform, called SmartRetro, to address the above limitations. First, SmartRetro is a decentralized platform where detection results contributed by all peers, stored on public blockchain, are universally accessible to all stakeholders. As a result, comparing with existing centralized services,

[1]Nest is a mobile App to connect your IoT devices (e.g., thermostat) so that you can adjust them and get notifications for important events [13].

SmartRetro produces consistent and comprehensive detection results. Further, relying on smart contracts, SmartRetro rewards participating detectors instantaneously when their detection results are accepted and committed to the blockchain. Such automated rewarding design incentivizes more participation. Third, SmartRetro enables consumers to upload the system information of their installed IoT devices through smart contracts, called SmartRetro contracts, that are executed on blockchain. Whenever new detection results are committed on the blockchain, SmartRetro contracts will be triggered automatically, resulting in timely security notification to the corresponding consumers.

In summary, the major contributions of this paper are as follows.

- We present SmartRetro, a blockchain-based incentive platform that can attract distributed detectors to retrospectively detect IoT systems and automatically provide security notification to consumers whenever new vulnerabilities are detected.
- We introduce a novel incentive scheme in SmartRetro that achieves automated incentive allocation without relying on a centralized service.
- We perform sound security analysis of SmartRetro, implement a prototype of SmartRetro on Ethereum and evaluate the prototype extensively to demonstrate SmartRetro's feasibility.

The remainder of this paper is organized as follows. Section II mainly shows the background of this paper, especially blockchain technology, smart contracts. In Section III, we provide the problem statement and adversary model. SmartRetro overview and details are shown in Section IV and V, respectively. In Section VI and VII, we conduct security and theoretical analysis about our proposed SmartRetro platform. Performance evaluation is performed in Section VIII. We present the related work and conclude this paper in Section IX and X, respectively.

II. Background

This section describes the background of blockchain technology and smart contracts, which are fundamental building blocks in our platform.

The blockchain technology starts to attract massive attention since 2009, when crypto-currency systems, such as Bitcoin [16], that built upon blockchain gains popularity. Essentially, blockchain is a distributed ledger that leverages existing technologies, including peer-to-peer (P2P) networks for data transmission, consensus schemes for data consistency, and encryption algorithms for security verification. The blockchain consists of a series of blocks connected in order, each of which records transaction data that is organized in form of Merkel tree. The peer nodes (often referred to as miners) collect transactions in the network and package them into blocks. Then, they participate in a distributed consensus process to determine which block will be appended to the current blockchain. Several types of consensus algorithms have been proposed, including Proof-of-Work (PoW) [17], (Delegated) Proof-of-Stake (PoS) [18], and Byzantine fault tolerance (BFT) algorithm [19].

Decentralization is the central tenet of blockchain technology, in which transactions are verified and completed by distributed miners without any centralized or trusted authorities. Additionally, decentralization also makes a blockchain-powered system more secure and hard to be manipulated since transaction commitment typically required the consent from a majority of stakeholders.

Smart contract, which was first implemented on on Ethereum [20] blockchain, is what makes blockchain to potentially be more influential than the Internet, the largest distributed system people have ever invented so far. Smart contracts are essentially pieces of codes running all peers in the network, yielding a global-scale distributed super-computer that is unhackable and unstoppable. As a result, whenever a smart contract is triggered with valid conditions, it is guaranteed to unambiguously execute the code to compute the final result. When a smart contract is deployed on Ethereum, a certain number of fees, called *gas*, is required to be reserved in the contract because the execution of smart contract is not free. This prevents adversaries from deploying meaningless contracts (e.g., an empty while loop) on the network simply to exhaust resources on peers.

III. Problem Statement

This paper presents SmartRetro, a system that incentivizes distributed detectors to participate in retrospective detection for IoT systems, and meanwhile ensures fair reward allocation among detectors and automated security feedback to IoT consumers whose installed devices are detected with vulnerabilities. Before diving into design detail, we first clarify our design scope as well as the adversary model and assumptions of this paper.

A. Design Scope

Before releasing, IoT systems/devices may experience scanning or analysis to uncover possible security issues. However, pre-release scanning alone is not enough to secure the entire lift-time of IoT devices because additional vulnerabilities could be uncovered after the devices are widely deployed. This raises a serious threat to the healthy of IoT ecosystem. For instance, a lot of unsecured IoT devices are infested and recruited in a botnet, which has been used to launch large-scale DDoS attacks against the Internet. The scope of this paper is to design a decentralized platform to attract distributed detectors to perform retrospective security detection of IoT devices to uncover post-installation security issues.

B. Adversary Model

SmartRetro is a permissionless system that can be joined by any distributed detectors. In order to perform retrospective detection, detectors can either built its own analysis software, for instance, based on CloudAV [14] and Vigilante [15], or leverage existing detection services (e.g., QUIXXI [12] and NVISO [21]). We assume that a malicious detector could i)

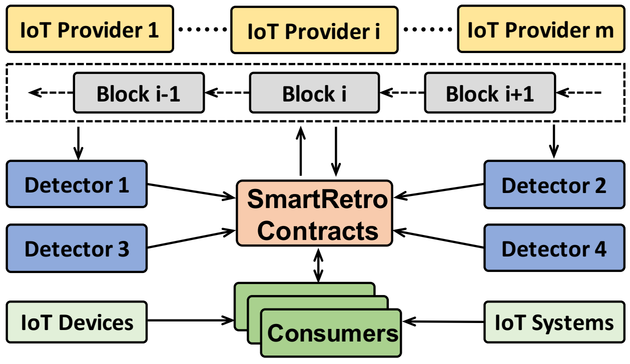

Fig. 2. The architecture of SmartRetro that leverages blockchain technology and smart contracts to incentivize distributed retrospective detections for IoT systems and send automated security feedback to consumers.

deliberately ignore certain vulnerabilities, ii) report incorrect detection results, or/and iii) plagiarize other detectors' detection reports in order to gain more rewards.

