

Relax-and-Recover (ReaR) Automated Testing

Gratien D'haese
IT3 Consultants
http://it3.be

http://relax-and-recover.org



Agenda

- Who am I?
- What is Relax-and-Recover?
- Relax-and-Recover Automated Testing
- DEMO

Who am I?



- Gratien D'haese
- IT3 Consultants (company)
  - > 25 years Unix experience
  - Unix/Linux Engineer (incl. Devops)
  - ReaR Support Contracts
- Relax-and-Recover (ReaR)
  - Major Open Source project
  - https://github.com/gdha



Some Basics

- What is Disaster Recovery?

The process by which a business function is restored to the normal, steady state after a disaster

- What is Business Continuity?

The way that a business function will operate after a disaster, until such time as the normal, steady state is restored

Business Continuity





Linux Disaster Recovery

Like any other UNIX Operating System, Linux is vulnerable for disaster to strike

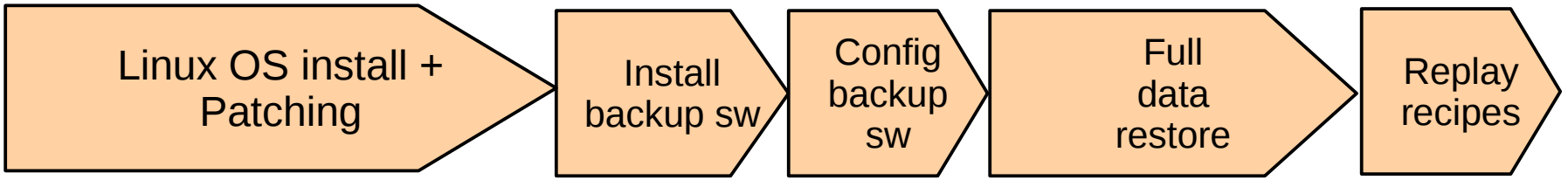
Question: “What shall I do if a disaster strikes?”

Dependent on:

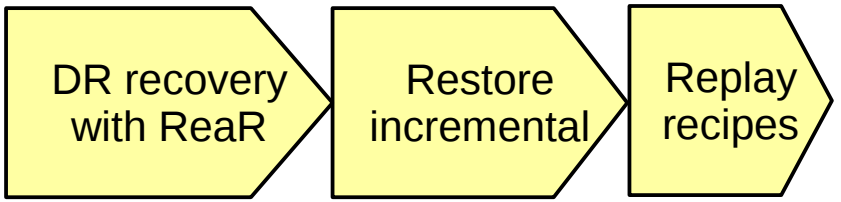
- Hardware failure (e.g. boot disk lost)
- Lost everything (fire, water, earthquake, theft)
- The answer: “Act immediately (with a disaster recovery plan)”

Why are backups not enough?

Disaster Recovery via re-install of Linux and restore data



Disaster Recovery with ReaR and restore data







Relax-and-Recover (ReaR) as DR solution

- ReaR is a tool that implements a DR work-flow for Linux
- Basically meaning:
  - Modular framework written in Bash
  - Easy to extend to own needs
  - Easy to deploy (set up and forget)
  - Integration for various Linux technologies
  - Integration with various back-up solutions
  - Attempts to make system recovery *as easy as possible*



Disaster Recovery – How It Works

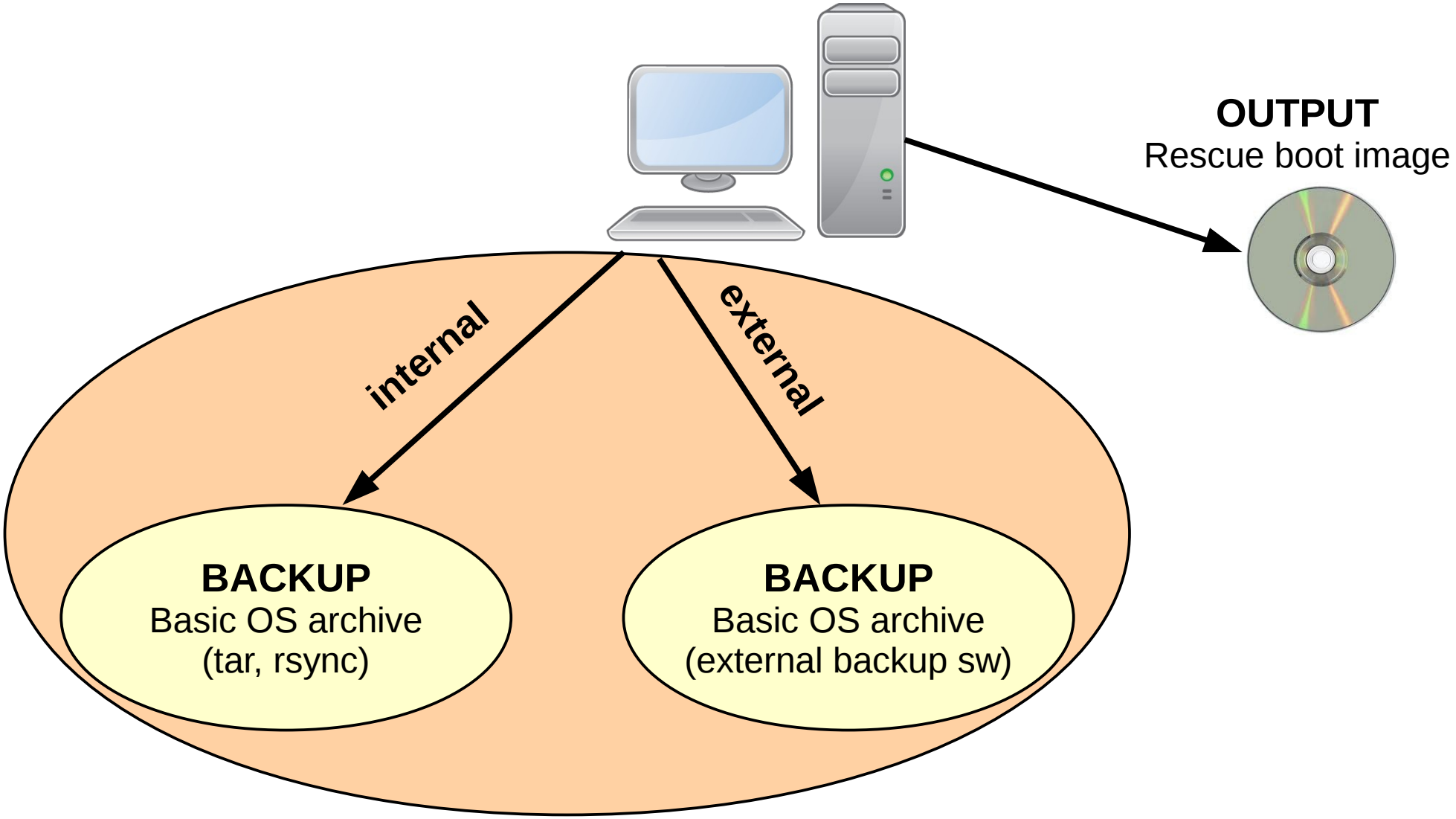
- Store the disk layout
  - Partitioning, LVM and RAID configuration
  - File systems, file system labels ...
  - Boot loader (GRUB, GRUB2, LILO, UEFI)
- Store the files (tgz, rsync, through backup software ...)
- Create bootable rescue media with system configuration (and backup data)
- **Can be done online** (no business interruption)

