Thoughts Improved

Thoughts, inklings of tech and some mild humor

Tech Recipe : Setup a RabbitMQ cluster on Ubuntu

> RabbitMQ (https://www.rabbitmq.com/) is an implementation of AMQP (http://en.wikipedia.org/wiki/Advanced_Message_Queuing_Protocol), the emerging standard for high performance enterprise messaging. The RabbitMQ server is a robust and scalable implementation of an AMQP broker.

I recently had to setup a RabbitMQ cluster from scratch on a brand new Ubuntu 14.04 server. This is easy. It turns out RabbitMQ can be installed via `apt-get` from the `extras` repo. I did quick check to see what version comes out of the box.

```
sudo apt-cache show rabbitmq-server
...
Version: 3.2.4-1
...
```

That's atleast 2 major versions behind the latest stable version! That's not good enough. I'd like to get my hands on the latest version of RabbitMQ. I'm going to be installing this and leaving the cluster running for a while. The RabbitMQ website has been very helpful here. It provides instructions on installing the latest stable version (http://www.rabbitmq.com/install-debian.html) as a `.deb` file. This turns out to be essentially a simple and pleasant experience.

I took that experience and realized almost all of it can be scripted. In this post I've codified the instructions as a bash script. I've attempted to create a script that can install RabbitMQ on any Ubuntu machine and then ring them up together in a cluster.

So let's get to it. The instructions are available in long form from the official RabbitMQ docs : http://www.rabbitmq.com/install-debian.html (http://www.rabbitmq.com/install-debian.html). Reading the docs is informative. But if you'd like to skim over those and quickly get RabbitMQ installed, it boils down to the following steps.

```
# Add the official rabbitmq source to your apt-get sources.list
sudo sh -c "echo 'deb http://www.rabbitmq.com/debian/ testing main' >

# Install the certificate
wget http://www.rabbitmq.com/rabbitmq-signing-key-public.asc
sudo apt-key add rabbitmq-signing-key-public.asc
rm rabbitmq-signing-key-public.asc

# Now install the latest rabbitmq
sudo apt-get update;
sudo apt-get install rabbitmq-server;
```

This should be familiar to anyone with Ubuntu/Debian experience. Now we can proceed to setting up the cluster.

Setup the RabbitMQ cluster

RabbitMQ clustering simply involves configuring multiple slaves to connect to a master. When RabbitMQ is installed it works as an independent server (and that's just fine, but clustering is recommended for production systems that want High Availability). You need to designate one server as a master, configure it, and then configure slaves to connect to this master. You can read a more thorough clustering guide on their site : https://www.rabbitmq.com/clustering.html (https://www.rabbitmq.com/clustering.html). In this post I've condensed it down to the essentials.

Setting up a cluster boils down to the following steps.

- Copy the erlang cookie from the master to the slaves
- Configure the master
- Configure the slaves to connect to the master
- Verify that the cluster is on

Copy the Erlang Cookie

RabbitMQ is built on Erlang (http://www.erlang.org/). Erlang systems which need to talk to each other must have the same magic cookie. This is a basic security mechanism to prevent unauthorized access to an Erlang system. An Erlang cookie is simply a hash stashed away in a file. It is in `/var/lib/rabbitmq/.erlang.cookie`.

Print the cookie on the master :

```
sudo cat /var/lib/rabbitmq/.erlang.cookie
```

Now login to **each** of your slaves and replace the erlang cookie. Make sure you copy it **exactly**, with no spaces or line endings.

```
sudo sh -c "echo 'COOKIE_FROM_MASTER' > /var/lib/rabbitmq/.erlang.coo
```

Setup the master

The cluster on the master needs to be reset, like so :

```
sudo rabbitmqctl stop_app;
sudo rabbitmqctl reset;
sudo rabbitmqctl start_app;
```

As taken from the RabbitMQ Manual (https://www.rabbitmq.com/man/rabbitmqctl.1.man.html), `rabbitmqctl reset` :

> Removes the node from any cluster it belongs to, removes all data from the management database, such as configured users and vhosts, and deletes all persistent messages.