Since our underlying blockchain system is permissionless and relies on Proof-of-Work to reach consensus, we assume that no single participator could control the majority (>50%) of computation capability of entire detector network. This assumption is not a fundamental limitation of SmartRetro since SmartRetro can be built on any type of blockchains. Thus, security issues of the underlying blockchain system, such as routing attacks [22] and eclipse attacks [23], are out of this paper's scope.

IV. SmartRetro Overview

In this section, we present the overview of SmartRetro platform that can incentivize distributed detectors to retrospectively detect IoT systems and automatically provide security feedback to consumers.

A. Architecture

The SmartRetro is built upon on a blockchain system. As shown in Fig. 2, SmartRetro has three major participators:

- **IoT providers** release IoT systems with different versions. They are also responsible for constructing and maintaining the underlying blockchain system.
- **IoT detectors** identify post-installation vulnerabilities on IoT systems, and are motivated to submit their detection reports to earn regards.
- **IoT consumers** are the downstream users of IoT systems. Any consumers who would like to track the security status and obtain the security issue notification for their installed IoT devices can opt in to participate in SmartRetro.

SmartRetro designs called SmartRetro contracts to ensure the automation of incentives allocation to detectors and security issue notification to consumers. Two types of SmartRetro contracts are developed: one is used by consumers to announce the installation of IoT devices, the other one is used by distributed detectors to submit retrospective detection results.

B. Workflow and Challenges

The workflow of SmartRetro is as follows.

Step #1: System installation announcement. Consumers who opt in SmartRetro to track the security issues of their installed IoT systems need to first announce their system installations. This announcement is organized as a SmartRetro contract containing incentives/rewards for attracting detectors' participation on detecting vulnerabilities on their IoT systems.

Step #2: Distributed retrospective detection. IoT detectors who are motivated by rewards perform retrospective security analysis on IoT systems. Once any new issue are uncovered, they report their detection results, which will be recorded in the blockchain once these results are verified by the majority of IoT providers.

Step #3: Decentralized and automated incentives. When the detection results match with the installation announcements, SmartRetro contracts will be triggered automatically to allocate rewards to the corresponding detectors fairly and instantly, without relying on any trusted authorities.

Step #4: Automated security issue notification. SmartRetro contracts enable consumers to define callbacks such that a notification or feedback about the security issues of their IoT systems is automatically sent to them whenever their SmartRetro contracts are called and executed.

While designing SmartRetro, we explicitly address the following challenges.

Preventing spoofed installation announcement. A malicious consumer could launch spoofing attacks by announcing a forged system installation, so that the consumers who have announced the authentic system installation would have to pay rewards when a vulnerability on the system is detected.

Preventing forging and plagiarizing detection results. A malicious detector could try to forge or plagiarize other detectors' detection results in order to steal rewards.

Ensuring the correctness of detection results without a central authority. It is crucial to ensure that no invalid detection results could be published, especially in a decentralized and permissionless system where a wide variety of detectors may participate.

Guarantee fair reward allocation. A malicious consumer could try to refuse to give any pre-defined rewards to detectors even if detectors identify legitimate vulnerabilities. SmartRetro needs to guarantee fair reward allocation without relying any centralized entity.

V. SmartRetro Design Detail

In this section, we detailedly describe our proposed SmartRetro platform that achieves decentralized incentive allocation and automated security feedback. Concretely, consumers release an announcement through SmartRetro contracts when installing an IoT systems. The distributed detectors perform retrospective detections for IoT systems and report their detection results. When a detection result is accepted and recorded in the blockchain, the detector can automatically obtain incentives from consumers without relying on a centralized authority. The accepting detection results can also trigger SmartRetro contracts, resulting in an automated security feedback to consumers.

A. System Installation Announcement

SmartRetro enables consumers to perform system installation announcements (SIAs) for better tracking the security status of their IoT devices. In order to prevent a spoofing SIA, consumers' signatures have to be inserted into SIA. Meanwhile, SmartRetro enforces each consumer to submit an incentive in SIA, which has the following two purposes: i) it is a deposit that will not be refunded once the released SIA is detected to be spoofed; ii) it acts as an incentive for attracting detectors to retrospectively detect installed IoT systems.

SIA is released in the form of SmartRetro contracts that will be automatically performed once it is triggered. Eq. 1 shows the construction of SIA that contains six elements.

$$SIA = \{U_i, E_i, S_i, V_i, in, fbk, H_{SIA}, U_{Sign_i}\}, \quad (1)$$

where U_i, E_i, S_i and V_i are the unique identifications of consumers, IoT devices, IoT systems and system versions, respectively. in denotes the inserted incentives in SIA, which can attract more detectors' participation. fbk is used to send security feedback to consumers when some vulnerabilities have occurred in this IoT system (described in Section V-D). H_{SIA} (see Eq. 2) is the hash value of U_i, E_i, S_i, V_i and in.

$$H_{SIA} = H(U_i||E_i||S_i||V_i||in||fbk), \quad (2)$$

U_{Sign_i} is the signature of U_i, which is calculated with U_i's private key $K_{U_i}^{-1}$, as Eq. 3 shows. Then, SmartRetro contracts that records various SIAs will be broadcast to the network.

$$U_{Sign_i} = Sign_{K_{U_i}^{-1}}(H_{SIA}), \quad (3)$$

In SmartRetro, the blockchain is constructed and maintained by IoT providers who have more resources compared to IoT devices. When receiving released SIAs in SmartRetro contracts, IoT providers firstly verify H_{SIA} and check signature U_{Sign_i} using consumer U_i's public key K_{U_i}. Only passing the above verification, SIA of SmartRetro contracts can be recorded in the blockchain. Using the existing consensus mechanism (e.g., PoW [17], PoS [18] and PBFT [24]), SmartRetro enables IoT providers to ensure data consistency in the blockchain.

