VIRTUAL TAPE DEVICE DRIVER

Kavitha B K

1BM05CS030

Rashmi Balasubramanyam

1BM05CS061

Rashmi H

1BM05CS062

Rohith Subramanyam

1BM05CS065

Under the guidance of

Mr. K N Shylesh
Lecturer
Dept. of Computer Science & Enginnering
B.M.S College of Engineering

Mr. Vivek B Joshi
Technical Team
Sun Microsystems India Pvt. Ltd.
Bangalore

AGENDA

- INTRODUCTION
- SYSTEM REQUIREMENT AND SPECIFICATION
- SYSTEM DESIGN
- SYSTEM IMPLEMENTATION
- SOFTWARE TESTING
- CONCLUSION & FUTURE ENHANCEMENTS

INTRODUCTION

- **Magnetic Tape**
  - Medium for storage
  - Primary and reliable way to backup and archive data
- **Tape Drive**
  - To read and write magnetic tapes
- **NDMP**
  - Used to backup data from **Network Attached Storage (NAS)** makes use of tapes

Properties of tape

- Sequential-access of data
- Very low average seek times
- Capacity- few megabytes to hundreds of gigabytes
- Low unit cost and long archival stability

Why virtualize a tape drive? Motivation for our project

During testing and validation of backup/storage specific applications, a physical tape is not always feasible due to the following time, space and cost factors:

- **Slow** average seek times
- **Expensive**
- Virtual Tape Libraries (VTL) available are **not open sourced**

Aim of our project

- To develop a pseudo tape driver for OpenSolaris which emulates a raw file as a tape
- Provide all the functionalities such as open, close, read, write, rewind, forward, seek, etc
- Use in quick testing in development environments (especially NAS and/or NDMP)

AGENDA

- ✓ INTRODUCTION
- SYSTEM REQUIREMENT AND SPECIFICATION
- SYSTEM DESIGN
- SYSTEM IMPLEMENTATION
- SOFTWARE TESTING
- CONCLUSION & FUTURE ENHANCEMENTS

SYSTEM REQUIREMENT SPECIFICATION

- Environment/Operating System:

  OpenSolaris

- Tools:

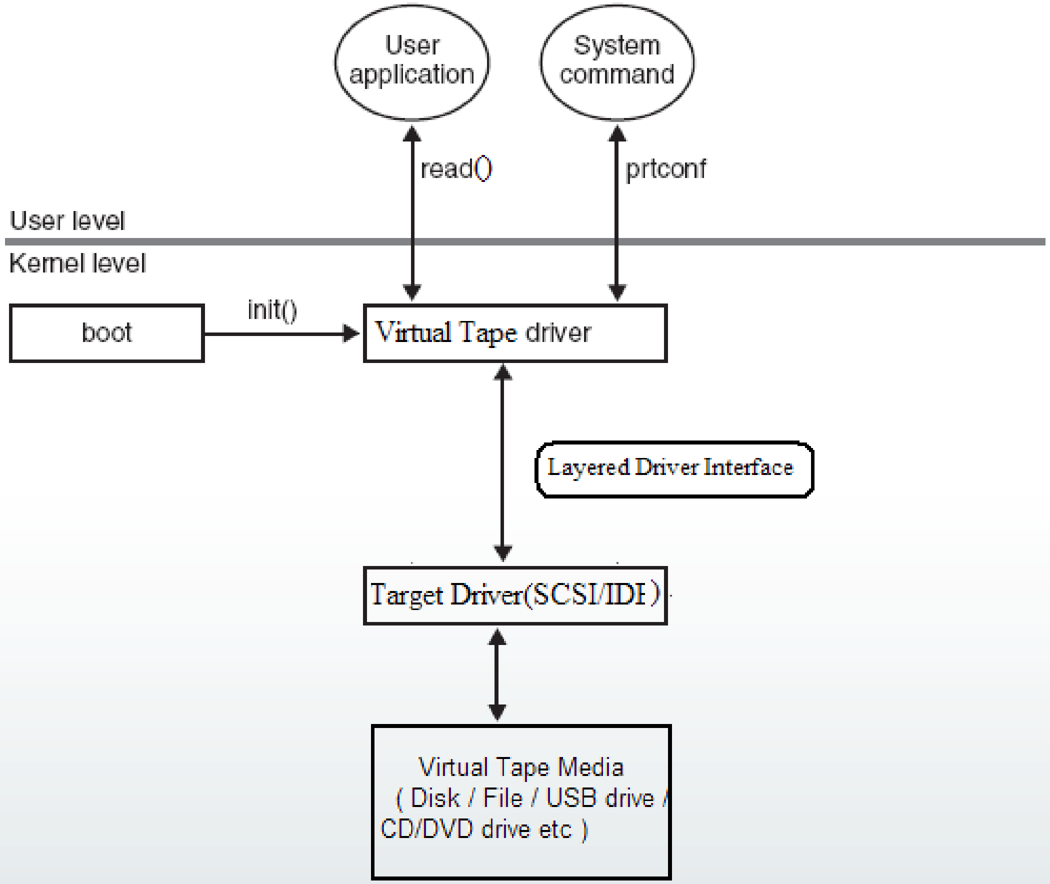
  Sun Studio

AGENDA

- ✓ INTRODUCTION
- ✓ SYSTEM REQUIREMENT AND SPECIFICATION
- SYSTEM DESIGN
- SYSTEM IMPLEMENTATION
- SOFTWARE TESTING
- CONCLUSION & FUTURE ENHANCEMENTS

SYSTEM DESIGN

Architecture



- ***Application program:***

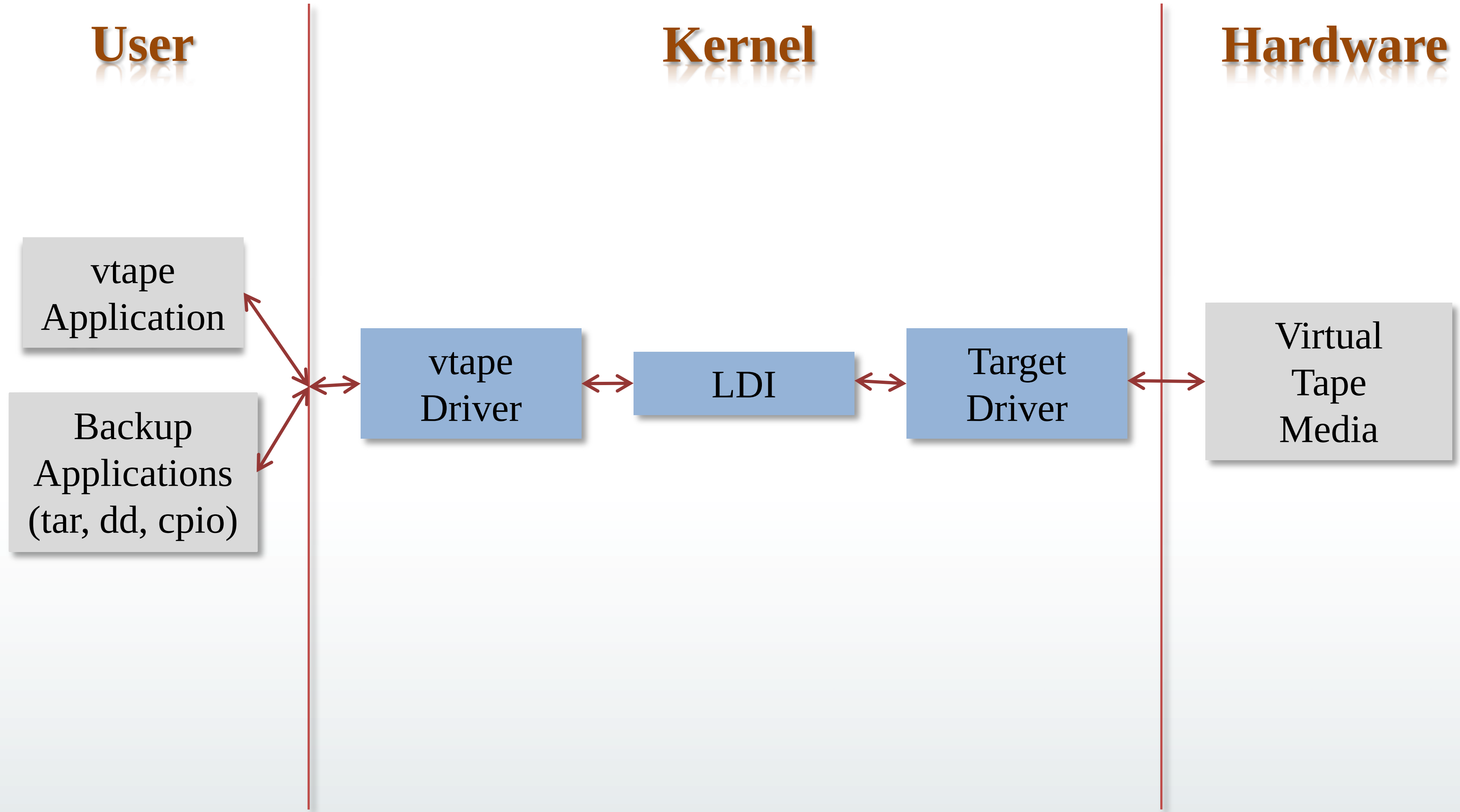
  When an application program in the User Land attempts to access a tape device the corresponding pseudo-driver is invoked

- ***Virtual/Pseudo Tape Driver:***

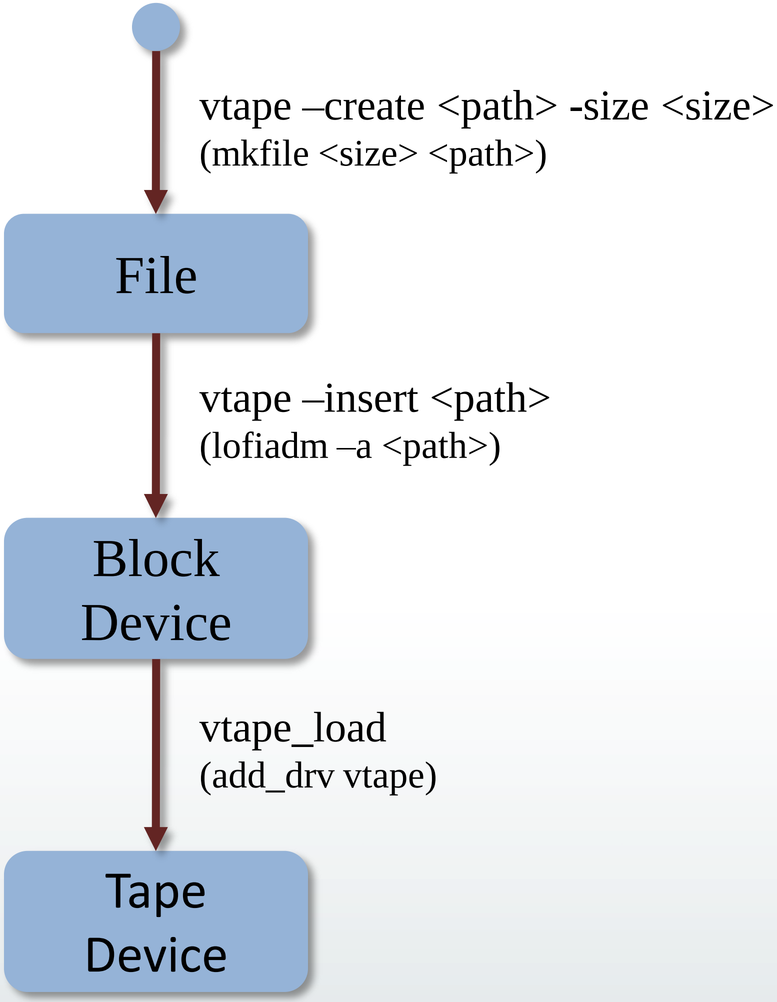
  - Loading and unloading the driver
  - Auto configuring the device
  - Providing I/O services for the driver

The above operations are accomplished using the services of the corresponding driver with the help of the Layered Driver Interfaces (LDI).

