### **Initialize an empty release**

Run the following commands to initialize a new release:

|  |
| --- |
| mkdir ~/greeter-release cd ~/greeter-release bosh init-release |

After executing this command, the filesystem tree should look like this:

|  |
| --- |
| $ tree . ├── config │ └── blobs.yml │ └── final.yml  ├── jobs ├── packages └── src |

### **Create a router job**

Create a router job with:

|  |
| --- |
| cd ~/greeter-release bosh generate-job router |

After executing this command, the filesystem tree should look like this:

|  |
| --- |
| $ tree  .  ├── config  │ └── blobs.yml  │ └── final.yml  ├── jobs  │ └── router  │ ├── monit  │ ├── spec  │ └── templates  ├── packages  └── src |

### **Update the router spec**

Open the file jobs/router/spec in a text editor and add the following content to it:

|  |
| --- |
| ---  name: router  templates:  ctl: bin/ctl  config.yml.erb: config/config.yml  packages:  - greeter  - ruby  properties:  port:  description: "Port on which server is listening"  default: 8080  upstreams:  description: "List of upstreams to proxy requests"  default: [] |

### **Update the router Monit config**

Open the file jobs/router/monit in a text editor and add the following content to it:

|  |
| --- |
| check process router  with pidfile /var/vcap/sys/run/router/router.pid  start program "/var/vcap/jobs/router/bin/ctl start"  stop program "/var/vcap/jobs/router/bin/ctl stop"  group vcap |

### **Create the router startup script**

### Open the file jobs/router/templates/ctl in a text editor and add the following content to it:

|  |
| --- |
| #!/bin/bash  RUN\_DIR=/var/vcap/sys/run/router  LOG\_DIR=/var/vcap/sys/log/router  PIDFILE=$RUN\_DIR/router.pid  RUNAS=vcap  export PATH=/var/vcap/packages/ruby/bin:$PATH  export BUNDLE\_GEMFILE=/var/vcap/packages/greeter/Gemfile  export GEM\_HOME=/var/vcap/packages/greeter/gem\_home/ruby/2.3.0  function pid\_exists() {  ps -p $1 &> /dev/null  }  case $1 in  start)  mkdir -p $RUN\_DIR $LOG\_DIR  chown -R $RUNAS:$RUNAS $RUN\_DIR $LOG\_DIR  echo $$ > $PIDFILE  export CONFIG\_FILE=/var/vcap/jobs/router/config/config.yml  exec chpst -u $RUNAS:$RUNAS \  bundle exec ruby /var/vcap/packages/greeter/router.rb \  -p <%= p("port") %> \  -o 0.0.0.0 \  >>$LOG\_DIR/server.stdout.log 2>>$LOG\_DIR/server.stderr.log  ;;  stop)  PID=$(head -1 $PIDFILE)  if [ ! -z $PID ] && pid\_exists $PID; then  kill $PID  fi  while [ -e /proc/$PID ]; do sleep 0.1; done  rm -f $PIDFILE  ;;  \*)  echo "Usage: ctl {start|stop}" ;;  esac  exit 0 |

### 

### **Create the router config template**

Create a config template for the router by opening the file jobs/router/templates/config.yml.erb and adding the following lines to it:

|  |
| --- |
| ---  upstreams: <%= p('upstreams') %> |

### **Create the app job**

Generate a job:

|  |
| --- |
| cd ~/greeter-release bosh generate-job app |

### **Update the app spec**

Open the file jobs/app/spec in a text editor and add the following lines:

|  |
| --- |
| ---  name: app  templates:  ctl: bin/ctl  packages:  - greeter  - ruby  properties:  port:  description: "Port on which server is listening"  default: 8080 |

### **Update the app Monit config**

Open the file jobs/app/monit and add the following lines:

|  |
| --- |
| check process app  with pidfile /var/vcap/sys/run/app/app.pid  start program "/var/vcap/jobs/app/bin/ctl start"  stop program "/var/vcap/jobs/app/bin/ctl stop"  group vcap |