B. Distributed Retrospective Detection

SmartRetro can incentivize and attract more detectors to perform retrospective detection for IoT systems. More distributed detectors' participations can not only easily introduce a more comprehensive detection results but also lighten the detection burden of IoT devices that does not require to run a complex detection system. SmartRetro can prevent the forging or plagiarizing detection results by following two methods. i) A deposit is enforced to be submitted when each detector reports its detection results. If any error occurs in the detection report, this deposit will never be refunded. ii) SmartRetro enables each detector to firstly submit the hash value of detection results. In this case, no one can plagiarize others' results as the sovereignty of detection results has already been stated in the initial report.

Algorithm 1 The Initialization Algorithm for IR_i and FR_i

```
1: Require: $D_i$, $S_i$, $V_i$, $dpt_1$, $dpt_2$, $res$
2: Compute:
3:     $ID_{IR} = H(D_i||S_i||V_i||dpt_1)$
4:     $ID_{FR} = H(ID_{IR}||dpt_2||res)$
5:     $D_{Sign_{FR}} = Sign_{K_{D_i}^{-1}}(ID_{FR})$
6:     $D_{Sign_{IR}} = Sign_{K_{D_i}^{-1}}(H_{FR}||ID_{FR})$
7: Result 1:
8:     $FR_i = \{ID_{IR}, dpt_2, res_i, ID_{FR}, D_{Sign_{FR}}\}$
9: Compute:
10:    $H_{FR} = H(FR_i)$
11: Result 2:
12:    $IR_i = \{D_i, S_i, V_i, dpt_1, H_{FR}, ID_{IR}, D_{Sign_{IR}}\}$
13: Return: $IR_i$ and $FR_i$
```

The submission of a detection result is divided into two phases: the first is to submit an initial report that only records the hash value of detection results, and the second is to submit a final report that details the detection results. Algorithm 1 show D_i's initialization for the initial and final detection report, i.e., IR_i and FR_i.

Initial report submission. The initial detection report IR_i is used to determine the order of vulnerability discovery for distributed detectors, which is also associated with incentive allocations (detailed in Section V-C). The construction of IR_i for a detector D_i contains seven elements as follows:

$$IR_i = \{D_i, S_i, V_i, dpt_1, H_{FR}, ID_{IR}, D_{Sign_{IR}}\}, \quad (4)$$

where dpt_1 is the D_i's deposit inserted in IR_i, which can prevent D_i from submitting a forged IR_i. H_{FR} is the hash values of D_i's final detection report FR_i, which is used to prevent detectors from plagiarizing others' detection results. ID_{IR} is the identification of IR_i, which is computed by the following equation:

$$ID_{IR} = H(D_i||S_i||V_i||dpt_1). \quad (5)$$

$D_{Sign_{IR}}$ is the signature of detector D_i, which is calculated with D_i's private key $K_{D_i}^{-1}$ (see Eq. 6).

$$D_{Sign_{IR}} = Sign_{K_{D_i}^{-1}}(H_{FR}||ID_{IR}). \quad (6)$$

Therefore, other detectors can only learn D_i has discovered some vulnerabilities instead of knowing the concrete content of detection results. That can defend against a plagiarized detection reports. After initialization, IR_i is then delivered to the network and verified by IoT providers. Only the valid IR_i can be recorded in the blockchain (described later).

Final report submission. The final detection report FR_i records the detailed retrospective detection results for IoT systems. SmartRetro enables detectors to submit FR_i only when the corresponding IR_i has been stored in the blockchain.

D_i's final detection report FR_i must correspond with IR_i, whose construction is as Eq. 7 shows.

$$FR_i = \{ID_{IR}, dpt_2, res_i, ID_{FR}, D_{Sign_{FR}}\}, \tag{7}$$

where dpt_2 is D_i's deposit for submitting FR_i. This can prevent compromised detectors from spoofing attacks, where dpt_2 would not be refunded if FR_i is verified to be forged. res_i is D_i's detailed detection results for an IoT system. ID_{FR} is the identification of FR_i, which is the hash value of H_{IR}, dpt_2 and res_i. $D_{Sign_{FR}}$ is the D_i's signature that is calculated using $K_{D_i}^{-1}$ (see Eq. 8).

$$D_{Sign_{FR}} = Sign_{K_{D_i}^{-1}}(ID_{FR}). \tag{8}$$

The submitted final report FR_i should correspond to the initial report IR_i that has already recorded in the blockchain. Concretely, H_{FR} in IR_i should equal to the hash value of FR_i, i.e., $H_{FR} = H(FR_i)$. FR_i should be firstly verified before it is accepted and written in the blockchain (described in Section V-C).

C. Decentralized and Automated Incentives

SmartRetro provides the decentralized and automated incentives that is a vital function for attracting more detectors to participate in retrospective detection for IoT systems. To ensure the correct incentive allocation, detection results should be correctly verified and aggregated in the blockchain. SmartRetro employs the PoW consensus scheme for guaranteeing the detection results in each local blockchain stored by IoT providers are consistent.

