Evaluation Methodology for Virtual Base Station Platforms in Radio Access Networks

Lin Tian†, Senior Member, IEEE, Yiqing Zhou‡, Senior Member, IEEE, Yuanyuan Wang*, Jun Yang*, Qian Sun*, Member, IEEE, Jinhong Yuan†, Fellow, IEEE, Bingqiang Yang‡

Abstract—Virtual Base Stations (BS) with scalable resources and flexible functions are necessary for the fifth-generation (5G) and future mobile systems. The virtualization platform is the fundamental component of virtual BSs and plays a very important role in determining BS performance. As BS virtualization platforms can be provided by both conventional BS manufactures and third-party software vendors, it is crucial to define unified methodology to evaluate the performance of different virtualization platforms. In this paper, for the first time in literature, we design a set of comprehensive performance metrics and propose evaluation methods for BS virtualization platforms. More specifically, there are two types of metrics, i.e., micro level and macro level metrics. Micro level metrics quantify the virtualization platforms’ basic performances including real-time performance and virtualization overhead. Macro level metrics quantify the platforms ability to support BS resource and function virtualization while satisfying the radio access network requirements. Correspondingly, two-level evaluation methodology is designed to collect micro and macro level metrics from different parts of a virtual BS by running proper workloads. Finally, using the proposed methodology, two representative virtualization platforms, i.e., KVM hypervisor and Virtual BS platform are evaluated. These experiments verify that the proposed methodology can effectively evaluate the performance of different platforms to provide a valuable reference for the selection of virtual BS platforms. Moreover, the proposed evaluation methodology is a significant step toward constructing an open ecosystem for radio access networks.

Index Terms—5G, Virtual Base Station, Virtualization Platforms, Performance Metrics, Evaluation Method

I. INTRODUCTION

Nowadays, there is an explosively increasing number of emerging mobile services, which require wireless networks to support a much broader range of usage scenarios such as Internet of Things, automatic driving, intelligent manufacturing, mobile cloud, et.al. In particular, 3GPP has defined three types of usage scenarios for 5G mobile networks, which are eMBB (enhanced Mobile BroadBand), mMTC (massive Machine Type Communications) and URLLC (Ultra-Reliable and Low Latency Communications) [1]. It is predicted that with the continuous emerging of new services there will be more usage scenarios to be supported in 5G [2] [3]. Obviously, different BS functions are required in different scenarios. For example, the BS functions for mMTC scenarios usually do not include mobile tracking, scheduling, HARQ, etc., while these functions are important for eMBB scenarios [4]. If we adopt the conventional BS development mode where software and hardware are tightly coupled, two key challenges have to be tackled, which are

1) High BS manufacturing cost. Fig. 1 illustrates the conventional BS architecture where all eMBB BS, mMTC BS, 4G BS, etc. should be developed separately to support different usage scenarios. This is because a conventional BS has specific functions which cannot be flexibly changed. Therefore, the BS manufacturing cost is high. If mobile operators want to deploy new types of services which are not supported by the current BSs, new BSs with a complete set of hardware and software have to be purchased and installed which results in high network operational cost.

2) Low BS resource utilization. Note that with the conventional BS architecture, each BS must be designed with high capacity to support the maximum possible traffic load in its cell. However, BSs’ traffic load varies throughout a day like a tide [5]. When a BS’s traffic load is low, most of its capacity is wasted because the physical resource of BSs cannot be shared among different BSs since they are isolated from each other [6], as shown in Fig. 1.

Fig. 1: Architecture of conventional BSs

Therefore, the conventional BS development mode with fixed capacity and functions is not suitable for 5G systems and future radio access networks. In order to solve the above problems and challenges, it is highly desirable that BS functions should be orchestrated on demand [7] and resources should be shared between different BSs. Inspired by virtualization technique in computer science, virtual Base Stations have been proposed to provide functionality of a physical BS without

* Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100190, China; Key laboratory of Mobile Computing and Pervasive Device, Beijing, 100080, China.
† University of New South Wales, Sydney, NSW 2052, Australia
‡ Huawei Technologies Co., LTD., Shenzhen, China.

Corresponding Author: Lin Tian (e-mail: tianlindd@ict.ac.cn)
direct correspondence to any real hardware [6][8]. Different virtual BSs can run on the same physical hardware platform in order to share processing resources (CPU, Memory, I/O etc.) and wireless resources (spectrum etc.) dynamically [9]. Moreover, virtual BSs can be created, upgraded, changed or deleted on demand. Recently, the virtual BS with function virtualization and resource virtualization has become a hot research topic and corresponding devices have been developed. For example, Cloud RAN [10], LightRadio [10] and Super Base Station [6] are all the new 5G BSs with virtual BS characteristics.