### **Create the app startup script**

Open the file jobs/app/templates/ctl in a text editor and add the following content to it:

|  |
| --- |
| #!/bin/bash  RUN\_DIR=/var/vcap/sys/run/app  LOG\_DIR=/var/vcap/sys/log/app  PIDFILE=$RUN\_DIR/app.pid  RUNAS=vcap  export PATH=/var/vcap/packages/ruby/bin:$PATH  export BUNDLE\_GEMFILE=/var/vcap/packages/greeter/Gemfile  export GEM\_HOME=/var/vcap/packages/greeter/gem\_home/ruby/2.3.0  function pid\_exists() {  ps -p $1 &> /dev/null  }  case $1 in  start)  mkdir -p $RUN\_DIR $LOG\_DIR  chown -R $RUNAS:$RUNAS $RUN\_DIR $LOG\_DIR  echo $$ > $PIDFILE  exec chpst -u $RUNAS:$RUNAS \  bundle exec ruby /var/vcap/packages/greeter/app.rb \  -p <%= p("port") %> \  -o 0.0.0.0 \  >>$LOG\_DIR/server.stdout.log 2>>$LOG\_DIR/server.stderr.log  ;;  stop)  PID=$(head -1 $PIDFILE)  if [ ! -z $PID ] && pid\_exists $PID; then  kill $PID  fi  while [ -e /proc/$PID ]; do sleep 0.1; done  rm -f $PIDFILE  ;;  \*)  echo "Usage: ctl {start|stop}" ;;  esac  exit 0 |

### **Create the Ruby package**

Generate the Ruby package:

|  |
| --- |
| cd ~/greeter-release  bosh generate-package ruby |

### **Create the Ruby spec**

Open the file packages/ruby/spec in a text editor and add the following lines to it:

|  |
| --- |
| ---  name: ruby  files:  - ruby/ruby-2.3.0.tar.gz  - ruby/bundler-1.11.2.gem |

### **Create the Ruby packaging script**

Edit the following file packages/ruby/packaging and add the following content to it:

|  |
| --- |
| set -e  tar xzf ruby/ruby-2.3.0.tar.gz  (  set -e  cd ruby-2.3.0  LDFLAGS="-Wl,-rpath -Wl,${BOSH\_INSTALL\_TARGET}" CFLAGS='-fPIC' ./configure --prefix=${BOSH\_INSTALL\_TARGET} --disable-install-doc --with-opt-dir=${BOSH\_INSTALL\_TARGET} --without-gmp  make  make install  )  ${BOSH\_INSTALL\_TARGET}/bin/gem install ruby/bundler-1.11.2.gem --local --no-ri --no-rdoc |

### **Download Ruby sources**

|  |
| --- |
| cd ~/greeter-release  curl https://cache.ruby-lang.org/pub/ruby/2.3/ruby-2.3.0.tar.gz --create-dirs -o blobs/ruby/ruby-2.3.0.tar.gz  curl https://rubygems.org/downloads/bundler-1.11.2.gem --create-dirs -o blobs/ruby/bundler-1.11.2.gem |

### **Create the greeter package**

Generate the greeter package with:

|  |
| --- |
| cd ~/greeter-release  bosh generate-package greeter |

### **Create the greeter spec**

Edit the file packages/greeter/spec and add the following content to it:

|  |
| --- |
| ---  name: greeter  dependencies:  - ruby  files:  - greeter/\*\*/\* |

### **Create the greeter packaging script**

Edit the file packages/greeter/packaging and add the following content to it:

|  |
| --- |
| set -e  cp -r greeter/\* ${BOSH\_INSTALL\_TARGET}  cd ${BOSH\_INSTALL\_TARGET}  find .  mkdir -p ${BOSH\_INSTALL\_TARGET}/gem\_home  /var/vcap/packages/ruby/bin/bundle install --local --no-prune --path ${BOSH\_INSTALL\_TARGET}/gem\_home |

## **Download greeter sources**

Download greeter sources with:

|  |
| --- |
| git clone https://github.com/Altoros/greeter.git ~/greeter-release/src/greeter |

**Configure your AWS account**

* Add a security group “app” that allows tcp connections on 8080 port. This group should belong to the same VPC as your bosh director instance.
* Create an Elastic IP for the router job.
* Try and do this in a new file called “app.tf” in your computer and then apply the changes with Terraform, or you can do it manually. Always prefer the more automated way!

### **Configure the blobstore**

Save the following file as config/final.yml:

|  |
| --- |
| ---  final\_name: greeter-release  blobstore:  provider: local  options:  blobstore\_path: /tmp/bosh-blobstore |

Save the following file as config/blobs.yml:

|  |
| --- |
| ruby/bundler-1.11.2.gem: {}  ruby/ruby-2.3.0.tar.gz: {} |

### **Create the release**

### Create a release by running:

|  |
| --- |
| cd ~/greeter-release  bosh create-release --force  bosh upload-release |

### 

### **Upload stemcell**

If you haven't done this before, upload a stemcell with:

|  |
| --- |
| bosh upload-stemcell https://bosh.io/d/stemcells/bosh-aws-xen-hvm-ubuntu-trusty-go\_agent |

### **Generate the deployment manifest**

From here until the end of this document, wherever you see “REPLACE\_WITH…” replace that text with the appropriate value. (hint - look to the AWS Console)

