

**A Preliminary Project Report**  
**on**  
**“Distributed Replicated Block Device (DRBD) Network Packet**  
**Tracing Module”**

Submitted to the  
Savitribai Phule Pune University  
In partial fulfillment for the award of the Degree of  
Bachelor of Engineering  
in  
Information Technology  
by

**Bharat Kothari - 43110**

**Animesh Landge - 43132**

**Shrijan Vats - 43258**

**Amod Dhopavkar - 43304**

Under the guidance of

**Prof. V. R. Jaiswal**

Sponsored by

**Veritas LLC, Pune**



Department Of Information Technology  
Pune Institute of Computer Technology College of Engineering  
Sr. No 27, Pune-Satara Road, Dhankawadi, Pune - 411 043

**2020-2021**

## SPONSORSHIP LETTER

Hi Prof. Radhika,

Based on the discussions with students groups by our interview panel, we have decided to select below groups. We will provide a official communication next week.

I can say this was a tough selection calls and hoping we will be able make all required efforts to ensure the students working with us. We will keep you in loop while we decide the project topics and always welcome any feedback from your side.

**Selected Groups:**

**Group 14**

Animesh Landge,  
Bharat Kothari,  
Amol Dhopavkar,  
Shrijan Vats

## ACKNOWLEDGEMENT

This project would not have been possible without the support of our friends and family. We would like to thank our guide from Veritas LLC, Mr. Swapnil Ujgare sir and our internal guide Prof. V. R. Jaiswal sir, and for their constant advice and support. Additionally, we also thank our external reviewers Prof. T. A. Rane sir and Prof. S. S. Mane sir whose constructive criticism and valuable inputs made us think in ways which would not have been possible.

We would like to thank BE Project coordinator Prof. R. V. Kulkarni ma'am for providing guidance throughout the way. In the end we would like to thank Dr. A. M. Bagade sir, the HOD of IT department, and the entire IT department for their constant support. We would also like to thank the principal, Dr. P. T. Kulkarni sir for providing us with this opportunity.

Bharat Kothari

43110

Animesh Landge

43132

Shrijan Vats

43258

Amod Dhopavkar

43304

## **ABSTRACT**

Distributed Replicated Block Device (DRBD) is an open-source replication storage system. DRBD works by layering over logical block devices on participating nodes. The aim of the project is to write a packet tracer and logger for DRBD.

DRBD is implemented as a loadable kernel module. We also want to write a loadable kernel module. This module will be responsible for communicating with the drbd module and preparing the log file. We will add an user land utility which will make system calls to our module and be responsible for analysing the data in the log file and display results and reports in graphical format.

## LIST OF TABLES

<b>Sr. No.</b>	<b>Table No.</b>	<b>Table Name</b>	<b>Page No.</b>
1.	Tab 1.	List of Figures	5
2.	Tab 2.	Abbreviations	5

## LIST OF FIGURES

<b>Sr. No.</b>	<b>Figure No.</b>	<b>Figure Name</b>	<b>Page No.</b>
1.	Fig 1.	Linux Architecture	10
2.	Fig 2.	Gantt Chart	15
3.	Fig 3.	Architecture Diagram	17
4.	Fig 4.	Character Driver Code	20
5.	Fig 5.	Data Structure of Log	21
6.	Fig 6.	Cscope	21
7.	Fig 7.	System Tap	22

Tab 1. List of Figures

## ABBREVIATIONS

<b>Sr. No.</b>	<b>Abbreviations</b>	<b>Full Form</b>
1.	DRBD	Distributed Replicated Block Device
2.	VM	Virtual Machine
3.	LKM	Linux Kernel Module

Tab 2. Abbreviations

# CONTENTS

Chapter	Topic	Page No
	<b>TITLE</b>	<b>1</b>
	<b>SPONSORSHIP LETTER</b>	<b>2</b>
	<b>ACKNOWLEDGEMENT</b>	<b>3</b>
	<b>ABSTRACT</b>	<b>4</b>
	List of Tables	5
	List of Figures	5
	List of Abbreviations	5
1	Introduction	8
	1.1 Title	8
	1.2 Overview	8
	1.3 Motivation	8
2	Literature Review	9
	2.1 Existing Methodologies	9
	2.1.1 DRBD	9
	2.2.2 DRBD vs RAID	9
	2.2 Proposed Methodologies	9
	2.2.1 Linux Architecture	9
	2.2.2 Character Device Driver	10
	2.2.3 Linux Kernel Module	11
3	Requirement Specifications and Analysis	13
	3.1 Problem Definition	13
	3.2 Concept	13
	3.3 Scope	13
	3.4 Objective	13
	3.5 Project Requirements	14

	3.5.1	Functional Requirements	14
	3.5.2	Non Functional Requirements	14
	3.5.3	Hardware Requirements	14
	3.5.4	Software Requirements	14
	3.6	Project Plan	14
	3.6.1	Modules Split-Up	14
	3.6.2	Gantt Chart	15
4		System Analysis and Design	16
	4.1	Architecture	16
	4.2	Components of the System	17
5		Implementation	18
	5.1	Stages of Implementation	18
	5.1.1	Character Device Driver	18
	5.1.2	Data Structure of Log File	21
	5.2	Elaborate Implementation Issues Techniques/ Software Tools etc.	21
	5.2.1	Cscope	21
	5.2.2	System Tap	22
6		Conclusion	23
	6.1	Conclusion	23
	6.2	Drawbacks	23
7		References	24
8		Appendices	25
	8.1	Project Weekly Log	25
	8.2	Plagiarism Report	26

## **1. INTRODUCTION**

### **1.1. TITLE**

Distributed Replicated Block Device (DRBD) Network Packet Tracing Module

### **1.2. OVERVIEW**

DRBD is an open source data replication software that is used to replicate the data at multiple nodes. It consists of a primary node, and one or several other secondary nodes. Our project is to build a packet tracer for DRBD which would maintain a log file of all the writes and reads at the primary and secondary nodes respectively.

DRBD is implemented as a loadable kernel module. We also write a loadable kernel module. The main aim of this module is to create a global queue for storing all packets being sent and received by the node in the cluster. From this queue, a log file is created in the user space.

We will also add a user space daemon. This daemon will be responsible to read and analyse the log file. This daemon will also display the data in a visual format (graphs).

### **1.3. MOTIVATION**

- Opportunity to work on an open-source project.
- We got first hand industry exposure by working with Veritas LLC, Pune.
- This project also gave us the opportunity to work on an industry standard replication project.
- We had the opportunity to learn about cluster networks, as well as about system programming.



