Handout: Rasa Workshop Day 4

Welcome to day 4 of the Rasa workshop! In this session, we'll deploy our assistants to a live server, connect to Telegram, and start improving the assistant using Rasa X.

Get the slides for today's session here.

Accessing your VM

The Rasa team has already set up a server for you to use for today's session (and pre-installed Rasa X & Rasa Open Source). To log in to your Rasa X instance, check your DMs in Slack. You should see a message from the Rasa team with the URL, password, and a link to the backend settings for your machine in the Google Cloud Platform console.

🤔Don't see a Slack message from the Rasa team? We might not have collected your info in our form. Drop us a message in the #workshop-help channel and we'll get you set up!

How we set up the VMs for the workshop...

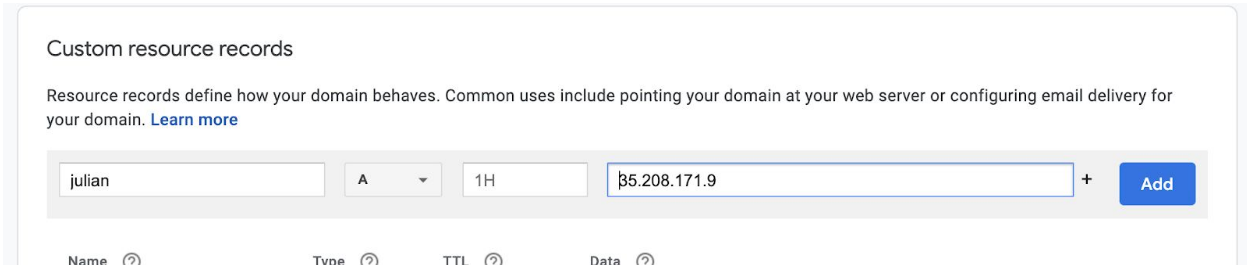
Note: These steps have already been completed for you ahead of the workshop - there's nothing to follow along with for this section. Feel free to sit back and watch during the demo (and ask questions)! After the workshop, you can use these instructions as a reference to set up a machine on your own hosting account.

1. Create the VM
    a. If using GCP, go to Compute Engine>VM instances
    b. Create a new VM with these specs:
        i. Assign a name for the machine

ii. Under Machine Configuration, select n1-standard-2 (2 vCPU, 7.5 GB memory)
iii. In Boot disk settings, select:
    1. Operating system: Ubuntu
    2. Version: Ubuntu 18.04 LTS
    3. Disk size: 50 GB
    4. You can find the minimum system requirements for using the one-line deploy script in the docs
iv. Check the boxes to allow HTTP and HTTPS traffic
v. Click the 'Management, security, disks, networking, sole tenancy' link to expand the settings
    1. On the Networking tab, click into the Network Interfaces settings and reserve a static external IP

vi. Click the button to create the instance

2. Assign a domain name
    a. Copy the external IP from the new instance and go to the DNS setting page for your domain
    b. Create a new A record and point it to your external IP address



Custom resource records

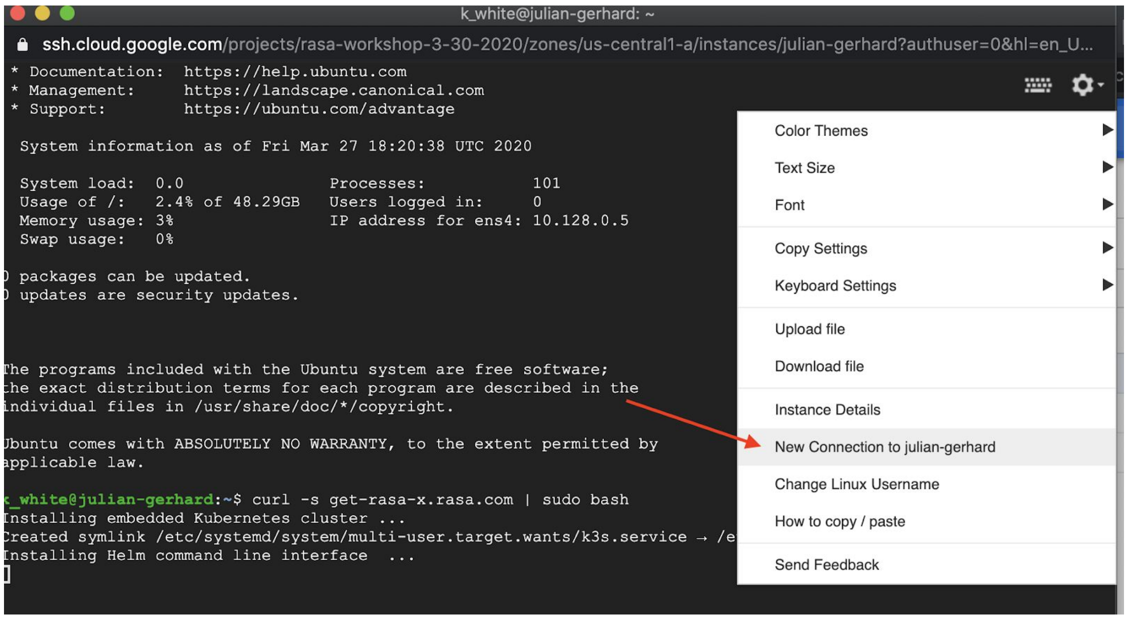
Resource records define how your domain behaves. Common uses include pointing your domain at your web server or configuring email delivery for your domain. Learn more