Disaster Recovery – Rescue Media

- Create “rescue linux” from running system
- Optimally compatible “tool box”
- Clone the system environment
  - Linux kernel and modules
  - Device driver configuration
  - Network configuration
  - Basic system software and tools
- Operate entirely in RAM (initrd)



DR Flow – BACKUP and OUTPUT





Usage of rear

- Shell scripts are stored under /usr/share/rear
- Scripts are executed via work-flows:
  - mkrescue (only make rescue image)
  - mkbackup (including make rescue image)
  - mkbackuponly (excluding make rescue image)
  - recover (the actual recovery part)
- Easy to incorporate new scripts, e.g. for information gathering of Hard- and Software, or other goodies



Getting started with ReaR

- Download it from
  - Our web-site
    - http://relax-and-recover.org/download/
  - The rear-snapshot rpm's build from GitHub
    - http://download.opensuse.org/repositories/Archiving:/Backup:/Rear:/Snapshot/
  - The official source
    - https://github.com/rear/rear
  - The official repo's (Fedora, RHEL and SLES)
    - yum install rear
    - zypper install rear



Testing ReaR

- ReaR is due to a wide range of options difficult to test
- Different Linux flavors are using similar tools with some minor differences :-(
- Too much to be able to test everything before a new release, and sometimes it is even not possible just because we do not have the hardware
- Is and will stay a challenge for the future...



Automated ReaR Testing

- Finding a right balance between CI Testing and Automated ReaR Testing
- Via OpenSuse Build Services we build daily a fresh ReaR package (for free)
- We have some excellent ReaR developers who do lots of coding (for free)
- Support via GitHub issues
  - Free support
  - Commercial support

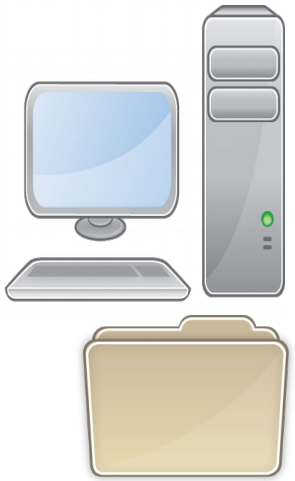


Automated ReaR Testing

- For customers with a support contract we wrote “Automated ReaR Testing”
- Currently only for CentOS 7 Linux distribution
- We start the process and it automatically does
  - DR image creation
  - Backup of system
  - PXE boot the recover system with DR image made
  - Restore backup
  - Reboot the recover system

Test Configuration

Hypervisor (**vagrant-host** computer)
192.168.33.1



PXE
Boot
(virtualbox)

PXE
Boot
(libvirt)

Virtual
machines

Client
192.168.33.10

Server
192.168.33.15

Recover
192.168.33.5

192.168.33.0

DHCP network



Set up vagrant environment

- Host system must be GNU/Linux based
- A hypervisor like KVM or VirtualBox
- Install “vagrant” from your distro, or from https://www.vagrantup.com/downloads.html
- KVM with libvirt needs the vagrant-libvirt plugin vagrant plugin install vagrant-libvirt
- Install “git” software to clone the Vagrantfile and scripts



Install the automated ReaR Testing software

- Is Open Source and licensed under GPLv3
- New code is written only for customers with a valid ReaR Support contracts
- git clone git@github.com:gdha/rear-automated-testing.git
- Go into directory “rear-automated-testing”
- Type “./rear-automated-test.sh -h” to see info
- Uses “*vagrant*” to drive the creation of the VMs
- Account vagrant/vagrant (and root/vagrant)

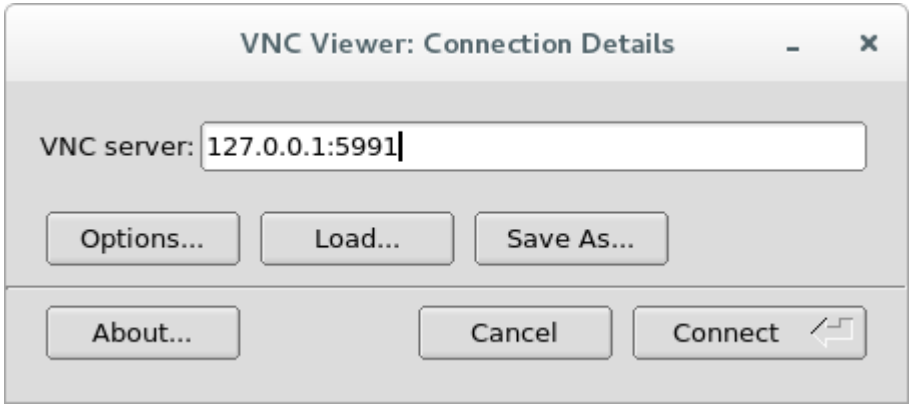


Login via vagrant or ssh

- Login via vagrant:
  - sudo vagrant ssh client
  - sudo vagrant ssh server
- Another way to login is via ssh:
  - ssh root@192.168.33.10 (client root pw is vagrant)
  - ssh root@192.168.33.15 (server root pw is vagrant)

Login via vncviewer

- If you install “tigervnc” you can use vncviewer
- Use address 127.0.0.1 (localhost)
- Port 5991 for “client”
- Port 5992 for “server”
- Port 5993 for “recover”





Try it yourself?

- https://github.com/gdha/rear-automated-testing

```
$ sudo ./rear-automated-test.sh -p libvirt

+-----------------------------------------------+
|    Relax-and-Recover Automated Testing script |
|              version 1.0                      |
+-----------------------------------------------+

Author: Gratien D'haese
Copyright: GPL v3

Current distro directory is centos7
Copy the Vagrantfile.libvirt to Vagrantfile
Bringing up the vagrant VMs client and server
Bringing machine 'client' up with 'libvirt' provider...
Bringing machine 'server' up with 'libvirt' provider...
==> server: Starting domain.
==> client: Starting domain.
```



Automated ReaR Testing

- Automated ReaR Testing speeds up
  - Validation processes
  - Bug hunting
  - Testing with beakerlib (option -t)
  - Trusworthy of “unstable” ReaR repository
  - Acceptance of ReaR within corporations and Open Source vendors
- Currently only available for CentOS 7 Linux distribution (situation March 2017)
- We accept pull requests and sponsoring

ReaR Automated Testing

demonstration



Need more of this?