## **2. LITERATURE REVIEW**

### **2.1. EXISTING METHODOLOGIES**

#### **2.1.1. DRBD**

DRBD stands for Distributed Replicated Block Device. It is an open source data replication storage solution. It consists of a primary node, and one or several other secondary nodes. DRBD works by layering over logical block devices on participating nodes. Primary node is responsible for functioning of the system.

When a write operation occurs at the primary node, it is simultaneously propagated to secondary nodes. In case the primary node fails, then one of the secondary nodes is promoted to the designation of primary node. [1]

DRBD consists of two independent modules -

- A kernel module, that implements a driver replicated across the primary and secondary nodes.
- User space applications, that are responsible for administration.

#### **2.1.2. DRBD vs RAID**

- RAID is designed for two or more disks connected locally whereas DRBD is designed to replicate a block-device over a network.
- We use RAID (software or hardware) to increase the reliability of the local storage over 2 or more disks. DRBD would sit on top of your RAID and replicate that data to another server for failover purposes.
- DRBD can also allow two servers to access that data at the same time, which you can't really do with RAID in a cluster-aware file system.

### **2.2. PROPOSED METHODOLOGIES**

#### **2.2.1. LINUX ARCHITECTURE**

The Linux System consists of four primary components -

- Hardware layer
- Kernel

- Shell
- User applications and utilities

The kernel is the central part of any operating system. It is responsible for the overall functioning of the computer and the hardware. The kernel is a separate entity as compared to the user space, and hence the programs written for the kernel are different as compared to the normal user programs.

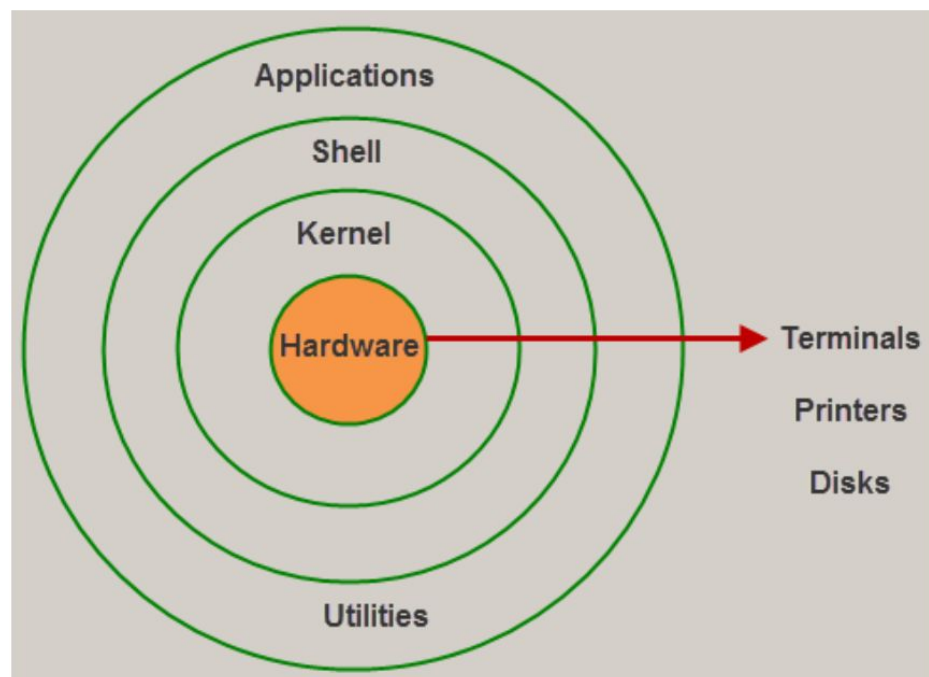


Fig 1. Linux Architecture

### 2.2.2. CHARACTER DEVICE DRIVER

There are two types of device drivers -

- Character Driver
- Block Driver

This division is made on the basis of speed and the organisation of data in the system.

Character drivers are usually slow speed devices, which handle comparatively smaller amounts of data and do not require frequent seeks or writes from and to the system. Examples - Keyboards, mouse, joysticks etc. These devices generally involve sequential read and write, i.e, byte by byte.

Block drivers are the ones which handle comparatively larger amounts of data. Devices such as hard disks, floppies and ram generally fall in this category. Reading and writing of data takes place at the block level in these devices. [2]

The operating system relies on two unique and fixed identifiers for identifying each device -

- Major Number
- Minor Number

Each unique identifier consists of two parts or halves. The first part is used to identify the type of the device (serial port, etc); and the second part is used to denote the device (1. disk, 2. serial port).

The major number usually identifies the driver, and the minor number denotes the device that is using the particular driver. Generally each driver has a major number that is responsible for all the minor numbers associated with that major number.

### **2.2.3. LOADABLE KERNEL MODULE**

Kernel configuration primarily refers to selecting the files that we want, when the kernel is compiled and run. Appending code to the kernel basically refers to adding the required files to the source code and recompiling the kernel. [5]

Loadable Kernel Module (LKM) refers to the code that is added to the kernel, without shutting it down completely.

LKMs are generally of three types -

- Device Drivers
- Filesystem Drivers
- System Calls

The kernel is designed so as to encapsulate the functions, so that new functions can be added to it without disturbing the basic functionalities. The new functions/ LKMs are added in complete isolation of the existing kernel code base.

LKMs are primarily used for these six tasks -

- Device drivers - The operating system (kernel) establishes contact with the hardware so as to ensure its proper functioning.
- Filesystem drivers - These drivers are used to understand the hierarchy of a filesystem in an operating system.
- System calls - User space applications and programs make use of system calls in order to perform certain tasks such as reading a file, creating a new process etc.
- Network drivers - These drivers are used to identify the network protocols..
- TTY line disciplines - A console command.
- Executable interpreters - Responsible to load and execute an executable file/ code.

### **3. REQUIREMENT SPECIFICATION AND ANALYSIS**

#### **3.1. PROBLEM DEFINITION**

Build a packet tracing module on DRBD to track the data packets transferred over a cluster in a network during data replication between multiple block devices and analyse the transfer of packets.

#### **3.2. CONCEPT**

Data Replication is the process of storing data in more than one site or node. It is useful in improving the availability of data. It is simply copying data from a database from one server to another server so that all the users can share the same data without any inconsistency. Packet tracing is the process by which you can verify the path of a packet through the layers to its destination. Using packet tracing it is possible to check the volume of data being transferred and various information related to the process of replication as at its core, replication occurs through the transfer of packets.

