Scoring Indicators

COURSE NAME: SERVER ADMINISTRATION

COURSE CODE: 6131B **QID:** 2102240135B

<table>
<tr><th>Q.No</th><th>Scoring Indicators</th><th>Split score</th><th>Sub Total</th><th>Total score</th></tr>
<tr><td></td><td>PART A</td><td></td><td></td><td>9</td></tr>
<tr><td>I. 1</td><td>Ubuntu , Fedora, CentOS, Debian, Linux Mint, openSUSE – write any one</td><td>1</td><td>1</td><td rowspan="9"></td></tr>
<tr><td>I. 2</td><td>rmdir</td><td>1</td><td>1</td></tr>
<tr><td>I. 3</td><td>userdel</td><td>1</td><td>1</td></tr>
<tr><td>I. 4</td><td>ext4</td><td>1</td><td>1</td></tr>
<tr><td>I. 5</td><td>netstat</td><td>1</td><td>1</td></tr>
<tr><td>I. 6</td><td>/etc/apache2/apache2.conf</td><td>1</td><td>1</td></tr>
<tr><td>I. 7</td><td>user</td><td>1</td><td>1</td></tr>
<tr><td>I. 8</td><td>IP addresses</td><td>1</td><td>1</td></tr>
<tr><td>I. 9</td><td>Dump,restore,trar,rsync –write any one</td><td>1</td><td>1</td></tr>
<tr><td></td><td>PART B</td><td></td><td></td><td>24</td></tr>
<tr><td>II. 1</td><td>
<table>
<tr><th>Feature</th><th>Linux</th><th>Windows</th></tr>
<tr><td>Source Code</td><td>Open-source (kernel and many applications)</td><td>Closed-source (proprietary)</td></tr>
<tr><td>User Interface</td><td>Diverse (various desktop environments)</td><td>Consistent (Windows desktop environment)</td></tr>
<tr><td>File System</td><td>Ext4, XFS, Btrfs (commonly used)</td><td>NTFS (predominantly used)</td></tr>
<tr><td>Command Line</td><td>Terminal (Bash, Zsh, etc.)</td><td>Command Prompt (cmd.exe)</td></tr>
<tr><td>Package Management</td><td>Package managers (apt, yum, etc.)</td><td>Installer packages (MSI, exe)</td></tr>
<tr><td>Software Installation</td><td>Software repositories and package managers</td><td>Downloadable installers (exe, MSI)</td></tr>
<tr><td>Security Model</td><td>Users with limited privileges (root access)</td><td>Users with administrative privileges</td></tr>
<tr><td>Networking</td><td>Built-in networking tools (ifconfig, ip)</td><td>GUI-based network settings</td></tr>
<tr><td>Virtualization</td><td>Built-in support for virtualization</td><td>Hyper-V (built into Windows Pro/Enterprise)</td></tr>
</table>
</td><td>3</td><td>3</td><td></td></tr>
</table>

			(KVM)				
		Customization	Highly customizable and configurable	Limited customization options			
		Compatibility	Compatibility with open-source software	Compatibility with many commercial software			
		Licensing	Generally free and open-source (GNU GPL)	Proprietary, licensed (commercial versions)			
	Write any three						
II. 2	ls cp pwd tar cat mv rm mkdir rmdir write any six				.5X6	3	
II. 3	software management in Ubuntu is done using DPMS and all the attendant applications built around it, such as APT and dpkg. *Querying for Information* 1) command to list all currently installed software is: master@ubuntu-server:~$ dpkg –l • Command to get basic information about the installed Bash package is master@ubuntu-server:~$ dpkg -l bash • To view the list of files that comes with the Bash package, type dpkg-query -L bash *Installing Software in Ubuntu* • use dpkg to install a .deb file directly, or you can use apt-get to install any software available in the Ubuntu repositories • Installing and uninstalling software on a system is considered an administrative or privileged function. • To use dpkg to install a .deb package named lynx_2.9.8-2ubuntu-amd64.deb sudo dpkg - - install lynx_2.9.8-2ubuntu-amd64.deb • Using apt-get to install software is little easier ,because apt will take care of any dependency issues. • Using apt to install software is that you need to know only a part of the name of the software .you don't need to know the exact version number • Command to install a package called lynx sudo apt-get install lynx *Removing Software in Ubuntu* • Uninstalling software (such as lynx) in Ubuntu using dpkg is as easy as typing this: sudo dpkg --remove lynx • To remove the lynx package using apt-get, type sudo apt-get remove lynx Write any one from each				1 1 1	3	

