# Who Decides Migration?

## A Migration Lock Mechanism for Virtual Machines

Xiaolin Wang, Yingwei Luo

Dept. of Computer Science and Technology,
Peking University, Beijing, China
{wxl, lyw}@pku.edu.cn

Zhenlin Wang

Dept. of Computer Science,
Michigan Technological University, Houghton, USA
zlwang@mtu.edu

*Abstract* - **Migration of virtual machines is an important feature for the management of a virtualized environment. Current strategies for managing migration consider more of resource scheduling and system maintenance, ignoring possible constraints on migration from virtual machines and applications. This paper introduces a migration locking mechanism and its implementation on Xen. The virtual machine migration locking mechanism provides a standard interface for virtual machine users to control migration status of their virtual machines according to their own wishes, such as security, hardware or performance demands and so on. But forced migration locking by the end users or applications could conflict with the management decisions by managers or the virtual machine monitors. How to coordinate different requirements between virtual machine users and managers needs further discussion.**

***Keywords*** **- virtual machine; virtual computing environment; migration; migration lock; migration strategy**

## I. INTRODUCTION

Virtual machine migration means migrating one virtual machine from one physical machine, called the source host, to another physical machine, called the destination host. Current mainstream virtual machine monitors support online migration of virtual machines, such as Xen live migration [1] and VMware VMotion [2]. Migration is an important feature of virtual machine management, which is a unique advantage brought by virtualization. As a result of the feature, we can deploy services more conveniently, flexibly and efficiently in a data center that utilizes virtualization technologies.

With the help of migration of virtual machines, a virtualized data center can improve resource utilization and save energy. However, current strategies for managing the migration of virtual machines focus only on system maintenance and resource scheduling [3] [4], without considering possible damaging effects caused by migration. The destination host might lack sufficient hardware resources or security measures to satisfy the demands of the migrated virtual machine. Whether a virtual machine can be migrated depends on the characteristics and requirements of the application running on it, which include the following aspects: (1) the application is running some critical code and does not want to be affected by migration of the virtual machine; (2) the application is performing some critical operation, migration will fail the operation; (3) some special optimizations, such as optimizing communication between virtual machines sharing the physical host, requires that several virtual machines run on the same physical machine at the same time; (4) dependence on special hardware resource. Migrating a virtual machine in the above scenarios will cause significant performance degradation or even system failure, so it's better not to migrate it. But the virtual machine monitor or the manager of data center cannot get to know such requirements of applications in time, so they cannot make a correct decision.

In this paper we propose a mechanism of migration lock, which involves three layers: the application layer, the Guest OS layer and the virtual machine monitor (VMM) layer. With a migration lock, users can declare whether their virtual machines are allowed to be migrated. When users don't expect their virtual machines to be migrated, they can lock their virtual machines up in current physical machine to prevent the virtual machine monitor from migrating the virtual machines. So we can fix a virtual machine to a physical machine when the virtual machine is reading or processing confidential data or when it has obtained sufficient resources in the current physical machine and does not expect to be migrated.

The rest of this paper is organized as follows. Section 2 introduces the design principles of our virtual machine migration locking mechanism. Section 3 gives a detailed description on how to implement the migration lock in Xen Hypervisor. Then, in Section 4, we describe how to set migration lock. The influence of the migration locking mechanism on management of the future virtualized computing environment is discussed in Section 5 and Section 6.

## II. Design of Migration Locking Mechanism

In a virtualized environment, virtual machines run above a virtual machine monitor, which is responsible for scheduling, allocating and managing resources [5]. Migration of virtual machines is also implemented by the virtual machine monitor. We can determine whether a virtual machine can be migrated by set a status flag for the virtual machine in the virtual machine monitor, as shown in Figure 1. If the virtual machine monitor or the virtual machine user attempts to migrate the virtual machine, the virtual machine monitor will check the status flag first. If it is permitted to be migrated, the migration operation will be executed; otherwise, the migration operation will be dropped.