A representative architecture of virtual BSs is illustrated in Fig. 2. In virtual BSs, the physical resources including RRH, CPU, storage, etc. are aggregated and form the virtualization resources pool. A virtual BS can be created, upgraded, changed or deleted by managing and orchestrating the virtual BS functions and resources dynamically. For example, in order to create a new virtual BS, the control center should select wireless and processing resources from the virtualization resources pool to construct the virtual BS infrastructure, then orchestrate an appropriate set of BS functions to run on the virtual BS infrastructure according to the network requirements. It can be seen from the architecture that the virtualization platform is the fundamental part of virtual BSs. The platform provides underlying hardware-independent virtualization resource pool for virtual BSs by physical resource abstraction, reconstruction and isolation. Therefore, both the functions and scales of virtual BSs can be defined flexibly based on the virtualization platform.

Since virtualization platforms for BSs can be provided by both conventional BS manufactures and third-party software vendors, comprehensive performance metrics definitions and evaluation methods for virtualization platforms are required urgently, which can be used not only to guide the design of efficient and flexible virtual BS systems, but also to compare performance of various platform products. Although there are performance metrics definitions for computer virtualization [12], we should reconsider the metrics and evaluation methods for virtualization platforms in BSs because of two main reasons. One reason is that the workloads are quite different between virtual BSs and virtual computers. In virtual BSs, the main workloads are communication-related such as BS protocol and baseband processing, while in virtual computers they are computation-related such as file servers, high performance computing tasks and so on. The other reason is that the requirements of virtual BSs are different with computers. For example, the real-time requirements of virtual BSs are much higher than that of virtual computers. Moreover, some important performances of BSs such as data rate, control/user plane latency do not exist in computers. Therefore, this paper analyzed and defined performance metrics of virtualization platforms to satisfy the requirements of BSs such as real-time, efficiency and so on, designed two-level evaluation methodology for measuring and calculating the new metrics, and implemented some experiments to verify the feasibility of virtualization platforms for BSs.

The remainder of the paper is organized as follows. Firstly, the implementation of BS virtualization platforms is discussed. Then we design the performance metrics and evaluation methods for BS virtualization platforms. Performance testing results of two representative virtualization platforms are reported. Finally, conclusions are drawn.

II. OVERVIEW OF BS VIRTUALIZATION PLATFORMS

In computing, virtualization refers to the act of creating virtual versions of computer hardware platforms, operation systems, storage devices and computer network resources [13]. The important feature of the virtual resources is underlying hardware-independent. Similar to the virtualization in computer science, the target of BS virtualization platform is to abstract the physical resources of BSs and organize them to one or more virtual forms that can be used by different BSs more efficiently and flexibly. As a result, the BS logical functions and the underlying hardware are decoupled. For instance, the virtual BS can occupy several CPU cores when its traffic load is high, while it will share one CPU core with other virtual BSs when the traffic load is low.

Fig. 3 depicts the framework of BS virtualization platform. BS virtualization platform is mainly comprised of three parts: physical resource abstraction, virtual resource pool management and virtual BS management. In the physical resource abstraction part, different types of hardware devices, such as CPU, Memory, network interface, accelerator, etc. are decoupled. There is a corresponding hardware resource abstraction entity on top of each device, which is used to manage the corresponding device and provide a unified operational interface.
to the upper layer. In the virtual resource pool management part, the physical resources are aggregated to form virtual resource pools, i.e., computing resource pool, storage pool, spectrum pool, et al. The resource mapping between the virtual resource pool and physical resources are maintained in this part. The virtual BS management part is responsible for the virtual BS lifecycle and migration management. For example, virtual BSs will be created by organizing and scheduling different virtual resources according to the network requirements in this part.

Fig. 3: Framework of BS Virtualization Platform

There are two different ways to implement BS virtualization platforms. One is based on the platforms widely used in the computer field [14] [15], which can be named as IT-based virtualization platform. The other is designed for BS specially [6], which is named as BS-specific virtualization platform. In both IT-based and BS-specific virtualization platforms, wireless resources can be split into slices for sharing among virtual BSs dynamically in order to achieve high resource efficiency [16][17]. Each virtual BS owns wireless resources in the multi-domain of space, time, frequency, etc. In fact, wireless resource virtualization can be viewed as an extension of dynamic radio resource allocation or sharing [16], which has been studied widely in radio access networks. In the following, we will mainly illustrate the differences of processing resource virtualization in the two types of platforms.

