SCIENCE CHINA
Information Sciences

• LETTER •

July 2020, Vol. 63 179102:1–179102:3
https://doi.org/10.1007/s11432-018-9914-5

Boosting performance of virtualized desktop infrastructure with physical GPU and SPICE

Shupan LI[1,2], Limin XIAO[1,2*], Chungang SHI[3], Liequan CHE[3], Changyou ZHANG[4] & Yuanzhang LI[5]

[1]*State Key Laboratory of Software Development Environment, Beihang University, Beijing* 100191*, China;*
[2]*School of Computer Science and Engineering, Beihang University, Beijing* 100191*, China;*
[3]*Beijing Jinghang Computation and Communication Research Institute, Beijing* 100074*, China;*
[4]*Institute of Software, Chinese Academy of Sciences, Beijing* 100190*, China;*
[5]*School of Computer Science and Technology, Beijing Institute of Technology, Beijing* 100081*, China*

Received 25 December 2018/Revised 26 February 2019/Accepted 5 May 2019/Published online 18 May 2020

Citation Li S P, Xiao L M, Shi C G, et al. Boosting performance of virtualized desktop infrastructure with physical GPU and SPICE. Sci China Inf Sci, 2020, 63(7): 179102, https://doi.org/10.1007/s11432-018-9914-5

Dear editor,

The virtual desktop infrastructure (VDI) is widely used by enterprises, as it can be easily deployed, has a reduced management cost, improves resource utilization, and can be accessed at any location by the network. Currently, there is a demand for the VDIs to support applications, such as playing 3D games, playing HD videos, and using 3D designing software. The software in those applications requires the GPU to support 3D rendering and consumes a lot of GPU resources. However, virtual GPUs (vGPUs) perform poorly and do not support 3D rendering. Therefore, a physical GPU (pGPU) must be used.

The virtual desktop software (VDS) is an essential software used in VDI, and it is used for transferring data, such as display information (DI) and operations, between the virtual machine (VM) in the VDI and the user. There are many VDS, such as SPICE, ICA, and PCoIP. As SPICE[1)] is open source and can be modified freely, it has been extensively used. The SPICE client can be install in windows, Linux and Android [1]. However, the SPICE server is designed for a VM with a vGPU, and cannot get the DI from the pGPU.

To enable VDIs to take advantage of the performance of the pGPUs and to enable the SPICE to show the DI from a pGPU to the users, we propose an optimized method for SPICE. The method obtains the DI from the GPU driver in the VM and sends the DI to the SPICE server using the VirtIO port [2], the SPICE server then sends the DI to the SPICE client by reusing the display channel. The method can both provide the VM with the necessary support to use vGPUs and pGPUs. The evaluations show that: (1) the proposed method can transfer the DI from the pGPU to users, when the VM uses a pGPU to support 3D; (2) it has excellent performance in terms of network traffic, CPU utilization, and latency of the DI.

Related work. VMs can use a pGPU by the VT-d and virtual machine monitor. There are mainly two types for the VM to use a pGPU, according to how the GPU commands in the VM are submitted to the pGPU. (1) The GPU commands in the VM are first submitted to the agent in the host and then submitted to the pGPU. (2) The GPU commands in the VM are directly sent to the pGPU. The GVT-g [3] in Intel and GRID vGPU in NVIDIA belong to the former type. The AMD SRIOV and GPU pass-through [4] belong to the latter type. However, if we get the DI from the GPU driver in the VM, the VM does not need to care about how the VM uses the pGPU. Look-

* Corresponding author (email: xiaolm@buaa.edu.cn)
1) SPICE. 2019.4. https://www.spice-space.org/.
2) Looking Glass. 2018.6. https://looking-glass.hostfission.com/.

ing Glass[2)] enables a VM to use an NVIDIA GPU, but NVIDIA has disabled the capture SDK. GVT-g enables SPICE to support the GPU in the Intel CPU, but Intel has removed the GPU in Xeon, which is used in the data center.

Design and implementation. Considering the functionality and maintenance, the optimized method should achieve the following goals.

• In the term of functionality: (1) It must be able to show the DI from the pGPU to the users. (2) It must also enable the VM to use the vGPU.

• In the term of maintenance: It must not change the source code of the QEMU and SPICE client. In the QEMU, it reuses VirtIO which is used to transfer data between the VM and the SPICE client. In the SPICE, it reuses the encoding that is supported by SPICE and the display channel which is used to transfer the DI between the SPICE server and client.