Deployment Diagram



File to Tape Device Transition



vtape –create <path> -size <size>
(mkfile <size> <path>)

File

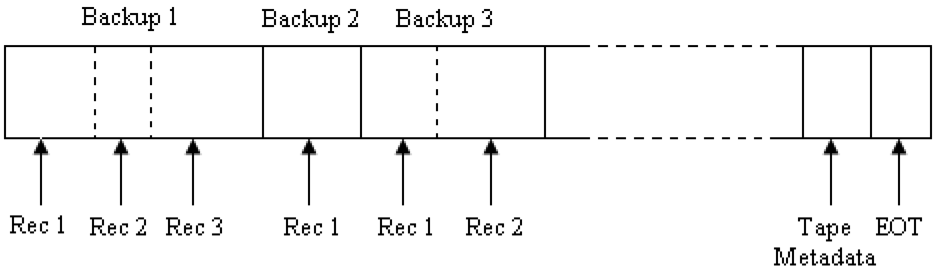
vtape –insert <path>
(lofiadm –a <path>)

Block Device

vtape_load
(add_drv vtape)

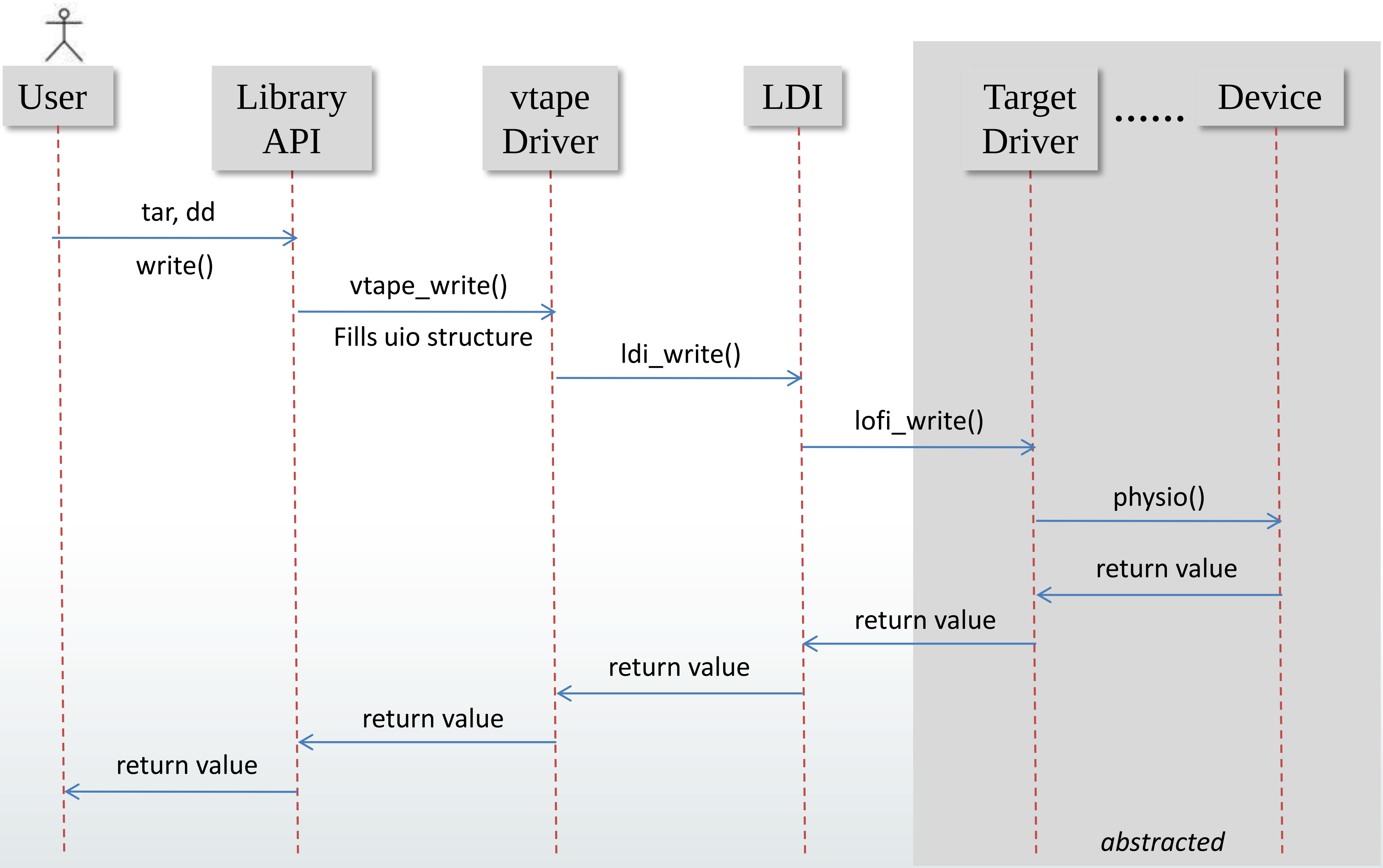
Tape Device

Structure of the tape

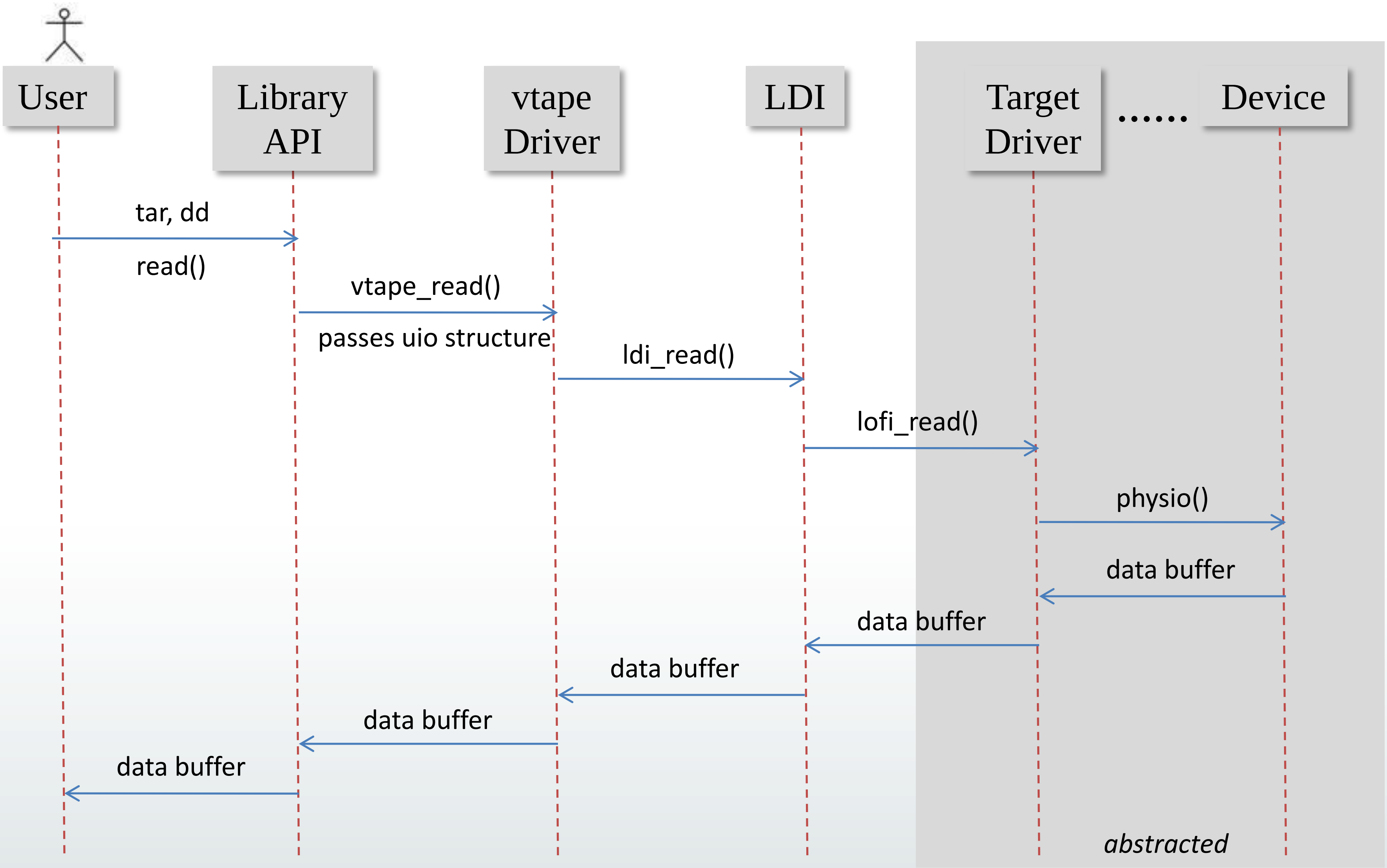


Sequence Diagrams

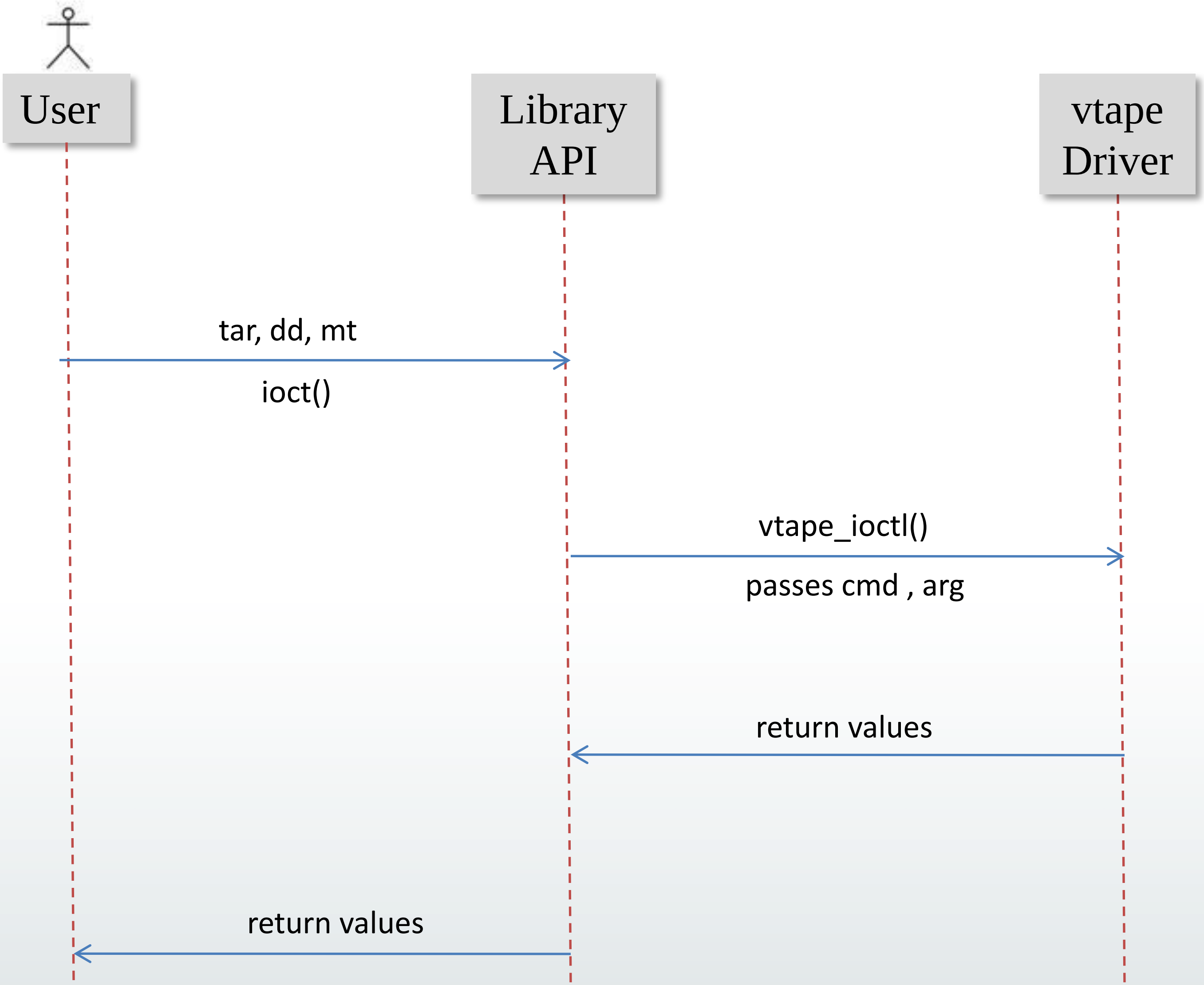
Write Sequence Diagram



Read Sequence Diagram



IOCTL Sequence Diagram



AGENDA

- ✓ INTRODUCTION
- ✓ SYSTEM REQUIREMENT AND SPECIFICATION
- ✓ SYSTEM DESIGN
- SYSTEM IMPLEMENTATION
- SOFTWARE TESTING
- CONCLUSION & FUTURE ENHANCEMENTS

