**1. What are the fundamental differences between DevOps & Agile?**

The differences between the two are listed down in the table below.

|  |  |  |
| --- | --- | --- |
| **DevOps vs Agile** | | |
| **Features** | **DevOps** | **Agile** |
| **Agility** | Agility in both Development & Operations | Agility in only Development |
| **Processes/ Practices** | Involves processes such as CI, CD, CT, etc. | Involves practices such as Agile Scrum, Agile Kanban, etc. |
| **Key Focus Area** | Timeliness & quality have equal priority | Timeliness is the main priority |
| **Release Cycles/ Development Sprints** | Smaller release cycles with immediate feedback | Smaller release cycles |
| **Source of Feedback** | Feedback is from self (Monitoring tools) | Feedback is from customers |
| **Scope of Work** | Agility & need for Automation | Agility only |

**Q2. What is the need for DevOps?**

According to me, this answer should start by explaining the general market trend. Instead of releasing big sets of features, companies are trying to see if small features can be transported to their customers through a series of release trains. This has many advantages like quick feedback from customers, better quality of software etc. which in turn leads to high customer satisfaction. To achieve this, companies are required to:

1. Increase deployment frequency
2. Lower failure rate of new releases
3. Shortened lead time between fixes
4. Faster mean time to recovery in the event of new release crashing

DevOps fulfills all these requirements and helps in achieving seamless software delivery. You can give examples of companies like Etsy, Google and Amazon which have adopted [DevOps to achieve levels of performance](https://www.edureka.co/blog/why-devops-improves-organization-performance/) that were unthinkable even five years ago. They are doing tens, hundreds or even thousands of code deployments per day while delivering world class stability, reliability and security.

If I have to test your knowledge on DevOps, you should know the difference between Agile and DevOps. The next question is directed towards that.

**Q3. How is DevOps different from Agile / SDLC?**

I would advise you to go with the below explanation:

Agile is a set of values and principles about how to produce i.e. develop software. Example: if you have some ideas and you want to turn those ideas into working software, you can use the Agile values and principles as a way to do that. But, that software might only be working on a developer’s laptop or in a test environment. You want a way to quickly, easily and repeatably move that software into production infrastructure, in a safe and simple way. To do that you need DevOps tools and techniques.

You can summarize by saying Agile software development methodology focuses on the development of software but DevOps on the other hand is responsible for development as well as deployment of the software in the safest and most reliable way possible. Here’s a blog that will give you more information on the [evolution of DevOps](https://www.edureka.co/blog/what-is-devops/).

Now remember, you have included DevOps tools in your previous answer so be prepared to answer some questions related to that.

**Q4. Which are the top DevOps tools? Which tools have you worked on?**

The most popular DevOps tools are mentioned below:

* Git : Version Control System tool
* Jenkins : Continuous Integration tool
* Selenium : Continuous Testing tool
* Puppet, Chef, Ansible : Configuration Management and Deployment tools
* Nagios : Continuous Monitoring tool
* Docker : Containerization tool

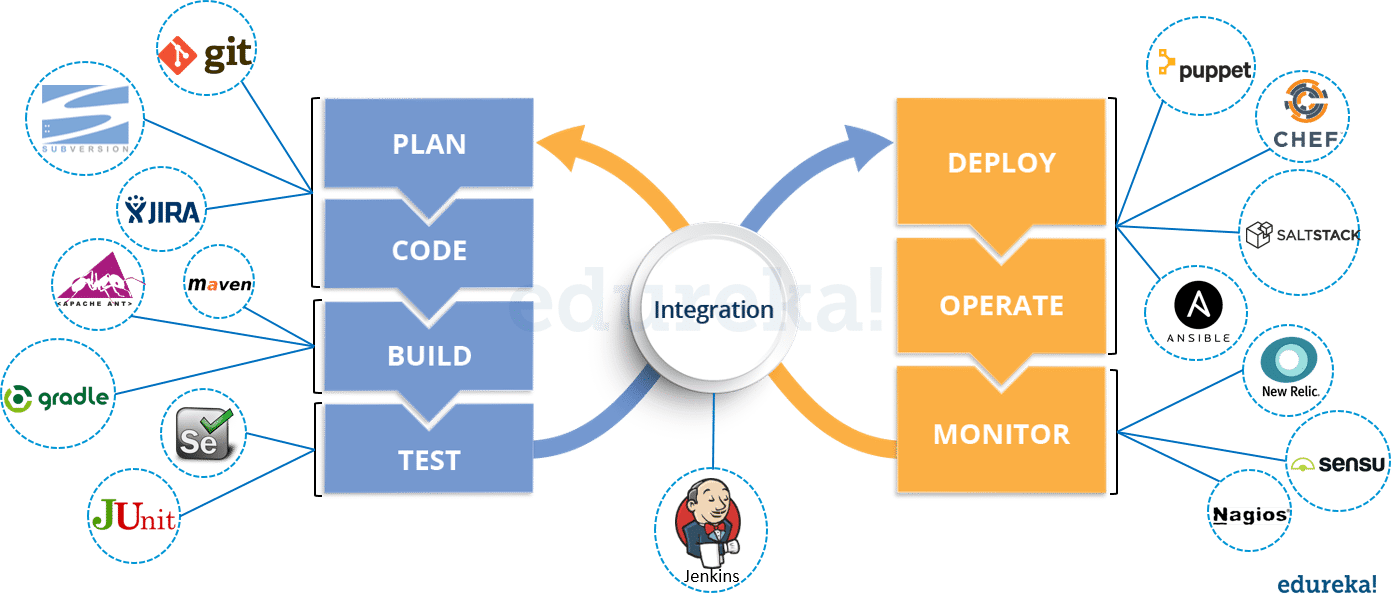
You can also mention any other tool if you want, but make sure you include the above tools in your answer.  
The second part of the answer has two possibilities:

1. If you have experience with all the above tools then you can say that I have worked on all these tools for developing good quality software and deploying those softwares easily, frequently, and reliably.
2. If you have experience only with some of the above tools then mention those tools and say that I have specialization in these tools and have an overview about the rest of the tools.

**Q5. How do all these tools work together?**

Given below is a generic logical flow where everything gets automated for seamless delivery. However, this flow may vary from organization to organization as per the requirement.

1. Developers develop the code and this source code is managed by Version Control System tools like Git etc.
2. Developers send this code to the Git repository and any changes made in the code is committed to this Repository.
3. Jenkins pulls this code from the repository using the Git plugin and build it using tools like Ant or Maven.
4. Configuration management tools like puppet deploys & provisions testing environment and then Jenkins releases this code on the test environment on which testing is done using tools like selenium.
5. Once the code is tested, Jenkins send it for deployment on the production server (even production server is provisioned & maintained by tools like puppet).
6. After deployment It is continuously monitored by tools like Nagios.
7. Docker containers provides testing environment to test the build features.



**Q6. What are the advantages of DevOps?**

For this answer, you can use your past experience and explain how DevOps helped you in your previous job. If you don’t have any such experience, then you can mention the below advantages.

Technical benefits:

* Continuous software delivery
* Less complex problems to fix
* Faster resolution of problems

Business benefits:

* Faster delivery of features
* More stable operating environments
* More time available to add value (rather than fix/maintain)

**Q7. What is the most important thing DevOps helps us achieve?**

According to me, the most important thing that DevOps helps us achieve is to get the changes into production as quickly as possible while minimizing risks in software quality assurance and compliance. This is the primary objective of DevOps. Learn more in this [DevOps tutorial](https://www.edureka.co/blog/devops-tutorial) blog.  
However, you can add many other positive effects of DevOps. For example, clearer communication and better working relationships between teams i.e. both the Ops team and Dev team collaborate together to deliver good quality software which in turn leads to higher customer satisfaction.

**Q8. Explain with a use case where DevOps can be used in industry/ real-life.**

There are many industries that are using DevOps so you can mention any of those use cases, you can also refer the below example:  
Etsy is a peer-to-peer e-commerce website focused on handmade or vintage items and supplies, as well as unique factory-manufactured items. Etsy struggled with slow, painful site updates that frequently caused the site to go down. It affected sales for millions of Etsy’s users who sold goods through online market place and risked driving them to the competitor.  
With the help of a new technical management team, Etsy transitioned from its waterfall model, which produced four-hour full-site deployments twice weekly, to a more agile approach. Today, it has a fully automated deployment pipeline, and its continuous delivery practices have reportedly resulted in more than 50 deployments a day with fewer disruptions.

**Q9. Explain your understanding and expertise on both the software development side and the technical operations side of an organization you have worked with in the past.**

For this answer, share your past experience and try to explain how flexible you were in your previous job. You can refer the below example:  
DevOps engineers almost always work in a 24/7 business-critical online environment. I was adaptable to on-call duties and was available to take up real-time, live-system responsibility. I successfully automated processes to support continuous software deployments. I have experience with public/private clouds, tools like Chef or Puppet, scripting and automation with tools like Python and PHP, and a background in Agile.

**Q10. What are the anti-patterns of DevOps?**

A pattern is common usage usually followed. If a pattern commonly adopted by others does not work for your organization and you continue to blindly follow it, you are essentially adopting an anti-pattern. There are myths about DevOps. Some of them include:

* DevOps is a process
* Agile equals DevOps?
* We need a separate DevOps group
* Devops will solve all our problems
* DevOps means Developers Managing Production
* DevOps is Development-driven release management
  1. DevOps is not development driven.
  2. DevOps is not IT Operations driven.
* We can’t do DevOps – We’re Unique
* We can’t do DevOps – We’ve got the wrong people

### ****Q1. What is Version control?****

This is probably the easiest question you will face in the interview. My suggestion is to first give a definition of Version control. It is a system that records changes to a file or set of files over time so that you can recall specific versions later. Version control systems consist of a central shared repository where teammates can commit changes to a file or set of file. Then you can mention the uses of version control.

Version control allows you to:

* Revert files back to a previous state.
* Revert the entire project back to a previous state.
* Compare changes over time.
* See who last modified something that might be causing a problem.
* Who introduced an issue and when.

## **Q2. What are the benefits of using version control?**

I will suggest you to include the following advantages of version control:

1. With Version Control System (VCS), all the team members are allowed to work freely on any file at any time. VCS will later allow you to merge all the changes into a common version.
2. All the past versions and variants are neatly packed up inside the VCS. When you need it, you can request any version at any time and you’ll have a snapshot of the complete project right at hand.
3. Every time you save a new version of your project, your VCS requires you to provide a short description of what was changed. Additionally, you can see what exactly was changed in the file’s content. This allows you to know who has made what change in the project.
4. A distributed VCS like Git allows all the team members to have complete history of the project so if there is a breakdown in the central server you can use any of your teammate’s local Git repository.

### ****Q3. Describe branching strategies you have used.****

This question is asked to test your branching experience so tell them about how you have used branching in your previous job and what purpose does it serves, you can refer the below points:

* Feature branching  
  A feature branch model keeps all of the changes for a particular feature inside of a branch. When the feature is fully tested and validated by automated tests, the branch is then merged into master.
* Task branching  
  In this model each task is implemented on its own branch with the task key included in the branch name. It is easy to see which code implements which task, just look for the task key in the branch name.
* Release branching  
  Once the develop branch has acquired enough features for a release, you can clone that branch to form a Release branch. Creating this branch starts the next release cycle, so no new features can be added after this point, only bug fixes, documentation generation, and other release-oriented tasks should go in this branch. Once it is ready to ship, the release gets merged into master and tagged with a version number. In addition, it should be merged back into develop branch, which may have progressed since the release was initiated.

In the end tell them that branching strategies varies from one organization to another, so I know basic branching operations like delete, merge, checking out a branch etc.

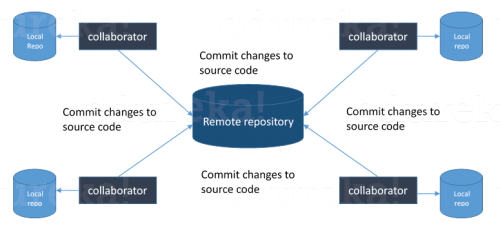
### ****Q4. Which VCS tool you are comfortable with?****

You can just mention the VCS tool that you have worked on like this: “I have worked on Git and one major advantage it has over other VCS tools like SVN is that it is a distributed version control system.”  
Distributed VCS tools do not necessarily rely on a central server to store all the versions of a project’s files. Instead, every developer “clones” a copy of a repository and has the full history of the project on their own hard drive.

### ****Q5. What is Git?****

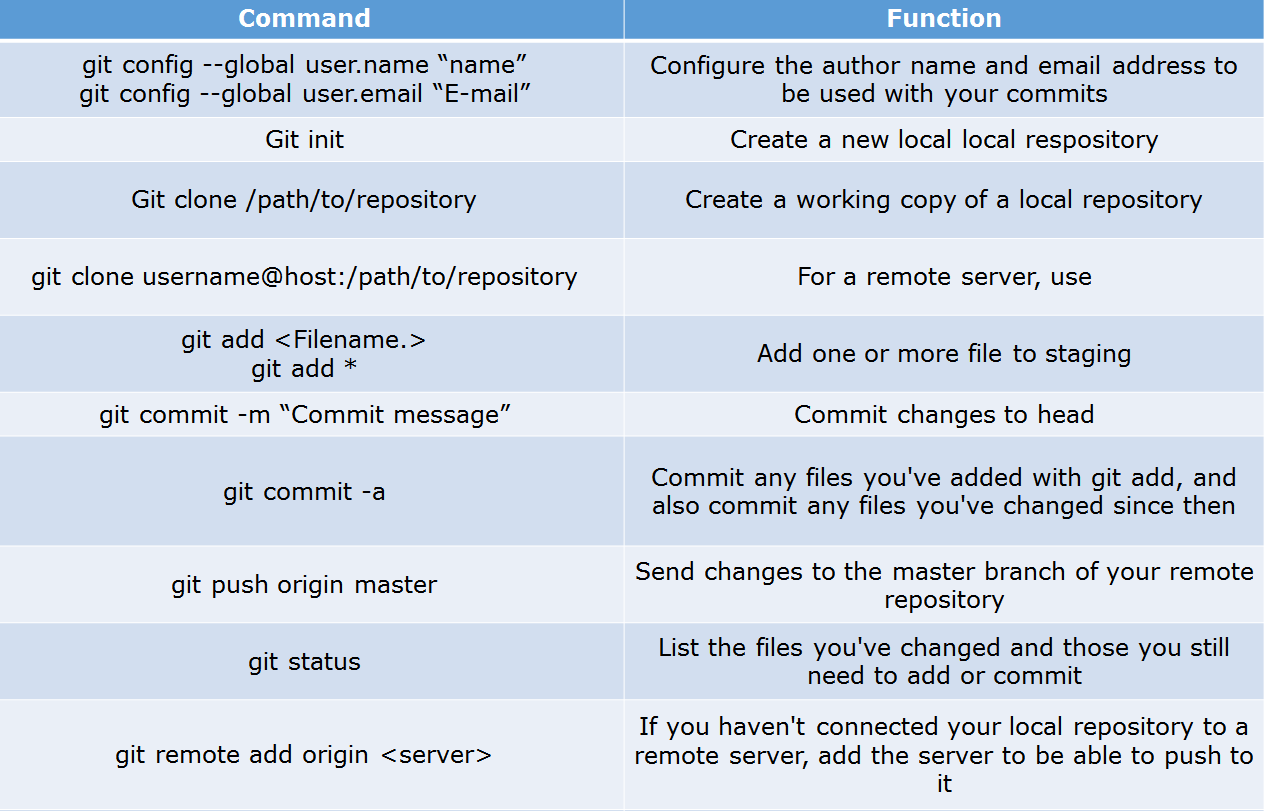
I will suggest that you attempt this question by first explaining about the architecture of git as shown in the below diagram. You can refer to the explanation given below:

* Git is a Distributed Version Control system (DVCS). It can track changes to a file and allows you to revert back to any particular change.
* Its distributed architecture provides many advantages over other Version Control Systems (VCS) like SVN one major advantage is that it does not rely on a central server to store all the versions of a project’s files. Instead, every developer “clones” a copy of a repository I have shown in the diagram below with “Local repository” and has the full history of the project on his hard drive so that when there is a server outage, all you need for recovery is one of your teammate’s local Git repository.
* There is a central cloud repository as well where developers can commit changes and share it with other teammates as you can see in the diagram where all collaborators are commiting changes “Remote repository”.



### ****Q6. Explain some basic Git commands?****

Below are some basic Git commands:



### ****Q7. In Git how do you revert a commit that has already been pushed and made public?****

There can be two answers to this question so make sure that you include both because any of the below options can be used depending on the situation:

* Remove or fix the bad file in a new commit and push it to the remote repository. This is the most natural way to fix an error. Once you have made necessary changes to the file, commit it to the remote repository for that I will use  
  **git commit -m “commit message”**
* Create a new commit that undoes all changes that were made in the bad commit.to do this I will use a command  
  **git revert <name of bad commit>**

### ****Q8. How do you squash last N commits into a single commit?****

There are two options to squash last N commits into a single commit. Include both of the below mentioned options in your answer:

* If you want to write the new commit message from scratch use the following command  
  **git reset –soft HEAD~N &&**  
  **git commit**
* If you want to start editing the new commit message with a concatenation of the existing commit messages then you need to extract those messages and pass them to Git commit for that I will use  
  **git reset –soft HEAD~N &&**  
  **git commit –edit -m”$(git log –format=%B –reverse .HEAD@{N})”**

### ****Q9. What is Git bisect? How can you use it to determine the source of a (regression) bug?****

I will suggest you to first give a small definition of Git bisect, Git bisect is used to find the commit that introduced a bug by using binary search. Command for Git bisect is  
**git bisect <subcommand> <options>**Now since you have mentioned the command above, explain what this command will do, This command uses a binary search algorithm to find which commit in your project’s history introduced a bug. You use it by first telling it a “bad” commit that is known to contain the bug, and a “good” commit that is known to be before the bug was introduced. Then Git bisect picks a commit between those two endpoints and asks you whether the selected commit is “good” or “bad”. It continues narrowing down the range until it finds the exact commit that introduced the change.

### ****Q10. What is Git rebase and how can it be used to resolve conflicts in a feature branch before merge?****

According to me, you should start by saying git rebase is a command which will merge another branch into the branch where you are currently working, and move all of the local commits that are ahead of the rebased branch to the top of the history on that branch.  
Now once you have defined Git rebase time for an example to show how it can be used to resolve conflicts in a feature branch before merge, if a feature branch was created from master, and since then the master branch has received new commits, Git rebase can be used to move the feature branch to the tip of master.  
The command effectively will replay the changes made in the feature branch at the tip of master, allowing conflicts to be resolved in the process. When done with care, this will allow the feature branch to be merged into master with relative ease and sometimes as a simple fast-forward operation.

### ****Q11. How do you configure a Git repository to run code sanity checking tools right before making commits, and preventing them if the test fails?****

I will suggest you to first give a small introduction to sanity checking, A sanity or smoke testdetermines whether it is possible and reasonable to continue testing.  
Now explain how to achieve this, this can be done with a simple script related to the pre-commit hook of the repository. The pre-commit hook is triggered right before a commit is made, even before you are required to enter a commit message. In this script one can run other tools, such as linters and perform sanity checks on the changes being committed into the repository.  
Finally give an example, you can refer the below script:  
**#!/bin/sh  
files=$(git diff –cached –name-only –diff-filter=ACM | grep ‘.go$’)  
if [ -z files ]; then  
exit 0  
fi  
unfmtd=$(gofmt -l $files)  
if [ -z unfmtd ]; then  
exit 0  
fi  
echo “Some .go files are not fmt’d”  
exit 1**This script checks to see if any .go file that is about to be committed needs to be passed through the standard Go source code formatting tool gofmt. By exiting with a non-zero status, the script effectively prevents the commit from being applied to the repository.

### ****Q12. How do you find a list of files that has changed in a particular commit?****

For this answer instead of just telling the command, explain what exactly this command will do so you can say that, To get a list files that has changed in a particular commit use command  
**git diff-tree -r {hash}**  
Given the commit hash, this will list all the files that were changed or added in that commit. The -r flag makes the command list individual files, rather than collapsing them into root directory names only.  
You can also include the below mention point although it is totally optional but will help in impressing the interviewer.  
The output will also include some extra information, which can be easily suppressed by including two flags:  
**git diff-tree –no-commit-id –name-only -r {hash}**  
Here –no-commit-id will suppress the commit hashes from appearing in the output, and –name-only will only print the file names, instead of their paths.

### ****Q13. How do you setup a script to run every time a repository receives new commits through push?****

There are three ways to configure a script to run every time a repository receives new commits through push, one needs to define either a pre-receive, update, or a post-receive hook depending on when exactly the script needs to be triggered.

* Pre-receive hook in the destination repository is invoked when commits are pushed to it. Any script bound to this hook will be executed before any references are updated. This is a useful hook to run scripts that help enforce development policies.
* Update hook works in a similar manner to pre-receive hook, and is also triggered before any updates are actually made. However, the update hook is called once for every commit that has been pushed to the destination repository.
* Finally, post-receive hook in the repository is invoked after the updates have been accepted into the destination repository. This is an ideal place to configure simple deployment scripts, invoke some continuous integration systems, dispatch notification emails to repository maintainers, etc.

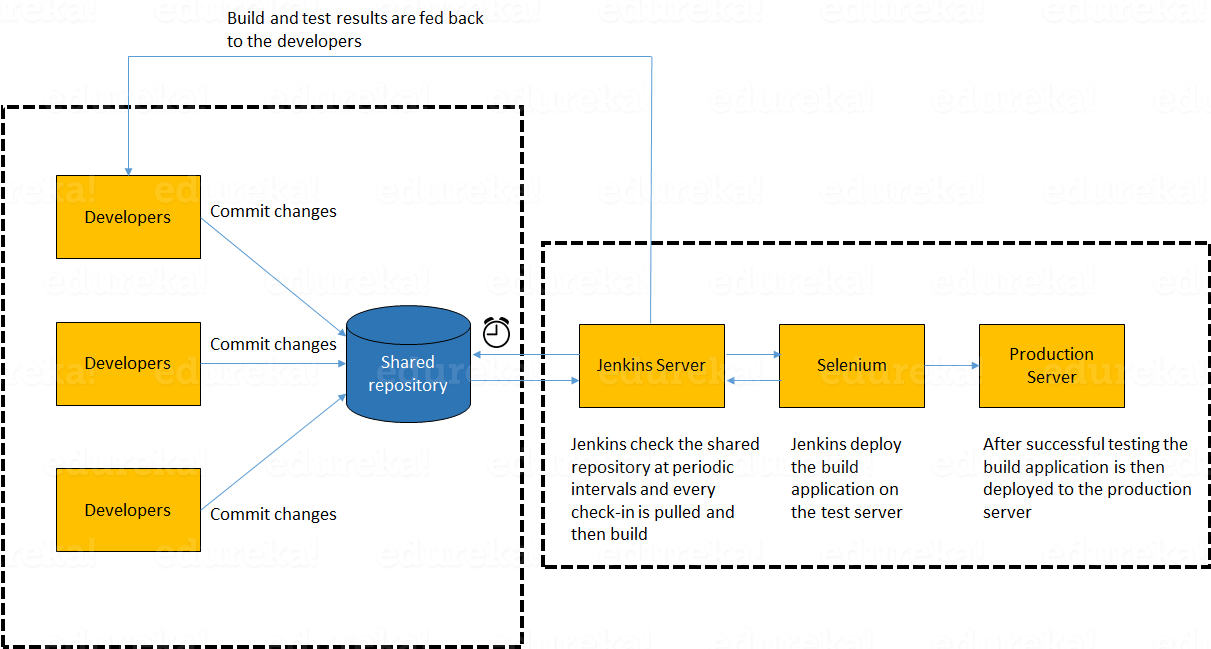
Hooks are local to every Git repository and are not versioned. Scripts can either be created within the hooks directory inside the “.git” directory, or they can be created elsewhere and links to those scripts can be placed within the directory.

### ****Q14. How will you know in Git if a branch has already been merged into master?****

I will suggest you to include both the below mentioned commands:  
git branch –merged lists the branches that have been merged into the current branch.  
git branch –no-merged lists the branches that have not been merged.

**Q1. What is meant by Continuous Integration?**

I will advise you to begin this answer by giving a small definition of Continuous Integration (CI). It is a development practice that requires developers to integrate code into a shared repository several times a day. Each check-in is then verified by an automated build, allowing teams to detect problems early.  
I suggest that you explain how you have implemented it in your previous job. You can refer the below given example:



In the diagram shown above:

1. Developers check out code into their private workspaces.
2. When they are done with it they commit the changes to the shared repository (Version Control Repository).
3. The CI server monitors the repository and checks out changes when they occur.
4. The CI server then pulls these changes and builds the system and also runs unit and integration tests.
5. The CI server will now inform the team of the successful build.
6. If the build or tests fails, the CI server will alert the team.
7. The team will try to fix the issue at the earliest opportunity.
8. This process keeps on repeating.

**Q2. Why do you need a Continuous Integration of Dev & Testing?**

For this answer, you should focus on the need of Continuous Integration. My suggestion would be to mention the below explanation in your answer:  
Continuous Integration of Dev and Testing improves the quality of software, and reduces the time taken to deliver it, by replacing the traditional practice of testing after completing all development. It allows Dev team to easily detect and locate problems early because developers need to integrate code into a shared repository several times a day (more frequently). Each check-in is then automatically tested.

**Q3. What are the success factors for Continuous Integration?**

Here you have to mention the requirements for Continuous Integration. You could include the following points in your answer:

* Maintain a code repository
* Automate the build
* Make the build self-testing
* Everyone commits to the baseline every day
* Every commit (to baseline) should be built
* Keep the build fast
* Test in a clone of the production environment
* Make it easy to get the latest deliverables
* Everyone can see the results of the latest build
* Automate deployment

**Q4. Explain how you can move or copy Jenkins from one server to another?**

I will approach this task by copying the jobs directory from the old server to the new one. There are multiple ways to do that;  I have mentioned them below:  
You can:

* Move a job from one installation of Jenkins to another by simply copying the corresponding job directory.
* Make a copy of an existing job by making a clone of a job directory by a different name.
* Rename an existing job by renaming a directory. Note that if you change a job name you will need to change any other job that tries to call the renamed job.

**Q5. Explain how can create a backup and copy files in Jenkins?**

Answer to this question is really direct. To create a backup, all you need to do is to periodically back up your JENKINS\_HOME directory. This contains all of your build jobs configurations, your slave node configurations, and your build history. To create a back-up of your Jenkins setup, just copy this directory. You can also copy a job directory to clone or replicate a job or rename the directory.

**Q6. Explain how you can setup Jenkins job?**

My approach to this answer will be to first mention how to create Jenkins job. Go to Jenkins top page, select “New Job”, then choose “Build a free-style software project”.  
Then you can tell the elements of this freestyle job:

* Optional SCM, such as CVS or Subversion where your source code resides.
* Optional triggers to control when Jenkins will perform builds.
* Some sort of build script that performs the build (ant, maven, shell script, batch file, etc.) where the real work happens.
* Optional steps to collect information out of the build, such as archiving the artifacts and/or recording javadoc and test results.
* Optional steps to notify other people/systems with the build result, such as sending e-mails, IMs, updating issue tracker, etc..

**Q7. Mention some of the useful plugins in Jenkins.**

Below, I have mentioned some important Plugins:

* Maven 2 project
* Amazon EC2
* HTML publisher
* Copy artifact
* Join
* Green Balls

These Plugins, I feel are the most useful plugins. If you want to include any other Plugin that is not mentioned above, you can add them as well. But, make sure you first mention the above stated plugins and then add your own.

**Q8. How will you secure Jenkins?**

The way I secure Jenkins is mentioned below. If you have any other way of doing it, please mention it in the comments section below:

* Ensure global security is on.
* Ensure that Jenkins is integrated with my company’s user directory with appropriate plugin.
* Ensure that matrix/Project matrix is enabled to fine tune access.
* Automate the process of setting rights/privileges in Jenkins with custom version controlled script.
* Limit physical access to Jenkins data/folders.
* Periodically run security audits on same.

## **Continuous Testing Interview Questions:**

Now let’s move on to the Continuous Testing questions.

### ****Q1. What is Continuous Testing?****

I will advise you to follow the below mentioned explanation:  
Continuous Testing is the process of executing automated tests as part of the software delivery pipeline to obtain immediate feedback on the business risks associated with in the latest build. In this way, each build is tested continuously, allowing Development teams to get fast feedback so that they can prevent those problems from progressing to the next stage of Software delivery life-cycle. This dramatically speeds up a developer’s workflow as there’s no need to manually rebuild the project and re-run all tests after making changes.

### ****Q2. What is Automation Testing?****

Automation testing or Test Automation is a process of automating the manual process to test the application/system under test. Automation testing involves use of separate testing tools which lets you create test scripts which can be executed repeatedly and doesn’t require any manual intervention.

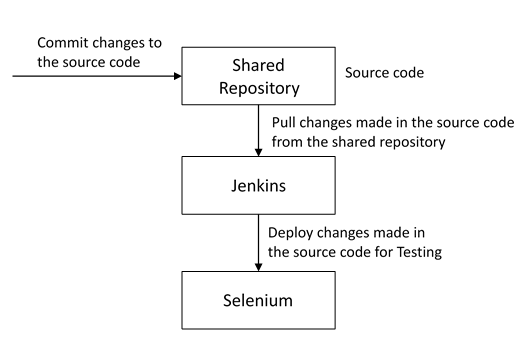
### ****Q3.**** ****What are the benefits of Automation Testing?****

I have listed down some advantages of automation testing. Include these in your answer and you can add your own experience of how Continuous Testing helped your previous company:

* Supports execution of repeated test cases
* Aids in testing a large test matrix
* Enables parallel execution
* Encourages unattended execution
* Improves accuracy thereby reducing human generated errors
* Saves time and money

### ****Q4. How to automate Testing in DevOps lifecycle?****

I have mentioned a generic flow below which you can refer to:  
In DevOps, developers are required to commit all the changes made in the source code to a shared repository. Continuous Integration tools like Jenkins will pull the code from this shared repository every time a change is made in the code and deploy it for Continuous Testing that is done by tools like Selenium as shown in the below diagram.  
In this way, any change in the code is continuously tested unlike the traditional approach.



### ****Q5. Why is Continuous Testing important for DevOps?****

You can answer this question by saying, “Continuous Testing allows any change made in the code to be tested immediately. This avoids the problems created by having “big-bang” testing left to the end of the cycle such as release delays and quality issues. In this way, Continuous Testing facilitates more frequent and good quality releases.”

### ****Q6. What are the key elements of Continuous Testing tools?****

Key elements of Continuous Testing are:

* **Risk Assessment:** It Covers risk mitigation tasks, technical debt, quality assessment and test coverage optimization to ensure the build is ready to progress toward next stage.
* **Policy Analysis:** It ensures all processes align with the organization’s evolving business and compliance demands are met.
* **Requirements Traceability:** It ensures true requirements are met and rework is not required. An object assessment is used to identify which requirements are at risk, working as expected or require further validation.
* **Advanced Analysis:** It uses automation in areas such as static code analysis, change impact analysis and scope assessment/prioritization to prevent defects in the first place and accomplishing more within each iteration.
* **Test Optimization:** It ensures tests yield accurate outcomes and provide actionable findings. Aspects include Test Data Management, Test Optimization Management and Test Maintenance
* **Service Virtualization:** It ensures access to real-world testing environments. Service visualization enables access to the virtual form of the required testing stages, cutting the waste time to test environment setup and availability.

### ****Q7. Which Testing tool are you comfortable with and what are the benefits of that tool?****

Here mention the testing tool that you have worked with and accordingly frame your answer. I have mentioned an example below:  
I have worked on Selenium to ensure high quality and more frequent releases.

Some advantages of Selenium are:

* It is free and open source
* It has a large user base and helping communities
* It has cross Browser compatibility (Firefox, chrome, Internet Explorer, Safari etc.)
* It has great platform compatibility (Windows, Mac OS, Linux etc.)
* It supports multiple programming languages (Java, C#, Ruby, Python, Pearl etc.)
* It has fresh and regular repository developments
* It supports distributed testing

### ****Q8. What are the Testing types supported by Selenium?****

Selenium supports two types of testing:  
**Regression Testing**: It is the act of retesting a product around an area where a bug was fixed.  
**Functional Testing**: It refers to the testing of software features (functional points) individually.

### ****Q9. What is Selenium IDE?****

My suggestion is to start this answer by defining Selenium IDE. It is an integrated development environment for Selenium scripts. It is implemented as a Firefox extension, and allows you to record, edit, and debug tests. Selenium IDE includes the entire Selenium Core, allowing you to easily and quickly record and play back tests in the actual environment that they will run in.  
Now include some advantages in your answer. With autocomplete support and the ability to move commands around quickly, Selenium IDE is the ideal environment for creating Selenium tests no matter what style of tests you prefer.

### ****Q10. What is the difference between Assert and Verify commands in Selenium?****

I have mentioned differences between Assert and Verify commands below:

* Assert command checks whether the given condition is true or false. Let’s say we assert whether the given element is present on the web page or not. If the condition is true, then the program control will execute the next test step. But, if the condition is false, the execution would stop and no further test would be executed.
* Verify command also checks whether the given condition is true or false. Irrespective of the condition being true or false, the program execution doesn’t halts i.e. any failure during verification would not stop the execution and all the test steps would be executed.

### ****Q11. How to launch Browser using WebDriver?****

The following syntax can be used to launch Browser:  
**WebDriver driver = new FirefoxDriver();**  
**WebDriver driver = new ChromeDriver();**  
**WebDriver driver = new InternetExplorerDriver();**

### ****Q12. When should I use Selenium Grid?****

For this answer, my suggestion would be to give a small definition of Selenium Grid. It can be used to execute same or different test scripts on multiple platforms and browsers concurrently to achieve distributed test execution. This allows testing under different environments and saving execution time remarkably.

## **onfiguration Management Interview Questions**

Now let’s check how much you know about Configuration Management.

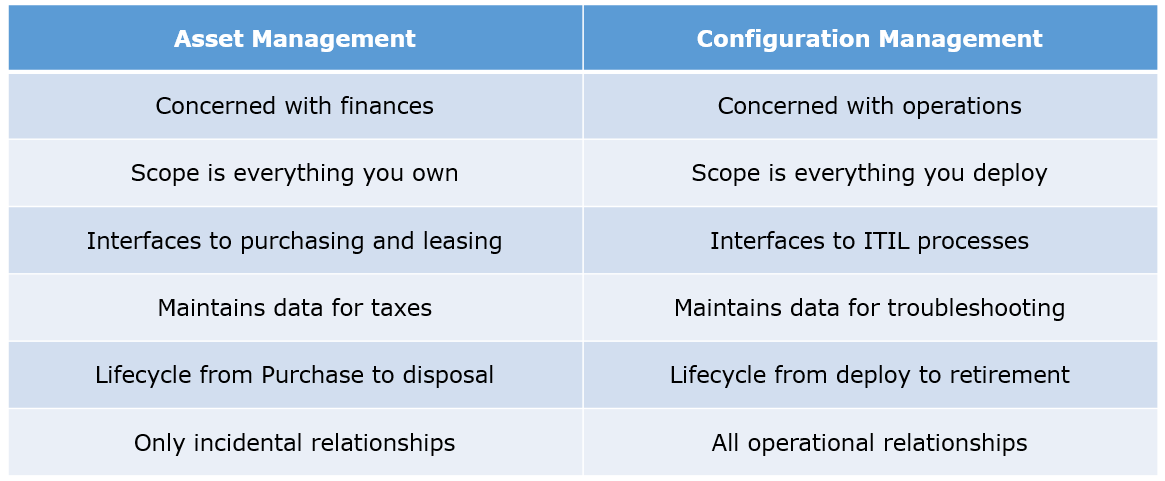
### ****Q1. What are the goals of Configuration management processes?****

The purpose of Configuration Management (CM) is to ensure the integrity of a product or system throughout its life-cycle by making the development or deployment process controllable and repeatable, therefore creating a higher quality product or system. The CM process allows orderly management of system information and system changes for purposes such as to:

* Revise capability,
* Improve performance,
* Reliability or maintainability,
* Extend life,
* Reduce cost,
* Reduce risk and
* Liability, or correct defects.

### ****Q2. What is the difference between Asset management and Configuration Management?****

Given below are few differences between Asset Management and Configuration Management:



### ****Q3. What is the difference between an Asset and a Configuration Item?****

According to me, you should first explain Asset. It has a financial value along with a depreciation rate attached to it. IT assets are just a sub-set of it. Anything and everything that has a cost and the organization uses it for its asset value calculation and related benefits in tax calculation falls under Asset Management, and such item is called an asset.  
Configuration Item on the other hand may or may not have financial values assigned to it. It will not have any depreciation linked to it. Thus, its life would not be dependent on its financial value but will depend on the time till that item becomes obsolete for the organization.

Now you can give an example that can showcase the similarity and differences between both:  
1) Similarity:  
Server – It is both an asset as well as a CI.  
2) Difference:  
Building – It is an asset but not a CI.  
Document – It is a CI but not an asset

### ****Q4. What do you understand by “Infrastructure as code”? How does it fit into the DevOps methodology? What purpose does it achieve?****

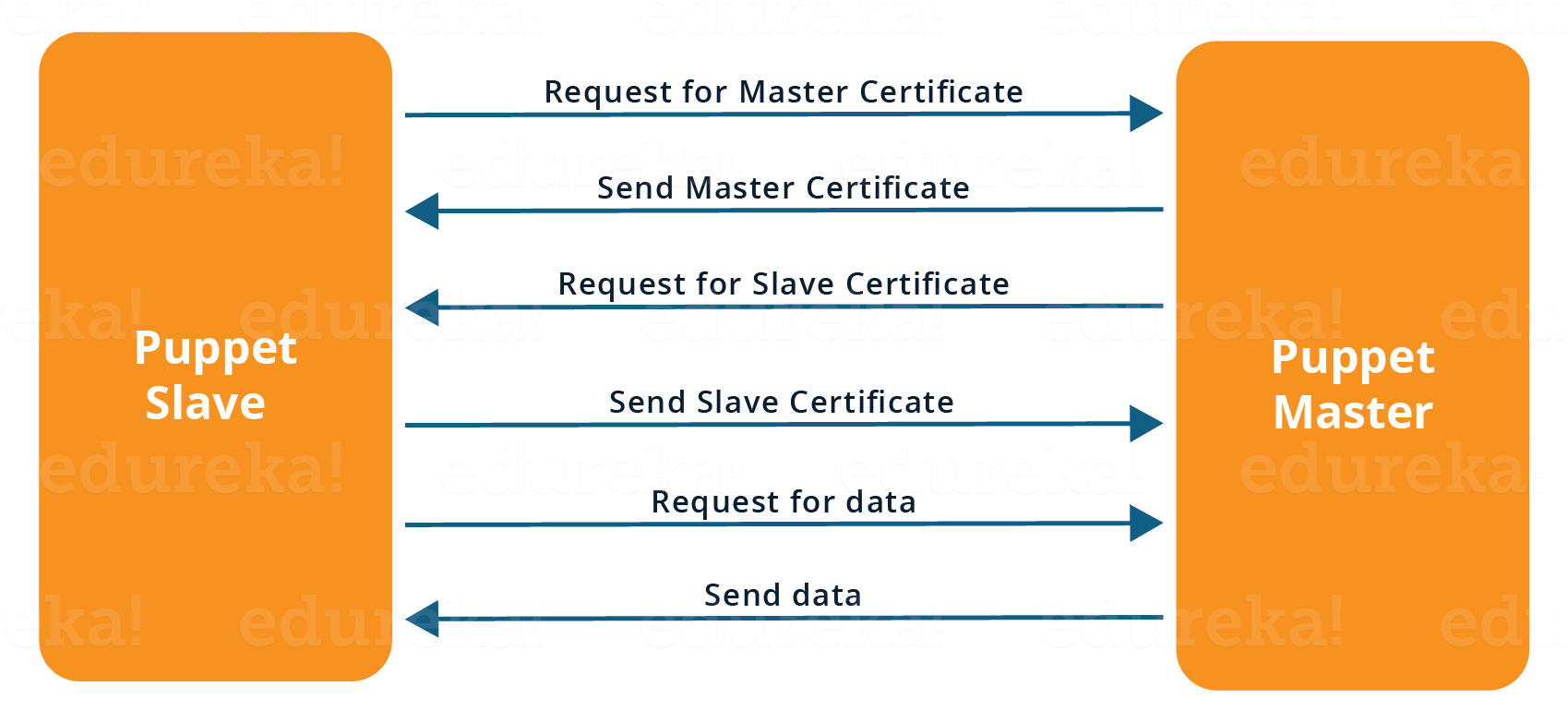
Infrastructure as Code (IAC) is a type of IT infrastructure that operations teams can use to automatically manage and provision through code, rather than using a manual process.  
Companies for faster deployments treat infrastructure like software: as code that can be managed with the DevOps tools and processes. These tools let you make infrastructure changes more easily, rapidly, safely and reliably.

