
Linux Academy.com
Learning Chef
Infrastructure Automation With Chef

Linux Academy.com

Learning Chef

After this course you will be able to:

- Understand DevOps and what Chef is
- Know the role of workstations, nodes, and Chef server
- Deploy and automate configurations of nodes
- Understanding of writing Recipes and Cookbooks
- Understand the Chef work flow
- Use Chef to automate the deployment your infrastructure

Linux Academy.com

Learning Chef

How this course is laid out:

- Will learn Chef by working with single nodes to start
- Learning to build cookbooks will follow a theme such as installing LAMP
- Additional exercises will be found at the end of the course
- Need access to LinuxAcademy.com lab servers or your own VM environment
  - VirtualBox
  - VMWare workstation
  - Parallels
- Have at least one “node” (server) available to connect to

Learning Chef

How this course is laid out:

- Will learn Chef by working with single nodes to start
- Need access to LinuxAcademy.com lab servers or your own VM environment
  - VirtualBox
  - VMWare workstation
  - Parallels
- Have at least one “node” (server) available to connect to


Linux Academy.com
Learning Chef
What Is DevOps?

Learning Chef

What is DevOps?

- DevOps is about *"How well people work together and how streamlined our Operations really are"* – Adam Jacob
- The application and infrastructure that runs on it are not treated as separate entities to each other and neither are the teams that manage each
- DevOps is part of continuous delivery where all aspects of the deployment process are automated
- DevOps is infrastructure as code


Linux Academy.com
Learning Chef
What Is Chef?

Learning Chef

What is Chef?

"Chef turns infrastructure into code. With Chef, you can automate how you build, deploy, and manage your infrastructure. Your infrastructure becomes as versionable, testable, and repeatable as application code.

Chef relies on reusable definitions known as recipes to automate infrastructure tasks. Examples of recipes are instructions for configuring web servers, databases and load balancers. Together, recipes describe what your infrastructure consists of and how each part of your infrastructure should be deployed, configured and managed."

Getchef.com

Learning Chef

What is Chef?

- Recipes are created using the Ruby language and while learning Chef we will learn a lot of the Ruby language
- Recipes rely primarily on *resources,* resources described a desired state of an element in the infrastructure.
  - Packages to be installed
  - Files to be created
  - Directories to be created
  - Services to be started
  - Config files to be updated
  - Commands to be executed
  - Essentially anything we need to do on our Linux nodes