Setup the slaves

Each slave machine needs to join the master to form a cluster. To do that the RabbitMQ slave needs to identify the master by a hostname (not IP), so you need to add the host name of the master to the `/etc/hosts` file **on each slave**. Here's how you would do it with `sed`. Make sure you adjust the IP and Hostname to your master server.

```
# You can `nano /etc/hosts`. The below does the same.
sudo sed -i "s/^$/MASTER_IP    MASTER_HOSTNAME\n/" /etc/hosts
```

Now login to each slave and connect it to the master as follows :

```
sudo rabbitmqctl stop_app;
sudo rabbitmqctl reset;
sudo rabbitmqctl join_cluster --ram rabbit@$MASTER_HOSTNAME;
sudo rabbitmqctl start_app;
```

Verify the cluster

And that's it. Let's verify if the cluster is connected. Run this on the master.

```
sudo rabbitmqctl cluster_status;
[{nodes,[{disc,[rabbit@master]},{ram,[rabbit@slave1]}]},
{running_nodes,[rabbit@master,rabbit@slave1]},
{cluster_name,<>},
{partitions,[]}]
```

You should see all the servers in your cluster in the `nodes` entry.

All-In-One Install Script

Ok, i know i said we can put this together in one file. And we can. But installation and setting up a cluster are two different operations. It would be more useful to have 2 standalone scripts; one for installation and another for setting up the cluster.

So, let's put together the first of these scripts. This is the **installation script**. It installs RabbitMQ on all servers asking for credentials along the way.

```
#!/bin/bash

function getHostname()
{
        local HOST=''

        while test -z "$HOST"
        do
          read -p "Enter the server's hostname [hostname or user@host
        done

        echo $HOST;
}

INSTALL_SCRIPT='
echo "Adding rabbitmq to /etc/apt/sources.list.d";
sudo sh -c "echo \"deb http://www.rabbitmq.com/debian/ testing main\"

echo "Installing the certificate";
wget http://www.rabbitmq.com/rabbitmq-signing-key-public.asc
sudo apt-key add rabbitmq-signing-key-public.asc
rm rabbitmq-signing-key-public.asc

echo "Installing rabbitmq-server";
sudo apt-get update;
sudo apt-get install rabbitmq-server;
';

echo "INSTALL RabbitMQ";
echo "================";

SERVER=$(getHostname);
while test "$SERVER" != "q"
do
        echo "ssh '$SERVER'";
```

```
        ssh -t $SERVER "bash -c '$INSTALL_SCRIPT'";
        SERVER=$(getHostname);
done

echo "Done";
```

The juice of it is in `INSTALL_SCRIPT`. The rest of the script just loops through user-provided hosts, installing RabbitMQ until the user finally quits (be pressing `q`). Copy the above script to a file, `install-rabbitmq.sh` and run it :

```
sudo chmod +x install-rabbitmq.sh;
./install-rabbitmq.sh
```

