import os

import itertools

import subprocess

import math

SERVICE\_NAME = 'uua-enricher'

CPU\_NUM = 3

HEAP\_SIZE = 6 \* GB

RAM\_SIZE = 8 \* GB

DISK\_SIZE = 3 \* GB

class Profile(Struct):

package = Default(String, SERVICE\_NAME)

cmdline\_flags = Default(String, '')

log\_level = Default(String, 'INFO')

instances = Default(Integer, 10)

kafka\_bootstrap\_servers = Default(String, '/s/kafka/bluebird-1:kafka-tls')

resources = Resources(

cpu = CPU\_NUM,

ram = RAM\_SIZE,

disk = DISK\_SIZE

)

install = Packer.install(

name = '{{profile.package}}',

version = Workflows.package\_version()

)

async\_profiler\_install = Packer.install(

name = 'async-profiler',

role = 'csl-perf',

version = 'latest'

)

setup\_jaas\_config = Process(

name = 'setup\_jaas\_config',

cmdline = '''

mkdir -p jaas\_config

echo "KafkaClient {

com.sun.security.auth.module.Krb5LoginModule required

principal=\\"discode@TWITTER.BIZ\\"

useKeyTab=true

storeKey=true

keyTab=\\"/var/lib/tss/keys/fluffy/keytabs/client/discode.keytab\\"

doNotPrompt=true;

};" >> jaas\_config/jaas.conf

'''

)

main = JVMProcess(

name = SERVICE\_NAME,

jvm = Java11(

heap = HEAP\_SIZE,

extra\_jvm\_flags =

'-Djava.net.preferIPv4Stack=true'

' -XX:+UseNUMA'

' -XX:+AggressiveOpts'

' -XX:+PerfDisableSharedMem' # http://www.evanjones.ca/jvm-mmap-pause.html

' -Dlog\_level={{profile.log\_level}}'

' -Dlog.access.output=access.log'

' -Dlog.service.output={{name}}.log'

' -Djava.security.auth.login.config=jaas\_config/jaas.conf'

),

arguments =

'-jar {{name}}-bin.jar'

' -admin.port=:{{thermos.ports[health]}}'

' -kafka.bootstrap.servers={{profile.kafka\_bootstrap\_servers}}'

' -kafka.application.id={{name}}.{{environment}}'

' -kafka.application.num.instances={{instances}}' # Used for static partitioning

' -kafka.application.server={{mesos.instance}}.{{name}}.{{environment}}.{{role}}.service.{{cluster}}.twitter.com:80'

' -com.twitter.finatra.kafkastreams.config.principal={{role}}'

' -thrift.client.id={{name}}.{{environment}}'

' -service.identifier="{{role}}:{{name}}:{{environment}}:{{cluster}}"'

' -local.cache.ttl.seconds=86400'

' -local.cache.max.size=400000000'

' {{profile.cmdline\_flags}}',

resources = resources

)

stats = Stats(

library = 'metrics',

port = 'admin'

)

job\_template = Service(

name = SERVICE\_NAME,

role = 'discode',

instances = '{{profile.instances}}',

contact = 'disco-data-eng@twitter.com',

constraints = {'rack': 'limit:1', 'host': 'limit:1'},

announce = Announcer(

primary\_port = 'health',

portmap = {'aurora': 'health', 'admin': 'health'}

),

task = Task(

resources = resources,

name = SERVICE\_NAME,

processes = [async\_profiler\_install, install, setup\_jaas\_config, main, stats],

constraints = order(async\_profiler\_install, install, setup\_jaas\_config, main)

),

health\_check\_config = HealthCheckConfig(

initial\_interval\_secs = 100,

interval\_secs = 60,

timeout\_secs = 60,

max\_consecutive\_failures = 4

),

update\_config = UpdateConfig(

batch\_size = 50,

watch\_secs = 90,

max\_per\_shard\_failures = 3,

max\_total\_failures = 0,

rollback\_on\_failure = False

)

)

PRODUCTION = Profile(

)

STAGING = Profile(

package = SERVICE\_NAME+'-staging',

cmdline\_flags = '',

kafka\_bootstrap\_servers = '/s/kafka/custdevel:kafka-tls'

)

DEVEL = STAGING(

log\_level = 'DEBUG',

)

prod\_job = job\_template(

tier = 'preferred',

environment = 'prod',

).bind(profile = PRODUCTION)

staging\_job = job\_template(

environment = 'staging'

).bind(profile = STAGING)

devel\_job = job\_template(

environment = 'devel'

).bind(profile = DEVEL)

jobs = []

for cluster in ['atla', 'pdxa']:

jobs.append(prod\_job(cluster = cluster))

jobs.append(staging\_job(cluster = cluster))

jobs.append(devel\_job(cluster = cluster))