### ****Q5. Which among Puppet, Chef, SaltStack and Ansible is the best Configuration Management (CM) tool? Why?****

This depends on the organization’s need so mention few points on all those tools:  
Puppet is the oldest and most mature CM tool. Puppet is a Ruby-based Configuration Management tool, but while it has some free features, much of what makes Puppet great is only available in the paid version. Organizations that don’t need a lot of extras will find Puppet useful, but those needing more customization will probably need to upgrade to the paid version.  
Chef is written in Ruby, so it can be customized by those who know the language. It also includes free features, plus it can be upgraded from open source to enterprise-level if necessary. On top of that, it’s a very flexible product.  
Ansible is a very secure option since it uses Secure Shell. It’s a simple tool to use, but it does offer a number of other services in addition to configuration management. It’s very easy to learn, so it’s perfect for those who don’t have a dedicated IT staff but still need a configuration management tool.  
SaltStack is python based open source CM tool made for larger businesses, but its learning curve is fairly low.

### ****Q6. What is Puppet?****

I will advise you to first give a small definition of Puppet. It is a Configuration Management tool which is used to automate administration tasks.  
Now you should describe its architecture and how Puppet manages its Agents. Puppet has a Master-Slave architecture in which the Slave has to first send a Certificate signing request to Master and Master has to sign that Certificate in order to establish a secure connection between Puppet Master and Puppet Slave as shown on the diagram below. Puppet Slave sends request to Puppet Master and Puppet Master then pushes configuration on Slave.  
Refer the diagram below that explains the above description.



### ****Q7. Before a client can authenticate with the Puppet Master, its certs need to be signed and accepted. How will you automate this task?****

The easiest way is to enable auto-signing in puppet.conf.  
Do mention that this is a security risk. If you still want to do this:

* Firewall your puppet master – restrict port tcp/8140 to only networks that you trust.
* Create puppet masters for each ‘trust zone’, and only include the trusted nodes in that Puppet masters manifest.
* Never use a full wildcard such as \*.

### ****Q8. Describe the most significant gain you made from automating a process through Puppet.****

For this answer, I will suggest you to explain you past experience with Puppet. you can refer the below example:  
I automated the configuration and deployment of Linux and Windows machines using Puppet. In addition to shortening the processing time from one week to 10 minutes, I used the roles and profiles pattern and documented the purpose of each module in README to ensure that others could update the module using Git. The modules I wrote are still being used, but they’ve been improved by my teammates and members of the community

### ****Q9. Which open source or community tools do you use to make Puppet more powerful?****

Over here, you need to mention the tools and how you have used those tools to make Puppet more powerful. Below is one example for your reference:  
Changes and requests are ticketed through Jira and we manage requests through an internal process. Then, we use Git and Puppet’s Code Manager app to manage Puppet code in accordance with best practices. Additionally, we run all of our Puppet changes through our continuous integration pipeline in Jenkins using the beaker testing framework.

### ****Q10. What are Puppet Manifests?****

It is a very important question so make sure you go in a correct flow. According to me, you should first define Manifests. Every node (or Puppet Agent) has got its configuration details in Puppet Master, written in the native Puppet language. These details are written in the language which Puppet can understand and are termed as Manifests. They are composed of Puppet code and their filenames use the .pp extension.  
Now give an exampl. You can write a manifest in Puppet Master that creates a file and installs apache on all Puppet Agents (Slaves) connected to the Puppet Master.

### ****Q11.**** ****What is Puppet Module and How it is different from Puppet Manifest?****

For this answer, you can go with the below mentioned explanation:  
A Puppet Module is a collection of Manifests and data (such as facts, files, and templates), and they have a specific directory structure. Modules are useful for organizing your Puppet code, because they allow you to split your code into multiple Manifests. It is considered best practice to use Modules to organize almost all of your Puppet Manifests.  
Puppet programs are called Manifests which are composed of Puppet code and their file names use the .pp extension.

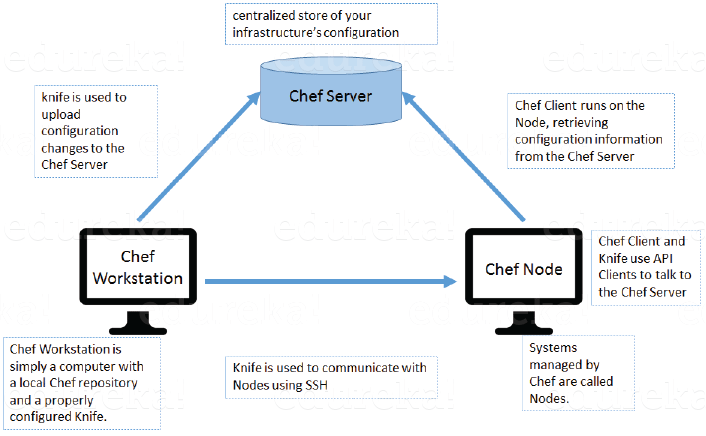
### ****Q12.**** ****What is Facter in Puppet?****

You are expected to answer what exactly Facter does in Puppet so according to me, you should say, “Facter gathers basic information (facts) about Puppet Agent such as hardware details, network settings, OS type and version, IP addresses, MAC addresses, SSH keys, and more. These facts are then made available in Puppet Master’s Manifests as variables.”

### ****Q13. What is Chef?****

Begin this answer by defining Chef. It is a powerful automation platform that transforms infrastructure into code. Chef is a tool for which you write scripts that are used to automate processes. What processes? Pretty much anything related to IT.  
Now you can explain the architecture of Chef, it consists of:

* **Chef Server:**The Chef Server is the central store of your infrastructure’s configuration data. The Chef Server stores the data necessary to configure your nodes and provides search, a powerful tool that allows you to dynamically drive node configuration based on data.
* **Chef Node:** A Node is any host that is configured using Chef-client. Chef-client runs on your nodes, contacting the Chef Server for the information necessary to configure the node. Since a Node is a machine that runs the Chef-client software, nodes are sometimes referred to as “clients”.
* **Chef Workstation:** A Chef Workstation is the host you use to modify your cookbooks and other configuration data.



### ****Q14. What is a resource in Chef?****

My suggestion is to first define Resource. A Resource represents a piece of infrastructure and its desired state, such as a package that should be installed, a service that should be running, or a file that should be generated.  
You should explain about the functions of Resource for that include the following points:

* Describes the desired state for a configuration item.
* Declares the steps needed to bring that item to the desired state.
* Specifies a resource type such as package, template, or service.
* Lists additional details (also known as resource properties), as necessary.
* Are grouped into recipes, which describe working configurations.

### ****Q15. What do you mean by recipe in Chef?****

For this answer, I will suggest you to use the above mentioned flow: first define Recipe. A Recipe is a collection of Resources that describes a particular configuration or policy. A Recipe describes everything that is required to configure part of a system.  
After the definition, explain the functions of Recipes by including the following points:

* Install and configure software components.
* Manage files.
* Deploy applications.
* Execute other recipes.

### ****Q16.**** ****How does a Cookbook differ from a Recipe in Chef?****

The answer to this is pretty direct. You can simply say, “a Recipe is a collection of Resources, and primarily configures a software package or some piece of infrastructure. A Cookbook groups together Recipes and other information in a way that is more manageable than having just Recipes alone.”

### ****Q17.**** ****What happens when you don’t specify a Resource’s action in Chef?****

My suggestion is to first give a direct answer: when you don’t specify a resource’s action, Chef applies the default action.  
Now explain this with an example, the below resource:  
**file ‘C:\Users\Administrator\chef-repo\settings.ini’ do**  
**content ‘greeting=hello world’**  
**end**  
is same as the below resource:  
**file ‘C:\Users\Administrator\chef-repo\settings.ini’ do**  
**action :create**  
**content ‘greeting=hello world’**  
**end**  
because: create is the file Resource’s default action.

### ****Q18. What is Ansible module?****

Modules are considered to be the units of work in Ansible. Each module is mostly standalone and can be written in a standard scripting language such as Python, Perl, Ruby, bash, etc.. One of the guiding properties of modules is idempotency, which means that even if an operation is repeated multiple times e.g. upon recovery from an outage, it will always place the system into the same state.

### ****Q19. What are playbooks in Ansible?****

Playbooks are Ansible’s configuration, deployment, and orchestration language. They can describe a policy you want your remote systems to enforce, or a set of steps in a general IT process. Playbooks are designed to be human-readable and are developed in a basic text language.  
At a basic level, playbooks can be used to manage configurations of and deployments to remote machines.

### ****Q20.**** ****How do I see a list of all of the ansible\_ variables?****

Ansible by default gathers “facts” about the machines under management, and these facts can be accessed in Playbooks and in templates. To see a list of all of the facts that are available about a machine, you can run the “setup” module as an ad-hoc action:  
**Ansible -m setup hostname**This will print out a dictionary of all of the facts that are available for that particular host.

### ****Q21. How can I set deployment order for applications?****

WebLogic Server 8.1 allows you to select the load order for applications. See the Application MBean Load Order attribute in Application. WebLogic Server deploys server-level resources (first JDBC and then JMS) before deploying applications. Applications are deployed in this order: connectors, then EJBs, then Web Applications. If the application is an EAR, the individual components are loaded in the order in which they are declared in the application.xml deployment descriptor.

### ****Q22. Can I refresh static components of a deployed application without having to redeploy the entire application?****

Yes, you can use weblogic.Deployer to specify a component and target a server, using the following syntax:  
java weblogic.Deployer -adminurl http://admin:7001 -name appname -targets server1,server2 -deploy jsps/\*.jsp

### ****Q23. How do I turn the auto-deployment feature off?****

The auto-deployment feature checks the applications folder every three seconds to determine whether there are any new applications or any changes to existing applications and then dynamically deploys these changes.

The auto-deployment feature is enabled for servers that run in development mode. To disable auto-deployment feature, use one of the following methods to place servers in production mode:

* In the Administration Console, click the name of the domain in the left pane, then select the Production Mode checkbox in the right pane.
* At the command line, include the following argument when starting the domain’s Administration Server:  
  -Dweblogic.ProductionModeEnabled=true
* Production mode is set for all WebLogic Server instances in a given domain.

### ****Q24. When should I use the external\_stage option?****

Set -external\_stage using weblogic.Deployer if you want to stage the application yourself, and prefer to copy it to its target by your own means.

### ****Q1. Why is Continuous monitoring necessary?****

I will suggest you to go with the below mentioned flow:  
Continuous Monitoring allows timely identification of problems or weaknesses and quick corrective action that helps reduce expenses of an organization. Continuous monitoring provides solution that addresses three operational disciplines known as:

* continuous audit
* continuous controls monitoring
* continuous transaction inspection

### ****Q2. What is Nagios?****

You can answer this question by first mentioning that Nagios is one of the monitoring tools. It is used for Continuous monitoring of systems, applications, services, and business processes etc in a DevOps culture. In the event of a failure, Nagios can alert technical staff of the problem, allowing them to begin remediation processes before outages affect business processes, end-users, or customers. With Nagios, you don’t have to explain why an unseen infrastructure outage affect your organization’s bottom line.  
Now once you have defined what is Nagios, you can mention the various things that you can achieve using Nagios.  
By using Nagios you can:

* Plan for infrastructure upgrades before outdated systems cause failures.
* Respond to issues at the first sign of a problem.
* Automatically fix problems when they are detected.
* Coordinate technical team responses.
* Ensure your organization’s SLAs are being met.
* Ensure IT infrastructure outages have a minimal effect on your organization’s bottom line.
* Monitor your entire infrastructure and business processes.

This completes the answer to this question. Further details like advantages etc. can be added as per the direction where the discussion is headed.

### ****Q3. How does Nagios works?****

I will advise you to follow the below explanation for this answer:  
Nagios runs on a server, usually as a daemon or service. Nagios periodically runs plugins residing on the same server, they contact hosts or servers on your network or on the internet. One can view the status information using the web interface. You can also receive email or SMS notifications if something happens.  
The Nagios daemon behaves like a scheduler that runs certain scripts at certain moments. It stores the results of those scripts and will run other scripts if these results change.

Now expect a few questions on Nagios components like Plugins, NRPE etc..

### ****Q4. What are Plugins in Nagios?****

Begin this answer by defining Plugins. They are scripts (Perl scripts, Shell scripts, etc.) that can run from a command line to check the status of a host or service. Nagios uses the results from Plugins to determine the current status of hosts and services on your network.   
Once you have defined Plugins, explain why we need Plugins. Nagios will execute a Plugin whenever there is a need to check the status of a host or service. Plugin will perform the check and then simply returns the result to Nagios. Nagios will process the results that it receives from the Plugin and take the necessary actions.

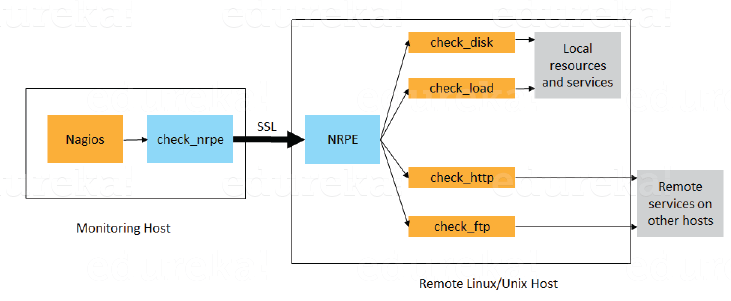
### ****Q5. What is NRPE (Nagios Remote Plugin Executor) in Nagios?****

For this answer, give a brief definition of Plugins. The NRPE addon is designed to allow you to execute Nagios plugins on remote Linux/Unix machines. The main reason for doing this is to allow Nagios to monitor “local” resources (like CPU load, memory usage, etc.) on remote machines. Since these public resources are not usually exposed to external machines, an agent like NRPE must be installed on the remote Linux/Unix machines.

I will advise you to explain the NRPE architecture on the basis of diagram shown below. The NRPE addon consists of two pieces:

* The check\_nrpe plugin, which resides on the local monitoring machine.
* The NRPE daemon, which runs on the remote Linux/Unix machine.

There is a SSL (Secure Socket Layer) connection between monitoring host and remote host as shown in the diagram below.



### ****Q6. What do you mean by passive check in Nagios?****

According to me, the answer should start by explaining Passive checks. They are initiated and performed by external applications/processes and the Passive check results are submitted to Nagios for processing.  
Then explain the need for passive checks. They are useful for monitoring services that are Asynchronous in nature and cannot be monitored effectively by polling their status on a regularly scheduled basis. They can also be used for monitoring services that are Located behind a firewall and cannot be checked actively from the monitoring host.

### ****Q7.**** ****When Does Nagios Check for external commands?****

Make sure that you stick to the question during your explanation so I will advise you to follow the below mentioned flow. Nagios check for external commands under the following conditions:

* At regular intervals specified by the command\_check\_interval option in the main configuration file or,
* Immediately after event handlers are executed. This is in addition to the regular cycle of external command checks and is done to provide immediate action if an event handler submits commands to Nagios.

### ****Q8. What is the difference between Active and Passive check in Nagios?****

For this answer, first point out the basic difference Active and Passive checks. The major difference between Active and Passive checks is that Active checks are initiated and performed by Nagios, while passive checks are performed by external applications.  
If your interviewer is looking unconvinced with the above explanation then you can also mention some key features of both Active and Passive checks:  
Passive checks are useful for monitoring services that are:

* Asynchronous in nature and cannot be monitored effectively by polling their status on a regularly scheduled basis.
* Located behind a firewall and cannot be checked actively from the monitoring host.

The main features of Actives checks are as follows:

* Active checks are initiated by the Nagios process.
* Active checks are run on a regularly scheduled basis.

### ****Q9. How does Nagios help with Distributed Monitoring?****

The interviewer will be expecting an answer related to the distributed architecture of Nagios. So, I suggest that you answer it in the below mentioned format:  
With Nagios you can monitor your whole enterprise by using a distributed monitoring scheme in which local slave instances of Nagios perform monitoring tasks and report the results back to a single master. You manage all configuration, notification, and reporting from the master, while the slaves do all the work. This design takes advantage of Nagios’s ability to utilize passive checks i.e. external applications or processes that send results back to Nagios. In a distributed configuration, these external applications are other instances of Nagios.

### ****Q10. Explain Main Configuration file of Nagios and its location?****

First mention what this main configuration file contains and its function. The main configuration file contains a number of directives that affect how the Nagios daemon operates. This config file is read by both the Nagios daemon and the CGIs (It specifies the location of your main configuration file).  
Now you can tell where it is present and how it is created. A sample main configuration file is created in the base directory of the Nagios distribution when you run the configure script. The default name of the main configuration file is nagios.cfg. It is usually placed in the etc/ subdirectory of you Nagios installation (i.e. /usr/local/nagios/etc/).

### ****Q11. Explain how Flap Detection works in Nagios?****

I will advise you to first explain Flapping first. Flapping occurs when a service or host changes state too frequently, this causes lot of problem and recovery notifications.  
Once you have defined Flapping, explain how Nagios detects Flapping. Whenever Nagios checks the status of a host or service, it will check to see if it has started or stopped flapping. Nagios follows the below given procedure to do that:

* Storing the results of the last 21 checks of the host or service analyzing the historical check results and determine where state changes/transitions occur
* Using the state transitions to determine a percent state change value (a measure of change) for the host or service
* Comparing the percent state change value against low and high flapping thresholds

A host or service is determined to have started flapping when its percent state change first exceeds a high flapping threshold. A host or service is determined to have stopped flapping when its percent state goes below a low flapping threshold.

### ****Q12. What are the three main variables that affect recursion and inheritance in Nagios?****

According to me the proper format for this answer should be:  
First name the variables and then a small explanation of each of these variables:

* Name
* Use
* Register

Then give a brief explanation for each of these variables. Name is a placeholder that is used by other objects. Use defines the “parent” object whose properties should be used. Register can have a value of 0 (indicating its only a template) and 1 (an actual object). The register value is never inherited.

### ****Q13. What is meant by saying Nagios is Object Oriented?****

Answer to this question is pretty direct. I will answer this by saying, “One of the features of Nagios is object configuration format in that you can create object definitions that inherit properties from other object definitions and hence the name. This simplifies and clarifies relationships between various components.”

### ****Q14. What is State Stalking in Nagios?****

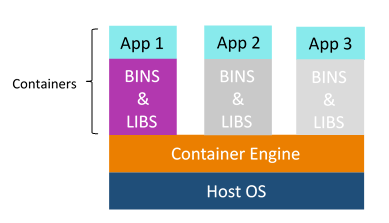
I will advise you to first give a small introduction on State Stalking. It is used for logging purposes. When Stalking is enabled for a particular host or service, Nagios will watch that host or service very carefully and log any changes it sees in the output of check results.  
Depending on the discussion between you and interviewer you can also add, “It can be very helpful in later analysis of the log files. Under normal circumstances, the result of a host or service check is only logged if the host or service has changed state since it was last checked.”

## **Containerization and Virtualization Interview Questions**

Let’s see how much you know about containers and VMs.

### ****Q1. What are containers?****

My suggestion is to explain the need for containerization first, containers are used to provide consistent computing environment from a developer’s laptop to a test environment, from a staging environment into production.  
Now give a definition of containers, a container consists of an entire runtime environment: an application, plus all its dependencies, libraries and other binaries, and configuration files needed to run it, bundled into one package. Containerizing the application platform and its dependencies removes the differences in OS distributions and underlying infrastructure.



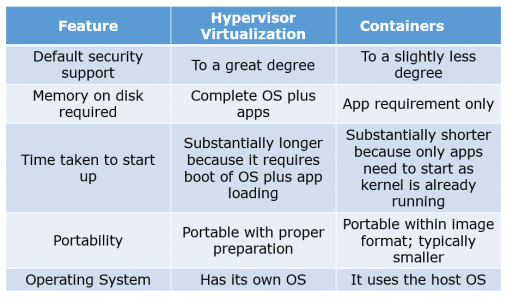
### ****Q2. What are the advantages that Containerization provides over virtualization?****

Below are the advantages of containerization over virtualization:

* Containers provide real-time provisioning and scalability but VMs provide slow provisioning
* Containers are lightweight when compared to VMs
* VMs have limited performance when compared to containers
* Containers have better resource utilization compared to VMs

### ****Q3. How exactly are containers (Docker in our case) different from hypervisor virtualization (vSphere)? What are the benefits?****

Given below are some differences. Make sure you include these differences in your answer:



### ****Q4. What is Docker image?****

I suggest that you go with the below mentioned flow:  
Docker image is the source of Docker container. In other words, Docker images are used to create containers. Images are created with the build command, and they’ll produce a container when started with run. Images are stored in a Docker registry such as registry.hub.docker.com because they can become quite large, images are designed to be composed of layers of other images, allowing a minimal amount of data to be sent when transferring images over the network.  
**Tip: Be aware of Dockerhub in order to answer questions on pre-available images.**

### ****Q5. What is Docker container?****

This is a very important question so just make sure you don’t deviate from the topic. I advise you to follow the below mentioned format:  
Docker containers include the application and all of its dependencies but share the kernel with other containers, running as isolated processes in user space on the host operating system. Docker containers are not tied to any specific infrastructure: they run on any computer, on any infrastructure, and in any cloud.  
Now explain how to create a Docker container, Docker containers can be created by either creating a Docker image and then running it or you can use Docker images that are present on the Dockerhub.  
Docker containers are basically runtime instances of Docker images.

### ****Q6. What is Docker hub?****

Answer to this question is pretty direct. Docker hub is a cloud-based registry service which allows you to link to code repositories, build your images and test them, stores manually pushed images, and links to Docker cloud so you can deploy images to your hosts. It provides a centralized resource for container image discovery, distribution and change management, user and team collaboration, and workflow automation throughout the development pipeline.

### ****Q7.**** ****How is Docker different from other container technologies?****

According to me, below points should be there in your answer:  
Docker containers are easy to deploy in a cloud. It can get more applications running on the same hardware than other technologies, it makes it easy for developers to quickly create, ready-to-run containerized applications and it makes managing and deploying applications much easier. You can even share containers with your applications.  
If you have some more points to add you can do that but make sure the above the above explanation is there in your answer.

### ****Q8.**** ****What is Docker Swarm?****

You should start this answer by explaining Docker Swarn. It is native clustering for Docker which turns a pool of Docker hosts into a single, virtual Docker host. Docker Swarm serves the standard Docker API, any tool that already communicates with a Docker daemon can use Swarm to transparently scale to multiple hosts.  
I will also suggest you to include some supported tools:

* Dokku
* Docker Compose
* Docker Machine
* Jenkins

### ****Q9. What is Dockerfile used for?****

This answer according to me should begin by explaining the use of Dockerfile. Docker can build images automatically by reading the instructions from a Dockerfile.  
Now I suggest you to give a small definition of Dockerfle. A Dockerfile is a text document that contains all the commands a user could call on the command line to assemble an image. Using docker build users can create an automated build that executes several command-line instructions in succession.

Now expect a few questions to test your experience with Docker.

### ****Q10.**** ****Can I use json instead of yaml for my compose file in Docker?****

You can use json instead of yaml for your compose file, to use json file with compose, specify the filename to use for eg:  
**docker-compose -f docker-compose.json up**

### ****Q11.**** ****Tell us how you have used Docker in your past position?****

Explain how you have used Docker to help rapid deployment. Explain how you have scripted Docker and used Docker with other tools like Puppet, Chef or Jenkins. If you have no past practical experience in Docker and have past experience with other tools in similar space, be honest and explain the same. In this case, it makes sense if you can compare other tools to Docker in terms of functionality.

### ****Q12. How to create Docker container?****

I will suggest you to give a direct answer to this. We can use Docker image to create Docker container by using the below command:  
**docker run -t -i <image name> <command name>**This command will create and start container.  
You should also add, If you want to check the list of all running container with status on a host use the below command:  
**docker ps -a**

### ****Q13. How to stop and restart the Docker container?****

In order to stop the Docker container you can use the below command:  
**docker stop <container ID>**  
Now to restart the Docker container you can use:  
**docker restart <container ID>**

### Q14. How far do Docker containers scale?

Large web deployments like Google and Twitter, and platform providers such as Heroku and dotCloud all run on container technology, at a scale of hundreds of thousands or even millions of containers running in parallel.

### Q15. What platforms does Docker run on?

I will start this answer by saying Docker runs on only Linux and Cloud platforms and then I will mention the below vendors of Linux:

* Ubuntu 12.04, 13.04 et al
* Fedora 19/20+
* RHEL 6.5+
* CentOS 6+
* Gentoo
* ArchLinux
* openSUSE 12.3+
* CRUX 3.0+

Cloud:

* Amazon EC2
* Google Compute Engine
* Microsoft Azure
* Rackspace

**Note that Docker does not run on Windows or Mac.**

### ****Q16. Do I lose my data when the Docker container exits?****

You can answer this by saying, no I won’t loose my data when Dcoker container exits. Any data that your application writes to disk gets preserved in its container until you explicitly delete the container. The file system for the container persists even after the container halts.

### ****1. How does HTTP work?****

The HTTP protocol works in a client and server model like most other protocols. A web browser using which a request is initiated is called as a client and a web server software which responds to that request is called a server. World Wide Web Consortium and the Internet Engineering Task Force are two important spokes in the standardization of the HTTP protocol. HTTP allows improvement of its request and response with the help of intermediates, for example a gateway, a proxy, or a tunnel. The resources that can be requested using the HTTP protocol, are made available using a certain type of URI (Uniform Resource Identifier) called a URL (Uniform Resource Locator). TCP (Transmission Control Protocol) is used to establish a connection to the application layer port 80 used by HTTP.

### ****2. Explain your understanding and expertise on both the software development side and the technical operations side of an organization you’ve worked for in the past.****

DevOps engineers almost always work in a 24/7 business critical online environment. I was adaptable to on-call duties and able to take up real-time, live-system responsibility. I successfully automated processes to support continuous software deployments. I have experience with public/private clouds, tools like Chef or Puppet, scripting and automation with tools like Python and PHP, and a background in Agile.

### ****3. Discuss your experience building bridges between IT Ops, QA and development.****

DevOps is all about effective communication and collaboration. I’ve been able to deal with production issues from the development and operations sides, effectively straddling the two worlds. I’m less interested in finding blame or playing the hero than I am with ensuring that all of the moving parts come together.

### ****4. What types of testing are needed?****

Software teams will often look for the “fair weather” path to system completion; that is, they start from an assumption that software will usually work and only occasionally fail. I believe to practice defensive programming in a pragmatic way, which often means assuming that the code will fail and planning for those failures. I try to incorporate unit test strategy, use of test harnesses, early load testing; network simulation, A/B and multi-variate testing  etc.

### ****5. Give me an example of how you would handle projects?****

As a professional with managerial responsibilities, I would demonstrate a clear understanding of DevOps project management tactics and also work with teams to set objectives, streamline workflow, maintain scope,  research and introduce new tools or frameworks, translate requirements into workflow and follow up. I would resort to CI, release management and other tools to keep interdisciplinary projects on track.

### ****6. What’s your career objective in your role as a DevOps engineer?****

My passion is breaking down the barriers and building and improving processes, so that the engineering and operations teams work better and smarter. That’s why I love DevOps. It’s an opportunity to be involved in the entire delivery system from start to finish.

### ****7. How would you make software deployable?****

The ability to script the installation and reconfiguration of software systems is essential towards controlled and automated change. Although there is an increasing trend for new software to enable this, older systems and products suffer from the assumption that changes would be infrequent and minor, and so make automated changes difficult. As a professional who appreciates the need to expose configuration and settings in a manner accessible to automation, I will work with concepts like Inversion of Control (IoC) and Dependency Injection, scripted installation, test harnesses, separation of concerns, command-line tools, and infrastructure as code.

## **8. What is the one most important thing DevOps helps do?**

The most important thing DevOps helps do is to get the changes into production as quickly as possible while minimizing risks in software quality assurance and compliance. That is the primary objective of DevOps. However, there are many other positive side-effects to DevOps. For example, clearer communication and better working relationships between teams which creates a less stressful working environment.

### ****9. Which scripting languages do you think are most important for a DevOps engineer?****

As far as scripting languages go, the simpler the better. In fact, the language itself isn’t as important as understanding design patterns and development paradigms such as procedural, object-oriented, or functional programming.

### ****10. How do you expect you would be required to multitask as a DevOps professional?****

I believe I’ll be expected to:

1. Focus attention on bridging communication gaps between Development and Operations teams.
2. Understand system design from an architect’s perspective, software development from a developer’s perspective,operations and infrastructure from the perspective of a seasoned Systems Administrator.
3. Execute – to be able to actually do what needs to be done.

### ****11. What testing is necessary to ensure that a new service is ready for production?****

DevOps is all about continuous testing throughout the process, starting with development through to production. Everyone shares the testing responsibility. This ensures that developers are delivering code that doesn’t have any errors and is of high quality, and it also helps everyone leverage their time most effectively.

### ****12. What’s a PTR in DNS?****

Pointer records are used to map a network interface (IP) to a host name. These are primarily used for reverse DNS. Reverse DNS is setup very similar to how normal (forward) DNS is setup.  When you delegate the DNS forward, the owner of the domain tells the registrar to let your domain use specific name servers.

### ****13. Describe two-factor authentication?****

Two-factor authentication is a security process in which the user provides two means of identification from separate categories of credentials; one is typically a physical token, such as a card, and the other is typically something memorized, such as a security code.

### ****14. Tell us about the CI tools that you are familiar with?****

The premise of CI is to get feedback as early as possible because the earlier you get feedback, the less things cost to fix. Popular open source tools include Hudson, Jenkins, CruiseControl and CruiseControl.NET. Commercial tools include ThoughtWorks’ Go, Urbancode’s Anthill Pro, Jetbrains’ Team City and Microsoft’s Team Foundation Server.

### ****15. What are the advantages of NoSQL database over RDBMS?****

The advantages are:

1. Less need for ETL
2. Support for unstructured text
3. Ability to handle change over  time
4. Breadth of functionality
5. Ability to scale horizontally
6. Support for multiple  data structures
7. Choice of vendors

### ****16. What is an MX record in DNS?****

MX records are mail exchange records used for determining the priority of email servers for a domain. The lowest priority email server is the first destination for email. If the lowest priority email server is unavailable, mail will be sent to the higher priority email servers.

### ****17. What is the difference between RAID 0 and RAID 1?****

RAID 1 offers redundancy through mirroring, i.e., data is written identically to two drives. RAID 0 offers no redundancy and instead uses striping, i.e., data is split across all the drives. This means RAID 0 offers no fault tolerance; if any of the constituent drives fails, the RAID unit fails.

### ****18. How would you prepare for a migration?****

Tips to answer: This question evaluates your experience of real projects with all the awkwardness and complexity they bring. Include terms like cut-over, dress rehearsals, roll-back and roll-forward, DNS solutions, feature toggles, branch by abstraction, and automation in your answer. Developing greenfield systems with little or no existing technology in place is always easier than having to deal with legacy components and configuration. As a candidate if you appreciate that any interesting software system will in effect be under constant migration, you will appear suitable for the role.

### ****19. What’s your systems background?****

Tips to answer: Some DevOps jobs require extensive systems knowledge, including server clustering and highly concurrent systems. As a DevOps engineer, you need to analyze system capabilities and implement upgrades for efficiency, scalability and stability, or resilience. It is recommended that you have a solid knowledge of OSes and supporting technologies, like network security, virtual private networks and proxy server configuration.

DevOps relies on virtualization for rapid workload provisioning and allocating compute resources to new VMs to support the next rollout, so it is useful to have in-depth knowledge around popular hypervisors. This should ideally include backup, migration and lifecycle management tactics to protect, optimize and eventually recover computing resources. Some environments may emphasize microservices software development tailored for virtual containers. Operations expertise must include extensive knowledge of systems management tools like Microsoft System Center, Puppet, Nagios and Chef. DevOps jobs with an emphasis on operations require detailed problem-solving, troubleshooting and analytical skills.

### ****20. What DevOp tools have you worked with?****

Tips to answer: Software configuration management and build/release (version control) tools, including Apache Subversion, Mercurial, Fossil and others, help document change requests. Developers can more easily follow the company’s best practices and policies while software changes.

Continuous integration (CI) tools such as Rational Build Forge, Jenkins and Semaphore merge all developer copies of the working code into a central version. These tools are important for larger groups where teams of developers work on the same codebase simultaneously. QA experts use code analyzers to test software for bugs, security and performance. If you’ve used HP’s Fortify Static Code Analyzer, talk about how it identified security vulnerabilities in coding languages. Also speak about tools like GrammaTech’s CodeSonar that you used to identify memory leaks, buffer underruns and other defects for C/C++ and Java code. It is essential that you have adequate command of the principal languages like Ruby, C#, .NET, Perl, Python, Java, PHP, Windows PowerShell, and are comfortable with the associated OS environments Windows, Linux and Unix.

### ****21. How much have you interacted with cloud based software development?****

Tips to answer: Share your knowledge around use of cloud platforms, provisioning new instances, coding new software iterations with the cloud provider’s APIs or software development kits, configuring clusters to scale computing capacity, managing workload lifecycles and so on. This is the perfect opportunity to discuss container-based cloud instances as an alternative to conventional VMs. Event-based cloud computing, such as AWS Lambda offers another approach to software development, a boon for experienced DevOps candidates. In your interview, mention experience handling big data, which uses highly scalable cloud infrastructures to tackle complex computing tasks.

### ****22. What other tools are you familiar with that might help you in this role?****

Tips to answer: DevOps is so diverse and inclusive that it rarely ends with coding, testing and systems. A DevOps project might rely on database platforms like SQL or NoSQL, data structure servers like Redis, or configuration and management issue tracking systems like Redmine. Web applications are popular for modern enterprises, making a background with Web servers, like Microsoft Internet Information Services, Apache Tomcat or other Web servers, beneficial. Make sure to bring across that you are familiar with Agile application lifecycle management techniques and tools.

### ****23. Are you familiar with just Linux or have you worked with Windows environments as well?****

Tips to answer: Demonstrate as much as you can, a clear understanding of both the environments including the key tools.

### ****24. How can you reduce load time of a dynamic website?****

Tips to answer: Talk about Webpage optimization, cached web pages, quality web hosting , compressed text files, Apache  fine tuning.

### ****25. Describe your experience implementing continuous deployment?****

Tips to answer: Answer with a comprehensive list of all the tools that you used. Include inferences of the challenges you faced and how you tackled them.

### ****26. How would you ensure traceability?****

Tips to answer: This question probes your attitude to metrics, logging, transaction journeys, and reporting. You should be able to identify that metric, monitoring and logging needs to be a core part of the software system, and that without them, the software is essentially not going to be able to appear maintained and diagnosed. Include words like SysLog, Splunk, error tracking, Nagios, SCOM, Avicode in your answer.

### ****27. What was your greatest achievement on a recent project?****

Tips to answer: Make sure you demonstrate your perfect understanding of both development and operations. Do not let your answer lean towards one particular skillset ignoring the other. Even if you have worked in an environment wherein you had to work more with one skillset, assure the intervewer that you are agile according to the needs of your organization.

### ****28. What problems did you face and how did you solve them in a way that met the team’s goals?****

Tips to answer: This questions aims to find out how much you can handle stress and non-conformity at work. Talk about your leadership skills to handle and motivate the team to solve problems together.Talk about CI, release management and other tools to keep interdisciplinary projects on track.

### ****29. Are you more Dev or Ops?****

Tips to answer: This is probably the trickiest question that you might face in the interview. Emphasize the fact that this depends a lot on the job, the company you are working for and the skills of people involved. You really have to be able to alternate between both sides of the fence at any given time. Talk about your experience and demonstrate how you are agile with both.

### ****30. What special training or education did it require for you to become a DevOps engineer?****

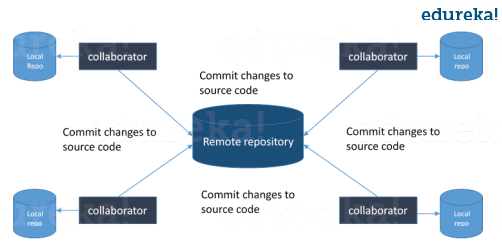
Tips to answer: DevOps is more of a mind-set or philosophy rather than a skill-set. The typical technical skills associated with DevOps Engineers today is Linux systems administration, scripting, and experience with one of the many continuous integration or configuration management tools like Jenkins and Chef. What it all boils down to is that whatever skill-sets you have, while important, are not as important as having the ability to learn new skills quickly to meet the needs. It’s all about pattern recognition, and having the ability to merge your experiences with current requirements.Proficiency in Windows and Linux systems administration, script development, an understanding of structured programming and object-oriented design, and experience creating and consuming RESTful APIs would take one a long way.

|  |  |
| --- | --- |
| **Git vs SVN** | |
| **Git** | **SVN** |
| 1. Git is a Decentralized Version Control tool | 1. SVN is a  Centralized Version Control tool |
| 2.  It belongs to the 3rd generation of Version Control tools | 2. It belongs to the 2nd generation of Version Control tools |
| 3. Clients can clone entire repositories on their local systems | 3.  Version history is stored on a server-side repository |
| 4. Commits are possible even if offline | 4. Only online commits are allowed |
| 5. Push/pull operations are faster | 5. Push/pull operations are slower |
| 6. Works are shared automatically by commit | 6. Nothing is shared automatically |

**Q2. What is Git?**

I will suggest you to attempt this question by first telling about the architecture of git as shown in the below diagram just try to explain the diagram by saying:

* Git is a Distributed Version Control system (DVCS). It can track changes to a file and allows you to revert back to any particular change.
* Its distributed architecture provides many advantages over other Version Control Systems (VCS) like SVN one major advantage is that it does not rely on a central server to store all the versions of a project’s files.
* Instead, every developer “clones” a copy of a repository I have shown in the diagram with “Local repository” and has the full history of the project on his hard drive so when there is a server outage all you need for recovery is one of your teammate’s local Git repository.
* There is a central cloud repository as well where developers can commit changes and share it with other teammates as you can see in the diagram where all collaborators are commiting changes “Remote repository”.



*Now, the next set of Git interview questions will test your experience with Git:*

**Q3. What is the command to write a commit message in Git?**

Answer to this is pretty straightforward.

Command that is used to write a commit message is “**git commit -a**”.

Now explain about -a flag by saying -a on the command line instructs git to commit the new content of all tracked files that have been modified. Also mention you can use “**git add<file>**” before git commit -a if new files need to be committed for the first time.

**Q4. What is ‘bare repository’ in Git?**

You are expected to tell the difference between a “working directory” and “bare repository”.

A “bare” repository in Git just contains the version control information and no working files (no tree) and it doesn’t contain the special .git sub-directory. Instead, it contains all the contents of the .git sub-directory directly in the main directory itself, where as working directory consist of:

1. A .git subdirectory with all the Git related revision history of your repo.
2. A working tree, or checked out copies of your project files.

**Q5. What language is used in Git?**

Instead of just telling the name of the language, you need to tell the reason for using it as well. I will suggest you to answer this by saying:

Git uses ‘C’ language. GIT is fast, and ‘C’ language makes this possible by reducing the overhead of run times associated with high level languages.

**Q6. In Git how do you revert a commit that has already been pushed and made public?**

There can be two answers to this question and make sure that you include both because any of the below options can be used depending on the situation:

* Remove or fix the bad file in a new commit and push it to the remote repository. This is the most natural way to fix an error. Once you have made necessary changes to the file, commit it to the remote repository for that I will use  
  **git commit -m “commit message”**
* Create a new commit that undoes all changes that were made in the bad commit.to do this I will use a command  
  **git revert <name of bad commit>**

**Q7. What is the difference between git pull and git fetch?**

Git pull command pulls new changes or commits from a particular branch from your central repository and updates your target branch in your local repository.

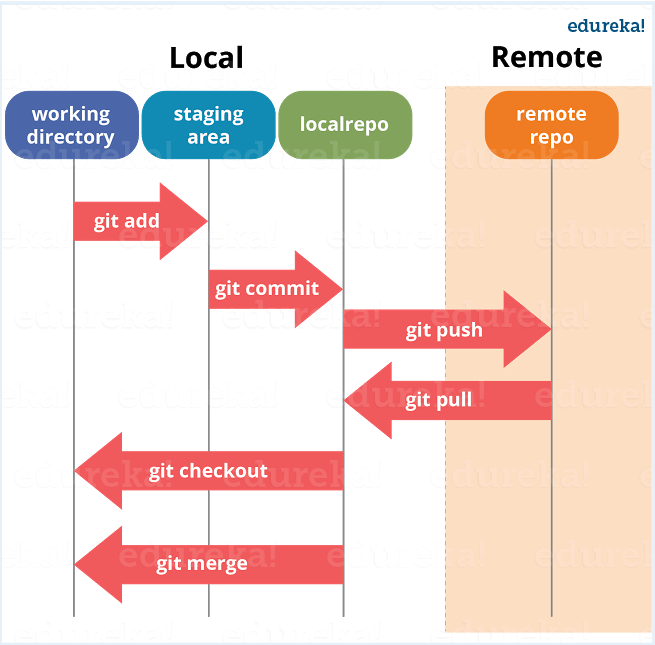
Git fetch is also used for the same purpose but it works in a slightly different way. When you perform a git fetch, it pulls all new commits from the desired branch and stores it in a new branch in your local repository. If you want to reflect these changes in your target branch, git fetch must be followed with a git merge. Your target branch will only be updated after merging the target branch and fetched branch. Just to make it easy for you, remember the equation below:

Git pull = git fetch + git merge

**Q8. What is ‘staging area’ or ‘index’ in Git?**

For this answer try to explain the below diagram as you can see:

That before completing the commits, it can be formatted and reviewed in an intermediate area known as ‘Staging Area’ or ‘Index’. From the diagram it is evident that every change is first verified in the staging area I have termed it as “stage file” and then that change is committed to the repository.