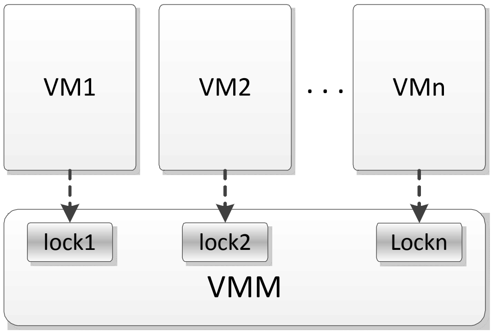


Figure 1. Architecture of Virtual Machine Migration Lock

The status flag can be set in the configuration file or the starting command of the virtual machine, which is called static migration lock. Besides that, it can also be set by the virtual machine user at run time according to the needs. We refer to this type of flag as a dynamic migration lock. With a dynamic migration lock, a virtual machine can control its own migration status dynamically, and avoid to be migrated when it doesn't expect to be migrated, as shown in figure 2.

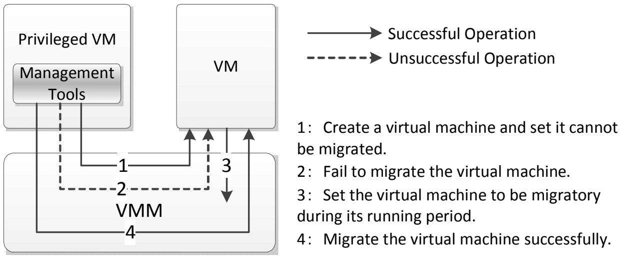

Privileged VM
Management Tools
VM
VMM
1
2
3
4
Successful Operation
Unsuccessful Operation
1: Create a virtual machine and set it cannot be migrated.
2: Fail to migrate the virtual machine.
3: Set the virtual machine to be migratory during its running period.
4: Migrate the virtual machine successfully.

Figure 2. Set Migration Lock Dynamically

## III. Implementation of Migration Locking Mechanism

According to the design described in Section 2, we have implemented the migration locking mechanism on Xen-3.3.1[6]. Detailed implementation can be divided into the following four aspects: Xen Hypervisor, Guest OS of Domain U, Application in Domain U and Xen Tools, which are elaborated in this section.

### A. Xen Hypervisor

Xen maintains a structure (struct domain) that represents the status of a domain for every virtual machine. We add a variable (bool_t *is_migratory*) to this structure, which indicates whether the corresponding domain can be migrated. The command line option or the configuration setting of migration lock is indeed implemented as setting of the variable at the end. The migration command will check the variable before executing the migration operation. If it is not permitted to be migrated, it will stop the migration operation.

Besides that, one hypercall with a number *__HYPERVISOR_set_migraton_flag* is appended in Xen hypervisor. The setting of i*s_migratory* in the domain structure is implemented through a handling function *do_set_migration_flag* for this hypercall.

### B. Guest OS of Domain U

In order to achieve the goal that virtual machine users can determine whether their virtual machines can be migrated, we have implemented a transmission mechanism in the guest OS of domain U, which can transfer the command of setting migration lock from the application level to the Xen hypervisor. Because Xen supports both para-virtualization and full-virtualization, we implement the transmission mechanism in both two types of virtual machines. Linux can be modified, so we use it as the guest OS of para-virtualized virtual machines; Windows operating system cannot be modified, so we use it as the guest OS of full-virtualized virtual machines.

*1) Linux Operating System:* The Linux operating system that we use is 64-bit, and its version number is 2.6.18. In this operating system, we append a function *sys_set_migration_flag* that calls the hypercall implemented in Xen hypervisor and a system call with a number 325 which calls the function.

*2) Windows Operating System:* Because Windows is not an open source operating system and it cannot be modified, we append a windows device driver through which the virtual machine migration lock can be set.

We first obtain the number of hypercall pages of Xen Hypervisor via CPUID instruction. Then a block memory with the same size as the hypercall pages is allocated for setting new hypercall pages for Windows Operating System by using WRMSR instruction. Now we can set migration lock by calling the hypercall we have implemented in Xen hypervisor through the function handling I/O operations (*MldDDKDeviceIOControl*) in the Windows device driver.

