云存储应用技术

实验五：消息队列

丁烨
dingye@dgut.edu.cn
网络空间安全学院

2019-12-05

東莞理工學院
DONGGUAN UNIVERSITY OF TECHNOLOGY

目录

搭建实验环境

RabbitMQ

搭建实验环境

准备 UNIX 实验环境

- 启用 VirtualBox 虚拟网卡
- 从开始菜单或桌面 Docker QuickStart Terminal 图标启动 Docker Toolbox
- 启动 Docker Container：`docker run -dti -p 22 ubuntu-sshd:18.04`
- 查询 SSH 端口：`docker ps -a`
- 通过 SSH 连接到 Container：`ssh -p <22> admin@192.168.99.100`
- 密码：`screencast`

目录

搭建实验环境

RabbitMQ

RabbitMQ 简介

- RabbitMQ
- https://www.rabbitmq.com/

- RabbitMQ 是实现了高级消息队列协议（AMQP）的一个开源消息队列系统
- RabbitMQ 服务器端是用 Erlang 语言编写的
- 大部分主流编程语言均有 RabbitMQ 的客户端，包括：Python、Java、Ruby、PHP、C#、JavaScript、Go、Elixir、Objective-C、Swift 等

- RabbitMQ 是目前市场占有率最大的消息队列系统

安装 RabbitMQ

- Ubuntu 18.04
- 安装 RabbitMQ：
- `sudo apt install rabbitmq-server`

- 启动 RabbitMQ（服务模式）：
- `sudo service rabbitmq-server start`
- 或（用户模式）：
- `rabbitmq-server`

- 如果需要将 RabbitMQ 在后台运行，可以使用 Screen

安装 RabbitMQ

- macOS Catalina
- 安装 RabbitMQ：
- `brew install rabbitmq`

- 启动 RabbitMQ：
- `rabbitmq-server`

- 如果需要 RabbitMQ 在后台运行，可以使用 screen

安装 RabbitMQ

- Windows 10（不推荐）
- 安装 RabbitMQ：
- https://github.com/rabbitmq/rabbitmq-server/releases

- 安装完毕后，RabbitMQ 的服务会在后台自动运行，如果需要关闭或重新启动，可以使用 Windows 的“系统服务”功能管理

- 大部分消息队列系统都不支持 Windows，RabbitMQ 是比较少见支持 Windows 平台的消息队列系统，但 Windows 下性能较差，不推荐使用

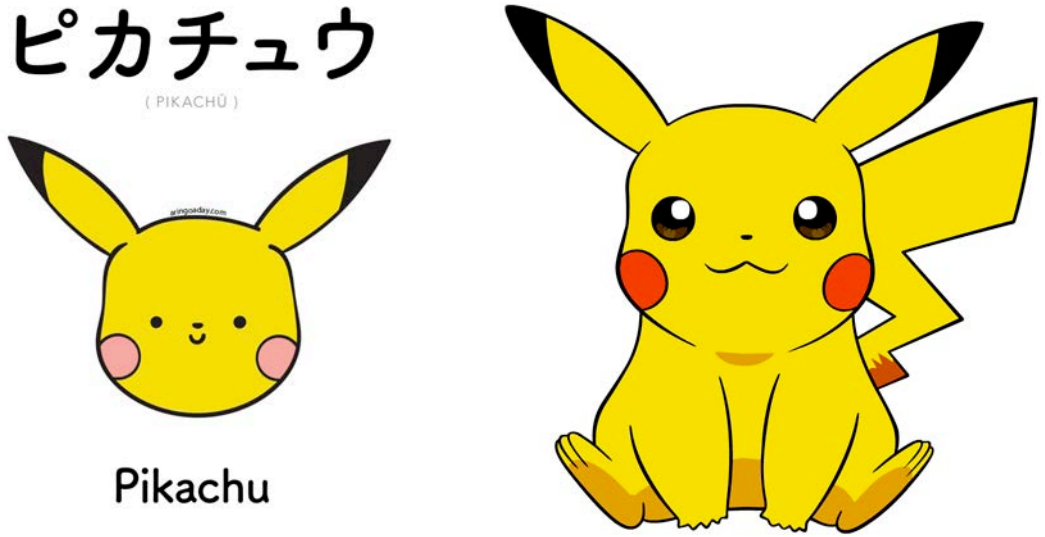
- RabbitMQ 客户端
- 大部分主流编程语言均有 RabbitMQ 的客户端，包括：Python、Java、Ruby、PHP、C#、JavaScript、Go、Elixir、Objective-C、Swift 等
- Python 客户端（Pika）：https://github.com/pika/pika
- Ruby 客户端（Bunny）：http://rubybunny.info/
- Go 客户端：http://godoc.org/github.com/streadway/amqp
- JavaScript 客户端：http://www.squaremobius.net/amqp.node/