Detection results aggregation is a critical premise for SmartRetro to correctly allocate incentives to the corresponding detectors. To achieve this aggregation, SmartRetro enables IoT providers to verify the submitted detection results and filter the invalid ones. This includes authenticity verification and correctness verification. When receiving detection results, IoT providers firstly perform authenticity verification by checking the report identifications (i.e., ID_{IR} and ID_{FR}) and the signatures (i.e., $D_{Sign_{IR}}$ and $D_{Sign_{FR}}$). Meanwhile, H_{FR} should also be verified and check whether $H_{FR} = H(FR_i)$ by recomputing the hash of FR_i. Algorithm 2 shows the process of authenticity verification for detection reports (IR_i and FR_i), which can defend against spoofing attacks launched by compromised detector(s). After authenticity verification, the correctness verification of detection results should be performed. In SmartRetro, IoT providers have richer resource (e.g., computation, storage and detection) compared to IoT devices, which can identify the detected vulnerabilities, i.e., res in detection report FR_i. This can be achieved by installing their own detection system (e.g., CloudAV [14] and Vigilante [15]) or using the existing third-party detection services (e.g., VirusTotal [9], Andrototal [11] and Ostorlab [8]). Only the detected vulnerabilities that have been verified by IoT providers can be eligible to be written in the blockchain. SmartRetro employs PoW-based consensus protocol to ensure the consistency of blockchain among multiple IoT providers. Namely, if a detected vulnerability has been verified and accepted by the majority of IoT providers, it will be recorded in the blockchain, permanently. Therefore, SmartRetro enables the detection result aggregation in the blockchain by performing authenticity and correctness verification and using PoW-based consensus scheme.

Algorithm 2 Authenticity Verification for Detection Reports.

```
1: function AUTHENTICITY VERIFICATION FOR $IR_i$ ( )
2: Require: $IR_i$ and $K_{D_i}$
3:     Compute: $ID'_{IR} = H(D_i||S_i||V_i||dpt_1)$
4:     if ($ID'_{IR}$ == $ID_{IR}$) && ($CheckSign_{K_{D_i}}(D_{Sign_{IR}})$) then
5:         Accept $IR_i$ and try to record it in the blockchain;
6:     else
7:         Drop $IR_i$ and break;
8:     end if
9: end function
10: function AUTHENTICITY VERIFICATION FOR $FR_i$ ( )
11: Require: $IR_i$, $FR_i$ and $K_{D_i}$
12:     Compute: $ID'_{FR} = H(ID_{IR}||dpt_2||res)$
13:     if ($ID'_{FR}$ == $ID_{FR}$) && ($CheckSign_{K_{D_i}}(D_{Sign_{FR}})$) then
14:         if $H_{FR}$ == H($FR_i$) then
15:             $FR_i$ is authentic;
16:         end if
17:     else
18:         Drop $FR_i$;
19:     end if
20: end function
```

Incentives allocation. SmartRetro enables each system installation announcement, i.e., SIA, to be released in the form of SmartRetro contracts. When a detection result is recorded in the blockchain, some SmartRetro contracts that record the corresponding SIA can be triggered, which can automatically allocate incentives to the detector who submitted the detection result. Note that the allocated incentives are in of SIA, which may be submitted by multiple consumers. In this case, the detector can obtain more than one incentive once finding a vulnerability. Besides, SmartRetro ensures that the entire incentive allocation is totally decentralized due to leveraging blockchain technology. Therefore, SmartRetro can prevent compromised consumers from repudiating incentives allocation because SmartRetro contracts can make the incentives in in SIA automatically allocated.

D. Automated Security Feedback

Security feedback is automatically provided to the related consumers once a vulnerability is newly discovered in SmartRetro. This is because SmartRetro makes consumers insert their feedback requirement, i.e., fdk, in SmartRetro contracts when releasing an SIA. Security feedback is accompanied with the incentive allocation, which all relies on the automated execution of SmartRetro contracts. Concretely, when the detection results are aggregated and recorded in the blockchain, the triggered SmartRetro contracts can not only introduce the automated incentive allocation, but also provide automated security feedback to consumers.

VI. Security Analysis

In this section, we perform the security analysis for SmartRetro that can defend against the attacks from vulnerable

IoT systems and the interferences from malicious consumers, detectors and IoT providers.

Security against the newly uncovered vulnerabilities. Unlike centralized detection services whose detection coverage is determined by the central entity's capability, SmartRetro's detection capability is determined by all distributed detectors. With properly designed incentive mechanism, SmartRetro can, arguably, attract more diverse detectors such that the combined detection capacity from all peer detectors outweigh any single centralized services. Besides, on SmartRetro, consumers can track the security status of their installed IoT systems by retrospective detection more timely, rather than waiting until the release from centralized services.

Security against the malicious consumers. In SmartRetro, malicious consumers could launch spoofing attacks to disrupt our incentive allocation through the following ways: i) imitating other consumers to release a spoofed SIA; ii) refusing to pay rewards to detectors. Both types of spoofing attacks can cause financial loss to benign consumers and detectors. SmartRetro mitigate the above attacks by enabling each consumer to submit incentives as a deposit, i.e., in, when releasing an SIA. As a result, once an SIA is detected to be spoofed, a compromised consumer suffers from financial losses because in will not be refunded. Meanwhile, SmartRetro enables consumers to add a signature U_{Sign_i} in a released SIA to identify the authenticity of an SIA.