### *C. Application in Domain U*

We write an application tool for users to set migration lock. For different operating systems, its implementation is different. The tool sets migration lock via the system call described in Section 3.2.1 for Linux, while it achieves the same goal via the I/O control operation on the device driver described in Section 3.2.2 for Windows.

Users can also set migration lock by calling the system call or I/O control operation in their own applications.

### *D. Xen Tools*

Besides setting migration lock in domain U, we also can set migration lock when domain U is being created or started. Also the migration lock should be checked when migrating domain U. Both creating domain U and migrating domain U are associated with Xen tools, so we add the corresponding mechanism to *xm* and *xend*.

*1) Create or Start Domain U:* Users can create domain U via the xm command in Dom0. The specific format of the command is: *xm create [-c] [options] configuration_file*. *xm* first parses the command and the configuration file, and then transfers the parsed results to *xend* which will call a function named *domain_create* for creating a domain. The function creates a domain by calling a hypercall whose number is __HYPERVISOR_domctl. Figure 3 shows the specific process of creating a domain.

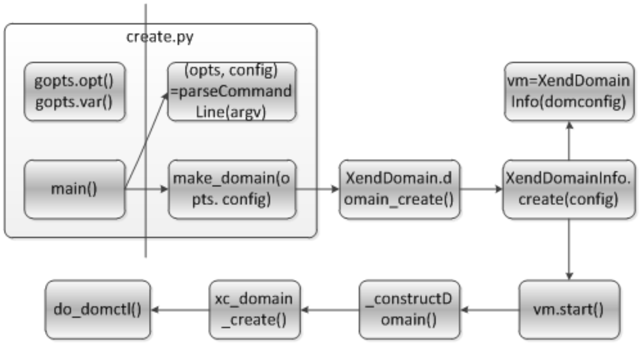


Figure 3. The Process of Creating Domain U

In order to set migration lock when creating or starting a domain, we append an option (*-migration*) in the configuration file and *xm* command, and parse and transfer the option in *xend*.

*2) Migrate Domain U:* Users issue the request of migration to Xen hypervisor via the command – *xm migrate*. After receiving the request, Xen hypervisor will call the function *domain_migrate* to prepare for migration. The migration process is divided into two parts, device migration, which is accomplished by function *xc_domain_save*, and memory migration, which is accomplished by function *migrateDevices*. Figure 4 illustrates the process.

According to the analysis of the migration process described above, we can implement the migration lock checking in the two functions - *domain_migrate* and *xc_domain_save*, to judge whether the domain is permitted to be migrated.

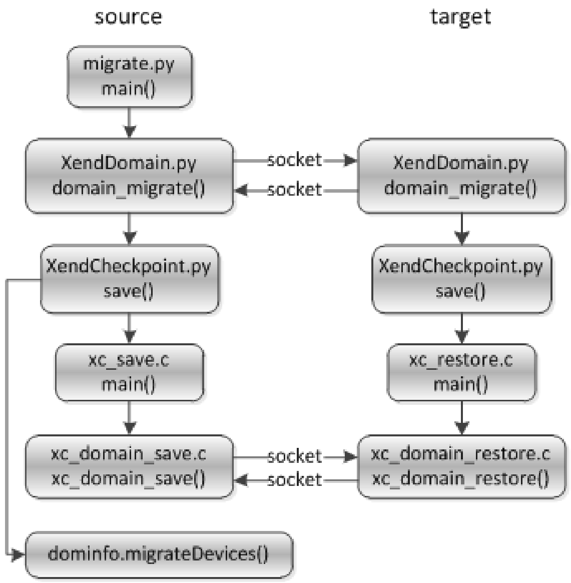


Figure 4. The Migration Process of Domain U

## IV. Setting Migration Lock

As described in Section 3, we can set migration lock in two phases: one is at the time when a virtual machine is created or started; the other is during the running period of a virtual machine.