SYSTEM IMPLEMENTATION

Goals

- Writing a character device driver
- Make use of Layered Device Interface
- Writing a user land application for creating a virtual tape media
- Writing user land utilities to perform operations over the tape

Writing Hardware Configuration File

- vtape.conf

- property=value

```
Ex.:- name="vtape" parent="pseudo" instance=1
      vtape_targ="/dev/lofi/1";
```

Implementation of the Device Driver

Structures implemented in the Driver

modlinkage structure

modldrv structure

dev_ops structure

cb_ops structure

drivers

```
static struct modlinkage modlinkage = {
    MODREV_1,
    &modldrv,
    NULL
};
```

```
static struct modldrv modldrv = {
    &mod_driverops,
    "Virtual Tape driver",
    &vtape_dev_ops
};
```

```
static struct dev_ops vtape_dev_ops = {
    DEVO_REV,                       /* devo_rev, */
    0,                              /* refcnt */
    vtape_info,                     /* getinfo */
    nulldev,                        /* identify */
    nulldev,                        /* probe */
    vtape_attach,                   /* attach */
    vtape_detach,                   /* detach */
    nodev,                          /* reset */
    &vtape_cb_ops,                  /* cb_ops */
    NULL,                           /* bus_ops */
    NULL                            /* power */
};
```

```
static struct cb_ops vtape_cb_ops = {
    vtape_open,                     /* open */
    vtape_close,                    /* close */
    nodev,                          /* strategy */
    nodev,                          /* print */
    nodev,                          /* dump */
    vtape_read,                     /* read */
    vtape_write,                    /* write */
    vtape_ioctl,                    /* ioctl */
    nodev,                          /* devmap */
    nodev,                          /* mmap */
    nodev,                          /* segmap */
    nochpoll,                       /* poll */
    ddi_prop_op,                    /* prop_op */
    NULL,                           /* streamtab */
    D_NEW | D_MP,                   /* cb_flag */
    CB_REV,                         /* cb_rev */
    nodev,                          /* aread */
    nodev                           /* awrite */
};
```

vtape_state: soft-state structure

```
typedef struct vtape_state {
            ldi_handle_t                        lh;
            ldi_ident_t                         li;
            dev_info_t                          *dip;
            minor_t                             minor;
            int                                 flags;
            kmutex_t                            lock;
            int                                 write_oft;
            int                                 read_oft;
            int                                 entry_flag;
            int                                 rec_size;
            int                                 eom_oft;
            offset_t                            oft;
            int                                 fileno;
            int                                 readfileno;
            int                                 fileflag;
            file_table_struct                   files[MAX_NO_OF_FILES];
    }vtape_state_t;
```

Entry Points

Loadable Module Entry Points

- **_init()**
- **_info()**
- **_fini()**

Autoconfiguration Entry Points

- **vtape_attach()**: The kernel calls the driver's attach() entry point to attach each instance of a device that is bound to the driver.
  - Allocating a soft-state structure for the device instance (ddi_soft_state_zalloc())
  - Initializing per-instance mutexes (mutex_init())
  - Creating minor device nodes for the device instance (ddi_create_minor_node())
  - Reporting that the device instance has attached (DDI_SUCCESS)

- **vtape_detach()**
  - The detach() entry point is called to detach an instance of a device that is bound to the driver
  - It frees the resources that were allocated in attach() entry point

- **vtape_info()**

  Gets the driver information. DDI_INFO_DEVT2DEVINFO / DDI_INFO_DEVT2INSTANCE can be passed as commands and getinfo() returns the dev_info_t pointer / the instance number associated with the device info pointer respectively.

Device Access Entry Points

- **vtape_open()**
- **vtape_close()**

I/O Request Handling

- **vtape_write()**

The vtape_write(9F) function takes the data written to the vtape device in the vtape_write() entry point and writes that data to the vtape_targ device. The ldi_write(9F) function uses the layered driver handle for the vtape_targ device to write the data to the vtape_targ device.

- **vtape_read()**

The vtape_read function reads data from the vtape device in the vtape_read() entry point and writes that data to the console. The ldi_read9F) function uses the layered driver handle for the vtape_targ device to write the data to the vtape_targ device. Succesive reads will retrieve successive backup files on the media. This is done by updating the uio_offset to current read position.

- # vtape_ioctl()

MTIOCLTOP- Magnetic Tape operation

```
struct  mtlop    {
                                        short        mt_op;
                                        short        pad[3];
                                        int64_t       mt_count;
};
```

1. MTFSF-	Forward space over Filemark
2. MTBSF-	Backward space over FIlemark
3. MTFSR-	Forward space over Record
4. MTBSR-	Forward space over Record
5. MTREW-	Rewind
6. MTERASE-	Erase
7. MTTELL-	Tell current position
8. MTSEEK-	Seek to specified position
9. MTWEOF-	Write EOF
10. MTEOM-	Write EOM

MTIOCGET- Status of device

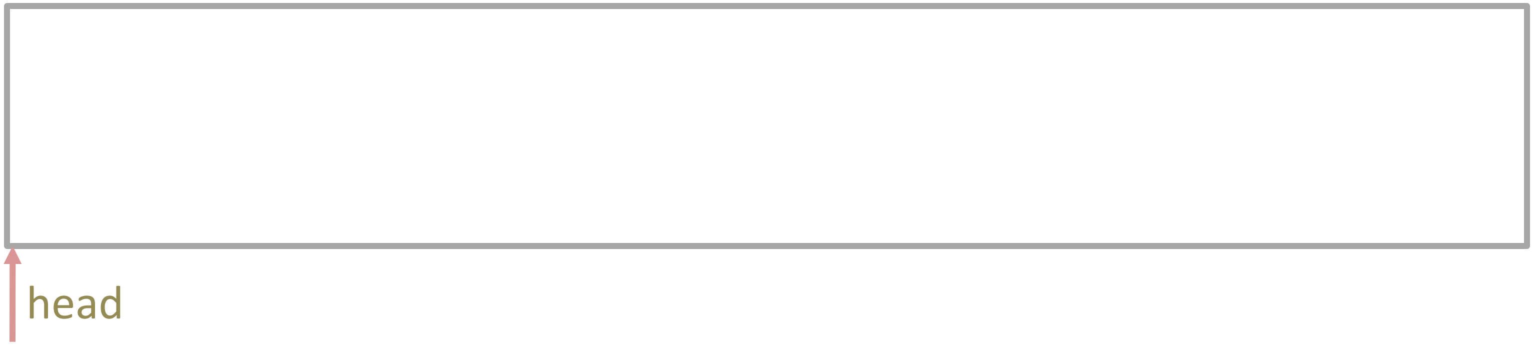
```
struct  mtget {
         short      mt_type;             /* type of magnetic tape device */
/* The following two registers are device dependent  */
        short      mt_dsreg;            /* “drive status” register */
        short      mt_erreg;            /* “error” register */
/* Optional error info */
        daddr_t   mt_resid;             /* residual count */
        daddr_t   mt_fileno;           /* file number of current position */
        daddr_t   mt_blkno;            /* block number of current position */
        ushort_t  mt_flags;
        short        mt_bf;             /* optimum blocking factor */
};
```

MTIOCGETDRIVETYPE- Configuration of device

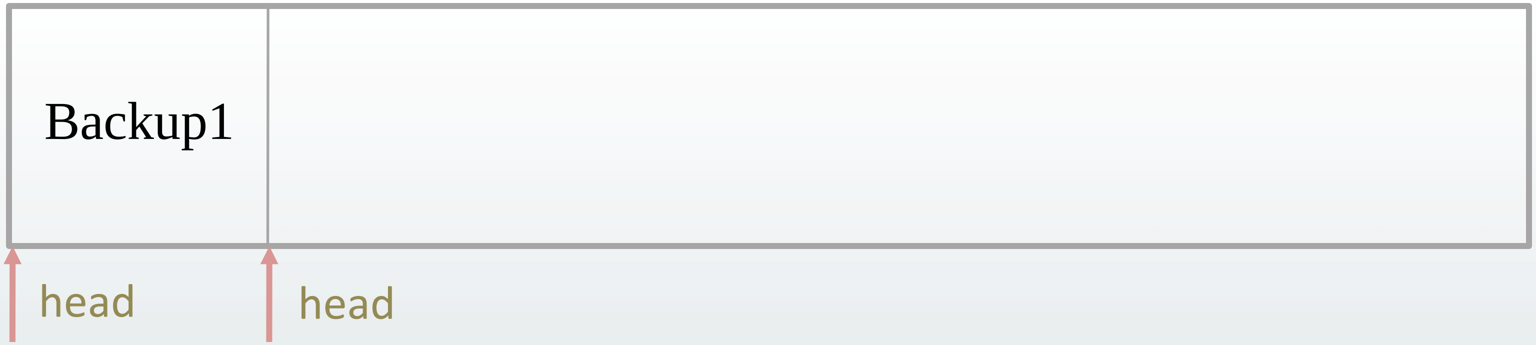
```
struct mtdrivetype {
        char        name[64];
        char        vid[25];                       /* Vendor id and product id */
        char        type;                          /* Drive type for driver */
        int         bsize;                         /* Block size */
        int         options;                       /* Drive options */
        int         max_rretries;                  /* Max read retries */
        int         max_wretries;                  /* Max write retries */
        uchar_t     densities[MT_NDENSITIES];      /* Density Codes, low->hi */
        uchar_t     default_density;               /* Default density chosen */
        uchar_t     speeds[MT_NSPEEDS];            /* Speed codes, low->hi */
        ushort_t    non_motion_timeout;            /* Seconds for non-motion */
        ushort_t    io_timeout;                    /* Seconds for data to from tape */
        ushort_t    rewind_timeout;                /* Seconds to rewind */
        ushort_t    space_timeout;                 /* Seconds to space anywhere */
        ushort_t    load_timeout;                  /* Seconds to load tape and ready */
        ushort_t    unload_timeout;                /* Seconds to unload */
        ushort_t    erase_timeout;                 /* Seconds to do long erase */
};
```

