Chef Best Practices

Configurations Management, using tools such as Chef, is an important part of DevOps, and needs to be done right to reap the full benefits of DevOps. In this post, we will look at some of the Chef Best Practices to follow when it comes to deploying 3-Tier Applications.

1. Start by keeping the Web, App and Database (DB) VPC different. This allows for reduced single point of failures and makes configurations easier.
2. Use the same configuration management code for different environments such as Dev, Test, QA, Staging, and Prod.
3. Use Immutable Infrastructure to reduce downtime and improve ease of upgradability. Infrastructure as code is a must.
4. Do not pre-bake the binaries with configuration values, but dynamically set them during deployment.
5. Keep the cookbooks separate and create readymades, with default values over the cookbooks for each environment.
6. Server configuration validation, using tools such as ServerSpec, are a must to validate the environment and configurations created by Chef and other automation tool

Some other Chef Best Practices and guidelines to follow are as below:

1. Maintaining Consistency: All instances should run the current approved version of code from a single truth source repository.
2. Use Common Archive: Deploy your apps and cookbooks from a common archive to ensure that all your instances have the approved code version deployed, thus guaranteeing the code is a static artifact.
3. Automate the Deployment Process: Create an automated deployment pipeline using tools such as Jenkins.
4. Deploying Code to Production: Deploy the update to every instance concurrently by executing a default “Deploy” command (for Apps) or “Update Custom Cookbooks” command (For Cookbooks).
5. Use a Rolling Deployment: Update an application in multiple phases.
6. Development Stacks: Use a development stack for tasks such as implementing new features or fixing bugs. A development stack is essentially a prototype production stack, with the same layers, apps, resources, and so on that are included on your production stack.
7. Production Stack: The production stack is the public-facing stack that supports your current application.
8. Use a Blue-Green Deployment Strategy: A blue-green deployment strategy is one common way to efficiently use separate stacks to deploy an application update to production. The blue environment is the production stack, which hosts the current application. The green environment is the staging stack, which hosts the updated application. Switch user traffic from the blue stack to the green stack when you are ready to deploy the updated application to production.