Name	Type	TTL	Data
julian	A	1H	35.208.171.9

3. Install Rasa X
    a. Go back to the VM Instances dashboard in GCP.
    b. Click the SSH button to connect to the VM instance.
    c. Run the one-line deploy script

```
curl -s get-rasa-x.rasa.com | sudo bash
```

d. As it's installing, open a new connection to the server in another window


k_white@julian-gerhard: ~
ssh.cloud.google.com/projects/rasa-workshop-3-30-2020/zones/us-central1-a/instances/julian-gerhard?authuser=0&hl=en_U...
Color Themes
Text Size
Font
Copy Settings
Keyboard Settings
Upload file
Download file
Instance Details
New Connection to julian-gerhard
Change Linux Username
How to copy / paste
Send Feedback

```
 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Fri Mar 27 18:20:38 UTC 2020

  System load:  0.0               Processes:           101
  Usage of /:   2.4% of 48.29GB   Users logged in:     0
  Memory usage: 3%                IP address for ens4: 10.128.0.5
  Swap usage:   0%

) packages can be updated.
) updates are security updates.


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

_white@julian-gerhard:~$ curl -s get-rasa-x.rasa.com | sudo bash
Installing embedded Kubernetes cluster ...
Created symlink /etc/systemd/system/multi-user.target.wants/k3s.service → /e
Installing Helm command line interface  ...
```

e. Open your ~/.bashrc file

```
nano ~/.bashrc
```

f. Paste the export statement below into the file, and save it. After you save it, you can go ahead and close this window. Note: you need to disconnect and reconnect to the server for the change to the .bashrc file to take effect (important for later steps)

```
export KUBECONFIG=/etc/rancher/k3s/k3s.yaml
```

g. In the terminal window where you're installing Rasa X, be sure to copy and save the Rasa X password that prints to the screen

4. Install the SSL

a. Open a connection to the server and install the CustomResourceDefinitions and cert-manager by running this command on your server:

```
kubectl apply --validate=false -f
https://github.com/jetstack/cert-manager/releases/download
/v0.14.0/cert-manager.yaml
```

b. Create a new file on your server and name it letsencrypt-issuer-production.yml. Add the following to the contents, replacing the placeholder with your email address:

```
apiVersion: cert-manager.io/v1alpha2
kind: ClusterIssuer
metadata:
  name: letsencrypt-prod
spec:
  acme:
   # The ACME server URL
    server: https://acme-v02.api.letsencrypt.org/directory
    # Email address used for ACME registration
    email: your.email@gmail.com
    # Name of a secret used to store the ACME account private key
    privateKeySecretRef:
        name: letsencrypt-prod
    # Enable the HTTP-01 challenge provider
    solvers:
    - http01:
        ingress:
            class: traefik
```