#### **3.3. SCOPE**

- Provide an enterprise grade solution to analyze the packet transfer in replication phase at multiple data nodes.
- Calculate the accuracy of data replicated at multiple data nodes in a cluster.
- Determine the bandwidth required over a network based on the accuracy of the transfer of packets.
- Determine the number of replication nodes required in a data cluster.
- Learning Scope :
  - Work on an open-source project with industry guidance
  - Learn Linux programming
  - Learn about

#### **3.4. OBJECTIVE**

- Understand how replication in DRBD works - code flow how/where we put the packet on air in code.
- Writing a kernel module to trace/keep track of logs in memory and on-disk.
- Enable tracing on the modules dynamically.

- Least or no performance impact is desirable.
- Writing utility/daemon to pull tracing information from kernel - parse the data for reporting/debugging problems
- Use a container to run the analyser tool.
- Eg: plot a graph, create sequential logs combining primary and secondary logs.

### **3.5. PROJECT REQUIREMENTS**

#### **3.5.1. FUNCTIONAL REQUIREMENTS**

- Log data and acknowledgement packets passed between nodes in a DRBD cluster
- Analyse above mentioned logs
- Represent analysis in graphical format

#### **3.5.2. NON-FUNCTIONAL REQUIREMENTS**

- No performance impact on DRBD cluster

#### **3.5.3. HARDWARE REQUIREMENTS**

- System with i5 or higher
- System with 8 GB of RAM

#### **3.5.4. SOFTWARE REQUIREMENTS**

- Oracle VM VirtualBox
- Two Virtual Machines
- CentOS running on two VMs
- DRBD

### **3.6. PROJECT PLAN**

#### **3.6.1. MODULE SPLIT-UP**

- Understanding and Identifying Data Flow Points in DRBD.
- Creating Global Queue for logs in the module.
- Calling our module functions from DRBD.

- Preparing a log file.
- Writing a user land utility to analyse data from a log file.
- Display analysis in graphical format.

### 3.6.2. GANTT CHART

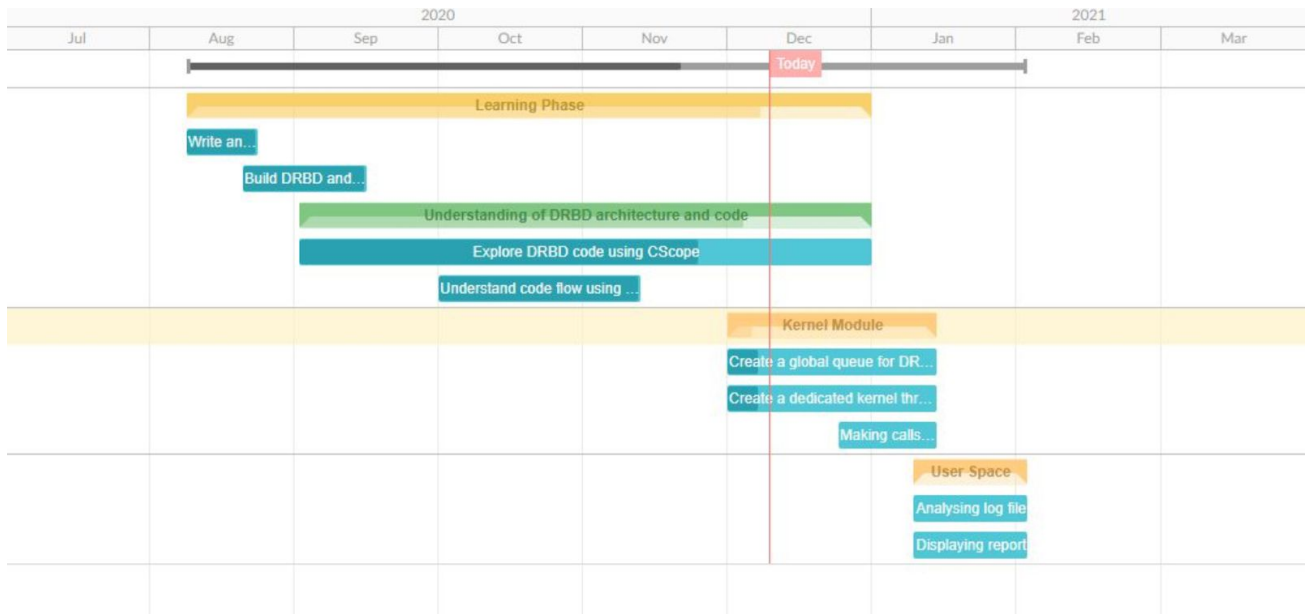


Fig 2. Gantt Chart

## **4. SYSTEM ANALYSIS AND DESIGN**

### **4.1. ARCHITECTURE**

Being an open source software, we can review and make changes in the code base of DRBD. We can identify the points where data packets are sent and received in the code.

We will implement our module as a loadable kernel module. This module will be responsible for communicating with the drbd module and preparing the log file. We will add an user land utility which will make system calls to our module and be responsible for analysing the data in the log file and display results and reports in graphical format.

The kernel space consists of a global queue, where the corresponding details will be enqueued and dequeued as and when the read and write operations take place at the primary and secondary nodes respectively. We then plan to write a user space utility that will take this log data as the input, and plot different graphs accordingly.