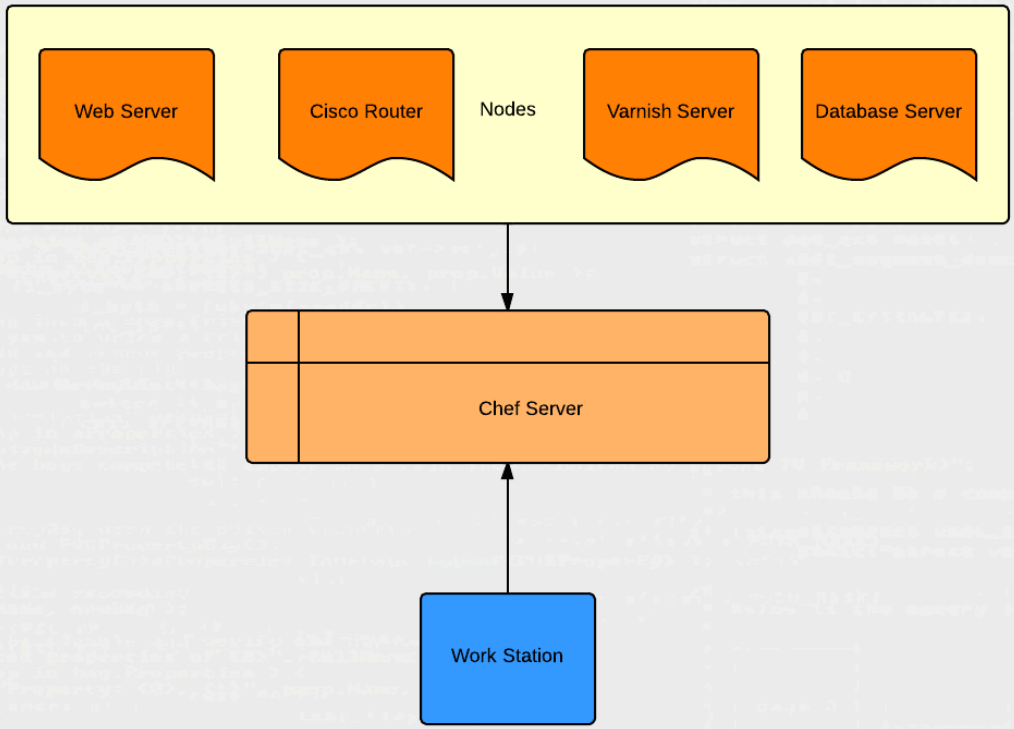
Learning Chef

What is Chef?

- Chef relies on either OpenSource Chef server or Chef enterprise to host configuration recipes, cookbooks, and node authentication for your infrastructure
- Nodes when performing a *convergence* will check in with the Chef server, download the required configuration, then execution the *recipe* instructions

Linux Academy.com

Learning Chef




Linux Academy.com
Learning Chef
Common Chef Terminology

Common Chef Terminology

Recipes:
Fundamental configuration element within an organization

Cookbook:
Defines a scenario and is the fundamental unit of configuration and policy distribution

Chef-Client:
Agent that runs locally on the node that is registered with the chef server

Convergence:
Occurs when chef-client configures the system/node based off the information collected from chef-server

Configuration Drift:
Occurs when the node state does not reflect the updated state of polices/configurations on the chef server

Resources:
A statement of configuration policy within a recipe
Describes the desired state of an element in the infrastructure and steps needed to configure

Common Chef Terminology

Provider:
Defines the steps that are needed to bring the piece of the system from its current state to the desired state

Attributes:
Specific details about the node, used by chef-client to understand current state of the node, the state of the node on the previous chef-client run, and the state of the node at the end of the client run

Data-bags:
A global variables stored as JSON data and is accessible from the Chef server

Workstation:
A computer configured with Knife and used to synchronize with chef-repo and interact with chef server

Chef Server:
Chef server is the hub for all configuration data, stores cookbooks, and the policies applied to the node

Knife:
Command line tool which provides an interface between the local chef-repo and chef-server

client.rb:
Configuration file for chef-client located at /etc/chef/client.rb on each node

Ohai:
Tool used to detect attributes on a node and then provide attributes to chef-client at the start of every chef-client run

Common Chef Terminology

Node Object:
Consists of run-list and node attributes that describe states of the node

Chef-Repo:
Located on the workstation and installed with the starter kit, should be synchronized with a version control system and stores Cookbooks, roles, data bags, environments, and configuration files

Organization:
Used in chef enterprise server to restrict access to objects, nodes environments, roles, data-bags etc

Environments:
Used to organize environments (Prod/Staging/Dev/QA) generally used with cookbook **versions**

Idempotence:
Means a recipe can run multiple times on the same system and the results will always be identical


Linux Academy.com
Learning Chef
Chef Server

Chef Server

Two types of Chef-server:

1. OpenSource Chef-server
   - Free version of Chef
   - Each instance of the server must be configured and managed locally (includes all aspects of managing the server, updates, migrations, scalability, etc.

