Exploiting Live Virtual Machine Migration

Jon Oberheide

University of Michigan

February 21, 2008

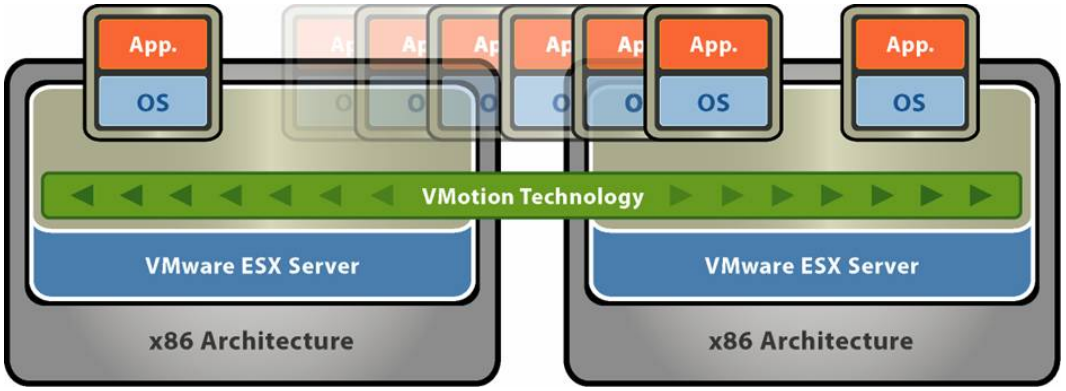
Black Hat DC

Game Plan

- **Introduction to VM migration**
- Live migration security
- Exploiting live migration
- Future attacks and wrap-up

Live VM Migration

- Transfer of a VM from one physical machine to another with little or no service downtime



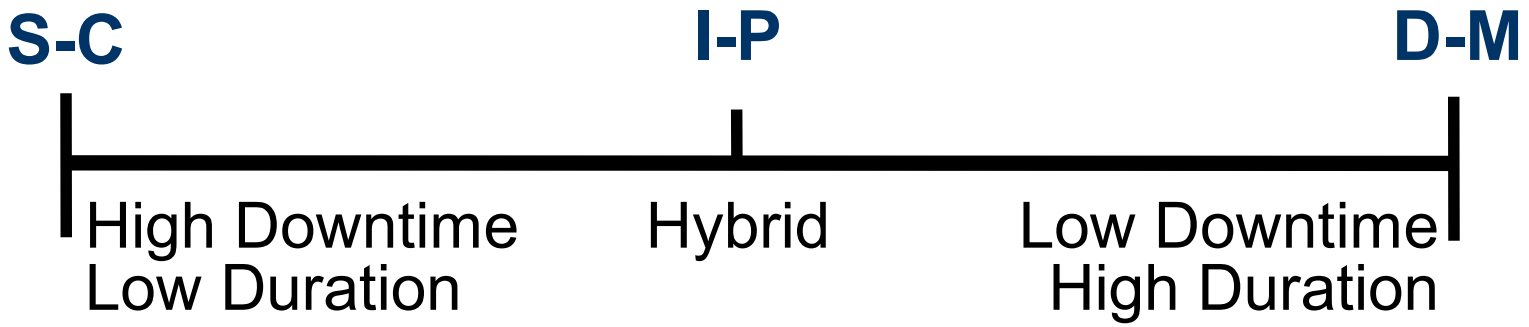
High Availability

Enhanced Mobility

Dynamic Load Balancing

Live Migration Methodology

- Minimize service downtime
- Minimize migration duration
- Migration Types:
  - Stop-and-copy (S-C)
  - Demand-migration (D-M)
  - Iterative precopy (I-P)



Stop and Copy

- Stop source VM
- Copy all pages over the network
- Start destination VM

Stop and Copy

Longest Service Downtime

Shortest Migration Duration

Demand Migration

- Copy over critical OS structures
- Start destination VM
- Page faults trigger network copy

Demand Migration

Shortest Service Downtime

Longest Migration Duration

Iterative Precopy

- Iteratively copy pages over network
- Keep copying dirtied pages until threshold
- At threshold, stop source VM, copy remaining pages, start destination VM

Iterative Precopy

Balances Service Downtime and Migration Duration

- Method used by VMware/Xen

Game Plan

- Introduction to VM migration
- **Live migration security**
- Exploiting live migration
- Future attacks and wrap-up

A Trip Down Memory Lane

- Physical machines
  - Machine state protected by MMU/hardware
  - Physical attacks (firewire device DMA)
- Virtual Machines
  - VM state protected by VMM/hypervisor
  - Software attacks (weak VMM isolation)

Can we break any more isolation boundaries?