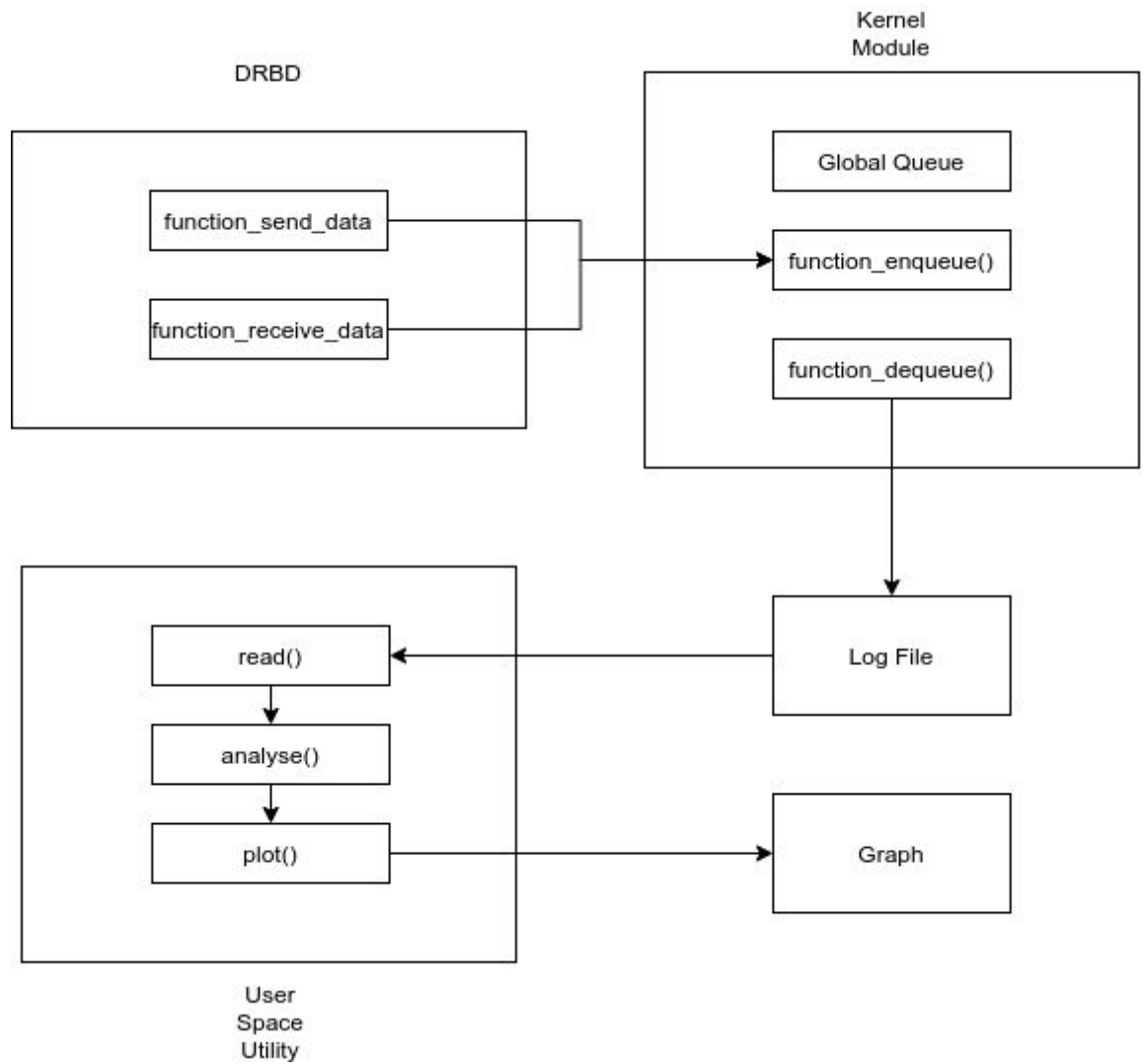


Fig 3. Architecture Diagram

#### 4.2. COMPONENTS OF THE SYSTEM

1. K-thread
2. User space entity
3. Kernel module
4. Global queue

## 5. IMPLEMENTATION

### 5.1. STAGES OF IMPLEMENTATION

#### 5.1.1. CHARACTER DEVICE DRIVER

```

1  #include<linux/kernel.h>
2  #include<linux/init.h>
3  #include<linux/module.h>
4  #include<linux/kdev_t.h>
5  #include<linux/fs.h>
6  #include<linux/cdev.h>
7  #include<linux/device.h>
8  #include<linux/slab.h>
9  #include<linux/uaccess.h>
10 #include<linux/ioctl.h>
11
12 #define mem_size 1024
13
14 // Define the ioctl code
15 #define WR_DATA_IOW('a', 'a', int32_t*)
16 #define RD_DATA_IOR('a', 'b', int32_t*)
17
18 int32_t val = 0;
19
20 dev_t dev = 0;
21 static struct class *dev_class;
22 static struct cdev my_cdev;
23 uint8_t *kernel_buffer;
24
25
26 static int __init chr_driver_init(void);
27 static void __exit chr_driver_exit(void);
28 static int my_open(struct inode *inode, struct file *file);
29 static int my_release(struct inode *inode, struct file *file);
30 static ssize_t my_read(struct file *filp, char __user *buf, size_t len, loff_t *off);
31 static ssize_t my_write(struct file *filp, const char __user *buf, size_t len, loff_t *off);
32 static long chr_ioctl(struct file *file, unsigned int cmd, unsigned long arg);

```

```

34 static struct file_operations fops =
35 {
36     .owner      = THIS_MODULE,
37     .read       = my_read,
38     .write      = my_write,
39     .open       = my_open,
40     .unlocked_ioctl = chr_ioctl,
41     .release    = my_release,
42 };
43
44
45 static int my_open(struct inode *inode, struct file *file)
46 {
47     /* creating physical memory */
48     if((kernel_buffer = kmalloc(mem_size, GFP_KERNEL)) == 0)
49     {
50         printk(KERN_INFO "cannot allocate memory to the kernel...\n");
51         return -1;
52     }
53     printk(KERN_INFO "Device file opened...\n");
54     return 0;
55 }
56
57 static int my_release(struct inode *inode, struct file *file)
58 {
59     kfree(kernel_buffer);
60     printk(KERN_INFO "Device file closed...\n");
61     return 0;
62 }
63
64 static ssize_t my_read(struct file *filp, char __user *buf, size_t len, loff_t *off)
65 {
66     if(copy_to_user(buf, kernel_buffer, mem_size) == 0)
67         printk(KERN_INFO "Data is read successfully...\n");
68     return mem_size;

```

```

71 static ssize_t my_write(struct file *filp, const char __user *buf, size_t len, loff_t *off)
72 {
73     if(copy_from_user(kernel_buffer, buf, len) == 0)
74         printk(KERN_INFO "Data is written successfully...\n");
75     return len;
76 }
77
78 static long chr_ioctl(struct file *file, unsigned int cmd, unsigned long arg)
79 {
80     switch(cmd)
81     {
82         case WR_DATA:
83             if(copy_from_user(&val, (int32_t*)arg, sizeof(val)) != 0)
84             {
85                 printk(KERN_INFO "ioctl write failed...\n");
86                 return 0;
87             }
88             printk(KERN_INFO "(ioctl) Value written = %d\n", val);
89             break;
90         case RD_DATA:
91             if(copy_to_user((int32_t*)arg, &val, sizeof(val)) != 0)
92             {
93                 printk(KERN_INFO "ioctl read failed...\n");
94                 return 0;
95             }
96             break;
97     }
98     return 0;
99 }
100