*If your interviewer has good knowledge on Git he/she will dig in deep, so the next set of Git interview questions will be more challenging.*

**Q9. What is Git stash?**

According to me you should first explain the need for Git stash.

Often, when you’ve been working on part of your project, things are in a messy state and you want to switch branches for sometime to work on something else. The problem is, you don’t want to do a commit of half-done work just so you can get back to this point later. The answer to this issue is Git stash.

Now explain what is Git stash.

Stashing takes your working directory that is, your modified tracked files and staged changes and saves it on a stack of unfinished changes that you can reapply at any time.

**Q10. What is Git stash drop?**

Begin this answer by saying for what purpose we use Git ‘stash drop’.

Git ‘stash drop’ command is used to remove the stashed item. It will remove the last added stash item by default, and it can also remove a specific item if you include it as an argument.

Now give an example.

If you want to remove a particular stash item from the list of stashed items you can use the below commands:

**git stash list:**It will display the list of stashed items like:  
stash@{0}: WIP on master: 049d078 added the index file  
stash@{1}: WIP on master: c264051 Revert “added file\_size”  
stash@{2}: WIP on master: 21d80a5 added number to log

If you want to remove an item named stash@{0} use command **git stash drop stash@{0}**.

**Q11. How do you find a list of files that has changed in a particular commit?**

For this answer instead of just telling the command, explain what exactly this command will do.

To get a list files that has changed in a particular commit use the below command:

**git diff-tree -r {hash}**

Given the commit hash, this will list all the files that were changed or added in that commit. The -r flag makes the command list individual files, rather than collapsing them into root directory names only.

You can also include the below mentioned point, although it is totally optional but will help in impressing the interviewer.

The output will also include some extra information, which can be easily suppressed by including two flags:

**git diff-tree –no-commit-id –name-only -r {hash}**

Here –no-commit-id will suppress the commit hashes from appearing in the output, and –name-only will only print the file names, instead of their paths.

**Q12. What is the function of ‘git config’?**

First tell why we need ‘**git config**‘.

Git uses your username to associate commits with an identity. The git config command can be used to change your Git configuration, including your username.

Now explain with an example.

Suppose you want to give a username and email id to associate commit with an identity so that you can know who has made a particular commit. For that I will use:

**git config –global user.name “Your Name”:**This command will add username.  
**git config –global user.email “Your E-mail Address”:**This command will add email id.

**Q13. What does commit object contains?**

Commit object contains the following components, you should mention all the three points present below:

* A set of files, representing the state of a project at a given point of time
* Reference to parent commit objects
* An SHAI name, a 40 character string that uniquely identifies the commit object.

**Q14. How can you create a repository in Git?**

This is probably the most frequently asked questions and answer to this is really simple.

To create a repository, create a directory for the project if it does not exist, then run command “**git init**”. By running this command .git directory will be created in the project directory.

**Q15. How do you squash last N commits into a single commit?**

There are two options to squash last N commits into a single commit include both of the below mentioned options in your answer:

* If you want to write the new commit message from scratch use the following command  
  **git reset –soft HEAD~N &&**  
  **git commit**
* If you want to start editing the new commit message with a concatenation of the existing commit messages then you need to extract those messages and pass them to Git commit for that I will use  
  **git reset –soft HEAD~N &&**  
  **git commit –edit -m”$(git log –format=%B –reverse**[**.HEAD@{N}**](mailto:HEAD..HEAD@%7b1%7d)**)”**

**Q16. What is Git bisect? How can you use it to determine the source of a (regression) bug?**

I will suggest you to first give a small definition of Git bisect.

Git bisect is used to find the commit that introduced a bug by using binary search. Command for Git bisect is  
**git bisect <subcommand> <options>**

Now since you have mentioned the command above explain them what this command will do.

This command uses a binary search algorithm to find which commit in your project’s history introduced a bug. You use it by first telling it a “bad” commit that is known to contain the bug, and a “good” commit that is known to be before the bug was introduced. Then Git bisect picks a commit between those two endpoints and asks you whether the selected commit is “good” or “bad”. It continues narrowing down the range until it finds the exact commit that introduced the change.

**Q17. How do you configure a Git repository to run code sanity checking tools right before making commits, and preventing them if the test fails?**

I will suggest you to first give a small introduction to sanity checking.

A sanity or smoke testdetermines whether it is possible and reasonable to continue testing.

Now explain how to achieve this.

This can be done with a simple script related to the pre-commit hook of the repository. The pre-commit hook is triggered right before a commit is made, even before you are required to enter a commit message. In this script one can run other tools, such as linters and perform sanity checks on the changes being committed into the repository.

Finally, give an example, you can refer the below script:

**#!/bin/sh**  
**files=$(git diff –cached –name-only –diff-filter=ACM | grep ‘.go$’)**  
**if [ -z files ]; then**  
**exit 0**  
**fi**  
**unfmtd=$(gofmt -l $files)**  
**if [ -z unfmtd ]; then**  
**exit 0**  
**fi**  
**echo “Some .go files are not fmt’d”**  
**exit 1**

This script checks to see if any .go file that is about to be committed needs to be passed through the standard Go source code formatting tool gofmt. By exiting with a non-zero status, the script effectively prevents the commit from being applied to the repository.

*The Interviewer has not started asking questions on branching yet, so the next set of Git interview questions will be dealing with branching in Git.*

**Q18. Describe branching strategies you have used?**

This question is asked to test your branching experience with Git so, tell them about how you have used branching in your previous job and what purpose does it serves, you can refer the below mention points:

* Feature branching  
  A feature branch model keeps all of the changes for a particular feature inside of a branch. When the feature is fully tested and validated by automated tests, the branch is then merged into master.
* Task branching  
  In this model each task is implemented on its own branch with the task key included in the branch name. It is easy to see which code implements which task, just look for the task key in the branch name.
* Release branching  
  Once the develop branch has acquired enough features for a release, you can clone that branch to form a Release branch. Creating this branch starts the next release cycle, so no new features can be added after this point, only bug fixes, documentation generation, and other release-oriented tasks should go in this branch. Once it is ready to ship, the release gets merged into master and tagged with a version number. In addition, it should be merged back into develop branch, which may have progressed since the release was initiated.

In the end tell them that branching strategies varies from one organization to another so I know basic branching operations like delete, merge, checking out a branch etc..

**Q19. How will you know in Git if a branch has already been merged into master?**

The answer is pretty direct.

To know if a branch has been merged into master or not you can use the below commands:

**git branch –merged** It lists the branches that have been merged into the current branch.  
**git branch –no-merged** It lists the branches that have not been merged.

**Q20. What is Git rebase and how can it be used to resolve conflicts in a feature branch before merge?**

According to me you should start by saying git rebase is a command which will merge another branch into the branch where you are currently working, and move all of the local commits that are ahead of the rebased branch to the top of the history on that branch.

Now, once you have defined Git rebase time for an example to show how it can be used to resolve conflicts in a feature branch before merge.

If a feature branch was created from the master, and since then the master branch has received new commits, Git rebase can be used to move the feature branch to the tip of master. The command effectively will replay the changes made in the feature branch at the tip of master, allowing conflicts to be resolved in the process. When done with care, this will allow the feature branch to be merged into master with relative ease and sometimes as a simple fast-forward operation.

*You can also expect some off track questions, so the next question in this Git interview questions blog will be regarding SubGit.*

**Q21. What is SubGit?**

Begin this answer by explaining what is SubGit used for.

SubGit is a tool for SVN to Git migration. It creates a writable Git mirror of a local or remote Subversion repository and uses both Subversion and Git as long as you like.

Now you can include some advantages like you can do a fast one-time import from Subversion to Git or use SubGit within Atlassian Bitbucket Server.We can use SubGit to create a bi-directional Git-SVN mirror of existing Subversion repository. You can push to Git or commit to Subversion at your convenience. Synchronization will be done by SubGit.

## **Why Do We Need Ansible?**

Well before I tell you what is Ansible, it is of utmost importance to understand the problems that were faced before Ansible.

Let us take a little flashback to the beginning of networked computing when deploying and managing servers reliably and efficiently has been a challenge. Previously, system administrators managed servers by hand, installing software, changing configurations, and administering services on individual servers.  
As data centers grew, and hosted applications became more complex, administrators realized they couldn’t scale their manual systems management as fast as the applications they were enabling. It also hampered the velocity of the work of the developers since the development team was agile and releasing software frequently, but IT operations were spending more time configuring the systems. That’s why server provisioning and configuration management tools came to flourish.

Consider the tedious routine of administering a server fleet. We always need to keep updating, pushing changes, copying files on them etc. These tasks make things very complicated and time consuming.

But let me tell you that there is a solution to the above stated problem. The solution is –**Ansible.**

But before I go ahead to explain you all about Ansible, let me get you familiarized with few Ansible terminologies:

## **Ansible Terms:**

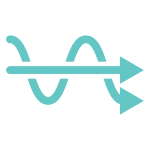
* **Controller Machine**: The machine where Ansible is installed, responsible for running the provisioning on the servers you are managing.
* **Inventory**: An initialization file that contains information about the servers you are managing.
* **Playbook**: The entry point for Ansible provisioning, where the automation is defined through tasks using YAML format.
* **Task**: A block that defines a single procedure to be executed, e.g. Install a package.
* **Module**: A module typically abstracts a system task, like dealing with packages or creating and changing files. Ansible has a multitude of built-in modules, but you can also create custom ones.
* **Role**: A pre-defined way for organizing playbooks and other files in order to facilitate sharing and reusing portions of a provisioning.
* **Play**: A provisioning executed from start to finish is called a play. In simple words, execution of a playbook is called a play.
* **Facts**: Global variables containing information about the system, like network interfaces or operating system.
* **Handlers**: Used to trigger service status changes, like restarting or stopping a service.

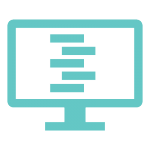
Ansible is a helpful tool that allows you to create groups of machines, describe how these machines should be configured or what actions should be taken on them. Ansible issues all commands from a central location to perform these tasks.

No other client software is installed on the node machines. It uses SSH to connect to the nodes. Ansible only needs to be installed on the control machine (the machine from which you will be running commands) which can even be your laptop. It is a simple solution to a complicated problem.

I am not boasting off when I say that Ansible has filled up all the holes in Configuration Management and IT Orchestration world. You will know it too, when you take a look at the benefits of Ansible mentioned below:

## **Advantages Of Using Ansible**

  
**Simple:**Ansible uses a simple syntax written in YAML called**playbooks**. YAML is a human-readable data serialization language. It is extraordinarily simple. So, no special coding skills are required and even people in your IT organization, who do not know what is Ansible can likely read a playbook and understand what is happening. Ansible always executes tasks in order. It is simple to install too (Don’t believe me? Check out my[***Ansible Installation***](https://www.edureka.co/blog/install-ansible/) blog). Altogether the simplicity ensures that you can get started quickly.

  
**Agentless:** Finally, Ansible is completely agentless. There are no agents/software or additional firewall ports that you need to install on the  client systems or hosts which you want to automate. You do not have to separately set up a management infrastructure which includes managing your entire systems, network and storage. Ansible further reduces the effort required for your team to start automating right away.

**Powerful & Flexible:** Ansible has powerful features that can enable you to model even the most complex IT workflows. In this aspect, Ansible’s batteries included approach (This philosophy means that something is self-sufficient, comes out-of-the-box ready to use, with everything that is needed) can manage the infrastructure, networks, operating systems and services that you are already using, as Ansible provides you with hundreds of modules to manage them. Together Ansible’s capabilities allow you to orchestrate the entire application environment regardless of where it is deployed.

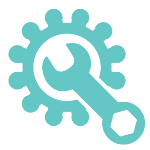
****

**Efficient:** No extra software on your servers means more resources for your applications. Also, since Ansible modules work via JSON, Ansible is extensible with modules written in a programming language you already know. Ansible introduces modules as basic building blocks for your software. So, you can even customize it as per your needs. For e.g. If you have an existing message sending module which sends messages in plain-text, and you want to send images too, you can add image sending features on top of it.

## **What Ansible Can Do?**

Ansible is usually grouped along with other Configuration Management tools like Puppet, Chef, SaltStack etc. Well, let me tell you, Ansible is not just limited to Configuration Management. It can be used in many different ways too. I have mentioned some of them below:

  
**Provisioning:** Your apps have to live somewhere. If you’re PXE (Preboot eXecution Environment) booting and kick starting bare-metal servers or Virtual Machines, or creating virtual or cloud instances from templates, Ansible & Ansible Tower helps to streamline this process. For example, if I want to test the debug version of an application that is built with Visual C++, I ought to meet some prerequisite requirements like having Visual C++ library DLLs (msvcr100d.dll). I will also need Visual Studio installed in your computer. This is when Ansible makes sure that the required packages are downloaded and installed in order to provision my application.

  
**Configuration Management:** It establishes and maintains consistency of the product performance by recording and updating detailed information which describes an enterprise’s hardware and software. Such information typically includes the versions and updates that have been applied to installed software packages and the locations and network addresses of hardware devices. For e.g. If you want to install the new version of Tomcat on all of the machines present in your enterprise, it is not feasible for you to manually go and update each and every machine. You can install Tomcat in one go on all of your machines with Ansible playbooks and inventory written in the most simple way. All you have to  do is list out the IP addresses of your nodes in the inventory and write a playbook to install Tomcat. Run the playbook from your control machine & it will be installed on all your nodes.

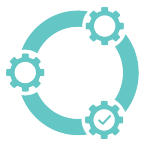
  
**Application Deployment:** When you define your application with Ansible, and manage the deployment with Ansible Tower, teams are able to effectively manage the entire application life cycle from development to production. For example, let’s say I want to deploy the Default Servlet Engine. There are a number of steps that needs to be undergone to deploy the engine.

* Move a .war application from dropins directory to apps directory
* Add server.xml file
* Navigate to the webpage to see your application.

But why worry about performing these steps one by one when we have a tool like Ansible. All you need to do is list these tasks in your Ansible playbook and sit back watching Ansible executing these tasks in order.

****

**Security and Compliance:** When you define your security policy in Ansible, scanning and remediation of site-wide security policy can be integrated into other automated processes. And it’ll be integral in everything that is deployed. It means that, you need to configure your security details once in your control machine and it will be embedded in all other nodes automatically. Moreover, all the credentials (admin users id’s & passwords) that are stored within Ansible are not retrievable in plain-text by any user.

  
**Orchestration:** Configurations alone don’t define your environment. You need to define how multiple configurations interact and ensure the disparate pieces can be managed as a whole. Out of complexity and chaos, Ansible brings order. Ansible provides Orchestration in the sense of aligning the business request with the applications, data, and infrastructure. It defines the policies and service levels through automated workflows, provisioning, and change management. This creates an application-aligned infrastructure that can be scaled up or down based on the needs of each application.

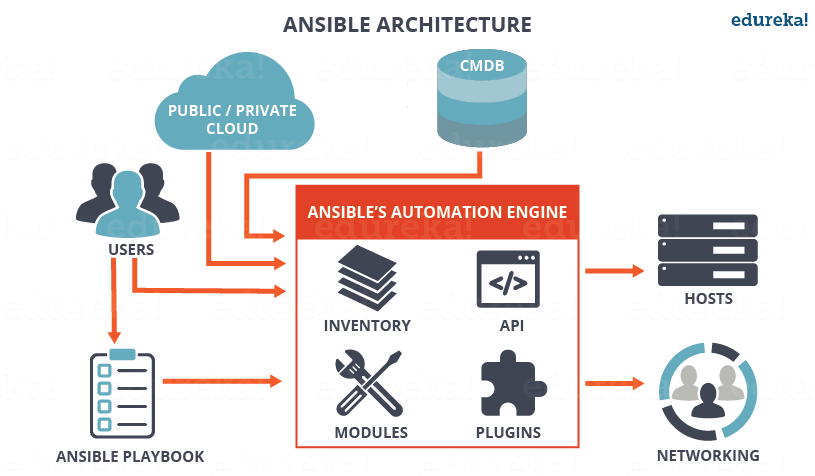
For example, Consider the situation where I want to deploy a new website in place of my existing one. For that, we will remove the existing website, and deploy our new website, and restart  the load balancer or the web cluster if needed. Now, if we just did something like this, users would notice downtime because we have not removed live traffic going to these machines via the load balancer. So, we need some type of pre-task, where we tell the load balancer to put this web server into maintenance mode, so that we can temporarily disable traffic from going to it, as it gets upgraded. Let’s say, I added a block up here, that says a pre-task will be to disable web node in the load balancer.

So, this is our pre-task, where we disable traffic, then down here, we upgrade the node using these various tasks. Finally, we need some type of post-task, which will enable traffic to this web node again, by taking it out of maintenance mode. These tasks can be written in Ansible playbooks and hence it helps to orchestrate the environment.

You will understand the working of Ansible better when you study about its architecture.

## **What is Ansible & its Architecture?**

Ansible architecture is fairly straightforward. Refer to the diagram below to understand the Ansible architecture:



As you can see, in the diagram above, the Ansible automation engine has a direct interaction with the users who write playbooks to execute the Ansible Automation engine. It also interacts with cloud services and Configuration Management Database (CMDB).

The Ansible Automation engine consists of:

* **Inventories:** Ansible inventories are lists of hosts (nodes) along with their IP addresses, servers, databases etc. which needs to be managed. Ansible then takes action via a transport – SSH for UNIX, Linux or Networking devices and WinRM for Windows system.
* **APIs:** APIs in Ansible are used as transport for Cloud services, public or private.
* **Modules:** Modules are executed directly on remote hosts through playbooks. The modules can control system resources, like services, packages, or files (anything really), or execute system commands. Modules do it by acting on system files, installing packages or making API calls to the service network. There are over 450 Ansible-provided modules that automate nearly every part of your environment. For e.g.
  + Cloud Modules like cloudformation which creates or deletes an AWS cloud formation stack;
  + Database modules like mssql\_db which removes MYSQL databases from remote hosts.
* **Plugins:** Plugins allows to execute Ansible tasks as a job build step. Plugins are pieces of code that augment Ansible’s core functionality. Ansible ships with a number of handy plugins, and you can easily write your own. For example,
  + Action plugins are front ends to modules and can execute tasks on the controller before calling the modules themselves.
  + Cache plugins are used to keep a cache of ‘facts’ to avoid costly fact-gathering operations.
  + Callback plugins enable you to hook into Ansible events for display or logging purposes.

There are a few more components in Ansible Architecture which are explained below:

**Networking**: Ansible can also be used to automate different networks. Ansible uses the same simple, powerful, and the agentless automation framework IT operations and development are already using. It uses a data model (a playbook or role) that is separate from the Ansible automation engine that easily spans different network hardware.

**Hosts**: The hosts in the Ansible architecture are just node systems which are getting automated by Ansible. It can be any kind of machine – Windows, Linux, RedHat etc.

**Playbooks:** Playbooks are simple files written in YAML format which describes the tasks to be executed by Ansible. Playbooks can declare configurations, but they can also orchestrate the steps of any manual ordered process, even if it contains jump statements. They can launch tasks synchronously or asynchronously.

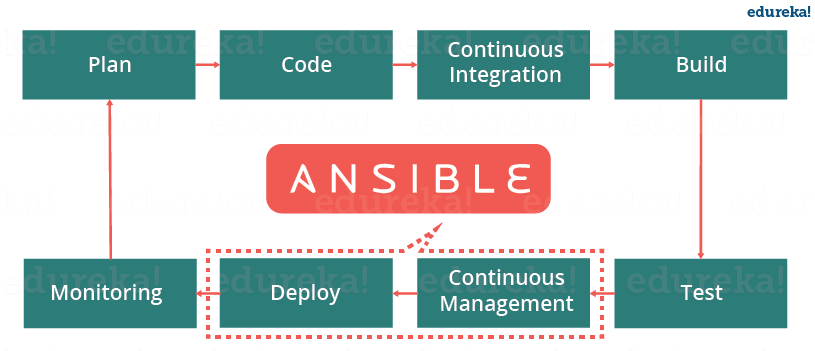
**CMDB** : It is a repository that acts as a data warehouse for IT installations. It holds data relating to a collection of IT assets (commonly referred to as configuration items (CI)), as well as to describe relationships between such assets.

**Cloud:** It is a network of remote servers hosted on the Internet to store, manage, and process data, rather than a local server. You can launch your resources and instances on cloud and connect to your servers.

[**Learn DevOps Now**](https://www.edureka.co/devops/)

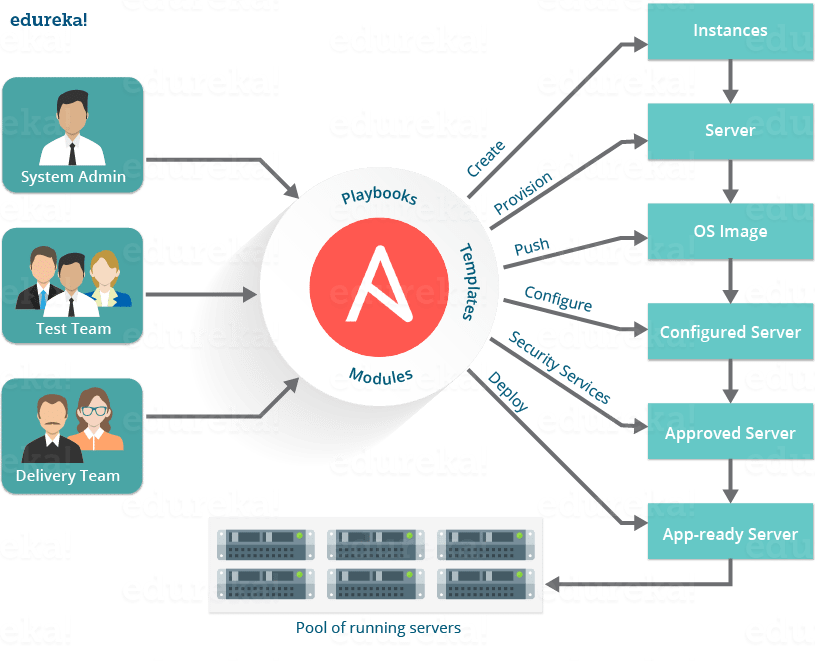
## **Ansible in DevOps**

In DevOps, as we know development and operations work is integrated. This integration is very important for modern test-driven application design. Hence, Ansible integrates this by providing a stable environment to both development and operations resulting in smooth orchestration. Refer to the image below to see how Ansible fits into DevOps:



Let us discuss now how Ansible manages the entire DevOps infrastructure. When developers begin to think of infrastructure as part of their application i.e as Infrastructure as code (**IaC**), stability and performance become normative. Infrastructure as Code is the process of managing and provisioning computing infrastructure (processes, bare-metal servers, virtual servers, etc.) and their configuration through machine-processable definition files, rather than physical hardware configuration or the use of interactive configuration tools. This is where Ansible automation plays a major role and stands out among its peers.

In DevOps, Sysadmins work tightly with developers, development velocity is improved, and more time is spent doing activities like performance tuning, experimenting, and getting things done, and less time is spent fixing problems. Refer to the diagram below to understand how the tasks of sysadmins and other users are simplified by Ansible.



At this point you know how beneficial using Ansible is. So, now let us see a real life example of how NASA has benefited through Ansible.

## **Ansible Case Study – A Real Life Usage by NASA**NASA - What Is Ansible - Edureka

Let us consider the business challenge that was faced by NASA.

NASA needed to move 65 applications from a traditional hardware based data center to a cloud-based environment for better agility and cost savings. The rapid timeline resulted in many applications being migrated ‘as it is’ to a cloud environment. This created an environment which spanned multiple virtual private clouds (VPCs) and AWS accounts that could not be managed easily. Even simple things, like ensuring every system administrator had access to every server, or simple security patching, were extremely cumbersome.

The solution was to leverage Ansible Tower to manage and schedule the cloud environment.

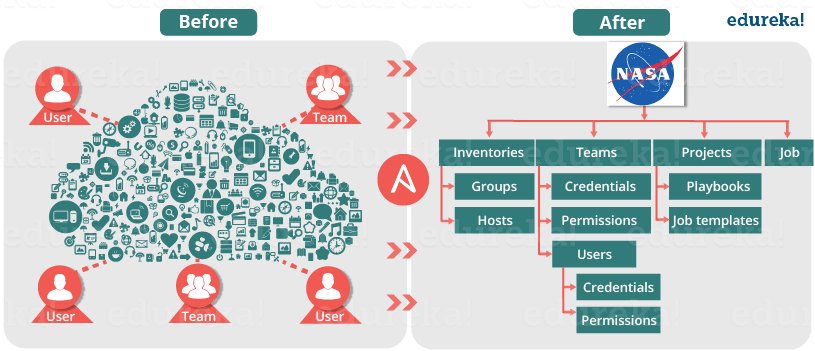
Hence, to solve the problems that NASA had with lack of centralized management and a diverse environment, they evaluated multiple solutions and decided on an implementation of Ansible Tower. NASA is now leveraging Ansible Tower to manage their environment in a very organized and scheduled way.

**How NASA is using Ansible:**

Ansible Tower provided with a dashboard which provided the status summary of all hosts and jobs which allowed NASA to group all contents and manage access permissions across different departments. It also helped to split up the organization by associating content and control permission for groups as well.

Ansible Tower is a web-based interface for managing Ansible. One of the top items in Ansible users’ wishlists was an easy-to-use UI for managing quick deployments and monitoring one’s configurations. Ansible management came up with Ansible Tower in response.

Further, Ansible divided the tasks among teams by assigning various roles. It managed the clean up of old job history, activity streams, data marked for deletion and system tracking info. Refer to the diagram below to understand how Ansible has simplified the work of NASA.



As a result, NASA has achieved the following efficiencies:

• NASA web app servers are being patched routinely and automatically through Ansible Tower with a very simple 10-line Ansible playbook.

• Ansible is also being used to re-mediate security issues and was leveraged to re-mediate OpenSSL issues. This not only saved time but allowed to quickly re-mediate a very daunting security issue.

• Every single week, both the full and mobile versions of www.nasa.gov are updated via Ansible, generally only taking about 5 minutes to do.

• OS level user accounts for mission critical staff are continually checked and created if missing. Now, everyone who needs access has access, even if that means adding or removing a user almost instantly from all servers.

• NASA has also integrated Ansible facts into their CMDB, CloudAware, for better management visibility of entire AWS inventory. As a result, it became possible to organize the inventory of AWS resources in a very granular way that was not possible before.

• Ansible is also used to ensure that the environment is compliant with necessary Federal security standards as outlined by FedRAMP and other regulatory requirements.

**Results:**

As a result of implementing Ansible, NASA is better equipped to manage its AWS environment. Ansible allowed NASA to provide better operations and security to its clients. It has also increased efficiency as a team.

If we see by the numbers:

• Updating**nasa.gov** went from over 1 hour to under 5 minutes

• Security Patching updates went from a multi-day process to 45 minutes

• Achieving near real-time RAM and disk monitoring (accomplished without agents)

• Provisioning OS Accounts across entire environment in under 10 minutes

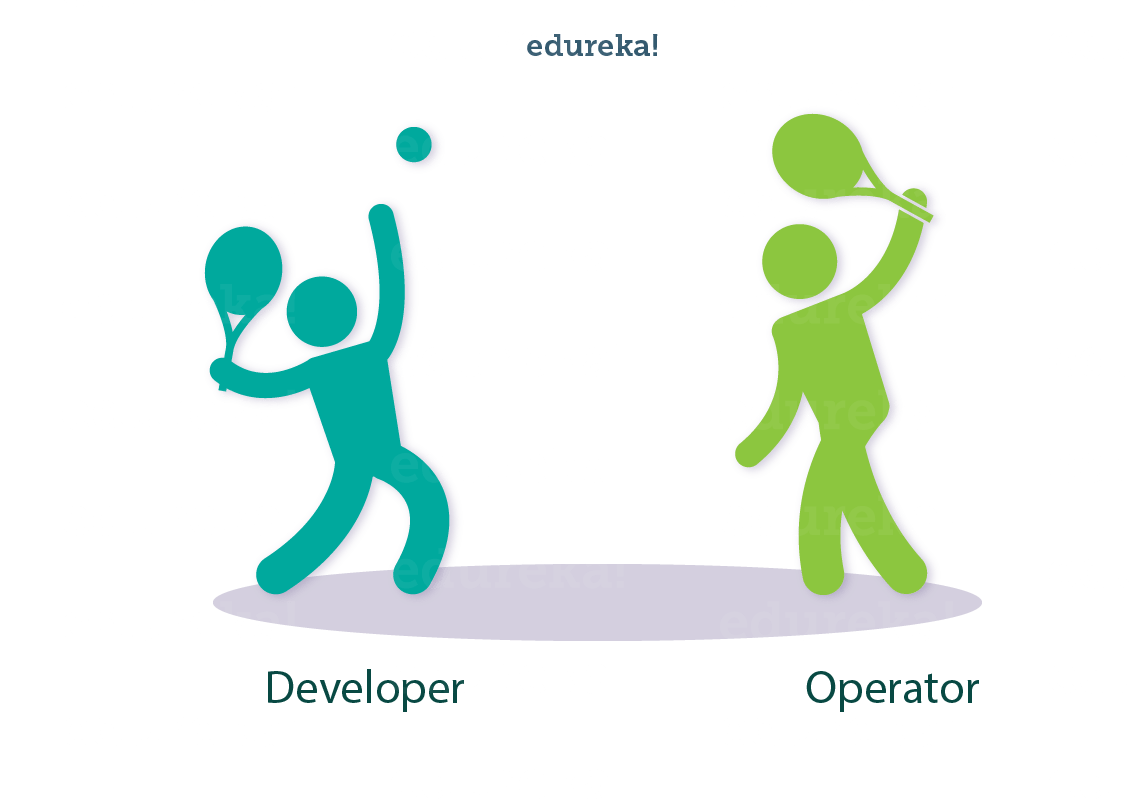
• Baselining standard AMIs (Amazon Machine Image) went from 1 hour of manual configuration to becoming an invisible and seamless background process

• Application stacks set up time reduced from 1-2 hours to under 10 minutes per stack.

I hope you have enjoyed this blog and learned what is Ansible. Now let us learn how to use Ansible using Adhoc commands and playbooks in my next blog on [***Ansible Tutorial***](https://www.edureka.co/blog/ansible-tutorial/).

## **What Is DevOps?**

In these fast-paced times, we see more emphasis being laid on faster delivery of software deployment. Because in order to stay competitive in the market the companies are expected to deploy quality software in defined timelines. Hence the roles of software developer and system admin have become very important. A lot of juggling of responsibilities happens between the two teams. Let us take a look at how do these individuals contribute to the deployment process.



A programmer or a software developer is responsible for developing the software. In simple words he is supposed to develop a software which has:

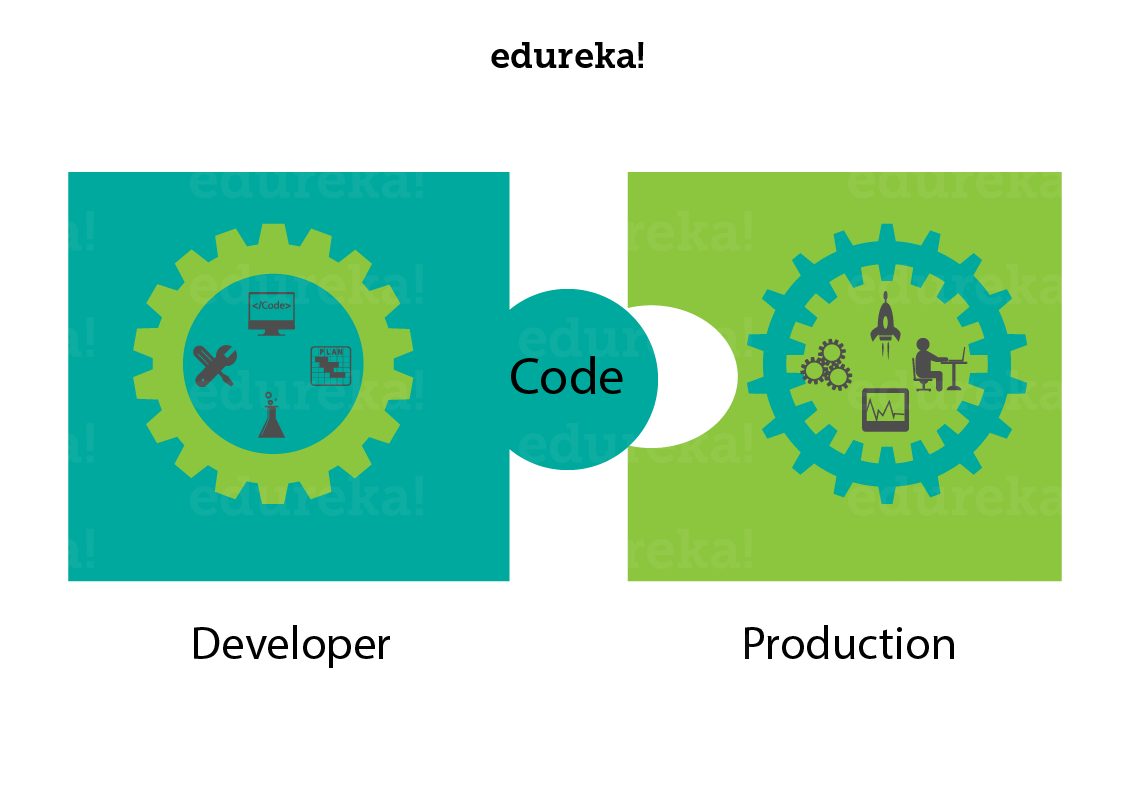
* New features
* Security Upgrades
* Bug Fixes

[**GET TABLEAU CERTIFIED TODAY!**](https://goo.gl/NuurzA)

But a developer may have to wait for weeks for the product to get deployed which is also known as**‘Time To Market’** in business terms. So this delay may put pressure on the developer because he is forced re-adjust his dependent activities like:

* Pending code
* Old code
* New products
* New features

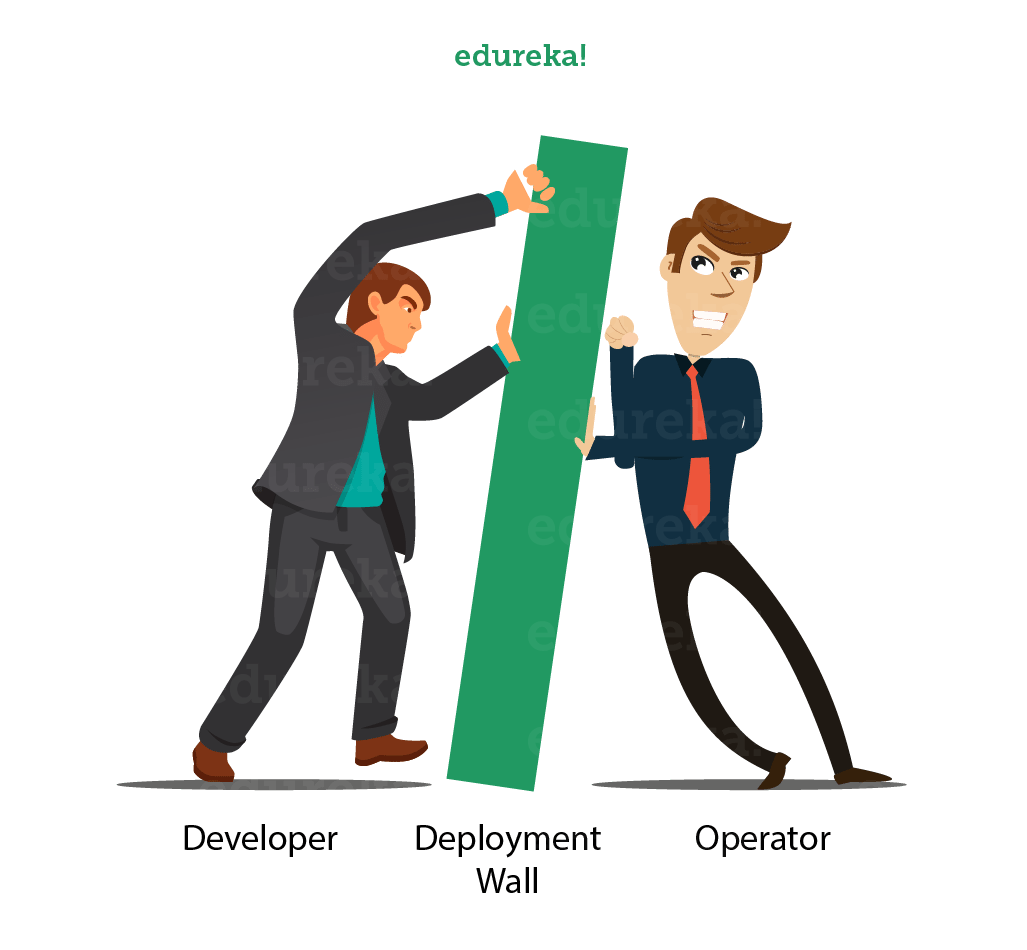
Also when the product is put into the production environment, the product may exhibit some unforeseen errors. This is because the developer writes code in the development environment which may be different from the production environment.



Let us go ahead and take a look at this process from the operations point of view. Now the operations team or the system administrators team is responsible for maintaining and assuring the up time of the production environment. Now as the company invests time and money in more products and services, the number of servers, admins have to take care of also keep growing.

This gives rise to more challenges because the tools that were used to manage the earlier amount of servers may not be sufficient to cater the needs of upcoming and growing number of servers. The operations team also needs to make slight changes to the code so that it fits into the production environment. Hence the need to schedule these deployments accordingly also grows, which leads to time delays.

When the code is deployed the operations team is also responsible to handle code changes or minor errors to the code. At times the operation team may feel pressurised and it may seem like developers have pushed their responsibilities to the operations side of the responsibility wall. As you may come to realise that none of the sides can be held culprit.

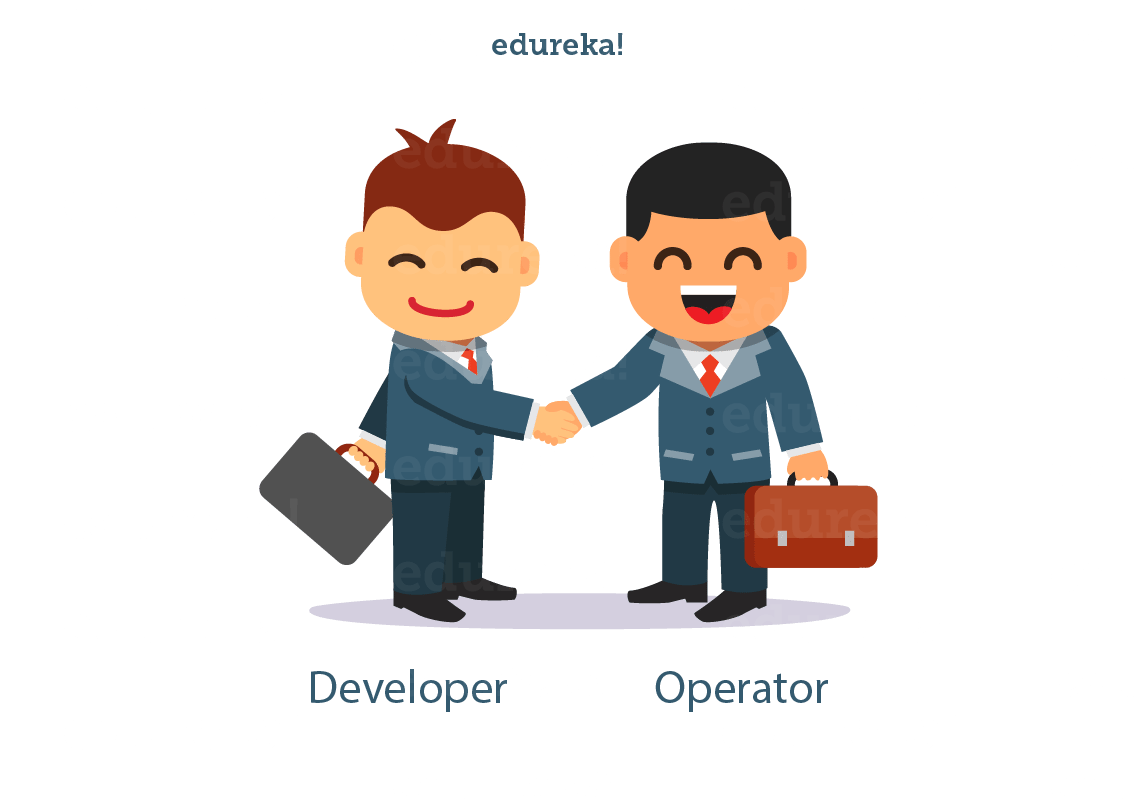


[**Become A DevOps Engineer Now**](https://www.edureka.co/devops)

What if these two teams could work together? What if they:

* could break down silos
* share responsibilities
* start thinking alike
* work as a team

Well, this is what DevOps does, it helps you get software developers and operations in sync to improve productivity. To simply define it with jargon terms. DevOps is the process of integrating Developers and Operations teams in order to improve collaborations and productivity. This is done with automation of workflows and productivity and continuous measurement of application performance.



