IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received July 21, 2018, accepted September 22, 2018, date of publication October 4, 2018, date of current version October 29, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2873758

Trusted Cooperation Among Virtual Base Stations in C-RAN

FENGYU TIAN[1,2], (Member, IEEE), ZHENG YAN[1,3], (Senior Member, IEEE), XUEQIN LIANG[1,3], AND PENG ZHANG [1]

[1]State Key Laboratory on Integrated Services Networks, School of Cyber Engineering, Xidian University, Xi'an 710071, China
[2]Key Lab of Information Network Security, Ministry of Public Security, Shanghai 201204, China
[3]Department of Communications and Networking, Aalto University, 02150 Espoo, Finland

Corresponding author: Zheng Yan (zhengyan.pz@gmail.com)

This work was supported in part by NSFC under Grants 61672410, U1536202, and 61802293, in part by the Project supported by the Natural Science Basic Research Plan in Shaanxi Province of China under Grant 2016ZDJC-06, in part by the Academy of Finland under Grant 308087, in part by the Key Lab of Information Network Security, Ministry of Public Security, under Grant C18614, in part by the National Postdoctoral Program for Innovative Talents under Grant BX20180238, in part by the Project funded by China Postdoctoral Science Foundation under Grant 2018M633461, in part by the National Key Research and Development Program of China under Grant 2016YFB0800704, and in part by the 111 Project under Grants B16037 and B08038.

ABSTRACT Cloud radio access network (C-RAN) has become one of the hot-spot research directions in both industry and academia to facilitate 5G development. Its architecture is mainly divided into three parts: 1) Virtual base station (VBS) pool including a number of base band units that apply for real-time cloud computing technology to carry out digital processing tasks; 2) Remote radio heads (RRH) that are used to collect the wireless signals from all wireless devices; 3) Front-haul network connecting the RRH to the VBS pool. The information of base stations (BS) resides in a centralized VBS pool, multiple operators can allow their VBS pools to exchange control data and cooperate with each other to provide high-quality 5G services. In this process, trustworthy cooperation and platform-trust authentication among multiple mobile operators are required to realize high networking performance and secure 5G environment. However, this topic has been scarcely studies and is still in its infancy. In this paper, we first propose a C-RAN inter-operator cooperation scheme (IOCS) to support the cooperation of multiple operators in C-RAN and allow them to share resources in a trustworthy and secure way based on trusted computing platform. An access trust management pool including VBS trust managers is set up in IOCS to administrate an operator's VBS pool. It allows an end user to consume other operators' network resources in a trustworthy manner when its own operator's network cannot satisfy the quality of service requirements. Furthermore, we design a trusted cooperative platform based on OpenStack to implement IOCS. Performance evaluation and simulation results illustrate the high operation efficiency in IOCS comparing with other schemes. Game theoretical analysis shows the condition of inter-operator cooperation by taking operator trust into concern.

INDEX TERMS Cloud radio access network (C-RAN), trusted computing, trust management, inter-operator cooperation, 5G.

I. INTRODUCTION

The 4th generation wireless networks are widely used currently; however, they neither offer a big data bandwidth and infinite network capabilities, nor satisfy the increasing demands of both mobile users and mobile network operators. For example, as the capital expenditure (CAPEX) or the operating expenditure (OPEX) cost of a network architecture reaches a high level, more and more mobile network operators focus on the percentage of Total Cost of Ownership (TCO) in the whole network architecture, such as air conditioning costs, base station selection costs, etc. [1], [2].

Cloud Radio Access Network (C-RAN) was proposed to overcome traditional problems in Radio Access Networks (RAN) and satisfy users' requirements on the next generation mobile communication networks and wireless systems (5G) [4]. As a new centralized architecture based on Software Defined Radio (SDR) networks [3], C-RAN is playing a very important role in a 5G environment (i.e., macro cell, micro cell, pico cell and indoor coverage) due to its mobility, flexibility, easy extension and low TCO. Generally speaking, a C-RAN architecture consists of three parts: i) a centralized Virtual Base Station (VBS) pool, composed of a number of Base Band Units (BBU), which is applied for real-time cloud computing technologies to carry out digital processing tasks; ii) Remote Radio Heads (RRH) that collect wireless signals from all wireless devices; iii) a front-haul network connecting the RRH to the VBS.

C-RAN holds a number of advantages. First, multiple VBSs are managed together as a whole, connected to many sector antennas that cover a wide area and share signals to minimize the computing-resource management cost. Second, the cooperative processing among base stations achieves the reconstruction of spectrum resources. Third, real-time cloud computing technology can dynamically implement load balancing, process aggregation and reduce inter-cell interference phenomenon. Multiple operators' VBS pools can collaborate with each other through this new C-RAN architecture to provide affordable, effective and highly efficient 5G services. For example, such a design is expected when an operator's network services can be provided by other operators' VBS pools for satisfying the quality requirements of 5G network services when its own BS is busy or overloaded. In this situation, trustworthy cooperation among operators becomes necessary.

A. MOTIVATIONS

When applying the excellent advantages of C-RAN to meet the service demands of 5G, the security and trust issues in C-RAN is an important and interesting topic. In different logical planes (i.e., physical plane, control plane and service plane) of C-RAN, various kinds of attacks [5] exist. For example, in the physical plane, due to the characteristics of C-RAN, it is especially vulnerable to such attacks as eavesdropping attack, jamming attack, impersonation attack, primary user emulation attack (PUEA) and wireless channel threats. The primary attacks in the control plane are Media Access Control (MAC) layer protocol attack and common control channel threats (e.g., MAC spoofing, extended Denial of Service (DoS) attack, jamming attack, etc.). The service plane is the most vulnerable layer among the three planes due to its importance. Such attacks as Transport and Application Layer Protocol Attack, cloud computing security threats, virtualization attack, user privacy threat and other security threats [6] could happen. There are many understudied issues worthy of research and exploration in the field of C-RAN security.

The infrastructures of C-RAN are established by multiple operators. Each of these infrastructures can serve its own users to satisfy their needs based on the referred architecture (centralized or partially centralized). Therefore, one open issue is how to fully employ the existing network resources offered by multiple operators and guarantee the reliability and security in such a cooperation environment under the C-RAN architecture. Moreover, we should ensure different operators' subscribers to flexibly consume available resources provided by multiple operators in a trusted way. In other words, we should ensure that the VBS pools in different operators and the entire cooperation process among them are reliable and secure. However, how to establish a trustworthy cooperation environment in C-RAN is rarely studied by current literature. Most schemes lack deep research on their applicability [7]–[9]. Therefore, maximizing resource utilization, gaining extra profits and ensuring high-quality networking services while ensuring trustworthy cooperation among operators is a practical issue to be solved.

B. MAIN CONTRIBUTIONS

In this paper, we propose an Inter Operator Cooperation Scheme (IOCS) to support multiple operators to share their resources in a trustworthy and secure way based on Trusted Computing Platform (TCP) technology in the infrastructure of C-RAN [10]–[12]. We also demonstrate how our IOCS can be realized based on OpenStack and virtual Trusted Platform Module (vTPM).

In our scheme, an access trust management pool that contains a number of VBS trust managers (TMs) is established, each of which manages the VBS pool (e.g., spectrum resources, etc.) of one operator. These VBS TMs can issue tokens and provide a number of user configurable Application Programming Interfaces (APIs) to establish a trust-maintenance mechanism among multiple operators. These tokens represent the identity of the VBS Pool, which can help the VBS Pool rent network resources and balance service loads across multiple operators' VBS pools. Using these APIs, the operators can implement some cooperative strategies to control trust maintenance for achieving trusted cooperation across multiple cellular networks owned by different mobile operators [11]. In particular, one operator can monitor the configuration information of the VBS pool in real time to verify and ensure that another operator's VBS pool is working as its TM expectation by embedding a trust cooperation strategy into the VBS TM.

Once a VBS pool lacks of networking resources, its own VBS trust manager can send a rental request to other VBS TMs. The other VBS TMs can challenge the trustworthiness of a remote VBS trust manager and make a corresponding response based on an established cooperation strategy by considering essential factors (e.g., renters' current status of resource consumption, renters' free resources, rental requests, priority of resource rental, the agreement with the requester, estimated resource demands in a requested period). Next, after comparing the offers from other operators' TMs, the requester chooses the most suitable one according to its own selection strategy (e.g., reputation of a renter) [12]. Finally, the requester locates the selected VBS Pool to start inter operator cooperation.

We further implement the proposed scheme by developing a trusted cooperative platform based on OpenStack to validate its performance. Performance evaluation and simulation results show that IOCS has highly efficient operation and correct trust attestation. Additionally, we analyze whether the operators would like to collaborate with each other by applying game theory. Specifically, the contributions of this paper can be summarized as below:

- We propose the IOCS to support trustworthy cooperation among operators based on TCP technology in C-RAN. We specify its detailed system structure, trust attestation and cooperation procedure and document it to be with flexibility, secure and trustworthy cooperation, generality and cost economy.
- We implement the IOCS through a trusted cooperative platform based on OpenStack and virtualization technology. The performance test results further show that our scheme offers sound availability, efficiency and effectiveness.
- We apply game theory to analyze the condition of inter-operator cooperation by taking the operator trust into concern.

The rest of the paper is organized as follows. Section II gives a brief overview of related work. In Section III, we introduce the system structure of the IOCS and its trust attestation and cooperation procedure. Section IV presents IOCS implementation, followed by security analysis, performance evaluation and analysis on operator cooperation with game theory. Finally, we conclude our work in the last section.

II. BACKGROUND AND RELATED WORK

In this section, we introduce the technical background of our work and review related work. We first briefly overview security issues and the requirements in C-RAN service plane. Then, we introduce the basic of trusted computing and computing platform integrity measurement, followed by a review on platform attestation and authentication. Finally, we introduce the game theory that will be applied to analyze the possibility and condition of multiple operator cooperation in Section IV.

A. SECURITY ISSUES AND SECURITY REQUIREMENTS OF C-RAN SERVICE PLANE

As one of the key technologies of 5G, C-RAN is expected to solve the challenges faced by traditional wireless access networks [13]. An increasing number of operators start to investigate the C-RAN application scenarios (e.g., macro cell, micro cell, pico cell, indoor coverage system, etc.) and attempt to apply it as an alternative of current cellular networks to support the growing end-user needs in a cost-effective manner.

In 2015, Wu *et al.* [5] proposed a novel logic structure of C-RAN, which includes physical plane, control plane, and service plane and refines the functional requirements of each layer, service cloud, service-oriented architecture and personal resource scheduling and management. With the development of C-RAN technology, the security of C-RAN architecture is more required than before. However, the main efforts of current C-RAN study only focus energy efficiency, resource allocation, and QoS enhancement. In this part, we leave along the threats and attacks in all the logic structure of C-RAN while mainly concentrate on the service plane and existing solutions to resist them [14]–[17].

The service plane of C-RAN is a service-oriented cloud platform, which directly interacts with the end-users and service providers. The interaction between users and service providers is not transparent. The users just attach importance to whether their requirements are satisfied while do not care who provides that the services. In C-RAN, the service (e.g., real-time data processing, information transmission of terminal devices and special channels, and dynamic traffic capacity allocation, etc.) is provided by the VBS pool. The service plane should prevent the VBS pool from invasion and provide an essential security module for Fine-Grained Access Control (FGAC) and identity authentication, etc. Current security research on the service layer can be divided into the following categories with respect to different kinds of attacks.