Representation of Tape Operations

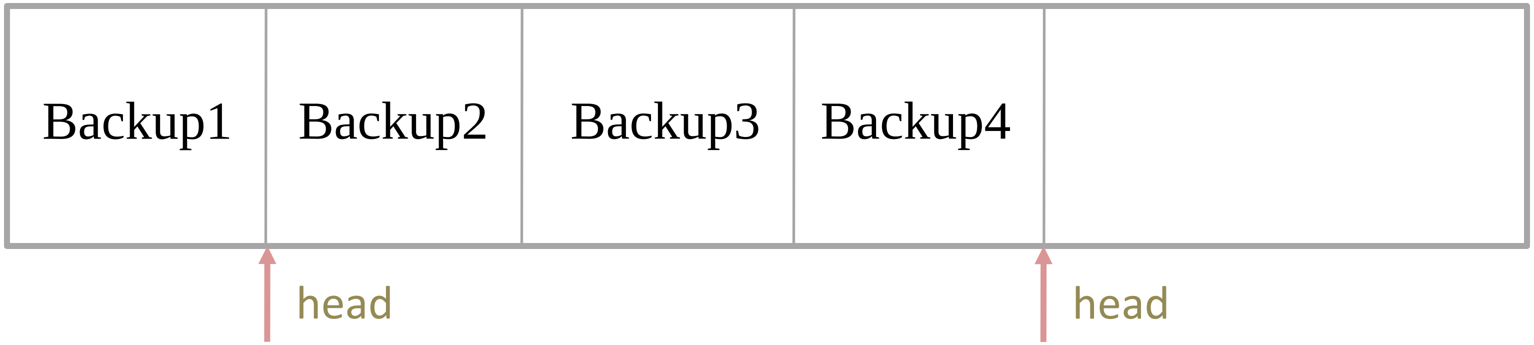
Initial Tape Representation



tar –cvf /dev/rmt/0node backup1



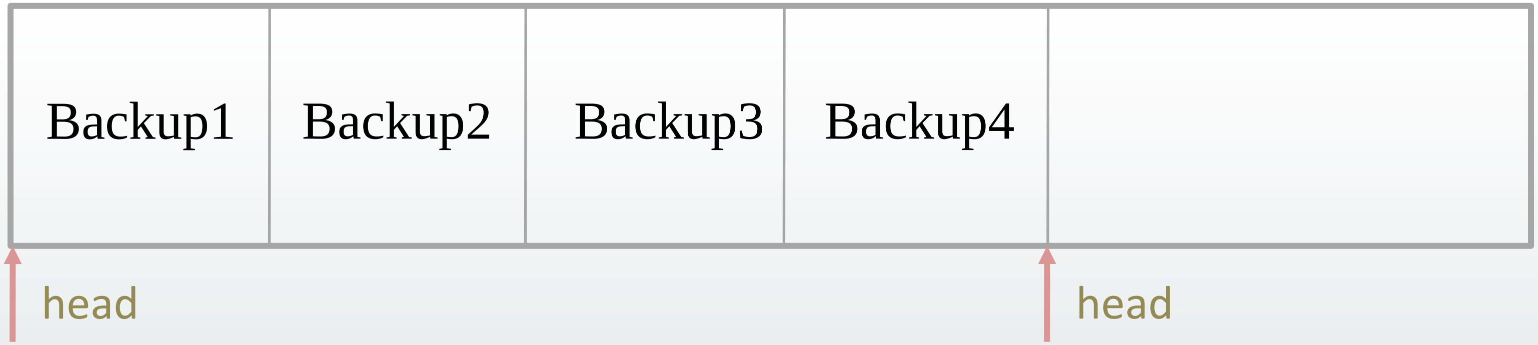
Similarly after backing up 3 more files:



Backup1 Backup2 Backup3 Backup4

head head

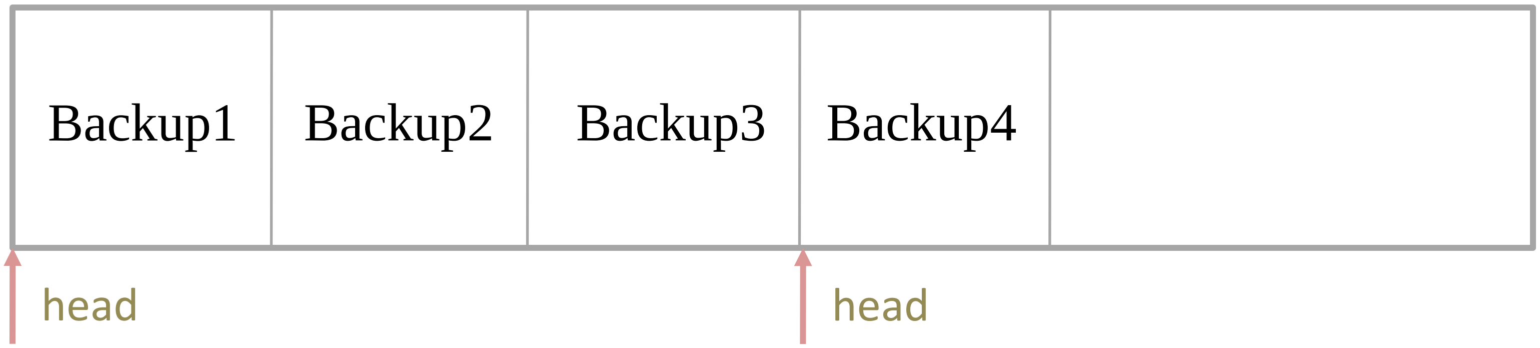
mt –f /dev/rmt/0node rewind



Backup1 Backup2 Backup3 Backup4

head head

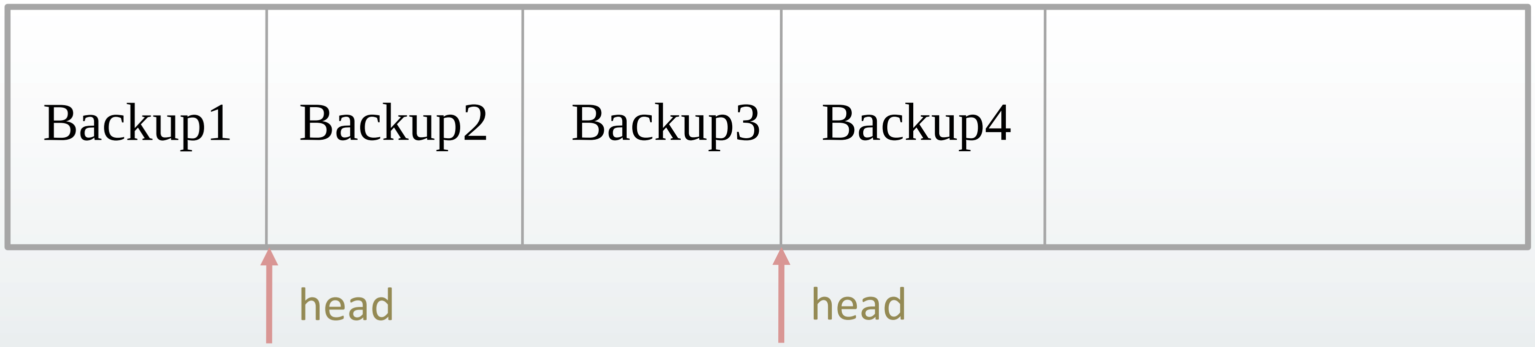
```
mt –f /dev/rmt/0node fsf 3
```



Backup1 Backup2 Backup3 Backup4

head head

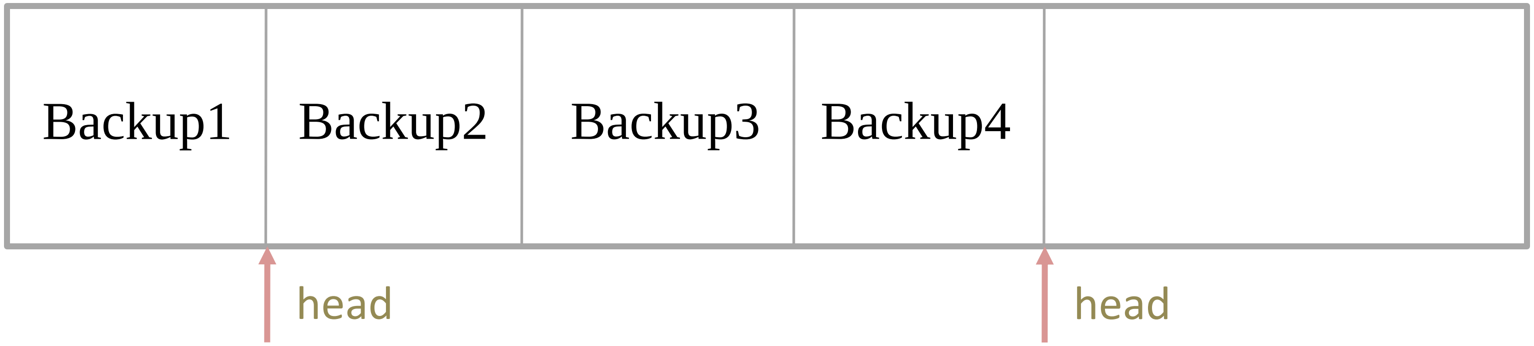
```
mt –f /dev/rmt/0node bsf 2
```



Backup1 Backup2 Backup3 Backup4

head head

```
mt –f /dev/rmt/0node seek 84
```

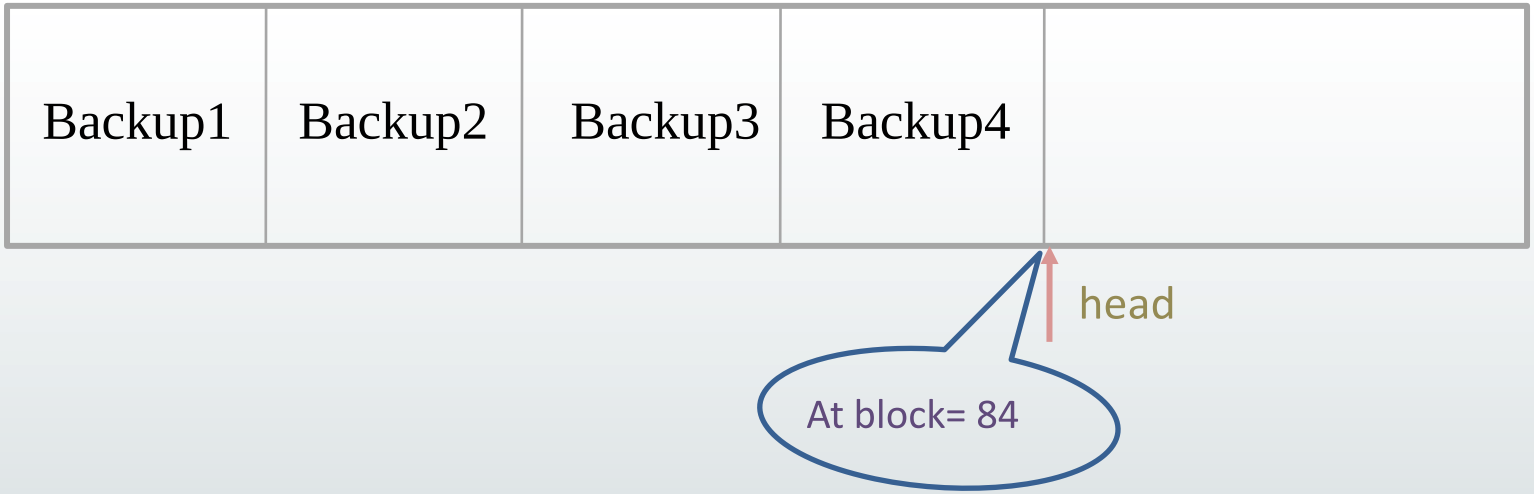


Backup1 Backup2 Backup3 Backup4

head

head

```
mt –f /dev/rmt/0node tell
```

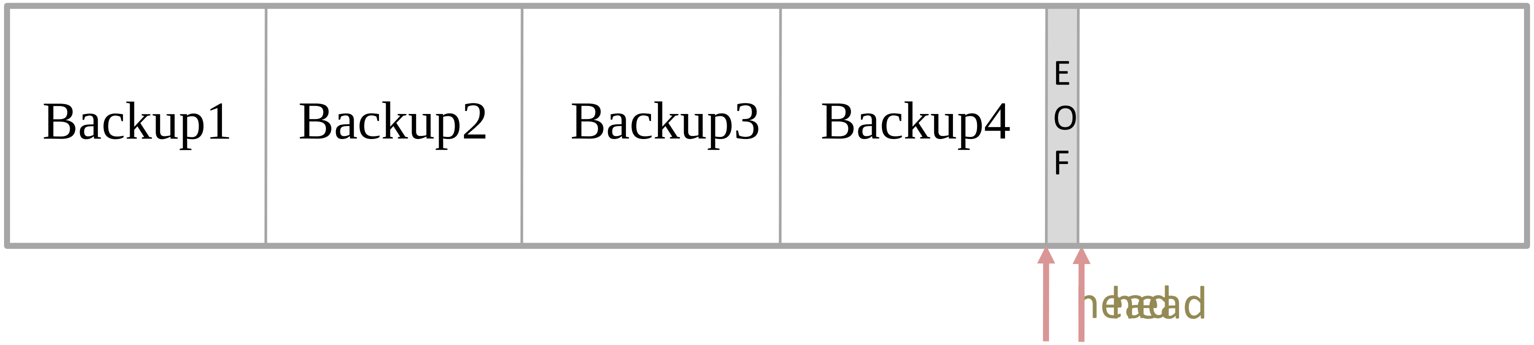


Backup1 Backup2 Backup3 Backup4

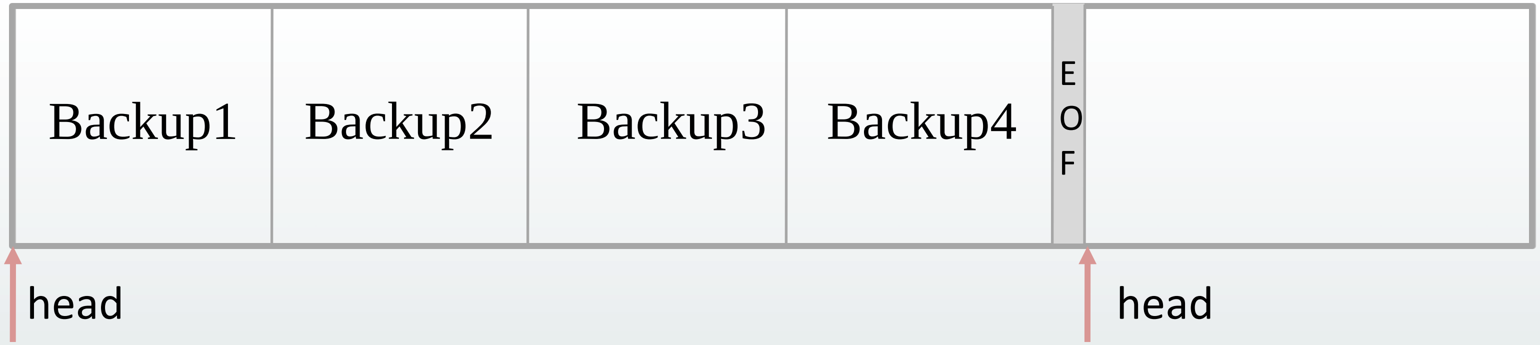
head

At block= 84

mt –f /dev/rmt/0node weof 1



mt –f /dev/rmt/0node erase



AGENDA

- ✓ INTRODUCTION
- ✓ SYSTEM REQUIREMENT AND SPECIFICATION
- ✓ SYSTEM DESIGN
- ✓ SYSTEM IMPLEMENTATION
- SOFTWARE TESTING
- CONCLUSION & FUTURE ENHANCEMENTS

SOFTWARE TESTING

Drivers should be rigorously tested in the following areas:

- Configuration Testing
- Functionality Testing
- Error Handling
- Testing Loading and Unloading
- DDI/DKI Compliance Testing
- Installation and Packaging Testing

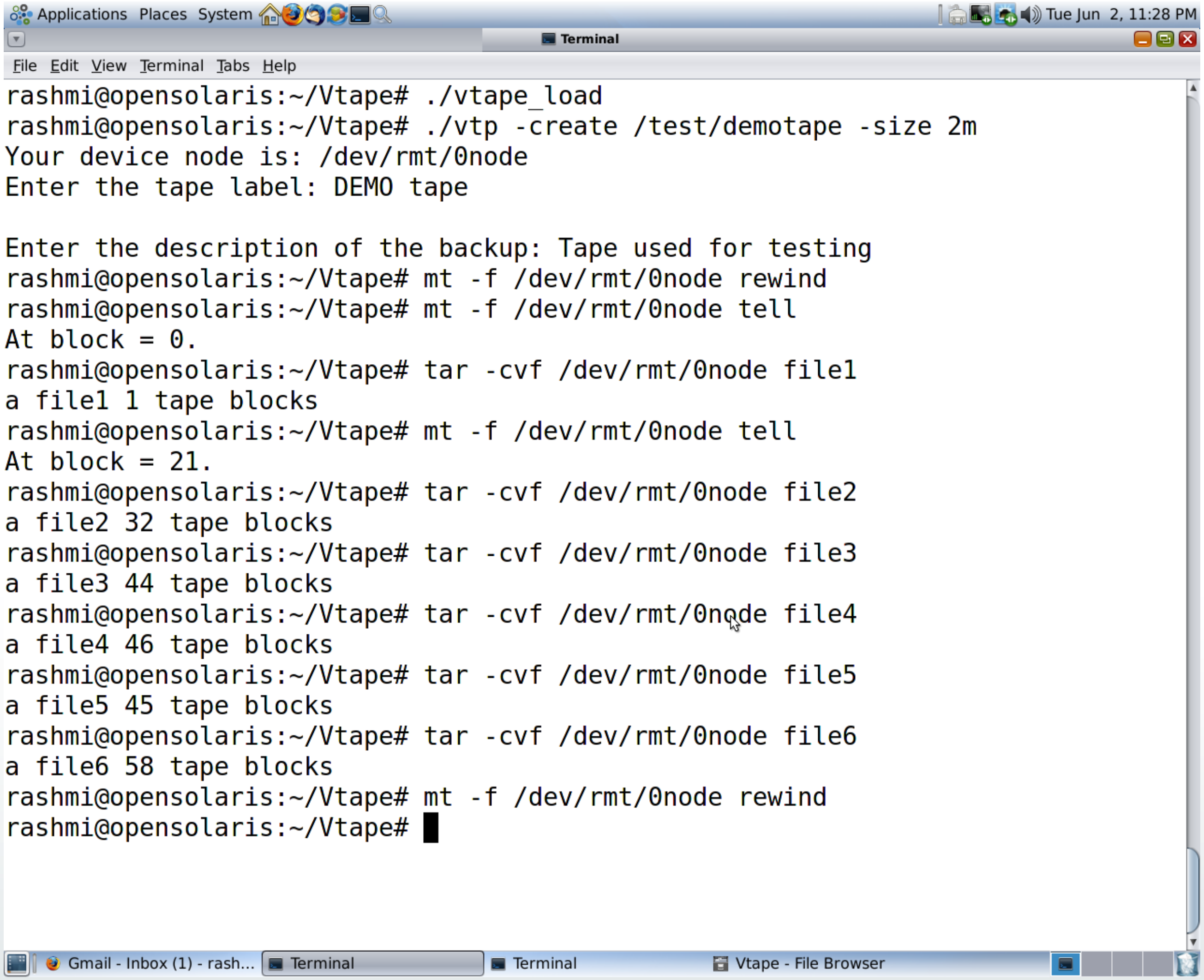
Tape Drivers

Tape drivers should be tested by performing several operations shown below:

- **cpio() & tar()-** archive and restore
- **mt()-** I/O controls that are specific to tape drivers.
- **dd()-** to write an entire disk partition to tape. Next, the data is read back, and the data is written to another partition of the same size. Then the two copies are compared.

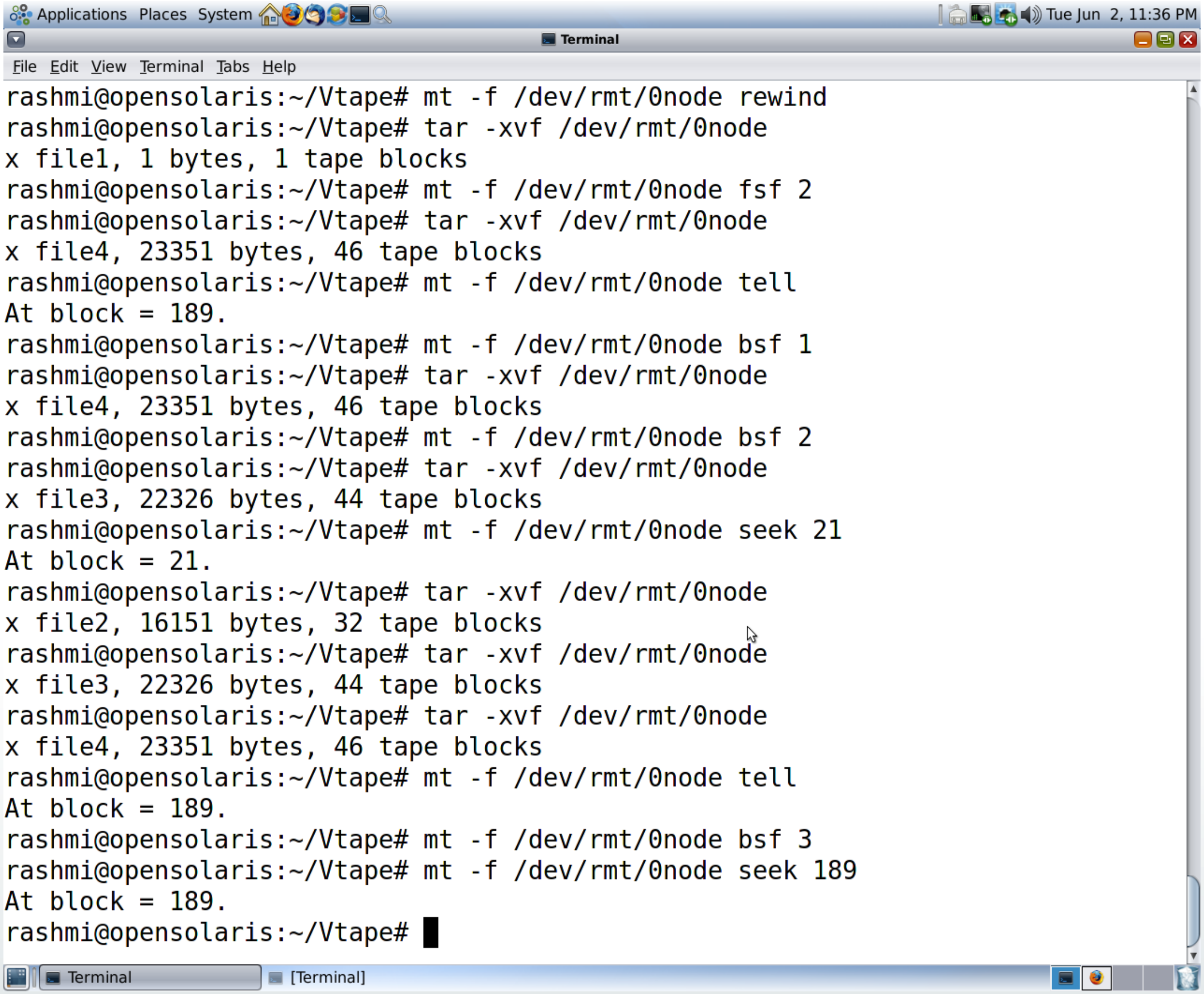
The above described test cases can be clearly seen in the screenshots which follow.