DevOps focuses on automating everything that lets them write small chunks of code that can be tested, monitored and deployed in hours which is different from writing large chunks of codes that takes weeks to deploy. So this was about DevOps. Let us move ahead and understand what is AWS and how it forms a crucial pairing with DevOps to give you AWS DevOps.

[**More Details About AWS Course**](https://www.edureka.co/cloudcomputing)

## **What Is AWS?**

If you go back a decade, the scenario of handling or more precisely storing data was different. Companies preferred storing data using their private servers. However, with more and better usage of internet, the trend has seen a paradigm shift for companies, as they are moving their data to cloud. This enables companies to focus more on core competencies and stop worrying about the storing and computation. These two points below talk about the significance of cloud:

***Fact:*** Netflix is a popular video streaming service which the whole world uses today, back in 2008 Netflix suffered a major database corruption, and for three days their operations were halted. The problem was scaling up, that is when they realized the need for a highly reliable, horizontally scalable, distributed systems in the cloud. Came in cloud services, and since then their growth has been off the charts.

**Prediction:**Gartner says, By 2020, a Corporate “No-Cloud” Policy Will Be as Rare as a “No-Internet” Policy Today. Interesting, isn’t it?

Since every company has started to adopt the cloud services. It can be claimed that cloud is the talk of the town. And AWS, in particular, is the leading cloud service provider in the market. Let us understand more about it.

### ****AWS****

AWS which stands for Amazon Web Services is an ‘**Amazon.com**‘ subsidiary which offers**cloud-computing services**at very affordable rates, therefore making its customer base strong from small-scale companies like Pinterest (which has just 5 employees) to big enterprises like D-Link.

### What Is Cloud Computing?

It is the use of remote servers on the internet to store, manage and process data rather than a local server or personal computer.

There are basically 3 categories in cloud computing:

### ****IaaS(Infrastructure as a service)****

* IaaS gives you a **server**in the cloud(virtual machine) that you have complete control over.
* In Iaas, you are responsible for managing everything from the Operating System on up to the application you are running.

### ****PaaS(Platform as a Service)****

* With PaaS, you have a combination of flexibility and simplicity.
* Flexible because it can be tailored to the application’s needs.
* Simple as no need for OS maintenance, versions, patches.

### ****SaaS(Software as a Service)****

* A software distribution model in which a third-party provider hosts applications.
* Instead of installing and maintaining software, you simply access it via the Internet.
* Automatic updates reduce the burden on in-house IT staff.

When we refer to AWS, it is more of an **IAAS.**

In case you wish to know about cloud computing in detail refer this link [What Is Cloud Computing?](https://www.edureka.co/blog/what-is-cloud-computing/)

[**Take Me To AWS Architect Course**](https://www.edureka.co/cloudcomputing)

## **AWS DevOps**

AWS is one of the best cloud service provider and DevOps on the other hand is the ‘need of the hour’ implementation of software development lifecycle. Following reason make AWS DevOps a highly popular amalgamation:

### ****AWS CloudFormation****

DevOps teams are required to create and release cloud instances and services more frequently than traditional development teams. AWS CloudFormation enables you to do just that.‘Templates’ of AWS resources like EC2 instances, ECS containers, and S3 storage buckets let you set up the entire stack without you having to bring everything together yourself.

### ****AWS EC2****

AWS EC2 speaks for itself. You can run containers inside EC2 instances. Hence you can leverage the AWS Security and management features. Another reason why AWS DevOps is a lethal combo.

### ****AWS CloudWatch****

This monitoring tool lets you track every resource that AWS has to offer. Plus it makes it very easy to use third party tools for monitoring like Sumo Logic etc

### ****AWS CodePipeline****

CodePipeline is one popular feature from AWS which highly simplifies the way you manage your CI/CD tool set. It lets you integrate with tools like GitHub, Jenkins, and CodeDeploy enabling you to visually control the flow of app updates from build to production.

### ****Instances In AWS****

AWS frequently creates and adds new instances to their list and the level of customisation with these instances allow you make it easy to use AWS DevOps together.

All these reasons make AWS on of the best platforms for DevOps. This pretty much brings us to the end of this AWS DevOps blog. Please let me know in the comments section below, whether you liked the blog or not. In case you wish to know more about AWS and DevOps Integration then this video would serve you purpose:

 Nagios monitors your entire IT infrastructure to ensure systems, applications, services, and business processes are functioning properly. In this Nagios tutorial, I will be covering the below topics:

1. Why We Need Continuous Monitoring?
2. What Is Continuous Monitoring?
3. What Is Nagios?
4. How To Install Nagios?
5. How To Add A Remote Server Using NRPE (Nagios Remote Plugin Executor).

Let’s begin this Nagios tutorial, by understanding why we need Continuous Monitoring because everything exists for a reason. So, let’s try to find out that reason.

## **Why We Need Continuous Monitoring?**

Continuous Monitoring Tools resolve any system errors ( low memory, unreachable server etc. ) before they have any negative impact on your business productivity.

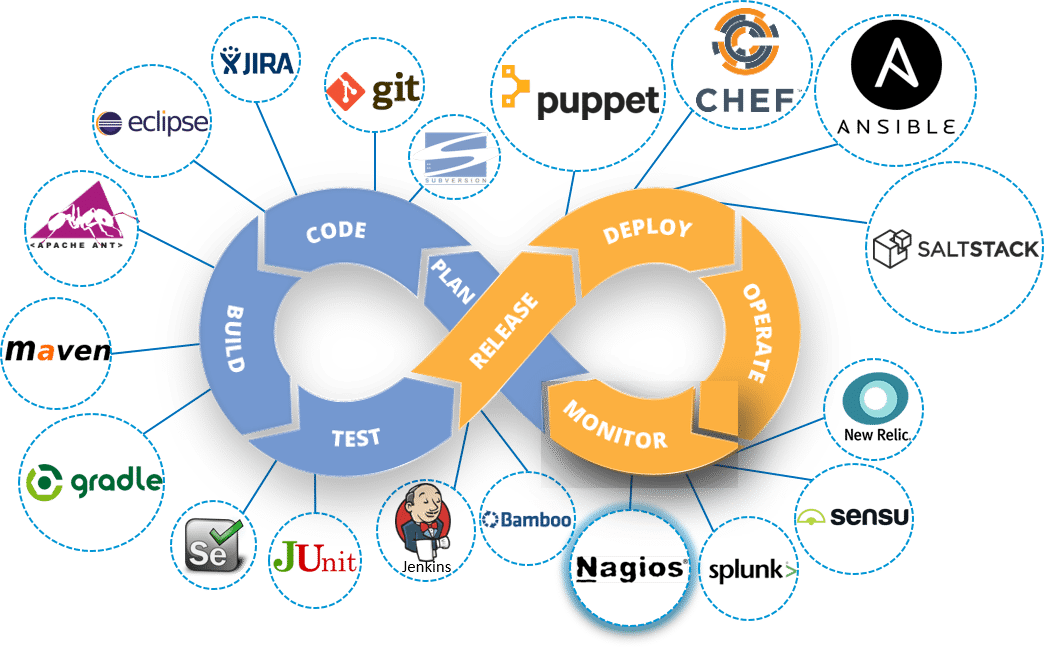
Important reasons to use a monitoring tool are:

* It detects any network or server problems
* It determines the root cause of any issues
* It maintains the security and availability of the service
* It monitors and troubleshoot server performance issues
* It allows us to plan for infrastructure upgrades before outdated systems cause failures
* It can respond to issues at the first sign of a problem
* It can be used to automatically fix problems when they are detected
* It ensures IT infrastructure outages have a minimal effect on your organization’s bottom line
* It can monitor your entire infrastructure and business processes

Yeah, it does a lot of cool work, but what is it?

## **What is Continuous Monitoring?**

Let me first tell you where Continuous Monitoring lies in the DevOps life-cycle, consider the diagram below:



[**Learn Nagios With DevOps Now**](https://www.edureka.co/devops)

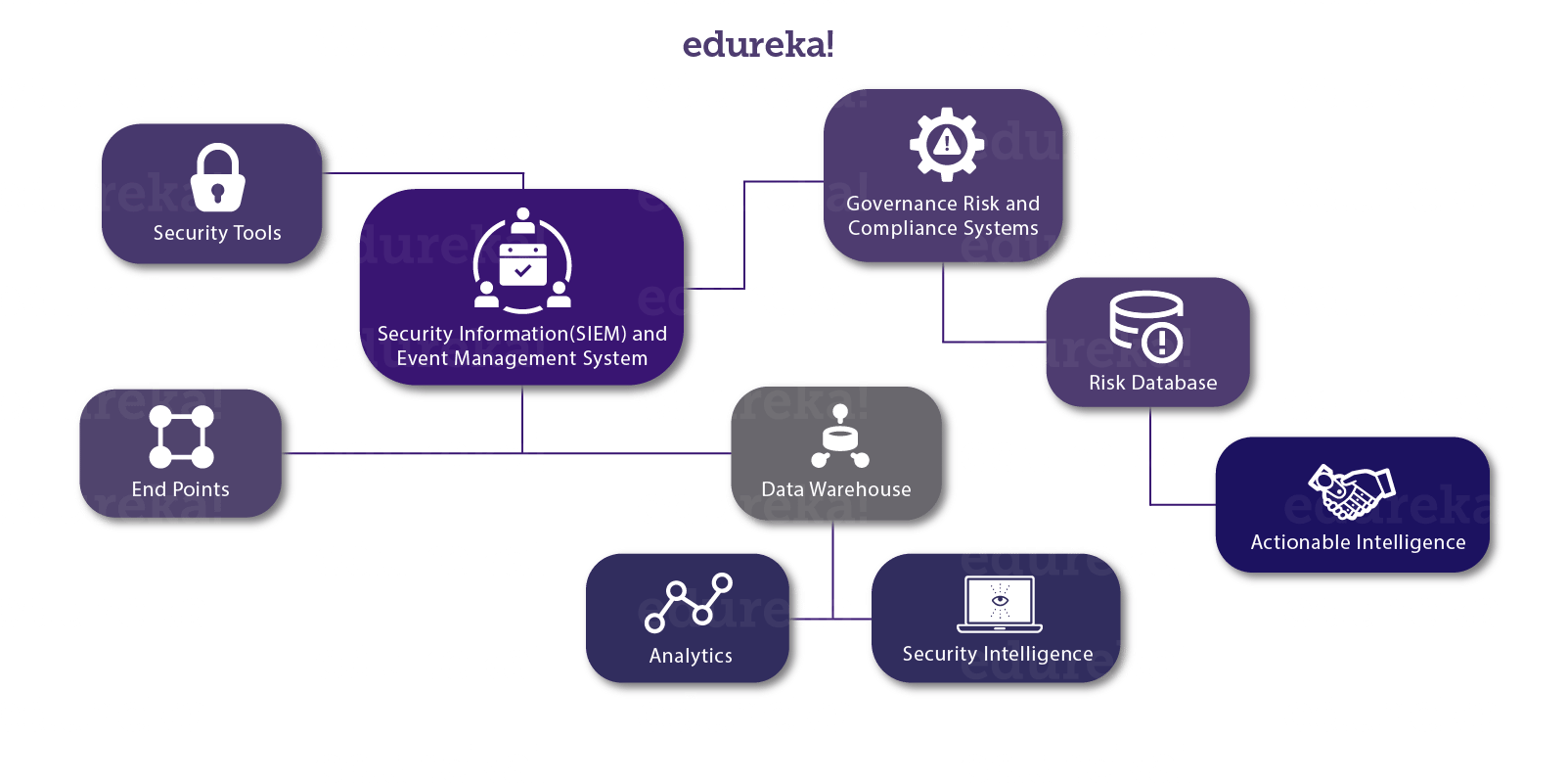
Looking at the diagram you must be thinking this is the last stage in the DevOps lifecycle, but this is not the case. There is no end to a DevOps life cycle, and that is the reason for the infinity symbol. Continuous Monitoring comes into the picture, once the application is deployed on the production servers.

Continuous Monitoring is all about the ability of an organization to detect, report, respond, contain and mitigate the attacks that occur, in its infrastructure.

Continuous Monitoring is actually not new, it’s been around for some time. For years our security professionals are performing static analysis from – system log, firewall logs, IDS logs, IPS logs etc. But, it did not provide proper analysis and response. Today’s Continuous Monitoring approache gives us the ability to aggregate all of the events that I discussed above, co-relate them, compare them and then estimate the organization’s risk posture.

If we take all these pieces and ensure the interlinking between them. This is the crux of Continuous Monitoring.

Let me explain this with a use-case. Consider the diagram below:

Now, let me explain you the above diagram:

1. We have various security tools, like Firewall, IDS, End Point Protection etc. they are connected with a ‘Security Information and Event Management system.
2. In order to achieve Continuous Monitoring, we need to have all the parts talking to each other, let me explain that to you.
3. So we have security tools and series of ‘End Points’,  this can include client and servers, routers, switches, mobile devices and so on.
4. These two groups can then talk to a Security Information and Event Management system (SIEM), through a common language and in more automated fashion.
5. Connected to this SIEM there are two important components, first one is a Data Warehouse. Now to this Data Warehouse, we will connect ‘Analytics’ and ‘Security Intelligence’.
6. Security intelligence (SI) is the information relevant to protecting an organization from external and insider threats as well as the processes, policies and tools designed to gather and analyze that information.
7. This SIEM is also connected to a ‘Governance Risk and Compliance System’ it basically provides dashboarding.
8. To this ‘Governance Risk and Compliance System’  we attach a risk database. This gives us ‘Actionable Intelligence’.
9. Actionable Intelligence is nothing but information that can be acted upon, with the further implication that actions *should*be taken.

So here, we are Monitoring the events on the on going basis and determining what level of risk we are experiencing. With this, we can correlate the events at the SIEM. We can perform Network Behavior and Anomaly Detection at the ‘Analytics Engine’. This is what Continuous Monitoring is all about:

The integration of an organization security tools, the aggregation, normalization and correlation of the data that is produced by the security tools. The analysis of that data, based on the organization’s risk goals and threat knowledge, and near real-time response to the risks identified.

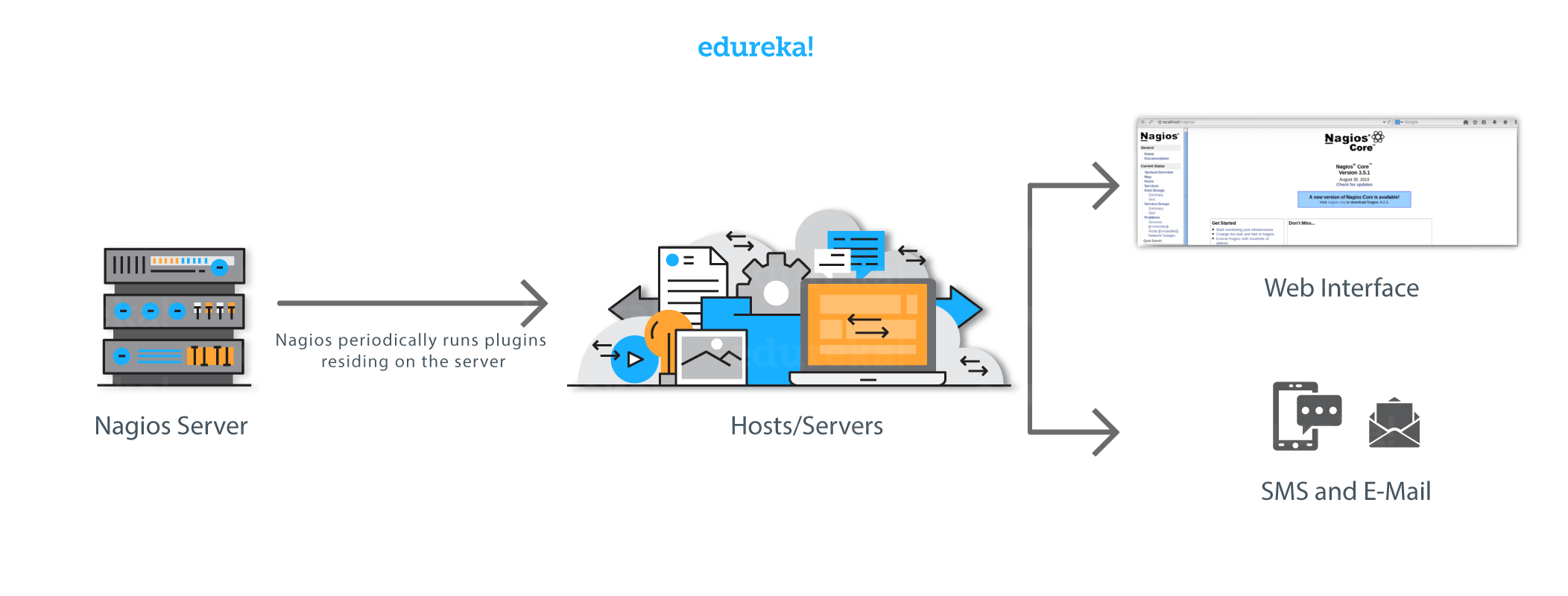
‘If you can’t measure it you can’t manage it’. I hope you know what I am talking about.

Next up in this Nagios tutorial, I will be introducing you to one of the most famous monitoring tool ‘Nagios’.

## **What is Nagios?**

Nagios is used for Continuous monitoring of systems, applications, services, and business processes etc in a DevOps culture. In the event of a failure, Nagios can alert technical staff of the problem, allowing them to begin remediation processes before outages affect business processes, end-users, or customers. With Nagios, you don’t have to explain why an unseen infrastructure outage affect your organization’s bottom line.

Let me explain to you how Nagios works. Consider the diagram below:

Nagios runs on a server, usually as a daemon or a service.

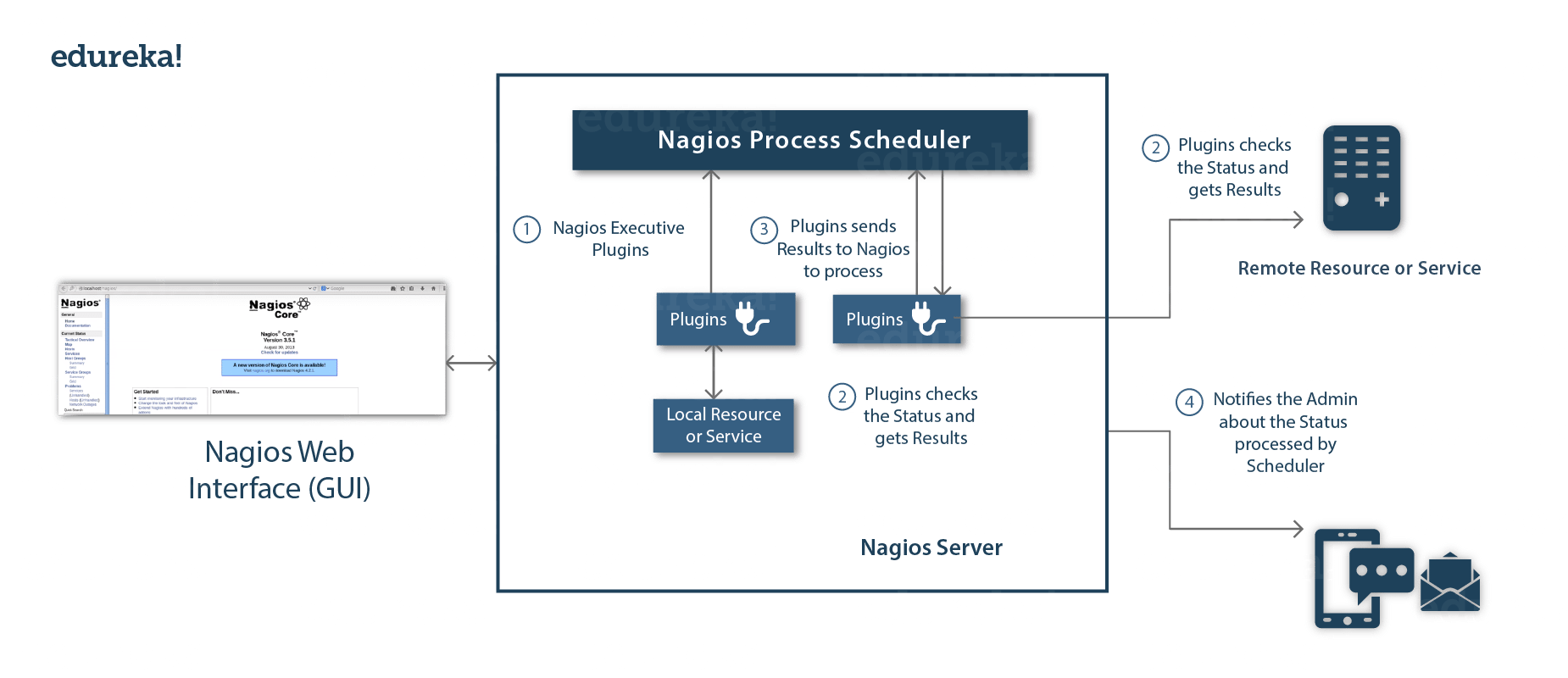
It periodically runs plugins residing on the same server, they contact hosts or servers on your network or on the internet. One can view the status information using the web interface. You can also receive email or SMS notifications if something happens.  
The Nagios daemon behaves like a scheduler that runs certain scripts at certain moments. It stores the results of those scripts and will run other scripts if these results change.

Plugins: These are compiled executables or scripts (Perl scripts, shell scripts, etc.) that can be run from a command line to check the status or a host or service. Nagios uses the results from the plugins to determine the current status of the hosts and services on your network.

Let’s now discuss it’s architecture.

## **Nagios Architecture:**

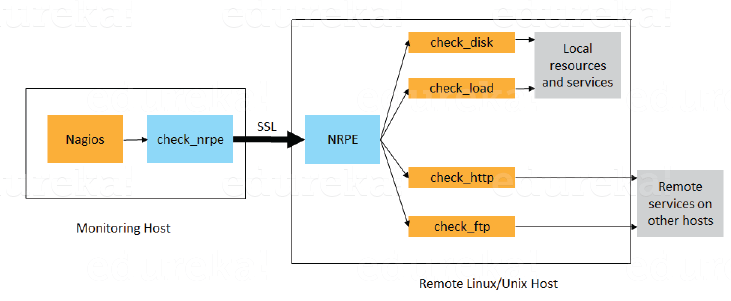
* Nagios is built on a server/agents architecture.
* Usually, on a network, a Nagios server is running on a host, and Plugins interact with local and all the remote hosts that need to be monitored.
* These plugins will send information to the Scheduler, which displays that in a GUI.



I also need to explain you NRPE (Nagios Remote Plugin Executor).

The NRPE addon is designed to allow you to execute Nagios plugins on remote Linux/Unix machines. The main reason for doing this is to allow Nagios to monitor “local” resources (like CPU load, memory usage, etc.) on remote machines. Since these public resources are not usually exposed to external machines, an agent like NRPE must be installed on the remote Linux/Unix machines.

Consider the diagram below:



* The check\_nrpe plugin, resides on the local monitoring machine.
* The NRPE daemon, runs on the remote Linux/Unix machine.
* There is a SSL (Secure Socket Layer) connection between monitoring host and remote host as shown in the diagram above.

Now in this Nagios tutorial, it is time for some **Hands-on**.

Let’s start by installing Nagios Core

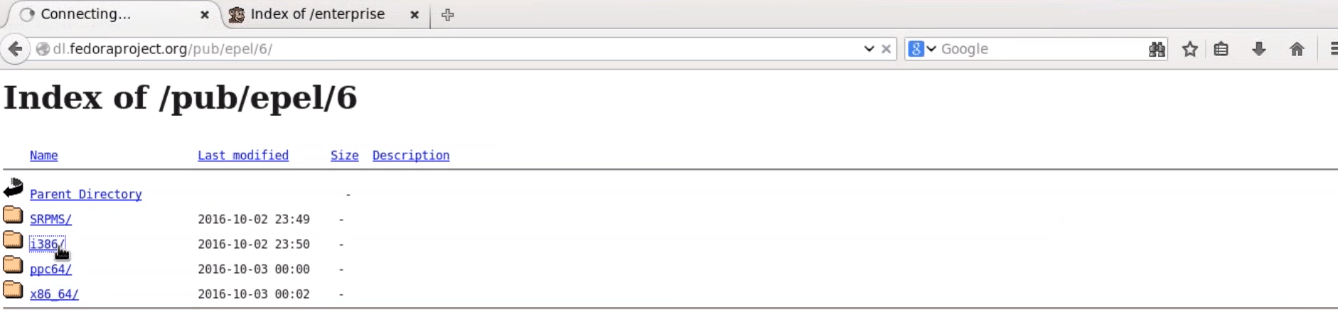
## **Install Nagios Core:**

The complete process to install Nagios can be summarized in four steps:

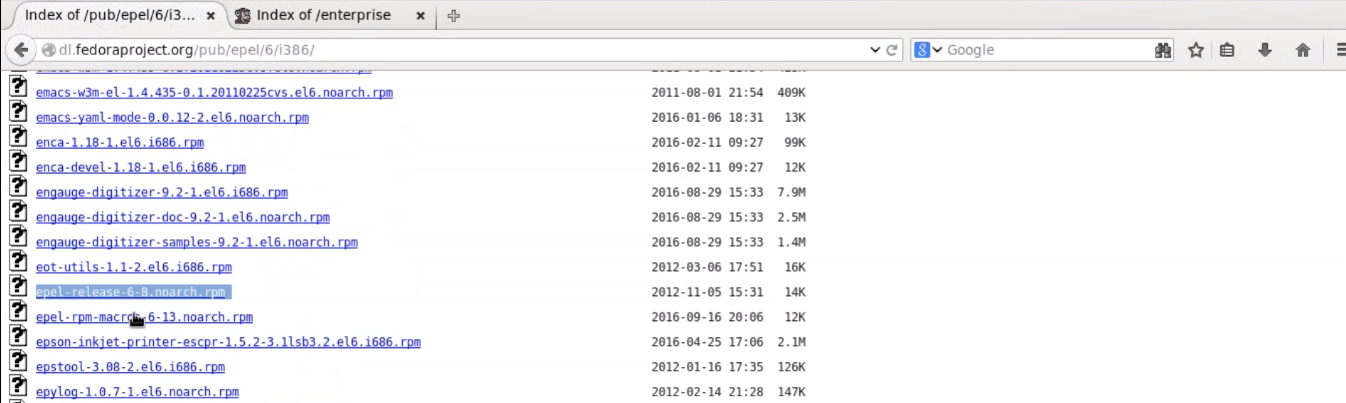
1. Install Required Packages In The Monitoring Server
2. Install Nagios Core, Nagios Plugins And NRPE (Nagios Remote Plugin Executor)
3. Set Nagios Password To Access The Web Interface
4. Install NRPE In Client

## **Step – 1: Install Required Packages On The Monitoring Server:**

Visit the website:[**http://dl.fedoraproject.org/pub/epel/6/**](http://dl.fedoraproject.org/pub/epel/6/)

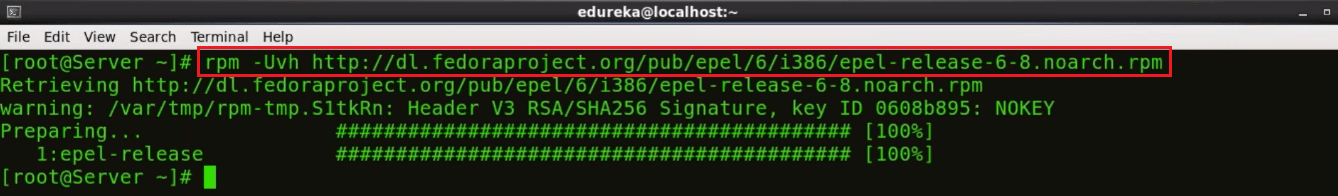


Click on i386, and then you will be redirected to a page.



Since I am using CentOS 6, so I will right click and copy the link location of ‘epel-release-6-8.noarch.rpm‘, as shown in the above screenshot.

Open the terminal and use *rpm -Uvh*command and paste the link.

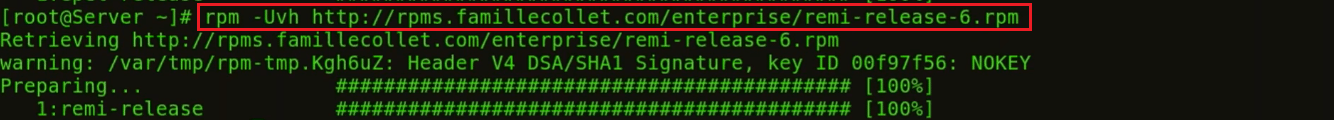


We need to download one more repository, for that visit the website ‘<http://rpms.famillecollet.com/enterprise/>‘



Right-click and copy the link location for ‘remi-release-6.rpm‘

Again open the terminal and use rpm -Uvh command and paste the link.



Fine, so we are done with the pre-requisites. Let’s proceed to the next step.

## **Step – 2: Install Nagios Core, Nagios Plugins And NRPE (Nagios Remote Plugin Executor):**

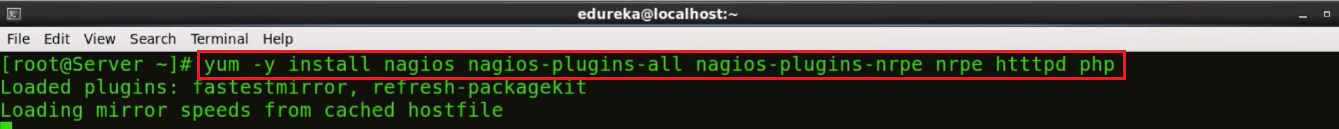
Execute the below command in the terminal:

yum -y install nagios nagios-plugins-all nagios-plugins-nrpe nrpe httpd php

This will install Nagios, Nagios Plugins, Plugins for NRPE, NRPE, Apache and PHP

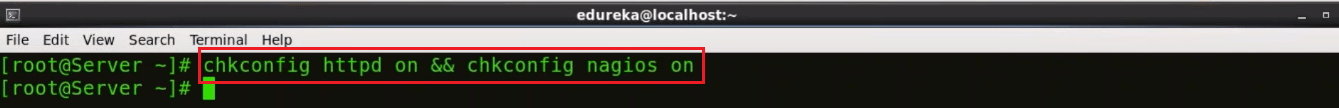
Apache web server is required to monitor the current web server status.

Php is used to process dynamic content of the site date.



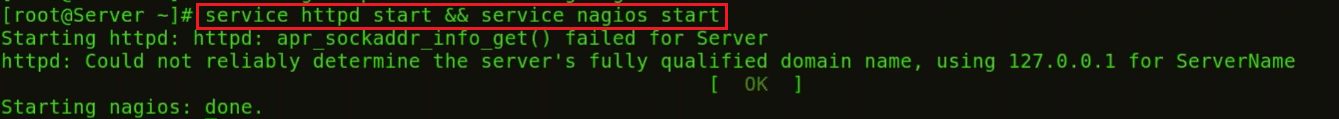
Next, we need to enable Apache and Nagios service:

chkconfig httpd on && chkconfig nagios on



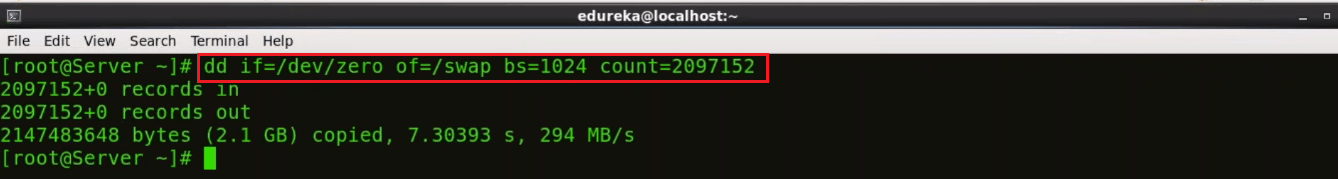
Our next step is to start Nagios and Apache:

service httpd start && service nagios start



Now, I will enable swap memory of at least 1GB. It’s time to create the swap file itself using the dd command:

dd if=/dev/zero of=/swap bs=1024 count=2097152

Swap is basically used to free some, not so frequently accessed information from RAM, and move it to a specific partition on our hard drive.

Now that you have created the swap partition, use the command *mkswap* to setup the swap partition. This is going to prepare the swap file by creating a linux swap area.

mkswap /swap

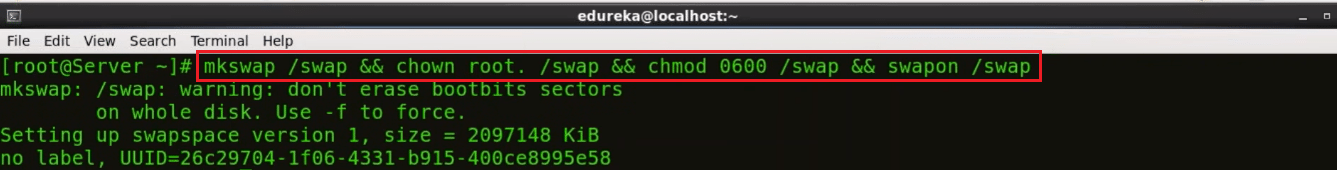
To prevent the file from being world-readable, you should set up the correct permissions on the swap file:

chown root. /swap

chmod 0600 /swap

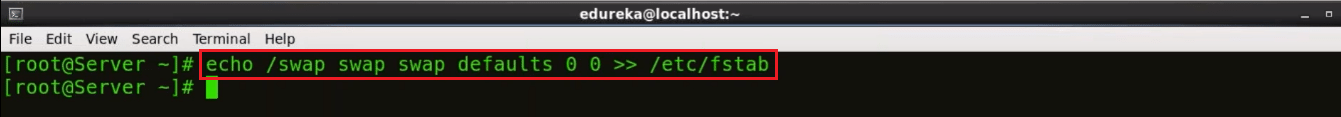
If you see no errors, your swap space is ready to use. To activate it immediately, type:

swapon /swap



This file will last on the virtual private server until the machine reboots. You can ensure that the swap is permanent by adding it to the fstab file.

echo /swap swap swap defaults 0 0 >> /etc/fstab



The operating system kernel can adjust how often it relies on swap through a configuration parameter known as **swappiness**.

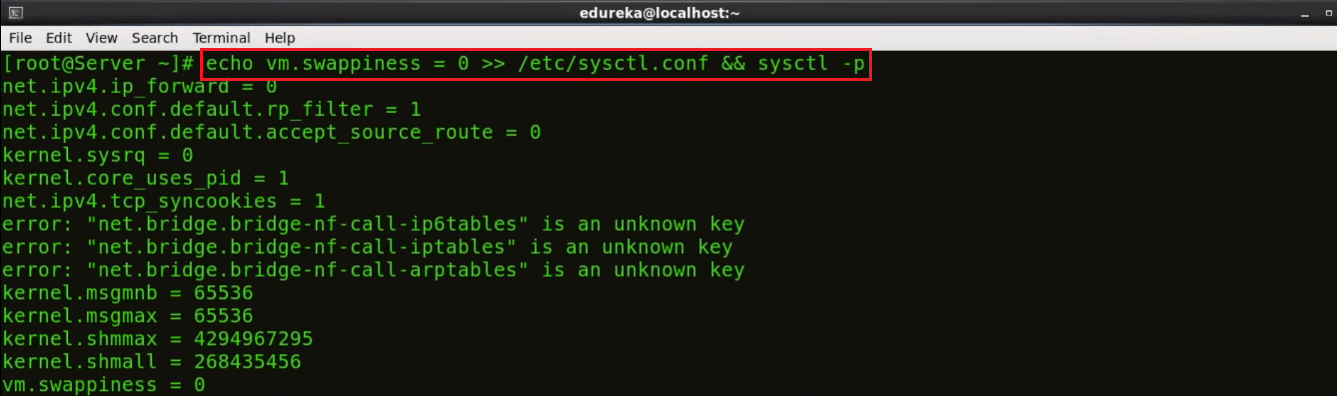
To find the current swappiness settings, type:

cat /proc/sys/vm/swappiness

Swapiness can be a value from 0 to 100. Swappiness near 100 means that the operating system will swap often and usually, too soon. Although swap provides extra resources, RAM is much faster than swap space. Anytime something is moved from RAM to swap, it slows down.

A swappiness value of 0 means that the operating will only rely on swap when it absolutely needs to. We can adjust the swappiness with the sysctl command. To make your VPS automatically apply this setting every time it boots up, you can add the setting to the /etc/sysctl.conf file:

echo vm.swappiness = 0 >> /etc/sysctl.conf && sysctl -p



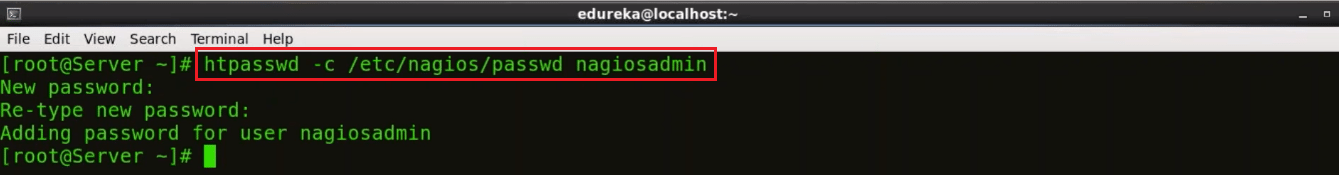
Finally, we are done with the second step.

Let’s proceed further and set Nagios password to access the web interface.

## **Step – 3: Set Nagios Password To Access The Web Interface:**

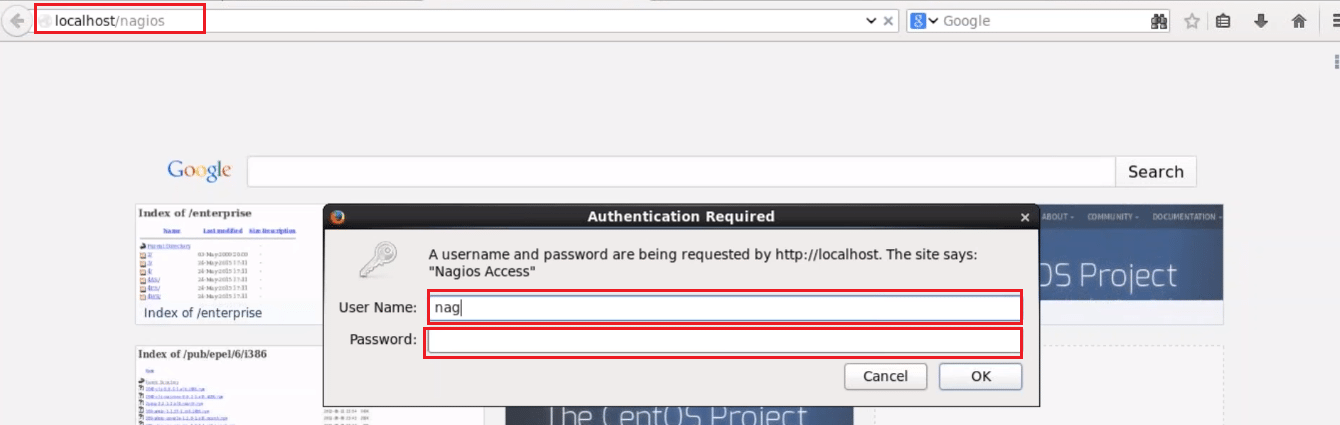
Set the password to access the web interface, use the below command:

htpasswd -c /etc/nagios/passwd nagiosadmin



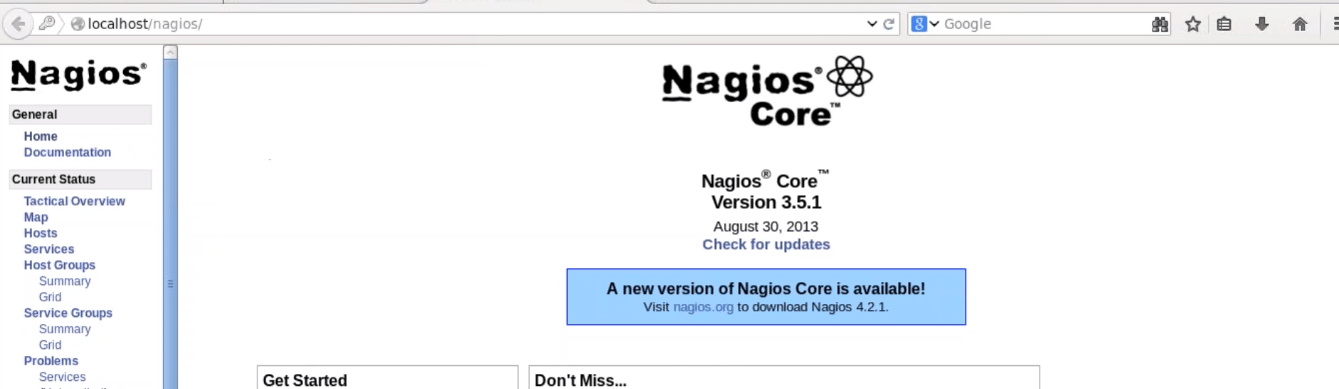
Type the password and confirm it by retyping it.