1) TRANSPORT AND APPLICATION LAYER PROTOCOL ATTACKS

C-RAN uses relevant protocols like TCP/UDP to provide application delivery services in the transport and application layers, which is similar to traditional wired networks. Thus, such attacks as TCP/UDP flooding attack, sequence number prediction attack, SQL injection, and FTP bounce attack could happen [14]. Corresponding solutions to resist the above attacks work herein.

2) CLOUD COMPUTING SECURITY THREATS

If a malicious user attacks C-RAN platform, the security of the VBS pool will be compromised very easily. Tari *et al.* [15] summarized the opportunities, solutions, and progress of cloud security and privacy research in recent years and discussed the shortcomings of traditional cryptography in dealing with cloud computing. For a good cloud system environment, the system should store and manage data with security, employ reliable trusted third party and trust management, and support fine-grained access control, and so on. Xiao and Xiao [16] defined the basic security requirements for building a trustworthy cloud platform system: i) outsourcing security which means the cloud platform should present the solutions to security problems in terms of trust authentication, authorization, and protecting privacy through encryption. ii) multi-tenancy security that the common cloud platform should ensure fairness, reasonability and security of resource allocation arrangement when running in a virtual environment. iii) massive data and intense computation security that the cloud system should have new strategies and protocols to ensure the security of massive data and intense computation processes.

3) VIRTUALIZATION THREATS

The traditional base station connects to a certain number of sector antennas that cover a small area and only handle transmission/reception signals in its coverage area. C-RAN gets together dispersive BSs from a VBS pool based on virtualization. In the pool, different BS closely cooperate with each other and efficiently share resources and processing power. Wang *et al.* [17] summarized four preventive methods (i.e., virtual machine-based trusted computing, virtual machine-based intrusion detection, virtual machine-based kernel protection, and virtual machine-based access control) to prevent the cloud platform from current common attacks (e.g., tampering guest or host machines, virtual machine covert channels, virtual machine-based rootkits and Virtual Machine Manager (VMM) attacks).

4) PRIVACY THREATS

In C-RAN application scenarios, the cloud platform dynamically allocates free spectrum resources based on users' geographic locations. In this process, malicious organization or unauthorized parties may steal user privacy (e.g., personal affairs, personal information, etc.). So, it is necessary to ensure a cloud service platform is safe and trusted.

In conclusion, one challenge of C-RAN is virtualization adoption to ultimately realize dynamic cloud-level resource allocation and management [32]. Current study does not concern how to support multiple operators and allow them to collaborate with each other in a trustworthy way in order to save each one's cost [6]. An ideal C-RAN platform should have a trusted cooperation mechanism to support multiple operators and allow them to collaborate with each other in the VBS pool. From the view of typical security requirements of the C-RAN service plane, designing a trusted authentication and trustworthy cooperation mechanism is a very effective way to overcome the above threats or attacks. But the literature still lacks studies on this issue.

B. TRUSTED COMPUTING AND INTEGRITY MEASUREMENT

In the C-RAN architecture, multiple VBS pools share a hardware platform which leads to reduced operating expenses and improves hardware utilization through virtualization. However, it has brought about a slew of concerns with regard to security. The applications of the VBS pool technology thus give rise to stringent security requirements in the areas of platform integrity, trusted authentication and trustworthy cooperation. Trusted Computing (TC) is a good solution to satisfy these security requirements based on the integration of hardware platform and trust root, which has gained great ground in industry and academia [18]. In 2007, the Trusted Computing Group (TCG) [19] proposed a series of technical specifications and core components, which includes the Trusted Platform Module (TPM) [12], [20], [22] and the TCG Software Stack (TSS) [21]. The virtualization technology makes it possible that multiple virtual machines (VM) share the resources of the same host machine. It is necessary to virtualize TPM (vTPM) so that its capabilities can be used for all VM running on the platform due to the limitations of the physical TPM. vTPM is designed to implement the same functions of the physical TPM, and it also has some unique features [22]: 1) a virtual TPM instance can migrate along with the migration of its associated VM; 2) after vTPM migrating, its keys and encrypted data cannot loss; 3) the vTPMs are isolated to each other among different VMs, but they can interact with each other in a same host machine. Intel®SGX [38] protects selected code and data from disclosure or modification through the use of enclaves, which are protected areas of execution in memory. These approaches and protocols can ensure a trustworthy cloud platform and provide security related services (e.g., integrity measurements of a system, remote attestation and sealing or binding, as well as secure execution). However, its usage or adoption in C-RAN is still lacking. They cannot ensure that the trust relationship between two parties can be sustained as expectation after attestation based on a sustaining policy.

In [23], Krautheim proposed a new private virtual infrastructure (PVI) scheme, which is a cloud resource management and security model. TPM is used as a basis for trust in the PVI. Through TPC technology, PVI gives rights to users based on their security level and restrict cloud resources that the user is allowed to access, which reduces the risk of a cloud platform provider by sharing the security responsibility between the cloud provider and the customers.

Sadeghi and Stüble [24], Sadeghi [25] first analyzed some drawbacks of remote trusted attestation based on computer hardware platforms and software configuration, which is not beneficial for backup and update of this platform [24]. They also discussed two shortcomings of existing solutions for virtual TPM. First, after one platform's VM and vTPM migrate to another platform with different integrity measurement, the VM can no longer access the keys of vTPM and the data protected by those keys. Not only that, but the original key generation strategies or other security properties are also missing. Second, when the software of one platform performs an update, the same problems happen. Then, they designed a novel property-based attestation vTPM architecture to solve the above problems through property-based attestation and sealing [25].

Nagarajan and Varadharajan [26] pointed out that the property-based trust attestation still has a problem: the whole attestation process introduces some uncertainties that reduce trust in the property-based attestation and cause such a problem that server customers (end-users or other demanders) cannot be completely sure whether the server truly satisfies the claimed properties. For server customers, these uncertainties mainly reflect in the following aspects: first, a dynamic system cannot go down in some scenarios, and it is highly stable and updated constantly with new functions. But, this dynamic system exists vulnerability that the measured value at boot time is not equal to the state of the system at the time of attestation. The server customers may not trust this value as time goes by. Second, in property-based attestation, the various properties of the platform need the corresponding binary measurements and property certificates to prove their credibility. However, a third-party Certification Authority (CA) is responsible for issuing and managing these certificates. CA only issues the certificates for each standalone component and not for the whole system environment of the server provider, which causes an uncertainty on the trust of all components coming together in one platform, and it is difficult to ensure that these components' property certificates are effective under the influence of each other. Third, no matter how credible CA property certificate is, the trust value of this certificate will be reduced as long as the server customers do not recognize that CA is trustworthy. For solving the above uncertainties, Nagarajan *et al.* designed a Trust Enhanced Security Model (TESM), which had led to a further development of trusted computing. However, it also has a problem that the authors did not consider: how to maintain the trust relationship between the both sides in next process. After a successful authentication, an unsafe scenario may occur that no matter how the certifier's platform changes, the challenger will still consider it trustworthy. In our scheme, we overcome this problem in the above schemes and maintain the trust to support cooperation fulfillment by creating a trust policy after trust attestation and let TPM ensure the execution of the trust policy.

C. GAME THEORY

Game theory belongs to applied mathematics. It is a mathematical modeling method to help rational decision makers choose conflict or cooperation in order to find an optimal behavior strategy [1]. Game considers the individual's predicted and actual behaviors in the game and studies their optimization strategies. In a game, every rational player wants to find an optimal solution to maximize their own utilities, which aims to increase benefits and reduce costs. There are some basic elements in a game, which mainly include player, strategy, payoff, outcome and equilibrium. In a game, each participant who has a right to make a decision becomes a player. Strategy is a player's plan that is feasible throughout the game. If there are some strategies for a player to choose in a game, then the game is called a finite game, otherwise it is an infinite game. The result of each player at the end of a game is payoff. In a game, the strategy a player chosen affects not only its own payoff but also the other players'. And the combination of all payoffs is the outcome of this game. Equilibrium is the meaning of balance literally. It refers to the optimal strategy combination of all players. Nash Equilibrium (NE) refers to a stable state where every player has no motivation to change its current strategy. NE is a game solution concept that can help forecast a player's action when given the other players' actions [34].

Generally, the games can be divided into cooperative game and non-cooperative game. The non-cooperative game cannot be realized by stimulation and restraint. From the time series of players' actions, the games are further divided into two categories: static game and dynamic game. In the static game, players choose their strategies at the same time or not, but the latter does not know what strategy the first player has taken. In the dynamic game, players have a sequence of actions, and the latter can observe the action selected by the first one. According to players' understanding of others, the games are divided into game with complete information and game with incomplete information. In a game with complete information, each player has accurate information about the characteristics of other players, such as their strategy spaces, and payoff functions. The game with incomplete information refers to a game that at least one player has the inaccuracy of information about the characteristics, strategic spaces and payoff functions of other players.

AI-Dhanhani *et al.* [36], [37] applied a repeated non-cooperative game to analyze the free-riding behaviors inside collaborative groups in educational social applications. Each player in this game model can post a request and answer others' requests based on Tit-for-Tat strategy. The simulation results show that the Tit-for-Tat strategy cannot eliminate free-riding behaviors effectively. The authors pointed out a possibility to suppress free-riding behaviors by introducing reputation [37] and punishment [36]. Shen *et al.* [35] applied a social dilemma game model to study whether network entities would like to accept a Global Trust Management system for unwanted traffic control. They proposed a trust-based punishment mechanism and an incentive mechanism to motive the adoption of the unwanted traffic control system. Gao *et al.* [33] also used a social dilemma model to analyze the acceptance of a cloud data access control system based on reputation. They added cloud service provider's reputation to the game and built a repeated public-goods game to study whether users want to store their own data at the cloud based on a reputation mechanism. In Section IV we will analyze the acceptance of IOCS for fostering cooperation among various operators based on the game theory.

In this paper, we aim to develop an effective scheme that supports multiple operators to cooperate with each other in a distributed environment. Through this scheme, multiple operators' VBS pools can collaborate in a secure and trustworthy way. We focus on trust maintenance based on initial trust attestation, which was not considered in the previous work of other researchers.

III. INTER-OPERATOR COOPERATION SCHEME

In this section, we detail the system structure, the protocol of trust attestation and cooperation, the procedure of network resource leasing, rental provision, rental selection and rental accounting in IOCS.

The IOCS aims to support multiple operators based on the C-RAN architecture and allows them to collaborate with each other in a trustworthy way in order to save each one's cost. It securely allows one operator's subscribers to consume network resources of the others to gain expected quality of 5G services when its own network resources cannot satisfy the QoS requirements.

We set up an access trust management pool that contains a number of VBS trust managers, each of which takes care of one operator's VBS pool (i.e., radio resources). In this pool, tokens are issued among VBS trust managers to allow radio resource rental and utilization in a general and secure way and balance network resource usage across multiple operators' VBS pools. Concretely, the trust manager of one operator applies trust attestation to ensure that the VBS pool and the VBS trust manager in another operator is trustworthy as expectation and embed its cooperation policy into it for secure cooperation. When an operator's network resource is not sufficient, its VBS trust manager contacts other VBS trust managers by providing a rental request. The other VBS trust managers offer rental to the requester based on their provision policies by considering such factors as current status of resource consumption (free resources), rental request, priority of resource rental, agreement with the requester, estimated resource demand in a requested period, etc. The requesting trust manager compares the offers from other TMs and decides the rental based on its decision strategy, e.g., credit venture evaluation of the rental providers and their reputation. Then, the access trust management pool locates the selected operator (TM and VBS pool) to handle the networking of another operator's subscribers.