A. IT-based virtualization platform

The virtualization platform used in the computer field is a set of software between virtual machines (VMs) and the underlying hardware, which hides the physical characteristics of hardware to provide an abstract, unified and virtual computing platform. There are two main types of virtualization platforms in the computer field, which are hypervisors and containers. The main difference between them is the level of abstraction in terms of virtualization and isolation. A hypervisor, or called virtual machine monitor (VMM), provides the abstraction of physical hardware, while a container implement isolation of processes at the operation system (OS) level [14]. As shown in Fig. 4(a), the hypervisor isolates the VM from the underlying host system and each VM needs a complete implementation of a guest OS including the binaries and libraries necessary for applications. In contrast, the container enables applications to run on the same OS kernel so the guest OS is not required as shown in Fig. 4(b). Therefore, the container is a lightweight alternative to the hypervisor by excluding the execution of guest OS, hardware virtualization, etc. At the same time, the hypervisor has better performances in isolation, security and supporting different guest OSs because each virtual machine is standalone and independent of the host kernel. The common hypervisors include VMware’s ESX, Microsoft’s Hyper-V, open source Xen, and KVM, etc. The representatives of containers are Docker and LXC. However, the virtualization platforms used in the computer field may not satisfy the real-time requirements of wireless communication systems because the virtualization brings extra processing delay to OS kernels. For instance, the kernel latency jitter of XEN exceeds 1ms sometimes [14] while the processing time of each 5G frame should be less than 1ms. So extra optimizations should be done when IT-based virtualization platforms are deployed in virtual BSs.

B. BS-specific virtualization platform

Although IT-based virtualization platforms are widely used, BS-specific virtualization platforms are designed for the following main reasons. First, the processing resources of most BSs are heterogeneous which not only include general purpose processors (GPP) but also include DSPs or FPGAs as accelerators, while IT-based virtualization platforms are usually used for GPPs. Second, IT-based virtualization platforms need many modifications for BSs to guarantee real-time requirements, so customized virtualization platforms for BSs may be more efficient [6]. Therefore, the targets of BS-specific
virtualization platforms are to maintain the mapping between the virtualization resource pool and heterogeneous computing resources, and to provide unified operational interfaces to construct virtual BSs in a fast and predictable manner. As shown in Fig. 4(c), BS-specific virtualization platforms schedule the BS-related processing tasks such as protocol, baseband processing, etc., to appropriate processors, manage the shared memory and network connections of virtual BSs. So BS-specific virtualization is not the general concept of the resource virtualization, and it realizes virtual BSs by resource sharing among BS processing tasks dynamically. In a word, BS-specific virtualization platforms operate at the task level, which is different from hypervisors’ hardware-level and containers’ OS-level. Because different kinds of hardware resources can be scheduled directly without the interruption of operating systems in the task-level virtualization, the processing delay brought by virtualization is minimized to guarantee BS’s real-time performance. Moreover, the shared resources in the platforms such as CPU, FPGA, DSP, memory, network connections, et al. are BS-specific. The virtualization platform of Super BS [6] is a representative of BS-specific platforms.

III. PERFORMANCE METRICS AND EVALUATION METHODS

To verify, compare and select virtualization platforms for BSs, appropriate performance metrics should be defined. There are two main criteria to choose the metrics. First, the main target of the metrics should reflect the virtualization platforms’ abilities to support virtual BSs. Second, the metrics should be fair for comparing virtualization platforms from different providers, regardless of how they implement the platforms. For example, some providers use hypervisors to implement platforms, whereas others use container virtualization technologies. The metrics and corresponding evaluation methods should be independent of these implementation details. Following the above criteria, we propose various metrics to comprehensively evaluate virtualization platforms performance (see Table I). These metrics are classified into two major classes, micro level and macro level.

A. Micro Level

Micro level metrics quantify the virtualization platforms’ basic performances such as task schedule latency, virtualization overhead, etc. Since some metrics are standard in virtual machines, we can use the classic benchmarks [12] [14][15] to evaluate these metrics directly.