Now, open the browser. Here, type your public IP or hostname/nagios. Consider the example below:

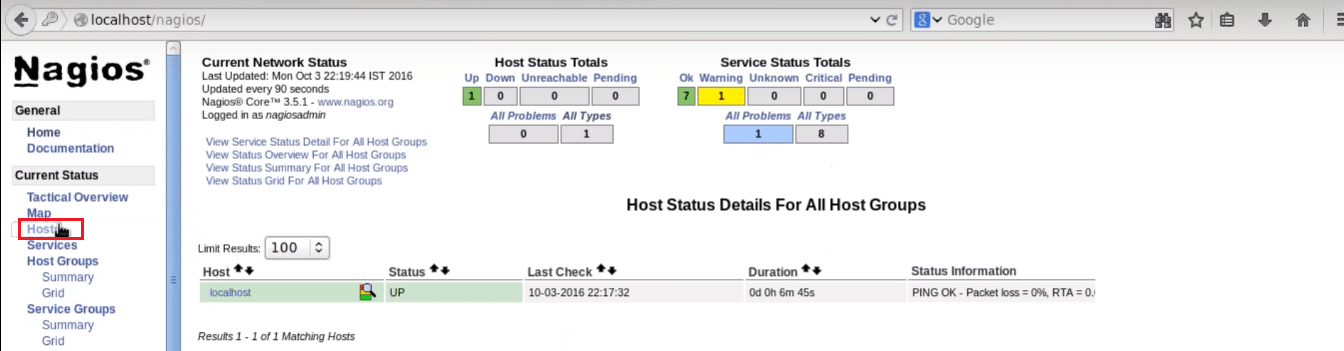


Here, give the user name and password. By default, the user name is nagiosadmin, and password is what you have set in the previous step. Finally, press OK.

After this, you will directed to Nagios Core dashboard.



You can click on hosts and see the what all hosts your Nagios Core is currently monitoring.

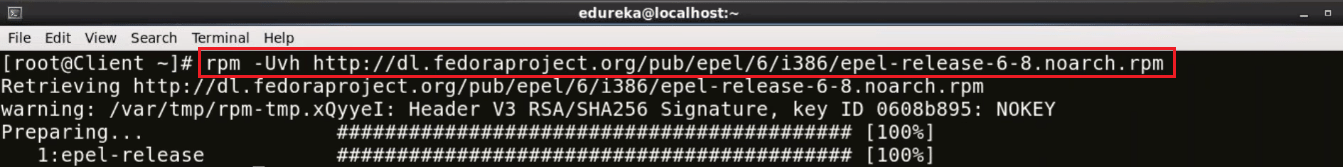


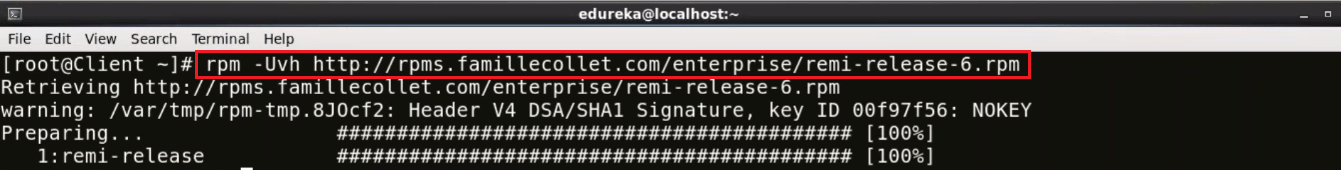
You can notice it is only monitoring one host, i.e. localhost. If I want my Nagios Core to monitor a remote host, I need to install NRPE in that remote host. This brings us to the next step, install NRPE In client/machine that you want Nagios to monitor.

## **Step – 4: Install NRPE In Client:**

Alrighty then, let’s install NRPE in the client machine.

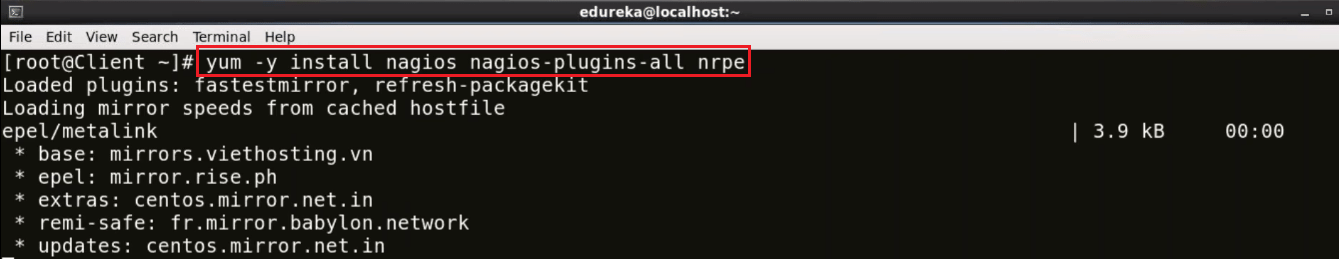
Firstly, you need to install the required packages like I did on my Nagios server machine. So, just execute the same commands, consider the below screenshots:





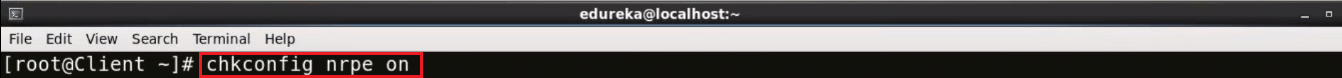
Now install Nagios, Nagios Plugins and NRPE in client:

yum -y install nagios nagios-plugins-all nrpe



Once it is installed, enable the NRPE service:

chkconfig nrpe on

  
Our next step is to edit the, nrpe.cfg file. I will be using the vi editor, you can choose any other editor also:



You need to add the IP address of your monitoring server, in the allowed host line, consider the below screenshot:



Here, the IP address of my monitoring server is 192.168.56.101.

Now, we need to setup firewall rules to allow connection between monitoring server and client.

iptables -N NRPE

-A option is used to append the new rule to the end of a chain. If you want to put it somewhere else in the chain, you can use the -I option which allows you to specify the position of the new rule.

The below command accepts tcp requests on ports 5666.

iptables -I INPUT -s 0/0 -p tcp --dport 5666 -j NRPE

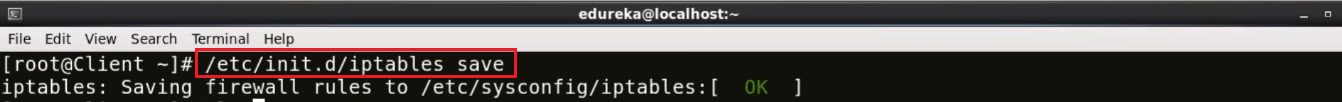
iptables -I NRPE -s 192.168.56.101 -j ACCEPT

iptables -A NRPE -s 0/0 -j DROP

This will basically configure iptables to accept packets from a particular host, in my case – 192.168.56.101, and drops the packets from other hosts.

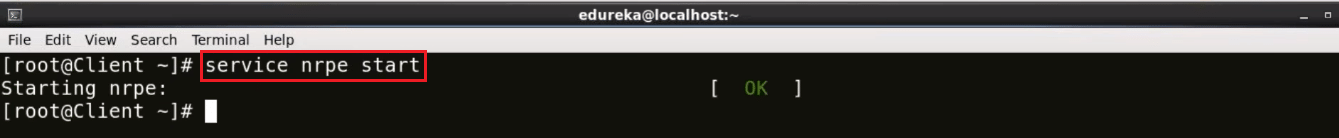
Now, I will save these configurations:

/etc/init.d/iptables save



Start NRPE service now.

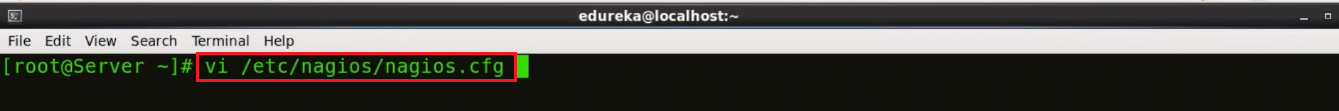
service nrpe start



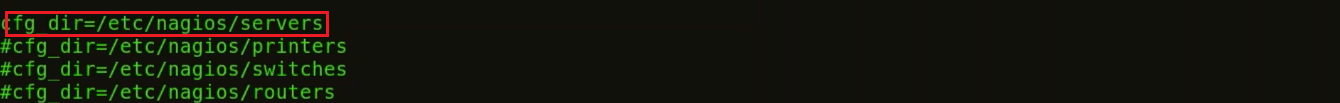
Now go back to the Monitoring server.

Here, I need to edit nagios.cfg file.

vi /etc/nagios/nagios.cfg

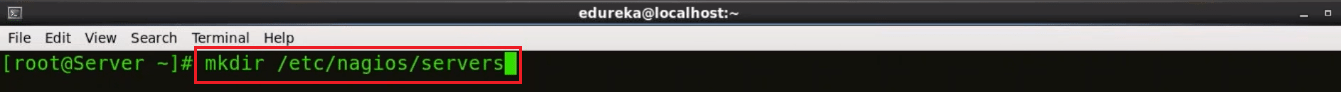


Uncomment the the line – cfg\_dir = etc/nagios/servers



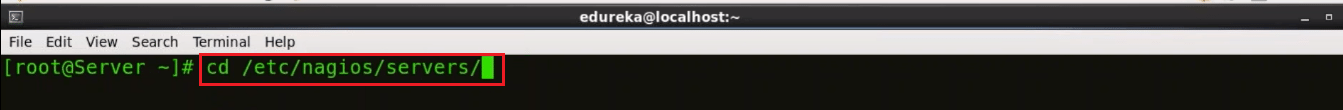
Make ‘server’ directory, for that use mkdir command.

mkdir /etc/nagios/servers/



Change your working directory to servers.

cd /etc/nagios/servers

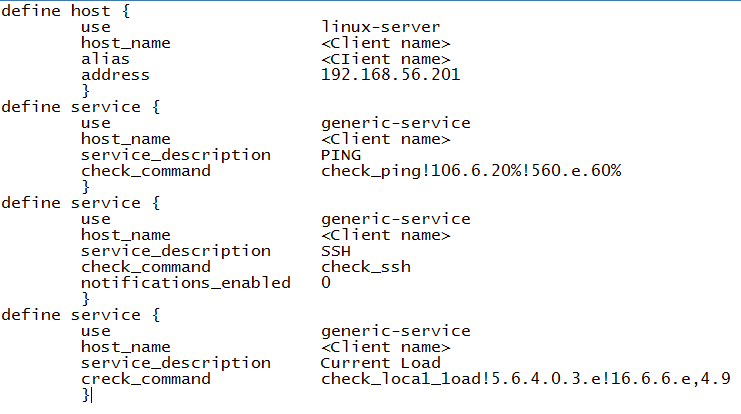


Create a new file in this directory with .cfg extension and edit it. I will name it as client.cfg, and I will be using **vi** editor.

vi /etc/nagios/servers/client.cfg



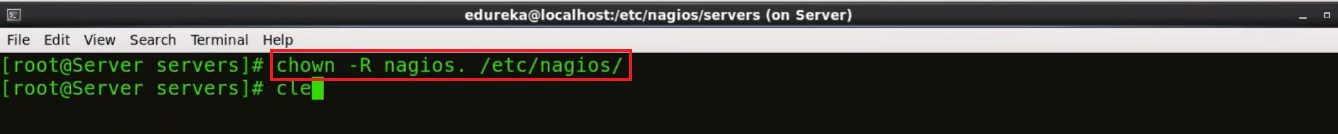
Here add the below lines:

This basically includes the kind of services I want to monitor. Give the hostname of the machine and its ip address which you want Nagios to monitor.

Similarly, you can add number of services that you want to monitor. The same configurations can be used to add ‘n’ number of clients.

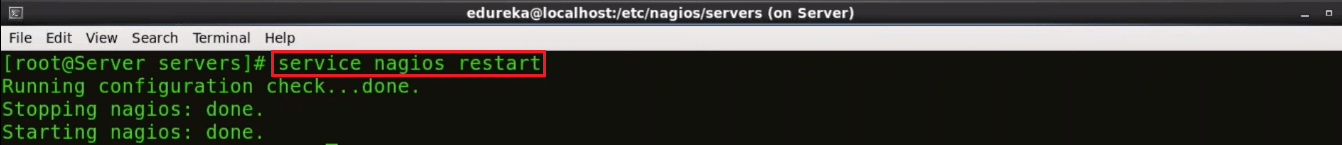
Last step guys, set the folder permissions correctly and restart Nagios.

chown -R nagios. /etc/nagios/



Now, restart Nagios

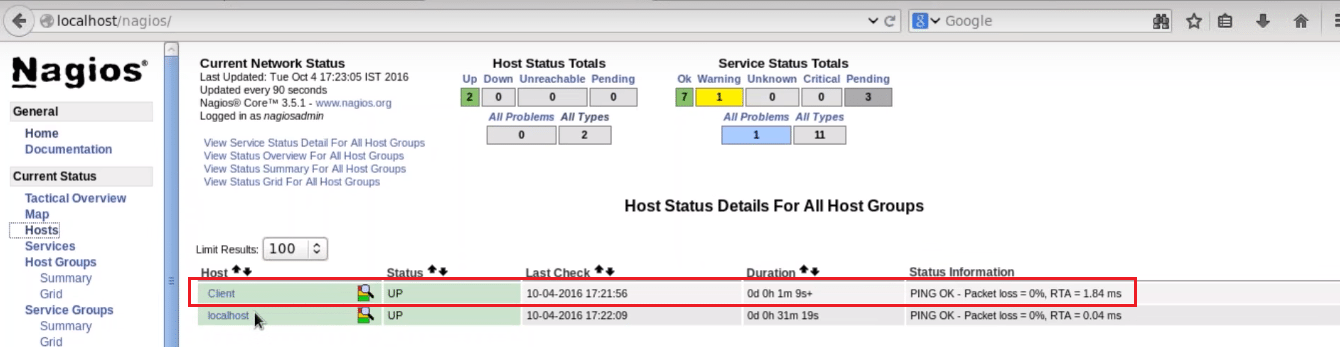
service nagios restart



Open the browser and again type the host name or public ip/nagios/. In my case it is localhost/nagios/.



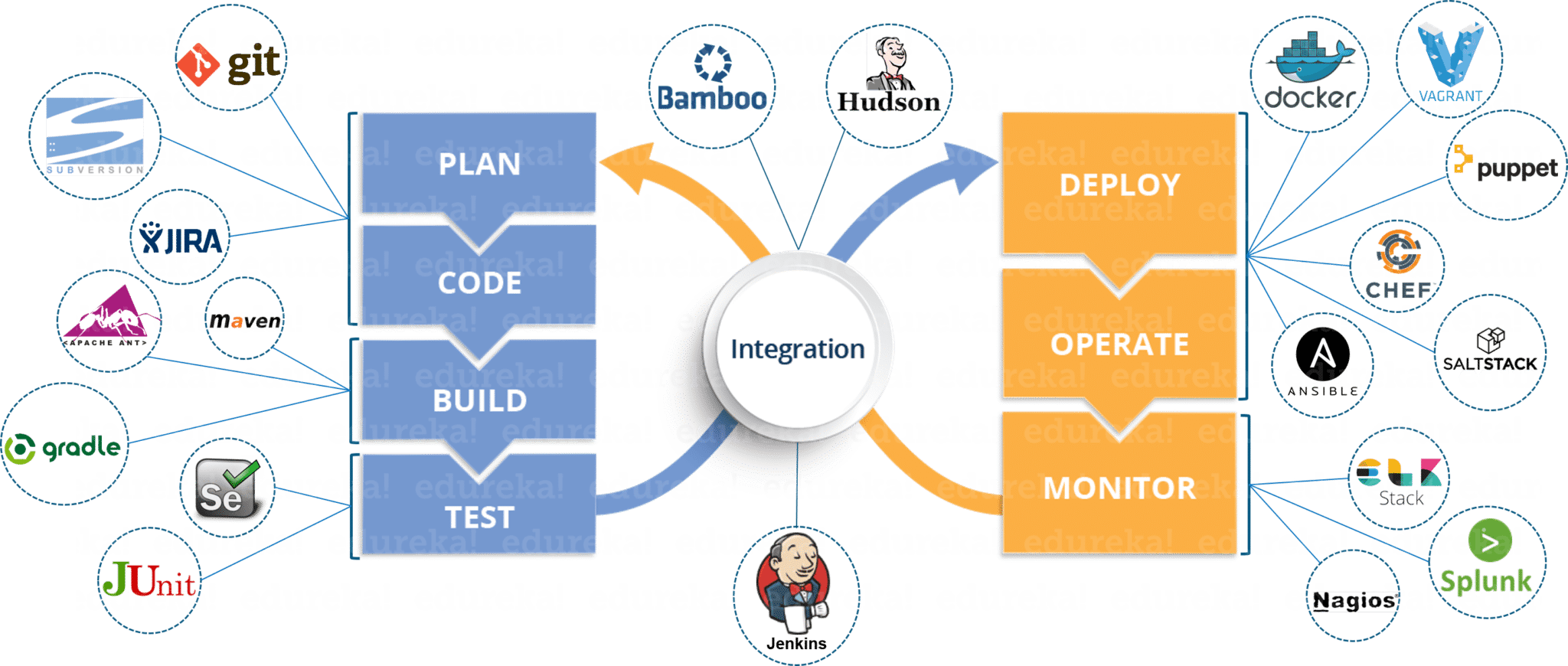
Click on hosts to see all the machines Nagios is currently monitoring.



Here you can notice, it is currently monitoring the client machine (hostname of the machine that I want Nagios to monitor). Basically, we have added a remote host using NRPE.

I hope you have enjoyed reading this Nagios tutorial, I will be coming up with more blogs on Nagios soon.

efore going any further, let’s recap what are the different tools and where they fall in the DevOps lifecycle.

**

Most Used DevOps Tools

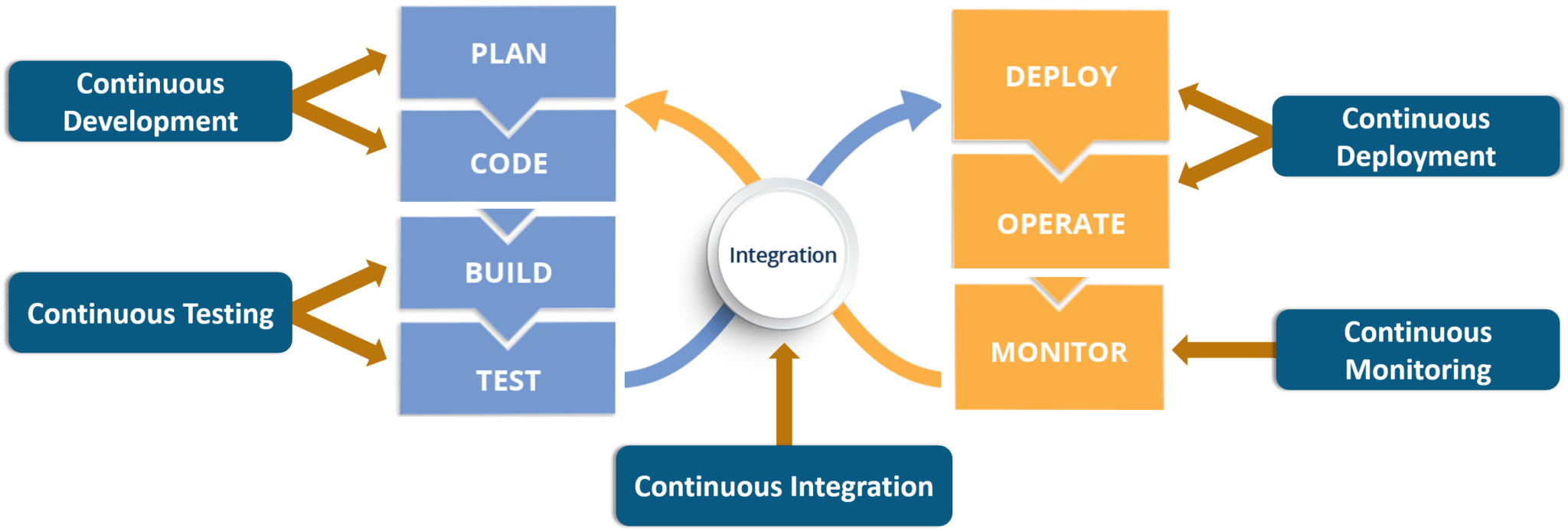
## **DevOps Lifecycle Phases (DevOps Tools)**

Well i’m pretty sure you’re impressed with the above image. But, you might still have problems relating the tools to various phases. Don’t you?

[**Learn More About DevOps Tools**](https://www.edureka.co/devops)

In that case, let’s take a step back and first understand what are the various phases present in the DevOps lifecycle. Below are the 5 different phases any software/ application has to pass through, when developed via the DevOps lifecycle:-

1. Continuous Development
2. Continuous Testing
3. Continuous Integration
4. Continuous Deployment
5. Continuous Monitoring



### ****1. Continuous Development****

This is the phase which involves ‘**planning**‘ and ‘**coding**‘ of the software application’s functionality. There are no tools for planning as such, but there are a number of tools for maintaining the code.

The vision of the project is decided during the ‘planing’ phase and the when they start writing the code, the act is referred to as ‘coding’ phase.

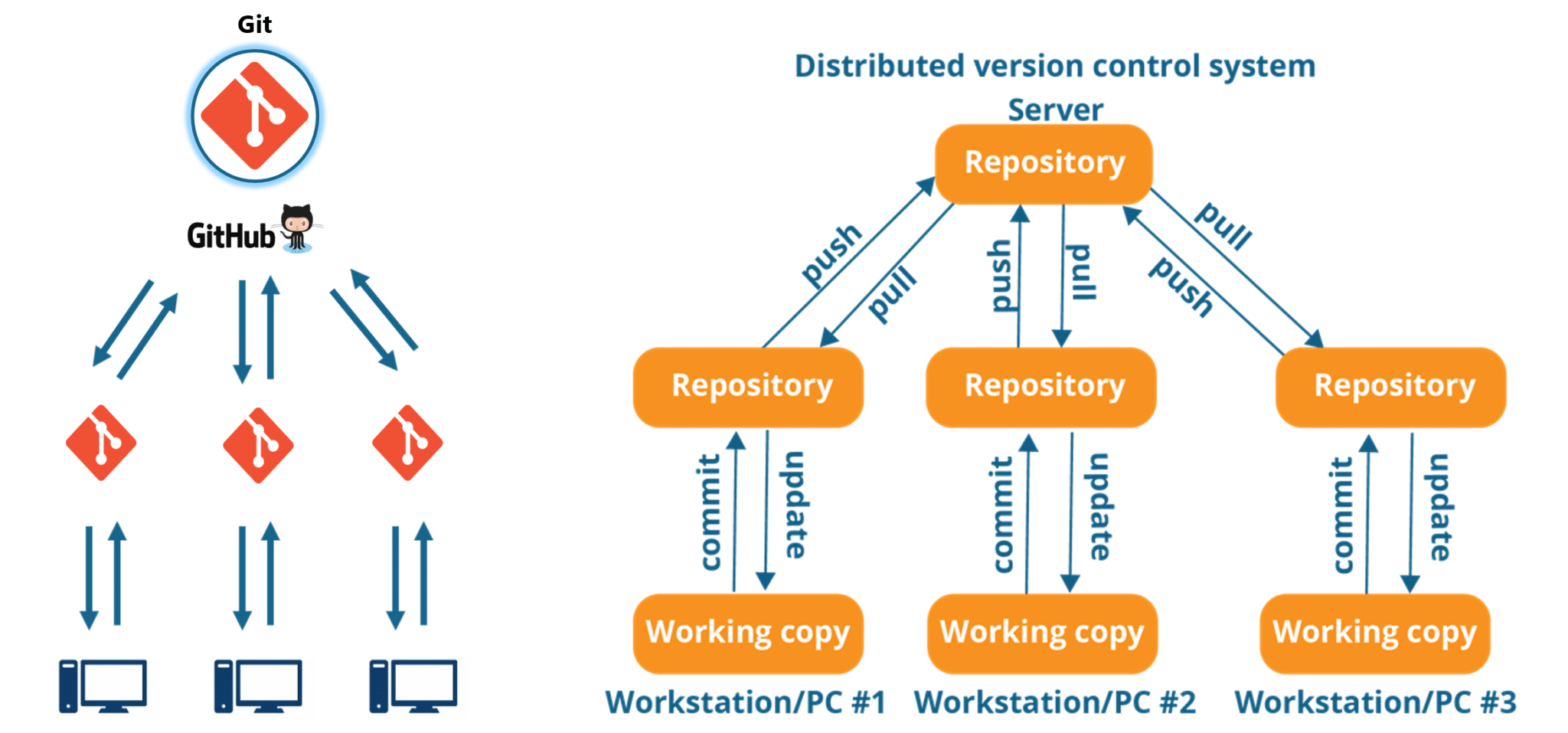
The code can be written in any language, but it is maintained by using **Version Control** tools. These are the Continuous Development DevOps tools. The most popular tools used are: **Git**, **SVN**,**Mercurial**, **CVS**and **JIRA**.

So why is it important to main versions of the code? Which of the Dev vs Ops problem does it solve? Let’s understand that first.

* Versions are maintained (in a central repository), to hold a single source of truth. So that all the developers can collaborate on the ‘latest committed’ code, and even operations can have access to that same code when they plan to make a release.
* Whenever a mishap happens during a release, or even if there are lots of bugs in the code (faulty feature), there is nothing to worry. Ops can quickly rollback the deployed code and thus revert back to the previous stable state.

So, which is my favorite tool? That has got to be **Git & GitHub**. Why? Because **Git** allows developers to collaborate with each other on a Distributed VCS (Version Control System).

Since there is no dependency on the central server, ‘pulls’ & ‘pushes’ to the repository can be made from remote locations. This central repository where the code is maintained is called **GitHub**.



Git is in-fact the world’s leading Version Control system. If you don’t want to take my word for it, you can just google that up. So let’s move on to the next topic in this DevOps tools blog. You can read more about Git from here: [*What is Git?*](https://www.edureka.co/blog/what-is-git/)

### ****2. Continuous Testing****

When the code is developed, it is maddening to release it straight to deployment. The code should first be tested for bugs and performance. Can we agree on that statement?

If yes, then what would be the procedure to perform the tests? Would it be manual testing? Well, it can be, but it is very inefficient. So, what is better? Automation testing? Exactly! Sounds amazing right?

Automation testing is the answer to a lot of cries of manual testers. Tools like **Selenium**, **TestNG**, **JUnit**/ **NUnit** are used to automate the execution of our test cases. So, what are its benefits?

* Automation testing saves a lot of time, effort and labor for executing the tests manually.
* Besides that, report generation is a big plus. The task of evaluating which test cases failed in a test suite gets simpler.
* These tests can also be scheduled for execution at predefined times. Brilliant right?

And the continuous use of these tools while developing the application is what forms the ‘**Continuous Testing**‘ phase during DevOps lifecycle. Which of these is my favorite tool? A combination of these tools actually!

Selenium is my favorite, but **Selenium** without **TestNG** is equivalent to a snake without a poisonous sting, atleast from the perspective of DevOps lifecycle.

Selenium does the automation testing, and the reports are generated by TestNG. But to automate this entire testing phase, we need a **trigger** right? So, what is the trigger? This is where the role of **Continuous Integration** tools like**Jenkins** coming into the picture.

### selenium testng jenkins - devops tools - edureka

You can read more about Selenium and automation testing from this blog of mine: [*What is Selenium?*](https://www.edureka.co/blog/what-is-selenium/) Now, lets move onto the next topic in this DevOps tools blog.

### ****3. Continuous Integration****

This is the most brilliant DevOps phase. It might not make sense during the first cycle of release, but then you will understand this phase’s importance going forward.

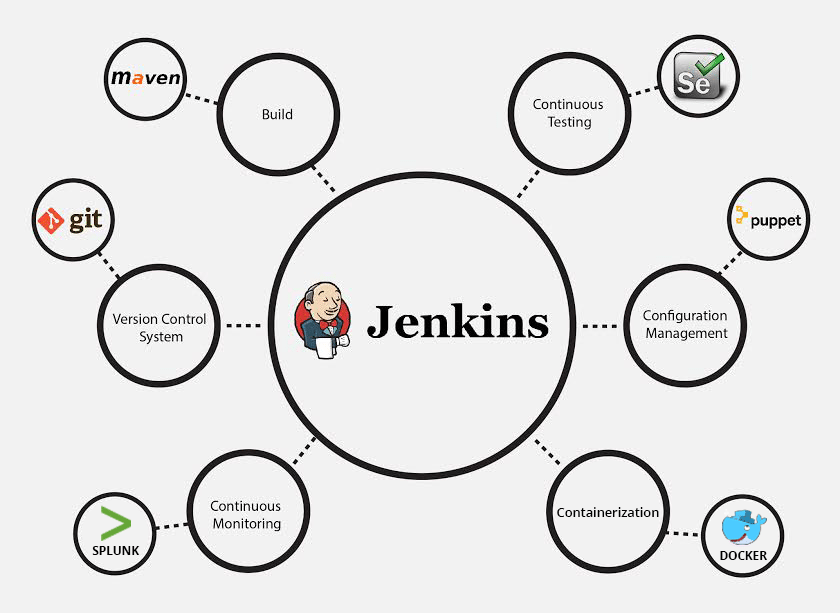
Wait, that is not completely correct. **Continuous Integration** (CI) plays a major role even during the first release. It helps massively to integrate the CI tools with configuration management tools for deployment.

Undisputedly, the most popular **CI** tool in the market is **Jenkins**. And personally, Jenkins is my favorite DevOps tool. Other popular CI tool are **Bamboo** and **Hudson**.

Why do I hold such a high regard for Continuous Integration tools? Because they are the one’s which hold the entire ‘DevOps structure’ together.

It is the CI tools which orchestrates the automation of tools falling under other DevOps lifecycle phases. Be it, Continuous Development tools, or Continuous Testing tools, or Continuous Deployment tools, or even Continuous Monitoring tools, the**Continuous Integration** tools can be integrated with all of them.

* When integrated with Git/ SVN, Jenkins can **schedule jobs** (pulling the code from shared repositories) automatically and make it ready for **builds** and **testing**(Continuous Development). Jenkins can build jobs either at scheduled times of day or when ever there is a commit pushed to the central repository.
* When integrated with testing tools like Selenium, we can achieve Continuous Testing. How? The developed code can be built using tools like Maven/ Ant/ Gradle.  
  When the code is built, then Selenium can ***automate the execution of that code***. How does it automate it? By creating a suite of test cases and executing the test cases one after the other.  
  The role of Jenkins/ Hudson/ Bamboo here would be to schedule/ automate “Selenium to automate test case execution”.
* When integrated with Continuous Deployment tools, Jenkins/ Hudson/ Bamboo can **trigger** the**deployments**planned by configuration management/ containerization tools.
* And finally, Jenkins/ Hudson can be integrated with monitoring tools like Splunk/ ELK/ Nagios/ NewRelic, to continuously **monitor** the **status & performance** of the server where the deployments have been made.



Because CI tools are capable of this and so much more, they are my favorite. Hence my statement: Jenkins is an elementary DevOps tool. You can read more about Jenkins here: [*What is Jenins?*](https://www.edureka.co/blog/what-is-jenkins/)

### ****4. Continuous Deployment****

This (Continuous Deployment) is the phase where action actually happens. We have seen the tools which help us build the code from scratch and also those tools which help in testing. Now it is time to understand why DevOps will be incomplete without **Configuration Management** tools or **Containerization** tools. Both set of tools here help in achieving Continuous Deployment (CD).

#### ****Configuration Management Tools****

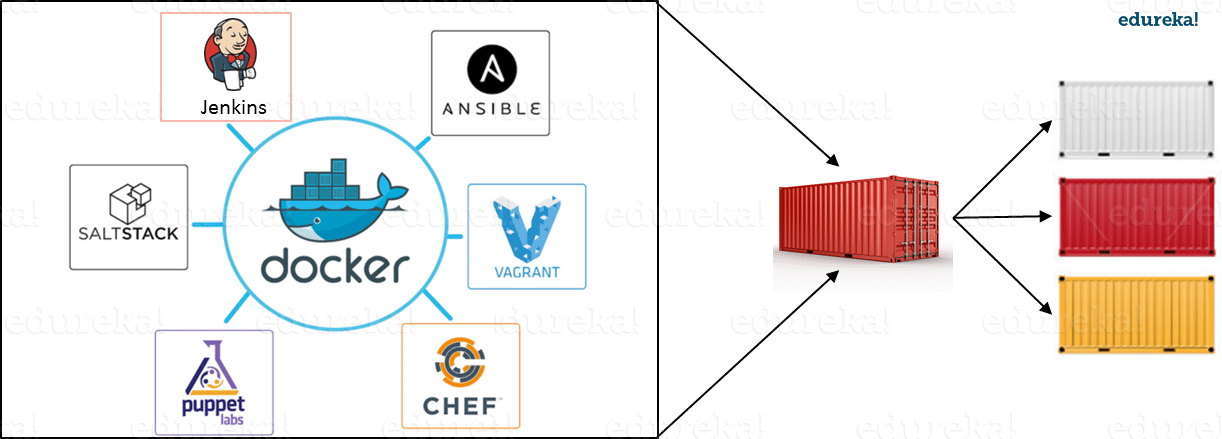
* Configuration Management is the act of establishing and maintaining consistency in an applications’ functional requirements and performance. In simpler words, it is the act of **releasing deployments** to servers, **scheduling updates** on all servers and most importantly keeping the **configurations consistent** across all the severs.
* For this, we have tools like **Puppet**, **Chef**, **Ansible**, **SaltStack** and more. But the best tool here is Puppet. Puppet & the other CM tools work based on the master-slave architecture. When there is a deployment made to the master, the master is responsible for replicating those changes across all the slaves, no matter the number! Amazing right?



* You can read more about Puppet here: [*What is Puppet?*](https://www.edureka.co/blog/what-is-puppet) Now let’s move onto Containerization.

#### ****Containerization Tools****

* Containerization tools are other set of tools which help in **maintaining consistency** across the environments where the application is developed, tested and deployed. It eliminates any chance of errors/ failure in production environment by **packaging** and **replicating** the same dependencies and packages used in development/ testing/ staging environment.
* The clear winner here is **Docker**, which was among the first containerization tool ever. Earlier, this act of maintaining consistency in environments was a challenge because VMs and servers were used, and their environments would have to be managed manually to achieve consitency. ***Docker containers*** threw this challenge up above and blew it out of the water. (Pun intended!)



* Another containerization tool is **Vagrant**. But off-late, a number of cloud solutions have started providing support for container services. **Amazon ECS**, **Azure  Container Service** and **Google Container Engine** are a few of the cloud services that have started radical support for Docker containers. This is the reason why Docker is the clear winner.

You can read more about Docker from here: [*What is Docker?*](https://www.edureka.co/blog/docker-tutorial) So now, let’s move on to the final topic in this DevOps tools blog.

### ****5. Continuous Monitoring****

Well, what is the point of developing an application and deploying it, if we do not monitor its performance. **Monitoring** is as important as developing the application because there will always be a chance of bugs which escape undetected during the testing phase.

Which tools fall under this phase? **Splunk**, **ELK Stack**, **Nagios**, **Sensu**, **NewRelic** are some of the popular tools for monitoring. When used in combination with Jenkins, we achieve Continuous Monitoring. So, how does monitoring help?

* To minimize the consequences of buggy features, monitoring is a big add-on. Buggy features most often tend to cause financial loss. So, all the more reason to perform continuous monitoring.
* Monitoring tools also report failure/ unfavorable conditions before your clients/ customers get to experience the faulty features. Don’t we all prefer this?

Which is my favorite tool here? I would prefer either Splunk or ELK stack. These two tools are major competitors. They pretty much provide the same features. But the way they provide the functionality is where they are different.

Splunk is a propriety tool (paid tool). But, this also effectively means that working on Splunk is very easy. ELK stack however, is a combination of 3 open-source tools: ElasticSearch, LogStash & Kibana. It maybe free, but setting it up is not as easy as a commercial tool like Splunk. You can try both of them to figure out the better for your organization. You can read more about Splunk here: [*What is Splunk?*](https://www.edureka.co/blog/what-is-splunk/)

Well these were the various phases of the DevOps lifecycle and the tools that fit seamlessly in those situations. I hope you understood the application of these DevOps tools in the industry.

Continuity is the heart of the DevOps lifecycle and people often get confused between the terms Continuous Delivery and Continuous Deployment. In this blog on Continuous Delivery vs Continuous Deployment, I’m going to compare these two highly conflicting terms in the industry.

So, before I deep dive into differentiating these terms, let me brief you about DevOps first.

## **What is DevOps?**

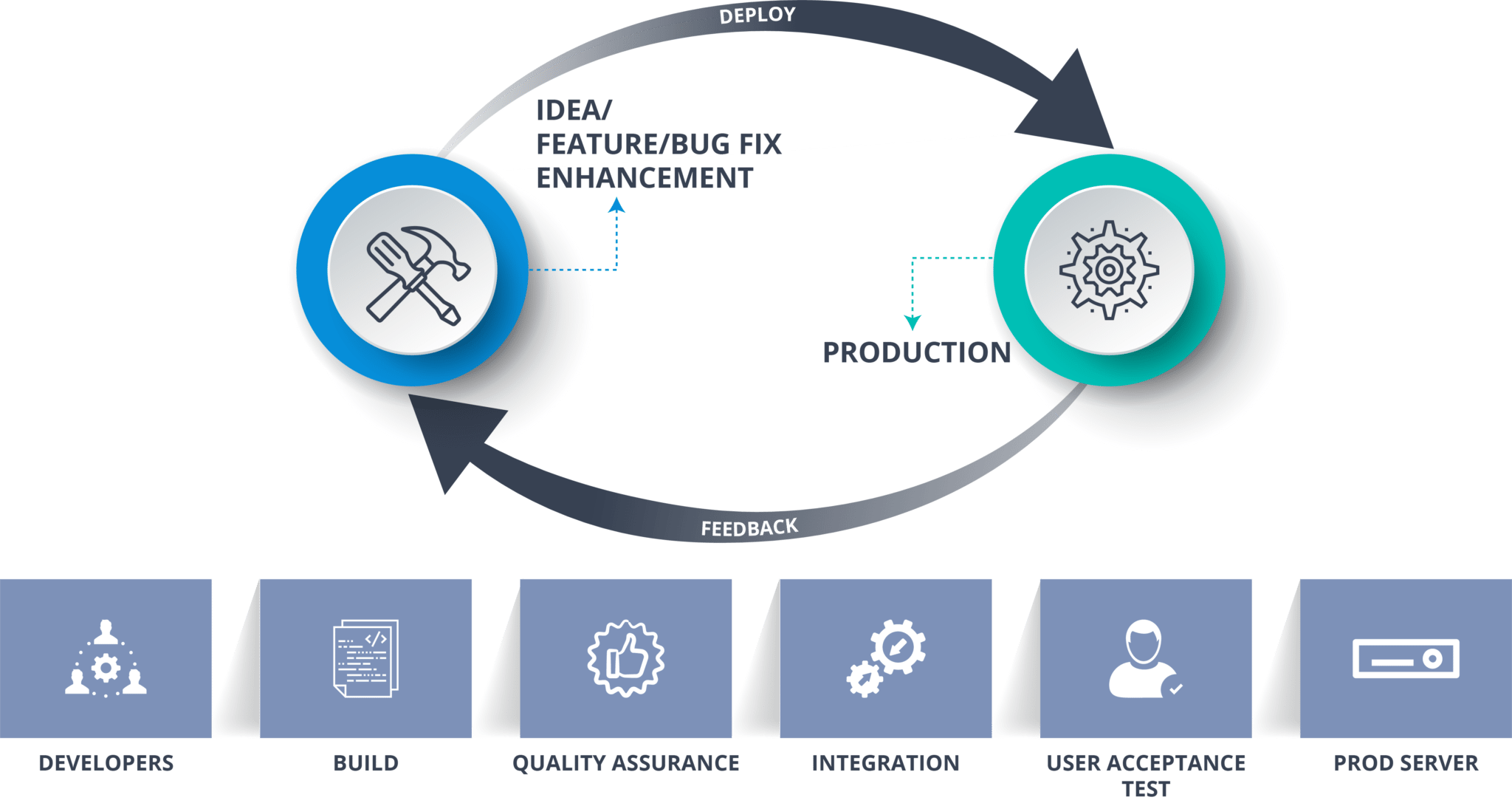
[***DevOps***](https://www.edureka.co/blog/what-is-devops/) is basically a software development strategy which bridges the gap between the dev side and the ops side of the company. In simple terms, you can say that DevOps is, how a developer gets a new feature, an enhancement request, or a change out to production, so that,  when the customers give feedback, the developers can improve based on that feedback.