II. 4	Linux periodically requires upgrades to fix bugs, improve performance, improve security, and add new features. These upgrades come out in two forms: in the form of a complete new kernel release and in the form of a patch .patches are usually distributed as files containing the changes made to the original source code. Primary source of Linux patch is official Linux kernel archive www.kernel.org. Here shows to apply a patch to update kernel Linux 4.20 to Linux 4.20.5 . Patch file name is patch-4.2.5.xz.	3	3	
II. 5	**Normal user** Normal user can access only what they own or have been permission to run • These users have limited access to system resources and are restricted from performing certain administrative tasks. **Superuser (root)** The superuser, often referred to as "root", is a special user account with unrestricted access to all commands and files on a Linux system.	1.5 1.5	3	
II.6	To start Apache on any distro that refers to Apache as httpd and also uses the service utility, use this command: [root@server ~]# service httpd start On Linux distributions running systemd, start the httpd daemon using the systemctl command like so: [root@fedora-server ~]# systemctl start httpd.service Debian-like systems like Ubuntu refer to the Apache binary as apache2, start Apache on such distros by running master@ubuntu-server:~$ sudo service apache2 start To shut down Apache, enter this command: [root@server ~]# service httpd stop Or on Ubuntu or Debian, you should instead run master@ubuntu-server:~$ sudo service apache2 stop write the commands from any one of the distro	1.5 1.5	3	
II. 7	filename fixed-address or fixed- get-lease-hostnames hardware max-lease-time next-server server-name use-lease-addr-for-default-route authoritative write any six	.5X6	3	
II. 8	1. First, create the mount point if it does not exist: [root@clientB ~]# mkdir -p /mnt/smb 2. Then run the mount.cifs command (via mount -t cifs) to do the actual mounting: [root@clientB ~]# mount -t cifs -o guest //serverA/samba-share /mnt/smb Here, //serverA/samba-share is the remote share being mounted, and /mnt/smb is the local mount point. If the remote share that you want to mount is protected with a username /password combination, you can supply the username/password along with the mount command like this example using the user yyang's account: # mount -t cifs -o username=yyang,password=19gan19 //serverA/samba-share /mnt/smb To unmount this directory, use the umount command: [root@clientB ~]# umount /mnt/smb On Debian-based distros such as Ubuntu, you might have to install the cifsutils package, if it is not already installed, so that you can use the mount.cifs command (via mount -t cifs). This can be done by running this command: master@ubuntu-server:~$ sudo apt-get install cifs-utils	1.5 1.5	3	