- 本课程采用 Python 作为例子

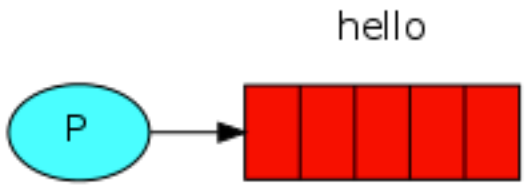
使用 RabbitMQ

- Pika（RabbitMQ Python 客户端）
- https://github.com/pika/pika

- `sudo apt install python3-pip`
- `pip3 install --user -U pika`

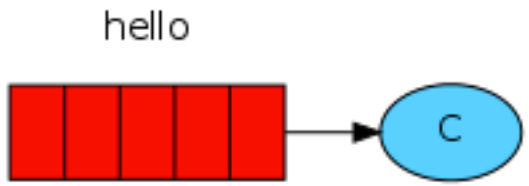


使用 RabbitMQ



- 发送端（Producer / Publisher）
- `import pika`
- `connection = pika.BlockingConnection(pika.ConnectionParameters(host='localhost'))`
- `channel = connection.channel()`
- `channel.queue_declare(queue='hello')`
- `channel.basic_publish(exchange='', routing_key='hello', body='Hello World!')`
- `print(" [x] Sent 'Hello World!'")`
- `connection.close()`

使用 RabbitMQ



- 接收端（Consumer / Subscriber）

```
import pika
connection = pika.BlockingConnection(pika.ConnectionParameters(host='localhost'))
channel = connection.channel()
channel.queue_declare(queue='hello')
print(' [*] Waiting for messages. To exit press CTRL+C')
def callback(ch, method, properties, body):
    print(" [x] Received %r" % body)
channel.basic_consume(queue='hello', auto_ack=True, on_message_callback=callback)
channel.start_consuming()
```

使用 RabbitMQ

```
~ » python3 producer.py
 [x] Sent 'Hello World!'
```

```
~ » python3 subscriber.py
 [*] Waiting for messages. To exit press CTRL+C
 [x] Received b'Hello World!'
```

- RabbitMQ 教程
- https://www.rabbitmq.com/getstarted.html

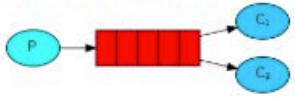
1 "Hello World!"

The simplest thing that does *something*

- Python
- Java
- Ruby
- PHP
- C#
- JavaScript
- Go
- Elixir
- Objective-C
- Swift
- Spring AMQP

2 Work queues

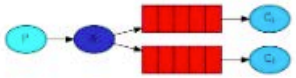
Distributing tasks among workers (the competing consumers pattern)



- Python
- Java
- Ruby
- PHP
- C#
- JavaScript
- Go
- Elixir
- Objective-C
- Swift
- Spring AMQP

3 Publish/Subscribe

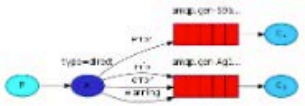
Sending messages to many consumers at once



- Python
- Java
- Ruby
- PHP
- C#
- JavaScript
- Go
- Elixir
- Objective-C
- Swift
- Spring AMQP

4 Routing

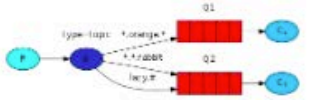
Receiving messages selectively



- Python
- Java
- Ruby
- PHP
- C#
- JavaScript
- Go
- Elixir
- Objective-C
- Swift
- Spring AMQP

5 Topics

Receiving messages based on a pattern (topics)



- Python
- Java
- Ruby
- PHP
- C#
- JavaScript
- Go
- Elixir
- Objective-C
- Swift
- Spring AMQP

- Understanding RabbitMQ Webinar
- https://content.pivotal.io/webinars/dec-12-understand-rabbitmq-for-developers-and-operators-webinar

作业

1) 在实验室的 Docker 虚拟机环境或任意 UNIX 环境下安装 RabbitMQ
2) 使用任意两套客户端（如 Pika，可使用 Screen）同时连接 RabbitMQ
3) 其中一个客户端作为发送者发送至少一条包含学号的消息，例如：Pika 20174110001
4) 另一个客户端作为接收者实时接收消息
5) 将全部的安装、部署、操作流程记录下来作为“实验过程”
6) 将两套客户端运行完毕的截图分别截取下来作为“实验结果”

```
~ » python3 producer.py
 [x] Sent 'Pika 20174110001'
~ »
```

```
~ » python3 subscriber.py
 [*] Waiting for messages. To exit press CTRL+C
 [x] Received b'Hello World!'
 [x] Received b'Pika 20174110001!'
```

作业

- 提交作业：
- 下载并完成本实验课对应实验报告，重命名为“<student-id>.docx”，例如：20174110001.docx
- 发送实验报告到：dingye@dgut.edu.cn
- 标题请注明：046039 Assignment 5
- 正文请注明姓名和学号
- 不要发送其他任何文件，只需要发送一个 .docx 文件即可

Thanks!