```

```

101 static int __init chr_driver_init(void)
102 {
103
104
105     /* Allocating Major Number */
106     if((alloc_chrdev_region(&dev, 0, 1, "my_Dev")) < 0)
107     {
108         printk(KERN_INFO"Cannot allocate the major number...\n");
109         return -1;
110     }
111     printk(KERN_INFO"Major = %d\nMinor = %d\n", MAJOR(dev), MINOR(dev));
112
113
114
115     /* creating cdev structure */
116     cdev_init(&my_cdev, &fops);
117
118
119
120     /* adding character device to system */
121     if((cdev_add(&my_cdev, dev, 1)) < 0)
122     {
123         printk(KERN_INFO"Cannot add the device to the system...\n");
124         goto r_class;
125     }
126
127
128
129     /* creating struct class */
130     if((dev_class = class_create(THIS_MODULE, "my_class")) == NULL)
131     {
132         printk(KERN_INFO"cannot create the struct class...\n");
133         goto r_class;
134     }
135
136
137     /* creating device */
138     if((device_create(dev_class, NULL, dev, NULL, "my_device")) == NULL)
139     {
140         printk(KERN_INFO"cannot create the device...\n");
141         goto r_device;
142     }
143     printk(KERN_INFO"device driver is inserted...\n");
144     return 0;
145
146
147 r_device:
148     class_destroy(dev_class);
149
150
151 r_class:
152     unregister_chrdev_region(dev, 1);
153     return -1;
154 }
155
156 void __exit chr_driver_exit(void)
157 {
158     device_destroy(dev_class, dev);
159     class_destroy(dev_class);
160     cdev_del(&my_cdev);
161     unregister_chrdev_region(dev, 1);
162     printk(KERN_INFO"device driver is removed successfully...\n");
163 }
164
165 module_init(chr_driver_init);
166 module_exit(chr_driver_exit);
167
168 MODULE_LICENSE("GPL");
169 MODULE_AUTHOR("Animesh");

```

Fig 4. Character Device Driver Code

### 5.1.2. DATA STRUCTURE OF LOG

```
struct dump_data{
    int    send_recv;
    int    type;
    enum   drbd_packet;
    void*   data;
}
```

Fig 5. Data Structure of Log

## 5.2. ELABORATE IMPLEMENTATION ISSUES/ TECHNIQUES/ SOFTWARE TOOLS ETC.

### 5.2.1. CSCCOPE

Cscope is used to browse the code in the console. Cscope generally works along with Vim. Using Cscope we identified the main important functions in DRBD source code like drbd\_send\_dblock etc. [6]

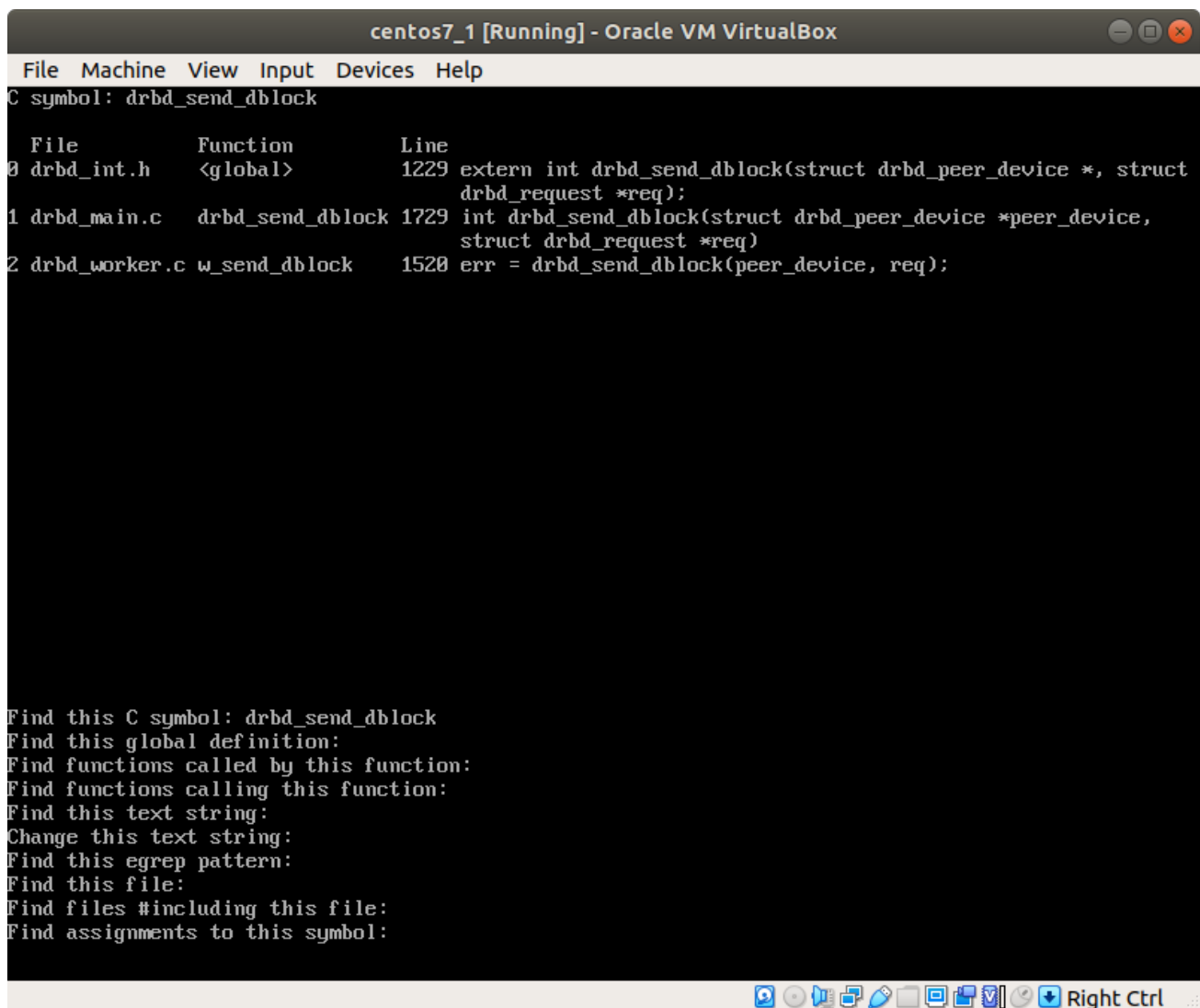


Fig 6. Cscope

### 5.2.2. SYSTEM TAP

System Tap is generally used along with Cscope to find the flow data, in the functions identified using Cscope. [7]

```
probe begin {  
    printf("System Tap Started\n");  
}  
  
probe module("drbd").function("drbd_send_dblock").call {  
    printf("drbd_send_dblock is called\n");  
}  
  
probe module("drbd").function("drbd_send_block").call {  
    printf("drbd_send_block is called\n");  
}  
  
probe module("drbd").function("drbd_receiver").call {  
    printf("drbd_receiver is called\n");  
}  
  
probe end {  
    printf("\n\nExit\n\n\n");  
}
```