The lending operator's VBS trust manager counts the rental time and consumed resources and reports to the renting one in a trustworthy way, supported by the lender's trust attestation, trust monitoring and trust assurance in accordance with the renting policy. A token is generated that contains the rental time, the consumed resources and the unit charge. It is signed by both the renting operator and the lending operator. This token can be applied to claim rental profits.

A. NOTATIONS

For easy reference, Table 1 summarizes the notations used in this section.

TABLE 1. Notations.

Notation	Description
TM_m	The trust manager m
O_i	The operator i
fr_i	The free resources of operator i
rr_i	The requested resources to operator i
$Sign(x, y)$	The signature of user y on x
cr	The consumed resources
$\alpha, \beta, \gamma, \delta$	The weighting parameters
RA	The total rental account
RR_m	The rental request of TM_m
RO_m	The rental offer of TM_m
rt_m	The rental time of TM_m
SK_m	The private key of TM_m
$rp_{i,j}$	The priority of resource rental TM_i to TM_j
er_i	The estimated resource demand of TM_i
$P_{i,j}$	The policy of TM_i on TM_j
$up_{i,j}$	The unit rental price of TM_i to TM_j
$TV_{i,j}$	The trust value of TM_i on TM_j
$T_{i,j}$	The personal trust value of TM_i according to past experiences on TM_j
$T_{n,j}$	The trust recommendation of TM_n on TM_j
$S_{i,j}$	The selection index of TM_i on TM_j
$Cert\_X$	The execution environment certificate of X
$Pool_n$	The virtual base station pool n

B. SYSTEM STRUCTURE

The system structure is shown in Fig. 1. There are a number of different operators: $O_1, O_2, \ldots, O_n$, each of which has its RHHs. Operator O_i owns its VBS pool ($Pool_i$). VBS trust manager TM_i manages $Pool_i$ and cooperates with other operators' VBS trust managers. VBS Pools communicate and cooperate with each other through their trust managers. VBS pools connect with core network functional units.

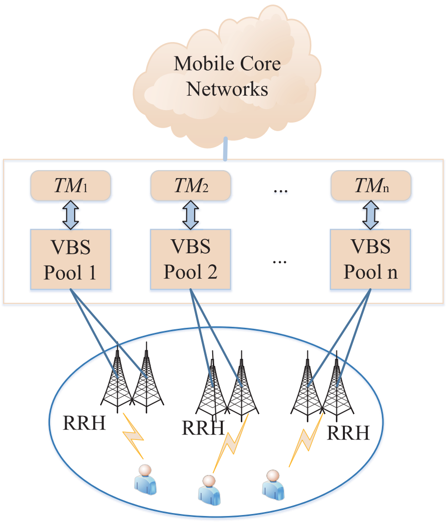


FIGURE 1. IOCS system structure.

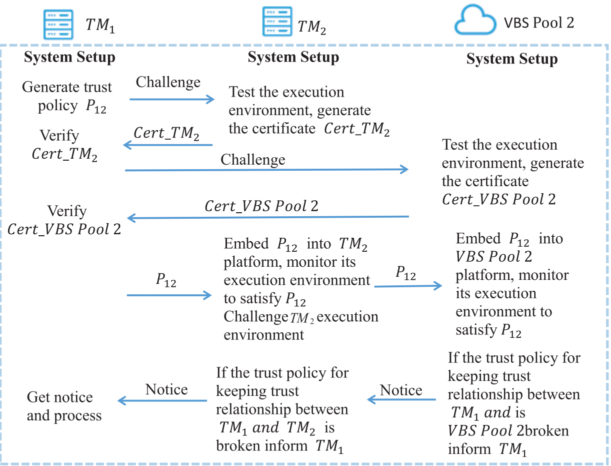

TM_1
TM_2
VBS Pool 2
System Setup
System Setup
System Setup
Generate trust policy P_{12}
Challenge
Test the execution environment, generate the certificate $Cert\_TM_2$
Verify $Cert\_TM_2$
$Cert\_TM_2$
Challenge
Test the execution environment, generate the certificate $Cert\_VBS\ Pool\ 2$
Verify $Cert\_VBS\ Pool\ 2$
$Cert\_VBS\ Pool\ 2$
P_{12}
Embed P_{12} into TM_2 platform, monitor its execution environment to satisfy P_{12} Challenge TM_2 execution environment
P_{12}
Embed P_{12} into $VBS\ Pool\ 2$ platform, monitor its execution environment to satisfy P_{12}
Get notice and process
Notice
If the trust policy for keeping trust relationship between TM_1 and TM_2 is broken inform TM_1
Notice
If the trust policy for keeping trust relationship between TM_1 and is $VBS\ Pool\ 2$broken inform TM_1

FIGURE 2. The protocol of trust attestation for secure cooperation.

C. TRUST ATTESTATION AND COOPERATION

In order to ensure trust cooperation among multiple VBS pools, we propose a protocol of trust attestation and cooperation, as shown in Fig. 2. Before O_1 collaborates with O_2, it needs to establish a trust relationship with O_2. It should also verify VBS $Pool_2$ and TM_2 in case of any intrusions and attacks happened. After mutual authentication, trust-maintenance mechanism is set up between the two parties. If both VBS pool and TM apply the TPM with a root trusted module (RTM), the trust maintenance is controlled through the conditions defined by the RTM [11]. The RTM (implemented by TPM) is applied for verifying some intended purposes of a client and ensuring that the VBS pool will work as its TM's expectation and a TM will work as another TM's expectation. Assume that Operator O_1 needs to collaborate with Operator O_2 with regard to resource lease, it performs the following preparation.

1) TM_1 generates its trust policy $P_{1,2}$ with O_2. TM_1 challenges the execution environment of TM_2. TM_2 replies its execution environment certificate $Cert\_TM_2$ and its VBS Pool's address.
2) TM_1 verifies $Cert\_TM_2$. If the verification is positive, it challenges VBS $Pool_2$'s execution environment by getting VBS $Pool_2$'s execution environment certificate $Cert\_Pool_2$.
3) By getting positive verification on $Cert\_Pool_2$, TM_1 sends $P_{1,2}$ to TM_2. TM_2 embeds $P_{1,2}$ into its execution platform, which monitors TM_2's execution environment and checks any changes that violate $P_{1,2}$ at TM_2. TM_2 further passes $P_{1,2}$ to VBS $Pool_2$, which then embeds $P_{1,2}$ into VBS $Pool_2$'s platform. TM_2 monitors whether the VBS $Pool_2$'s execution environment satisfies $P_{1,2}$ by checking any changes at VBS $Pool_2$.
4) If $P_{1,2}$ is violated at either VBS $Pool_2$ or TM_2, TM_2 will inform TM_1 to perform corresponding actions, e.g., stop cooperation with TM_2 and deny rental account in TM_2.

If TM and VBS pool apply TCP with a RTM, we can apply the trust-maintenance mechanism as described in [11] and [32] to achieve trust attestation and ensure that a VBS Pool will work as its TM's expectation and one TM will work as another TM's expectation, and vice versa. Note that the same trust attestation is normally performed by TM_2 on TM_1 in order to ensure TM_1 will perform as our protocol design.

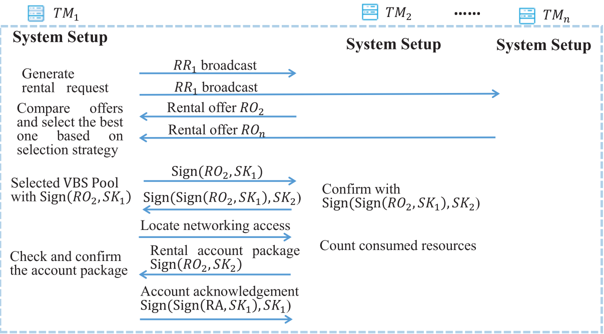

TM_1
TM_2
......
TM_n
System Setup
System Setup
System Setup
Generate rental request
RR_1 broadcast
RR_1 broadcast
Compare offers and select the best one based on selection strategy
Rental offer RO_2
Rental offer RO_n
Selected VBS Pool with Sign(RO_2, SK_1)
Sign(RO_2, SK_1)
Sign(Sign(RO_2, SK_1), SK_2)
Confirm with Sign(Sign(RO_2, SK_1), SK_2)
Locate networking access
Count consumed resources
Check and confirm the account package
Rental account package Sign(RO_2, SK_2)
Account acknowledgement Sign(Sign(RA, SK_1), SK_1)

FIGURE 3. The procedure of resource renting.

D. COOPERATION PROCEDURE

When an operator's resources are not sufficient, it requests rental from other operators. The procedure about TM_1 to rent resources from TM_2 is shown in Fig. 3.

1) TM_1 generates a rental request RR_1, which contains the volume of requested resources (rr_1) and rental time (rt_1), $RR_1 = (rr_1, rt_1)$;
2) TM_1 broadcasts RR_1 to the other TMs. When the other TMs receive RR_1, they check their free resources and provide rental offer RP_1 (refer to Section III-E for details);
3) TM_1 compares all rental offers and selects the best one based on its selection strategy (more details can be found in Section III-E);
4) Suppose that TM_2 is selected, TM_1 requests locating networking access to the selected VBS pool $Pool_2$ with message $Sign(RO_2, SK_1)$, where SK_1 is the private key of TM_1;
5) TM_2 replys $Sign(Sign(RO_2, SK_1), SK_2)$ for confirming the selection;
6) TM_1 locates networking access to TM_2, which processes the request of TM_1 accordingly with its VBS pool $Pool_2$. Meanwhile, TM_2 counts the consumed resources RA during the whole rental period;
7) After the rental time rt_1 is over, TM_2 sends the rental account package $Sign(RA, SK_2)$, where RA is the rental account (refer to Section III-E for a method of RA calculation);
8) TM_1 checks and confirms the account package if it believes that the previous attested trusted platform of TM_2 and VBS $Pool_2$ are not intruded and then sends its acknowledgement $Sign(Sign(RA, SK_2), SK_1)$ to TM_2.

E. RENTAL PROVISION, SELECTION AND ACCOUNTING

When TM_j gets a rental request from TM_i, it generates its rental offer by considering the following factors:

- current status of resource consumption, i.e., free resources (fr_j); (Herein, we hold such a policy that each operator should satisfy its own users first before lending any extra resources to other operator.)
- requested resource (rr_i);
- priority of resource rental to the requesting TM_i ($rp_{i,j}$);
- agreement between TM_i and TM_j;
- estimated local resource demand (er_j) in the demanded future period et_i.

Algorithm 1 specifies how to calculate the rental offer. Our algorithm aims to ensure the offered rental can satisfy the requesting party's QoS expectation.

Algorithm 1 Rental Offer Calculation

```
If there is no request from a TM ($TM_i$) with higher priority,
that is, there is no requesting $TM_K$ with $rp_{k,j} > rp_{i,j}$;
  If $fr_j < rr_i$, reject request; Else
    If $er_j + rr_i < fr_j$
      Generate rental offer package $RO_i$ based on agreed
      price $up_{i,j}$ between two involved operators.
    Else
      Process the next TM request.
```

1) RENTAL SELECTION

If TM_i gets multiple rental offers, the requesting trust manager compares the offers (unit price $up_{i,j}$) from other operators and decides the rental based on its selection strategy.

Trust value ($TV_{i,j}$) of TM_i on TM_j is generated based on TM_i's personal trust according to past experiences ($T_{i,j}$) and other TM's recommendation ($T_{n,j}$) as shown in the following equation:

$$TV_{i,j} = \alpha \times T_{i,j} + \beta \times \frac{\sum_{n=1}^{N} \left(1 - |T_{i,j} - T_{n,j}|\right) \times T_{n,j}}{N-1},$$

where α and β are weighting parameters to balance the contribution of $T_{i,j}$ and $T_{n,j}$ in trust value generation. Herein, the use deviation $|T_{i,j} - T_{n,j}|$ to tailor the contribution of $T_{n,j}$ in $TV_{i,j}$ calculation in order to resist bad mouthing attacks.