We can foresee in a customized workshop on consultancy basis, or set-up in-house full automated ReaR testing for customers with a valid support contract

See http://www.it3.be/rear-support



Gratien D'haese
IT3 Consultants



Backup Slides in case live demonstration
Is not possible



The help and usage page

```
$ sudo ./rear-automated-test.sh -h
+-------------------------------------------------------------+
|    Relax-and-Recover Automated Testing script               |
|           version 1.0                                       |
+-------------------------------------------------------------+

Author: Gratien D'haese
Copyright: GPL v3

Usage: rear-automated-test.sh [-d distro] [-b <boot method>] [-s <server IP>] [-p
provider] [-c rear-config-file.conf] [-t test] -vh

      -d: The distribution to use for this automated test (default: centos7)
      -b: The boot method to use by our automated test (default: PXE)
      -s: The <boot server> IP address (default: 192.168.33.15)
      -p: The vagrant <provider> to use (default: virtualbox)
      -c: The ReaR config file we want to use with this test (default: PXE-
                                          booting-example-with-URL-style.conf)
      -t: The ReaR validation test directory (see tests directory; no default)
      -h: This help message.
      -v: Revision number of this script.
```



Starting the script

```
$ sudo ./rear-automated-test.sh
+-----------------------------------------------+
|   Relax-and-Recover Automated Testing script  |
|           version 1.0                         |
+-----------------------------------------------+

Author: Gratien D'haese
Copyright: GPL v3

Current distro directory is centos7
Copy the Vagrantfile.virtualbox to Vagrantfile
Bringing up the vagrant VMs client and server
Bringing machine 'client' up with 'virtualbox' provider...
Bringing machine 'server' up with 'virtualbox' provider...
==> client: Checking if box 'centos/7' is up to date...
==> client: A newer version of the box 'centos/7' is available! You currently
```




```
==> server: Setting hostname...
==> server: Configuring and enabling network interfaces...
==> server: Rsyncing folder: /projects/rear/rear-automated-testing/centos7/ => /vagrant
==> server: Machine already provisioned. Run `vagrant provision` or use the `--provision`
==> server: flag to force provisioning. Provisioners marked to run always will still run.

Sleep for 5 seconds [Control-C is now possible]

----------------------------------------------------------------------
Current machine states:

client                running (virtualbox)
server                running (virtualbox)
recover               poweroff (virtualbox)

This environment represents multiple VMs. The VMs are all listed
above with their current state. For more information about a specific
VM, run `vagrant status NAME`.
----------------------------------------------------------------------
```



Check if 'eth1' is active on client [known issue https://github.com/mitchellh/vagrant/issues/8166]
Connection to 127.0.0.1 closed.
Check if 'eth1' is active on server
Connection to 127.0.0.1 closed.
Doing ping tests to VMs client and server
client is up and running - ping test OK
server is up and running - ping test OK



Update rear on the VM client

Loaded plugins: fastestmirror
Determining fastest mirrors
 * base: ftp.belnet.be
Resolving Dependencies
--> Running transaction check
---> Package rear.x86_64 0:2.00-44.git201702131649.el7 will be updated
---> Package rear.x86_64 0:2.00-74.git201703170925.el7 will be an update
--> Finished Dependency Resolution

Dependencies Resolved

Package	Arch	Version	Repository	Size
Updating:				
rear	x86_64	2.00-74.git201703170925.el7	Archiving_Backup_Rear_Snapshot	459 k



ReaR version that will be tested is:

Relax-and-Recover 2.00-git201703170925 / 2017-03-17

Content of /etc/rear/local.conf is:

```
OUTPUT=PXE
OUTPUT_PREFIX_PXE=$HOSTNAME
BACKUP=NETFS
BACKUP_URL=nfs://192.168.33.15/export/nfs/tftpboot
PXE_TFTP_URL=nfs://10.0.2.2/root/.config/VirtualBox/TFTP
PXE_CONFIG_URL=nfs://10.0.2.2/root/.config/VirtualBox/TFTP/pxelinux.cfg
PXE_RECOVER_MODE="unattended"
PXE_CREATE_LINKS="IP"

USE_STATIC_NETWORKING=y
KERNEL_CMDLINE="$KERNEL_CMDLINE net.ifnames=0"
```



Run rear –v mkbackup

Relax-and-Recover 2.00-git201703170925 / 2017-03-17
Using log file: /var/log/rear/rear-client.log
Using backup archive '/tmp/rear.mbxNVLH8555SO9t/outputfs/client/backup.tar.gz'
Creating disk layout
Creating root filesystem layout
Copying logfile /var/log/rear/rear-client.log into initramfs as '/tmp/rear-client-partial-2017-03-19T09:43:41+0100.log'
Copied kernel+initrd 108M to nfs://10.0.2.2/root/.config/VirtualBox/TFTP
Created pxelinux config 'rear-client' and symlinks for IP adresses in nfs://10.0.2.2/root/.config/VirtualBox/TFTP/pxelinux.cfg
Copying resulting files to nfs location
Saving /var/log/rear/rear-client.log as rear-client.log to nfs location
Creating tar archive '/tmp/rear.mbxNVLH8555SO9t/outputfs/client/backup.tar.gz'
Preparing archive operation
Archived 410 MiB in 64 seconds [avg 6567 KiB/sec]
The rear mkbackup was successful