2. Chef-server enterprise (hosted)
   - Scalable by design
   - Available organizations
   - Always available
   - Resource-based access control

3. Chef-server enterprise (on-premise)
   - Scalable by design
   - Available organizations
   - Hosted on-premise behind your firewall
   - Managed yourself

Chef Server: Enterprise

- Allows creation of organizations
  - Organizations separate the infrastructure, policies, and cookbooks
  - Nodes are registered in organizations
  - Nothing can be shared between organizations
  - Enterprise chef server can contain many different organizations
  - OpenSource chef the local individual server acts as an organization and does not allow creation of organizations
  - Organizations can represent different companies, department, infrastructures, applications, and so forth
- For each organization in order to start bootstrapping nodes you need to download the starter kit
- Starter kit provides security credentials (validation.pem keys) to authenticate each node to the chef server
- Chef enterprise scales by design to handle thousands of nodes and different organizations

Chef Server: Role of the server

- Stores system configuration information (policies for nodes)
- Authenticates workstations and nodes
- Delivers configurations to nodes
- Chef server holds the configuration and the node checks-in to receive instructions on its desired state
- The node downloads configuration instructions from the server and does all of the work


Linux Academy.com
Learning Chef
Chef Workstation

Learning Chef

Chef Workstation

- Developing cookbooks and recipes
- Synchronizing chef-repo with git/svn (source control)
- Using knife to upload and manage node and chef server configurations and policies
  - Organizations
  - Nodes
  - Environments
  - Data bags

Chef Workstation: Organization Starter Kit

- Each organization has a starter kit which provides the chef-repo as well as security credentials for communicating with the server
- Knife.rb configuration file is located in the chef-repo (starter kit)
- Starter kit can be re-downloaded from the chef-server

Chef Workstation: Organization Starter Kit

- Each time you download a new starter kit the security keys for that repo to the server are reset
- As a result you have to redistribute the security keys to other team members if a new starter kit for the organization is downloaded
- During bootstrap oragnization-validator.pem is copied to the node as validation.pem
- During the first chef-client run the validation.pem signs the client.pem key which is used to authenticate the node against the chef server

Chef Workstation: Organization Starter Kit

- Security considerations for validation.pem
  - It should be removed first when the chef-client convergence runs
  - If chef-client locks then key is located on node if node is compromised then your entire infrastructure is
  - Chef-client (super market cookbook) provides this functionality
  - Can always write your own recipe to remove validation.pem and run it first in the run_list
    - file "/etc/chef/validation.pem" do
    - action :delete
    - End

Chef Workstation: Knife

Knife is the command line tool used to provide an interface between your local chef-repo and the chef-server

- Creating cookbooks
- Uploading cookbooks to chef server
- Managing roles and run_lists
- Searching chef-server node object data
- Bootstrapping nodes
- Essentially everything we need to do as a DevOps admin

Chef Workstation: Knife

Chef and knife interface can be installed to OSX and Linux:

curl –L https://www.opscode.com/chef/install.sh | sudo bash

Download and install Chef manually for windows:

http://www.getchef.com/chef/install/

This course will use a Linux CentOS 6.5 Image as the workstation

Chef Workstation: Knife configuration file (knife.rb)

- Is found ***inside*** the chef-repo which is available inside the starter kit of an organization
    - Configure proxy settings here if workstation is behind a proxy

Chef Workstation: Using knife to bootstrap a node

Bootstrapping connects the workstation to the node to install:

- Installs knife
- Ohai:
  Used to detect attributes on a node and report them to chef-client at the start of every client run it is required for chef-client to work and it builds the node object
- Ruby
- Chef-client
- A few other additional items

Learning Chef

Chef Workstation: Using knife to bootstrap a node

Knife bootstrap <address> -x user –P password –N nodename

- For starting in this course we will define a node name
- In production it is best practice not to define –N and let the FQDN work as the node name

Note: For linuxacademy.com lab servers when bootstrapping use your public hostname


Linux Academy.com
Learning Chef
Chef Server & Nodes

Chef Server & Nodes

A node can be *anything* that can run the chef-client agent

- Smart phones
- Switches
- APIs
- FreeBSD servers
- Unix Servers
- Windows Servers
- Linux Servers
- Etc.