II.9	**dump and restore** The **dump** tool works by making a copy of an entire file system. The dump utility stores all the information about its dumps in the /etc/dumpdates file. here is the command to perform a level 0 dump to /dev/st0 of the /dev/sda1 file system: [root@server ~]# dump -0 -f /dev/st0 /dev/sda1 The **restore** program reads the dumpfiles created by dump and extracts individual files and directories from them. Although restore is a command-line tool, it does offer a more intuitive interactive mode that lets you go through your directory structure from the tape. A typical invocation of restore is as follows: [root@server ~]# restore -ivf /dev/st0	1.5 1.5	3	
II.10	• /directory/to/export • client • ip_network • permissions Permissions- secure, ro, rw, noaccess, root_squash, no_root_squash, all_squash	2 1	3	
	PART C			
III.	**Selecting a Linux Distribution:** Choose a Linux distribution suitable for your needs. Popular choices include Ubuntu Server, CentOS, Debian, and Red Hat Enterprise Linux (RHEL). **Hardware Requirements:** Ensure that the hardware meets the requirements for running the chosen Linux distribution. This includes CPU, RAM, storage, and network connectivity. **Installation:** Boot the server from the installation media (CD, DVD, USB). Follow the prompts to select language, keyboard layout, timezone, and other preferences. Partition the disk(s) according to your needs. Common partitions include root (/), swap, and /home. Install the base system and select additional packages if needed. **Initial Configuration:** Set the hostname for the server. Configure the network settings, including IP address, subnet mask, gateway, and DNS servers. Update the system packages to ensure you have the latest security patches and updates. **User and Group Management:** Create a non-root user account with sudo privileges for administrative tasks. Set up additional user accounts as needed. Create and manage user groups for better access control. **Security Configuration:** Configure firewall settings using tools like iptables or firewalld to restrict access to services. Harden the server by disabling unnecessary services and removing unnecessary packages. Enable SELinux (Security-Enhanced Linux) or AppArmor if available to enforce mandatory access control policies. **SSH Configuration:** Secure SSH access by configuring SSH daemon settings. Disable SSH root login and password authentication if possible. Implement key-based authentication for SSH access. **Additional Software Installation:** Install any additional software or packages required for your specific use case. **Monitoring and Logging:** Set up monitoring tools such as Nagios, Zabbix, or Prometheus to monitor server health and performance. Configure logging to collect and analyze system logs for troubleshooting and security purposes. **Backup and Recovery:**	7	7	7

	Set up a backup strategy to regularly backup important data and configurations. Test the backup and recovery process to ensure data can be restored in case of failure.			
IV	**STEP 1 :** **Getting and Unpacking the Source Package** Software that comes in source form is generally made available as a tarball—that is,its individual files are archived into a single large file and then compressed • Obtaining a copy of the source code # wget -P /usr/local/src \http://ftp.gnu.org/gnu/hello/hello-2.10.tar.gz The file will be automatically saved into your /usr/local/src/ working directory. • After copying over the file or downloading the file, you will need to unpack (or untar) it. When unpacked, a tarball will generally create a new directory for all of its files. • First change your current working directory to the /usr/local/src directory where the hello tarball was downloaded and saved to: [root@server ~]# cd /usr/local/src • Next use the tar command to unpack and decompress the hello archive: [root@server src]# tar -xvzf hello-2.10.tar.gz Change to the new directory and list its contents. [root@server src]# cd hello-2.10; ls While you're in the /usr/local/src/hello-2.10 directory, use a pager to view the INSTALL file that comes with the hello program: [root@server hello-2.10]# less INSTALL The INSTALL file typically has directions for compiling and installing the package. **Step 2 :** **Configuring the Package** Most packages ship with an auto-configuration script; it is safe to assume they do, unless their documentation says otherwise. These scripts are typically named configure (or config), and they can accept parameters. To see what configure options come with a package, simply run [root@server hello-2.10]# ./configure –help If you are happy with the default options that the configure script offers, type the following: [root@server hello-2.10]# ./configure a run of the configure script will create a special type of file called a makefile. Makefiles are the foundation of the compilation phase. STEP 3 : **Compiling the Package** [root@server hello-2.10]#make The make tool reads all of the makefiles that were created by the configure script.These files tell make which files to compile and the order in which to compile them. STEP 4 : **Installing the Package** once the compile completes successfully, run the following: [root@server hello-2.10]# make install This will install the package into the location specified by the default prefix (or the --prefix) argument that was used with the configure script earlier. **Testing the Software** [root@server hello-2.10]# /usr/local/bin/hello Hello, world	2 2 1 1 1	7	7
V	**(i) /etc/fstab** is a configuration file thet mount can use • It contains information about filesystems and how they should be mounted on the system • This file contains a list of all partitions known to the system. • During the boot process, this list is read and the items in it are automatically mounted with the options specified therein. Format of entries in the /etc/fstab file • /dev/device /dir/to/mount fstype Parameters fs_freq fs_passno Filesystem: This field specifies the device or partition that contains the filesystem Mount Point: This field specifies the directory where the filesystem should be mounted. Filesystem Type: This field specifies the type of the filesystem	3.5	7	7