Security against the compromised IoT detectors. Compromised IoT detectors can try to forge or plagiarize the detection results submitted by other benign detectors so as to gain more rewards. SmartRetro prevents this attack by enabling each detector to firstly submit the initial report IR_i that only records the hash value, i.e., H_{FR} of its detection results. This prevents the detection results from being plagiarized by others because: i) the detection results can not be learned from IR_i; ii) IR_i can be used to declare the order that one vulnerability is identified even though the detection results can be publicly viewed in FR_i. Meanwhile, IoT detectors are required to submit a deposit (i.e., dpt_1 and dpt_2) when reporting detection results. Later, our authenticity and correctness verification mechanism can filter the forged or plagiarized detection results, resulting in financial losses from those malicious detectors.

Security against the misbehaved IoT providers. The submitted detection results are stored in the blockchain that is maintained by distributed IoT providers. In SmartRetro, one compromised IoT provider could impact our reward allocation by selectively packaging detection results, such as deliberately ignore some valid detection results and only including detection results from colluding detectors. SmartRetro addresses the above security threat by leveraging PoW-based consensus scheme. In this case, the detection results contained in the final blockchain are statisically determined by the majority of IoT providers rather than a single one. Thus, any minority set of compromised IoT providers have minimal impact on SmartRetro's reward allocation.

VII. Theoretical Analysis

In this section, we perform theoretical analysis of the incentive mechanism designed in SmartRetro platform. Note that since the detection reports (IR_i and FR_i) are all submitted in the form of SmartRetro contracts, certain *gas* fees are required to deploy these smart contracts on blockchain. Therefore, the net benefit of an IoT detector equals its received reward for retrospective detection minus the costs for submitting its detection results. Consider that the number of IoT systems is m, each IoT system has a version number up to l, and the total number of IoT devices is n.

Incentives for retrospective detections. The rewards allocated to detectors in SmartRetro are mainly determined by the inserted in in SIA. Thus, the total incentives (denoted by In_{total}) of IoT ecosystem are calculated as follows:

$$In_{total} = \sum_{i=1}^{m}\sum_{j=1}^{l} in \cdot n \cdot P_{ij}, \tag{9}$$

where P_{ij} is the probability that IoT device E_k $(1 \leq k \leq n)$ have installed an IoT system S_i $(1 \leq i \leq m)$ with version V_j $(1 \leq i \leq l)$. We use $\{S_i, V_j\}$ to denote the IoT system and version pair. We define DC_i as the detection capability of detector D_i. Namely, the probability that D_i can uncover a new vulnerability is DC_i. Therefore, the proportion of detection capability (denoted by DCP_i) can be calculated from Eq. 10, where q is the number of detectors.

$$DCP_i = \frac{DC_i}{\sum_{j=1}^{q} DC_j}. \tag{10}$$

Thus, the detector D_i can gain the following incentives (In_{D_i} in Eq.), which is positively correlated with its detection ability. We can learn more inserted incentives in or the larger detection capability proportion DCP_i can help detectors gain more incentives. Especially, if there is only one detector in SmartRetro, the allocated incentives grow linearly with its detection capability.

$$In_{D_i} = In_{total} \cdot DCP_i. \tag{11}$$

Cost for submitting detection results. The cost of detector D_i is the *gas* that is deposited in SmartRetro contracts of IR_i and FR_i. We assume n_1 and n_2 are the number of D_i's submitted IR_i and FR_i, respectively. Therefore, the cost (denoted by C_i) is calculated as follows.

$$C_i = gas_1 \cdot n_1 + gas_2 \cdot n_2, \tag{12}$$

where gas_1 and gas_2 are the consumed *gas* that is required to execute SmartRetro contracts of IR_i and FR_i, respectively. Thus, in SmartRetro, the net reward of detector D_i, denoted by B_{D_i}, is computed according to Eq. 13.

$$B_{D_i} = In_{D_i} - C_i. \tag{13}$$

In SmartRetro, any fake detection report can also cause D_i's financial loss. We define the fake report rate (denoted by ϑ_i) as the probability that the submitted detection reports (i.e., IR_i

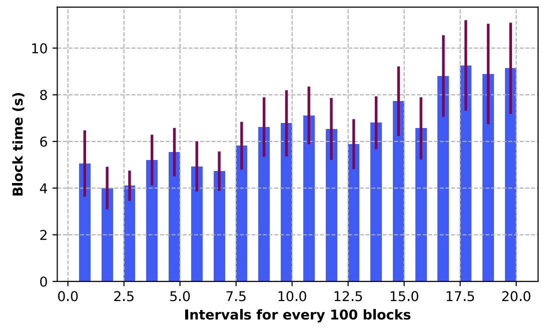

Fig. 3. The block time of SmartRetro platform.

and FR_i) are forged or plagiarizing. Thus, the loss of D_i's submitting fake report is as Eq. 14 shows.

$$L_{D_i} = (dpt_1 \cdot n_1 + dpt_2 \cdot n_2) \cdot \vartheta_i. \tag{14}$$

Therefore, the net reward of D_i ia adjusted as Eq. 15 if the loss of faking report is taken into consideration.

$$B_{D_i} = In_{D_i} - C_i - L_{D_i}. \tag{15}$$

We can learn that improving detection capability and enhancing the credibility of detection results enable detectors to gain more rewards. This essentially regulate detectors' behavior and ensures the healthiness of our platform.

VIII. Performance Evaluation

In this section, we evaluate SmartRetro platform with the following metrics: i) the incentives that detectors can gain from retrospective detection for IoT systems; ii) the cost that detectors pay when submitting detection results; iii) the punishment when misbehaved detectors declare fake reports; iv) the payout that consumers provide to incentivize decentralized retrospective detections and automated security feedbacks.

