Linux

**Q:** explain yum and apt ?

GPT 4 Turbo–

yum (Yellowdog Updater, Modified) and apt (Advanced Package Tool) are both command-line package management utilities that help users to install, update, and remove software packages. While they serve similar functions, they are used in different Linux distributions and have some differences in functionality and command syntax.

**YUM**

* **Distribution**: Primarily used in Red Hat-based distributions such as CentOS, Fedora, and RHEL (Red Hat Enterprise Linux).
* **Package Format**: Deals with .rpm (Red Hat Package Manager) files.
* **Repository Configuration**: Uses .repo files stored in /etc/yum.repos.d/.
* **Basic Commands**:
  + yum install [package]: Install a package.
  + yum remove [package]: Remove a package.
  + yum update [package]: Update a specific package. If no package is specified, all packages are updated.
  + yum search [keyword]: Search for a package by keyword.
  + yum info [package]: Display detailed information about a package.

**APT**

* **Distribution**: Primarily used in Debian-based distributions such as Ubuntu, Debian, and Linux Mint.
* **Package Format**: Handles .deb packages.
* **Repository Configuration**: Uses sources listed in /etc/apt/sources.list, as well as additional .list files in /etc/apt/sources.list.d/.
* **Basic Commands**:
  + apt install [package]: Install a package.
  + apt remove [package]: Remove a package.
  + apt update: Update the list of available packages.
  + apt upgrade: Upgrade all upgradable packages.
  + apt search [keyword]: Search in package descriptions for a keyword.
  + apt show [package]: Display detailed information about a package.

**Key Differences**

* **Dependency Resolution**: Both handle dependency resolution but may do so differently due to the different packaging systems.
* **Performance**: yum uses a more memory-intensive database to keep track of information, which can make it slower in some cases compared to apt.
* **Interface**: While both offer command-line interfaces, their commands and options differ, as shown above.

In summary, while yum and apt perform many of the same roles, they do so in different environments and with different packages, syntax, and systems.

------------------------------------------------------------------------------------------------------------------  
**Q:** can we use yum in ubuntu ?

GPT 4 Turbo–

No, you typically cannot use yum directly on Ubuntu as it is designed for managing .rpm packages and is used primarily with Red Hat-based Linux distributions like CentOS, Fedora, and RHEL. Ubuntu, being a Debian-based distribution, uses apt for package management, which handles .deb packages.

If you need to use RPM packages in Ubuntu, there are other tools designed for that purpose, such as alien, which can convert RPM packages to the DEB format. However, it's generally recommended to find and use the DEB versions of software packages when using Ubuntu, as these are specifically packaged and tested for compatibility with Debian-based systems.

YUM | OS: Amazon Linux | Commands

Update the System

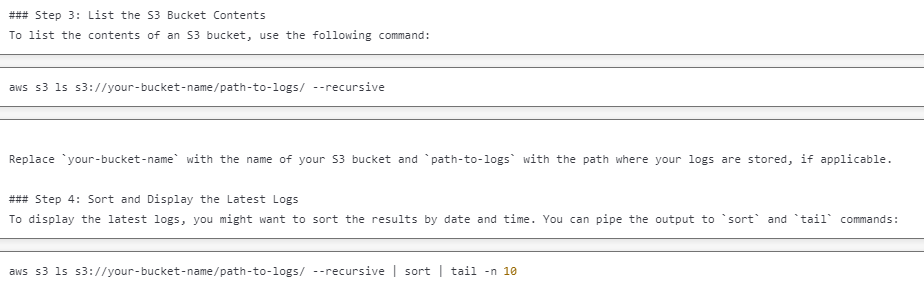
* sudo yum update -y

YUM | AWS Configure

* To work with AWS in Amazon Linux, (AWS Configure)

  
  
YUM | AWS S3

* To view latest logs in s3 bucket



* To copy nginx.conf file from current directory to s3 bucket

aws s3 cp nginx.conf s3://a208085-triva-pradeepa-websdk-poc/

* **Checking All Processes in Amazon Linux EC2**

There are several commands you can use to view and monitor processes running on your Amazon Linux EC2 instance:

**Basic Process Commands**

**ps command** - Shows currently running processes

* ps aux # Shows all processes for all users in detailed format
* ps -ef *# Another format showing all processes*

**top command** - Interactive real-time process viewer

* top *# Press 'q' to exit*

**htop command** - Enhanced version of top (may need installation)

* sudo yum install htop # Install htop if not available
* htop *# Interactive process viewer with more features*

**Specific Process Filtering**

Find processes by name:

* pgrep [process\_name]
* ps aux | grep [process\_name]

List processes sorted by resource usage:

* ps aux --sort=-%cpu *# Sort by CPU usage*
* ps aux --sort=-%mem *# Sort by memory usage*

**System Monitoring Tools**

For system activity information:

* sudo yum install sysstat *# Install if not available*
* sar -u *# CPU usage*
* vmstat 1 5 *# System stats every 1 second, 5 times*

Process tree view:

* pstree *# Visual tree of processes*

These commands should work on all Amazon Linux versions. The output provides information about process IDs, CPU/memory usage, running time, and the commands that started the processes.