Chef Server & Nodes

Chef server stores all policy and configuration information for nodes
The node uses the client.pem file (created by validator.pem) during the chef-client run in order to authenticate against the chef server

Chef-client run:

- Authenticate against the chef-server using client.pem
- Builds the node object and runs ohai
- Syncronizes with the chef-server (sends node object information and receives cookbooks/policies)
- Executes/compiles the desired policies
- Runs the node object
- Completes


Linux Academy.com
Learning Chef
Chef Configuration Concepts

Chef Configuration Concepts

Policy:

- A collection of system configurations that you define (roles/data bags/ environments)
- The policy states the state that each resources should be in but not how to get there
- Chef-client will pull the policy and configure the node so that it matches the state of the policy

- Policy concept examples:
  - *If* it should be installed
  - If it is not installed then install it
  - If it is already installed then do nothing
  - A file should exist if not create it
  - If a file exists but does not have correct content

Chef Configuration Concepts

Resources: Defines the desired state for a single configuration item present on a node that is under management by Chef

- Does the configuration on a node and maps to providers
- Recipes are stored in cookbooks
- Represent a piece of the system and its desired state
- Building blocks of Chef configuration
- When chef-client is run on a node the resource is executed by the provider which is handled by Chef and the OS itself
- Information as to what provider to use (ie what package manager to use) is populated when ohai is run at the start of each chef-client run

Chef Configuration Concepts

Most common ***resources*** in configuration management

Package:
Used to manage packages such as installing a required package

Template:
Used to manage the contents of a Ruby template in the cookbook

Service:
Manage system services
What run-levels to start the service in
Current state of the service (running/stopped/etc)

Chef-client nodes concepts

- Nodes can be configured to automatically running chef-client in intervals
- When convergence occurs the node can put itself back into compliance (desired state and how to get there)
- Chef server does not know the status of a node until a convergence is run
    - Something could happen to cause the node to get out of compliance but a convergence will check policies/configuration and put the node back into compliance based off your defined configurations

Learning Chef

Key concepts:

Recipes are made up of a collection of resources

Cookbooks are made up of a collection of recipes

Nodes receive their policy based off of roles and individual node configurations

A run list defines the order in which you want your recipes to run during convergence

Configuration drift occurs when the desired state of the node does not match the desired state as defined in the policies on the chef server

Configuration drift can be resolved by automating/running the chef-client agent


Linux Academy.com
Learning Chef
Attribute Precedence

Learning Chef

Key concepts:

Attributes are specific details about a node

Attributes describe:

- The current state of the node
- What the sate of the node was at the end of the previous chef-client run
- What the state of the node should be at the end of the current chef-client run

Attributes can be defined by:

- The state of the node (ohai)
- Cookbooks (Our attribute files)
- Roles
- Environments

Attribute Precedence

After the node object is rebuilt in the chef run, all attributes loaded in the chef-client are then compared. The node is updated based on attribute precedence and at the very end of the convergence (chef-client) the node object is uploaded to the chef server

Node object defines the current state of the node (made up of attributes)
Node object is stored on the chef server so that it can be searched
Node object is updated at each convergence

If there are attributes with the same names then attribute precedence determines which attribute is applied to the node and the node object.

Attribute Precedence: Levels of precedence

default:

Automatically reset at the start of every chef client run is the lowest level of precedence

force_default:

Used in a cookbook or recipe to override an existing “default” attribute

Normal:

A setting that persists in the node object

Override:

Automatically reset at the start of every chef-client most often should be used only when required

force_override:

used to ensure that an attribute defined in a cookbook (by an attribute file or by a recipe) takes precedence over an override attribute set by a role or an environment

Automatic:

Contains data populated by Ohai at the beginning of every chef-client run and cannot be modified and always has the highest attribute precedence

Attribute Precedence: Levels of precedence

	Attribute Files	Node / Recipe	Environment	Role
default	1	2	3	4
force_default	5	6		
normal	7	8		
override	9	10	12	11
force_override	13	14		
automatic		15		


Linux Academy.com
Learning Chef
Node Object

Node Object

Node object is made up of the run lists which define what recipes to run during a chef-client as well as the attributes that define information about the node

Attributes are built during the chef-client run process:

- Data about the node is collected by Ohai
- The node object previously saved during the last chef-client run
- The rebuilt node object from the current chef-client run

Once the node object is rebuilt all attributes are compared and then updated based on attribute precedence

At the end of every chef-client run the node object that defines the current state of the node is uploaded to the chef server to be searched.