A Trip Down Memory Lane

Of course! Functionality always usurps security!

- Migration-enabled VMs
  - Full VM state exposed to network
  - Trades off security for management capabilities
  - Authentication, confidentiality, isolation concerns

VM Migration Security

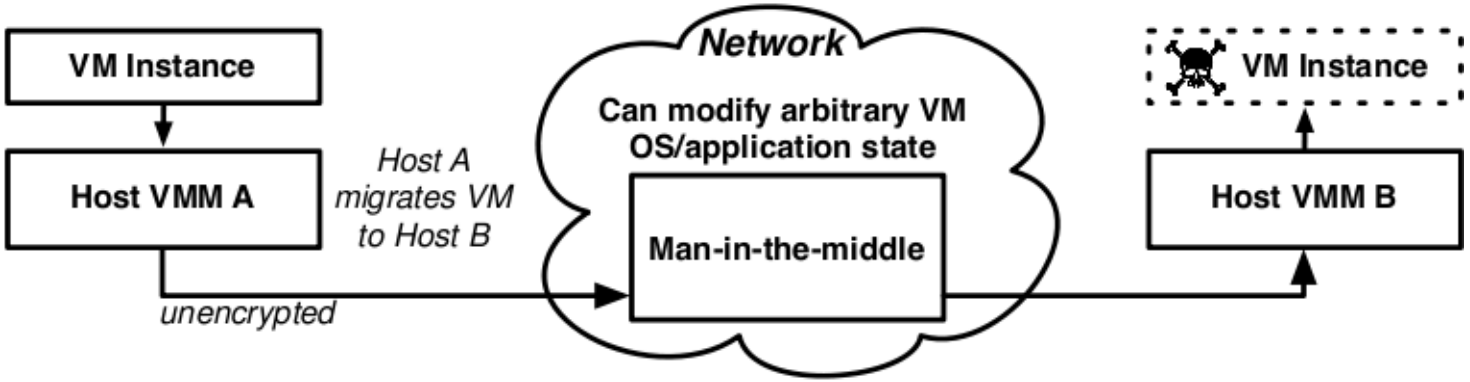
- Migration data plane
  - Network transit path over which migration occurs
- Security of data plane
  - Unauthenticated, insecure migration data plane
- Full access granted to VM state
  - OS/kernel memory
  - Application state
  - Sensitive data, passwords, keys, etc
- VMware and Xen migrations vulnerable

Breaching the Data Plane

- Breach of data plane means game over
  - Entire virtual machine may be compromised
  - Kernel, userspace applications, data
- Requirement for breach
  - Manipulate traffic along migration path between source and destination VMM
- Need to perform MITM attack
  - ARP/DHCP spoofing
  - DNS spoofing/poisoning
  - IP/route hijacking

Breaching the Data Plane

- Passive Attacks
  - Snarf sensitive data, passwords, keys in memory
- Active Attacks
  - Manipulate authentication services
    - sshd, /bin/login, pam, etc
  - Manipulate kernel structures
    - slip rootkits into memory



Game Plan

- Introduction to VM migration
- Live migration security
- **Exploiting live migration**
- Future attacks and wrap-up

Exploiting VM Migration

- Xensploit
  - Non-weapons-grade proof-of-concept tool
  - Works against Xen and VMware migrations
- Attack classes
  - VM application/userland exploits
  - OS/kernel exploits
  - VMM subversion

VM Application Exploits

- sshd authentication bypass
  - Identify pubkey authentication routines
  - Manipulate to allow unrestricted root access
  - Access wide-open after migration completes
- Cron daemon shellcode injection
  - Privileged, inconspicuous daemon
  - Inject HTTP GET + execve shellcode
  - Payload fetch/exec on next find_jobs invocation

Exploitation Example

sshd authentication bypass

- Before migration:
  - attacker denied access to VM
- During migration
  - Xensploit manipulates the in-memory object code of sshd as it crosses the wire
- After migration
  - attacker achieves unrestricted root access to VM

Before Migration

- Attacker attempts to gain root access to the target virtual machine via ssh

```
jonojono@jonojono ~ $ ssh root@vm-test
Password:
Password:
Password:
Permission denied (keyboard-interactive).
jonojono@jonojono ~ $
```