Real-time performance: In order to satisfy the real-time requirements of RAN, the BSs’ tasks have strict processing deadlines. However, virtualization platforms usually bring extra processing delay to OS kernels. Therefore, the real-time performance of virtualization platforms should be measured when used in RAN systems. To evaluate the real-time performance, we can use the tool Cyclictest which is widely used to obtain the metrics including kernel latency and kernel latency jitter.

Virtualization overhead: Since any virtualization solutions are not resource-free, their overhead would lead to negative impacts on BSs. The overhead can be measured by comparing the workload’s performance of a virtualization platform with that of a physical machine running the same configuration. To comprehensively evaluate the virtualization overhead, we not only select the classic workloads with different characteristics including CPU-, memory-, disk I/O-, network I/O-intensive workloads [12][15], but also use BS-related workloads including protocol, baseband processing, etc. Moreover, we use Overhead Ratio $O_p$ to measure the virtualization overheads, which is defined as follows [15]

$$O_p = \left| \frac{P_m - P_b}{P_b} \right| \times 100\%$$

where $P_m$ denotes the workload’s performance in the virtualization platform, $P_b$ indicates the workload’s baseline performance in the physical machine.

B. Macro Level

Macro level metrics quantify virtualization platforms’ ability to support virtual BSs running correctly and efficiently. These metrics reflect the BS resource and function virtualization performance of a virtualization platform while satisfying the RAN requirements.

BS requirements satisfaction: It is a fundamental requirement that virtualization platforms can ensure the virtual BS running as a physical BS. In other words, the virtual BS should satisfy the radio access performance requirements defined by certain standards such as 3GPP 36.913 [18] for 4G and 3GPP 38.913 [1] for 5G. Satisfaction Degree $D_s$ is defined as 1 or 0, representing whether or not the BS requirements are satisfied, respectively. Despite all the requirements defined by standards can be tested, in order to simplify the evaluation process, we recommend to evaluate the representative performances, such as peak data rate, throughput, control plane/user plane latency, etc. The reason is that these performances are affected by virtualization technologies more significantly than others, such as coverage and mobility.

BS resource virtualization: One of the most important characteristics of virtualization platforms is to support BS resource virtualization. The resources of virtual BSs running on the same virtualization platform can be shared dynamically, so the virtual BSs can scale up and down based on demand to improve the resource efficiency. To measure the BS resource virtualization performances, hardware resource utilization rate (RUR) and BS performance slowdown (PS) during scaling are defined. RUR is a metric to quantify the resource efficiency of virtualization platforms, which can be calculated by averaging CPU occupation, memory occupation, etc. Although the resource efficiency can be optimized by some novel virtual BS resource allocation algorithms [19], RUR reflects the basic efficiency performance brought by the virtualization platform without any optimized algorithms. Moreover, it is better not to slow down the BS performance, for instance, reduce the BS throughput, when the virtualization platforms changing resources of a BS. Thus, PS is defined to reflect whether the BS performance is affected when the BS resources are scaling up or down.
TABLE I: Virtualization Platform Performance Metrics

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
<th>Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real-time performance</td>
<td>Measures the OS kernels delay of virtualization platforms</td>
<td>Kernel latency, Kernel latency jitter</td>
</tr>
<tr>
<td>Virtualization overhead</td>
<td>Measures virtualization overhead using various workload (CPU, Memory, Network, Disk, BS-related)</td>
<td>Overhead Ratio</td>
</tr>
<tr>
<td>BS requirements satisfaction</td>
<td>Measures whether virtual BSs satisfy the radio access performance requirements defined by standards</td>
<td>Satisfaction Degree</td>
</tr>
<tr>
<td>BS resource virtualization</td>
<td>Measures the resource efficiency performance of virtualization platforms</td>
<td>Hardware resource utilization rate, BS performance slowdown during scaling</td>
</tr>
<tr>
<td>BS live migration</td>
<td>Measures the migration performance of virtual BS</td>
<td>Total migration time, Migration downtime</td>
</tr>
<tr>
<td>BS function virtualization</td>
<td>Measures the ability of virtualization platforms to support BS function virtualization</td>
<td>Virtual BS creating latency</td>
</tr>
</tbody>
</table>

**BS live migration**: In order to improve energy and resource efficiency, the virtualization platform can migrate a virtual BS from one physical processing unit, for example, a server to another. Downtime in the migration process can cause temporary communication interruption. If the communication interruption period is longer than some threshold, users will consider it a radio link failure and trigger a new connection with BS. Therefore, virtualization platforms should improve the live migration performance to reduce the negative effect on the communication between users and BSs. Total migration time and downtime are main metrics to quantify the performance of BS live migration. Total migration time is the time from start of the BS live migration process until the virtualization platform notifies that the source resources can be free. Downtime is the time period that BS stop working during migration process.