Linux Academy.com
Learning Chef
Environments

Learning Chef

Environments: What are environments?

"An environment is a way to map an organizations real-life workflow to what can be configured and managed using the Chef server"

Apply different cookbook versions to specific environments (dev/prod/staging/qa/)

Define Environment level attributes

Environments allow sharing of cookbooks within an organization

Environments: Creating environments

- Environment information can be stored in JSON files or .rb files
- Environments will be located in chef-repo/environments

- Example dev.rb (development environment file)

```
name "dev"
description ”Development environment”
cookbook "security", "= 0.1.0”
cookbook ”motd", "= 0.2.0"
cookbook "apache", "= 0.2.0"
override_attributes({
“author" => {
        “name" => true
  }
})
```

Environments: Attributes

Two types of attribute precedence can be set on the environment level

- default a default attribute defined in the environment will take precedence over a default attribute defined in a cookbook attribute file.
- Override has higher precedence than default, force_default, and normal

Environments: Methods of assigning a node to an environment

- Modify the client.rb file with an environment variable
- Knife-flip to do it from the workstation
- Assign it manually on the node

Environments: Lab goals

- ✓ Configure two environments: Dev/Production
- ✓ Modify Apache cookbooks with newer versions
- ✓ Set newer versions to dev and older versions to production
- ✓ Configure Client.rb to look at proper environment


Linux Academy.com
Learning Chef
Search

Learning Chef

Search:

Chef search allows queries to be made for any type of data that is indexed by the chef server. Search queries the chef server for stored information.

Can Search:

- Data bags
- Environments
- Roles
- Nodes

Search: Methods of search

Search with Knife:

Syntax: knife search node “key:pattern” –a (attribute)

- Search nested attributes
  - “memory_total:*” –a memory.total
- Can use basic “wild cards” in the pattern “ipaddress:192.168.*”
- Can search based off ranges “ipaddress:[192.168.* TO 192.172.*]”

Search inside of recipes

Example:

Use Ruby recipe to search all nodes that are running an outdated package


Linux Academy.com
Learning Chef
Data Bags

Data Bags

A data bag is a global variable that is stored in JSON data and accessible from the chef server. A data bag is indexed for searching and can be loaded by a recipe or accessed during search.

Types of data stored in a data bag:

- ✓ Users to be added to a system
- ✓ Admins to be added to sudo
- ✓ API/DB Credentials (More secure and better than environment attributes for credentials)
- ✓ Much more

Learning Chef

Data Bags

```
{
        "id": “anthony",
        "comment": ”Anthony admin user",
        "uid": 2005,
        "gid": 0,
        "home": "/home/anthony",
        "shell": "/bin/bash"

}
```


Linux Academy.com
Learning Chef
Roles

Learning Chef

Roles

"A role is a way to define certain patterns and processes that exist across nodes in an organization as belonging to a single job function"

- Up until this point we have assigned recipes to be run for each node
  - Instead of updating run_lists for a node all we have to do is update a role on the server
  - Prevents us from having to manually touch all nodes that need the change
- A role is essentially a listing of recipes and attributes that are to be executed on a node
- Instead of assigning a run list for each node we assign the node a role
- A base role can be assigned inside of a roles run_list

Learning Chef

Roles: Role management with knife

- Knife role create role_name
- Chef-repo/roles/rolename.rb
- Knife role from file chef-repo/roles/rolename.rb
- Knife role list –w (views a list of roles on the chef server)
- Knife role delete role_name

Roles: Production

How can we spin up new nodes (like auto scaling) without having to manually modify run_lists in a node?