We implement our proposed SmartRetro platform on a private test blockchain that is built using the current Ethereum system [25]. This test blockchain runs on Dell PowerEdge R710 with Ubuntu 14.04, Inter(R) Xeon(R), CPU X5560 @ 2.80GHz and 35G memory. The cryptocurrency `ether` is employed to evaluate the above metrics of SmartRetro. We use solidity language [26] of 225 lines to implement SmartRetro contracts for releasing SIAs and submitting detection reports (i.e., IR_i and FR_i). Fig. 3 shows the block time of SmartRetro platform, where we can learn the average block time is 6.5 seconds. In our SmartRetro implementation, there are 100 consumers, each of which has 5 IoT devices, and 5 IoT systems, each of which has 2 versions. Besides, each version of one IoT system has up to 10 potential vulnerabilities. Therefore, up to 50 vulnerabilities can occur in each consumer's IoT devices. For an SIA, we set the value of the inserted incentives, i.e., in, is 0.001∼0.003, 0.002∼0.004, 0.003∼0.005 ether. For the submission of IR_i or FR_i, 5 ethers should be required as a deposit, i.e., $dpt_1 = dpt_2 = 5$ ethers, in our implemented SmartRetro. 0.001∼0.003, 0.002∼0.004, 0.003∼0.005 ether. For the submission of IR_i or FR_i, 5 ethers should be required as a deposit, i.e., $dpt_1 = dpt_2 = 5$ ethers, in our implemented SmartRetro. Besides, some *gas* should be consumed to execute SmartRetro contracts that record SIA, IR_i and FR_i. In the implementation, 404502, 356890 and 541324 *gas* are required for running SmartRetro contracts for SIA, IR_i and FR_i, respectively. The incentives of detectors are derived from retrospective detections for IoT systems. By leveraging SmartRetro contracts, the incentives in inserted in SIAs will be automatically transformed to the detector D_i once the final detection report FR_i is written in the blockchain.

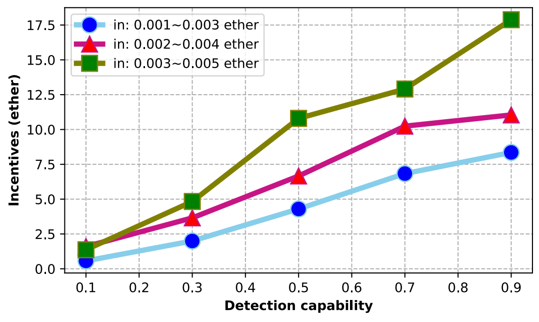

Fig. 4. Incentives *vs.* Detection Capability (DC).

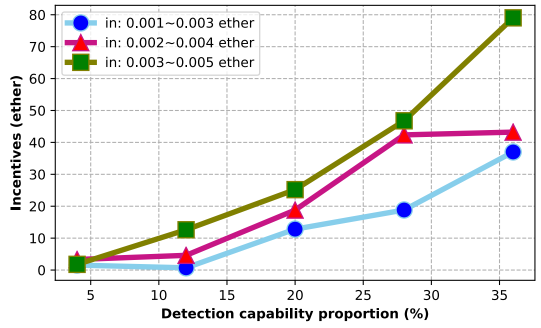

Fig. 5. Incentives *vs.* DC Proportion.

A. Incentives for retrospective detection

We firstly evaluate the obtained incentives for retrospective detection when there is only one detector in SmartRetro. Fig. 4 shows this relationship between allocated incentives and detection capability (DC) for different inserted incentives, i.e., in in SIAs. We can learn the obtained incentives increase nearly linearly with detector's DC. This is the same as our theoretical analysis for allocated incentives of retrospective detection (see Section VII). Meanwhile, more inserted incentives in can enables detectors gain more incentives for the same detection capability. For example, 0.02 ∼ 0.04 and 0.03 ∼ 0.05 ether as an incentive in each SIA can make detectors gain 1.5 and 2 times incentives compared to the case of 0.01 ∼ 0.03 ether. Then, we evaluate the incentives of multiple detectors that simultaneously run in SmartRetro. Fig. 6 shows the incentive variation for different DC proportions. We can learn the detectors with higher DC proportion can gain more incentives. We define incentives proportion as the ratio of allocated incentives to the total, which has been be evaluated in Fig. 5. We can learn the amount of inserted incentives in in SIAs have no effect on incentives proportion.

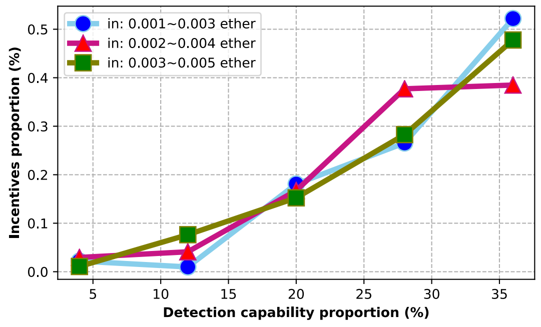

Fig. 6. Incentives Proportion *vs.* DC Proportion.

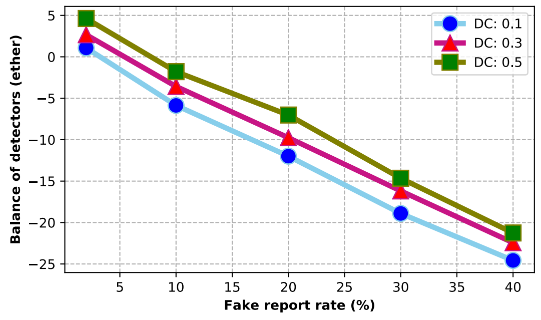

Fig. 7. The balance of detectors for different fake report rates.