To display the DI from the pGPU to the users, we need to add an agent module into the VM and a GPU module into the SPICE server. The architecture of the prototype is shown in Figure 1(a). The two modules are explained below.

• The GPU module: it is mainly used to control the agent module, get the DI from the VirtIO port, and send the DI to the display channel of SPICE. As it is in the SPICE server, it needs to handle the semaphores from SPICE server, such as "open port", "close port", "client connection", and "client disconnection". The "open port" and the "close port" indicate the status of the VirtIO port used to transfer the DI. The "client connection" and the "client disconnection" indicate whether the SPICE client is connected to the SPICE server. The GPU module works on the following conditions: (1) the VirtIO port is open, (2) the SPICE client is connected, and (3) the VM uses the pGPU.

• The agent module: it is mainly used to obtain the DI from the driver of the pGPU using the capturing SDK provided by the GPU manufacturer, and send the DI to the GPU module using the VirtIO port. It is a service in the operating system (OS), and it starts as the OS starts. As it is in the VM, it needs to handle some semaphores from the OS, such as "resolution change" and "send data completely". The "resolution change" indicates that the resolution of the DI has been changed. If the resolution is changed, the resolution of the DI needs to be changed. The "send data completely" means that the data have been sent to the VirtIO port. If it gets "send data completely", it will send the next frames of the DI.

To make the GPU module and the agent module work together, some messages must be transferred between them by the VirtIO port. The messages sent from the CPU module to the agent module include "capability", "start", and "stop". The "capability" is used to negotiate the encoding type of the DI. The "start" and "stop" respectively are used to start or stop capturing the DI from the pGPU. The messages sent from the agent module to the CPU module include "resolution", "display information", and "encoding type". The "resolution" is used to indicate the resolution of the desktop. The "display information" contains the frames of the DI received by the agent. The "encoding type" is used to show the encoding types of the DI, such as H.264 and MPEG. The sequences of the messages are shown in Figure 1(b).

To handle the semaphores from the OS and the messages from the GPU module, there are many threads in the agent module. The "transporting thread" is mainly used to send messages and receive messages from the VirtIO port. The "instruction processing" is mainly used to decode the messages from GPU module in the SPICE and semaphores from the OS, then send those commands or data in the messages to other threads. The "capturing thread" is mainly used to capture the frames of the DI from the GPU driver and encode the DI to H.264. The "semaphore monitor" is mainly used to monitor the semaphore of the "resolution change" from the OS. If the resolution of the DI changes, the "capturing thread" will be restarted. The relation and the messages among the threads are shown in Figure 1(c).

To avoid changing the source code of the SPICE client and to enable the VM to use both the vGPU or the pGPU, we need to encode the DI obtained from the pGPU to H.264, which is supported by the SPICE client, and reuse the display channel in SPICE. The method to reuse the display channel is depicted in Figure 1(d). According to the type of GPU used by the VM, the SPICE server decides whether the DI from the pGPU is sent to the SPICE client or the DI from the vGPU is sent to the SPICE client. The status of the VirtIO port shows which type of GPU the VM uses. If the port is open, the VM uses the pGPU. Otherwise, the VM uses the vGPU.

For a VM that has a pGPU assigned, the VM uses the pGPU after the driver of the GPU is started, and the VM uses the vGPU before the driver of the pGPU is started, as shown in Figure 1(e). If the SPICE client both connects to and disconnects from the SPICE server when the VM uses the vGPU, the SPICE server will use the original function in SPICE to handle the DI from the vGPU. If the SPICE client both connects to and disconnect from the SPICE server when the VM