**But, what are the factors that developers can work on?**

Well, developers can work on mainly 3 important factors:

* The software which is delivered.
* The efficiency and performance of environment to which the software is delivered.
* Making the process of delivering software more efficient, capable and faster at a lower cost.

So, when you deliver software it’s not just delivering it to the production, but,  there’s an entire software delivery lifecycle involved with it.



**Fig 1:** DevOps Methodology – Continuous Delivery vs Continuous Deployment

As you can refer to the diagram, the developers build the software and store it in a software configuration management or version control system. After that, the QA environments assure the quality, and the system integration test, user acceptance test is performed. Finally, when the software passes through all these stages, it reaches the production where the software actually runs and customers interact with it.

Now, the environments at which developers are working might be the same as the environments, customers are working in, but the configurations may differ. So to match these configurations, automated deployment is required.

Automated deployment is the ability to get software deployed in any environment at any given time and Continuous Delivery is that capability to deploy this software to any particular environment at any given time.

Now, you might have heard about large web companies deploying changes every day, all the way onto their prod servers.

How do they do deploy so many times so frequently?

Basically, any change that a developer makes, gets deployed all the way to production and this is nothing but known as Continuous Deployment.

Want to explore more about DevOps?[Learn Now](https://www.edureka.co/devops)

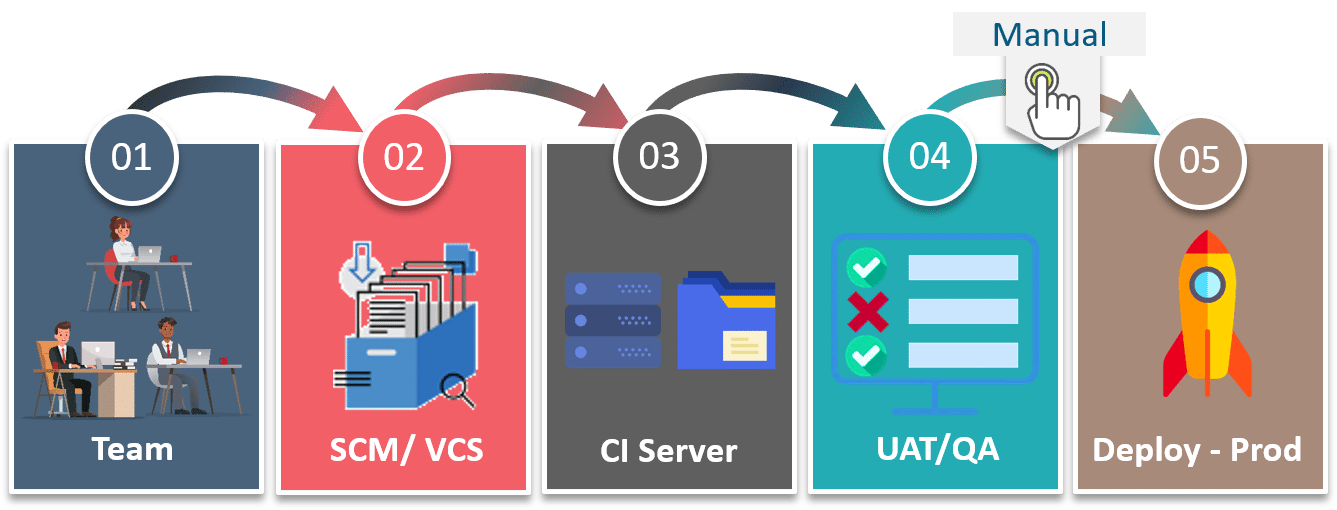
## **Continuous Delivery vs Continuous Deployment**

Now that you have a basic understanding of both the terms let me define Continuous Delivery and Continuous Deployment for you.

### ****Continuous Delivery****

Continuous Delivery is a software development practice where you build software in such a way that the software can be released to the production at any time.

You achieve Continuous Delivery by continuously integrating the products built by the development team, running automated tests on those built products to detect problems and then push those files into production-like environments to ensure that the software works in production.



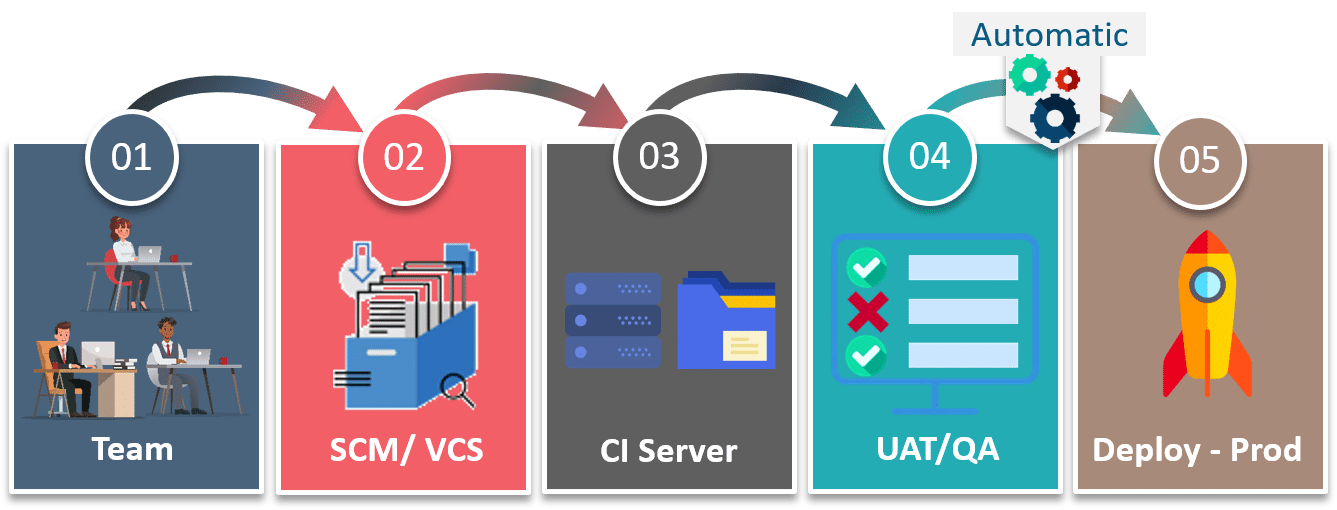
**Fig 2:** Continuous Delivery – Continuous Delivery vs Continuous Deployment

The benefit of continuous delivery lies in the fact that the code is ready to deploy at all times. So, as you can see here the Quality Assurance team tests if each feature is working or not, and then they manually deploy it to production based on the need of business increasing the quality and velocity of the product. So, each and every change is not deployed on to the production.

Now, let me tell how different is Continuous Deployment from Continuous Delivery.

### ****Continuous Deployment****

Continuous deployment means that every change that you make, goes through the pipeline, and if it passes all the tests, it automatically gets deployed into production. So, with this approach, the quality of the software release completely depends on the quality of the test suite as everything is automated.



**Fig 3:** Continuous Deployment – Continuous Delivery vs Continuous Deployment

For example, if you have a function to check various conditions in the test suite, then in Continuous Delivery a manual test can be performed to check the quality of the function. So, if anyone finds out that there could more cases included in that particular function, then it would not be deployed on to production.

But, in the case of Continuous Deployment there would be no approval required, so that function would be automatically deployed on to the prod servers.

So, if we have to summarize in a single line, then, in the world of DevOps using Continuous Deployment, there’s no release approval required. So, the code moves automatically from the developer site to the production site, which is not the case with continuous delivery.

It is always recommended that we should not use, Continuous Deployment as we need to consider many factors before releasing the software like marketing the product before it’s out to the world,  but we must do Continuous Delivery so that we have the capability to deliver the software to any given environment at any given time.

## **Continuous Delivery vs Continuous Deployment | Edureka**

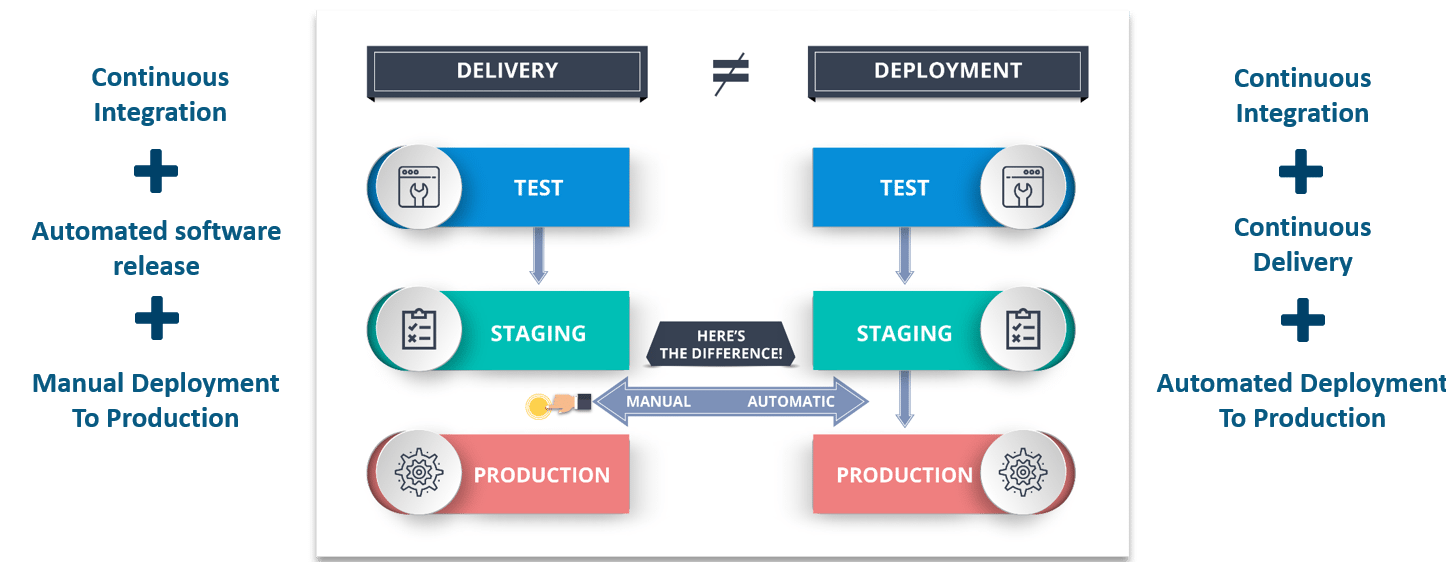
Have you ever wondered if there is any case of Continuous Deployment with Continuous Delivery?

## **Continuous Delivery with Continuous Deployment**

Well, there are many cases where you use both of them. Let me show you one example:

Consider a situation where a customer finds out a bug in the software and sends a feedback to the Dev team.

The Dev team has to recreate that bug as quickly as possible and then fix it. So, in such emergency situations if the team uses continuous delivery then they have the ability to provide that environment which has the same configuration of that of the customer, to deploy the right version of the software to that environment and use automated testing to assist every change. Since the final steps are fully automated Continuous Deployment quickly solves the problem.



**Fig 4:** Difference Between Continuous Delivery & Continuous Deployment – Continuous Delivery vs Continuous Deployment

Therefore, by not skipping any step, but ensuring that fixes are quickly tested and implemented, and delivering the highest possible quality make Continuous Delivery and Continuous Deployment go hand in hand.

So, hopefully, that distinguishes for you that Continuous Delivery is a capability that you must have and a Continuous Deployment is an option you might choose based on the kind of products you’re delivering.

hef, Puppet, Ansible, and SaltStack are all “configuration management” tools, which means they are designed to deploy, configure and manage servers. But do you know which among Chef vs Puppet vs Ansible vs Saltstack is the best tool for IT automation?

I have written this blog to let you know the pros and cons of each of these tools, after which you will be able to decide the most appropriate tool for your organization’s need and environment. These tools are very simple to use yet powerful enough to automate complex multi-tier IT application environments. Therefore, in this “Chef vs Puppet vs Ansible vs Saltstack” blog, I will answer many such questions for you.

You can go through the below table to get an overview of the metrics on which I will be comparing these tools.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Metrics** | **Chef** | **Puppet** | **Ansible** | **Saltstack** |
| Availability | ✔ | ✔ | ✔ | ✔ |
| Ease of Setup | Not very easy | Not very easy | Easy | Not very easy |
| Management | Not very easy | Not very easy | Easy | Easy |
| Scalability | Highly Scalable | Highly Scalable | Highly Scalable | Highly Scalable |
| Configuration language | DSL(Ruby) | DSL(PuppetDSL) | YAML(Python) | YAML(Python) |
| Interoperability | High | High | High | High |
| Pricing (upto 100 nodes) | $13700 | $11200-$19900 | $10,000 | $15,000(approx.) |

These are many more factors on which you can compare these tools. Let’s dig deeper into each tool and understand the difference between Chef vs Puppet vs Ansible vs Saltstack.

[**Learn Devops From Experts!**](https://www.edureka.co/devops)

## **Chef vs Puppet vs Ansible vs Saltstack**

### ****Availability****

Let me compare chef vs puppet vs ansible vs saltstack on the basis of availability. All the tools are highly available which means that there are multiple servers or multiple instance present. Say, if your main master or server goes down, there is always a backup server or the different master to take its place. Let us take a look at each tool one by one:

**Chef –** When there is a failure on the primary server i.e. chef server, it has a backup server to take the place of the primary server.

**Puppet –** It has multi-master architecture, if the active master goes down, the other master takes the active master place.

**Ansible** – It runs with a single active node, called the Primary instance. If primary goes down, there is a Secondary instance to take its place.

**Saltstack** – It can have multiple masters configured. If one master is down, agents connect with the other master in the list. Therefore it has multiple masters to configure salt minions.

### ****Ease of Setup****

When I talk about ease of setup, let me add my personal experience because when I was installing chef, puppet and saltstack, I faced some issues but when I was installing Ansible, it was just like a cake walk. So let us focus on each tool one by one:

**Chef** – Chef has a master-agent architecture. Chef server runs on the master machine and Chef client runs as an agent on each client machine.  Also, there is an extra component called workstation, which contains all the configurations which are tested and then pushed to central chef server. Therefore, it is not that easy.

**Puppet** – Puppet also has a master-agent architecture. Puppet serverruns on the master machine and Puppet clientsruns as an agent on each client machine. After that, there is also a certificate signing between the agent and the master. Therefore, it is also not that easy.

**Ansible** – It has only master running on the server machine, but no agents running on the client machine. It uses ssh connection to login to client systems or the nodes you want to configure. Client machine VM requires no special setup, hence it is faster to setup!

**Saltstack** – Here Server is called as salt master and clients are called as salt minions which run as agents in the client machine.

Apart from this “Chef vs puppet vs Ansible vs Saltstack” blog, if you want to get trained from professionals on these technologies, you can opt for a structured training from edureka! Click below to know more.

[**Become Devops Certified Professional!**](https://www.edureka.co/devops)

### ****Management****

Before I explain the difference between these tools on the basis of management, let me tell you puppet and chef follows pull configurations & Ansible and Saltstack follows push configuration. You must be wondering what are these configurations? In push configuration, all the configurations present in the central server will be pushed to the nodes whereas, in the pull configuration, the slave nodes will automatically pull all the configurations from the central server without any commands.

**Chef** – You need to be a programmer to manage the configurations as it offers configurations in Ruby DSL. Client pulls the configurations from the Server.

**Puppet** – Not very easy to manage the configurations as it uses its own language called Puppet DSL(Domain Specific Language). Client pulls the configurations from the Server. It is quite system-administrator oriented and there is non-immediate remote execution.

**Ansible** – Easy to learn to manage the configurations as it uses YAML i.e. Yet Another Markup Language which closely resembles English. Server pushes configurations to all the nodes. Good for real-time application and there is immediate remote execution.

**Saltstack** – Easy to learn to manage the configurations as it uses YAML as well. Server pushes configurations to all the clients. Immediate remote execution

### ****Scalability****

All the four tools are highly scalable. Suppose if you need to configure around 50 nodes today, and tomorrow say 500. Not a problem with these tools. It can handle large infrastructure, you just need to specify the IP address and the hostname of the nodes that you want to configure and rest of the task will be handled by these tools. Therefore, all these tools are highly scalable.

### ****Configuration Language****

**Chef** – Chef uses Ruby Domain Specific Language (Ruby DSL). It has a steep Learning Curve and its developer oriented.

**Puppet** – Puppet uses its own puppet Domain Specific Language (Puppet DSL). It is not very easy to learn and its system administrator oriented.

**Ansible** – Ansible uses YAML i.e Yet Another Markup Language (Python). It is quiteeasy to learn and its administrator oriented. Python is inbuilt into most Unix and Linux deployments nowadays, so setting the tool up and running is quicker.

**Saltstack** – Salstack also uses YAML (Python). It is again easy to learn and administrator oriented.

Next, let us move forward and compare chef vs puppet vs ansible vs saltstack on the basis of interoperability.

### ****Interoperability****

In these tools, master or main server or you can also say control machine, has to be on Linux/Unix but their slaves or the nodes that they have to configure can be on windows. Let us have a look at each tool one by one:

**Chef** – Chef Server works only on Linux/Unix but Chef Client and Workstation can be on windows as well.

**Puppet** – Puppet Master works only on Linux/Unix but Puppet Agent also works on windows.

**Ansible** – Ansible supports windows machines as well but the Ansible server has to be on Linux/Unix machine.

**Saltstack** – Salt Master works only on Linux/Unix but Salt minions can work on windows as well.

Apart from this “Chef vs puppet vs Ansible vs Saltstack” blog, if you want to get trained from professionals on these technologies, you can opt for a structured training from edureka! Click below to know more.

[**Get Certified Now!**](https://www.edureka.co/devops)

### ****Pricing****

The enterprise cost for the configurations tools are as follows:

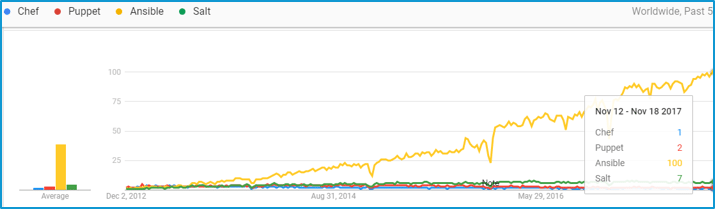
**Chef** – Chef Automate gives you everything you need to build, deploy in $137 node/ annual.

**Puppet** – Pricing for puppet ranges from $112 per node/year with a standard support plan to $199 per node/year with the premium plan.

**Ansible** – The pricing for Ansible Tower for standard IT operations up to 100 nodes is $10,000/year. This includes 8\*5 support whereas premium offers 24\*7 support for $14000/ year.

**Saltstack** – The cost for Saltstack Enterprise per 100 nodes is $15,00/ year(approx). You can contact the support for the current annual subscription price.

Now towards the end, I would like you to show the popularity of these tools i.e Chef vs Puppet vs Ansible vs Saltstack. Let’s have a look at the data trends in the image given below which shows how these tools have dominated the IT field for past 5 years.



As you can see above, puppet and chef are the old players whereas Ansible and saltstack are new players, and Ansible looks very promising with the growing trend. So, to conclude, all the four tools have their own advantages and categories in which they are better than the other. My only intention here is to help you in your decision making. So, it is necessary that you choose the appropriate tool which can be tailored according to your needs. If you want to know more about Chef, Puppet and Ansible, you can check out our blog on [Chef Tutorial](https://www.edureka.co/blog/chef-tutorial/), [Puppet Tutorial](https://www.edureka.co/blog/puppet-tutorial/) and [Ansible Tutorial](https://www.edureka.co/blog/ansible-tutorial/).

*If you found this blog on “****Chef vs Puppet vs Ansible vs Saltstack****” relevant, check out the*[***DevOps training***](https://www.edureka.co/devops)*by Edureka, a trusted online learning company with a network of more than 250,000 satisfied learners spread across the globe. The Edureka DevOps Certification Training course helps learners gain expertise in various DevOps processes and tools such as Puppet, Jenkins, Nagios and GIT for automating multiple steps in SDLC.*

Got a question for us? Please mention it in the comments section of this “Chef vs Puppet vs Ansible vs Saltstack” blog and we will get back to you as soon as possible.

**eneral DevOps Interview Questions**

This category will include questions that are not related to any particular DevOps stage. Questions here are meant to test your understanding about DevOps rather than focusing on a particular tool or a stage.

**Q1. What are the fundamental differences between DevOps & Agile?**

The differences between the two are listed down in the table below.

|  |  |  |
| --- | --- | --- |
| **DevOps vs Agile** | | |
| **Features** | **DevOps** | **Agile** |
| **Agility** | Agility in both Development & Operations | Agility in only Development |
| **Processes/ Practices** | Involves processes such as CI, CD, CT, etc. | Involves practices such as Agile Scrum, Agile Kanban, etc. |
| **Key Focus Area** | Timeliness & quality have equal priority | Timeliness is the main priority |
| **Release Cycles/ Development Sprints** | Smaller release cycles with immediate feedback | Smaller release cycles |
| **Source of Feedback** | Feedback is from self (Monitoring tools) | Feedback is from customers |
| **Scope of Work** | Agility & need for Automation | Agility only |

**Q2. What is the need for DevOps?**

According to me, this answer should start by explaining the general market trend. Instead of releasing big sets of features, companies are trying to see if small features can be transported to their customers through a series of release trains. This has many advantages like quick feedback from customers, better quality of software etc. which in turn leads to high customer satisfaction. To achieve this, companies are required to:

1. Increase deployment frequency
2. Lower failure rate of new releases
3. Shortened lead time between fixes
4. Faster mean time to recovery in the event of new release crashing

DevOps fulfills all these requirements and helps in achieving seamless software delivery. You can give examples of companies like Etsy, Google and Amazon which have adopted [DevOps to achieve levels of performance](https://www.edureka.co/blog/why-devops-improves-organization-performance/) that were unthinkable even five years ago. They are doing tens, hundreds or even thousands of code deployments per day while delivering world class stability, reliability and security.

If I have to test your knowledge on DevOps, you should know the difference between Agile and DevOps. The next question is directed towards that.

### ****Q3. How is DevOps different from Agile / SDLC?****

I would advise you to go with the below explanation:

Agile is a set of values and principles about how to produce i.e. develop software. Example: if you have some ideas and you want to turn those ideas into working software, you can use the Agile values and principles as a way to do that. But, that software might only be working on a developer’s laptop or in a test environment. You want a way to quickly, easily and repeatably move that software into production infrastructure, in a safe and simple way. To do that you need DevOps tools and techniques.

You can summarize by saying Agile software development methodology focuses on the development of software but DevOps on the other hand is responsible for development as well as deployment of the software in the safest and most reliable way possible. Here’s a blog that will give you more information on the [evolution of DevOps](https://www.edureka.co/blog/what-is-devops/).

Now remember, you have included DevOps tools in your previous answer so be prepared to answer some questions related to that.

### ****Q4. Which are the top DevOps tools? Which tools have you worked on?****

The most popular DevOps tools are mentioned below:

* Git : Version Control System tool
* Jenkins : Continuous Integration tool
* Selenium : Continuous Testing tool
* Puppet, Chef, Ansible : Configuration Management and Deployment tools
* Nagios : Continuous Monitoring tool
* Docker : Containerization tool

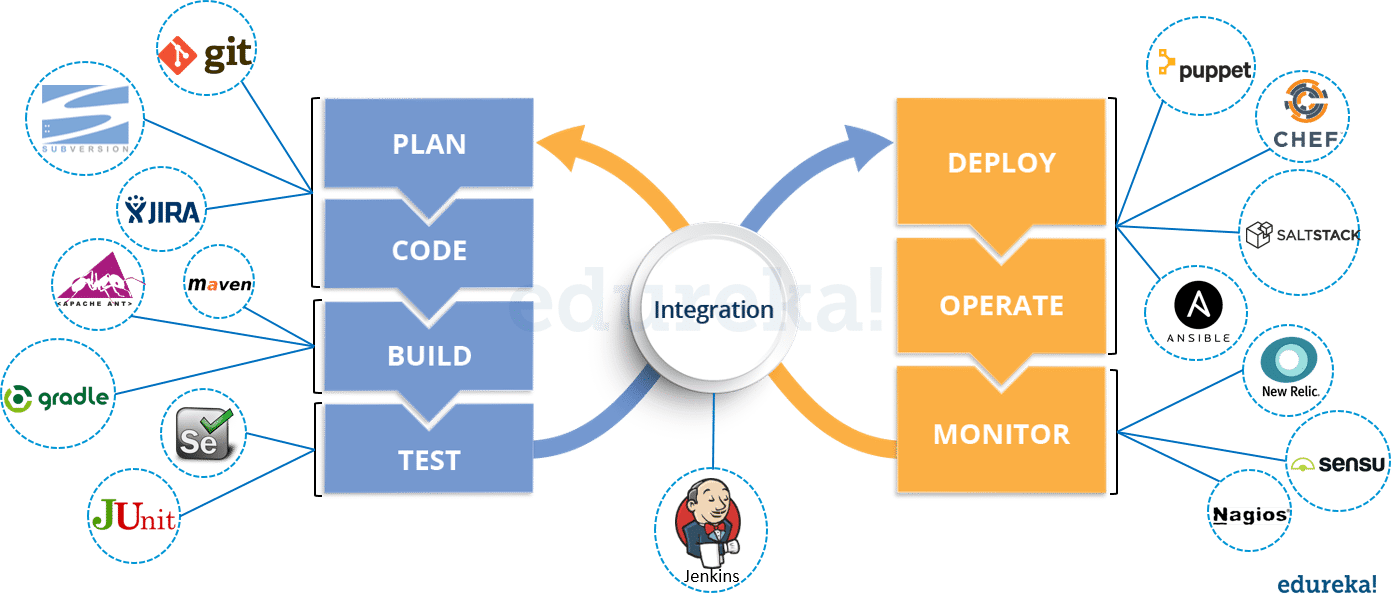
You can also mention any other tool if you want, but make sure you include the above tools in your answer.  
The second part of the answer has two possibilities:

1. If you have experience with all the above tools then you can say that I have worked on all these tools for developing good quality software and deploying those softwares easily, frequently, and reliably.
2. If you have experience only with some of the above tools then mention those tools and say that I have specialization in these tools and have an overview about the rest of the tools.

### ****Q5. How do all these tools work together?****

Given below is a generic logical flow where everything gets automated for seamless delivery. However, this flow may vary from organization to organization as per the requirement.

1. Developers develop the code and this source code is managed by Version Control System tools like Git etc.
2. Developers send this code to the Git repository and any changes made in the code is committed to this Repository.
3. Jenkins pulls this code from the repository using the Git plugin and build it using tools like Ant or Maven.
4. Configuration management tools like puppet deploys & provisions testing environment and then Jenkins releases this code on the test environment on which testing is done using tools like selenium.
5. Once the code is tested, Jenkins send it for deployment on the production server (even production server is provisioned & maintained by tools like puppet).
6. After deployment It is continuously monitored by tools like Nagios.
7. Docker containers provides testing environment to test the build features.



### ****Q6. What are the advantages of DevOps?****

For this answer, you can use your past experience and explain how DevOps helped you in your previous job. If you don’t have any such experience, then you can mention the below advantages.

Technical benefits:

* Continuous software delivery
* Less complex problems to fix
* Faster resolution of problems

Business benefits:

* Faster delivery of features
* More stable operating environments
* More time available to add value (rather than fix/maintain)

### ****Q7. What is the most important thing DevOps helps us achieve?****

According to me, the most important thing that DevOps helps us achieve is to get the changes into production as quickly as possible while minimizing risks in software quality assurance and compliance. This is the primary objective of DevOps. Learn more in this [DevOps tutorial](https://www.edureka.co/blog/devops-tutorial) blog.  
However, you can add many other positive effects of DevOps. For example, clearer communication and better working relationships between teams i.e. both the Ops team and Dev team collaborate together to deliver good quality software which in turn leads to higher customer satisfaction.

### ****Q8. Explain with a use case where DevOps can be used in industry/ real-life.****

There are many industries that are using DevOps so you can mention any of those use cases, you can also refer the below example:  
Etsy is a peer-to-peer e-commerce website focused on handmade or vintage items and supplies, as well as unique factory-manufactured items. Etsy struggled with slow, painful site updates that frequently caused the site to go down. It affected sales for millions of Etsy’s users who sold goods through online market place and risked driving them to the competitor.  
With the help of a new technical management team, Etsy transitioned from its waterfall model, which produced four-hour full-site deployments twice weekly, to a more agile approach. Today, it has a fully automated deployment pipeline, and its continuous delivery practices have reportedly resulted in more than 50 deployments a day with fewer disruptions.

### ****Q9. Explain your understanding and expertise on both the software development side and the technical operations side of an organization you have worked with in the past.****

For this answer, share your past experience and try to explain how flexible you were in your previous job. You can refer the below example:  
DevOps engineers almost always work in a 24/7 business-critical online environment. I was adaptable to on-call duties and was available to take up real-time, live-system responsibility. I successfully automated processes to support continuous software deployments. I have experience with public/private clouds, tools like Chef or Puppet, scripting and automation with tools like Python and PHP, and a background in Agile.

### ****Q10. What are the anti-patterns of DevOps?****

A pattern is common usage usually followed. If a pattern commonly adopted by others does not work for your organization and you continue to blindly follow it, you are essentially adopting an anti-pattern. There are myths about DevOps. Some of them include:

* DevOps is a process
* Agile equals DevOps?
* We need a separate DevOps group
* Devops will solve all our problems
* DevOps means Developers Managing Production
* DevOps is Development-driven release management
  1. DevOps is not development driven.
  2. DevOps is not IT Operations driven.
* We can’t do DevOps – We’re Unique
* We can’t do DevOps – We’ve got the wrong people

[**Learn DevOps Now >>**](https://www.edureka.co/devops/)

## **Version Control System (VCS) Interview Questions**

Now let’s look at interview questions on VCS:

### ****Q1. What is Version control?****

This is probably the easiest question you will face in the interview. My suggestion is to first give a definition of Version control. It is a system that records changes to a file or set of files over time so that you can recall specific versions later. Version control systems consist of a central shared repository where teammates can commit changes to a file or set of file. Then you can mention the uses of version control.

Version control allows you to:

* Revert files back to a previous state.
* Revert the entire project back to a previous state.
* Compare changes over time.
* See who last modified something that might be causing a problem.
* Who introduced an issue and when.

## **Q2. What are the benefits of using version control?**

I will suggest you to include the following advantages of version control:

1. With Version Control System (VCS), all the team members are allowed to work freely on any file at any time. VCS will later allow you to merge all the changes into a common version.
2. All the past versions and variants are neatly packed up inside the VCS. When you need it, you can request any version at any time and you’ll have a snapshot of the complete project right at hand.
3. Every time you save a new version of your project, your VCS requires you to provide a short description of what was changed. Additionally, you can see what exactly was changed in the file’s content. This allows you to know who has made what change in the project.
4. A distributed VCS like Git allows all the team members to have complete history of the project so if there is a breakdown in the central server you can use any of your teammate’s local Git repository.

### ****Q3. Describe branching strategies you have used.****

This question is asked to test your branching experience so tell them about how you have used branching in your previous job and what purpose does it serves, you can refer the below points:

* Feature branching  
  A feature branch model keeps all of the changes for a particular feature inside of a branch. When the feature is fully tested and validated by automated tests, the branch is then merged into master.
* Task branching  
  In this model each task is implemented on its own branch with the task key included in the branch name. It is easy to see which code implements which task, just look for the task key in the branch name.
* Release branching  
  Once the develop branch has acquired enough features for a release, you can clone that branch to form a Release branch. Creating this branch starts the next release cycle, so no new features can be added after this point, only bug fixes, documentation generation, and other release-oriented tasks should go in this branch. Once it is ready to ship, the release gets merged into master and tagged with a version number. In addition, it should be merged back into develop branch, which may have progressed since the release was initiated.

In the end tell them that branching strategies varies from one organization to another, so I know basic branching operations like delete, merge, checking out a branch etc.

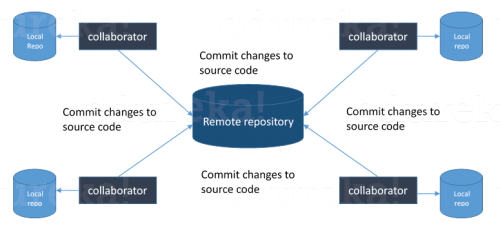
### ****Q4. Which VCS tool you are comfortable with?****

You can just mention the VCS tool that you have worked on like this: “I have worked on Git and one major advantage it has over other VCS tools like SVN is that it is a distributed version control system.”  
Distributed VCS tools do not necessarily rely on a central server to store all the versions of a project’s files. Instead, every developer “clones” a copy of a repository and has the full history of the project on their own hard drive.

### ****Q5. What is Git?****

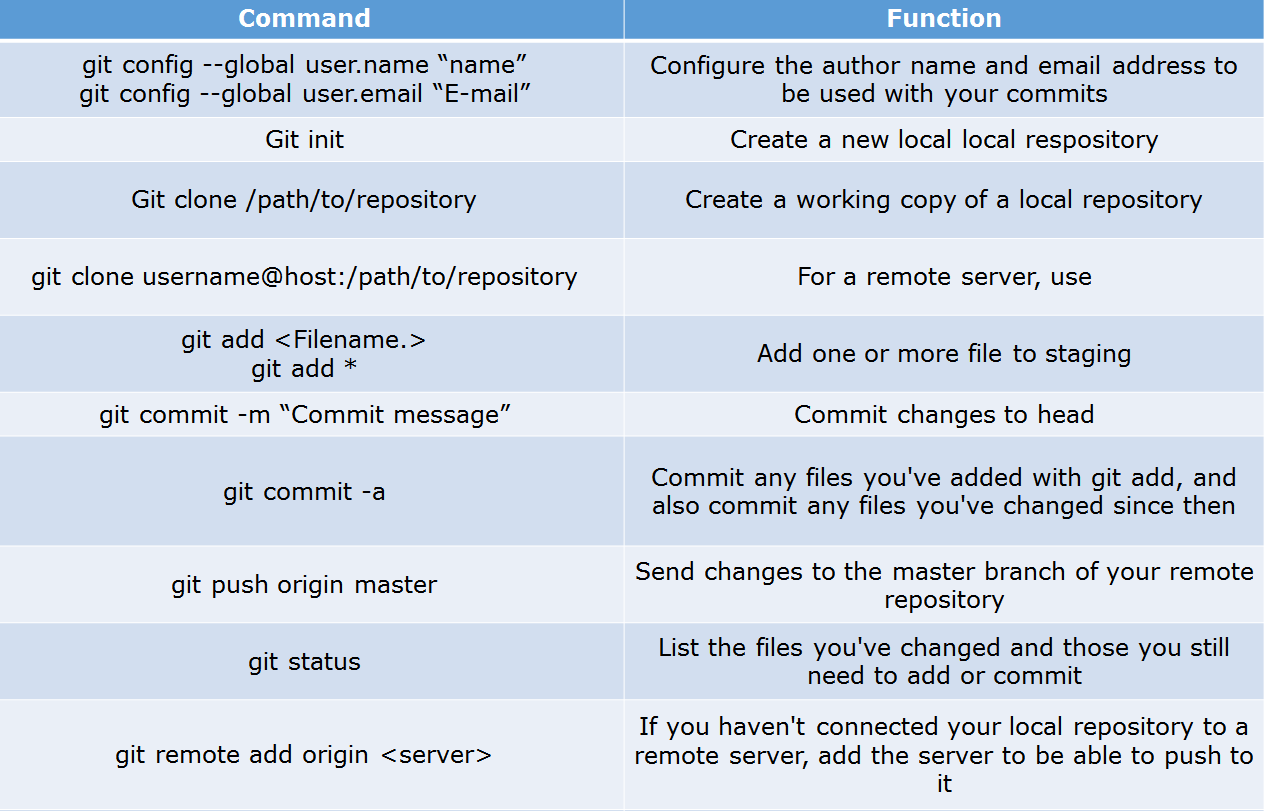
I will suggest that you attempt this question by first explaining about the architecture of git as shown in the below diagram. You can refer to the explanation given below:

* Git is a Distributed Version Control system (DVCS). It can track changes to a file and allows you to revert back to any particular change.
* Its distributed architecture provides many advantages over other Version Control Systems (VCS) like SVN one major advantage is that it does not rely on a central server to store all the versions of a project’s files. Instead, every developer “clones” a copy of a repository I have shown in the diagram below with “Local repository” and has the full history of the project on his hard drive so that when there is a server outage, all you need for recovery is one of your teammate’s local Git repository.
* There is a central cloud repository as well where developers can commit changes and share it with other teammates as you can see in the diagram where all collaborators are commiting changes “Remote repository”.



### ****Q6. Explain some basic Git commands?****

Below are some basic Git commands:



### ****Q7. In Git how do you revert a commit that has already been pushed and made public?****

There can be two answers to this question so make sure that you include both because any of the below options can be used depending on the situation:

* Remove or fix the bad file in a new commit and push it to the remote repository. This is the most natural way to fix an error. Once you have made necessary changes to the file, commit it to the remote repository for that I will use  
  **git commit -m “commit message”**
* Create a new commit that undoes all changes that were made in the bad commit.to do this I will use a command  
  **git revert <name of bad commit>**

### ****Q8. How do you squash last N commits into a single commit?****

There are two options to squash last N commits into a single commit. Include both of the below mentioned options in your answer:

* If you want to write the new commit message from scratch use the following command  
  **git reset –soft HEAD~N &&**  
  **git commit**
* If you want to start editing the new commit message with a concatenation of the existing commit messages then you need to extract those messages and pass them to Git commit for that I will use  
  **git reset –soft HEAD~N &&**  
  **git commit –edit -m”$(git log –format=%B –reverse .HEAD@{N})”**

### ****Q9. What is Git bisect? How can you use it to determine the source of a (regression) bug?****

I will suggest you to first give a small definition of Git bisect, Git bisect is used to find the commit that introduced a bug by using binary search. Command for Git bisect is  
**git bisect <subcommand> <options>**Now since you have mentioned the command above, explain what this command will do, This command uses a binary search algorithm to find which commit in your project’s history introduced a bug. You use it by first telling it a “bad” commit that is known to contain the bug, and a “good” commit that is known to be before the bug was introduced. Then Git bisect picks a commit between those two endpoints and asks you whether the selected commit is “good” or “bad”. It continues narrowing down the range until it finds the exact commit that introduced the change.

### ****Q10. What is Git rebase and how can it be used to resolve conflicts in a feature branch before merge?****

According to me, you should start by saying git rebase is a command which will merge another branch into the branch where you are currently working, and move all of the local commits that are ahead of the rebased branch to the top of the history on that branch.  
Now once you have defined Git rebase time for an example to show how it can be used to resolve conflicts in a feature branch before merge, if a feature branch was created from master, and since then the master branch has received new commits, Git rebase can be used to move the feature branch to the tip of master.  
The command effectively will replay the changes made in the feature branch at the tip of master, allowing conflicts to be resolved in the process. When done with care, this will allow the feature branch to be merged into master with relative ease and sometimes as a simple fast-forward operation.

### ****Q11. How do you configure a Git repository to run code sanity checking tools right before making commits, and preventing them if the test fails?****

I will suggest you to first give a small introduction to sanity checking, A sanity or smoke testdetermines whether it is possible and reasonable to continue testing.  
Now explain how to achieve this, this can be done with a simple script related to the pre-commit hook of the repository. The pre-commit hook is triggered right before a commit is made, even before you are required to enter a commit message. In this script one can run other tools, such as linters and perform sanity checks on the changes being committed into the repository.  
Finally give an example, you can refer the below script:  
**#!/bin/sh  
files=$(git diff –cached –name-only –diff-filter=ACM | grep ‘.go$’)  
if [ -z files ]; then  
exit 0  
fi  
unfmtd=$(gofmt -l $files)  
if [ -z unfmtd ]; then  
exit 0  
fi  
echo “Some .go files are not fmt’d”  
exit 1**This script checks to see if any .go file that is about to be committed needs to be passed through the standard Go source code formatting tool gofmt. By exiting with a non-zero status, the script effectively prevents the commit from being applied to the repository.