	Options: This field specifies mount options for the filesystem. Options control various aspects of how the filesystem is mounted, such as read/write permissions, access control, handling of errors, etc (ii)**File System Check (fsck)**: fsck serves as a critical tool for maintaining file system health, automatically checking and repairing data consistency during the boot process. • fsck is automatically executed during every boot, determined by the presence of /.autofsck and configurations in /etc/fstab. • If the ".autofsck" file contains the value "1", automatic file system checking is enabled, and fsck will be executed during boot if necessary. • If the ".autofsck" file contains the value "0" or if the file is not present, automatic file system checking is disabled, and fsck will not be executed during boot • Similar to Windows ScanDisk, its purpose is to check and repair any file system damage before proceeding with the boot process. • Shutdown Condition: If a clean shutdown occurred, the /.autofsck file is deleted, and fsck runs as specified in /etc/fstab. • In case of a hard shutdown, fsck checks all local disks listed in /etc/fstab. • If fsck identifies a problem during boot, it prompts the user with information about the issue and seeks permission to perform repairs. Generally, agreeing to repair is recommended	3.5		
VI	Command-Line User Management 1. useradd 2. userdel 3. usermod 4. groupadd 5. groupdel 6. groupmod useradd -useradd allows you to add a single user to the system example: useradd -c "Ying Yang" yyang usermod - The usermod command allows you to modify an existing user in the system. example: usermod -u 1600 bogususer userdel - it removes existing users example: userdel -r bogususer groupadd - groupadd command adds groups to the /etc/group file example: groupadd research groupdel- groupdel command removes existing groups specified in the /etc/group file. example: groupdel bogusgroup groupmod-The groupmod command allows you to modify the parameters of an existing group example: groupmod -n bogusgroup bogus	3 4	7	7
VII	(i) Killall The 'killall' command is a straightforward and powerful tool for managing processes in Linux. Its basic usage is to terminate all instances of a specific process. The command works by sending a signal to each instance of the process, instructing it to terminate. The command operates based on the process name, not the process ID. This feature makes it particularly useful when you want to terminate all instances of a specific process without having to identify each process ID individually. (ii) ps The **ps** command in Linux is a powerful tool that allows you to view information about the processes running on your Linux system. It stands for "process status" and it's a command that you can use from the command line to get information about the currently running processes. (iii) renice The renice command alters the nice value of one or more running processes. The *nice value* is the decimal value of the system scheduling priority of a process. By default, the processes affected are specified by their process IDs. When you specify a process group, the request applies to all processes in the process group. (iv) Top	3.5 3.5	7	7

	The top command is used to show the active Linux processes. It provides a dynamic real-time view of the running system. Usually, this command shows the summary information of the system and the list of processes or threads which are currently managed by the Linux kernel.			
VIII	**A boot loader** is essential for any operating system to initiate on standard PC hardware, serving as the first software program that runs when the computer starts. Example : GRUB, GRUB 2 and less commonly used LILO **GRUB** is the default boot loader for many modern Linux distributions, such as Fedora, Red Hat Enterprise Linux (RHEL), openSUSE, Debian, Mandrake, CentOS, Ubuntu, and more. GRUB is designed to comply with the Multiboot Specification, ensuring compatibility with various operating systems. The GRUB boot process occurs in stages, with each stage managed by specific GRUB image files. Two stages are essential for the boot process, while additional stages are optional and depend on the specific system setup. **GRUB 2** serves as the successor to the GRUB Legacy boot loader. Many mainstream Linux distributions have adopted GRUB 2, including Debian, Ubuntu, Kubuntu, and others. Primary configuration file is named grub.cfg. Not intended for direct user editing; content is automatically generated. Multiple files/scripts used for configuring GRUB's menu, stored under /etc/grub.d/ directory **LILO (Linux Loader)** LILO is a boot manager enabling the booting of multiple operating systems, each residing on its dedicated partition. Configuration is managed by a simple file (/etc/lilo.conf), specifying bootable partitions and the chosen kernels. LILO Operates as a "two-stage boot loader," involving two stages in the boot process. Stage 1: Presents a prompt or boot menu for the selection of the operating system during the boot process. Upon selecting an OS, LILO enters the second stage, initiating the actual boot process. Despite ongoing maintenance, mainstream Linux distributions no longer include LILO as a default boot manager.	1 2 2 2	7	7
IX	Several utilities are available for viewing the system routing table from the command line 1. **route** 2. **netstat** 3. **ip route** **route** Using route is one of the easiest ways to display route table—simply run route without any parameters. Here is a complete run, along with the output: [root@server ~]# route Kernel IP routing table Destination Gateway Genmask Flags Metric Ref Use Iface 10.10.2.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0 192.168.1.0 0.0.0.0 255.255.255.0 U 0 0 0 eth1 The first is the 10.10.2.0 network, which is accessible via the first Ethernet device, eth0. The second is the 192.168.1.0 network, which is connected via the second Ethernet device, eth1. **netstat** Normally, the netstat program is used to display the status of all of the network connections on a host. However, with the -r option, it can also display the kernel routing table. Note that most other UNIX-based operating systems support the use of this method of viewing routes. [root@server ~]# netstat –r Like the route command, netstat can also take the -n parameter so that it does not perform hostname resolution. To use the netstat utility to display the IPv6 routing table, you can run the command: [root@server ~]# netstat -rn -A inet6 **ip route**	1 2 2	7	7