Make client area readable for others on PXE boot server 10.0.2.2

Halting the client VM before doing the recovery

Recover VM will use the client IP address after it has been fully restored

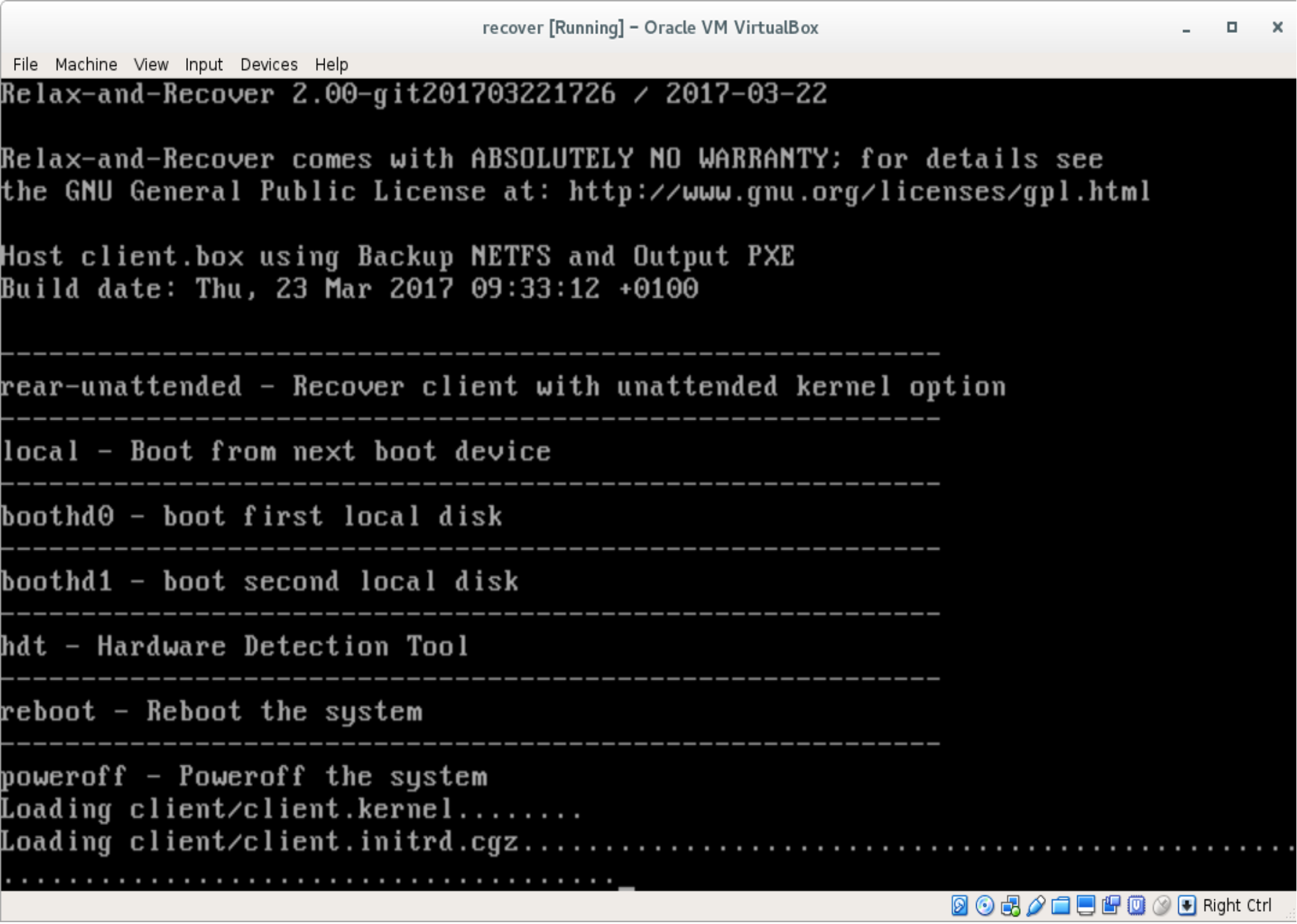
==> client: Attempting graceful shutdown of VM...

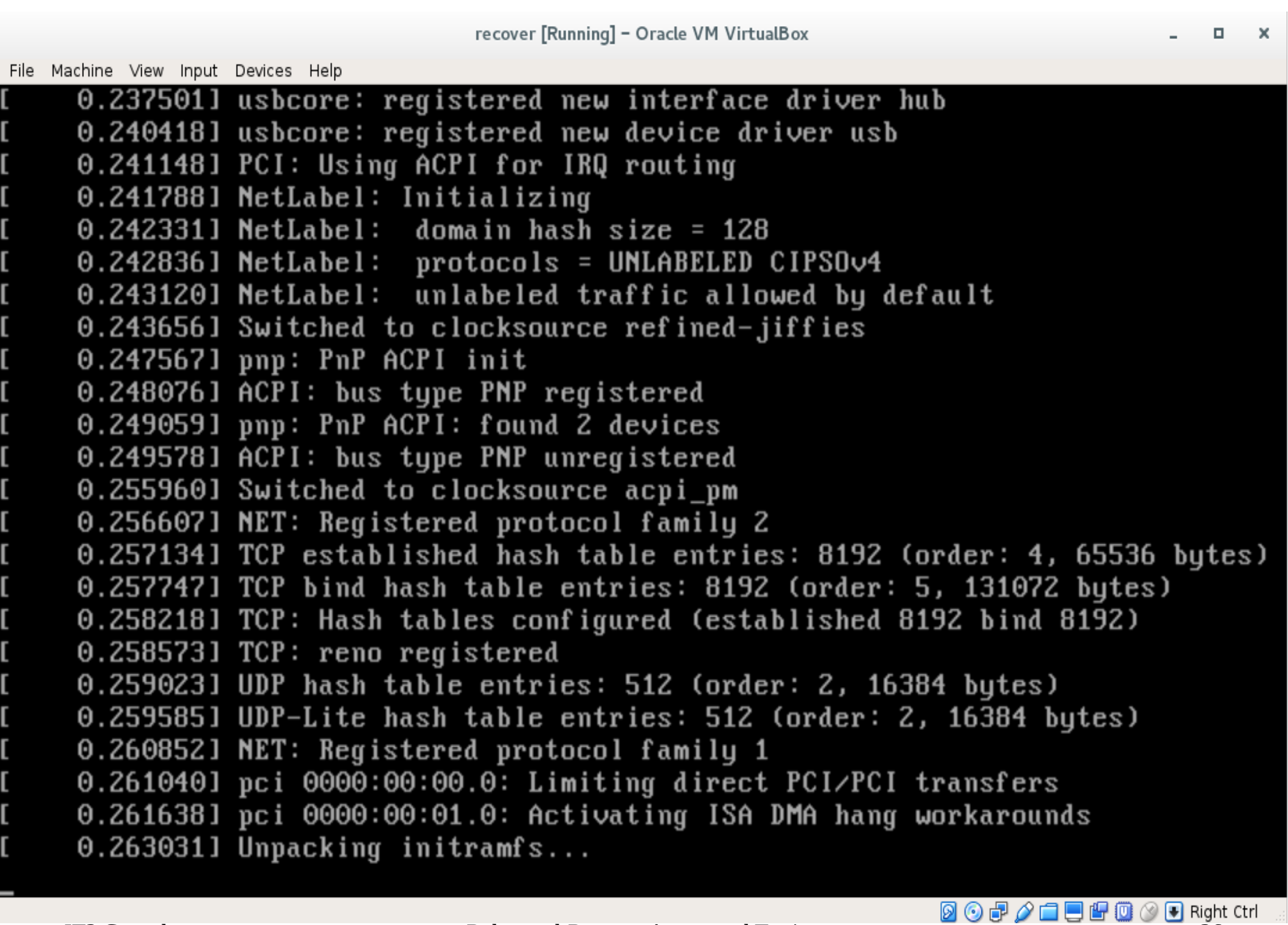
Copied private key of client VB to recover VB config area