B. Punishment for faking detection reports

As described in Section V-B, the detectors will also insert deposits (i.e., dpt_1 and dpt_2) in the detection reports (i.e., IR_i and FR_i) when reporting their detection results. In particular, these deposits will not to be refunded if the submitted IR_i and FR_i are detected to be forged or plagiarized. Therefore, SmartRetro enables detectors to be punished when they fake detection reports for IoT systems.

We adjust the value of fake report rate ϑ_i to evaluate the balance of detectors. In SmartRetro, we set 3 ethers is required as the deposit for detectors to submit IR_i or FR_i. As Fig. 7 shows, with the increasing ϑ_i, the balance of detectors will become less and less. This is because the larger value of ϑ_i can cause more punishments for an IoT detector. In particular, this balance can be the reach break-even point, for example, at $\vartheta_i = 5\%$ for the DC of 0.3. Meanwhile, the detector with higher DC will have a higher balance for the same ϑ_i. This is because the higher DC can help detectors to gain more incentives for performing retrospective detections under the same punishments.

C. Payout of consumers

The incentives allocated to detectors in SmartRetro derive from the submitted in when consumers release SIAs. From the perspective of consumers, their inserted incentives in SIA can help to keep a track of security status of IoT systems by attracting more detectors' participation, and automatically obtain security feedback.

We evaluate the payout of consumers when the number of discovered vulnerabilities changes. In our experiment settings, up to 50 vulnerabilities can occur in one consumer's IoT devices. Fig. 8 shows the experimental result, where consumers will provide more payouts if more vulnerabilities are identified in their IoT systems. Meanwhile, when more in is required to be inserted in SIA, the consumers will have more payouts.

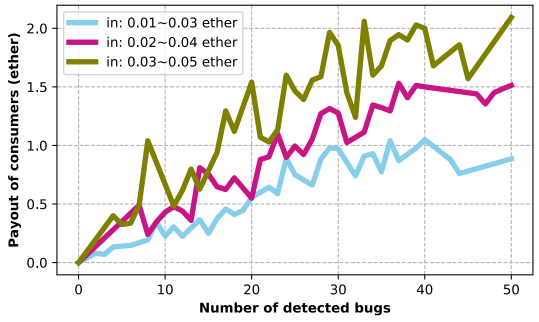

Fig. 8. The payout of consumers *vs.* the number of detected vulnerabilities.

IX. Related Work

In this section, we discuss the related work about retrospective detection. In particular, three main areas are included: retrospective detection, incentive schemes, and blockchain-based security solutions.

Retrospective detection for IoT systems. Retrospective detection can help to identify the newly discovered vulnerabilities and track the security status for IoT systems. CloudAV [14] introduces a novel solution for malware detection by enabling distributed endhosts to provide N-version protection as an in-cloud network service. CloudEyes [6] is presented as a cloud-based anti-malware system for providing efficient and reliable security detections for IoT devices, which can provide retrospective orientations for abnormal data fragments. Vigilante [15] is an end-to-end security mechanism, which uses a collaborative worm detection to detect and contain the Internet worms. Using a self-certifying alert (SCA), the detectors in Vigilante can retrospectively detect the worms in the Internet. Liu et al. [5] propose a method of retrospective detections for malware attacks by cloud computing, which is mainly based on Portable Executable (PE) format file relationships. Ke et al. [27] propose PPV that can be used to perform source and path verification for the secure communication between IoT devices. However, these methods either are based on centralized detections that fail to obtain a comprehensive result, or can not provide an automated security feedback.

Incentive schemes in the blockchain. Stephanos et al., present IKP platform that can incentivize certificate authorities (CAs) to behave normally and detectors to report misbehaved certificates [28]. Andrychowicz et al., illustrate the Bitcoin system can be used as incentives to make multiparty computation protocols (MPCs) more secure [29]. Kumaresan et al., try to build a model to incentivize correct computations (e.g., verifiable computation, secure computation with restricted leakage, fair secure computation, and noninteractive bounties) [30]. However, these mechanisms can not fully be adapted to the area of IoT retrospective detections duo to the limited resource of IoT devices.

Blockchain-based security enhancements. Boudguiga et al., introduce the blockchain-based infrastructure for securing the update of IoT devices, which relies on some distributed nodes for validating and checking the innocuousness of released IoT systems before they are installed by IoT devices [31]. Ali et al., propose Blockstack, a blockchain-based global naming and storage system [32]. It separates control and data plane and introduces more secure verification for naming and storage without modifying the underlying blockchain. Rodrigues et al., propose a blockchain-based solution to mitigate DDoS attacks, which can record the occurrence of attacks through smart contracts and share detections among domains [33]. Garman et al., employ the blockchain technology to build a novel anonymous credential that no longer relies on a trusted credential issuer [34]. They provide a proof for the security of the proposed decentralized anonymous credential system that enables strong privacy guarantees.

X. Conclusion

In this paper, we propose a blockchain-based incentive platform, called SmartRetro, to motivate retrospective detections for IoT systems. SmartRetro supports and incentivizes distributed detections by attracting more detectors' participation, which is conducive to construct a more comprehensive report. Based on the detection contribution, each detector can automatically gain incentives that eliminate the need of a decentralized authority. Meanwhile, SmartRetro can keep a track of security status of IoT systems and identify which IoT devices have been affected. In particular, an automated security feedback can be delivered to the corresponding consumers when a new vulnerability is discovered. We make the security analysis that demonstrates SmartRetro can benefit to build a higher security guarantee under retrospective detections for IoT systems. We implement SmartRetro in Ethereum and evaluate the performance of SmartRetro. The experimental results show the detectors and consumers in SmartRetro can be beneficial financially and safely, respectively.