	the iproute package provides advanced IP routing and network device configuration tools. The ip command can also be used to manipulate the routing table on a Linux host. This is done by using the route object with the ip command. To view the contents of all the routing tables on the system, type this: [root@server ~]# ip route show table all	2		
X	Starting and Testing the FTP Server 1. Start the FTP service: [root@server ~]# service vsftpd start On Linux distributions running systemd, you can alternatively start the vsftpd daemon using the systemctl command: [root@fedora-server ~]# systemctl start vsftpd.service 2. Launch the command-line FTP client program, and connect to the local FTP server as an anonymous user: [root@server ~]# ftp localhost 3. Enter the name of the anonymous FTP user when prompted—that is, type ftp: Name (localhost:root): ftp 4. Enter anything at all when prompted for the password: Password: 5. Use the ls (or dir) FTP command to perform a listing of the files in the current directory on the FTP server: ftp> ls 6. Use the pwd command to display our present working directory on the FTP server: ftp> pwd 7. Using the cd command, try to change to a directory outside of the allowed anonymous FTP directory; for example, try to change the directory to the /boot directory of the local file system: ftp> cd /boot 550 8. Log out of the FTP server using the bye FTP command: ftp> bye 221 Goodbye.	7	7	7
XI	The typical route command is structured as follows route cmd type address netmask mask gw gway dev dn The parameters are as follows: Parameter Description cmd Either add or del, depending on whether you are adding or deleting a route. If you are deleting a route, the only other parameter you need is address. type Either -net or -host, depending on whether address represents a network address or a router address. Address The destination network for which you want to specify a route. netmask mask Sets the netmask of the address address to mask. gw gway Sets the gateway address for address to gway. Typically used for the default route. dev dn Sends all packets destined to address through the network device dn Here's how to set the default route on a sample host that has a single Ethernet device and a default gateway at 192.168.1.1: [root@server ~]# route add -net default gw 192.168.1.1 dev eth0 To add a default route to a system without an existing default route using the ip route utility, type this: [root@server ~]# ip route add default via 192.168.1.1	2 3 2	7	7
XII	Configuration options found in the sshd_config file: • AuthorizedKeysFile • Ciphers		7	7