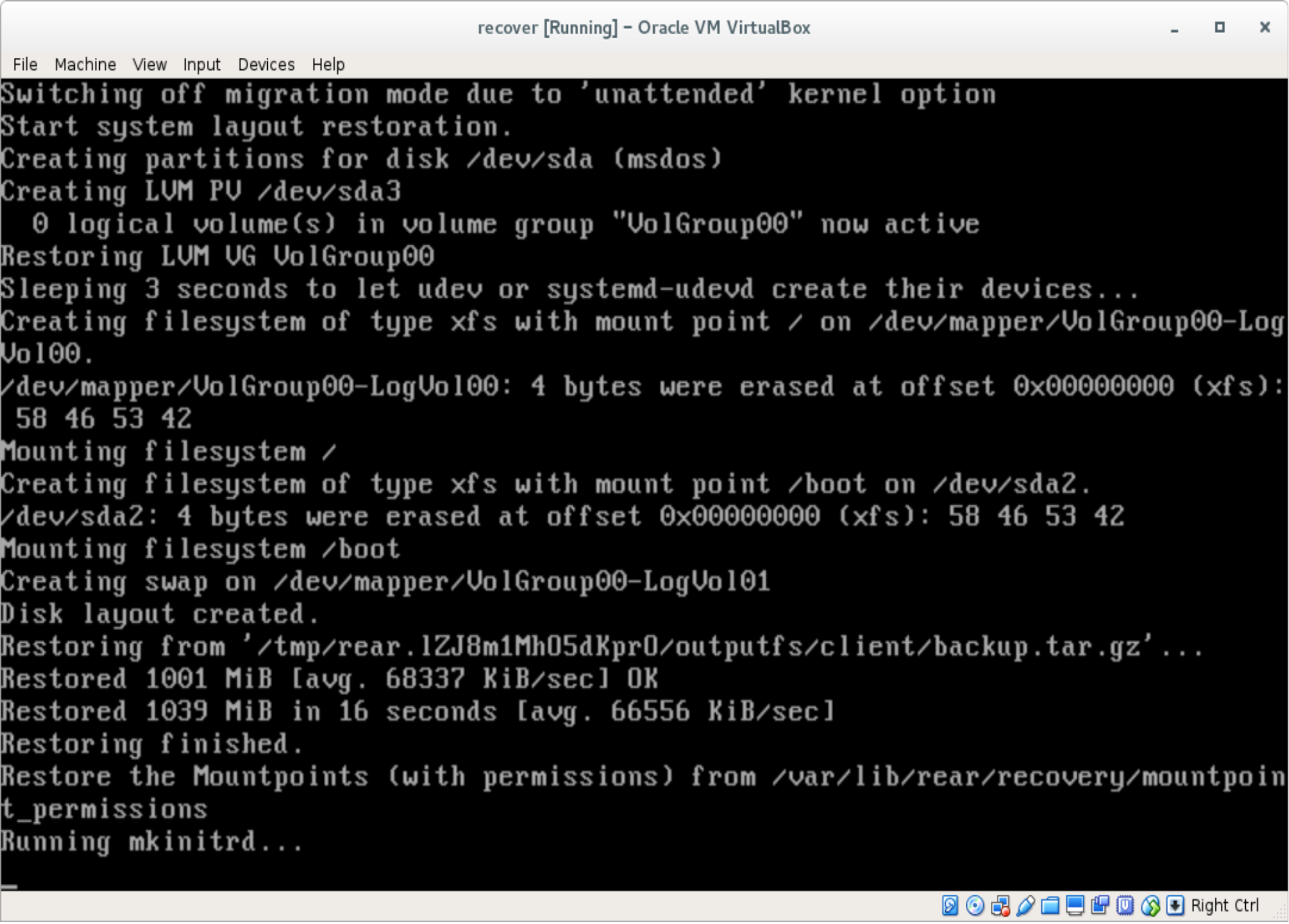
Starting the recover VM

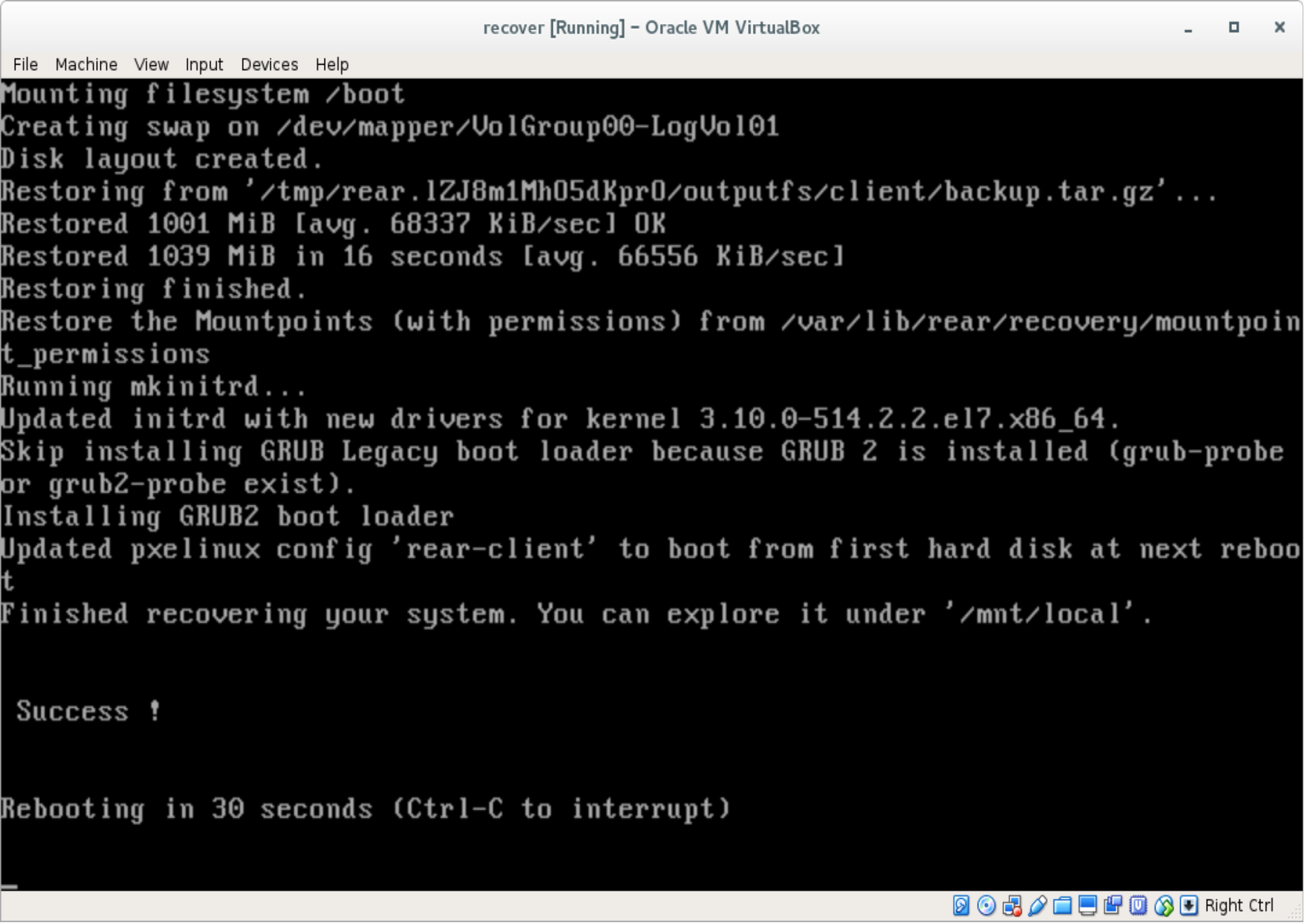
Bringing machine 'recover' up with 'virtualbox' provider...