**BS function virtualization**: Virtualization platforms decouple the BS functionalities and physical equipment, so the modular BS functions which are defined as Virtual Network Functions (VNF) [7] can be orchestrated to form new virtual BS based on the requirements of different usage scenarios, such as mobile broadband, auto-driving, etc. In order to evaluate the ability of virtualization platforms to support BS function virtualization, virtual BS creating latency is defined. We measure this latency as the time period for a platform to construct a new virtual BS. At the same time, we should evaluate whether the new virtual BS satisfies the radio access requirements with the methods defined in 'BS requirements satisfaction' above.

**C. Two-level evaluation methodology**

Because the defined performance metrics for BS virtualization platforms are classified into micro level and macro level, a two-level performance evaluation methodology is designed to comprehensively analyze the platform performance. As illustrated by Fig. 5, micro and macro level metrics are measured in different parts of the virtual BS. In order to evaluate the micro level metrics, benchmark workloads should be run on the platforms to obtain the underlying performance such as kernel latency, virtualization overhead. Because the virtual platforms are used for virtual BS, BS-related tasks including protocol, baseband processing, etc., are the main workloads to evaluate the micro level performance besides some classic benchmark workloads such as STREAM, Iperf [15], etc. for virtualization machine.

Macro level metrics are evaluated by running complete virtual BSs with different traffic loads which is configured dynamically according to performance test requirements. For example, in order to obtain resource utilization rate performance, we will create multiple virtual BSs and configure BSs with both high traffic load and low traffic load. The macro level metrics evaluation can be considered as a black-box approach because these metrics are collected from virtual BSs directly.

IV. TESTING RESULTS AND ANALYSIS

Based on the performance metrics and evaluation methods defined in the previous sections, this section presents the testing results of two types of virtualization platforms as case studies. One is the KVM hypervisor (QEMU 2.5.1,
forms can support the CPU processing resources shared among platforms increases. This is because the virtualization platform decreases as the number of BSs running on the virtualization in average. We can see that the BS average CPU occupation traffic loads to reflect the basic resource efficiency of virtualization platforms are not large. Although the Disk I/O Throughput overhead is relatively big, it is also acceptable because the disk access is not frequent in BS systems.

In the virtualization overhead evaluation, we choose STREAM as the memory-intensive workload, Iperf as the network-intensive workload, Bonnie++ as the disk I/O-intensive workload, and Y-cruncher as the CPU-intensive workload, all of which are popular evaluation benchmarks [15]. STREAM measures memory data throughput by implementing four typical vector operations, namely Copy, Scale, Add and Triad. Bonnie++ also carries out various storage operations such as reading/writing byte-size and block-size data. Therefore, the results of different operations in STREAM and Bonnie++ are recorded. It can be seen from Table II that the virtualization overhead performance difference of the two platforms are not large. Although the Disk I/O Throughput overhead is relatively big, it is also acceptable because the disk access is not frequent in BS systems.

B. Macro Level performance test

Considering the real-time issue of KVM as shown above, we test BS requirements satisfaction performances only for the Super BS virtualization platform. The peak data rate and control plane/user plane latency are measured as the study cases to obtain Satisfaction Degree $D_s$. In the field trial of the Super BS, the peak data rate is 152 Mbps (20 MHz Bandwidth), the control plane latency is 9.6 ms and the user plane latency is 4.9 ms. Therefore, Satisfaction Degree $D_s$ of the three KPIs are 1.

Fig. 6 depicts the BS average CPU occupation with different traffic loads to reflect the basic resource efficiency of virtualization platforms. In the test, the average CPU occupation is measured by the number of CPU cores used by one BS in average. We can see that the BS average CPU occupation decreases as the number of BSs running on the virtualization platforms increases. This is because the virtualization platforms can support the CPU processing resources shared among BSs dynamically. When a BS is in a high traffic load, more hardware resources are allocated to it. When the traffic load is low, the redundant hardware resources could be dynamically re-assigned to other BSs with a higher traffic load. Moreover, the resources multiplexing gain (i.e. the lines slope in Fig. 6) has relationship with the traffic load distributions of BSs. If the BSs have similar traffic load patterns, for example, they all have no loads or 50 Mbps loads in Fig. 6, their resource requirements will be simultaneously either high or low. Thus the multiplexing gain is obviously smaller than that at the conditions where BSs have diverse traffic loads, for instance, some BSs have 20 Mbps loads and some have no loads in Fig. 6.