	• HostKey • Port Specifies • Protocol • AllowTcpForwarding • X11Forwarding • ListenAddress ■ **AuthorizedKeysFile** Specifies the path to the file that contains the public keys that can be used for user authentication. The default is //.ssh/authorized_keys. ■ **Ciphers** This is a comma-separated list of ciphers allowed for the SSH protocol version 2. Examples of supported ciphers are 3des-cbc, aes256-cbc, aes256-ctr, arcfour, and blowfish-cbc. ■ **HostKey** Defines the file containing a private host key used by SSH. The default is either /etc/ssh/ssh_host_rsa_key or /etc/ssh/ssh_host_dsa_key or /etc /ssh/ssh_host_ecdsa_key, or /etc/ssh/ssh_host_ed25519 for protocol version 2. 22-ch22.indd 592 19/11/15 2:54 PM CHAPTER 22 Secure Shell (SSH) 593 NPL_2010 / Linux Administration: ■ **Port** Specifies the port number on which sshd listens. The default value is 22. ■ **Protocol** Specifies the protocol versions sshd supports. The possible values are 1 and 2. Note that protocol version 1 is generally considered insecure now. ■ **AllowTcpForwarding** Specifies whether Transmission Control Protocol (TCP) forwarding is permitted. The default is yes. ■ **X11Forwarding** Specifies whether X11 forwarding is permitted. The argument must be yes or no. The default is no. ■ **ListenAddress** Specifies the local address on which the SSH daemon listens. By default, OpenSSH will listen on both Internet Protocol version 4 (IPv4) and Internet Protocol version 6 (IPv6) sockets. But if you need to specify a particular interface address, you can tweak this directive.	3 4		
XIII	■ rpc.statd This process is responsible for sending notifications to NFS clients whenever the NFS server is restarted without being gracefully shut down. It provides status information about the server to rpc.lockd when queried. This is done via the Network Status Monitor (NSM) RPC protocol. ■ rpc.rquotad As its name suggests, rpc.rquotad supplies the interface between NFS and the quota manager. NFS users/clients will be held to the same quota restrictions that would apply to them if they were working on the local file system instead of via NFS. It is not required in NFSv4. ■ rpc.mountd When a request to mount a partition is made, the rpc.mountd daemon takes care of verifying that the client has the appropriate permission to make the request. This permission is stored in the /etc/exports file. It is automatically started by the NFS server init scripts. It is not required in NFSv4. ■ rpc.nfsd The main component to the NFS system, this is the NFS server /daemon. It works in conjunction with the Linux kernel either to load or unload the kernel module as necessary. It is, of course, still relevant in NFSv4. ■ rpc.lockd The rpc.statd daemon uses this daemon to handle lock recovery on crashed systems. It also allows NFS clients to lock files on the server. The nfslock service is no longer used in NFSv4. ■ rpc.idmapd This is the NFSv4 ID name-mapping daemon. It provides this functionality to the NFSv4 kernel client and server by translating user and group IDs to names and vice versa. ■ rpc.svcgssd This is the server-side rpcsec_gss daemon. The rpcsec_gss protocol allows the use of the gss-api generic security API to provide advanced security in NFSv4. ■ rpc.gssd This provides the client-side transport mechanism for the authentication mechanism in NFSv4 and higher. Write any 7	1 X 7	7	7
XIV	we will assume that the printer has built-in networking capabilities and that it has an IP address of 192.168.1.200 listening on port 9100—that is, the device URI will be socket://192.168.1.200:9100. 1. While logged into the system as the superuser, launch any virtual terminal and list	7	7	7

	the printer queues that you currently have configured for your system. Use the lpstat utility: [root@server ~]# lpstat -a Imagine-printer accepting requests since Sat 24....PST 2. Use the lpinfo command to get a list of printer models and drivers supported by the CUPS server. For our sample scenario, we are interested in the Fuji Xerox DocuPrint series of printers, so we'll pipe the long output of the command to the grep command to narrow down the list: [root@server ~]# lpinfo -m | grep -i fuji 3. Now issue the lpadmin command to add the printer. (Please note that the entire command is long because of all its options, so it spans several lines in this sample listing.) Type the following: [root@server ~]# lpadmin -p "Imagine-printer-number-2" -E \ -v socket://192.168.1.200 \ -m "lsb/usr/cupsfilters/Fuji_Xerox-DocuPrint_CM305_df-PDF.ppd" \ -D "You only need to imagine to print here" \ -L "Building 3"4. 4. Use the lpstat command again to list all the printers that are present: [root@fedora-server ~]# lpstat -a Imagine-printer accepting requests since Sun 31 ... 8 PM EDT Imagine-printer-number-2 accepting requests since Sun 31 … 6 PM EDT 5. You can also view the printer you just added on the CUPS web interface. Point your web browser to this URL: http://localhost:631/printers			