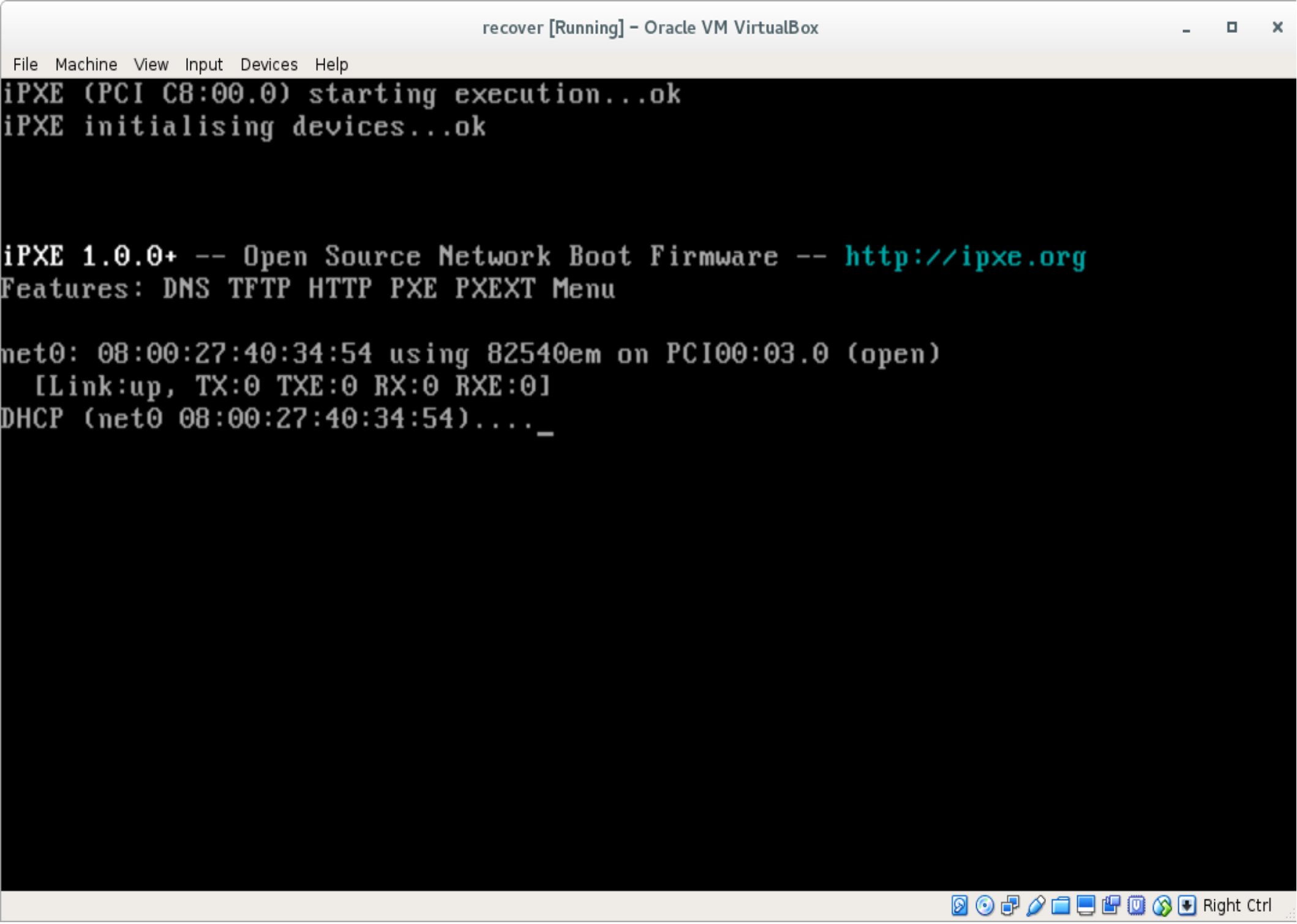
==> recover: Checking if box 'clink15/pxe' is up to date...