Fig 7. System Tap

## **6. CONCLUSION**

### **6.1. CONCLUSION**

Thus in this project, we have studied about DRBD which is an open source, data replication technology. DRBD is used to replicate the data between a primary, and one or multiple secondary nodes.

We also started the implementation of a network packet tracing module for DRBD, that will basically maintain a log of all the reads and writes at the primary and secondary data locations. This is done by using a global queue.

We will also develop a user space daemon that will help represent all the data from the log in a visual format (in the form of graphs, etc).

### **6.2. DRAWBACKS**

DRBD is comparatively more difficult to set up as compared to RAID. DRBD also does not promise the performance benefits, as in the case of RAID (especially RAID 0).

The packet tracing module will be written inside the DRBD send data function and called as and when the read or write operation is performed. Thus the entire DRBD code needs to be re-compiled in case of any error in the code.

## 7. REFERENCES

- [1] <https://www.linbit.com/drbd/>
- [2] [https://linux-kernel-labs.github.io/refs/heads/master/labs/device\\_drivers.html](https://linux-kernel-labs.github.io/refs/heads/master/labs/device_drivers.html)
- [3] <https://www.elprocus.com/linux-operating-system/>
- [4] <https://developer.ibm.com/tutorials/l-drbd/>
- [5] <https://tldp.org/HOWTO/Module-HOWTO/x73.html>
- [6] [https://courses.cs.washington.edu/courses/cse451/12sp/tutorials/tutorial\\_cscope.html](https://courses.cs.washington.edu/courses/cse451/12sp/tutorials/tutorial_cscope.html)
- [7] [https://access.redhat.com/documentation/en-us/red\\_hat\\_enterprise\\_linux/5/html/](https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/5/html/)
- [8] D. A. Patterson, P. Chen, G. Gibson and R. H. Katz, "Introduction to redundant arrays of inexpensive disks (RAID)," Digest of Papers. COMPCON Spring 89. Thirty-Fourth IEEE Computer Society International Conference: Intellectual Leverage, San Francisco, CA, USA, 1989, pp. 112-117, doi: 10.1109/COMPCON.1989.301912.
- [9] M. Wiesmann, F. Pedone, A. Schiper, B. Kemme and G. Alonso, "Understanding replication in databases and distributed systems," Proceedings 20th IEEE International Conference on Distributed Computing Systems, Taipei, Taiwan, 2000, pp. 464-474, doi: 10.1109/ICDCS.2000.840959.



## 8. APPENDICES

### 8.1. PROJECT WEEKLY LOG

Savitribai Phule Pune University













Faculty of Information Technology

**PUNE INSTITUTE OF COMPUTER TECHNOLOGY, PUNE.**  
**Department of Information Technology**  
**(Academic Year: 2020-21)**

**Semester - I**

**Monthly Planning Sheet**

Academic Year: 2020 - 21

Week No.	Activity Planned	Activity Completed Status	Student Signature	Guide Signature
<b>Week 1</b>	Analysing the problem statement	Completed		
<b>Week 2</b>	Learn about DRBD	Completed		
<b>Week 3</b>	Learn about Kernel Modules	Completed		
<b>Week 4</b>	Write linux character driver	Completed		
<b>Week 5</b>	Build DRBD and configure it	Completed		
<b>Week 6</b>	Understanding DRBD architecture and code	Completed		
<b>Week 7</b>	Learn about Cscope and System Tap	Completed		
<b>Week 8</b>	Explore DRBD code using Cscope	Completed		
<b>Week 9</b>	Explore DRBD code using Cscope	Completed		
<b>Week 10</b>	Understand code flow using System Tap	Completed		
<b>Week 11</b>	Create global queue for DRBD packets	Ongoing		
<b>Week 12</b>	Create dedicated kernel thread	Ongoing		

Project Coordinator

Internal Guide




## 8.2. PLAGIARISM REPORT



### Document Information

Analyzed document	21_VRJ_Project_Phase_1_Report-New.pdf (D90235099)
Submitted	12/19/2020 3:50:00 PM
Submitted by	Vishal Jaiswal
Submitter email	vrjaiswal@pict.edu
Similarity	6%
Analysis address	vrjaiswal.pict@analysis.arkund.com

### Sources included in the report

SA	<b>Pune Institute of Computer Technology / Final Report (3).pdf</b>		
	Document Final Report (3).pdf (D68239032)		3
	Submitted by: nvburadkar@pict.edu Receiver: nvburadkar.pict@analysis.arkund.com		
W	URL: <a href="https://lists.linbit.com/pipermail/drbd-user/2010-February/013449.html">https://lists.linbit.com/pipermail/drbd-user/2010-February/013449.html</a>		1
	Fetches: 12/19/2020 3:50:00 PM		
W	URL: <a href="https://www.elprocus.com/linux-operating-system/">https://www.elprocus.com/linux-operating-system/</a>		1
	Fetches: 12/19/2020 3:51:00 PM		
W	URL: <a href="https://sites.ualberta.ca/dept/chemeng/AIX-43/share/man/info/C/a_doc_lib/aixbman/c...">https://sites.ualberta.ca/dept/chemeng/AIX-43/share/man/info/C/a_doc_lib/aixbman/c ...</a>		1
	Fetches: 12/19/2020 3:51:00 PM		
SA	<b>Thesis_luhtala_final_pdf_a.pdf</b>		
	Document Thesis_luhtala_final_pdf_a.pdf (D14518752)		1