- Pass in cloud-init information about the node
- Create Virtual Machine Images after a node has been bootstrapped
  - Do not name the node but instead let it use the FQDN as the node name
  - Modify /etc/chef/first-boot.json to include the “role” of the node


Linux Academy.com
Learning Chef
Knife Plugins

Knife Plugins

Knife can be extended with the use of plugins

- Knife-flip: Manage environments for nodes
- Cloud platforms: azure, ec2, google, linode, openstack, rackspace, etc
- Custom plugins


Linux Academy.com
Learning Chef
Production Work Flow & Unattended Installs

Learning Chef

Work Flow:

- Nodes should be bootstrapped and managed from the workstation
- Nodes should be assigned roles and environments
- Attributes specific to roles/environments should be configured accordingly
- For auto scaling environments such as AWS “unattended” bootstrapping should be used

Work Flow: Unattended Bootstrapping

In order to have an unattended install the node when first created must meet the following criteria:

- Must be able to authenticate to the chef server
- Must be able to configure a run_list
- Must be able to access the chef-validator.pem so that it may create a new client.pem and identity on the chef server
- Must have a unique node name
- Client.rb file must have proper configuration so that it knows what server to communicate with
- Modify the file.json in /etc/chef to include the run_list for the node
  - This will ideally include a way to automate the running of chef-client

Linux Academy.com

AWS CSA

Certification

Example:

Simple code that when run will create an S3 bucket.

```
{

 "Description" : "This template will create an s3bucket",

 "Resources" : {
  "S3Bucket" : {
   "Type" : "AWS::S3::Bucket",
   "Properties" : {
    "AccessControl" : "PublicRead",
    "BucketName" : "gigitygumdrops"
   }
  }
 },

 "Outputs" : {
  "BucketName" : {
   "Value" : { "Ref" : "S3Bucket" },
   "Description" : "Created bucket for storing websites"
  }
 }
}
```

Version control your infrastructure!

Linux Academy.com

PRESENTATION TITLE

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean pellentesque facilisis felis. Suspendisse ipsum erat, facilisis at euismod vel, consequat vitae nulla. Curabitur fringilla, ligula a sagittis venenatis, odio velit ornare ligula, non semper ligula eros eget neque. Ut sagittis vulputate est, in mollis libero varius eget. Cras felis felis, feugiat a sem a, pharetra elementum arcu.

In odio lectus, placerat ut felis vitae, ullamcorper facilisis elit.

List Title:

- Linux List Item Number 1
- Linux List Item Number 2
- Linux List Item Number 3
- Linux List Item Number 4
- AWS List Item Number 1
- AWS List Item Number 2
- AWS List Item Number 3
- AWS List Item Number 4