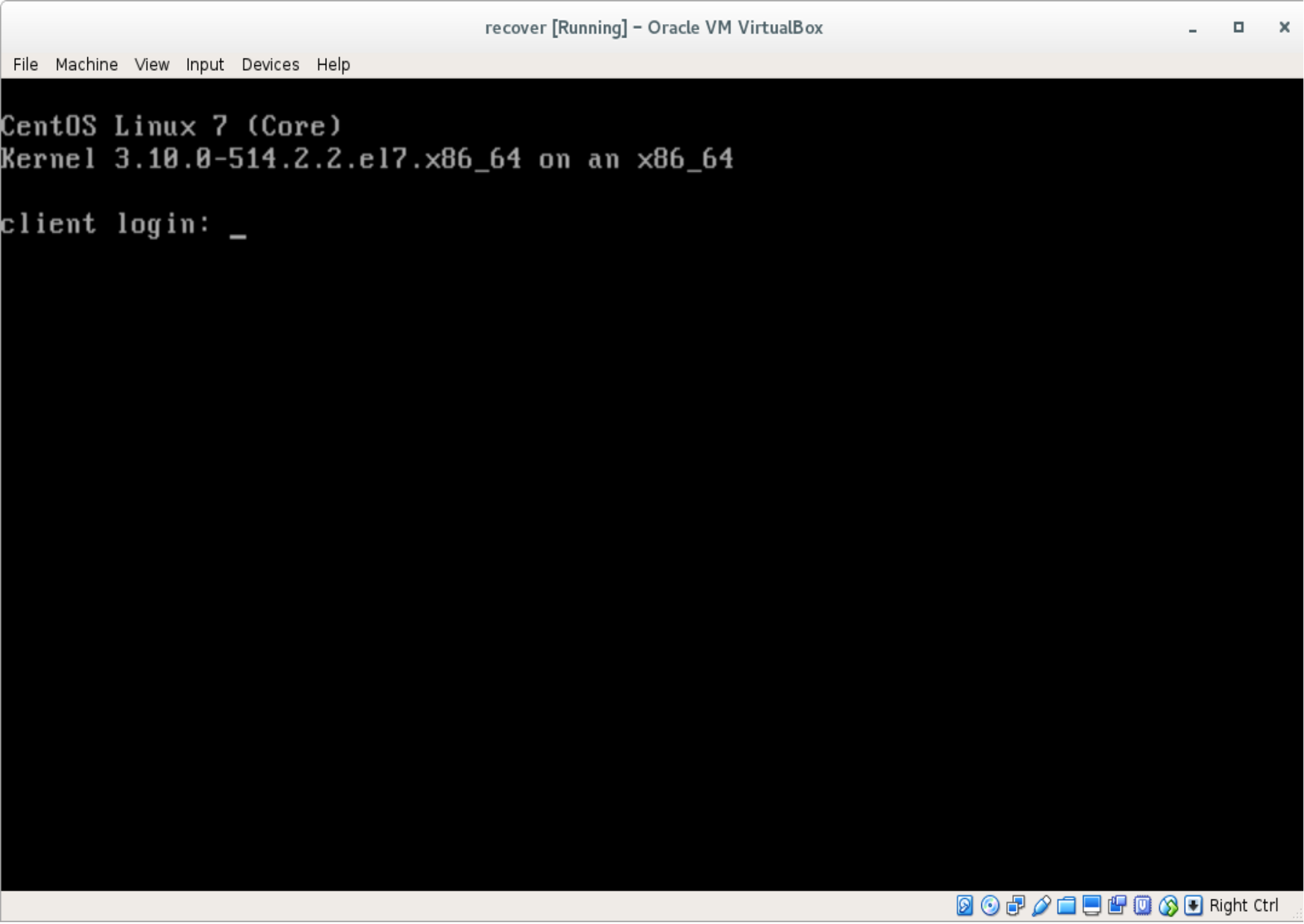

recover [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Relax-and-Recover 2.00-git201703221726 / 2017-03-22
Relax-and-Recover comes with ABSOLUTELY NO WARRANTY; for details see
the GNU General Public License at: http://www.gnu.org/licenses/gpl.html
Host client.box using Backup NETFS and Output PXE
Build date: Thu, 23 Mar 2017 09:33:12 +0100
rear-unattended - Recover client with unattended kernel option
local - Boot from next boot device
boothd0 - boot first local disk
boothd1 - boot second local disk
hdt - Hardware Detection Tool
reboot - Reboot the system
poweroff - Poweroff the system
Loading client/client.kernel.....
Loading client/client.initrd.cgz.....
Right Ctrl