There are two ways for setting the migration lock when creating or starting the virtual machine.

The first way is appending the option (*-migraton*) to the command of creating or starting a domain. The specific format is:

*xm create config_file -migration=yes|no*

the migration flag set to be yes or no indicates that the virtual machine can or cannot to be migrated.

The second way is appending the following configuration:

*migration = yes|no*

to the configuration file of the virtual machine.

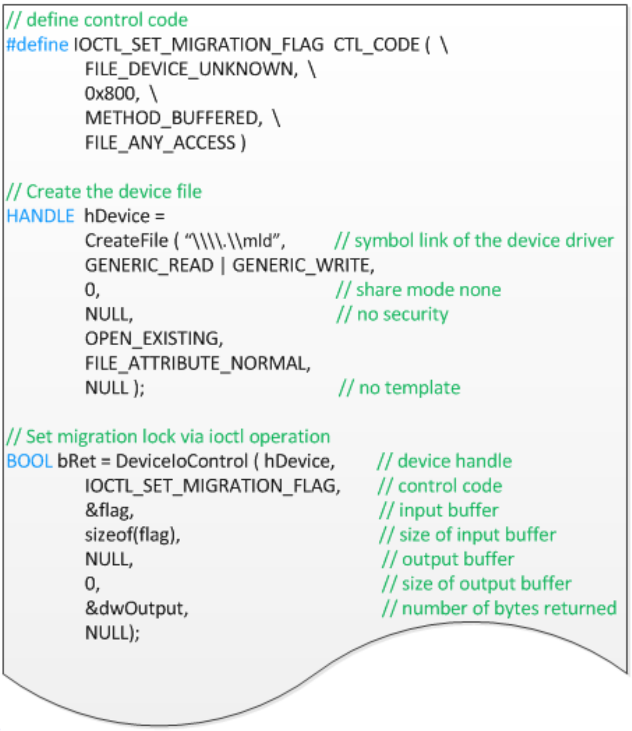
In order to increase flexibility, we support the virtual machine users to set migration lock in the guest operating system during the virtual machine is running. So virtual machine users can set migration lock dynamically according to the current running status of the virtual machines.

Virtual machine users can set migration lock dynamically via the lock setting application tool *set_migration_flag* provided by us. The specific format for running the application is:

*set_migration_flag flag*

here flag is the parameter and its value must be 1 or 0 indicating whether the virtual machine can or cannot be migrated.

In addition, virtual machine users can also set migration in their own application programs via the interface provided by us. The interface and its usage are different for different guest operating system of virtual machines. For GNU/Linux operating system which runs upon a para-virtualized virtual machine, we provide a system call interface named syscall(325, flag, &ret). Here 325 is the system call number, flag is the same as described above and ret is the result of the system call. For Windows operating system running on a full-virtualized virtual machine, we provide a device driver in which setting migration lock is implemented. Users just need to create the device file and set migration lock via ioctl operations in their own application programs. Figure 5 shows how to set migration lock in an application program in Windows operating system.



```
// define control code
#define IOCTL_SET_MIGRATION_FLAG  CTL_CODE ( \
          FILE_DEVICE_UNKNOWN, \
          0x800, \
          METHOD_BUFFERED, \
          FILE_ANY_ACCESS )

// Create the device file
HANDLE  hDevice =
          CreateFile ( "\\\\.\\mld",        // symbol link of the device driver
          GENERIC_READ | GENERIC_WRITE,
          0,                                // share mode none
          NULL,                             // no security
          OPEN_EXISTING,
          FILE_ATTRIBUTE_NORMAL,
          NULL );                           // no template

// Set migration lock via ioctl operation
BOOL bRet = DeviceIoControl ( hDevice,      // device handle
          IOCTL_SET_MIGRATION_FLAG,         // control code
          &flag,                            // input buffer
          sizeof(flag),                     // size of input buffer
          NULL,                             // output buffer
          0,                                // size of output buffer
          &dwOutput,                        // number of bytes returned
          NULL);
```

Figure 5. Set Migration Lock
in Windows Application

## V. How to Use Migration Lock

Virtual machine migration locking mechanism enables virtual machines to control migration by themselves and assures that they cannot be migrated if they don't expect to. So the current migration strategies should take migration lock into consideration. Which virtual machine should be locked and how long it will be locked have a great influence on migration strategies. Whether we should migrate a virtual machine immediately or wait until the virtual machine is unlocked, how long a virtual machine will be locked, and what about the dynamic change of locking/unlocking, all these will make migration strategies more complex. We should make decisions according to actual application scenarios.