- Attacker is denied access to the VM

sshd Authentication Code

- Source code from OpenSSH's auth2-pubkey.c:

```
        if (key != NULL)
                key_free(key);
        xfree(pkalg);
        xfree(pkblob);
#ifdef HAVE_CYGWIN
        if (check_nt_auth(0, authctxt->pw) == 0)
                authenticated = 0;
#endif
        return authenticated;
}

/* return 1 if user allows given key */
static int
user_key_allowed2(struct passwd *pw, Key *key, char *file)
{
        char line[SSH_MAX_PUBKEY_BYTES];
        int found_key = 0;
        FILE *f;
        u_long linenum = 0;
        struct stat st;
        Key *found;
        char *fp;

        /* Temporarily use the user's uid. */
        temporarily_use_uid(pw);

        debug("trying public key file %s", file);

        /* Fail quietly if file does not exist */
        if (stat(file, &st) < 0) {
                                                        196,2-9       63%
```

During Migration

- Xensploit manipulates the object code of sshd's authentication routines as it crosses the wire

```
805da77:       0f 84 23 fd ff ff       je     805d7a0 <user_key_allowed2+0x80>
805da7d:       89 3c 24                mov    %edi,(%esp)
805da80:       e8 37 e5 fe ff          call   804bfbc <fclose@plt>
805da85:       8d 85 8c df ff ff       lea    0xffffdf8c(%ebp),%eax
805da8b:       89 44 24 04             mov    %eax,0x4(%esp)
805da8f:       c7 04 24 15 0e 08 08    movl   $0x8080e15,(%esp)
805da96:       e8 d5 28 01 00          call   8070370 <logit>
805da9b:       e8 20 bd 01 00          call   80797c0 <restore_uid>
805daa0:       81 c4 9c 20 00 00       add    $0x209c,%esp
805daa6:       31 c0                   xor    %eax,%eax
805daa8:       5b                      pop    %ebx
805daa9:       5e                      pop    %esi
805daaa:       5f                      pop    %edi
805daab:       5d                      pop    %ebp
805daac:       c3                      ret
805daad:       8d 76 00                lea    0x0(%esi),%esi

0805dab0 <user_key_allowed>:
805dab0:       55                      push   %ebp
805dab1:       89 e5                   mov    %esp,%ebp
```

- Xensploit injects a ***mov $0x1,%eax*** instruction into user_key_allowed2, returning 1 (true)

After Migration

- Attacker again attempts to gain root access via ssh on the target virtual machine

```
jonojono@jonojono ~ $ ssh root@vm-test
Last login: Thu Oct 18 15:18:37 2007 from jonojono.eecs.umich.edu
fjbox1 ~ # id
uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10(wheel),11(flop
py),20(dialout),25(at),26(tape),27(video),1006(vmware)
fjbox1 ~ #
```

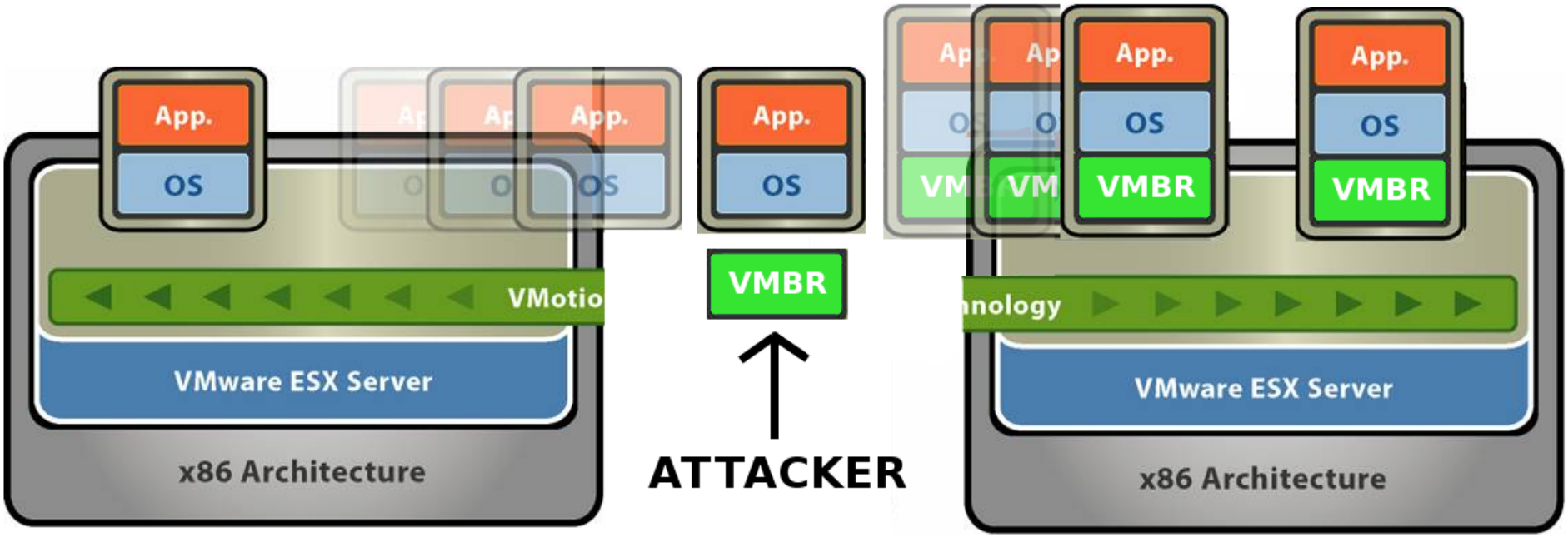
- No authentication is necessary as sshd's routines have been manipulated by Xensploit
- Root access is granted to the attacker