Linux Academy.com

PRESENTATION TITLE

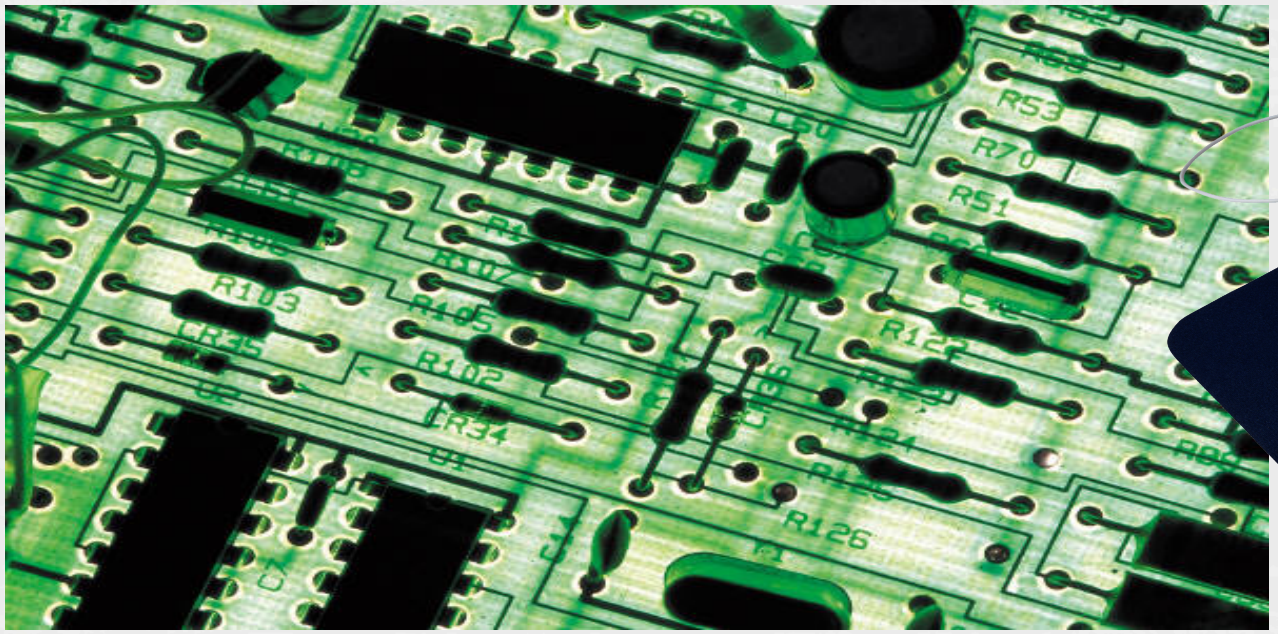


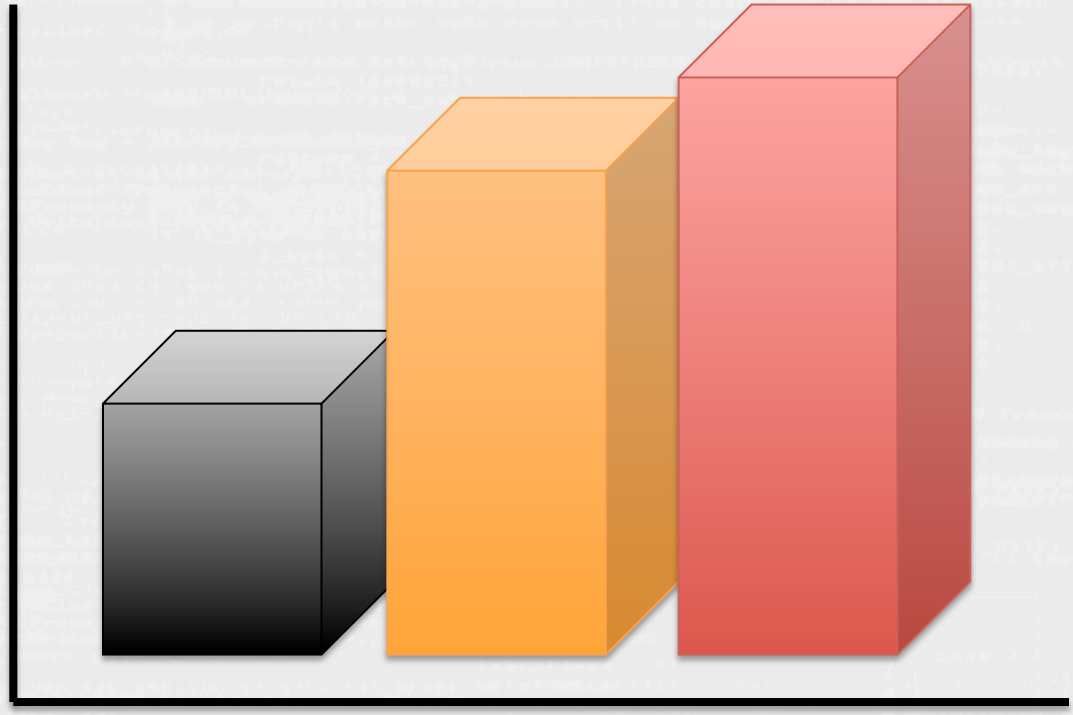
Image Title.

Content area for image description or caption. Even regular content simply associated with the image. Content area for image description or caption. Even regular content simply associated with the image.

Linux Academy.com

PRESENTATION TITLE

Chart Title:



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean pellentesque facilisis felis. Suspendisse ipsum erat, facilisis at euismod vel, [illegible]equat vitae nulla.

- Lorem
- Ipsum
- Dolor