c. Save the file, and then apply it to your cluster with this command:
   `sudo kubectl apply -f letsencrypt-issuer-production.yml`
d. Now, create another file and name it cert-request.yml. Add the following to the contents, replacing the placeholder for secret name and domain name.

```
apiVersion: cert-manager.io/v1alpha2
kind: Certificate
metadata:
  name: letsencrypt-prod
  namespace: rasa
spec:
  secretName: <name for your secret, e.g. rasa-tls>
  issuerRef:
    name: letsencrypt-prod
    kind: ClusterIssuer
  commonName: <domain name for your Rasa X server, e.g. myassistant.com>
  dnsNames:
  - <domain name for your Rasa X server>
```

e. Save the file and apply it to your cluster:
   `sudo kubectl apply -f cert-request.yml`
f. Finally, update the ingress in your Helm chart settings. Create a file called values.yml and add the following, replacing the placeholders for secret name and domain:

```
ingress:
  annotations:
    ingress.annotations.certmanager.k8s.io/cluster-issuer: letsencrypt-prod
    kubernetes.io/ingress.class: traefik
  hosts:
    - host: <your domain name>
      paths:
      - /
  tls:
   - secretName: <secret name you specified in cert-request.yml>
```

```
    hosts:
      - <your domain name>
```

g. Run the helm upgrade command so the new chart settings can take effect:

```
helm upgrade \
    --values values.yml \
    --namespace rasa \
    --reuse-values \
    rasa rasa-x/rasa-x
```

h. Navigate to https://yourdomain.com to verify the installation 🎉

Make sure your GitHub Repo is up to date

Before we finish deployment, make sure you've pulled in the latest changes from the upstream repository (the one you forked from).

1. Fetch the latest branches and commits from upstream (if you get an error saying no upstream found, run `git remote add upstream https://github.com/RasaHQ/financial-demo.git` and then re-run the fetch command)
    a. `git fetch upstream`
2. Checkout your local master branch
    a. `git checkout master`
3. Merge changes from upstream/master into your local master branch
    a. `git merge upstream/master`
4. Push the updates to your remote repository
    a. `git push origin master`

Build the Action Server Image

Your assistant's custom action code runs in a Docker container, which is defined by a Docker image. To create a Docker image, we'll list the container's requirements in a Dockerfile, build the image, and then push it to a container registry (DockerHub).

Note that the repo for our assistant already includes a Dockerfile in the root of the project.

1. Build and push the image to DockerHub
    a. Run this command in the same directory as your Dockerfile:
        i. `docker build . -t`

`<account_username>/<repository_name>:<custom_image_tag>`

   b. Push the image to the container registry
      i. Log in from the terminal (or log in from the Docker desktop app)
         1. `docker login --username <account_username> --password <account_password>`
      ii. Push the image to DockerHub
         1. `docker push <account_username>/<repository_name>:<custom_image_tag>`
   c. Next: Reference the image (and tag) in your deployment

Haven't installed Docker? Skip this section and see the Note in the Finish Deployment section.

Get Telegram API Credentials

Before you can connect your assistant to Telegram, you need to register your bot with Telegram and get an API token and username.

Go to Bot Father on Telegram and type /newbot

Follow the prompts to register your bot and get your API token and username.

💡 Tip: When registering your bot, the **name** is displayed in contact details. The **username** is a short name used in mentions and must end in bot.

Finish Deployment

Additional settings can be configured in the deployment using environment variables.

1. Set 5 environment variables. Be sure to replace the placeholders in the Telegram credentials:

   a. `export ACTION_SERVER_IMAGE="karenwhite/dayfourdemo"`

   b. `export ACTION_SERVER_TAG="version-0.3"`

   c. `export ENABLE_DUCKLING="True"`

   d. `export RASA_VERSION="1.10.0"`

e.
```
export
ADDITIONAL_CHANNEL_CREDENTIALS="telegram.access_token='<yo
ur
token>',telegram.verify='username',telegram.webhook_url='h
ttps://<yourdomain>.com/webhooks/telegram/webhook/'"
```

2. Run the one-line deploy script to apply the environment variables
```
curl -s get-rasa-x.rasa.com | sudo -E bash
```

Note: Haven't installed Docker? No worries! Reference the Docker image and tag as written in instructions 1a. and 1b above (it's publicly hosted).

Kubectl Reference

kubectl get pods	Lists the pods running in the k8s cluster, along with status
kubectl logs <service name> example: **kubectl logs rasa-app-55b866cd99-k6nqp**	List the logs for a single pod, for example, the action server