Cluster Setup Script

Here is the script to setup a cluster. This script asks for the master host, and subsequent slave hosts setting up the cluster as it goes. You'll need to make sure you can connect to the hosts from the machine your logged in to, and your remote user account can do `sudo`

```
#!/bin/bash
set -e

function getHostname()
{
        local HOST=''

        while test -z "$HOST"
        do
          read -p "$1 : " HOST
        done

        echo $HOST;
}

SETUP_MASTER_SCRIPT='
sudo rabbitmqctl stop_app;
sudo rabbitmqctl reset;
sudo rabbitmqctl start_app;
';

# Step 1 : Setup the Master. Get the erlang cookie

echo "Setup RabbitMQ Master";
echo "====================";

OUT=/tmp/master.out
MASTER_HOSTNAME=$(getHostname "Enter the master server's hostname [ho
echo "[$MASTER_HOSTNAME] Setting up master";
ssh -t $MASTER_HOSTNAME "bash -c '$SETUP_MASTER_SCRIPT sudo cat /var/
COOKIE=$(cat $OUT | tail -n1)
rm $OUT;
echo "Master's Erlang Cookie : '$COOKIE'"

MASTER_IP=$(getHostname "Enter the master server's IP as seen from th
```

```
# Step 2 : Setup the slaves

SETUP_SLAVE_SCRIPT="
sudo sed -i \"s/^$/$MASTER_IP    $MASTER_HOSTNAME\n/\" /etc/hosts
sudo bash -c \"echo -n '$COOKIE' > /var/lib/rabbitmq/.erlang.cookie\"
sudo rabbitmqctl stop_app;
sudo rabbitmqctl reset;
sudo rabbitmqctl join_cluster --ram rabbit@$MASTER_HOSTNAME;
sudo rabbitmqctl start_app;
sudo rabbitmqctl cluster_status;
";

echo "Setup RabbitMQ Slaves";
echo "=====================";

S="Enter the server's hostname [hostname or user@hostname] or 'q' to
SERVER=$(getHostname $S);
while test "$SERVER" != "q"
do
        echo "ssh '$SERVER'";
        ssh -t $SERVER "bash -c '$SETUP_SLAVE_SCRIPT'";
        SERVER=$(getHostname $S);
done

echo "Done";
```

Again, the juice of it is in `SETUP_MASTER_SCRIPT` and `SETUP_SLAVE_SCRIPT`. The script first sets up the master, then loops through user-provided slaves hooking them up to the cluster. Copy the above script to a file, `setup-rabbitmq-cluster.sh` and run it :

```
sudo chmod +x setup-rabbitmq-cluster.sh;
./setup-rabbitmq-cluster.sh
```

You can download this script from the Github Gist :
https://gist.github.com/adilbaig/1a12028b6eff3cd8c78a
(https://gist.github.com/adilbaig/1a12028b6eff3cd8c78a)

Summary

In this post we've learned how to setup a RabbitMQ cluster on Ubuntu, and have created scripts to automate the process. Did this work for you? Leave your comments below.

Advertisements

JANUARY 3, 2015 THOUGHTSIMPROVED RABBITMQ, UBUNTU

6 thoughts on “Tech Recipe : Setup a RabbitMQ cluster on Ubuntu”

1. Emmy says:
   This is very cool

REPLY ☐JULY 17, 2015 AT 5:58 PM

2. Pingback: RabbitMQ cluster on a single machine | alex.theedom
3. Victor says:

   I had to restart the service on slave-nodes right after updating the erlang cookie, for the next commands to work properly.

   sudo service rabbitmq-server restart

   REPLY ☐DECEMBER 24, 2016 AT 4:41 AM
4. Mike Dollar says:

   Question, I've noticed that most of the cluster setup tutorials and documentation refer to two slave-nodes (or more) and a master. Is that because a minimum of two slave-nodes is required?

   REPLY ☐FEBRUARY 24, 2017 AT 7:40 AM

   - thoughtsimproved says:

     No, RabbitMQ does not require a minimum, although other systems may. Systems that have an auto-failover mechanism generally require a consensus before switching over, and that works best in odd numbers. So, in case of a network partition you still end up with one master instead of 2. Hence you see systems recommending a cluster with an odd number of machines, starting at 3.

     In a master-slave setup, like RabbitMQ, this doesn't apply, but other factors like the capacity of a node, still do. You can even go with 2 servers for minimum service redundancy so long as they both don't fail. In practise I've found RabbitMQ to be solid, it's almost always a hardware or network failure that causes downtime.

     In most cases, I think 3 is an acceptable amount of redundancy for the cost of hardware.

     PS: Here's rabbitmq recommending a cluster of 3
     https://www.rabbitmq.com/production-checklist.html

     REPLY ☐FEBRUARY 24, 2017 AT 10:45 AM
5. Pingback: RabbitMQ Clustering – soul_departed

BLOG AT WORDPRESS.COM.