An example decision function is shown in (1).

$$S_{i,j} = \gamma \times TV_{i,j} + \delta \times \frac{1}{up_{i,j}}, \tag{1}$$

Obviously, the decision is made by considering trust value $TV_{i,j}$ and unit rental price $up_{i,j}$. Parameters γ and δ are weighting factors to weight $TV_{i,j}$ and $up_{i,j}$.

The requesting trust manager TM_i selects TM_j with the biggest $S_{i,j}$. Then the access trust management pool locates the selected operator to handle the networking access of another operator's subscribers.

2) RENTAL ACCOUNTING

The rental time ($rt_{i,j}$) and consumed resources ($cr_{i,j}$) are counted by the lending operator's VBS trust manager and reported to the renting one in a trustworthy way. A token is generated that contains the rental time, consumed resources and unit charge. It is signed by both the renting operator and the lending operator. This token can be used for claiming rental profit. Concretely,

$$RA_{i,j} = rt_{i,j} \times cr_{i,j} \times up_{i,j}. \tag{2}$$

The token is $Sign(Sign(RA, SK_2), SK_1)$.

IV. SECURITY ANALYSIS AND PERFORMANCE EVALUATION

In this section, we analyze the security of the proposed scheme and evaluate the performance of IOCS based on scheme implementation in terms of trust attestation accuracy, operation efficiency, and reasonable resource consumption (including operation time, memory cost, CPU consumption, and communication cost). Furthermore, we apply game theory to analyze the possibility of potential cooperation among multiple operators.

A. SCHEME IMPLEMENTATION

We implemented IOCS in Lenovo ThinkServer TS250 with Intel Xeon E3-1225 v5 CPU of 3.30GHz, 2TB hard disk and 32GB RAM. We deployed a cloud computing management platform to simulate the access trust management pool that managed different operators' VBS pools and TMs based on OpenStack. The OpenStack is an open source cloud computing resource-management tool. It can quickly deploy a virtual environment and create multiple virtual servers in this environment [27], [28]. Besides, any users can deploy their own applications in these servers. We built up eleven 64-bit RedHat Enterprise Linux Servers (simulated eleven operator nodes) in OpenStack to simulate the process of remote trust attestation and cooperation among different VBS pools and TMs. A MySQL database is installed in each server, which is applied to simulate the network resources of each operator. Then, we used TPM emulator (one vTPM instance developed by Mario Strasser in 2004 [29], [30]) to simulate the RTM and store the AIK in each server. The emulator can implement the functions of vTPM, such as platform identity authentication, platform attestation and certification. Based on the above, we implemented the IOCS with C/C++ language and demonstrated our scheme's functions by calling the API provided by the TSS. In the TPM emulator, TrouSers is used to provide the same functions and APIs of TSS, and TCG Device Driver Library (TDDL) is used to facilitate interaction between TSS and TPM device drivers. We employed the TPM emulator daemon (TPMD) that is a user space daemon to implement the functions of TPM, such as a daemon application, a TPM emulator engine and cryptographic functions [10].

Based on our implementation, we comprehensively verified and tested the features of our IOCS. Through the TPM emulator, each virtual machine can provide its own PCR lists, system event logs and credentials to complete trust attestation. Once trust authentication is successful, cross-network cooperation among multiple mobile operators' VBS pools will be established based on a relevant trust policy. We designed a heartbeat mechanism, which is a data packet that can detect the validity of nodes, to ensure reliable communications between the tenant and renter. If any party violates the trust policy, the TPM can warn the other party through the heartbeat mechanism. For example, when the virtual machine configuration (e.g. router network or firewall configuration) changes, the PCR will change accordingly and then the warning will be sent if this change is against the trust maintenance or sustainment policy.

B. SECURITY ANALYSIS

In Section II, we discussed the security issues and requirements of the C-RAN service plane. The motivation of our work is to design a comprehensive and universal solution for trusted cooperation across multiple cellular networks in C-RAN. Our scheme has the following security features.

First, the IOCS provides security in three levels. The OpenStack platform can issue tokens to verify the identity of the VBS Pool. The RTM module not only ensures the security of virtual machine, but also realizes remote authentication. We dynamically maintain the trust relationship between the requester and the resource provider based on the procedure oftrust cooperation according to the policy for trust relationship maintenance.

Second, the trust attestation we adopt has an improved integrity measurement. It can measure specified related configuration files or log files based on our proposed trust attestation protocol and trust cooperation procedure, as well as the trust maintenance or sustainment policy. During the cooperation of two operators, the RTM of one party warns another party through the heartbeat mechanism whenever the platform's configuration status and key files changes are detected.

Finally, the selection strategy based on the trust value of each node effectively prevents threats from malicious nodes. For example, when the tenant gets multiple rental offers, it can select the most reliable one based on its trust in the renter.

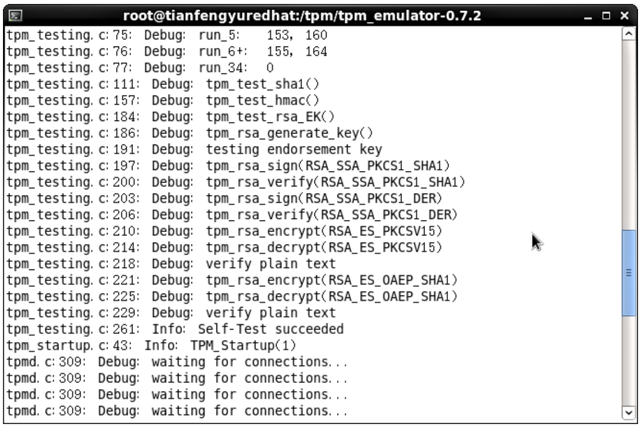
C. PERFORMANCE TEST AND EVALUATION

1) CORRECTNESS TEST

We first tested execution correctness of our scheme implementation in the experimental environment by verifying whether each module is connected and integrated successfully.

a: TPM and TDDL Integration Test

After successfully installing and deploying the TPM emulator, we started to test this module's deployment and running state. The start-up process of the TPM includes the detection of a random number generator, the detection of the cryptographic algorithm engine, the detection of a key system, platform self-test, plain text verification, etc. When the TPM is officially launched, it begins to wait for the TrouSers to connect to it via the API of the TDDL. The TPM emulator start-up process is shown in Fig. 4.



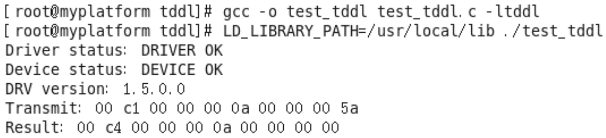
```
tpm_testing.c:75:  Debug:  run_5:    153, 160
tpm_testing.c:76:  Debug:  run_6+:   155, 164
tpm_testing.c:77:  Debug:  run_34:   0
tpm_testing.c:111:  Debug:  tpm_test_sha1()
tpm_testing.c:157:  Debug:  tpm_test_hmac()
tpm_testing.c:184:  Debug:  tpm_test_rsa_EK()
tpm_testing.c:186:  Debug:  tpm_rsa_generate_key()
tpm_testing.c:191:  Debug:  testing endorsement key
tpm_testing.c:197:  Debug:  tpm_rsa_sign(RSA_SSA_PKCS1_SHA1)
tpm_testing.c:200:  Debug:  tpm_rsa_verify(RSA_SSA_PKCS1_SHA1)
tpm_testing.c:203:  Debug:  tpm_rsa_sign(RSA_SSA_PKCS1_DER)
tpm_testing.c:206:  Debug:  tpm_rsa_verify(RSA_SSA_PKCS1_DER)
tpm_testing.c:210:  Debug:  tpm_rsa_encrypt(RSA_ES_PKCSV15)
tpm_testing.c:214:  Debug:  tpm_rsa_decrypt(RSA_ES_PKCSV15)
tpm_testing.c:218:  Debug:  verify plain text
tpm_testing.c:221:  Debug:  tpm_rsa_encrypt(RSA_ES_OAEP_SHA1)
tpm_testing.c:225:  Debug:  tpm_rsa_decrypt(RSA_ES_OAEP_SHA1)
tpm_testing.c:229:  Debug:  verify plain text
tpm_testing.c:261:  Info:  Self-Test succeeded
tpm_startup.c:43:  Info:  TPM_Startup(1)
tpmd.c:309:  Debug:  waiting for connections...
tpmd.c:309:  Debug:  waiting for connections...
tpmd.c:309:  Debug:  waiting for connections...
tpmd.c:309:  Debug:  waiting for connections...
```

FIGURE 4. **TPM emulator successfully starts.**

After following the previous steps and completing the installation and deployment of the modules, we applied our test program to test the TDDL. When the TPM emulator and TDDL are integrated successfully, we can get TDDL driver status, TDDL device status, TDDL version and other information from the TDDL test program, as shown in Fig. 5.

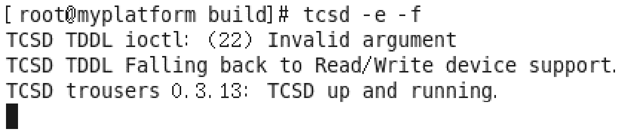
b: TPM and TSS Integration Test

When the above integration was successful, we started to test whether the connection between TPM and TSS is successful. TSS needs to detect the running status of the TPM. In other words, after the TPM successfully runs, the TSS can be initiated and run. As shown in Fig. 6, the successful operation of Trusted Computing Group Service Daemon (TCSD) shows the success of TSS and TPM integration. When the above steps are completed, we can get the PCR value of the platform by running the "tpm_readpcr" command, as shown in Fig. 7.



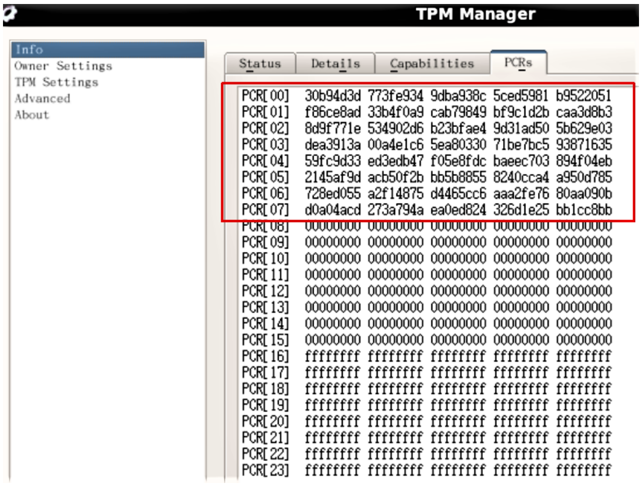
```
[root@myplatform tddl]# gcc -o test_tddl test_tddl.c -ltddl
[root@myplatform tddl]# LD_LIBRARY_PATH=/usr/local/lib ./test_tddl
Driver status:  DRIVER OK
Device status:  DEVICE OK
DRV version:  1.5.0.0
Transmit:  00 c1 00 00 00 0a 00 00 00 5a
Result:  00 c4 00 00 00 0a 00 00 00 00
```

FIGURE 5. **TPM and TDDL successfully integrate.**



```
[root@myplatform build]# tcsd -e -f
TCSD TDDL ioctl: (22) Invalid argument
TCSD TDDL Falling back to Read/Write device support.
TCSD trousers 0.3.13: TCSD up and running.
```