Save the following as ~/deployment/greeter.yml:

|  |
| --- |
| ---  name: greeter-release  releases:  - name: greeter-release  version: latest  compilation:  workers: 4  network: private  cloud\_properties:  instance\_type: m4.large  availability\_zone: REPLACE\_WITH\_AZ  update:  canaries: 1  canary\_watch\_time: 30000  update\_watch\_time: 30000  max\_in\_flight: 1  max\_errors: 1  networks:  - name: private  type: manual  subnets:  - range: 10.0.0.0/24  gateway: 10.0.0.1  dns:  - 8.8.8.8  - 8.8.4.4  reserved:  - 10.0.0.1 - 10.0.0.6  static:  - 10.0.0.7  - 10.0.0.8  cloud\_properties:  subnet: REPLACE\_WITH\_SUBNET\_ID  security\_groups: [app, bosh]  - name: public  type: vip  resource\_pools:  - name: infrastructure  size: 4  stemcell:  name: bosh-aws-xen-hvm-ubuntu-trusty-go\_agent  version: latest  network: private  cloud\_properties:  instance\_type: t2.small  availability\_zone: REPLACE\_WITH\_AZ  jobs:  - name: app  templates:  - name: app  instances: 1  resource\_pool: infrastructure  networks:  - name: private  static\_ips:  - 10.0.0.7  properties: {}  - name: router  templates:  - name: router  instances: 1  resource\_pool: infrastructure  networks:  - name: private  static\_ips:  - 10.0.0.8  default: [dns, gateway]  - name: public  static\_ips:  - REPLACE\_WITH\_ELASTIC\_IP  properties:  upstreams:  - 10.0.0.7:8080 |

### **Deploy!**

Finally, everything is ready for deployment:

|  |
| --- |
| bosh -d greeter-release -n deploy ~/deployment/greeter.yml |

Let's check if everything has been deployed as intended:

|  |
| --- |
| curl "http://REPLACE\_WITH\_ELASTIC\_IP:8080" |

To list all your VMs, execute this command:

|  |
| --- |
| bosh vms |

### **Scale your deployment**

In your ~/deployment/greeter.yml manifest:

1. Add the 10.0.0.9 IP to the private static pool /networks/name:private/subnets/gateway=10.0.0.1/static/-
2. Add the 10.0.0.9 IP from the private static pool to /jobs/name:app/networks/name:private/static\_ips/-
3. Increase the number of instances in /jobs/name:app/instances by 1
4. Append 10.0.0.9:8080 to the /jobs/name:router/properties/upstreams/- array

The best way to do this is to create an [opfile](https://bosh.io/docs/cli-ops-files.html). You can create that file or update the manifest manually. Note, that when identifying properties path we use the same syntax, as it is used in opfiles, so them can be copied directly.

Deploy once again:

|  |
| --- |
| bosh -d greeter-release -n deploy -o ~/deployment/greeter-opfile.yml ~/deployment/greeter.yml |

And if you curl the router multiple times, you should see greetings from different upstreams:

|  |
| --- |
| curl "http://$REPLACE\_WITH\_ELASTIC\_IP:8080" |

### 

### **Experimenting with the deployment**

## Exercise 1 (easy)

Change port that router listens on from 8080 to 8081. Don’t forget to make corresponding change in the app security group settings.

## Exercise 2 (easy)

Change port that each app listens on from 8080 to 8081. Don’t forget to update router properties as well.

## Exercise 3 (easy)

Assign public IP address to one of the app VMs. Make router to connect to this app using public IP address.

## Exercise 4 (medium)

Change all App VMs flavor to t2.medium, and Router VM to t2.micro. (Tip: you will have to create additional resource pull in the deployment manifest. See [Resource Pools block definition](http://bosh.io/docs/deployment-manifest.html#resource-pools) from the official documentation for reference).

## Exercise 5 (medium)

Assign a persistent disk to each of the App VMs. (Tip: See [Official Documentation](http://bosh.io/docs/deployment-manifest.html#jobs) for the description of the available job properties)

After this change is done, ssh to the App VM (using ‘bosh ssh’ command) and execute ‘df -h’ command. Find the folder to which you persistent disk is mounted.

## Exercise 6 (hard)

Add new property to the Router job that should be called “message”. Instead of printing “Router says ...”, router should now print the value from the message property and then message from the App. This will require modifying router.rb file from src/router folder. Router config file should also be modified to include new property. Property definition should be also added to the router spec file. After all this is done - recreate and reupload the release. Having new release ready you have to include new property in the deployment manifest and redeploy everything.