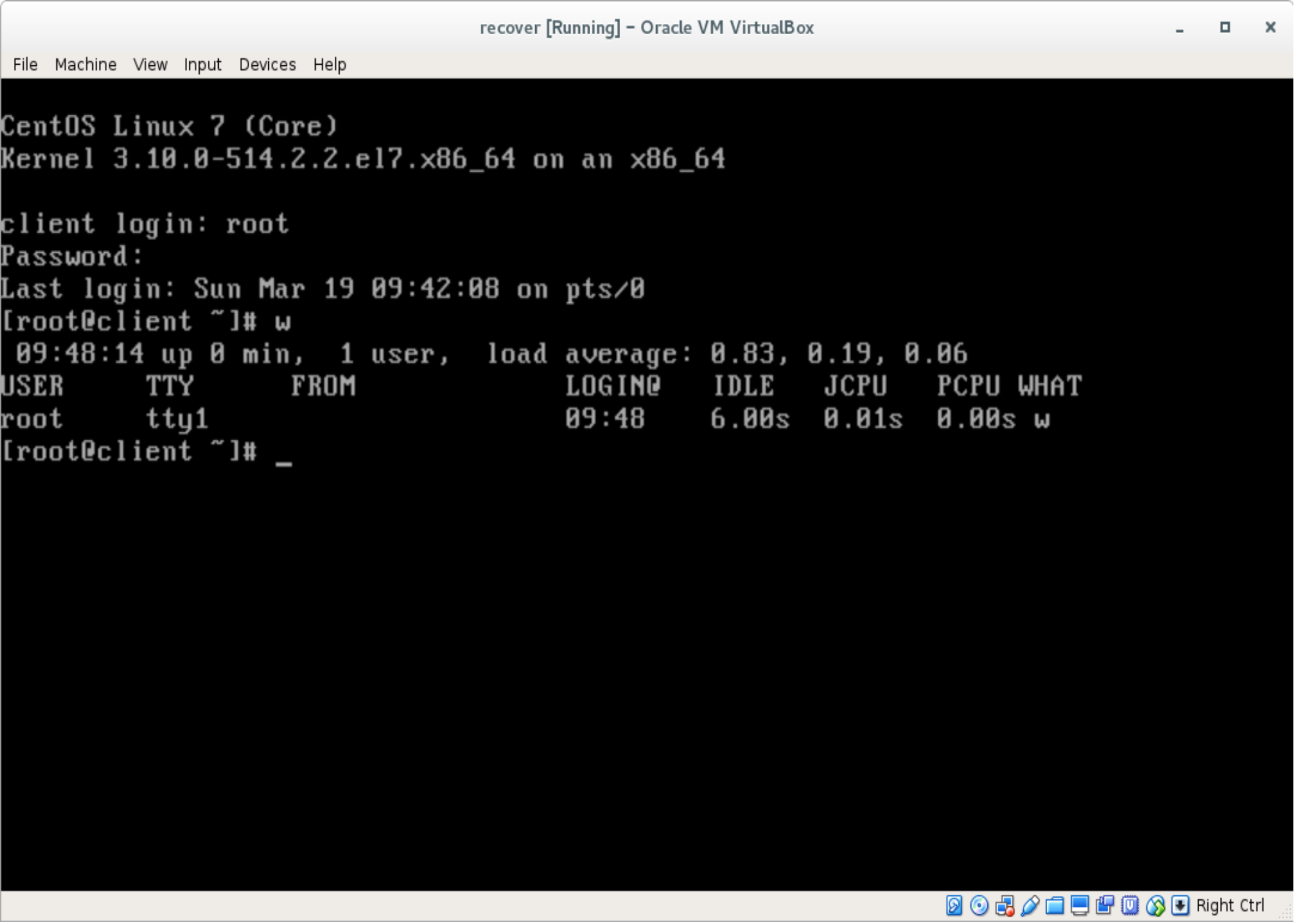

recover [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
[0.237501] usbcore: registered new interface driver hub
[0.240418] usbcore: registered new device driver usb
[0.241148] PCI: Using ACPI for IRQ routing
[0.241788] NetLabel: Initializing
[0.242331] NetLabel: domain hash size = 128
[0.242836] NetLabel: protocols = UNLABELED CIPSOv4
[0.243120] NetLabel: unlabeled traffic allowed by default
[0.243656] Switched to clocksource refined-jiffies
[0.247567] pnp: PnP ACPI init
[0.248076] ACPI: bus type PNP registered
[0.249059] pnp: PnP ACPI: found 2 devices
[0.249578] ACPI: bus type PNP unregistered
[0.255960] Switched to clocksource acpi_pm
[0.256607] NET: Registered protocol family 2
[0.257134] TCP established hash table entries: 8192 (order: 4, 65536 bytes)
[0.257747] TCP bind hash table entries: 8192 (order: 5, 131072 bytes)
[0.258218] TCP: Hash tables configured (established 8192 bind 8192)
[0.258573] TCP: reno registered
[0.259023] UDP hash table entries: 512 (order: 2, 16384 bytes)
[0.259585] UDP-Lite hash table entries: 512 (order: 2, 16384 bytes)
[0.260852] NET: Registered protocol family 1
[0.261040] pci 0000:00:00.0: Limiting direct PCI/PCI transfers
[0.261638] pci 0000:00:01.0: Activating ISA DMA hang workarounds
[0.263031] Unpacking initramfs...
Right Ctrl


recover [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Switching off migration mode due to 'unattended' kernel option
Start system layout restoration.
Creating partitions for disk /dev/sda (msdos)
Creating LVM PV /dev/sda3
0 logical volume(s) in volume group "VolGroup00" now active
Restoring LVM VG VolGroup00
Sleeping 3 seconds to let udev or systemd-udevd create their devices...
Creating filesystem of type xfs with mount point / on /dev/mapper/VolGroup00-LogVol00.
/dev/mapper/VolGroup00-LogVol00: 4 bytes were erased at offset 0x00000000 (xfs): 58 46 53 42
Mounting filesystem /
Creating filesystem of type xfs with mount point /boot on /dev/sda2.
/dev/sda2: 4 bytes were erased at offset 0x00000000 (xfs): 58 46 53 42
Mounting filesystem /boot
Creating swap on /dev/mapper/VolGroup00-LogVol01
Disk layout created.
Restoring from '/tmp/rear.lZJ8m1MhO5dKprO/outputfs/client/backup.tar.gz'...
Restored 1001 MiB [avg. 68337 KiB/sec] OK
Restored 1039 MiB in 16 seconds [avg. 66556 KiB/sec]
Restoring finished.
Restore the Mountpoints (with permissions) from /var/lib/rear/recovery/mountpoint_permissions
Running mkinitrd...
Right Ctrl


recover [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Mounting filesystem /boot
Creating swap on /dev/mapper/VolGroup00-LogVol01
Disk layout created.
Restoring from '/tmp/rear.lZJ8m1MhO5dKprO/outputfs/client/backup.tar.gz'...
Restored 1001 MiB [avg. 68337 KiB/sec] OK
Restored 1039 MiB in 16 seconds [avg. 66556 KiB/sec]
Restoring finished.
Restore the Mountpoints (with permissions) from /var/lib/rear/recovery/mountpoint_permissions
Running mkinitrd...
Updated initrd with new drivers for kernel 3.10.0-514.2.2.el7.x86_64.
Skip installing GRUB Legacy boot loader because GRUB 2 is installed (grub-probe or grub2-probe exist).
Installing GRUB2 boot loader
Updated pxelinux config 'rear-client' to boot from first hard disk at next reboot
Finished recovering your system. You can explore it under '/mnt/local'.
Success !
Rebooting in 30 seconds (Ctrl-C to interrupt)
Right Ctrl


recover [Running] – Oracle VM VirtualBox
File Machine View Input Devices Help
iPXE (PCI C8:00.0) starting execution...ok
iPXE initialising devices...ok
iPXE 1.0.0+ -- Open Source Network Boot Firmware -- http://ipxe.org
Features: DNS TFTP HTTP PXE PXEXT Menu
net0: 08:00:27:40:34:54 using 82540em on PCI00:03.0 (open)
[Link:up, TX:0 TXE:0 RX:0 RXE:0]
DHCP (net0 08:00:27:40:34:54)....
Right Ctrl


recover [Running] – Oracle VM VirtualBox
File Machine View Input Devices Help
CentOS Linux 7 (Core)
Kernel 3.10.0-514.2.2.el7.x86_64 on an x86_64
client login: _
Right Ctrl


recover [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
CentOS Linux 7 (Core)
Kernel 3.10.0-514.2.2.el7.x86_64 on an x86_64
client login: root
Password:
Last login: Sun Mar 19 09:42:08 on pts/0
[root@client ~]# w
09:48:14 up 0 min, 1 user, load average: 0.83, 0.19, 0.06
USER TTY FROM LOGIN@ IDLE JCPU PCPU WHAT
root tty1 09:48 6.00s 0.01s 0.00s w
[root@client ~]# _
Right Ctrl



Beaker Tests

- sudo ./rear-automated-test.sh -t check-changed-disk-layout

```
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
:: [   LOG    ] :: Test option rear checklayout
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
:: [   PASS   ] :: Command 'rear -v checklayout > output' (Expected 0, got 0)
:: [   PASS   ] :: Command 'grep 'Disk layout is identical.' output' (Expected 0, got 0)
:: [   PASS   ] :: There should be at least one swap disk (Assert: "3" should be >= "2")
:: [   PASS   ] :: Stop all swap disks (Expected 0, got 0)
:: [   PASS   ] :: Command 'rear -v checklayout > output' (Expected 1, got 1)
:: [   PASS   ] :: Command 'grep 'Disk layout has changed.' output' (Expected 0, got 0)
:: [   PASS   ] :: Start all swap disks (Expected 0, got 0)
:: [   PASS   ] :: Command 'rear -v checklayout > output' (Expected 0, got 0)
:: [   PASS   ] :: Command 'grep 'Disk layout is identical.' output' (Expected 0, got 0)
:: [   LOG    ] :: Duration: 2s
:: [   LOG    ] :: Assertions: 9 good, 0 bad
:: [   PASS   ] :: RESULT: Test option rear checklayout
```