FIGURE 6. **TSS successfully starts.**



```
PCR[00]  30b94d3d 773fe934 9dba938c 5ced5981 b9522051
PCR[01]  f86ce8ad 33b4f0a9 cab79849 bf9c1d2b caa3d8b3
PCR[02]  8d9f771e 534902d6 b23bfae4 9d31ad50 5b629e03
PCR[03]  dea3913a 00a4e1c6 5ea80330 71be7bc5 93871635
PCR[04]  59fc9d33 ed3edb47 f05e8fdc baeec703 894f04eb
PCR[05]  2145af9d acb50f2b bb5b8855 8240cca4 a950d785
PCR[06]  728ed055 a2f14875 d4465cc6 aaa2fe76 80aa090b
PCR[07]  d0a04acd 273a794a ea0ed824 326d1e25 bb1cc8bb
PCR[08]  00000000 00000000 00000000 00000000 00000000
PCR[09]  00000000 00000000 00000000 00000000 00000000
PCR[10]  00000000 00000000 00000000 00000000 00000000
PCR[11]  00000000 00000000 00000000 00000000 00000000
PCR[12]  00000000 00000000 00000000 00000000 00000000
PCR[13]  00000000 00000000 00000000 00000000 00000000
PCR[14]  00000000 00000000 00000000 00000000 00000000
PCR[15]  00000000 00000000 00000000 00000000 00000000
PCR[16]  ffffffff ffffffff ffffffff ffffffff ffffffff
PCR[17]  ffffffff ffffffff ffffffff ffffffff ffffffff
PCR[18]  ffffffff ffffffff ffffffff ffffffff ffffffff
PCR[19]  ffffffff ffffffff ffffffff ffffffff ffffffff
PCR[20]  ffffffff ffffffff ffffffff ffffffff ffffffff
PCR[21]  ffffffff ffffffff ffffffff ffffffff ffffffff
PCR[22]  ffffffff ffffffff ffffffff ffffffff ffffffff
PCR[23]  ffffffff ffffffff ffffffff ffffffff ffffffff
```

FIGURE 7. **PCR value of the platform integrity metric.**

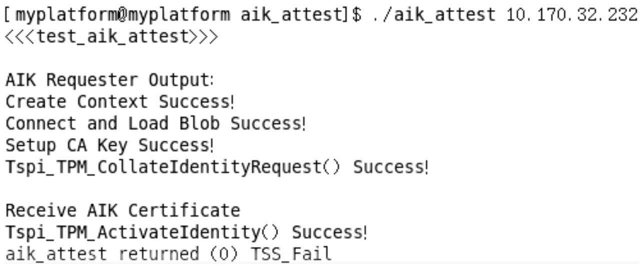
2) TRUST ATTESTATION TEST

a: AIK Certificate Application and Issue Test

A certificate application platform is running on a 64-bit RedHat Server virtual machine platform with a trusted computing environment. First, the virtual machine platform needs to apply for an AIK certificate from the trusted third party (TTP server), and then the platform information of the applicant, such as EK and public key, is verified by the TTP. The process is shown in Fig. 8, which mainly applies the asymmetric data block (asymBlob), the hashes of session key and the requester's AIK public key for issuing AIK certificate. After passing the verification, TTP issues the AIK. Once receiving the AIK, the certificate application platform activates it via the Tspi_TPM_ActivateIdentity() interface as presented in Fig. 9.

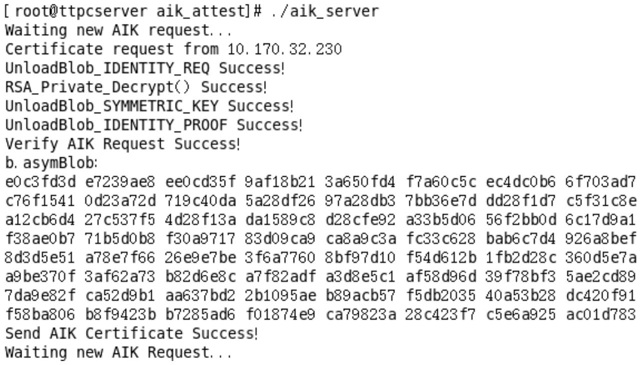
b: Trust Attestation Process

First, the tenant sends a request connection to the renters, then the renters authenticate it. The verification mainly includes of identity information, metrics and PCR values, and the



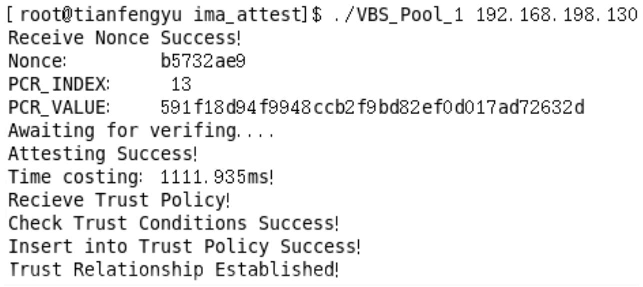
```
[myplatform@myplatform aik_attest]$ ./aik_attest 10.170.32.232
<<<test_aik_attest>>>

AIK Requester Output:
Create Context Success!
Connect and Load Blob Success!
Setup CA Key Success!
Tspi_TPM_CollateIdentityRequest() Success!

Receive AIK Certificate
Tspi_TPM_ActivateIdentity() Success!
aik_attest returned (0) TSS_Fail
```

FIGURE 8. The process of AIK certificate application.



```
[root@ttpcserver aik_attest]# ./aik_server
Waiting new AIK request...
Certificate request from 10.170.32.230
UnloadBlob_IDENTITY_REQ Success!
RSA_Private_Decrypt() Success!
UnloadBlob_SYMMETRIC_KEY Success!
UnloadBlob_IDENTITY_PROOF Success!
Verify AIK Request Success!
b.asymBlob:
e0c3fd3d e7239ae8 ee0cd35f 9af18b21 3a650fd4 f7a60c5c ec4dc0b6 6f703ad7
c76f1541 0d23a72d 719c40da 5a28df26 97a28db3 7bb36e7d dd28f1d7 c5f31c8e
a12cb6d4 27c537f5 4d28f13a da1589c8 d28cfe92 a33b5d06 56f2bb0d 6c17d9a1
f38ae0b7 71b5d0b8 f30a9717 83d09ca9 ca8a9c3a fc33c628 bab6c7d4 926a8bef
8d3d5e51 a78e7f66 26e9e7be 3f6a7760 8bf97d10 f54d612b 1fb2d28c 360d5e7a
a9be370f 3af62a73 b82d6e8c a7f82adf a3d8e5c1 af58d96d 39f78bf3 5ae2cd89
7da9e82f ca52d9b1 aa637bd2 2b1095ae b89acb57 f5db2035 40a53b28 dc420f91
f58ba806 b8f9423b b7285ad6 f01874e9 ca79823a 28c423f7 c5e6a925 ac01d783
Send AIK Certificate Success!
Waiting new AIK Request...
```

FIGURE 9. The content of AIK certificate.

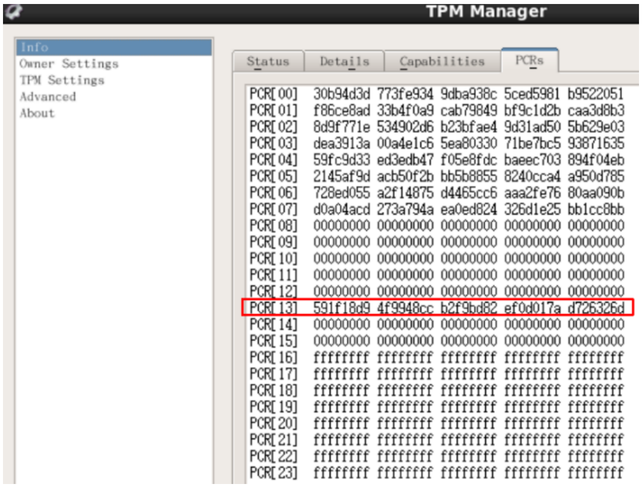


```
[root@tianfengyu ima_attest]$ ./VBS_Pool_1 192.168.198.130
Receive Nonce Success!
Nonce:        b5732ae9
PCR_INDEX:     13
PCR_VALUE:    591f18d94f9948ccb2f9bd82ef0d017ad72632d
Awaiting for verifing....
Attesting Success!
Time costing: 1111.935ms!
Recieve Trust Policy!
Check Trust Conditions Success!
Insert into Trust Policy Success!
Trust Relationship Established!
```

FIGURE 10. Successful trust relationship establishment.

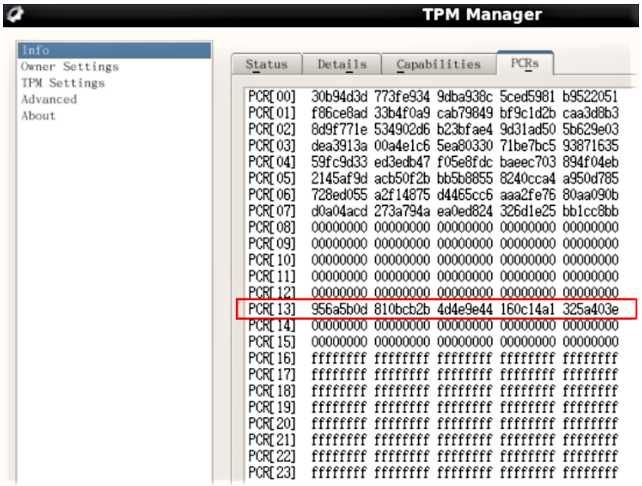
random number sent by the tenant. When this process is over, the trust between the tenant and the renters will be initiated. Next, the verifier sends the trust policy (the demands of trust maintenance) to the tenant, and vice versa. Both parties embed the trust policy into their TPM's execution environment by encoding it. The trust policy limits the tenant's execution environment by checking any changes that disobey the policy at the VBS pool of the tenant. The result is shown in Fig. 10 and Fig. 11.