### ****Q12. How do you find a list of files that has changed in a particular commit?****

For this answer instead of just telling the command, explain what exactly this command will do so you can say that, To get a list files that has changed in a particular commit use command  
**git diff-tree -r {hash}**  
Given the commit hash, this will list all the files that were changed or added in that commit. The -r flag makes the command list individual files, rather than collapsing them into root directory names only.  
You can also include the below mention point although it is totally optional but will help in impressing the interviewer.  
The output will also include some extra information, which can be easily suppressed by including two flags:  
**git diff-tree –no-commit-id –name-only -r {hash}**  
Here –no-commit-id will suppress the commit hashes from appearing in the output, and –name-only will only print the file names, instead of their paths.

### ****Q13. How do you setup a script to run every time a repository receives new commits through push?****

There are three ways to configure a script to run every time a repository receives new commits through push, one needs to define either a pre-receive, update, or a post-receive hook depending on when exactly the script needs to be triggered.

* Pre-receive hook in the destination repository is invoked when commits are pushed to it. Any script bound to this hook will be executed before any references are updated. This is a useful hook to run scripts that help enforce development policies.
* Update hook works in a similar manner to pre-receive hook, and is also triggered before any updates are actually made. However, the update hook is called once for every commit that has been pushed to the destination repository.
* Finally, post-receive hook in the repository is invoked after the updates have been accepted into the destination repository. This is an ideal place to configure simple deployment scripts, invoke some continuous integration systems, dispatch notification emails to repository maintainers, etc.

Hooks are local to every Git repository and are not versioned. Scripts can either be created within the hooks directory inside the “.git” directory, or they can be created elsewhere and links to those scripts can be placed within the directory.

### ****Q14. How will you know in Git if a branch has already been merged into master?****

I will suggest you to include both the below mentioned commands:  
git branch –merged lists the branches that have been merged into the current branch.  
git branch –no-merged lists the branches that have not been merged.

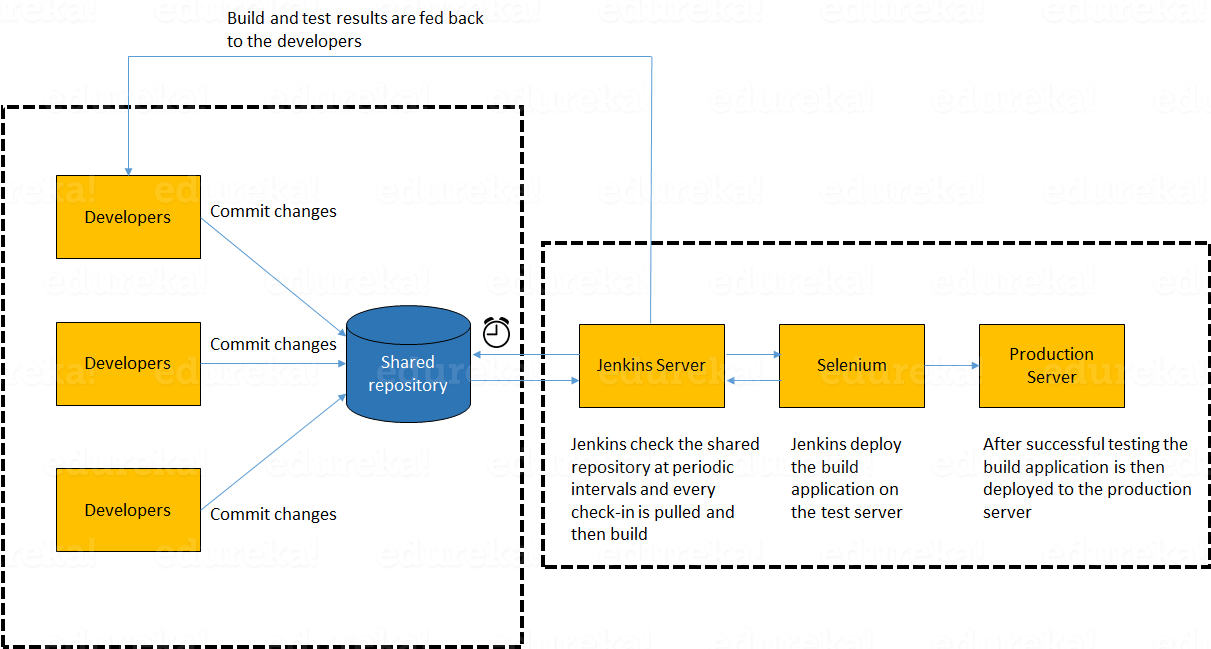
[**Learn GIT With DevOps >>**](https://www.edureka.co/devops/)

## **Continuous Integration questions**

Now, let’s look at Continuous Integration interview questions:

### ****Q1. What is meant by Continuous Integration?****

I will advise you to begin this answer by giving a small definition of Continuous Integration (CI). It is a development practice that requires developers to integrate code into a shared repository several times a day. Each check-in is then verified by an automated build, allowing teams to detect problems early.  
I suggest that you explain how you have implemented it in your previous job. You can refer the below given example:



In the diagram shown above:

1. Developers check out code into their private workspaces.
2. When they are done with it they commit the changes to the shared repository (Version Control Repository).
3. The CI server monitors the repository and checks out changes when they occur.
4. The CI server then pulls these changes and builds the system and also runs unit and integration tests.
5. The CI server will now inform the team of the successful build.
6. If the build or tests fails, the CI server will alert the team.
7. The team will try to fix the issue at the earliest opportunity.
8. This process keeps on repeating.

### ****Q2. Why do you need a Continuous Integration of Dev & Testing?****

For this answer, you should focus on the need of Continuous Integration. My suggestion would be to mention the below explanation in your answer:  
Continuous Integration of Dev and Testing improves the quality of software, and reduces the time taken to deliver it, by replacing the traditional practice of testing after completing all development. It allows Dev team to easily detect and locate problems early because developers need to integrate code into a shared repository several times a day (more frequently). Each check-in is then automatically tested.

### ****Q3. What are the success factors for Continuous Integration?****

Here you have to mention the requirements for Continuous Integration. You could include the following points in your answer:

* Maintain a code repository
* Automate the build
* Make the build self-testing
* Everyone commits to the baseline every day
* Every commit (to baseline) should be built
* Keep the build fast
* Test in a clone of the production environment
* Make it easy to get the latest deliverables
* Everyone can see the results of the latest build
* Automate deployment

### ****Q4. Explain how you can move or copy Jenkins from one server to another?****

I will approach this task by copying the jobs directory from the old server to the new one. There are multiple ways to do that;  I have mentioned them below:  
You can:

* Move a job from one installation of Jenkins to another by simply copying the corresponding job directory.
* Make a copy of an existing job by making a clone of a job directory by a different name.
* Rename an existing job by renaming a directory. Note that if you change a job name you will need to change any other job that tries to call the renamed job.

### ****Q5. Explain how can create a backup and copy files in Jenkins?****

Answer to this question is really direct. To create a backup, all you need to do is to periodically back up your JENKINS\_HOME directory. This contains all of your build jobs configurations, your slave node configurations, and your build history. To create a back-up of your Jenkins setup, just copy this directory. You can also copy a job directory to clone or replicate a job or rename the directory.

### ****Q6. Explain how you can setup Jenkins job?****

My approach to this answer will be to first mention how to create Jenkins job. Go to Jenkins top page, select “New Job”, then choose “Build a free-style software project”.  
Then you can tell the elements of this freestyle job:

* Optional SCM, such as CVS or Subversion where your source code resides.
* Optional triggers to control when Jenkins will perform builds.
* Some sort of build script that performs the build (ant, maven, shell script, batch file, etc.) where the real work happens.
* Optional steps to collect information out of the build, such as archiving the artifacts and/or recording javadoc and test results.
* Optional steps to notify other people/systems with the build result, such as sending e-mails, IMs, updating issue tracker, etc..

### ****Q7. Mention some of the useful plugins in Jenkins.****

Below, I have mentioned some important Plugins:

* Maven 2 project
* Amazon EC2
* HTML publisher
* Copy artifact
* Join
* Green Balls

These Plugins, I feel are the most useful plugins. If you want to include any other Plugin that is not mentioned above, you can add them as well. But, make sure you first mention the above stated plugins and then add your own.

### ****Q8. How will you secure Jenkins?****

The way I secure Jenkins is mentioned below. If you have any other way of doing it, please mention it in the comments section below:

* Ensure global security is on.
* Ensure that Jenkins is integrated with my company’s user directory with appropriate plugin.
* Ensure that matrix/Project matrix is enabled to fine tune access.
* Automate the process of setting rights/privileges in Jenkins with custom version controlled script.
* Limit physical access to Jenkins data/folders.
* Periodically run security audits on same.

[**Master Jenkins In DevOps**](https://www.edureka.co/devops/)

## **ontinuous Testing Interview Questions:**

Now let’s move on to the Continuous Testing questions.

### ****Q1. What is Continuous Testing?****

I will advise you to follow the below mentioned explanation:  
Continuous Testing is the process of executing automated tests as part of the software delivery pipeline to obtain immediate feedback on the business risks associated with in the latest build. In this way, each build is tested continuously, allowing Development teams to get fast feedback so that they can prevent those problems from progressing to the next stage of Software delivery life-cycle. This dramatically speeds up a developer’s workflow as there’s no need to manually rebuild the project and re-run all tests after making changes.

### ****Q2. What is Automation Testing?****

Automation testing or Test Automation is a process of automating the manual process to test the application/system under test. Automation testing involves use of separate testing tools which lets you create test scripts which can be executed repeatedly and doesn’t require any manual intervention.

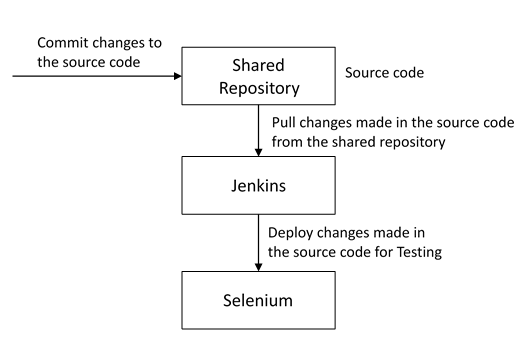
### ****Q3.**** ****What are the benefits of Automation Testing?****

I have listed down some advantages of automation testing. Include these in your answer and you can add your own experience of how Continuous Testing helped your previous company:

* Supports execution of repeated test cases
* Aids in testing a large test matrix
* Enables parallel execution
* Encourages unattended execution
* Improves accuracy thereby reducing human generated errors
* Saves time and money

### ****Q4. How to automate Testing in DevOps lifecycle?****

I have mentioned a generic flow below which you can refer to:  
In DevOps, developers are required to commit all the changes made in the source code to a shared repository. Continuous Integration tools like Jenkins will pull the code from this shared repository every time a change is made in the code and deploy it for Continuous Testing that is done by tools like Selenium as shown in the below diagram.  
In this way, any change in the code is continuously tested unlike the traditional approach.



### ****Q5. Why is Continuous Testing important for DevOps?****

You can answer this question by saying, “Continuous Testing allows any change made in the code to be tested immediately. This avoids the problems created by having “big-bang” testing left to the end of the cycle such as release delays and quality issues. In this way, Continuous Testing facilitates more frequent and good quality releases.”

### ****Q6. What are the key elements of Continuous Testing tools?****

Key elements of Continuous Testing are:

* **Risk Assessment:** It Covers risk mitigation tasks, technical debt, quality assessment and test coverage optimization to ensure the build is ready to progress toward next stage.
* **Policy Analysis:** It ensures all processes align with the organization’s evolving business and compliance demands are met.
* **Requirements Traceability:** It ensures true requirements are met and rework is not required. An object assessment is used to identify which requirements are at risk, working as expected or require further validation.
* **Advanced Analysis:** It uses automation in areas such as static code analysis, change impact analysis and scope assessment/prioritization to prevent defects in the first place and accomplishing more within each iteration.
* **Test Optimization:** It ensures tests yield accurate outcomes and provide actionable findings. Aspects include Test Data Management, Test Optimization Management and Test Maintenance
* **Service Virtualization:** It ensures access to real-world testing environments. Service visualization enables access to the virtual form of the required testing stages, cutting the waste time to test environment setup and availability.

### ****Q7. Which Testing tool are you comfortable with and what are the benefits of that tool?****

Here mention the testing tool that you have worked with and accordingly frame your answer. I have mentioned an example below:  
I have worked on Selenium to ensure high quality and more frequent releases.

Some advantages of Selenium are:

* It is free and open source
* It has a large user base and helping communities
* It has cross Browser compatibility (Firefox, chrome, Internet Explorer, Safari etc.)
* It has great platform compatibility (Windows, Mac OS, Linux etc.)
* It supports multiple programming languages (Java, C#, Ruby, Python, Pearl etc.)
* It has fresh and regular repository developments
* It supports distributed testing

### ****Q8. What are the Testing types supported by Selenium?****

Selenium supports two types of testing:  
**Regression Testing**: It is the act of retesting a product around an area where a bug was fixed.  
**Functional Testing**: It refers to the testing of software features (functional points) individually.

### ****Q9. What is Selenium IDE?****

My suggestion is to start this answer by defining Selenium IDE. It is an integrated development environment for Selenium scripts. It is implemented as a Firefox extension, and allows you to record, edit, and debug tests. Selenium IDE includes the entire Selenium Core, allowing you to easily and quickly record and play back tests in the actual environment that they will run in.  
Now include some advantages in your answer. With autocomplete support and the ability to move commands around quickly, Selenium IDE is the ideal environment for creating Selenium tests no matter what style of tests you prefer.

### ****Q10. What is the difference between Assert and Verify commands in Selenium?****

I have mentioned differences between Assert and Verify commands below:

* Assert command checks whether the given condition is true or false. Let’s say we assert whether the given element is present on the web page or not. If the condition is true, then the program control will execute the next test step. But, if the condition is false, the execution would stop and no further test would be executed.
* Verify command also checks whether the given condition is true or false. Irrespective of the condition being true or false, the program execution doesn’t halts i.e. any failure during verification would not stop the execution and all the test steps would be executed.

### ****Q11. How to launch Browser using WebDriver?****

The following syntax can be used to launch Browser:  
**WebDriver driver = new FirefoxDriver();**  
**WebDriver driver = new ChromeDriver();**  
**WebDriver driver = new InternetExplorerDriver();**

### ****Q12. When should I use Selenium Grid?****

For this answer, my suggestion would be to give a small definition of Selenium Grid. It can be used to execute same or different test scripts on multiple platforms and browsers concurrently to achieve distributed test execution. This allows testing under different environments and saving execution time remarkably.

## **Configuration Management Interview Questions**

Now let’s check how much you know about Configuration Management.

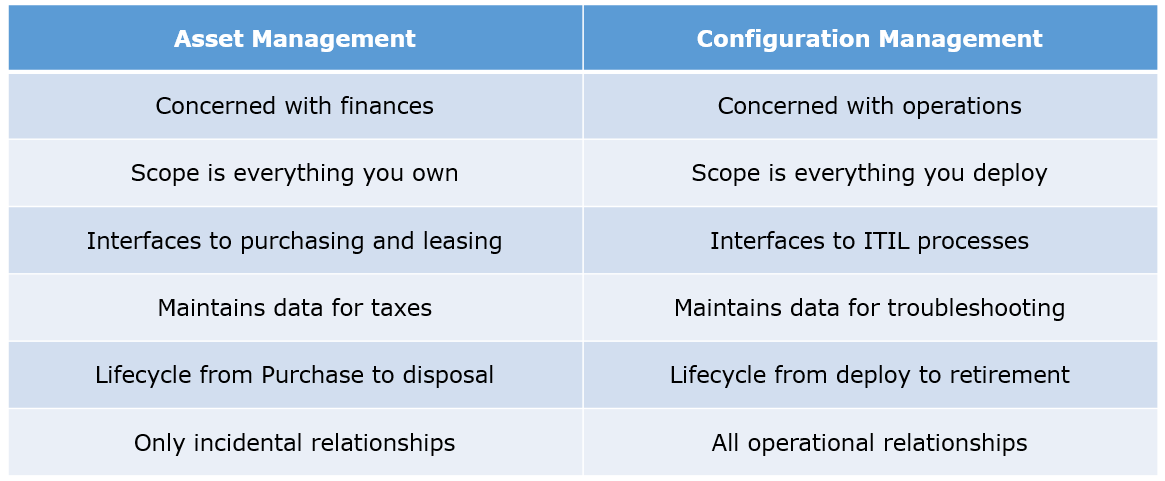
### ****Q1. What are the goals of Configuration management processes?****

The purpose of Configuration Management (CM) is to ensure the integrity of a product or system throughout its life-cycle by making the development or deployment process controllable and repeatable, therefore creating a higher quality product or system. The CM process allows orderly management of system information and system changes for purposes such as to:

* Revise capability,
* Improve performance,
* Reliability or maintainability,
* Extend life,
* Reduce cost,
* Reduce risk and
* Liability, or correct defects.

### ****Q2. What is the difference between Asset management and Configuration Management?****

Given below are few differences between Asset Management and Configuration Management:



### ****Q3. What is the difference between an Asset and a Configuration Item?****

According to me, you should first explain Asset. It has a financial value along with a depreciation rate attached to it. IT assets are just a sub-set of it. Anything and everything that has a cost and the organization uses it for its asset value calculation and related benefits in tax calculation falls under Asset Management, and such item is called an asset.  
Configuration Item on the other hand may or may not have financial values assigned to it. It will not have any depreciation linked to it. Thus, its life would not be dependent on its financial value but will depend on the time till that item becomes obsolete for the organization.

Now you can give an example that can showcase the similarity and differences between both:  
1) Similarity:  
Server – It is both an asset as well as a CI.  
2) Difference:  
Building – It is an asset but not a CI.  
Document – It is a CI but not an asset

### ****Q4. What do you understand by “Infrastructure as code”? How does it fit into the DevOps methodology? What purpose does it achieve?****

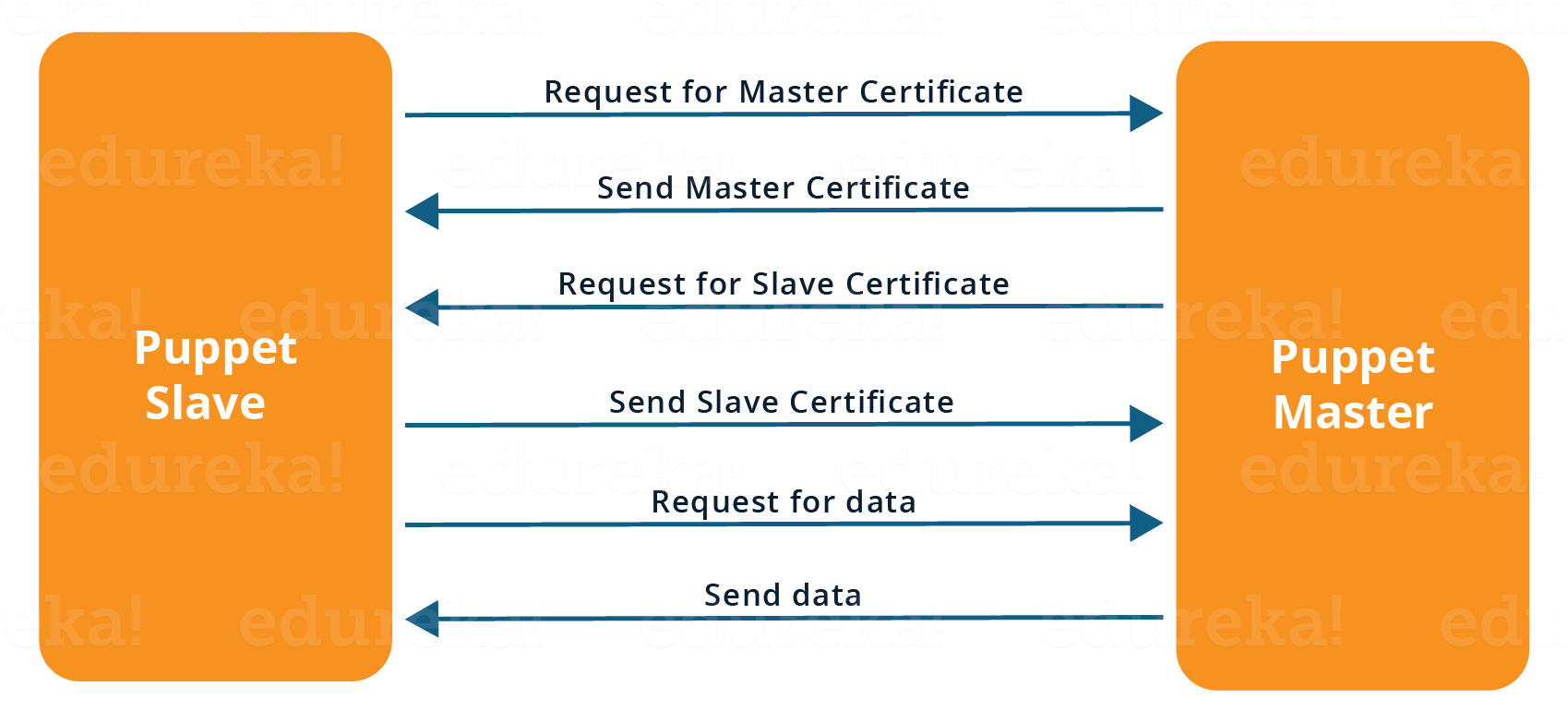
Infrastructure as Code (IAC) is a type of IT infrastructure that operations teams can use to automatically manage and provision through code, rather than using a manual process.  
Companies for faster deployments treat infrastructure like software: as code that can be managed with the DevOps tools and processes. These tools let you make infrastructure changes more easily, rapidly, safely and reliably.

### ****Q5. Which among Puppet, Chef, SaltStack and Ansible is the best Configuration Management (CM) tool? Why?****

This depends on the organization’s need so mention few points on all those tools:  
Puppet is the oldest and most mature CM tool. Puppet is a Ruby-based Configuration Management tool, but while it has some free features, much of what makes Puppet great is only available in the paid version. Organizations that don’t need a lot of extras will find Puppet useful, but those needing more customization will probably need to upgrade to the paid version.  
Chef is written in Ruby, so it can be customized by those who know the language. It also includes free features, plus it can be upgraded from open source to enterprise-level if necessary. On top of that, it’s a very flexible product.  
Ansible is a very secure option since it uses Secure Shell. It’s a simple tool to use, but it does offer a number of other services in addition to configuration management. It’s very easy to learn, so it’s perfect for those who don’t have a dedicated IT staff but still need a configuration management tool.  
SaltStack is python based open source CM tool made for larger businesses, but its learning curve is fairly low.

### ****Q6. What is Puppet?****

I will advise you to first give a small definition of Puppet. It is a Configuration Management tool which is used to automate administration tasks.  
Now you should describe its architecture and how Puppet manages its Agents. Puppet has a Master-Slave architecture in which the Slave has to first send a Certificate signing request to Master and Master has to sign that Certificate in order to establish a secure connection between Puppet Master and Puppet Slave as shown on the diagram below. Puppet Slave sends request to Puppet Master and Puppet Master then pushes configuration on Slave.  
Refer the diagram below that explains the above description.



### ****Q7. Before a client can authenticate with the Puppet Master, its certs need to be signed and accepted. How will you automate this task?****

The easiest way is to enable auto-signing in puppet.conf.  
Do mention that this is a security risk. If you still want to do this:

* Firewall your puppet master – restrict port tcp/8140 to only networks that you trust.
* Create puppet masters for each ‘trust zone’, and only include the trusted nodes in that Puppet masters manifest.
* Never use a full wildcard such as \*.

### ****Q8. Describe the most significant gain you made from automating a process through Puppet.****

For this answer, I will suggest you to explain you past experience with Puppet. you can refer the below example:  
I automated the configuration and deployment of Linux and Windows machines using Puppet. In addition to shortening the processing time from one week to 10 minutes, I used the roles and profiles pattern and documented the purpose of each module in README to ensure that others could update the module using Git. The modules I wrote are still being used, but they’ve been improved by my teammates and members of the community

### ****Q9. Which open source or community tools do you use to make Puppet more powerful?****

Over here, you need to mention the tools and how you have used those tools to make Puppet more powerful. Below is one example for your reference:  
Changes and requests are ticketed through Jira and we manage requests through an internal process. Then, we use Git and Puppet’s Code Manager app to manage Puppet code in accordance with best practices. Additionally, we run all of our Puppet changes through our continuous integration pipeline in Jenkins using the beaker testing framework.

### ****Q10. What are Puppet Manifests?****

It is a very important question so make sure you go in a correct flow. According to me, you should first define Manifests. Every node (or Puppet Agent) has got its configuration details in Puppet Master, written in the native Puppet language. These details are written in the language which Puppet can understand and are termed as Manifests. They are composed of Puppet code and their filenames use the .pp extension.  
Now give an exampl. You can write a manifest in Puppet Master that creates a file and installs apache on all Puppet Agents (Slaves) connected to the Puppet Master.

### ****Q11.**** ****What is Puppet Module and How it is different from Puppet Manifest?****

For this answer, you can go with the below mentioned explanation:  
A Puppet Module is a collection of Manifests and data (such as facts, files, and templates), and they have a specific directory structure. Modules are useful for organizing your Puppet code, because they allow you to split your code into multiple Manifests. It is considered best practice to use Modules to organize almost all of your Puppet Manifests.  
Puppet programs are called Manifests which are composed of Puppet code and their file names use the .pp extension.

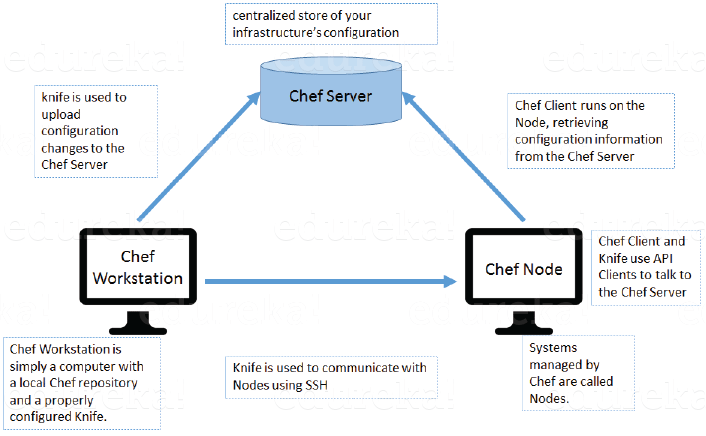
### ****Q12.**** ****What is Facter in Puppet?****

You are expected to answer what exactly Facter does in Puppet so according to me, you should say, “Facter gathers basic information (facts) about Puppet Agent such as hardware details, network settings, OS type and version, IP addresses, MAC addresses, SSH keys, and more. These facts are then made available in Puppet Master’s Manifests as variables.”

### ****Q13. What is Chef?****

Begin this answer by defining Chef. It is a powerful automation platform that transforms infrastructure into code. Chef is a tool for which you write scripts that are used to automate processes. What processes? Pretty much anything related to IT.  
Now you can explain the architecture of Chef, it consists of:

* **Chef Server:**The Chef Server is the central store of your infrastructure’s configuration data. The Chef Server stores the data necessary to configure your nodes and provides search, a powerful tool that allows you to dynamically drive node configuration based on data.
* **Chef Node:** A Node is any host that is configured using Chef-client. Chef-client runs on your nodes, contacting the Chef Server for the information necessary to configure the node. Since a Node is a machine that runs the Chef-client software, nodes are sometimes referred to as “clients”.
* **Chef Workstation:** A Chef Workstation is the host you use to modify your cookbooks and other configuration data.



### ****Q14. What is a resource in Chef?****

My suggestion is to first define Resource. A Resource represents a piece of infrastructure and its desired state, such as a package that should be installed, a service that should be running, or a file that should be generated.  
You should explain about the functions of Resource for that include the following points:

* Describes the desired state for a configuration item.
* Declares the steps needed to bring that item to the desired state.
* Specifies a resource type such as package, template, or service.
* Lists additional details (also known as resource properties), as necessary.
* Are grouped into recipes, which describe working configurations.

### ****Q15. What do you mean by recipe in Chef?****

For this answer, I will suggest you to use the above mentioned flow: first define Recipe. A Recipe is a collection of Resources that describes a particular configuration or policy. A Recipe describes everything that is required to configure part of a system.  
After the definition, explain the functions of Recipes by including the following points:

* Install and configure software components.
* Manage files.
* Deploy applications.
* Execute other recipes.

### ****Q16.**** ****How does a Cookbook differ from a Recipe in Chef?****

The answer to this is pretty direct. You can simply say, “a Recipe is a collection of Resources, and primarily configures a software package or some piece of infrastructure. A Cookbook groups together Recipes and other information in a way that is more manageable than having just Recipes alone.”

### ****Q17.**** ****What happens when you don’t specify a Resource’s action in Chef?****

My suggestion is to first give a direct answer: when you don’t specify a resource’s action, Chef applies the default action.  
Now explain this with an example, the below resource:  
**file ‘C:\Users\Administrator\chef-repo\settings.ini’ do**  
**content ‘greeting=hello world’**  
**end**  
is same as the below resource:  
**file ‘C:\Users\Administrator\chef-repo\settings.ini’ do**  
**action :create**  
**content ‘greeting=hello world’**  
**end**  
because: create is the file Resource’s default action.

### ****Q18. What is Ansible module?****

Modules are considered to be the units of work in Ansible. Each module is mostly standalone and can be written in a standard scripting language such as Python, Perl, Ruby, bash, etc.. One of the guiding properties of modules is idempotency, which means that even if an operation is repeated multiple times e.g. upon recovery from an outage, it will always place the system into the same state.

### ****Q19. What are playbooks in Ansible?****

Playbooks are Ansible’s configuration, deployment, and orchestration language. They can describe a policy you want your remote systems to enforce, or a set of steps in a general IT process. Playbooks are designed to be human-readable and are developed in a basic text language.  
At a basic level, playbooks can be used to manage configurations of and deployments to remote machines.

### ****Q20.**** ****How do I see a list of all of the ansible\_ variables?****

Ansible by default gathers “facts” about the machines under management, and these facts can be accessed in Playbooks and in templates. To see a list of all of the facts that are available about a machine, you can run the “setup” module as an ad-hoc action:  
**Ansible -m setup hostname**This will print out a dictionary of all of the facts that are available for that particular host.

### ****Q21. How can I set deployment order for applications?****

WebLogic Server 8.1 allows you to select the load order for applications. See the Application MBean Load Order attribute in Application. WebLogic Server deploys server-level resources (first JDBC and then JMS) before deploying applications. Applications are deployed in this order: connectors, then EJBs, then Web Applications. If the application is an EAR, the individual components are loaded in the order in which they are declared in the application.xml deployment descriptor.

### ****Q22. Can I refresh static components of a deployed application without having to redeploy the entire application?****

Yes, you can use weblogic.Deployer to specify a component and target a server, using the following syntax:  
java weblogic.Deployer -adminurl http://admin:7001 -name appname -targets server1,server2 -deploy jsps/\*.jsp

### ****Q23. How do I turn the auto-deployment feature off?****

The auto-deployment feature checks the applications folder every three seconds to determine whether there are any new applications or any changes to existing applications and then dynamically deploys these changes.

The auto-deployment feature is enabled for servers that run in development mode. To disable auto-deployment feature, use one of the following methods to place servers in production mode:

* In the Administration Console, click the name of the domain in the left pane, then select the Production Mode checkbox in the right pane.
* At the command line, include the following argument when starting the domain’s Administration Server:  
  -Dweblogic.ProductionModeEnabled=true
* Production mode is set for all WebLogic Server instances in a given domain.

### ****Q24. When should I use the external\_stage option?****

Set -external\_stage using weblogic.Deployer if you want to stage the application yourself, and prefer to copy it to its target by your own means.

## **ontinuous Monitoring Interview Questions**

Let’s test your knowledge on Continuous Monitoring.

### ****Q1. Why is Continuous monitoring necessary?****

I will suggest you to go with the below mentioned flow:  
Continuous Monitoring allows timely identification of problems or weaknesses and quick corrective action that helps reduce expenses of an organization. Continuous monitoring provides solution that addresses three operational disciplines known as:

* continuous audit
* continuous controls monitoring
* continuous transaction inspection

### ****Q2. What is Nagios?****

You can answer this question by first mentioning that Nagios is one of the monitoring tools. It is used for Continuous monitoring of systems, applications, services, and business processes etc in a DevOps culture. In the event of a failure, Nagios can alert technical staff of the problem, allowing them to begin remediation processes before outages affect business processes, end-users, or customers. With Nagios, you don’t have to explain why an unseen infrastructure outage affect your organization’s bottom line.  
Now once you have defined what is Nagios, you can mention the various things that you can achieve using Nagios.  
By using Nagios you can:

* Plan for infrastructure upgrades before outdated systems cause failures.
* Respond to issues at the first sign of a problem.
* Automatically fix problems when they are detected.
* Coordinate technical team responses.
* Ensure your organization’s SLAs are being met.
* Ensure IT infrastructure outages have a minimal effect on your organization’s bottom line.
* Monitor your entire infrastructure and business processes.

This completes the answer to this question. Further details like advantages etc. can be added as per the direction where the discussion is headed.

### ****Q3. How does Nagios works?****

I will advise you to follow the below explanation for this answer:  
Nagios runs on a server, usually as a daemon or service. Nagios periodically runs plugins residing on the same server, they contact hosts or servers on your network or on the internet. One can view the status information using the web interface. You can also receive email or SMS notifications if something happens.  
The Nagios daemon behaves like a scheduler that runs certain scripts at certain moments. It stores the results of those scripts and will run other scripts if these results change.

Now expect a few questions on Nagios components like Plugins, NRPE etc..

### ****Q4. What are Plugins in Nagios?****

Begin this answer by defining Plugins. They are scripts (Perl scripts, Shell scripts, etc.) that can run from a command line to check the status of a host or service. Nagios uses the results from Plugins to determine the current status of hosts and services on your network.   
Once you have defined Plugins, explain why we need Plugins. Nagios will execute a Plugin whenever there is a need to check the status of a host or service. Plugin will perform the check and then simply returns the result to Nagios. Nagios will process the results that it receives from the Plugin and take the necessary actions.

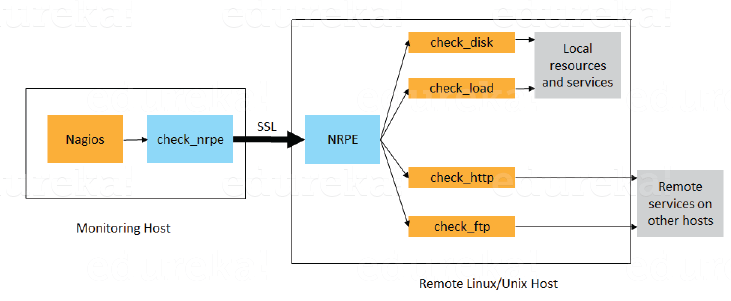
### ****Q5. What is NRPE (Nagios Remote Plugin Executor) in Nagios?****

For this answer, give a brief definition of Plugins. The NRPE addon is designed to allow you to execute Nagios plugins on remote Linux/Unix machines. The main reason for doing this is to allow Nagios to monitor “local” resources (like CPU load, memory usage, etc.) on remote machines. Since these public resources are not usually exposed to external machines, an agent like NRPE must be installed on the remote Linux/Unix machines.

I will advise you to explain the NRPE architecture on the basis of diagram shown below. The NRPE addon consists of two pieces:

* The check\_nrpe plugin, which resides on the local monitoring machine.
* The NRPE daemon, which runs on the remote Linux/Unix machine.

There is a SSL (Secure Socket Layer) connection between monitoring host and remote host as shown in the diagram below.



### ****Q6. What do you mean by passive check in Nagios?****

According to me, the answer should start by explaining Passive checks. They are initiated and performed by external applications/processes and the Passive check results are submitted to Nagios for processing.  
Then explain the need for passive checks. They are useful for monitoring services that are Asynchronous in nature and cannot be monitored effectively by polling their status on a regularly scheduled basis. They can also be used for monitoring services that are Located behind a firewall and cannot be checked actively from the monitoring host.

### ****Q7.**** ****When Does Nagios Check for external commands?****

Make sure that you stick to the question during your explanation so I will advise you to follow the below mentioned flow. Nagios check for external commands under the following conditions:

* At regular intervals specified by the command\_check\_interval option in the main configuration file or,
* Immediately after event handlers are executed. This is in addition to the regular cycle of external command checks and is done to provide immediate action if an event handler submits commands to Nagios.

### ****Q8. What is the difference between Active and Passive check in Nagios?****

For this answer, first point out the basic difference Active and Passive checks. The major difference between Active and Passive checks is that Active checks are initiated and performed by Nagios, while passive checks are performed by external applications.  
If your interviewer is looking unconvinced with the above explanation then you can also mention some key features of both Active and Passive checks:  
Passive checks are useful for monitoring services that are:

* Asynchronous in nature and cannot be monitored effectively by polling their status on a regularly scheduled basis.
* Located behind a firewall and cannot be checked actively from the monitoring host.

The main features of Actives checks are as follows:

* Active checks are initiated by the Nagios process.
* Active checks are run on a regularly scheduled basis.

### ****Q9. How does Nagios help with Distributed Monitoring?****

The interviewer will be expecting an answer related to the distributed architecture of Nagios. So, I suggest that you answer it in the below mentioned format:  
With Nagios you can monitor your whole enterprise by using a distributed monitoring scheme in which local slave instances of Nagios perform monitoring tasks and report the results back to a single master. You manage all configuration, notification, and reporting from the master, while the slaves do all the work. This design takes advantage of Nagios’s ability to utilize passive checks i.e. external applications or processes that send results back to Nagios. In a distributed configuration, these external applications are other instances of Nagios.

### ****Q10. Explain Main Configuration file of Nagios and its location?****

First mention what this main configuration file contains and its function. The main configuration file contains a number of directives that affect how the Nagios daemon operates. This config file is read by both the Nagios daemon and the CGIs (It specifies the location of your main configuration file).  
Now you can tell where it is present and how it is created. A sample main configuration file is created in the base directory of the Nagios distribution when you run the configure script. The default name of the main configuration file is nagios.cfg. It is usually placed in the etc/ subdirectory of you Nagios installation (i.e. /usr/local/nagios/etc/).

### ****Q11. Explain how Flap Detection works in Nagios?****

I will advise you to first explain Flapping first. Flapping occurs when a service or host changes state too frequently, this causes lot of problem and recovery notifications.  
Once you have defined Flapping, explain how Nagios detects Flapping. Whenever Nagios checks the status of a host or service, it will check to see if it has started or stopped flapping. Nagios follows the below given procedure to do that:

* Storing the results of the last 21 checks of the host or service analyzing the historical check results and determine where state changes/transitions occur
* Using the state transitions to determine a percent state change value (a measure of change) for the host or service
* Comparing the percent state change value against low and high flapping thresholds

A host or service is determined to have started flapping when its percent state change first exceeds a high flapping threshold. A host or service is determined to have stopped flapping when its percent state goes below a low flapping threshold.

### ****Q12. What are the three main variables that affect recursion and inheritance in Nagios?****

According to me the proper format for this answer should be:  
First name the variables and then a small explanation of each of these variables:

* Name
* Use
* Register

Then give a brief explanation for each of these variables. Name is a placeholder that is used by other objects. Use defines the “parent” object whose properties should be used. Register can have a value of 0 (indicating its only a template) and 1 (an actual object). The register value is never inherited.

### ****Q13. What is meant by saying Nagios is Object Oriented?****

Answer to this question is pretty direct. I will answer this by saying, “One of the features of Nagios is object configuration format in that you can create object definitions that inherit properties from other object definitions and hence the name. This simplifies and clarifies relationships between various components.”

### ****Q14. What is State Stalking in Nagios?****

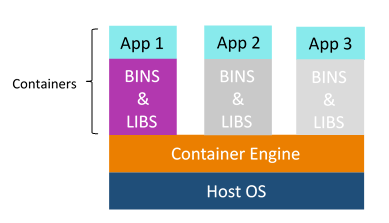
I will advise you to first give a small introduction on State Stalking. It is used for logging purposes. When Stalking is enabled for a particular host or service, Nagios will watch that host or service very carefully and log any changes it sees in the output of check results.  
Depending on the discussion between you and interviewer you can also add, “It can be very helpful in later analysis of the log files. Under normal circumstances, the result of a host or service check is only logged if the host or service has changed state since it was last checked.”

## **Containerization and Virtualization Interview Questions**

Let’s see how much you know about containers and VMs.