SCREENSHOTS

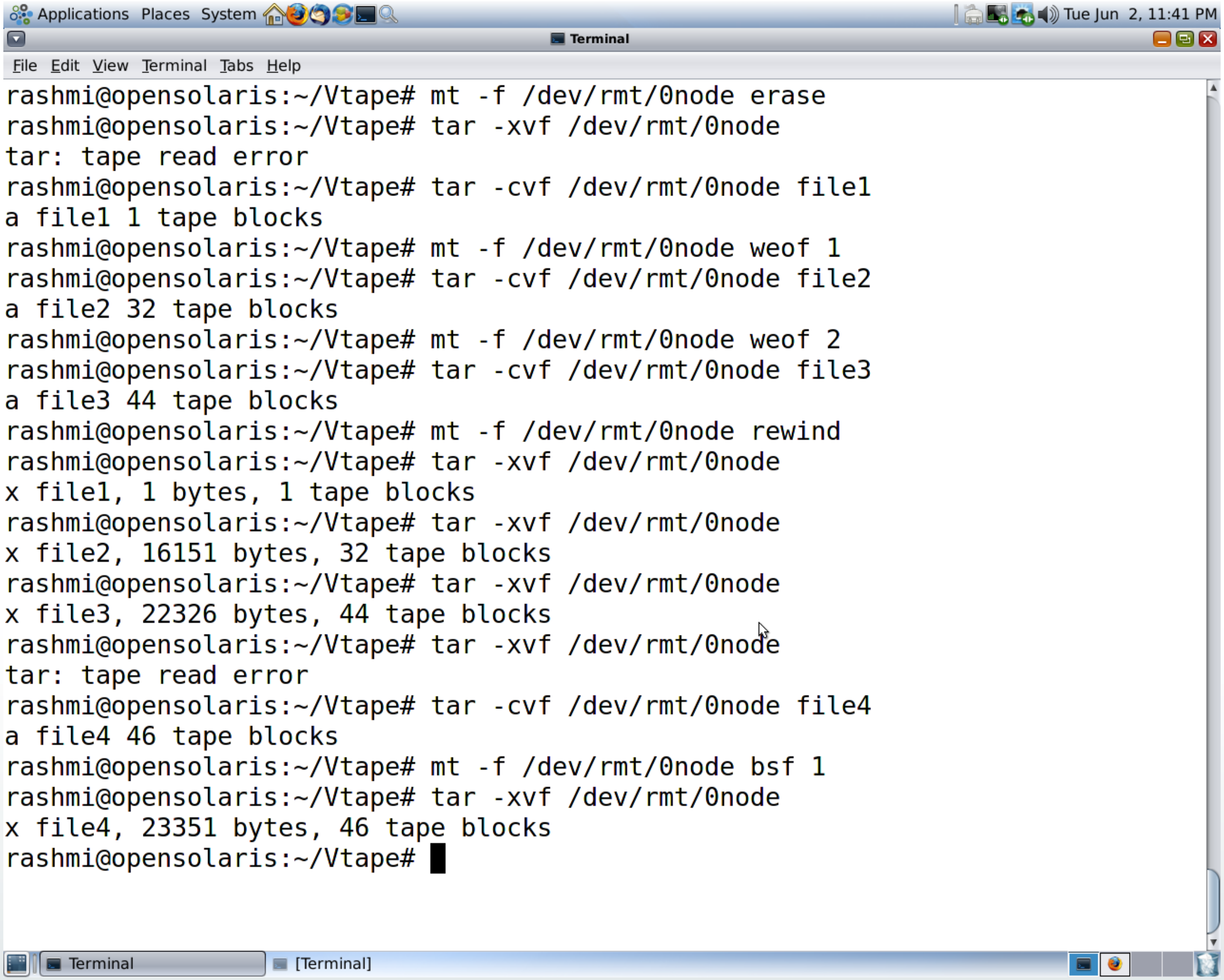


Applications Places System
Tue Jun 2, 11:28 PM
Terminal
File Edit View Terminal Tabs Help
Gmail - Inbox (1) - rash...
Terminal
Terminal
Vtape - File Browser

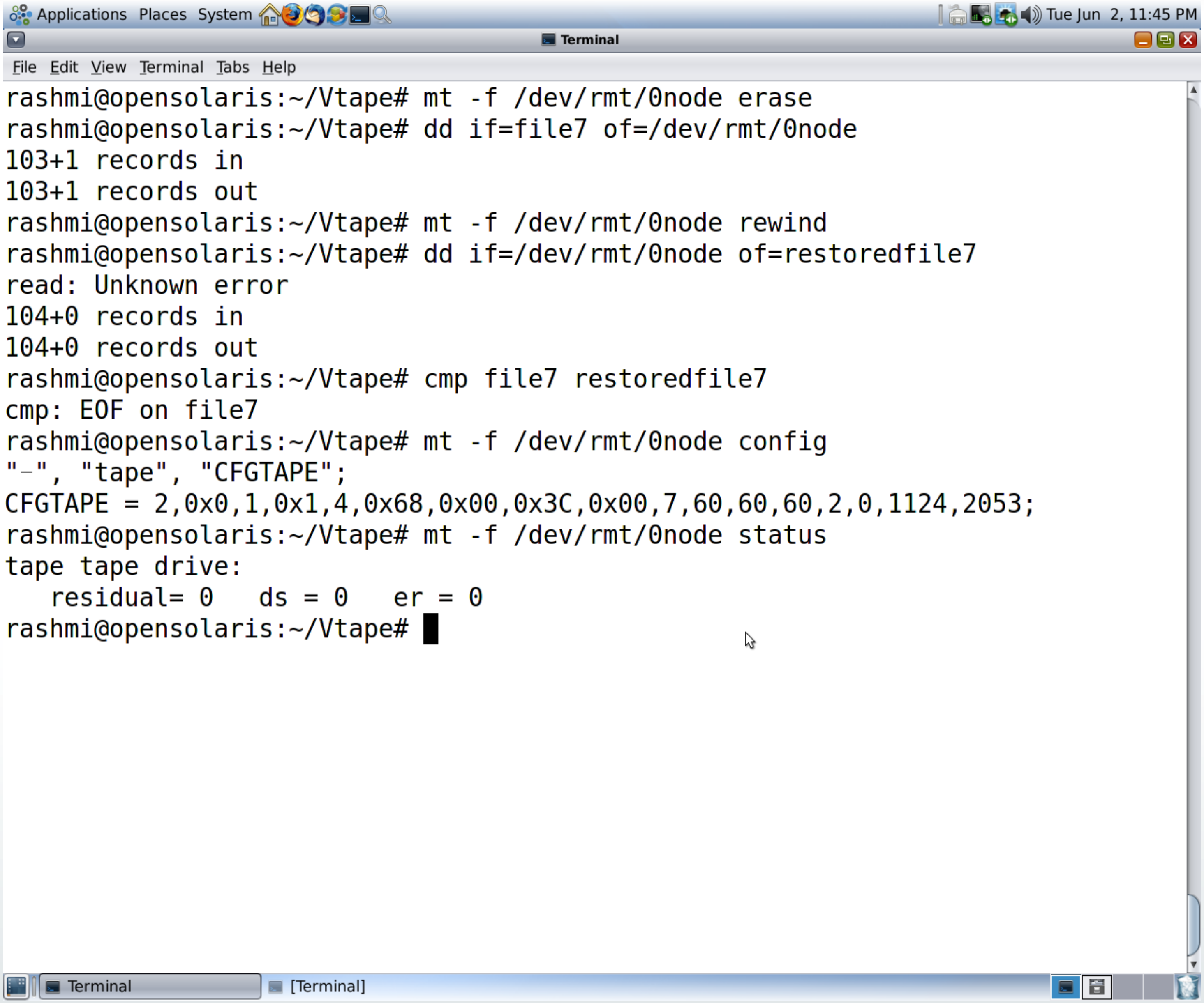
```
rashmi@opensolaris:~/Vtape# ./vtape_load
rashmi@opensolaris:~/Vtape# ./vtp -create /test/demotape -size 2m
Your device node is: /dev/rmt/0node
Enter the tape label: DEMO tape

Enter the description of the backup: Tape used for testing
rashmi@opensolaris:~/Vtape# mt -f /dev/rmt/0node rewind
rashmi@opensolaris:~/Vtape# mt -f /dev/rmt/0node tell
At block = 0.
rashmi@opensolaris:~/Vtape# tar -cvf /dev/rmt/0node file1
a file1 1 tape blocks
rashmi@opensolaris:~/Vtape# mt -f /dev/rmt/0node tell
At block = 21.
rashmi@opensolaris:~/Vtape# tar -cvf /dev/rmt/0node file2
a file2 32 tape blocks
rashmi@opensolaris:~/Vtape# tar -cvf /dev/rmt/0node file3
a file3 44 tape blocks
rashmi@opensolaris:~/Vtape# tar -cvf /dev/rmt/0node file4
a file4 46 tape blocks
rashmi@opensolaris:~/Vtape# tar -cvf /dev/rmt/0node file5
a file5 45 tape blocks
rashmi@opensolaris:~/Vtape# tar -cvf /dev/rmt/0node file6
a file6 58 tape blocks
rashmi@opensolaris:~/Vtape# mt -f /dev/rmt/0node rewind
rashmi@opensolaris:~/Vtape#
```