The trust policy is a new kind of control specification. After the trust relationship is established, we further tested the effect of the trust policy when the platform configuration changes. The trust policy has a certain flexibility; it can allow or restrict the tenant to change some configurations. We made three assumptions. The first one is that the trust policy specifies that the packet forwarding function and the status of source routing in the platform must be closed during the cooperation process. The second one is that the tenant opens these two functions without permission applying Linux commands. The last one is that the TPM



```
PCR[00]  30b94d3d 773fe934 9dba938c 5ced5981 b9522051
PCR[01]  f86ce8ad 33b4f0a9 cab79849 bf9c1d2b caa3d8b3
PCR[02]  8d9f771e 534902d6 b23bfae4 9d31ad50 5b629e03
PCR[03]  dea3913a 00a4e1c6 5ea80330 71be7bc5 93871635
PCR[04]  59fc9d33 ed3edb47 f05e8fdc baeec703 894f04eb
PCR[05]  2145af9d acb50f2b bb5b8855 8240cca4 a950d785
PCR[06]  728ed055 a2f14875 d4465cc6 aaa2fe76 80aa090b
PCR[07]  d0a04acd 273a794a ea0ed824 326d1e25 bb1cc8bb
PCR[08]  00000000 00000000 00000000 00000000 00000000
PCR[09]  00000000 00000000 00000000 00000000 00000000
PCR[10]  00000000 00000000 00000000 00000000 00000000
PCR[11]  00000000 00000000 00000000 00000000 00000000
PCR[12]  00000000 00000000 00000000 00000000 00000000
PCR[13]  591f18d9 4f9948cc b2f9bd82 ef0d017a d726326d
PCR[14]  00000000 00000000 00000000 00000000 00000000
PCR[15]  00000000 00000000 00000000 00000000 00000000
PCR[16]  ffffffff ffffffff ffffffff ffffffff ffffffff
PCR[17]  ffffffff ffffffff ffffffff ffffffff ffffffff
PCR[18]  ffffffff ffffffff ffffffff ffffffff ffffffff
PCR[19]  ffffffff ffffffff ffffffff ffffffff ffffffff
PCR[20]  ffffffff ffffffff ffffffff ffffffff ffffffff
PCR[21]  ffffffff ffffffff ffffffff ffffffff ffffffff
PCR[22]  ffffffff ffffffff ffffffff ffffffff ffffffff
PCR[23]  ffffffff ffffffff ffffffff ffffffff ffffffff
```

FIGURE 11. PCR value [13] of the platform integrity metric.



```
PCR[00]  30b94d3d 773fe934 9dba938c 5ced5981 b9522051
PCR[01]  f86ce8ad 33b4f0a9 cab79849 bf9c1d2b caa3d8b3
PCR[02]  8d9f771e 534902d6 b23bfae4 9d31ad50 5b629e03
PCR[03]  dea3913a 00a4e1c6 5ea80330 71be7bc5 93871635
PCR[04]  59fc9d33 ed3edb47 f05e8fdc baeec703 894f04eb
PCR[05]  2145af9d acb50f2b bb5b8855 8240cca4 a950d785
PCR[06]  728ed055 a2f14875 d4465cc6 aaa2fe76 80aa090b
PCR[07]  d0a04acd 273a794a ea0ed824 326d1e25 bb1cc8bb
PCR[08]  00000000 00000000 00000000 00000000 00000000
PCR[09]  00000000 00000000 00000000 00000000 00000000
PCR[10]  00000000 00000000 00000000 00000000 00000000
PCR[11]  00000000 00000000 00000000 00000000 00000000
PCR[12]  00000000 00000000 00000000 00000000 00000000
PCR[13]  956a5b0d 810bcb2b 4d4e9e44 160c14a1 325a403e
PCR[14]  00000000 00000000 00000000 00000000 00000000
PCR[15]  00000000 00000000 00000000 00000000 00000000
PCR[16]  ffffffff ffffffff ffffffff ffffffff ffffffff
PCR[17]  ffffffff ffffffff ffffffff ffffffff ffffffff
PCR[18]  ffffffff ffffffff ffffffff ffffffff ffffffff
PCR[19]  ffffffff ffffffff ffffffff ffffffff ffffffff
PCR[20]  ffffffff ffffffff ffffffff ffffffff ffffffff
PCR[21]  ffffffff ffffffff ffffffff ffffffff ffffffff
PCR[22]  ffffffff ffffffff ffffffff ffffffff ffffffff
PCR[23]  ffffffff ffffffff ffffffff ffffffff ffffffff
```

FIGURE 12. PCR value [13] of the platform integrity metric when network configuration changes.

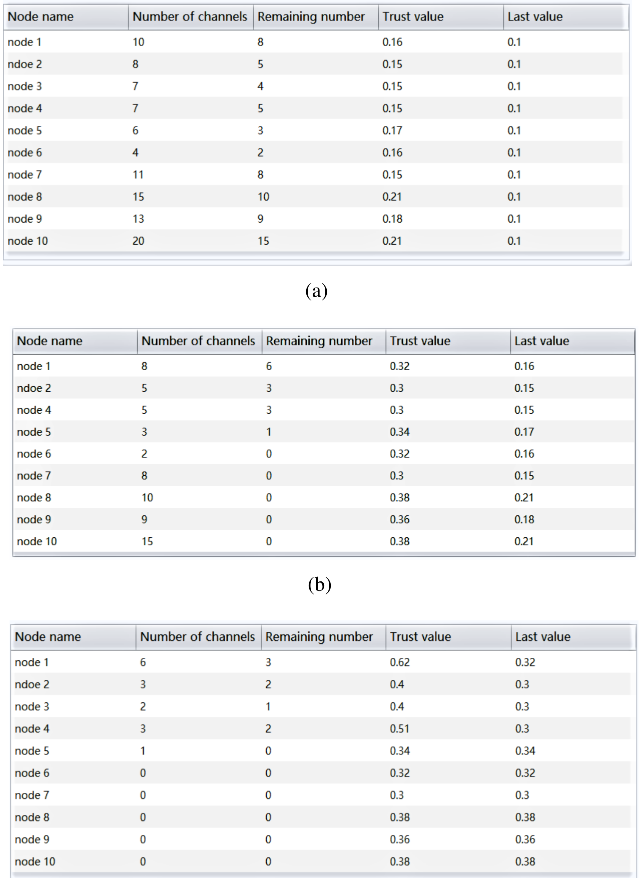
of the tenant will warn the renter with the application of heartbeat mechanism. We present the results of the TPM's measurement on the configuration file in Fig. 12, which is compared with the original one (Fig. 11). Once the measurement value changes, the TPM re-evaluates the PCR. As shown in Fig. 13, the PCR [13] changes from "591f18d94f9948ccb2f9bd82ef0d017ad72632d" to "956a5b0d810bcb2b4d4e9e44160c14a1325a403e", which violates the trust policy. The cooperation between the two parties is ruined due to such a change.

3) SYSTEM PERFORMANCE EVALUATION

We investigated the performance of IOCS by evaluating its operation time, communication cost, memory cost, and CPU usage, which influence the overall system operational efficiency.

a: Operation Time

In terms of operation time, we monitored and counted the execution time of each process for one to one mapping between different nodes' trust cooperation. The results are



Node name	Number of channels	Remaining number	Trust value	Last value
node 1	10	8	0.16	0.1
ndoe 2	8	5	0.15	0.1
node 3	7	4	0.15	0.1
node 4	7	5	0.15	0.1
node 5	6	3	0.17	0.1
node 6	4	2	0.16	0.1
node 7	11	8	0.15	0.1
node 8	15	10	0.21	0.1
node 9	13	9	0.18	0.1
node 10	20	15	0.21	0.1

(a)

Node name	Number of channels	Remaining number	Trust value	Last value
node 1	8	6	0.32	0.16
ndoe 2	5	3	0.3	0.15
node 4	5	3	0.3	0.15
node 5	3	1	0.34	0.17
node 6	2	0	0.32	0.16
node 7	8	0	0.3	0.15
node 8	10	0	0.38	0.21
node 9	9	0	0.36	0.18
node 10	15	0	0.38	0.21

(b)

Node name	Number of channels	Remaining number	Trust value	Last value
node 1	6	3	0.62	0.32
ndoe 2	3	2	0.4	0.3
node 3	2	1	0.4	0.3
node 4	3	2	0.51	0.3
node 5	1	0	0.34	0.34
node 6	0	0	0.32	0.32
node 7	0	0	0.3	0.3
node 8	0	0	0.38	0.38
node 9	0	0	0.36	0.36
node 10	0	0	0.38	0.38

(c)

FIGURE 13. The trust value of the node trust selection.

TABLE 2. Operation time of each main process.

Execution procedures	Time cost (in ms)
AIK Certificate Application and Issue	613.793
Trust credential transmission	246.597
Trust credential verification	62.548
Platform integrity check	681.829
Trust verification	990.974
Trust policy transmission	120.421
Total trust relationship establishment	1111.355

shown in Table 2, which mainly includes the process of AIK certificate application and issue, the process of trust verification (i.e., the process of trust credential transmission, the process of trust credential verification, and the process of platform integrity check), the process of policy transmission, and the process of entire trust relationship establishment. Here, we assume that the system has no malicious and malfunctioning nodes. When the number of nodes is gradually increasing, the operation time for the entire trust relationship establishment among different nodes is concluded in Table 3. We further compared IOCS with one scheme about cloud service attestation [10] and another scheme [31] regarding operation time in terms of some similar procedures. The results are shown in Fig. 14, where the four abbreviations

TABLE 3. Trust relationship establishment time with different number of operator nodes.

Number of nodes	Time cost of successful attestation (in ms)
1	1111.36
2	1378.45
3	1567.34
4	1756.23
5	1945.12
6	2134.01
7	2322.90
8	2511.79
9	2700.68
10	2889.57

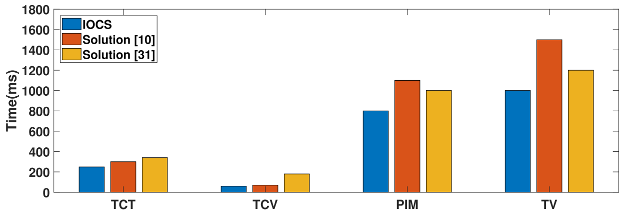


FIGURE 14. Operation time of different solutions' main process.

at abscissa axis refer to trust credential transmission, trust credential verification, platform integrity metrics, and trust verification, respectively. We can see that IOCS is the most efficient one.

b: Memory Cost and CPU Usage

In our scheme implementation, we applied many open source software, such as OpenStack, MySQL, TPM emulator and so on. These modules take up a certain amount of computing and storage resources. Therefore, we need to calculate the resource consumption in IOCS (i.e., memory cost and CPU usage). Assuming that there are two virtual nodes collaborate with each other credibly. We applied the average CPU (CPU avg) to represent CPU usages, which is the average percentage of CPU spent in 60 seconds. For the OpenStack and MySQL, we denote the memory cost as the dedicated memory (DP), which are about 123676-KB and 154712-KB, respectively. The DP refers to actual physical memory footprint in RAM. For the TPM emulator, TSS, remote procedure call (RPC) and daemon in the virtual machine, we also used the virtual memory size (VMS). The VMS includes the process and its shared library memory footprint in the virtual machine. It is the address space of the process and its shared library, and it does not reside in RAM. Due to its shared nature, the VMS of TSS is a bit large, which is 100214-KB. For others, the TPMD occupies 7056-KB and RPC costs 47586-KB. The daemon's DP is 1500-KB. Considering the general configuration of a business server nowadays, the resource costs of the IOCS is acceptable. In addition, we compared the CPU utilization with and without the application of IOCS. As drawn in Fig. 15, we found that applying the IOCS costs about 7% more CPU overhead

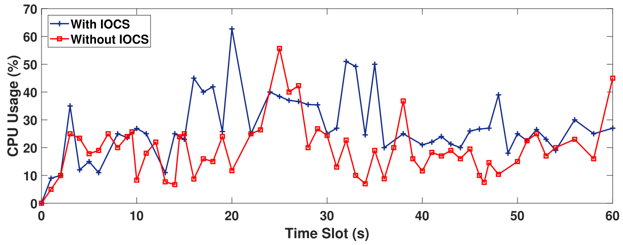


FIGURE 15. CPU Usage of the platform.

compared with the CPU highest occupancy. For CPU average, the IOCS occupies 25% of the CPU.

c: Communication Cost

In the trust attestation process, the size of the random number is about 40-bit. The AIK is an RSA signature key with a default size of 2048-bit. The process of trust credential transmission mainly includes the integrity metrics list and some PCR. The transmitted information is in the form of hash values, which is about 521-byte. In the trust cooperation process, the size of the heartbeat package in the heartbeat mechanism is about 5-byte. The total size of the trust policy is about 2-KB. Therefore, the total communication cost of the process of trust relationship establishment is about 3-KB in our scheme. For 5G networks, this communication cost is very small.

TABLE 4. Notation about game theoretical analysis.

