Providing Extra Memory for Virtual Machines by Sharing Compressed Swap Pages

Jonghee Yun1, Chiyong Lee and Chuck Yoo, Member, IEEE
1Samsung Electronics, Suwon, Korea, Korea University, Seoul, Korea
1yun.jonghee@gmail.com, cylee@os.korea.ac.kr, chuckyoo@os.korea.ac.kr

Abstract—Virtualization systems are currently being applied to embedded devices such as TVs and cell phones. However, these embedded devices have insufficient memory, which can decrease the performance of an application. To provide more memory and improve the performance of applications on a virtual machine, we propose an effective swapping method. When a swap-out occurs on a virtual machine, we compress the swapped page and send it to another virtual machine with sufficient memory. If a compressed swapped page already exists, it will be shared with other virtual machines. By compressing and sharing the swapped pages, our method can reduce their number by 79%.

I. INTRODUCTION

A virtualization system can improve the security and efficiency of resources. Virtualization systems have recently been applied to TVs and cell phones. For example, a cell phone can run two virtual machines. One is simply used for a calling application, and the other is for applications downloaded from the web by users.

Running several virtual machines concurrently can cause lack of memory. In particular, application performance on each virtual machine can be degraded if the embedded devices have insufficient memory.

When lack of memory occurs, the virtual machine begins swapping in order to create extra memory for an application. However, the swapping causes an overhead in disk I/O, which decreases the performance of the application. In addition, instead of a disk, most embedded devices use flash memory for a swapping area. However, using flash memory for swapping is inappropriate. First, a re-write operation on flash memory is slow because the flash memory has to erase the block including the page to modify the page data. Second, swapping too frequently shortens the lifetime of the flash memory.

There have been different studies [1], [2] on providing extra memory using swapping and compression. However, such methods have certain restrictions on the provision of extra memory, and pass up the opportunity to share identical pages.

In this paper, we propose a new swapping method for virtual machines to remove the disk I/O overhead and provide extra memory to a virtual machine by compressing and sharing the swapped pages. In addition, our swapping method can be run on embedded devices without a disk.

II. RELATED WORK

The sharing of identical pages has been proposed to reduce memory usage and disk I/O operations [3]. This method applies to the Xen virtualization architecture [4]. Xen has two types of domains: Domain-0 and Domain-U. Domain-0 deals with requests for hardware resources such as disks. Domain-U is a virtual machine that can run various applications. Using the Xen architecture, Domain-0 in [3] checks all page requests to find identical pages. It reduces the memory consumption through the elimination of duplicated pages. However, a comparison of all pages causes poor application performance.

The compression of the main memory was proposed to improve web server performance [1]. This mechanism compresses pages into their own compressed-page cache area, which shows an improvement in web server throughput and a reduction in disk bandwidth. However, the method in [1] uses its own main memory to store compressed pages, and there is therefore a restriction in creating a sufficient amount of memory.

The authors in [2] introduced a virtualization mechanism in which a third-party virtual machine sends pages to the main virtual machine for the quality of service (QoS) and reliability of an application on the main virtual machine. Instead of swapping, this mechanism uses ballooning to provide extra memory to the main virtual machine. However, if there is no extra memory for ballooning in a third-party virtual machine, the main virtual machine has to offer its own memory for storing the pages of the third-party machine.

Although these studies provide extra memory by swapping memory use, each virtual machine consumes its own memory for the compressed swapped pages, which decreases the available memory in each virtual machine. In addition, the studies discussed above do not consider the possibility of sharing identical pages among two or more virtual machines.

III. SHARING OF COMPRESSED SWAP PAGES

We applied our method to a Xen-based virtualization architecture. The proposed method stores compressed swapped pages in the main memory on another virtual machine with sufficient memory. In addition, the compressed swapped pages can be shared between virtual machines. We use Domain-0 for storing and sharing such pages, because this domain is always running and uses less memory than other virtual machines.

Fig 1 shows the swapped-page identification and page sharing procedures. As shown in 1) of Fig 1, when swapping...
occurs because of lack of memory in Domain-U, Domain-U uses a swapping callback function to hook and compress a swapped page. After the compression, Domain-U sends the compressed swapped page to Domain-0 for swapped page identification, as in 2) of Fig 1. For the swapped page identification, Domain-0 checks the page sharing table to determine whether an identical compressed swapped page exists in a swapping area, as shown in 3) of Fig 1. Finally, as 4) in Fig 1 shows, if a compressed swapped page already exists, Domain-0 does not store the page in the swapping area. Instead of storing the compressed swapped page, Domain-0 gives its address to Domain-U. Through this sharing step, Domain-0 can reduce the amount of memory consumed by the compressed swapped pages.

We create a page-sharing table to manage the shared swapped pages. The page-sharing table maintains information about the compressed swapped pages in Domain-0, and consists of two lists. The list of pages indicates the physical memories of the compressed swapped pages. In addition, the list of page entries maintains information regarding these pages. Each entry contains the page id, page address, number of shared domains, and length. Fig 2 describes the page-sharing table and an example of sharing a compressed swapped page using Domain-U1 and Domain-U2.

IV. EXPERIMENTAL RESULTS

To clarify the tendency for using swapped pages, three main memory sizes are given to Domain-U: 154 MB, 156 MB, and 160 MB. We compiled Linux kernel sources on Domain-U1 and Domain-U2 at the same time.

Fig 3 shows the number of compressed swapped pages while compiling the Linux kernel source. For the 154 MB memory, Domain-U1 swaps out 8,008 pages and Domain-U2 swaps out 8,329 pages, and Domain-0 consumes 3,200 of these. The results of this method indicate that the number of swapped pages is reduced by 79% through compressing and sharing.

Fig 4 shows the number of pages shared between Domain-U1 and Domain-U2. The difference in the number of pages between Domain-U1 and Domain-U2 comes from the swapping policy for a Linux kernel. As shown in Fig 4, Domain-U with 160 MB of memory shares 99 of the shared pages, and Domain-U with 154 MB of memory shares 182. These results indicate that more pages are shared in lower-memory systems.

V. CONCLUSION

It is important to secure a sufficient amount of memory for a virtual machine on an embedded device. Although some researchers have provided methods to improve the performance of an application by providing extra memory, a new scheme is necessary for a virtual machine. We propose two techniques to solve this issue. The first stores the compressed swapped pages on another virtual machine that has sufficient memory. The second shares the compressed swapped pages with virtual machines. Our experiments show that the proposed method significantly reduces the memory usage of swapped pages in a virtual machine.

REFERENCES