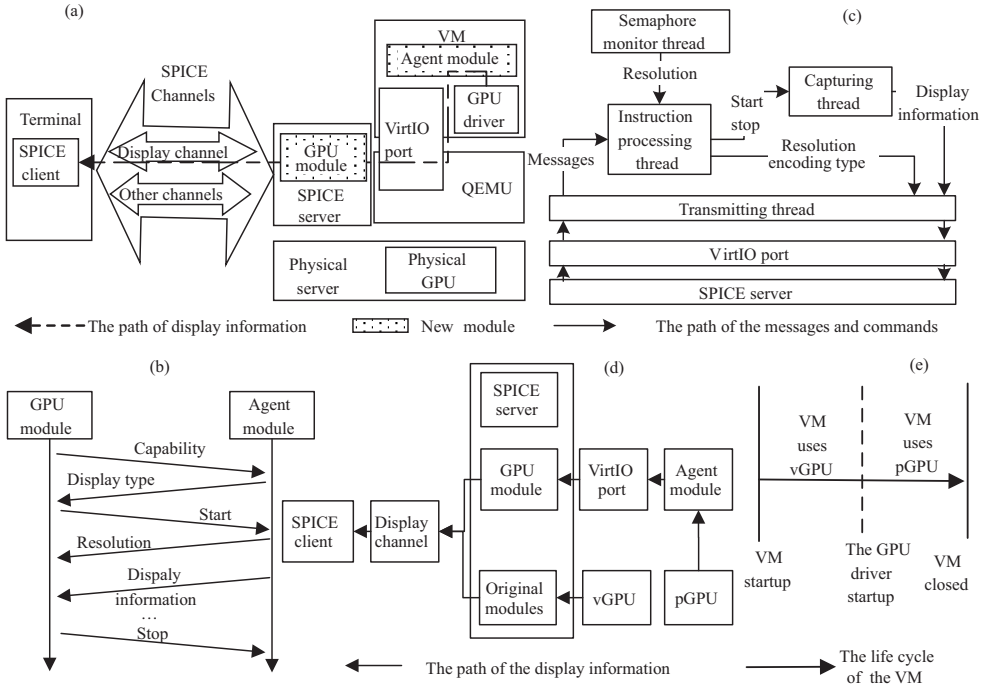

Figure 1 (a) Architecture of the prototype. (b) Sequence of messages between the GPU module and the agent module. (c) Messages among the threads in the agent module. (d) Reusing the display channel. (e) Relation between the life cycle of the VM and which type of GPU the VM uses.

uses the pGPU, the SPICE server will use the GPU module to handle the DI from the pGPU. However, if the SPICE client connects to the SPICE server when the VM uses the vGPU and disconnects to the SPICE serve when the VM uses the pGPU, the DI must be switched from the vGPU to the pGPU. The switch time is when the VirtIO port is opened by the agent module.

Experimental evaluation. Herein, the experiment environment, the function evaluation, and performance evaluation are described. All the three parts can be found in Appendix A. The environment includes the PM, the VM, the terminal, and the network. In the function evaluation, we show that: (1) the original SPICE cannot show the DI from the pGPU to the SPICE client, but the optimized SPICE can do; (2) the pGPU can support 3D, but the vGPU (QXL) cannot. In the performance evaluation, we evaluated the optimized SPICE from the following aspects, network traffic, CPU utilization, and latency of DI. We can see that the optimized SPICE has an excellent performance.

Conclusion. Herein, we optimized the SPICE for the VDI using a pGPU. This enables the VDI to leverage the advantages of the pGPU. A prototype based on KVM, QEMU, AMD Radeon WX4100 is implemented. The evaluations show that the optimized SPICE can display the DI obtained from the pGPU to the users and has an excellent performance.

Acknowledgements This work was supported by National Key Research and Development Program of China (Grant No. 2017YFB1010000), Fund of the State Key Laboratory of Software Development Environment (Grant No. SKLSDE-2018ZX-10), National Natural Science Foundation of China (Grant No. 61772053), and Science Challenge Project (Grant No. TZ2016002).

Supporting information Appendix A. The supporting information is available online at info.scichina.com and link.springer.com. The supporting materials are published as submitted, without typesetting or editing. The responsibility for scientific accuracy and content remains entirely with the authors.

References

1 Tan Y A, Xue Y, Liang C, et al. A root privilege management scheme with revocable authorization for Android devices. J Netw Comput Appl, 2018, 107: 69–82
2 Russell R. Virtio: towards a de-facto standard for virtual I/O devices. ACM SIGOPS Oper Syst Rev, 2008, 42: 95–103
3 Tian K, Dong Y, Cowperthwaite D. A full GPU virtualization solution with mediated pass-through. In: Proceedings of USENIX Annual Technical Conference, 2014. 121–132
4 Shah A, Kay A, BenYehuda M, et al. PCI device passthrough for KVM. In: Proceedings of the 2nd edition of the KVM Forum, Napa Valley, 2008