* how to check node js processes
* **Checking Node.js Processes in Linux**
* There are several ways to identify and monitor Node.js processes running on your system:

**Basic Methods**

**Using ps with grep**:

* ps aux | grep node

*# or*

* ps aux | grep nodejs

**Using pgrep**:

* pgrep -l node

**Node.js Specific Methods**

**PM2** (if you're using PM2 to manage Node.js applications):

*# Install PM2 if not already installed*

* npm install pm2 -g

*# List all PM2 managed processes*

* pm2 list

*# Get detailed information about a specific process*

* pm2 show [app-name or id]

*# Monitor processes in real-time*

* pm2 monit

**Forever** (if using Forever):

*# List all Forever managed processes*

* forever list

**Advanced Monitoring**

**Using lsof to check Node.js processes and their ports**:

* sudo lsof -i -P | grep node

**Finding Node.js processes using specific port**:

* sudo lsof -i :3000 *# Replace 3000 with your port number*

**Detailed process information**:

*# Get process ID first*

pgrep node

*# Then check detailed info*

* ps -p [PID] -o pid,vsz,cmd,start\_time

**Monitor Node.js processes with htop**:

* htop -p $(pgrep -d ',' node)
* These commands will help you identify running Node.js processes, their resource usage, and other relevant information for monitoring and management.

how to start node js application in ec2 amazon linux

**Starting Node.js Applications on Amazon Linux EC2**

* Here are several methods to start and manage Node.js applications on your Amazon Linux EC2 instance:

**Method 1: Direct Execution**

*# Navigate to your application directory*

* cd /path/to/your/app

*# Start your application*

* node app.js
* This method is simple but not recommended for production as the application will stop if you close the terminal or if the process crashes.
* **Method 2: Using PM2 (Recommended for Production)**
* PM2 is a process manager for Node.js applications with built-in load balancing.
* *1# Install PM2 globally*
* *2*npm install pm2 -g
* *3*
* *4# Start your application*
* *5*pm2 start app.js --name "my-app"
* *6*
* *7# Make PM2 start on system boot*
* *8*pm2 startup
* *9*pm2 save
* *10*
* *11# Other useful PM2 commands*
* *12*pm2 list *# List all applications*
* *13*pm2 stop my-app *# Stop application*
* *14*pm2 restart my-app *# Restart application*
* *15*pm2 logs my-app *# View logs*
* *16*pm2 monit *# Monitor CPU/Memory usage*
* **Method 3: Using Systemd**
* Create a systemd service file:
* *1*sudo nano /etc/systemd/system/my-nodejs-app.service
* Add the following content:
* *1*[Unit]
* *2*Description=My Node.js Application
* *3*After=network.target
* *4*
* *5*[Service]
* *6*Environment=NODE\_ENV=production
* *7*Type=simple
* *8*User=ec2-user
* *9*WorkingDirectory=/path/to/your/app
* *10*ExecStart=/usr/bin/node /path/to/your/app/app.js
* *11*Restart=on-failure
* *12*
* *13*[Install]
* *14*WantedBy=multi-user.target
* Then enable and start the service:
* *1*sudo systemctl enable my-nodejs-app
* *2*sudo systemctl start my-nodejs-app
* *3*sudo systemctl status my-nodejs-app
* **Method 4: Using nohup (Simple Background Process)**
* *1# Start application in background*
* *2*nohup node app.js > output.log 2>&1 &
* *3*
* *4# Find the process ID*
* *5*echo $!
* **Additional Setup Tips**

1. **Set up environment variables**:
2. *1# In your .bashrc or .bash\_profile*

* *2*export NODE\_ENV=production

1. **Configure your application to listen on the correct port**:
2. *1// In your Node.js app*
3. *2***const** port = process.env.PORT || 3000;

* *3*app.listen(port);

1. **Set up a reverse proxy with Nginx** (recommended for production):
2. *1*sudo yum install nginx

* *2*sudo nano /etc/nginx/conf.d/my-nodejs-app.conf
* Add configuration:
* *1*server {
* *2* listen 80;
* *3* server\_name your\_domain\_or\_ip;
* *4*
* *5* location / {
* *6* proxy\_pass http://localhost:3000;
* *7* proxy\_http\_version 1.1;
* *8* proxy\_set\_header Upgrade $http\_upgrade;
* *9* proxy\_set\_header Connection 'upgrade';
* *10* proxy\_set\_header Host $host;
* *11* proxy\_cache\_bypass $http\_upgrade;
* *12* }
* *13*}
* Then restart Nginx:
* *1*sudo systemctl restart nginx
* For production environments, PM2 or systemd are recommended as they provide automatic restarts and better process management.

sudo find / -name "app.js"