Acknowledgments

This work was supported by National Key R&D Program of China (2018YFB0803405), the National Natural Foundation of China (61472212 and 61572278), EU Marie Curie Actions CR-OWN (FP7-PEOPLE-2013-IRSES-610524), and the R&D Program of Shenzhen (JCYJ20170307153259323).

References

[1] Statista. Internet of things (iot) connected devices installed base worldwide from 2015 to 2025 (in billions). https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide,2018.

[2] Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J Alex Halderman, Luca Invernizzi, Michalis Kallitsis, et al. Understanding the mirai botnet. In *USENIX Security Symposium*, 2017.

[3] Scott Hilton. Dyn analysis summary of friday october 21 attack. http://hub.dyn.com,2016.

[4] Octave klaba. Octave klaba twitter. https://twitter.com/olesovhcom/status/778830571677978624.

[5] Shun-Te Liu and Yi-Ming Chen. Retrospective detection of malware attacks by cloud computing. *International Journal of Information Technology, Communications and Convergence*, 1(3):280–296, 2011.

[6] Hao Sun, Xiaofeng Wang, Rajkumar Buyya, and Jinshu Su. Cloudeyes: Cloud-based malware detection with reversible sketch for resource-constrained internet of things (iot) devices. *Software: Practice and Experience*, 47(3):421–441, 2017.

[7] Safaa Salam Hatem, Mahmoud M El-Khouly, et al. Malware detection in cloud computing. *Int J Adv Comput Sci Appl*, 5(4):187–192, 2014.

[8] Ostorlab. Secure your mobile app. https://ostorlab.co,2018.

[9] VirusTotal. https://www.virustotal.com,2018.

[10] NVISO. https://www.nviso.be,2018.

[11] Andrototal. Androltotal is a free service to scan suspicious apks against multiple mobile antivirus apps. http://andrototal.org,2018.

[12] QUIXXI. How secure is your mobile app? https://quixxi.com,2018.

[13] Nest Labs. Nest. https://play.google.com/store/apps/details?id=com.nest.android, 2018.

[14] Jon Oberheide, Evan Cooke, and Farnam Jahanian. Cloudav: N-version antivirus in the network cloud. In *USENIX Security Symposium*, pages 91–106, 2008.

[15] Manuel Costa, Jon Crowcroft, Miguel Castro, Antony Rowstron, Lidong Zhou, Lintao Zhang, and Paul Barham. Vigilante: End-to-end containment of internet worms. In *ACM SIGOPS Operating Systems Review*, volume 39, pages 133–147. ACM, 2005.

[16] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.

[17] Bitcoinwiki. Proof of work. https://en.bitcoin.it/wiki/Proof_of_work, 2018.

[18] Wikipedia. Proof-of-stake. https://en.wikipedia.org/wiki/Proof-of-stake, 2018.

[19] Wikipedia. Byzantine fault tolerance. https://en.wikipedia.org/wiki/Byzantine_fault_tolerance,2018.

[20] Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper*, 151:1–32, 2014.

[21] NVISO. https://www.nviso.be,2018.

[22] Bgpstream. https://bgpstream.com,2018.

[23] Atul Singh, Tsuen-wan johnny Ngan, Peter Druschel, and Dan S. Wallach. Eclipse attacks on overlay networks: Threats and defenses. In *IEEE INFOCOM*, 2006.

[24] Miguel Castro and Barbara Liskov. Practical byzantine fault tolerance and proactive recovery. *ACM Transactions on Computer Systems (TOCS)*, 20(4):398–461, 2002.

[25] Official go implementation of the ethereum protocol. https://github.com/ethereum/go-ethereum, 2018.

[26] The solidity contract-oriented programming language. https://github.com/ethereum/solidity, 2018.

[27] Bo Wu, Ke Xu, Qi Li, Zhuotao Liu, Yih-Chun Hu, J. Reed Martin, Meng Shen, and Fan Yang. Enabling efficient source and path verification via probabilistic packet marking. In *IEEE/ACM IWQoS*, 2018.

[28] Stephanos Matsumoto and Raphael M Reischuk. Ikp: Turning a pki around with decentralized automated incentives. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pages 410–426. IEEE, 2017.

[29] Marcin Andrychowicz, Stefan Dziembowski, Daniel Malinowski, and Lukasz Mazurek. Secure multiparty computations on bitcoin. In *Security and Privacy (SP), 2014 IEEE Symposium on*, pages 443–458. IEEE, 2014.

[30] Ranjit Kumaresan and Iddo Bentov. How to use bitcoin to incentivize correct computations. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 30–41. ACM, 2014.

[31] Aymen Boudguiga, Nabil Bouzerna, Louis Granboulan, Alexis Olivereau, Flavien Quesnel, Anthony Roger, and Renaud Sirdey. Towards better availability and accountability for iot updates by means of a blockchain. In *IEEE EuroS&PW*, 2017.

[32] Muneeb Ali, Jude Nelson, Ryan Shea, and Freedman Michael J. Blockstack: A global naming and storage system secured by blockchains. In *2016 USENIX Annual Technical Conference (USENIX ATC16)*, 2016.

[33] Bruno Rodrigues, Thomas Bocek, Andri Lareida, David Hausheer, Sina Rafati, and Burkhard Stiller. A blockchain-based architecture for collaborative ddos mitigation with smart contracts. In *Security of Networks and Services in an All-Connected World*, pages 16–29, Cham, 2017. Springer International Publishing.

[34] Christina Garman, Matthew Green, and Ian Miers. Decentralized anonymous credentials. In *NDSS*, 2014.