In Fig. 7, the total migration time and downtime are compared between KVM and Super BS virtualization platforms under the BSs different traffic loads. It can be seen that BS live migration performance of the Super BS is higher than that of KVM and the performance gain increases with the traffic loads. Since the radio link reset timer is usually set as 50 ms, the communication link between users and BSs will not be affected when downtime is shorter than 50 ms. We can see that the downtime of the Super BS is always below 50 ms, so the communication links do not need to reset during the BS live migration.

Finally, we measure the virtual BS creating latency to evaluate the BS function virtualization performance of different platforms. In the conventional BS development mode without the function virtualization, BSs for different application scenarios should be re-developed. Even if the re-development time is excluded, the BS creating latency is usually several minutes at least. However, both KVM and Super BS platforms only need ten to twenty seconds to create a new virtual BS in our test. Therefore, virtualization platforms improve the BS deployment efficiency greatly.

C. Observations

Firstly, from the performance test of BS virtualization platforms, it is can be seen that the virtual BS mode is feasible for radio access networks. Furthermore, the virtual BS has many advantages over the conventional BS mode, such as high resource efficiency, flexible functions support and easy deployment.
### TABLE II: Micro Level Performance Test Results

<table>
<thead>
<tr>
<th>Category</th>
<th>Metrics</th>
<th>KVM</th>
<th>Super BS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real-time performance</td>
<td>Min Kernel latency</td>
<td>4us</td>
<td>4us</td>
</tr>
<tr>
<td></td>
<td>Max Kernel latency</td>
<td>5us</td>
<td>4us</td>
</tr>
<tr>
<td></td>
<td>Kernel latency jitter</td>
<td>368us</td>
<td>2us</td>
</tr>
<tr>
<td>Virtualization overhead</td>
<td>BS protocol Overhead Ratio</td>
<td>16.7%</td>
<td>11.2%</td>
</tr>
<tr>
<td></td>
<td>CPU Computation Time Overhead Ratio</td>
<td>2.42%</td>
<td>1.9%</td>
</tr>
<tr>
<td></td>
<td>Memory Data Throughput Overhead Ratio</td>
<td>Copy: 5.8%</td>
<td>Copy: 5.4%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Scale: 4.7%</td>
<td>Scale: 4.7%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Add: 4.78%</td>
<td>Add: 4.76%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Triad: 4.79%</td>
<td>Triad: 4.78%</td>
</tr>
<tr>
<td></td>
<td>Network Throughput Overhead Ratio</td>
<td>0.4%</td>
<td>0.3%</td>
</tr>
<tr>
<td></td>
<td>Disk I/O Throughput Overhead Ratio</td>
<td>Byte Data Writing: 7.2%</td>
<td>Byte Data Writing: 3.6%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Byte Data Reading: 8.5%</td>
<td>Byte Data Reading: 13.5%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Block Data Writing: 13%</td>
<td>Block Data Writing: 11%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Block Data Reading: 22.4%</td>
<td>Block Data Reading: 27%</td>
</tr>
</tbody>
</table>

Moreover, the performance results can help us to investigate how to customize the virtualization platform based on the requirements of radio access networks. For example, the reason that the KVM platform does not satisfy BSs real-time requirements is mainly due to its kernel latency jitter. Thus optimization schemes should focus on reducing the kernel latency jitter in order to improve its performance.

Thirdly, in order to promote the development of virtual BSs, BS virtualization platforms should be studied further. There is more space for improvement in real-time guarantee, virtualization overhead, resource efficiency, etc. of the BS virtualization platforms.

### V. CONCLUSION

The virtual BS development mode is suitable for 5G systems and the virtualization platform is the significant component of a virtual BS. However, there are no performance metrics and evaluation methods designed for the virtualization platforms in BSs. This paper defines six categories of performance metrics for BS virtualization platforms specially, which are real-time performance, virtualization overhead, BS requirements satisfaction, BS resource virtualization, BS live migration and BS function virtualization. Then corresponding evaluation methodology are introduced for measuring the new metrics. Applying the performance metrics and evaluation methods defined in this paper, testing results of the KVM hypervisor and the virtualization platform of the Super BS are obtained. Proper application of these metrics can help users quickly choose the best virtualization platforms and help developers to detect performance bottlenecks of various systems.

### REFERENCES


