

SOLUTION: How to sync the system clock ...



Uli Zappe 44 posts since

Dec 27, 2006 Hi,

since Fusion Beta 1, I've been running OPENSTEP on Fusion with big success; although an unsupported guest OS, it runs rock-solid. You'll need special VMware drivers for mouse, network, graphics and sound, but thankfully, all these are available on the net. "Shared folders" are easy to emulate via NFS mounts. So basically all VMware tools can be replaced/emulated, except one:

There's no obvious way to automatically reset the system clock of the guest OS. This is necessary in 2 situations:

- OPENSTEP resumes from suspension

In both cases, OPENSTEP does not "know" the system clock has gotten out of sync, and will use the trailing time.

In the following, I will describe how I successfully solved this problem. Though I will describe it for OPENSTEP, it should be basically usable for all non-supported guest OSes, at least if they are Unix-like. So I hope this will be useful for other VMware users. Since the VMware Forum does not seem to have a specific style for code, I will use italics to display code. Example values in the code which you'll have to replace by your own values will be bold.

The basic idea is to use the network connection between guest and host instead of VMware's proprietary communication channel for supported guest OSes. This implies you must have a **bridged network** configuration, so that Mac OS X can address OPENSTEP via the network. In my example, the OPENSTEP virtual machine has the IP **192.168.1.6** and Mac OS X has the IP **192.168.1.1**.

Files in the OPENSTEP guest OS

There are 2 files we need in the OPENSTEP guest OS. First, we need a utility that actually sets the time:

/usr/local/bin/setTime

```
#!/bin/sh  
rdate my.timeserver.org
```

I used rdate since that works most easily on OPENSTEP; but note that not all time servers are rdate compatible.

Since rdate needs root privileges, but root login via rsh/rlogin is disabled on OPENSTEP, we need to make setTime suid root so that we can execute it with normal user privileges:

```
% chmod 4755 /usr/local/bin/setTime
```

Since the virtual machine is only reachable from within my local network, this is fine with me from a security point of view.

The second file we need is in my home directory in OPENSTEP (I use the same user name on OPENSTEP and Mac OS X, and my example will assume this identity):

~/.rhosts

```
192.168.1.1 myusername
```

SOLUTION: How to sync the system clock ...

This file takes care that I will be able to rsh from Mac OS X without having to provide a password.

Files in Mac OS X

First, we create a little program that actually sets the time on the OPENSTEP guest OS remotely:

/usr/local/bin/nswake

```
#!/bin/sh
rsh -t 7 192.168.1.6 /usr/local/bin/setTime 2>/dev/null
sleep 6
```

Note that rsh needs no user name (since you log in from an account of the same name) and no password (because of the .rhosts file installed on OPENSTEP). Since the automatic invocation of this program will also happen when OPENSTEP is suspended or shut down (see below), the -t 7 option makes for a relatively short timeout of rsh in case OPENSTEP is unreachable. sleep 6 takes care that launchd (see below) will not invoke this program several times within a short timespan.

Now we need 2 mechanisms to trigger nswake when (a) OPENSTEP resumes from suspension and (b) Mac OS X awakes from sleep.

(a) is taken care of by Mac OS X's launchd system (note you need Mac OS X >= 10.4 for that!). launchd is set up to control the VMware virtual machine directory. When the guest OS is resumed from suspension (but also when it is suspended), VMware will change the number of files within the virtual machine directory. This can be used to trigger launchd. The following config file takes care of that:

~/Library/LaunchAgents/nswake.plist

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
<key>Label</key>
<string>nswake</string>
<key>Program</key>
<string>/usr/local/bin/nswake</string>
<key>ServiceDescription</key>
<string>Set time in OPENSTEP VMware client after sleep/suspension</string>
<key>WatchPaths</key>
<array>
<string>/my/path/to/virtual/machine/directory/</string>
</array>
</dict>
</plist>
```

SOLUTION: How to sync the system clock ...

(b) is a little trickier; you would think that there's some easy (scriptable) way in Mac OS X to trigger a command based on a wake-up event, but I couldn't find any, so I wrote a tiny Cocoa app to do this. Nothing fancy, but still a bit scary for those who don't know Cocoa. Still, here's the source; if anyone's interested, I could offer an archive with the source and the compiled application.

The app is a normal Cocoa app called WakeNotifier. It has just one class, Controller:

Controller.h

```
#import <Cocoa/Cocoa.h>
@interface Controller : NSObject
{
}
- (void) wakeUp:(id)notification;
@end
```

Controller.m

```
@implementation Controller
- (id) init
{
self = [super init];
if (self != nil)
{
NSNotificationCenter *wakeNotificationCenter = [[NSWorkspace sharedWorkspace] notificationCenter];
[wakeNotificationCenter addObserver:self selector:@selector(wakeUp:) name:NSWorkspaceDidWakeNotification
object:nil];
}
return self;
}
- (void) wakeUp:(id)notification
{
NSLog(@"Setting time on OPENSTEP virtual machine");
[NSTask launchedTaskWithPath:@"/usr/local/bin/nswake" arguments:[NSArray array]];
}
@end
```

As usual with Cocoa apps, you must make the Controller instance a delegate of File's Owner in Interface Builder. Also, you must add these lines to **Info.plist**:

```
<key>LSBackgroundOnly</key>
<string>1</string>
```

SOLUTION: How to sync the system clock ...

This makes WakeNotifier run in the background. To make it actually run as soon as you log in, you must add it to your startup items in System Preferences.

That's it. Now the time will be reliably adjusted in OPENSTEP when you wake up your Mac or resume your guest OS from suspension.

I hope that helps people in similar situations as me.

Uli