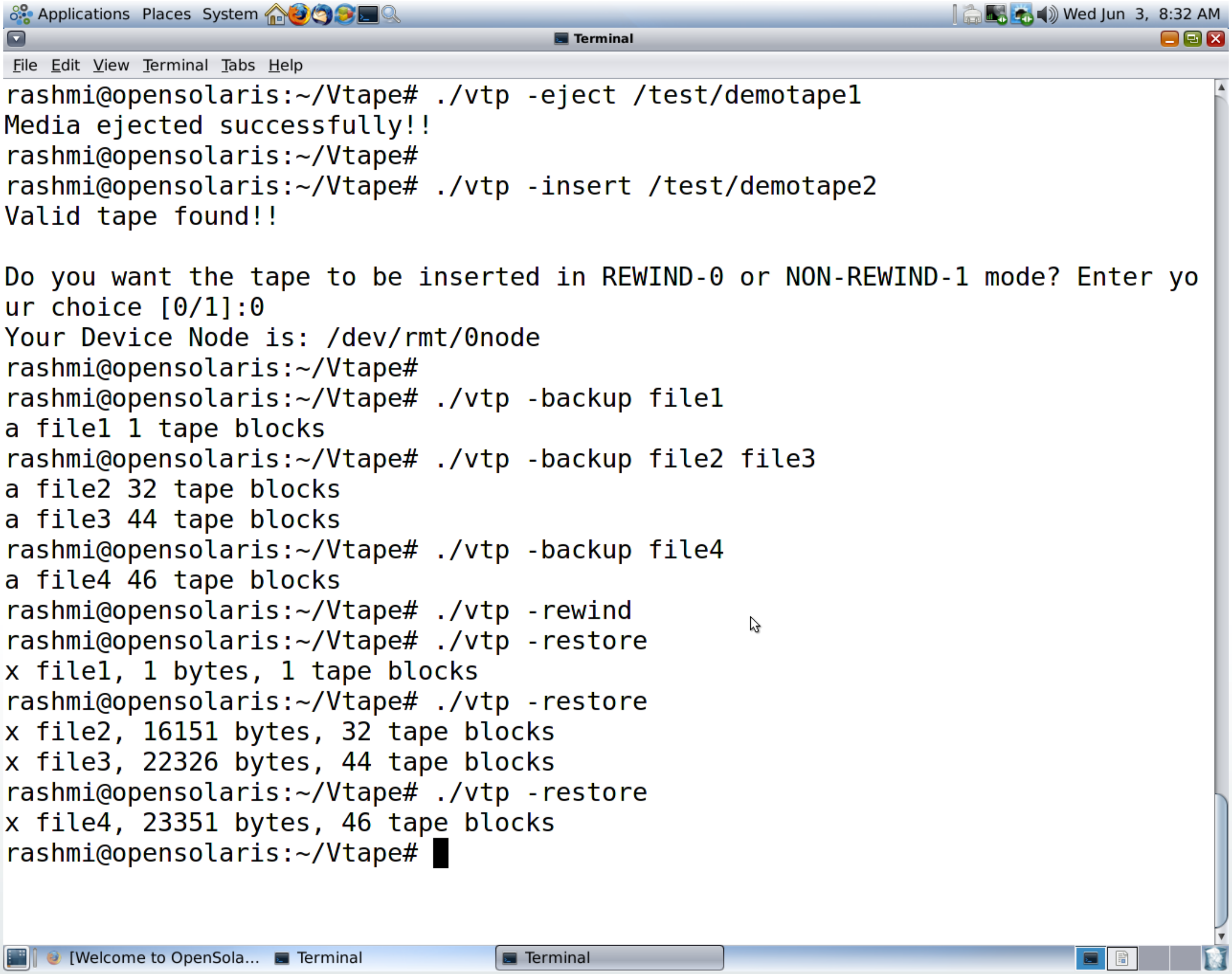
```
rashmi@opensolaris:~/Vtape# mt -f /dev/rmt/0node rewind
rashmi@opensolaris:~/Vtape# tar -xvf /dev/rmt/0node
x file1, 1 bytes, 1 tape blocks
rashmi@opensolaris:~/Vtape# mt -f /dev/rmt/0node fsf 2
rashmi@opensolaris:~/Vtape# tar -xvf /dev/rmt/0node
x file4, 23351 bytes, 46 tape blocks
rashmi@opensolaris:~/Vtape# mt -f /dev/rmt/0node tell
At block = 189.
rashmi@opensolaris:~/Vtape# mt -f /dev/rmt/0node bsf 1
rashmi@opensolaris:~/Vtape# tar -xvf /dev/rmt/0node
x file4, 23351 bytes, 46 tape blocks
rashmi@opensolaris:~/Vtape# mt -f /dev/rmt/0node bsf 2
rashmi@opensolaris:~/Vtape# tar -xvf /dev/rmt/0node
x file3, 22326 bytes, 44 tape blocks
rashmi@opensolaris:~/Vtape# mt -f /dev/rmt/0node seek 21
At block = 21.
rashmi@opensolaris:~/Vtape# tar -xvf /dev/rmt/0node
x file2, 16151 bytes, 32 tape blocks
rashmi@opensolaris:~/Vtape# tar -xvf /dev/rmt/0node
x file3, 22326 bytes, 44 tape blocks
rashmi@opensolaris:~/Vtape# tar -xvf /dev/rmt/0node
x file4, 23351 bytes, 46 tape blocks
rashmi@opensolaris:~/Vtape# mt -f /dev/rmt/0node tell
At block = 189.
rashmi@opensolaris:~/Vtape# mt -f /dev/rmt/0node bsf 3
rashmi@opensolaris:~/Vtape# mt -f /dev/rmt/0node seek 189
At block = 189.
rashmi@opensolaris:~/Vtape# 
```



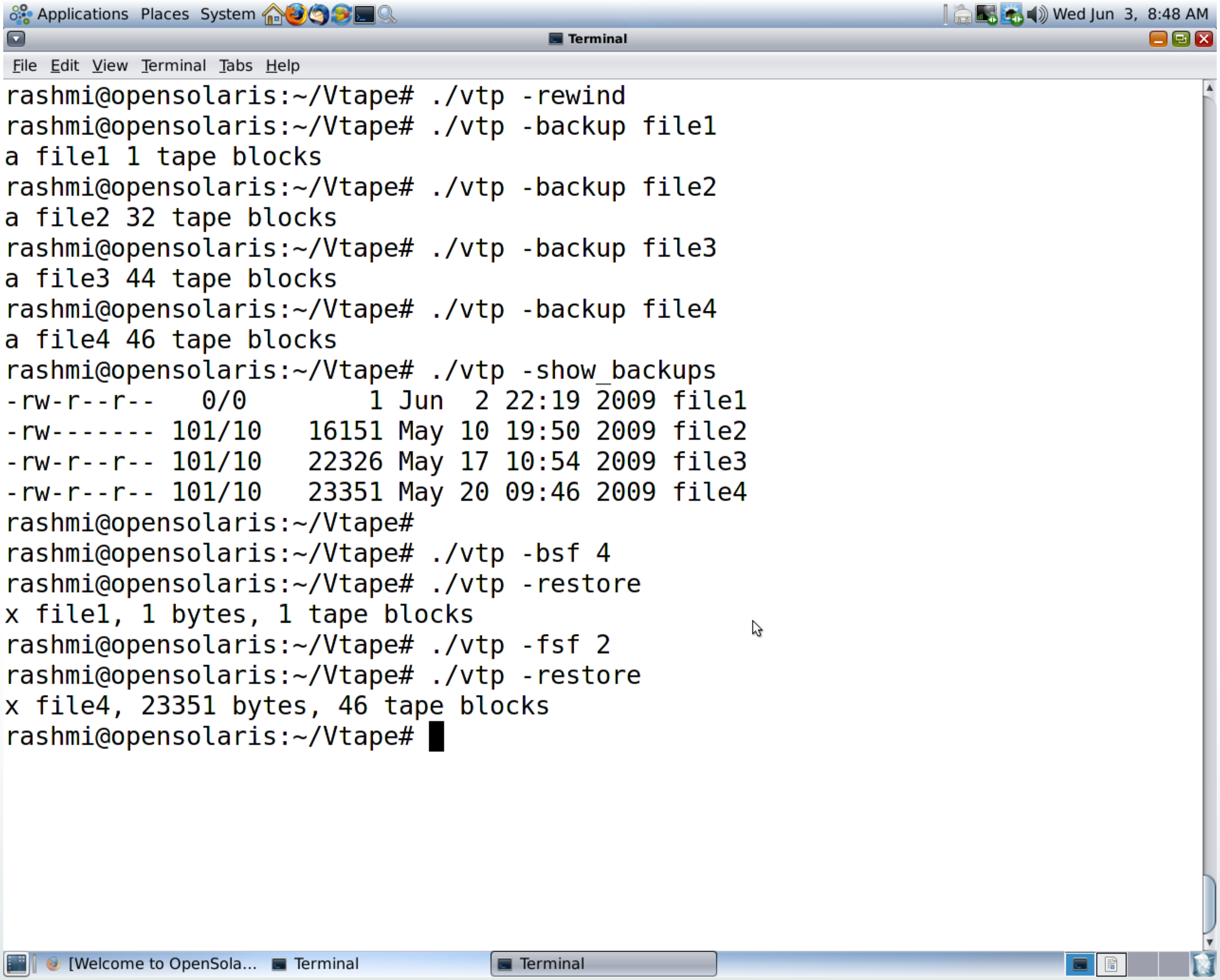
```
rashmi@opensolaris:~/Vtape# mt -f /dev/rmt/0node erase
rashmi@opensolaris:~/Vtape# tar -xvf /dev/rmt/0node
tar: tape read error
rashmi@opensolaris:~/Vtape# tar -cvf /dev/rmt/0node file1
a file1 1 tape blocks
rashmi@opensolaris:~/Vtape# mt -f /dev/rmt/0node weof 1
rashmi@opensolaris:~/Vtape# tar -cvf /dev/rmt/0node file2
a file2 32 tape blocks
rashmi@opensolaris:~/Vtape# mt -f /dev/rmt/0node weof 2
rashmi@opensolaris:~/Vtape# tar -cvf /dev/rmt/0node file3
a file3 44 tape blocks
rashmi@opensolaris:~/Vtape# mt -f /dev/rmt/0node rewind
rashmi@opensolaris:~/Vtape# tar -xvf /dev/rmt/0node
x file1, 1 bytes, 1 tape blocks
rashmi@opensolaris:~/Vtape# tar -xvf /dev/rmt/0node
x file2, 16151 bytes, 32 tape blocks
rashmi@opensolaris:~/Vtape# tar -xvf /dev/rmt/0node
x file3, 22326 bytes, 44 tape blocks
rashmi@opensolaris:~/Vtape# tar -xvf /dev/rmt/0node
tar: tape read error
rashmi@opensolaris:~/Vtape# tar -cvf /dev/rmt/0node file4
a file4 46 tape blocks
rashmi@opensolaris:~/Vtape# mt -f /dev/rmt/0node bsf 1
rashmi@opensolaris:~/Vtape# tar -xvf /dev/rmt/0node
x file4, 23351 bytes, 46 tape blocks
rashmi@opensolaris:~/Vtape#
```