## 8.3. REVIEW 1 CHECKLIST

Savitribai Phule Pune

Faculty of Information

**PUNE INSTITUTE OF COMPUTER  
TECHNOLOGY, PUNE.**  
Department of Information Technology  
**PROJECT REVIEW – I**

Group Id :		21	Date : 22/09/2020	
Project Title :				
Sr.No	Roll No.	Student Name	Contact Details	Internal / External Guide Details
1	43110	Bharat kothari	bharatkothari1008@gmail.com	InternalGuide Name : Prof VR Jaiswal External guide: Prof Tushar rane Prof Swapnil mane
2	43132	Animesh landge	anni.landge@gmail.com	Mentor Name, email & Mobile No. : Mr Swapnil Ujgare 9657765662
3	43258	Shrijan vats	Shrijanv.123@gmail.com	Swapnil.Ujgare@veritas.com
4	43304	Amod dhopavkar	amoddhopavkar2@gmail.com	

**REVIEW – I CHECKLIST : FINALIZATION OF SCOPE** **25 Marks**

<b>PROJECT STATEMENT</b>	
1. Is the statement short and concise (10-20 words maximum)?	Y
2. Does the statement gives clear indication about what your project will accomplish?	Y
3. Can a person who is not familiar with the project understand scope of the project by reading the Project Problem Statement?	Y
<b>REQUIREMENT: SCOPE AND OBJECTIVES</b>	
Does the Scope and Objectives establish the "context" for the proposed project by referencing to the following elements:	
a. Are all aspects of the requirements document (i.e., Functional Spec.) addressed in the design?	Y
b. Is the architecture / block diagram well defined and understood?	Y
c. The project's objective of study (what product, process, resource etc.) is being addressed?	Y
d. The project's purpose is the purpose of project addressed properly (why it's being pursued: to evaluate, reduce, increase, etc.)?	Y
e. The project's viewpoint: Is the project's viewpoint is understood? (Who is the project's end user)?	Y
f. Is the project goal statement in alignment with the sponsoring organization's business goals and mission?	Y
<b>ANALYSIS</b>	
1. Is information domain analysis complete, consistent and accurate?	Y

Page 14 of

Savitribai Phule Pune

Faculty of Information

**PUNE INSTITUTE OF COMPUTER  
TECHNOLOGY, PUNE.**  
**Department of Information Technology**  
**PROJECT REVIEW – I**

2. Is problem statement categorized in identified area and targeted towards specific area therein?	Y
3. Are external and internal interfaces properly defined?	Y <sup>+</sup>
4. Does the Use Case Model properly reflect the actors and their roles and responsibilities?	Y <sup>+</sup>
5. Are all requirements traceable to system level?	Y
6. Is similar type of methodology / model is used for existing work?	NA
7. Are requirements consistent with schedule, resources, and budget?	Y

Savitribai Phule Pune

Faculty of Information

**PUNE INSTITUTE OF COMPUTER  
TECHNOLOGY, PUNE.**  
**Department of Information Technology**  
**PROJECT REVIEW – I**  
**STUDENT PERFORMANCE  
EVALUATION**

<b>Students' Contribution and</b>				
<b>Particulars</b>	<b>Marks(25M)</b>			
	<b>Group Members</b>			
	1	2	3	4
1. Background and Topic (4 M)	3	3	3	3
2. Project Scope and Objectives (4M)	4	4	4	4
3. Literature Survey (5 M)	4	4	4	4
4. Project Planning (4 M)	4	4	4	4
5. Presentation Skills (4 M)	4	4	4	4
6. Question and Answer (4 M)	3	3	3	3
<b>Total(25M)</b>	22	22	22	22
<b>Comments (If any)</b>				

# To be filled by internal guide &amp; reviewer(s) only.

\* Whether the presentation / evaluation is as per the schedule. : YES / NO (If NO mention the reasons for the same.)

**Review – I: Deliverables**

<ul style="list-style-type: none"> <li>• Problem Statement / Title</li> <li>• Purpose, Scope, Objectives</li> <li>• Abstract (System Overview)</li> <li>• Introduction (Architecture and High-Level Design)</li> </ul>	<ul style="list-style-type: none"> <li>• H/W, S/W &amp; other requirement, Test Environment/Tools</li> <li>• Literature Survey</li> <li>• References</li> <li>• Project Plan 1.0 (<b>Gantt Chart</b>)</li> </ul>
--	--

Name &amp; Signature of evaluation committee –

Prof Tushar rane  
Name of Reviewer 1

Prof Swapnil Mane  
Name of Reviewer 2

Prof Vishal jaiswal  
Name of Internal Guide

Page 15 of



## 8.4. REVIEW 2 CHECKLIST

Savitribai Phule Pune University

Faculty of Information Technology

PUNE INSTITUTE OF COMPUTER TECHNOLOGY, PUNE.

Department of Information Technology

**PROJECT REVIEW – II**  
**(Academic Year: 2020-21)**

Group Id :		21	Date : 10/12/2020	
Project Title : Distributed Replicated Block Device (DRBD) Network Packet Tracing Module				
Sr. No.	Roll No.	Student Name	Contact Details	Internal / External Guide Details
1	43110	Bharat Kothari	bharatkothari1008@gmail.com	Guide Name : Prof. Vishal Jaiswal
2	43132	Animesh Landge	anni.landge@gmail.com	Mentor Name, email & Mobile No. : Mr. Swapnil Ujgare Swapnil.Ujgare@veritas.com
3	43258	Shrijan Vats	shrijanv.123@gmail.com	
4	43304	Amod Dhopavkar	amoddhopavkar2@gmail.com	

**REVIEW – II CHECKLIST : DESIGN****25 Marks**

DESIGN	
1. Are requirements reflected in the system architecture?	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *
2. Does the design support both project (product) and project goals?	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *
3. Does the design address all the issues from the requirements?	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *
4. Is effective modularity achieved and modules are functionally independent?	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *
5. Are structural diagrams (Class, Object, etc.) well defined and understood?	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *
6. Are all class associations clearly defined and understood? (Is it clear which classes provide which services)?	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *
7. Are the classes in the class diagram clear? (What they represent in the architecture design document?)	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *
8. Is inheritance appropriately used?	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *
9. Are the multiplicities in the use case diagram depicted in the class diagram?	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *
10. Are behavioral diagrams (use case, sequence, activity, etc.) well defined and understood?	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *
11. Is aggregation/containment (if used) clearly defined and understood?	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *
12. Does each case have clearly defined actors and input/output?	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *
13. Is all concurrent processing (if used) clearly understood and reflected in the sequence diagrams?	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *
14. Are all objects used in sequence diagram?	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *
15. Does the sequence diagram match class diagram?	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *
16. Are the symbols used in all diagrams correspond to UML standards?	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *

**PUNE INSTITUTE OF COMPUTER TECHNOLOGY, PUNE.****Department of Information Technology****PROJECT REVIEW – II**  
**(Academic Year: 2020-21)****STUDENT PERFORMANCE EVALUATION**

Students' Contribution and Performance				
Particulars	Marks(25M)			
	Group Members			
	1	2	3	4
1. System Architecture & Literature Survey (Review-I)	Y/Y	Y/N	Y/Y/N	Y/N
2. Project Design (5 M)	5	5	5	5
3. Methodology /Algorithms and Project Features (5 M)	4	4	4	4
4. Project Planning (2 M)	2	2	2	2
5. Basic details of Implementation (5 M)	4	4	4	4
6. Presentation Skills ( 4 M)	4	4	4	4
7. Question and Answer (4 M)	4	4	4	4
8. Summarization of ultimate findings of the Project	Y/Y	Y/N	Y/Y/N	Y/N
<b>Total(25M)</b>	<b>23</b>	<b>23</b>	<b>23</b>	<b>23</b>
Comments (if any)				

# To be filled by internal guide &amp; reviewer(s) only.

\* Whether the presentation / evaluation is as per the schedule. : YES / ~~NO~~ (If NO mention the reasons for the same.)**Review – II: Deliverables**

YES

<ul style="list-style-type: none"> <li>• Problem Statement / Title</li> <li>• Abstract</li> <li>• Introduction</li> <li>• Literature Survey (comparison with existing system)</li> <li>• Methodology</li> <li>• Design / algorithms / techniques used</li> </ul>	<ul style="list-style-type: none"> <li>• Modules Split-up</li> <li>• Proposed System</li> <li>• Software Tools / Technologies to be used</li> <li>• Proposed Outcomes</li> <li>• Partial Report (Semester – I)</li> <li>• Project Plan 2.0 (Gantt Chart)</li> </ul>
--	---

Name &amp; Signature of evaluation committee -

Prof. Tushar Rane

Name of Reviewer 1

Prof. Swapnil Mane

Name of Reviewer 2

Prof. Vishal Jaiswal

Name of Internal Guide

## 8.5. RESEARCH REVIEW CHECKLIST

Savitribai Phule Pune University

Faculty of Information Technology

**PUNE INSTITUTE OF COMPUTER TECHNOLOGY, PUNE.**  
**Department of Information Technology**  
**RESEARCH PUBLICATION REVIEW – I**  
**(Academic Year: 2020-21)**

Group Id :		21		Date : 21/10/2020	
Project Title : Distributed Replicated Block Device (DRBD) Network Packet Tracing Module					
Sr. No.	Roll No.	Student Name	Contact Details	Internal / External Guide Details	
1	43110	Bharat Kothari	bharatkothari1008@gmail.com	Guide Name : Prof. Vishal Jaiswal	
2	43132	Animesh Landge	anni.landge@gmail.com	Mentor Name, email & Mobile No. :  Mr. Swapnil Ujgare Swapnil.Ujgare@veritas.com	
3	43258	Shrijan Vats	shrijanv.123@gmail.com		
4	43304	Amod Dhobavkar	amoddhobavkar2@gmail.com		

**RESEARCH PUBLICATION REVIEW – I CHECKLIST****25 Marks**

Publication based on the Proposed Methodology	
1. Is the Problem Clearly defined and concise? (Which Challenge / issue is addressed by this research?)	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *
2. Is Abstract precisely written and are Keywords correctly identified?	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *
3. Is motivation/significance of the research work is defined?	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *
4. Is Literature Survey comprehensive, systematic?	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *
5. Is comparative analysis demonstrated through implementation?	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *
6. Is new methodology/algorithm proposed precisely?	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *
7. Does the system architecture/ workflow diagram match the proposed methodology?	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *
8. Is conclusion with future scope communicated effectively?	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *
9. Is plagiarism checked?	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *
10. Are the WoS /Scopus indexed and /or UGC listed international journals and/or Scopus indexed international conferences identified?	Y / <del>N</del> / <del>NA</del> / <del>NC</del> *



**PUNE INSTITUTE OF COMPUTER TECHNOLOGY, PUNE.****Department of Information Technology****RESEARCH PUBLICATION REVIEW – I****(Academic Year: 2020-21)****STUDENT PERFORMANCE EVALUATION**

Students' Contribution and Performance				
Particulars	Marks(25M)			
	Group Members			
	1	2	3	4
1. System Architecture & Literature Survey (Review-I)	Y/N	Y/N	Y/N	Y/N
2. Precise Title, Abstract and Keywords (2 M)	2	2	2	2
3. Motivation and scope of research work (2 M)	2	2	2	2
4. Literature Survey and identification of research gap (5 M)	4	4	4	4
5. Proposed Methodology /Algorithm/System Architecture (5M)	4	4	4	4
6. Effective Conclusion and Future Scope (2 M)	2	2	2	2
7. Relevant References (3 M)	3	3	3	3
8. Effective Technical Writing and Presentation Skills (4 M)	4	4	4	4
9. Originality (Plagiarism <20%) (2M)	2	2	2	2
10. Identification of quality journals/international conferences	Y/N	Y/N	Y/N	Y/N
<b>Total(25M)</b>	<b>23</b>	<b>23</b>	<b>23</b>	<b>23</b>
Comments (if any)				

# To be filled by internal guide &amp; reviewer(s) only.

\* Whether the presentation / evaluation is as per the schedule. : YES / ~~NO~~ (If NO mention the reasons for the same.)**Research Publication Review – I: Deliverables**

YES

<ul style="list-style-type: none"> <li>Paper Title, Abstract and keywords</li> <li>Introduction</li> <li>Literature Survey</li> <li>Proposed Methodology/ Algorithm</li> <li>System Architecture/ Workflow Diagram</li> </ul>	<ul style="list-style-type: none"> <li>Conclusion and Future Scope</li> <li>References</li> <li>Identified WoS /Scopus indexed and /or UGC listed international journals and/or Scopus indexed international conferences.</li> </ul>
---	--

Name &amp; Signature of evaluation committee –

**Prof. Tushar Rane**

Name of Reviewer 1

**Prof. Swapnil Mane**

Name of Reviewer 2

**Prof. Vishal Jaiswal**

Name of Internal Guide