VM Kernel Exploits

- Kernel manipulation
  - Direct access to in-memory kernel image
  - More complexity but more power
  - Leverage all your DMA attack payloads
- Stealthy backdoor drop
  - network/syscall/ioctl trigger
- Full-blown VMBR hoisting

VMBR Hoisting

- Virtual Machine-Based Rootkits
  - Slip in extra virt layer a la SubVirt/Blue Pill/Vitriol



Subverting the VMM

- Mangle migration payload
  - Exploit a vulnerability and subvert VMM
- Leverage Xen dom0 vulns
  - Present in Xen daemon migration routines
  - <= 3.1.0 release vulnerable
  - Undoubtedly more...
- Instantly own all hosted VMs
  - And all future migrated VMs!

Subverting the VMM

- Xen's libxc/xc_domain_restore.c:

```
    unsigned long region_pfn_type[MAX_BATCH_SIZE];
...
    for ( ; ; )
    {
        int j, nr_mfns = 0;
....
        if ( !read_exact(io_fd, &j, sizeof(int)) )
...
        if ( j == -1 )
...
        if ( j == -2 )
...
        if ( j == 0 )
...
        if ( j > MAX_BATCH_SIZE )
...
        if ( !read_exact(io_fd, region_pfn_type, j*sizeof(unsigned long)) )
```

- No check for signed integer j < 0
- Stack overflow of region_pfn_type in Xen VMM

Game Plan

- Introduction to VM migration
- Live migration security
- Exploiting live migration
- **Future attacks and wrap-up**

Additional Attacks

Lots more juice in the migration orange!

- Fraudulent migration requests
  - Owned VMMs snarfing up VMs
- False resource advertising
  - Migration-enabled load balancing
- Future attacks inevitable
  - Increased functionality
  - Increased complexity
  - Increased security risk

Just Encrypt It, Stupid!

- Encryption goes a long way!
- Fingerprinting migrations
  - Reconnaissance / targeting
  - Enabled by iterative-precopy method
  - Similar to VBR attacks
- Increased complexity
  - Full PKI adds considerable deployment complexity
- Not currently implemented!

Vendor Response

- VMware
  - Use separate network for migration paths
  - Use hardware-based crypto cards
  - VMotion/Virtual Infrastructure 3 vulnerable
- XenSource
  - Consult vendor/distribution for security fixes
  - Latest open-source release still at risk
  - Unsure of migration status in XenServer4
- Microsoft Hyper-V
  - Will they get it right?

The Big Picture

- VM migration paradigm
  - VERY useful functionality
  - Awareness of security risk necessary
  - Better isolation, access control, authentication
- Until then...
  - Severe weaknesses exist in extensively deployed systems
  - Valuable weapon for pentester/attacker

Questions?

QUESTIONS?

- Contact info:
  - Jon Oberheide <jonojono@umich.edu>
  - PhD student, University of Michigan
  - Advisor: Farnam Jahanian
  - Research Group: http://www.eecs.umich.edu/fjgroup/