Notations	Description
$u_I^i(t)$	The utility of tenant i at time t
$u_J^j(t)$	The utility of renter j at time t
$ug(t)$	The unit resource usage payment by subscribers at time t in the system (assume that the subscribers of i and j pay the same unit fee of resources)
$uf(t)$	The basic maintenance and operation cost of unit resource in a normal situation at time t
$rf(t)$	The basic unit rental fee paid by tenant at time t
I	The total amount of resources of tenant i
J	The total amount of resources of renter j
I'	The rented amount of resources of tenant i
J'	The lent amount of resources of render j
$TV_j(t)$	The trust value of renter j, $TV_j(t) \in [0,1]$
$TV_i(t)$	The trust value of tenant i, $TV_i(t) \in [0,1]$
$lo_i(t)$	The total loss of tenant i at time t
$A_{x,y}$	The strategy space of tenant and renter

4) ANALYSIS ON IOCS ACCEPTANCE BASED ON GAME THEORY

We modeled a static game model to analyze IOCS's acceptance in practice. The tenant (i.e. tenant operator) and the renter (i.e. renter operator) are the players in this game. The players know each other's information. We mainly analyze whether all players (i.e., operators) would collaborate with each other given their limited network resources and their status. For easy presentation, Table 4 summarizes the notations applied in our analysis.

Our model holds a number of assumptions for game theoretical analysis to investigate the acceptance of tenant operators and renter operators in the IOCS and its deployment condition.

Assumption 1: We assume there are only two types of players for simplification, both of which should treat their own subscribers with high priority. These two types of players are the renter with extra network resources, and the tenant without sufficient network resources. Each operator should firstly serve its own subscribers, if extra resources can be located, it can offer rental services to other operators.

Assumption 2: All players are rational in the game. They all take a personal optimal strategy to maximize their utilities. The renter's strategy space is $\{A_{J,1}, A_{J,2}\}$, where $A_{J,1}$ means the renter cooperates with the tenant and $A_{J,2}$ means the renter refuses to cooperate with the tenant. The tenant's strategy space is$\{A_{I,1}, A_{I,2}\}$, where $A_{I,1}$ stands for the tenant cooperates with the renter and $A_{I,2}$ stands for the tenant refuses to cooperate with the renter.

Assumption 3: Renters cannot be fully trusted. For a renter, dishonest behaviors in this collaborative process can affect its own trust value. For a renter with low reputation, its tenant has a high possibility to suffer loss.

Assumption 4: In this model, both the tenant i and the renter j want to make a profit from the rental.

Assumption 5: We assume that the subscribers of any operator pay the same unit resource usage fee.

Assumption 6: The tenant selects the renter that can provide it with sufficient resources it requested.

In our analysis, we consider the impact of the trust value on operator cooperation. We assume that there is a certain security risk to use the shared resources of network provided by a renter, which may cause some loss to tenant in this cooperation (i.e., $lo_i(t)$). The trust value of the renter represents the security degree of network resources provided by it. In other words, the higher the trust value of the renter, the smaller the tenant's loss. However, the higher the trust value of the renter, the more expensive its maintenance and operation costs. The renter need to pay more maintenance fee to obtain a higher trust value.

J denotes the total amount of resources of renter j, among which there are J' redundant resources that can be rented. ug(t) is the unit resource usage payment by subscribers at time t. $uf_j(t)$ stands for the basic unit operation and maintenance cost of renter j at time t in a normal situation (e.g., without security and trust enhancement technologies applied). $TV_j(t)$ is the trust value of renter j, which requires renter j to spend more maintenance fee to maintain a higher trust value. If renter j does not rent its redundant resources, its utility at time t can be described as (3):

$$u_J^j(t) = ug(t) \times \left(J - J'\right) - uf_j(t) \times J \times \left(1 + TV_j(t)\right). \tag{3}$$

Imagine tenant i renting some resources I' from renter j. Renter j obtains rental benefit $rf(t) \times I' \times \left(1 + TV_j(t)\right)$ from tenant i at time t, where $rf(t)$ is the basic unit resource rental fee and $rf(t) \times I' \times TV_j(t)$ stands for the extra benefit that

TABLE 5. Utility matrix with trust.

	$A_{I,1}$	$A_{I,2}$
$A_{J,1}$	$u_J^j(t) = ug(t) \times (J - J') - uf_j(t) \times (1 + TV_j(t)) \times J + rf(t) \times I' \times (1 + TV_j(t)),$ $u_I^i(t) = ug(t) \times I - uf_i(t) \times I \times (1 + TV_i(t)) + (ug(t) - rf(t) \times (1 + TV_j(t))) \times I' - (1 - TV_j(t)) \times lo_i(t).$	$u_J^j(t) = ug(t) \times (J - J') - uf_j(t) \times (1 + TV_j(t)) \times J + rf(t) \times I' \times (1 + TV_j(t)),$ $u_I^i(t) = ug(t) \times I - uf_i(t) \times I \times (1 + TV_i(t)).$
$A_{J,2}$	$u_J^j(t) = ug(t) \times (J - J') - uf_j(t) \times J \times (1 + TV_j(t)),$ $u_I^i(t) = ug(t) \times I - uf_i(t) \times I \times (1 + TV_i(t)) + (ug(t) - rf(t) \times (1 + TV_j(t))) \times I' - (1 - TV_j(t)) \times lo_i(t).$	$u_J^j(t) = ug(t) \times (J - J') - uf_j(t) \times J \times (1 + TV_j(t))$ $u_I^i(t) = ug(t) \times I - uf_i(t) \times I \times (1 + TV_i(t))$

tenant i's trust value brings. Hence, the utility of renter j at time t can be calculated as (4):

$$u_J^j(t) = ug(t) \times \left(J - J'\right) - uf_j(t) \times \left(1 + TV_j(t)\right) \times J + rf(t) \times I' \times \left(1 + TV_j(t)\right). \quad (4)$$

I is the total amount of resources of tenant i, the resources of which are insufficient to satisfy its subscribers' demands. $uf_i(t)$ stands for the basic unit cost of tenant i for operation and maintenance. We conclude the If it does not obtain extra resources, its utility at time t can be described as (5):

$$u_I^i(t) = ug(t) \times I - uf_i(t) \times I \times (1 + TV_i(t)). \quad (5)$$

Renter j has extra resources, so tenant i can obtain extra resources I' from the renter when they make an agreement. It obtains resource usage payment $ug(t) \times I'$ from its subscribers at time t, while it has to pay rental fee $rf(t) \times I' \times TV_j(t)$ to renter j. $lo_i(t)$ is the possible total loss of tenant i at time t. The higher the trust value of renter, the less the tenant's loss. Hence, the utility $u_I^i(t)$ of tenant i that successfully rents I' resources from renter j at time t can be concluded as (6):

$$u_I^i(t) = ug(t) \times I - uf_i(t) \times I \times (1 + TV_i(t)) + \left(ug(t) - rf(t) \times \left(1 + TV_j(t)\right)\right) \times I' - \left(1 - TV_j(t)\right) \times lo_i(t). \quad (6)$$

The utility matrix is summarized in Table 5.

$$u_I^i(t) = ug(t) \times I - uf_i(t) \times I \times (1 + TV_i(t)) + \left(ug(t) - rf(t) \times \left(1 + TV_j(t)\right)\right) \times I' - \left(1 - TV_j(t)\right) \times lo_i(t).$$

According to the pure strategy Nash equilibrium (NE) analysis, we get the optimal solution by referring to tenant i.

There are two scenarios according to whether the value of $\left(ug(t) - rf(t) \times \left(1 + TV_j(t)\right)\right) \times I' - \left(1 - TV_j(t)\right) \times lo_i(t)$ is bigger than 0 or not.

1) When it is bigger than 0, according to the line method, we calculate the pure strategy NE is achieved when both players choose to cooperate.
2) When it is non-positive, according to the line method, the pure strategy Nash equilibrium is achieved when the renter chooses cooperation and the tenant chooses defection.

We also calculate the mixed strategy Nash equilibrium with trust. We denote α to be the probability that renter j collaborates with tenant i, and β is the probability that tenant i collaborates with renter j.

Given α, the expected utility of tenant i when it chooses cooperation ($\beta = 1$) and rejects cooperation ($\beta = 0$) are:

$$\prod I(\alpha, 1) = (ug(t) - uf_i(t) \times (1 + TV_i(t))) \times I + (ug(t) - rf(t) \times (1 + TV_j(t))) \times I' - (1 - TV_j(t)) \times lo_i(t).$$

$$\prod I(\alpha, 0) = (ug(t) - uf_i(t) \times (1 + TV_i(t))) \times I.$$

Given β, the expected utilities of renter j when it chooses cooperation ($\alpha = 1$) and rejects cooperation ($\alpha = 0$) are:

$$\prod J(1, \beta) = ug(t) \times (J - J') - uf_j(t) \times (1 + TV_j(t)) \times J + rf(t) \times I' \times (1 + TV_j(t)).$$

$$\prod J(0, \beta) = ug(t) \times (J - J') - uf_j(t) \times J \times (1 + TV_j(t)).$$

1) When $ug(t) \times I' - rf(t) \times (1 + TV_j(t)) \times I' - (1 - TV_j(t)) \times lo_i(t) > 0$, the mixed strategy NE of this game is the same as the pure strategy NE. To be specific, NE is achieved when both the renter and the tenant choose to cooperate.
2) When $ug(t) \times I' - rf(t) \times (1 + TV_j(t)) \times I' - (1 - TV_j(t)) \times lo_i(t) \leq 0$, we can get the following result: $\prod I(\alpha, 1) \leq \prod I(\alpha, 1)$. This means, it is better for tenant i to reject cooperation. Since $rf(t) \times I' \times (1 + TV_j(t))$ is a positive value, we can get the following result: $\prod J(0, \beta) \leq \prod J(1, \beta)$. Therefore, it is beneficial for renter ðÍ′— to choose cooperation.

In general, the renter's dominant strategy is cooperation, which means no matter what actions the tenant chooses, the renter can always obtain more profits by cooperation. On the other hand, the tenant chooses strategies with regard to the value of $ug(t) \times I' - rf(t) \times (1 + TV_j(t)) \times I' - (1 - TV_j(t)) \times lo_i(t)$. Obviously, the cooperation condition of the renter and the tenant is $(lo_i(t) - rf(t) \times I') \times TV_j(t) + ug(t) \times I' - rf(t) \times I' - lo_i(t) > 0$. When $TV_j(t) = 0$, big loss will make the tenant have no profit. When $TV_j(t) = 1$, it is possible to set proper $rf(t)$ and $ug(t)$ to benefit both parties.

Based on the above analysis, we can conclude that one operator can offer rental services to other operators to obtain extra benefits if extra resources can be located. So normal operators have a willingness to rent their network resources. However, if the reputation of the renter is low and its network resources are not secure, the tenant's potential loss could be more than that the tenant can endure or more than the benefits that the tenant can obtain, then the tenant would not choose the renter to cooperate. In this situation, the tenant may prefer to choose a renter that can make it have low loss and high utility. Therefore, the tenant will choose those renters that have a high trust value to reduce its potential losses. The less the loss that a tenant expects in the cooperation process with the renter, the higher the trust of the renter should provide. The only downside is that the tenant's utility will be less than before since it need to pay more rental fee (refer to (6)). For the renter, if it wants to attract more tenants and get more benefits, the simple approach is to increase the security of its own network resources to gain high trust, which of course will cost the renter more.

V. CONCLUSION AND FUTURE WORK

In this paper, we proposed the IOCS to support multiple operators to share resources in a trustworthy and secure way based on the TCP technology in the infrastructure of C-RAN. Trust cooperation across multiple cellular networks is flexibly supported. It achieves trust attestation and cooperation among different VBS pools by applying RTM to monitor VBS pool configuration and ensure their changes compatible with trust policy. IOCS meets the demands of the 5G. It effectively reduces the TCO and roaming price by sharing and balancing the entire resources in the network. Performance evaluation and simulation results show that the IOCS has high operation efficiency and trust attestation accuracy. The game theoretical analysis further provides the utilities of renter operators and tenant operators, as well as their cooperation condition in the IOCS with their trust being considered.