For a virtual machine, under what conditions it should be locked and when to lock/unlock often need to comprise with the flexibility of managing a virtualized computing environment. We should try to minimize the influence on conventional migration strategies. A virtual machine should use migration lock only when it needs to be locked. In a virtualized computing environment, if virtual machines use migration lock arbitrarily, it will be impossible to manage and maintain the virtualized computing environment, losing advantage brought by virtualization.

Allocating an effective time slot for using migration lock is a way to avoid using migration lock arbitrarily. When the virtual machine monitor attempts to migrate a virtual machine, whether the time the virtual machine has been locked exceeds the time slot will be checked first. If it exceeds the time slot, the virtual machine monitor will modify the status of migration lock and migrate the virtual machine. The hard quota avoids the problem of dead lock on migration but fails to address the issue that the lock sometimes need to be maintained longer than the quota.

For a virtualized computing environment, when a virtual machine should be locked and how to avoid using migration lock arbitrarily need further study from technology and management aspects.

## VI. Conclusion

In this paper, we propose the problems caused by virtual machine migration and present a solution named virtual machine migration lock. We provide standard interface for setting and resetting the migration lock to enable virtual machines to control migration by themselves and assure that they cannot be migrated if they don’t expect to.

In future work, we will take further study on the usage of migration lock. On the one hand, when a virtual machine should be locked, how long it should be locked and how to avoid using migration lock arbitrarily are the problems to address. On the other hand, virtual machine migration strategies should take the migration lock into account, which will bring challenges to the management of the future virtualized computing environment. Is virtual machine migration lock necessary? We propose the mechanism of migration lock, but whether it will be accepted in the future application scenarios such as virtualized data center needs further study.

## Acknowledgement

This work is supported by National Science Foundation of China under Grant No.61170055, 61232008 and 61272158; the National High Technology Research 863 Program of China under Grant No.2012AA010905; the Research Fund for the Doctoral Program of Higher Education of China under Grant No.20110001110101; Special Foundation of Industry Development for Biology, Internet, New Energy and New Material of Shenzhen under Grant No.JC201104210107A. Zhenlin Wang is also supported by NSF Career CCF0643664.

## References

[1] Christopher Clark, Keir Fraser, Steven Hand, Jacob Corm Hansen, Eric Jul, Christian Limpach, Ian Pratt and Andrew Warfield. Live migration of virtual machines. In Proceedings of the 2nd USENIX Symposium on Networked Systems Design & Implementation (NSDI’05), 2005.5.

[2] Michael Nelson, Beng-Hong Lim and Greg Hutchins. Fast transparent migration for virtual machines, In Proceedings of the USENIX Annual Technical Conference, 2005.4.

[3] Xiaofei Liao, Liting Hu and Hai Jin. Energy optimization schemes in cluster with virtual machines. Cluster Computing, 2010, Volume 13, Number 2, Pages 113-126.

[4] Timothy Wood, Prashant Shenoy, Arun Venkataramani, and Mazin Yousif. Black-box and gray-box strategies for virtual machine migration. In Proceedings of 4th USENIX Symposium on Networked Systems Design & Implementation (NSDI’07), 2007.4.

[5] James E. Smith and Ravi Nair. Virtual machines - versatile platform for systems and processes. Elsevier Inc., 2005.

[6] Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauser, Ian Pratt, Andrew Warfield. Xen and the art of virtualization. In Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP’03), 2003.10.