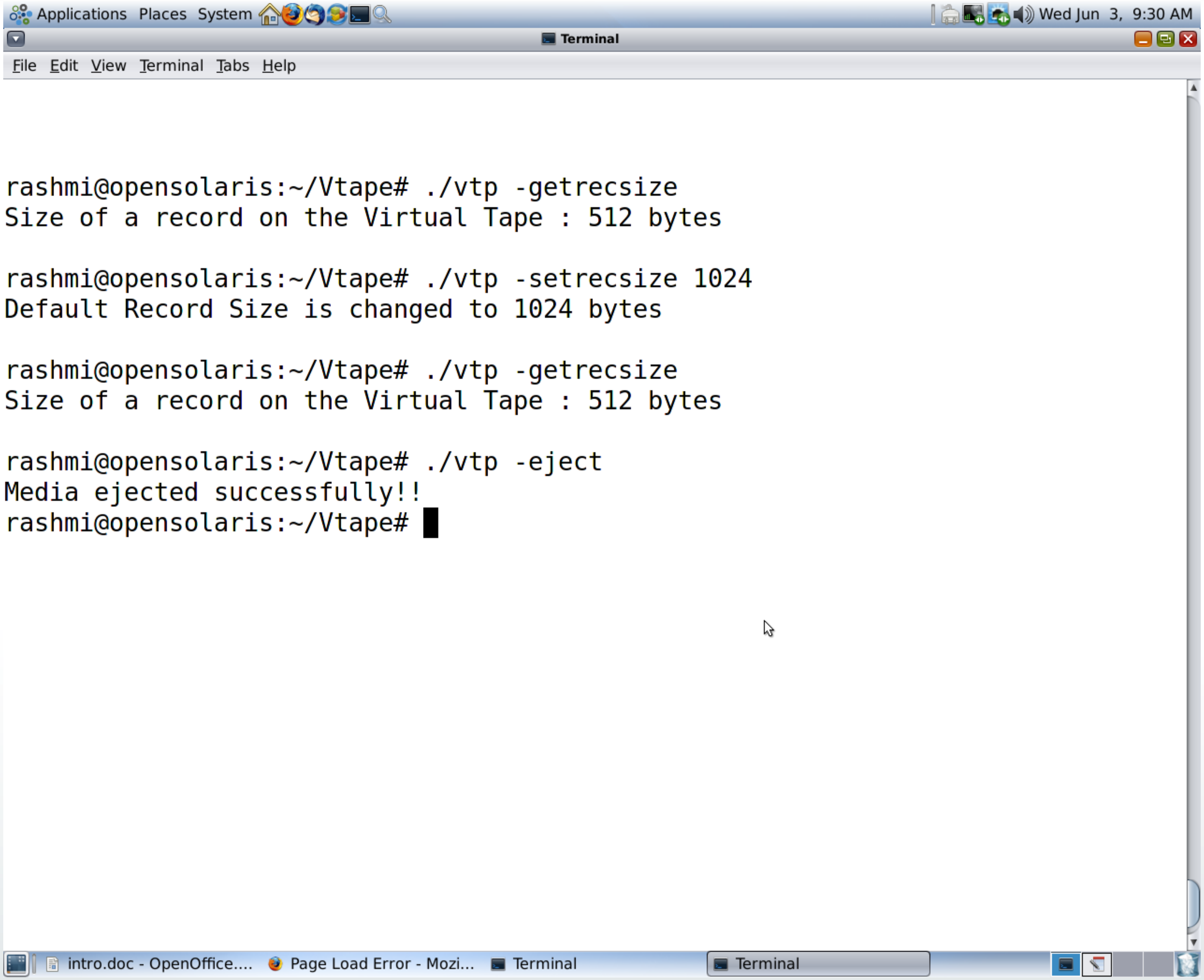
```
rashmi@opensolaris:~/Vtape# mt -f /dev/rmt/0node erase
rashmi@opensolaris:~/Vtape# dd if=file7 of=/dev/rmt/0node
103+1 records in
103+1 records out
rashmi@opensolaris:~/Vtape# mt -f /dev/rmt/0node rewind
rashmi@opensolaris:~/Vtape# dd if=/dev/rmt/0node of=restoredfile7
read: Unknown error
104+0 records in
104+0 records out
rashmi@opensolaris:~/Vtape# cmp file7 restoredfile7
cmp: EOF on file7
rashmi@opensolaris:~/Vtape# mt -f /dev/rmt/0node config
"-", "tape", "CFGTAPE";
CFGTAPE = 2,0x0,1,0x1,4,0x68,0x00,0x3C,0x00,7,60,60,60,2,0,1124,2053;
rashmi@opensolaris:~/Vtape# mt -f /dev/rmt/0node status
tape tape drive:
   residual= 0   ds = 0   er = 0
rashmi@opensolaris:~/Vtape#
```



```
rashmi@opensolaris:~/Vtape# ./vtp -eject /test/demotape1
Media ejected successfully!!
rashmi@opensolaris:~/Vtape#
rashmi@opensolaris:~/Vtape# ./vtp -insert /test/demotape2
Valid tape found!!

Do you want the tape to be inserted in REWIND-0 or NON-REWIND-1 mode? Enter yo
ur choice [0/1]:0
Your Device Node is: /dev/rmt/0node
rashmi@opensolaris:~/Vtape#
rashmi@opensolaris:~/Vtape# ./vtp -backup file1
a file1 1 tape blocks
rashmi@opensolaris:~/Vtape# ./vtp -backup file2 file3
a file2 32 tape blocks
a file3 44 tape blocks
rashmi@opensolaris:~/Vtape# ./vtp -backup file4
a file4 46 tape blocks
rashmi@opensolaris:~/Vtape# ./vtp -rewind
rashmi@opensolaris:~/Vtape# ./vtp -restore
x file1, 1 bytes, 1 tape blocks
rashmi@opensolaris:~/Vtape# ./vtp -restore
x file2, 16151 bytes, 32 tape blocks
x file3, 22326 bytes, 44 tape blocks
rashmi@opensolaris:~/Vtape# ./vtp -restore
x file4, 23351 bytes, 46 tape blocks
rashmi@opensolaris:~/Vtape#
```



```
rashmi@opensolaris:~/Vtape# ./vtp -rewind
rashmi@opensolaris:~/Vtape# ./vtp -backup file1
a file1 1 tape blocks
rashmi@opensolaris:~/Vtape# ./vtp -backup file2
a file2 32 tape blocks
rashmi@opensolaris:~/Vtape# ./vtp -backup file3
a file3 44 tape blocks
rashmi@opensolaris:~/Vtape# ./vtp -backup file4
a file4 46 tape blocks
rashmi@opensolaris:~/Vtape# ./vtp -show_backups
-rw-r--r--   0/0        1 Jun  2 22:19 2009 file1
-rw------- 101/10   16151 May 10 19:50 2009 file2
-rw-r--r-- 101/10   22326 May 17 10:54 2009 file3
-rw-r--r-- 101/10   23351 May 20 09:46 2009 file4
rashmi@opensolaris:~/Vtape#
rashmi@opensolaris:~/Vtape# ./vtp -bsf 4
rashmi@opensolaris:~/Vtape# ./vtp -restore
x file1, 1 bytes, 1 tape blocks
rashmi@opensolaris:~/Vtape# ./vtp -fsf 2
rashmi@opensolaris:~/Vtape# ./vtp -restore
x file4, 23351 bytes, 46 tape blocks
rashmi@opensolaris:~/Vtape# 
```



```
rashmi@opensolaris:~/Vtape# ./vtp -getrecsize
Size of a record on the Virtual Tape : 512 bytes

rashmi@opensolaris:~/Vtape# ./vtp -setrecsize 1024
Default Record Size is changed to 1024 bytes

rashmi@opensolaris:~/Vtape# ./vtp -getrecsize
Size of a record on the Virtual Tape : 512 bytes

rashmi@opensolaris:~/Vtape# ./vtp -eject
Media ejected successfully!!
rashmi@opensolaris:~/Vtape#
```

AGENDA

- ✓ INTRODUCTION
- ✓ SYSTEM REQUIREMENT AND SPECIFICATION
- ✓ SYSTEM DESIGN
- ✓ SYSTEM IMPLEMENTATION
- ✓ SOFTWARE TESTING
- CONCLUSION & FUTURE ENHANCEMENTS

CONCLUSION

Tape is the most reliable way to backup data

- Therefore the use of virtual will go a long way in assisting the research and development being carried out in this domain especially during testing

- We are also very proud to be a part of the open source community and very happy to contribute to OpenSolaris.

- We also found Virtual Tape Device Driver a very interesting and learning experience. It helped us learn the nuances of kernel programming and integrating our modules with the different kernel modules giving us an invaluable insight into the Solaris Kernel Architecture

FUTURE ENHANCEMENTS

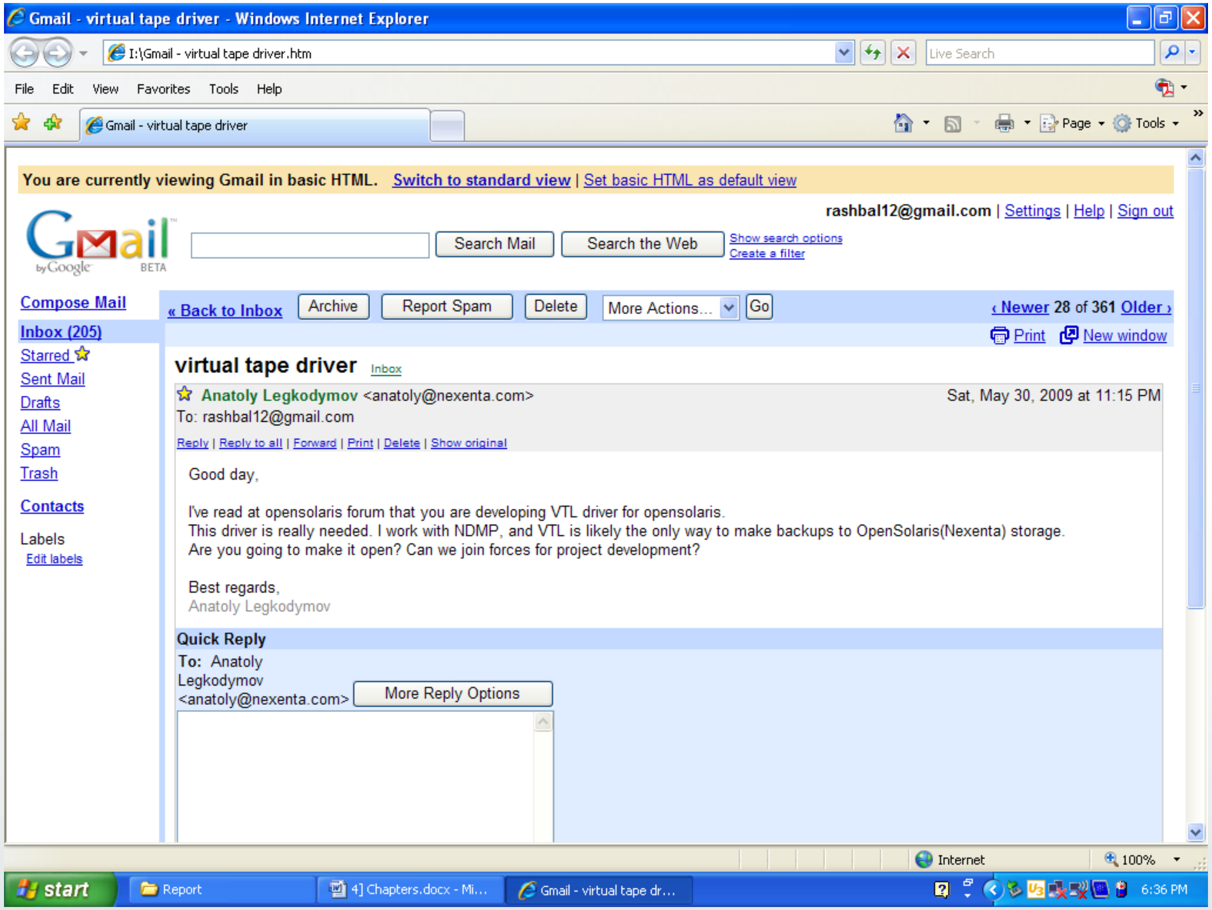
- To develop a GUI for our application to perform backup, restore and other ioctl-like operations on our virtual tape
- To emulate non-rewind version of tape (Multiple nodes for each instance)
- To write a single backup on multiple volumes of tapes (To have multiple instances of tape simultaneously)
- Emulation of our virtual tape on CD-ROM
- To integrate with NDMP protocol
- To make it compatible with both Little Endian and Big Endian Format

REFERENCES

- Writing Device Drivers- http://docs.sun.com/app/docs/doc/816-4854/preface-1?l=en&q=device+drivers&a=view
- Device Driver Tutorials- http://docs.sun.com/app/docs/doc/817-5789
- Open Solaris source code- http://www.opensolaris.org/os/downloads/
- Man pages- ioctl(9E), mt(9F), mtio(7I), lofiadm, mkfs, tar, cpio, dd, copyin(), copyout, etc
- Device Drivers- Hands on Lab- www.opensolaris.org/os/community/advocacy/events/techdays/beijing/Writing_Device_Drivers_For_Solaris.pdf-lab
- OpenGrok : Source Code Browser- http://src.opensolaris.org/source/
- Wikipedia
- Google Search

FEEDBACK

As the purpose of this project was to aid the Research and Development taking place in the field of data storage especially NDMP, this has already been satisfied during the course of our project. The mail below bears testimony to this fact.


Gmail - virtual tape driver - Windows Internet Explorer
I:\Gmail - virtual tape driver.htm
Live Search
File Edit View Favorites Tools Help
Gmail - virtual tape driver
Page Tools
You are currently viewing Gmail in basic HTML. Switch to standard view | Set basic HTML as default view
rashbal12@gmail.com | Settings | Help | Sign out
Gmail by Google BETA
Search Mail
Search the Web
Show search options
Create a filter
Compose Mail
Inbox (205)
Starred
Sent Mail
Drafts
All Mail
Spam
Trash
Contacts
Labels
Edit labels
« Back to Inbox
Archive
Report Spam
Delete
More Actions...
Go
‹ Newer 28 of 361 Older ›
Print
New window
Inbox
Reply | Reply to all | Forward | Print | Delete | Show original
Quick Reply
More Reply Options
Internet
100%
start
Report
4] Chapters.docx - Mi...
Gmail - virtual tape dr...
6:36 PM

virtual tape driver

Anatoly Legkodymov <anatoly@nexenta.com> Sat, May 30, 2009 at 11:15 PM
To: rashbal12@gmail.com

Good day,

I've read at opensolaris forum that you are developing VTL driver for opensolaris.
This driver is really needed. I work with NDMP, and VTL is likely the only way to make backups to OpenSolaris(Nexenta) storage.
Are you going to make it open? Can we join forces for project development?

Best regards,
Anatoly Legkodymov

To: Anatoly Legkodymov <anatoly@nexenta.com>

THANK YOU

ANY QUESTIONS?