Regarding the future work, we plan to further optimize our scheme in the following ways. We are ready to propose more effective and safe rental strategies in order to help inter-operator cooperation. For example, through blockchain technology, the traceability and non-repudiation of transactions among different operators can be guaranteed. Meanwhile, trustworthy trust evaluation can be further provided with high quality and granularity.

REFERENCES

[1] Marketing Charts. *Mobile Network Operators Face Cost Crunch*. Accessed: Jan. 20, 2018. [Online]. Available: http://www.marketingcharts.com/wp/direct/mobile-networkoperators-face-cost-crunch-17700

[2] Juniper Research. *Press Release: Mobile Network Operator Revenues*. Accessed: Jan. 20, 2018. [Online]. Available: http://juniperresearch.com/viewpressrelease.php?pr=245

[3] J. Mitola, "Cognitive radio for flexible mobile multimedia communications," in *Proc. IEEE Int. Workshop Mobile Multimedia Commun. (MoMuC)*, San Diego, CA, USA, Nov. 1999, pp. 3–10.

[4] I. Chih-Lin, C. Rowell, S. Han, Z. Xu, G. Li, and Z. Pan, "Toward green and soft: A 5G perspective," *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 66–73, Feb. 2014.

[5] J. Wu, Z. Zhang, Y. Hong, and Y. Wen, "Cloud radio access network (C-RAN): A primer," *IEEE Netw.*, vol. 29, no. 1, pp. 35–41, Jan. 2015.

[6] F. Y. Tian, P. Zhang, and Z. Yan, "A survey on C-RAN security," *IEEE Access*, vol. 5, pp. 13372–13386, 2017.

[7] B. Niu, Y. Zhou, H. Shah-Mansouri, and V. W. Wong, "A dynamic resource sharing mechanism for cloud radio access networks," *IEEE Trans. Wireless Commun.*, vol. 15, no. 12, pp. 8325–8338, Dec. 2016.

[8] X. Huang, G. Xue, R. Yu, and S. Leng, "Joint scheduling and beamforming coordination in cloud radio access networks with QoS guarantees," *IEEE Trans. Veh. Technol.*, vol. 65, no. 7, pp. 5449–5460, Jul. 2016.

[9] D. B. Rawat, S. Shetty, and K. Raza, "Secure radio resource management in cloud computing based cognitive radio networks," in *Proc. Int. Conf. Parallel Process. Workshops*, Pittsburgh, PA, USA, 2012, pp. 288–295.

[10] J. Ren, L. Liu, D. Zhang, Q. Zhang, and H. Ba, "Tenants attested trusted cloud service," in *Proc. IEEE 9th Int. Conf. Cloud Comput.*, San Francisco, CA, USA, Jun./Jul. 2016, pp. 600–607.

[11] Z. Yan and P. Cofta, "A mechanism for trust sustainability among trusted computing platforms," in *Proc. Int. Conf. Trust, Privacy Secur. Digit. Bus. (TrustBus)*, Zaragoza, Spain, 2004, pp. 11–19.

[12] Z. Yan, *Trust Management in Mobile Environments: Autonomic and Usable Models*. Hershey, PA, USA: IGI Global, 2013.

[13] "C-RAN: The road towards green RAN," China Mobile Res. Inst., Beijing, China, White Paper Version 3.0, Oct. 2011.

[14] Y. Zou, J. Zhu, X. Wang, and L. Hanzo, "A survey on wireless security: Technical challenges, recent advances, and future trends," *Proc. IEEE*, vol. 104, no. 9, pp. 1727–1765, Sep. 2016.

[15] Z. Tari, X. Yi, U. S. Premarathne, P. Bertok, and I. Khalil, "Security and privacy in cloud computing: vision, trends, and challenges," *IEEE Cloud Comput.*, vol. 2, no. 2, pp. 30–38, Mar./Apr. 2015.

[16] Z. Xiao and Y. Xiao, "Security and privacy in cloud computing," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 2, pp. 843–859, 2nd Quart., 2012.

[17] X. Wang, Q, Wang, X. Hu, and J. Lu, "Security technology in virtualization system: State of the art and future direction," in *Proc. IET Int. Conf. Inf. Sci. Control Eng. (ICISCE)*, Shenzhen, China, 2012, pp. 1–7.

[18] R. L. Kay, *Trusted Computing is Real and It's Here, White paper presented on Trusted Computing Group Website*. Accessed: Mar. 23, 2018. [Online]. Available: https://www.trustedcomputinggroup.org

[19] Trusted Computing Group. Accessed: Mar. 26, 2018. [Online]. Available: https://www.trustedcomputinggroup.org

[20] Trusted Computing Group. *TPM Specification Version 1.2*. Accessed: Mar. 26, 2018. [Online]. Available: http://www.trustedcomputinggroup.org/specs/TPM/

[21] Trusted Computing Group. *TCG Software Stack Specification Version 1.2*. Accessed: Apr. 10, 2018. [Online]. Available: https://www.trustedcomputinggroup.org/specs/TSS/

[22] R. Perez, R. Sailer, and L. van Doorn, "vTPM: Virtualizing the trusted platform module," in *Proc. 15th Conf. USENIX Secur. Symp.*, Vancouver, BC, Canada, 2006, Art. no. 21.

[23] F. J. Krautheim, "Private virtual infrastructure for cloud computing," in *Proc. Conf. Hot Topics Cloud Comput. (HotCloud)*, San Diego, CA, USA, 2009, Art. no. 5.

[24] A.-R. Sadeghi and C. Stüble, "Property-based attestation for computing platforms: Caring about properties, not mechanisms," in *Proc. Workshop New Secur. Paradigms*, Halifax, NS, Canada, Sep. 2004, pp. 67–77.

[25] A.-R. Sadeghi, C. Stüble, and M. Winandy, "Property-based TPM virtualization," in *Proc. Int. Conf. Inf. Secur.*, in Lecture Notes in Computer Science, vol. 5222, Sep. 2008, pp. 1–16.

[26] A. Nagarajan and V. Varadharajan, "Dynamic trust enhanced security model for trusted platform based services," *Future Gener. Comput. Syst.*, vol. 27, no. 5, pp. 564–573, May 2011.

[27] O. Sefraoui, M. Aissaoui, and M. Eleuldj, "OpenStack: Toward an open-source solution for cloud computing," *Int. J. Comput. Appl.*, vol. 55, no. 3, pp. 38–42, Oct. 2012.

[28] A. Corradi, M. Fanelli, and L. Foschini, "VM consolidation: A real case based on OpenStack Cloud," *Future Gener. Comput. Syst.*, vol. 32, pp. 118–127, Mar. 2014.

[29] M. Strasser and P. E. Sevnic, "A software-based TPM emulator for Linux," M.S. thesis, Dept. Comput. Sci., Swiss Federal Inst. Technol. Zurich, ETH Zürich, Zürich, Switzerland, 2004.

[30] C. Liu, J. Lin, and B. Fang, "T-YUN: Trustworthiness verification and audit on the cloud providers," *IEICE Trans. Inf. Syst.*, vol. 96, no. 11, pp. 2344–2353, Nov. 2013.

[31] R. Neisse, D. Holling, and A. Pretschner, "Implementing trust in cloud infrastructures," in *Proc. 11th IEEE/ACM Int. Symp. Cluster, Cloud Grid Comput.*, Washington, DC, USA, May 2011, pp. 524–533.

[32] G. Xu, Y. Tang, Z. Yan, and P. Zhang, "TIM: A trust insurance mechanism for network function virtualization based on trusted computing," in *Proc. 10th Int. Conf. Secur., Privacy Anonymity Comput., Commun. Storage*, Dec. 2017, pp. 139–152.

[33] L. Gao, Z. Yan, and L. T. Yang, "Game theoretical analysis on acceptance of a cloud data access control system based on reputation," *IEEE Trans. Cloud Comput.*, to be published, doi: 10.1109/TCC.2016.2632110.

[34] X. Liang and Z. Yan, "A survey on game theoretical methods in human–machine networks," *Future Gener. Comput. Syst.*, to be published, doi: 10.1016/j.future.2017.10.051.

[35] Y. Shen, Z. Yan, and R. Kantola, "Analysis on the acceptance of global trust management for unwanted traffic control based on game theory," *Comput. Secur.*, vol. 47, pp. 3–25, Nov. 2014.

[36] A. Al-Dhanhani, R. Mizouni, H. Otrok, and A. Al-Rubai, "Game theoretical analysis of collaborative social applications," in *Proc. Int. Conf. Collaborative Comput., Netw., Appl. Worksharing (CollaborateCom)*, Pittsburgh, PA, USA, Oct. 2012, pp. 628–634.

[37] A. Al-Dhanhani, R. Mizouni, H. Otrok, and A. Al-Rubai, "A game theoretical model for collaborative groups in social applications," *Expert Syst. Appl.*, vol. 41, no. 11, pp. 5056–5065, Sep. 2014.

[38] *Intel SGX*. Accessed: May 23, 2018. [Online]. Available: https://software.intel.com/en-us/sgx

FENGYU TIAN (M'18) received the B.Eng. degree in telecommunications engineering from the Henan University of Science and Technology, Luoyang, China, in 2015, and the M.Eng. degree in electronics and communication engineering from Xidian University, Xi'an, China, in 2018, respectively. His research interests are in security, privacy, and trust management in 5G network.

ZHENG YAN (M'06–SM'14) received the B.Eng. degree in electrical engineering and the M.Eng. degree in computer science and engineering from Xi'an Jiaotong University, Xi'an, China, in 1994 and 1997, respectively, the M.Eng. degree in information security from the National University of Singapore, Singapore, in 2000, and the Lic.Sci. and D.Sc. degrees in technology in electrical engineering from the Helsinki University of Technology, Helsinki, Finland. Before joining academia in 2011, she has been a Senior Researcher with the Nokia Research Center, Helsinki, Finland, since 2000. She is currently a Professor with Xidian University, China, and a Visiting Professor and Finnish Academy Research Fellow with Aalto University, Finland. Her research interests are in trust, security, privacy, and security-related data analytics. She served as a general chair or program chair for a number of international conferences, including the IEEE TrustCom 2015. She is a Founder Steering Committee Co-Chair of the IEEE Blockchain Conference. She received several awards, including the 2017 Best Journal Paper Award issued by the IEEE Communication Society Technical Committee on Big Data and the Outstanding Associate Editor of 2017 for the IEEE Access. She is an Associate Editor of the IEEE Internet of Things Journal, *Information Fusion*, *Information Sciences*, the IEEE Access, and JNCA. She is an inventor of 19 patents, all of them having been adopted in industry.

XUEQIN LIANG received the B.Sc. degree in applied mathematics from Anhui University, Anhui, China, 2015. She is currently pursuing the Ph.D. degree with Xidian University, Xi'an, China, and Aalto University, Finland. Her research interests are in game theory-based security solutions, cloud computing security and trust, and IoT security.

PENG ZHANG received the Ph.D. degree in computer and communication engineering from the Beijing University of Posts and Telecommunications, China. He conducted his Post-Doctoral Research with the Helsinki University of Technology (1999–2001). He is currently a Computer Scientist with an interest in trust and mobile services. He achieved over 60 paper publications and invented 10 granted patents. He also served as an organization committee member for numerous international conferences and a reviewer for many prestigious journals.