### ****Q1. What are containers?****

My suggestion is to explain the need for containerization first, containers are used to provide consistent computing environment from a developer’s laptop to a test environment, from a staging environment into production.  
Now give a definition of containers, a container consists of an entire runtime environment: an application, plus all its dependencies, libraries and other binaries, and configuration files needed to run it, bundled into one package. Containerizing the application platform and its dependencies removes the differences in OS distributions and underlying infrastructure.



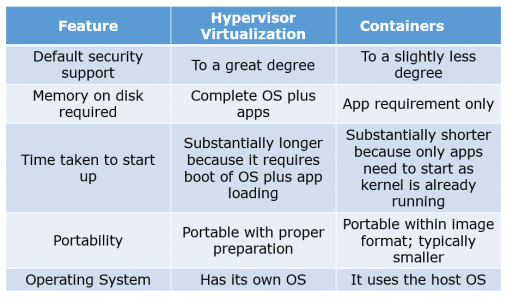
### ****Q2. What are the advantages that Containerization provides over virtualization?****

Below are the advantages of containerization over virtualization:

* Containers provide real-time provisioning and scalability but VMs provide slow provisioning
* Containers are lightweight when compared to VMs
* VMs have limited performance when compared to containers
* Containers have better resource utilization compared to VMs

### ****Q3. How exactly are containers (Docker in our case) different from hypervisor virtualization (vSphere)? What are the benefits?****

Given below are some differences. Make sure you include these differences in your answer:



### ****Q4. What is Docker image?****

I suggest that you go with the below mentioned flow:  
Docker image is the source of Docker container. In other words, Docker images are used to create containers. Images are created with the build command, and they’ll produce a container when started with run. Images are stored in a Docker registry such as registry.hub.docker.com because they can become quite large, images are designed to be composed of layers of other images, allowing a minimal amount of data to be sent when transferring images over the network.  
**Tip: Be aware of Dockerhub in order to answer questions on pre-available images.**

### ****Q5. What is Docker container?****

This is a very important question so just make sure you don’t deviate from the topic. I advise you to follow the below mentioned format:  
Docker containers include the application and all of its dependencies but share the kernel with other containers, running as isolated processes in user space on the host operating system. Docker containers are not tied to any specific infrastructure: they run on any computer, on any infrastructure, and in any cloud.  
Now explain how to create a Docker container, Docker containers can be created by either creating a Docker image and then running it or you can use Docker images that are present on the Dockerhub.  
Docker containers are basically runtime instances of Docker images.

### ****Q6. What is Docker hub?****

Answer to this question is pretty direct. Docker hub is a cloud-based registry service which allows you to link to code repositories, build your images and test them, stores manually pushed images, and links to Docker cloud so you can deploy images to your hosts. It provides a centralized resource for container image discovery, distribution and change management, user and team collaboration, and workflow automation throughout the development pipeline.

### ****Q7.**** ****How is Docker different from other container technologies?****

According to me, below points should be there in your answer:  
Docker containers are easy to deploy in a cloud. It can get more applications running on the same hardware than other technologies, it makes it easy for developers to quickly create, ready-to-run containerized applications and it makes managing and deploying applications much easier. You can even share containers with your applications.  
If you have some more points to add you can do that but make sure the above the above explanation is there in your answer.

### ****Q8.**** ****What is Docker Swarm?****

You should start this answer by explaining Docker Swarn. It is native clustering for Docker which turns a pool of Docker hosts into a single, virtual Docker host. Docker Swarm serves the standard Docker API, any tool that already communicates with a Docker daemon can use Swarm to transparently scale to multiple hosts.  
I will also suggest you to include some supported tools:

* Dokku
* Docker Compose
* Docker Machine
* Jenkins

### ****Q9. What is Dockerfile used for?****

This answer according to me should begin by explaining the use of Dockerfile. Docker can build images automatically by reading the instructions from a Dockerfile.  
Now I suggest you to give a small definition of Dockerfle. A Dockerfile is a text document that contains all the commands a user could call on the command line to assemble an image. Using docker build users can create an automated build that executes several command-line instructions in succession.

Now expect a few questions to test your experience with Docker.

### ****Q10.**** ****Can I use json instead of yaml for my compose file in Docker?****

You can use json instead of yaml for your compose file, to use json file with compose, specify the filename to use for eg:  
**docker-compose -f docker-compose.json up**

### ****Q11.**** ****Tell us how you have used Docker in your past position?****

Explain how you have used Docker to help rapid deployment. Explain how you have scripted Docker and used Docker with other tools like Puppet, Chef or Jenkins. If you have no past practical experience in Docker and have past experience with other tools in similar space, be honest and explain the same. In this case, it makes sense if you can compare other tools to Docker in terms of functionality.

### ****Q12. How to create Docker container?****

I will suggest you to give a direct answer to this. We can use Docker image to create Docker container by using the below command:  
**docker run -t -i <image name> <command name>**This command will create and start container.  
You should also add, If you want to check the list of all running container with status on a host use the below command:  
**docker ps -a**

### ****Q13. How to stop and restart the Docker container?****

In order to stop the Docker container you can use the below command:  
**docker stop <container ID>**  
Now to restart the Docker container you can use:  
**docker restart <container ID>**

### Q14. How far do Docker containers scale?

Large web deployments like Google and Twitter, and platform providers such as Heroku and dotCloud all run on container technology, at a scale of hundreds of thousands or even millions of containers running in parallel.

### Q15. What platforms does Docker run on?

I will start this answer by saying Docker runs on only Linux and Cloud platforms and then I will mention the below vendors of Linux:

* Ubuntu 12.04, 13.04 et al
* Fedora 19/20+
* RHEL 6.5+
* CentOS 6+
* Gentoo
* ArchLinux
* openSUSE 12.3+
* CRUX 3.0+

Cloud:

* Amazon EC2
* Google Compute Engine
* Microsoft Azure
* Rackspace

**Note that Docker does not run on Windows or Mac.**

### ****Q16. Do I lose my data when the Docker container exits?****

You can answer this by saying, no I won’t loose my data when Dcoker container exits. Any data that your application writes to disk gets preserved in its container until you explicitly delete the container. The file system for the container persists even after the container halts.

## **Additional Questions**

### ****1. How does HTTP work?****

The HTTP protocol works in a client and server model like most other protocols. A web browser using which a request is initiated is called as a client and a web server software which responds to that request is called a server. World Wide Web Consortium and the Internet Engineering Task Force are two important spokes in the standardization of the HTTP protocol. HTTP allows improvement of its request and response with the help of intermediates, for example a gateway, a proxy, or a tunnel. The resources that can be requested using the HTTP protocol, are made available using a certain type of URI (Uniform Resource Identifier) called a URL (Uniform Resource Locator). TCP (Transmission Control Protocol) is used to establish a connection to the application layer port 80 used by HTTP.

### ****2. Explain your understanding and expertise on both the software development side and the technical operations side of an organization you’ve worked for in the past.****

DevOps engineers almost always work in a 24/7 business critical online environment. I was adaptable to on-call duties and able to take up real-time, live-system responsibility. I successfully automated processes to support continuous software deployments. I have experience with public/private clouds, tools like Chef or Puppet, scripting and automation with tools like Python and PHP, and a background in Agile.

### ****3. Discuss your experience building bridges between IT Ops, QA and development.****

DevOps is all about effective communication and collaboration. I’ve been able to deal with production issues from the development and operations sides, effectively straddling the two worlds. I’m less interested in finding blame or playing the hero than I am with ensuring that all of the moving parts come together.

### ****4. What types of testing are needed?****

Software teams will often look for the “fair weather” path to system completion; that is, they start from an assumption that software will usually work and only occasionally fail. I believe to practice defensive programming in a pragmatic way, which often means assuming that the code will fail and planning for those failures. I try to incorporate unit test strategy, use of test harnesses, early load testing; network simulation, A/B and multi-variate testing  etc.

### ****5. Give me an example of how you would handle projects?****

As a professional with managerial responsibilities, I would demonstrate a clear understanding of DevOps project management tactics and also work with teams to set objectives, streamline workflow, maintain scope,  research and introduce new tools or frameworks, translate requirements into workflow and follow up. I would resort to CI, release management and other tools to keep interdisciplinary projects on track.

### ****6. What’s your career objective in your role as a DevOps engineer?****

My passion is breaking down the barriers and building and improving processes, so that the engineering and operations teams work better and smarter. That’s why I love DevOps. It’s an opportunity to be involved in the entire delivery system from start to finish.

### ****7. How would you make software deployable?****

The ability to script the installation and reconfiguration of software systems is essential towards controlled and automated change. Although there is an increasing trend for new software to enable this, older systems and products suffer from the assumption that changes would be infrequent and minor, and so make automated changes difficult. As a professional who appreciates the need to expose configuration and settings in a manner accessible to automation, I will work with concepts like Inversion of Control (IoC) and Dependency Injection, scripted installation, test harnesses, separation of concerns, command-line tools, and infrastructure as code.

## **8. What is the one most important thing DevOps helps do?**

The most important thing DevOps helps do is to get the changes into production as quickly as possible while minimizing risks in software quality assurance and compliance. That is the primary objective of DevOps. However, there are many other positive side-effects to DevOps. For example, clearer communication and better working relationships between teams which creates a less stressful working environment.

### ****9. Which scripting languages do you think are most important for a DevOps engineer?****

As far as scripting languages go, the simpler the better. In fact, the language itself isn’t as important as understanding design patterns and development paradigms such as procedural, object-oriented, or functional programming.

### ****10. How do you expect you would be required to multitask as a DevOps professional?****

I believe I’ll be expected to:

1. Focus attention on bridging communication gaps between Development and Operations teams.
2. Understand system design from an architect’s perspective, software development from a developer’s perspective,operations and infrastructure from the perspective of a seasoned Systems Administrator.
3. Execute – to be able to actually do what needs to be done.

### ****11. What testing is necessary to ensure that a new service is ready for production?****

DevOps is all about continuous testing throughout the process, starting with development through to production. Everyone shares the testing responsibility. This ensures that developers are delivering code that doesn’t have any errors and is of high quality, and it also helps everyone leverage their time most effectively.

### ****12. What’s a PTR in DNS?****

Pointer records are used to map a network interface (IP) to a host name. These are primarily used for reverse DNS. Reverse DNS is setup very similar to how normal (forward) DNS is setup.  When you delegate the DNS forward, the owner of the domain tells the registrar to let your domain use specific name servers.

### ****13. Describe two-factor authentication?****

Two-factor authentication is a security process in which the user provides two means of identification from separate categories of credentials; one is typically a physical token, such as a card, and the other is typically something memorized, such as a security code.

### ****14. Tell us about the CI tools that you are familiar with?****

The premise of CI is to get feedback as early as possible because the earlier you get feedback, the less things cost to fix. Popular open source tools include Hudson, Jenkins, CruiseControl and CruiseControl.NET. Commercial tools include ThoughtWorks’ Go, Urbancode’s Anthill Pro, Jetbrains’ Team City and Microsoft’s Team Foundation Server.

### ****15. What are the advantages of NoSQL database over RDBMS?****

The advantages are:

1. Less need for ETL
2. Support for unstructured text
3. Ability to handle change over  time
4. Breadth of functionality
5. Ability to scale horizontally
6. Support for multiple  data structures
7. Choice of vendors

### ****16. What is an MX record in DNS?****

MX records are mail exchange records used for determining the priority of email servers for a domain. The lowest priority email server is the first destination for email. If the lowest priority email server is unavailable, mail will be sent to the higher priority email servers.

### ****17. What is the difference between RAID 0 and RAID 1?****

RAID 1 offers redundancy through mirroring, i.e., data is written identically to two drives. RAID 0 offers no redundancy and instead uses striping, i.e., data is split across all the drives. This means RAID 0 offers no fault tolerance; if any of the constituent drives fails, the RAID unit fails.

### ****18. How would you prepare for a migration?****

Tips to answer: This question evaluates your experience of real projects with all the awkwardness and complexity they bring. Include terms like cut-over, dress rehearsals, roll-back and roll-forward, DNS solutions, feature toggles, branch by abstraction, and automation in your answer. Developing greenfield systems with little or no existing technology in place is always easier than having to deal with legacy components and configuration. As a candidate if you appreciate that any interesting software system will in effect be under constant migration, you will appear suitable for the role.

### ****19. What’s your systems background?****

Tips to answer: Some DevOps jobs require extensive systems knowledge, including server clustering and highly concurrent systems. As a DevOps engineer, you need to analyze system capabilities and implement upgrades for efficiency, scalability and stability, or resilience. It is recommended that you have a solid knowledge of OSes and supporting technologies, like network security, virtual private networks and proxy server configuration.

DevOps relies on virtualization for rapid workload provisioning and allocating compute resources to new VMs to support the next rollout, so it is useful to have in-depth knowledge around popular hypervisors. This should ideally include backup, migration and lifecycle management tactics to protect, optimize and eventually recover computing resources. Some environments may emphasize microservices software development tailored for virtual containers. Operations expertise must include extensive knowledge of systems management tools like Microsoft System Center, Puppet, Nagios and Chef. DevOps jobs with an emphasis on operations require detailed problem-solving, troubleshooting and analytical skills.

### ****20. What DevOp tools have you worked with?****

Tips to answer: Software configuration management and build/release (version control) tools, including Apache Subversion, Mercurial, Fossil and others, help document change requests. Developers can more easily follow the company’s best practices and policies while software changes.

Continuous integration (CI) tools such as Rational Build Forge, Jenkins and Semaphore merge all developer copies of the working code into a central version. These tools are important for larger groups where teams of developers work on the same codebase simultaneously. QA experts use code analyzers to test software for bugs, security and performance. If you’ve used HP’s Fortify Static Code Analyzer, talk about how it identified security vulnerabilities in coding languages. Also speak about tools like GrammaTech’s CodeSonar that you used to identify memory leaks, buffer underruns and other defects for C/C++ and Java code. It is essential that you have adequate command of the principal languages like Ruby, C#, .NET, Perl, Python, Java, PHP, Windows PowerShell, and are comfortable with the associated OS environments Windows, Linux and Unix.

### ****21. How much have you interacted with cloud based software development?****

Tips to answer: Share your knowledge around use of cloud platforms, provisioning new instances, coding new software iterations with the cloud provider’s APIs or software development kits, configuring clusters to scale computing capacity, managing workload lifecycles and so on. This is the perfect opportunity to discuss container-based cloud instances as an alternative to conventional VMs. Event-based cloud computing, such as AWS Lambda offers another approach to software development, a boon for experienced DevOps candidates. In your interview, mention experience handling big data, which uses highly scalable cloud infrastructures to tackle complex computing tasks.

### ****22. What other tools are you familiar with that might help you in this role?****

Tips to answer: DevOps is so diverse and inclusive that it rarely ends with coding, testing and systems. A DevOps project might rely on database platforms like SQL or NoSQL, data structure servers like Redis, or configuration and management issue tracking systems like Redmine. Web applications are popular for modern enterprises, making a background with Web servers, like Microsoft Internet Information Services, Apache Tomcat or other Web servers, beneficial. Make sure to bring across that you are familiar with Agile application lifecycle management techniques and tools.

### ****23. Are you familiar with just Linux or have you worked with Windows environments as well?****

Tips to answer: Demonstrate as much as you can, a clear understanding of both the environments including the key tools.

### ****24. How can you reduce load time of a dynamic website?****

Tips to answer: Talk about Webpage optimization, cached web pages, quality web hosting , compressed text files, Apache  fine tuning.

### ****25. Describe your experience implementing continuous deployment?****

Tips to answer: Answer with a comprehensive list of all the tools that you used. Include inferences of the challenges you faced and how you tackled them.

### ****26. How would you ensure traceability?****

Tips to answer: This question probes your attitude to metrics, logging, transaction journeys, and reporting. You should be able to identify that metric, monitoring and logging needs to be a core part of the software system, and that without them, the software is essentially not going to be able to appear maintained and diagnosed. Include words like SysLog, Splunk, error tracking, Nagios, SCOM, Avicode in your answer.

### ****27. What was your greatest achievement on a recent project?****

Tips to answer: Make sure you demonstrate your perfect understanding of both development and operations. Do not let your answer lean towards one particular skillset ignoring the other. Even if you have worked in an environment wherein you had to work more with one skillset, assure the intervewer that you are agile according to the needs of your organization.

### ****28. What problems did you face and how did you solve them in a way that met the team’s goals?****

Tips to answer: This questions aims to find out how much you can handle stress and non-conformity at work. Talk about your leadership skills to handle and motivate the team to solve problems together.Talk about CI, release management and other tools to keep interdisciplinary projects on track.

### ****29. Are you more Dev or Ops?****

Tips to answer: This is probably the trickiest question that you might face in the interview. Emphasize the fact that this depends a lot on the job, the company you are working for and the skills of people involved. You really have to be able to alternate between both sides of the fence at any given time. Talk about your experience and demonstrate how you are agile with both.

### ****30. What special training or education did it require for you to become a DevOps engineer?****

Tips to answer: DevOps is more of a mind-set or philosophy rather than a skill-set. The typical technical skills associated with DevOps Engineers today is Linux systems administration, scripting, and experience with one of the many continuous integration or configuration management tools like Jenkins and Chef. What it all boils down to is that whatever skill-sets you have, while important, are not as important as having the ability to learn new skills quickly to meet the needs. It’s all about pattern recognition, and having the ability to merge your experiences with current requirements.Proficiency in Windows and Linux systems administration, script development, an understanding of structured programming and object-oriented design, and experience creating and consuming RESTful APIs would take one a long way.

[**View Upcoming DevOps Batches Now >>**](https://www.edureka.co/devops)

## **DevOps Engineer - The buzz that never ENDS!**



Looking to become or hire a DevOps Engineer? Wait my friend; before you proceed any further and understand the core responsibilities of a DevOps Engineer, let me make things clearer to you. You can often listen to two great misconceptions tied to DevOps culture that do a great harm; people think that:

* DevOps engineers are Sysadmin who can even code and are the ones who are solely responsible for system uptime
* Sysadmins are not necessary for a DevOps environment since developers take over system admin tasks

**And you know what, both are WRONG!!**

## **So What Exactly is DevOps Culture then?**

The original idea of DevOps is to smooth out the interaction between development and operations. It is a method which will help you to achieve great results – not a job position. You can define DevOps as**sysadmin participating in an agile development process alongside developers and using the same agile techniques for their systems work.**

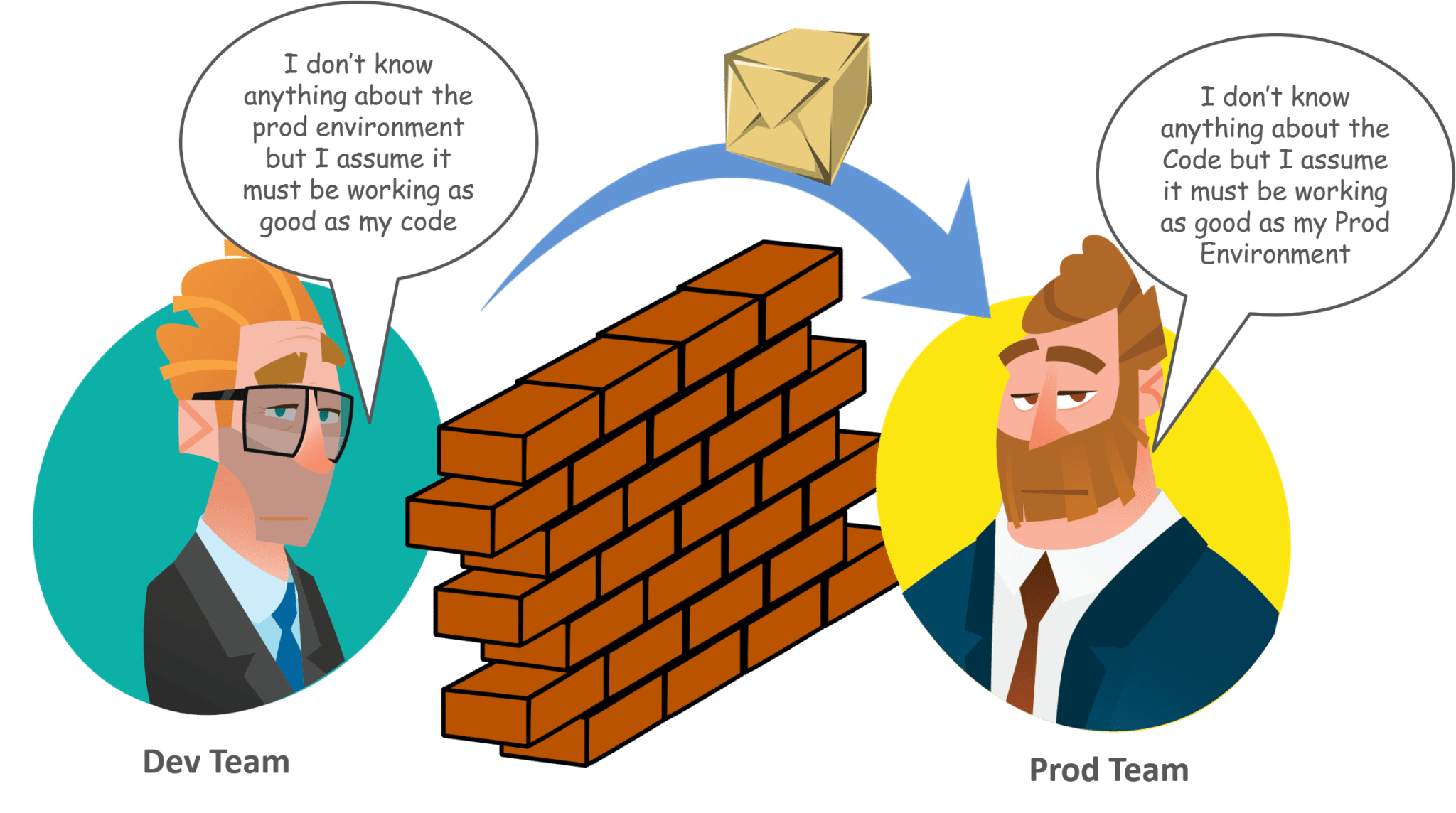
DevOps means different things to different people, as “agile development” itself covers a lot of ground.

The “DevOps” movement is about having everyone understand how the entire system works, and everyone being able to express what their underlying business value is.

* Developers are all about innovation, making something cool and new
* Sysadmins are focused on never letting the site go down

Before DevOps came along developers weren’t incentivized to worry about the site uptime and vice versa

Want to become a DevOps Engineer?[Click and start your career in DevOps](https://www.edureka.co/devops)



## **But WHO are these DevOps Engineer? and WHY should anyone hire a DevOps Engineer?**

First of all **WHO is this DevOps Engineer**? Well, these are people more precisely an elite group of people who not only have years of experience but also have in-depth knowledge of the technicalities in the domain. Let me clear the **FACT**that there’s no formal career track or an Abra-Kadabra trick for becoming a DevOps engineer. These people are either developers who get interested in deployment and network operations, or are sysadmins who have a passion for scripting and coding, and move into the development side where they can improve the planning of test and deployment. Either way, these people are the one who has pushed beyond their defined areas of competence and have a more holistic view of their technical environments.

It won’t be surprising if while looking for the DevOps Engineer role you found a smaller number of companies creating that title. If DevOps is understood primarily as a mindset, it can get really ambiguous and awfully fuzzy. Let’s check out this list of core DevOps attributes:

* Ability to use a wide variety of open source technologies and tools
* Ability to code and script
* Experience with systems and IT operations
* Comfort with frequent, incremental code testing and deployment
* Strong grasp of automation tools
* Data management skills
* A strong focus on business outcomes
* Comfort with collaboration, open communication and reaching across functional borders

**What does any Company ultimately wants to get successful – *Quality with Speed at Least expenditure.***

Coming to the next question **WHY or WHEN should anyone even hire a DevOps Engineer?** Looking the current market trend, DevOps is an essential role to hire full-time early on. Hire people when your product has shipped, or when you have at least three developers on board. Live products require more monitoring than you can expect. If you haven’t hired this person yet, realize that DevOps work is already happening in the background and your team would move faster if that function was explicitly owned.

If you’re not ready to make this leap, consider talking to a consultant to help set a basic foundation. With the right plan, you could hire freelancers in order to manage costs but, this will only be a temporary solution.

A great DevOps engineer is seriously a game changer. As a DevOps engineer, you will ensure that the system is running smoothly and being monitored so that you can respond to issues as it arises. You will ensure that your developers are never doing repetitive tasks, and the infrastructure is kept up to date as the stack evolves. As the processes change and the company grows, the DevOps engineer automates as much as possible to accelerate work. Because of DevOps, developers can focus on their core work so you can deliver products earlier and more reliably.

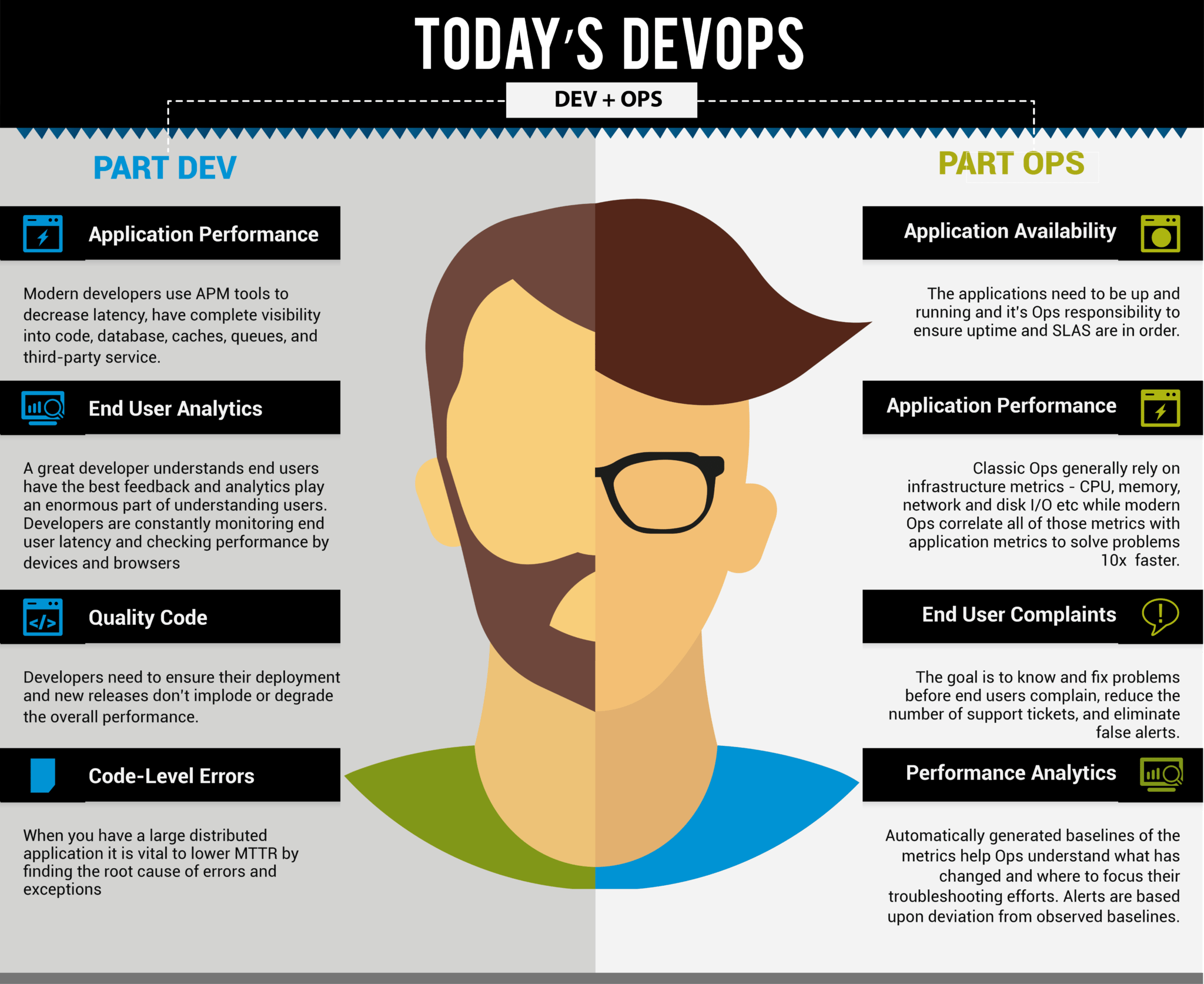
## **MWare**

**Key Responsibilities:**

* Design, build and maintain the CI/CD infrastructure and tools to deliver Horizon Cloud Service
* Design and Develop the test automation to validate the builds in the CI/CD pipeline
* Work closely with development teams to ensure that solutions are designed with customer user experience, scale/performance and operability in mind

**Requirement and Preferred Skill:**

* Bachelors degree in Computer Science or equivalent
* 5+ years of experience with Python other scripting languages such as Perl, Ruby etc
* Experience in developing and maintaining CI/CD process for enterprise SaaS and on-prem applications using tools like Git, Gerrit, Jenkins, Maven, Gradle, etc
* Expertise in Linux System Admin and Bash Scripting
* Experience with configuration management tools such as Ansible, Chef, Fabric, Puppet or SaltStack
* Hands-on experience in build and administer VMs and Containers using tools such as Docker, Vagrant, Kubernetes
* Experience in designing and maintaining cloud-based solutions with Azure



## **What is the Average DevOps Engineer Salary?**

IBM: 506143 per annum

Amazon: 1012629 per annum

Accenture: 711800 per annum

TCS: 549074 per annum

Sprinklr: 1028933 per annum

Infosys: 616667 per annum

OLA: 1250000 per annum

Directi: 1250000 per annum

## **Interested in becoming a DevOps Engineer! But HOW should you prepare?**

**“Dreams doesn’t work unless you do”**

If you are really interested in becoming a DevOps Engineer then put all your efforts in. If interested you can visit the following links and keep yourself focused to become a DevOps Engineer

Check out the blog about top 10 DevOps tools you must learn in 2018

[DevOps Top 10 Tools you must learn in 2018](https://goo.gl/8DxR1S)

Check this YouTube playlist which consists of more than 100+ free videos on DevOps.

[DevOps Tutorial YouTube Playlist](https://goo.gl/nzyZZx)

If you want to really wanna get DevOps certification for your career, you can visit our landing page and check out the content and features of [DevOps Training and Certification](http://www.edureka.co/devops) from Edureka.

## **Crack DevOps Engineer Interview**

If you are a DevOps engineer or looking forward to becoming one, then with a strategic understanding of tools, processes and utilization of technology, be assured that 2018 is going to be your year. According to Gartner, organizations around the world are increasingly adopting the DevOps culture and by the end of 2016, 25 percent of top global 2000 organizations have already adopted DevOps as a mainstream strategy.

If you plan to give an interview for a DevOps-centric role, here is a comprehensive list of the most popular[**DevOps Interview Questions**](https://www.edureka.co/blog/interview-questions/top-devops-interview-questions-2016/)**.** These questions are framed keeping in mind the business requirement of the current market. Always remember the FACT:

**“The EXPERT in anything was once a BEGINNER”**

## **DevOps Practice**

In reality, DevOps practice emphasizes that, a ‘Dev’ guy should be able to perform the tasks carried out by an ‘Ops’ guy and an ‘Ops’ guy should be able to perform the tasks carried out by a ‘Dev’ guy.

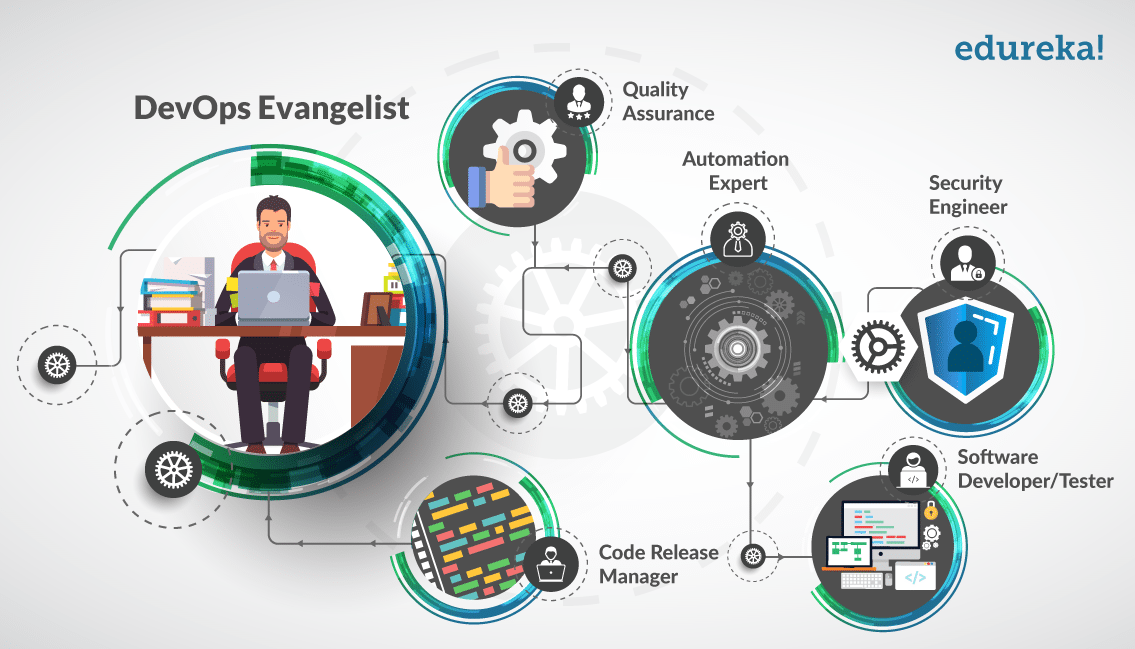
Now that you know a DevOps professional is one with multiple skills, are you also wondering, what are the various tasks a DevOps professional is responsible for in his organization? If that’s the case, read on.

## **DevOps Roles**

Let me first list down the various roles along with their description.

1. [**DevOps Evangelist**](https://www.edureka.co/blog/devops-roles-which-is-your-dream/#DevOpsEvangelist) – The principal officer (leader) responsible for implementing DevOps
2. [**Release Manager**](https://www.edureka.co/blog/devops-roles-which-is-your-dream/#ReleaseManager) – The one releasing new features & ensuring post-release product stability
3. [**Automation Expert**](https://www.edureka.co/blog/devops-roles-which-is-your-dream/#AutomationExpert) – The guy responsible for achieving automation & orchestration of tools
4. [**Software Developer/ Tester**](https://www.edureka.co/blog/devops-roles-which-is-your-dream/#SWDeveloperOrTester) – The one who actually develops the code and tests it
5. [**Quality Assurance**](https://www.edureka.co/blog/devops-roles-which-is-your-dream/#QualityAssurance) – The one who ensures quality of the product confirms to its requirement
6. [**Security Engineer**](https://www.edureka.co/blog/devops-roles-which-is-your-dream/#SecurityEngineer) – The one always monitoring the product’s security & health

One important point to note here is that, a lot of these roles will involve a team or group of professionals like Software Developers/ Testers, Security Engineers & Quality Assurance. So you should think of these roles & their relation in the form of a hierarchy.



Based on the company’s requirement, even Automation Experts can have more than one person donning the role. So, let me talk more about each of these DevOps roles.

### ****DevOps Evangelist****

Like I mentioned earlier, this is the key role in DevOps. This role is assumed by a leader or that person who is responsible for ensuring that the DevOps strategy is implemented in the end-to-end development of the product, while bringing about a positive difference in the environment.

The DevOps Evangelist also finds ways to improve the existing architecture of the product keeping in mind the various automation tools available and the skills that ‘Dev’ and ‘Ops’ guy consist of. Managing other DevOps roles and obtaining full efficiency from the team is his primary target.

### ****Release Manager****

This role is a key player in the product deliverables because he needs to integrate the new/ updated features with the existing framework. This role can either be the same guy assuming the role of Release Engineer or a Release Manager who is leading a team of Release Engineers.

This role’s key area of focus is to co-ordinate and manage the product from development through deployment. Since this role is involved in an important manner, it should be donned by a technical person (manager) that understands how technology works and how various structures fall in place.

Depending on the requirement, there will be multiple Managers leading their respective teams working on multiple features (projects).

### ****Automation Expert****

The importance of this role need not be emphasized again because DevOps is all about automation. And being an automation expert, this role should be able understand what can be automated and how a product stack can be integrated with another product stack.

This role which is also called Integration Specialists, analyzes, designs, and implements strategies for continuous deployments while ensuring high availability on production and pre-production systems.

### ****Software Developer/ Tester****

This role(s) does the actual root-level development of the software. The traditional coders and programmers fall under this bracket. Besides development, the professionals are also responsible for testing the code.

The designation may remain the same, but the new role of software developer/tester dramatically increases the scope of responsibilities as they are responsible not only for turning new requirements into code, but unit testing, deployment, and ongoing monitoring as well.

### ****Quality Assurance****

This is the new role introduced in DevOps, which actually guarantees the quality of the product. Often also referred to as Experience Assurance, this role is a step beyond traditional testing and quality checking.

Here, the functionality of the product is tested to its limits to bring out every flaw and to improve on the under performance of every standalone feature because the experience of your client matters. This role is always in the lookout for opportunities to improve any and every feature to bring customer satisfaction.

### ****Security Engineer****

Security Engineers are as important as any other role because they are the one’s monitoring the deliverability of the product. They work side by side with developers, embedding their recommendations (security patches) much earlier on in the process.

Besides external security, they also monitor the systems to check its performance, report any downtime faced by the system and drill down to find out what caused it. Preventing any downtime or security risk from happening again is a key task of Security Engineers.

I hope by now you got a good understanding of the various DevOps roles that can exist in an organization.

## **evOps Skills:**

DevOps is a buzzword in the industry right now. Every organization is using DevOps practices, but what are these companies looking for, in a DevOps engineer. This blog on DevOps skills will explain the skills required to become a successful DevOps professional. These skills are collected after consulting the recruiters from various companies.

Below are the 6 most important DevOps skills, that organizations look for:

1. [Linux Fundamentals And Scripting](https://www.edureka.co/blog/devops-skills#Linux)
2. [Knowledge On Various DevOps Tools And Technologies](https://www.edureka.co/blog/devops-skills#knowledge)
3. [Continuous Integration And Continuous Delivery](https://www.edureka.co/blog/devops-skills#CI)
4. [Infrastructure as Code (IAC)](https://www.edureka.co/blog/devops-skills#IAC)
5. [DevOps Key Concepts](https://www.edureka.co/blog/devops-skills#key)
6. [Soft Skills](https://www.edureka.co/blog/devops-skills#soft)

## Linux Fundamentals And Scripting:

As a DevOps professional you might have to provision your infrastructure, therefore in order to automate it, you should know at least one scripting language.

Most of the companies have their environment on Linux, also many CM tools like – Puppet, Chef and Ansible have their master nodes on Linux.

To become a DevOps engineer you should be aware of Linux fundamentals and knowledge on one scripting language is a must. Scripting language can be any for example – Python, Ruby, Pearl etc.

## **Knowledge On Various DevOps Tools And Technologies:**

DevOps involves various phases and there are multiple tools available for those stages. You should have good  knowledge on these **development**, **testing** and **deployment** technologies.

## Knowledge On Various DevOps Tools - DevOps Skills - Edureka

## **Continuous Integration And Continuous Delivery:**

Knowledge on various tools is not enough, you should know where to use these tools.

These tools should be used in order to facilitate Continuous Integration and Continuous Delivery. Even Continuous Deployment in few cases, but Continuous Deployment is not considered as a good practice. Consider the below diagram to understand the difference.

## Continuous Integration vs Continuous Delivery vs Continuous Deployment - DevOps Skills - Edureka

## **Infrastructure As Code:**

Infrastructure as Code (IAC) is a type of IT infrastructure that operations teams can automatically manage and provision through code, rather than using a manual process. Infrastructure as Code is sometimes referred to as programmable infrastructure.

With the advent of IAC, the line between a developer and a sysops guy is getting blurred.

There are multiple tools available, like – Puppet, Chef, Ansible, Saltstack etc.

You can refer the below blog series:

* [**Puppet**](https://www.edureka.co/blog/what-is-puppet/)
* [**Chef**](https://www.edureka.co/blog/what-is-chef/)
* [**Ansible**](https://www.edureka.co/blog/what-is-ansible/)

If you want to become a DevOps professional, this skill is a must.

## Infrastructure As Code - DevOps Skills - Edureka

## **DevOps Key Concepts:**

Till now, I have only discussed the technical skills. But remember, DevOps is not a technology, it is a methodology.

This methodology aims at bringing the **Dev** and the **Ops** part of the organization together to release good quality software in time. If you understand the key concepts of this methodology, you will be able to provide better solutions to various business problems.

Refer the below blog to understand the DevOps methodology:

[**DevOps Tutorial**](https://www.edureka.co/blog/devops-tutorial)

## **Soft Skills:**

Whenever developers and operations staff communicate with each other clearly, they not only develop and release software in time with good quality but, can help in marketing the application with lower costs and better quality.

A DevOps professional also sometimes help in cultivating positive relationships with business and customers.

Those interested in the field need to be able to listen attentively, negotiate, solve problems and build teams.