

前端系统部署

- 1 概述
- 2 Nginx服务器安装与配置
 - 2.1 nginx服务器安装
 - 2.2 前置条件
 - 2.3 编译
 - 2.4 添加到linux服务
 - 2.5 启动
 - 2.6 增加dns
 - 2.7 nginx配置
 - 2.8 启动自动收集日志的脚本
- 3 前端代码打包部署
 - 3.1 安装jenkins
 - 3.2 修改前端代码的发布文件
 - 3.3 配置jenkins任务

概述

第三代结算系统前端采用angular框架开发，代码编译打包后为一系列的js，html，css等文件，独立部署在web服务器的部署目录即可。前端采用的web服务器为nginx（1.17版本），操作系统为redhat7.4

Nginx服务器安装与配置

nginx服务器安装

nginx安装有两种方式，第一种yum源直接安装，需要连接外网。第二种，下载指定版本的nginx源码包，编译安装。一般服务器环境不能连外网，所以这里我们采用第二种方式。编译之后的可执行文件是可以拷贝到其他linux服务器运行的。当nginx需要版本更新，漏洞升级时，我们需要下载新的代码，以及代码依赖的其他模块代码，进行nginx的升级操作。

编译与安装需要用root用户执行。安装完成后，可以使用普通用户启动。

前置条件

准备一台安装了gcc环境的服务器。（注意：运行通过模板复制出来的服务器不带gcc）

准备好nginx代码包，以及相关的代码包。最小约束如下：

nginx-1.17.8.tar.gz

nginx-sticky-module-ng

openssl-1.1.0j.tar.gz

pcre-8.43.tar.gz

以上代码可以在10.1.19.205服务器/data/software/nginx-compile的目录查看

编译

将以上压缩包通过tar命令解压缩，不要改变目录位置。进入nginx源码文件夹。执行下面的命令：

```
./configure \  
--prefix=/data/nginx-server \  
--pid-path=/data/nginx-server/nginx.pid \  
--with-pcre=../pcre-8.43 \  
--with-http_gzip_static_module \  
--with-openssl=../openssl-1.1.0j \  
--with-http_stub_status_module \  
--with-http_ssl_module \  
--add-module=../nginx-sticky-module-ng
```

```
make && make install
```

上面的configuration命令中的prefix项，即为nginx编译之后的安装目录。

编译后的/data/nginx-server目录就可以拷贝到其他任意机器执行了。

添加到linux服务

编译成功之后的nginx-server包拷贝到其他服务器后，需要将nginx做成开启自启动，以及linux系统服务项。

配置nginx可以启动80端口，将nginx脚本拷贝到系统目录，并做成开机启动项

```
setcap cap_net_bind_service=+ep /data/nginx-server/sbin/nginx  
cp /data/software/scripts/nginx /etc/init.d/  
chmod 777 /etc/init.d/nginx  
chkconfig nginx on
```

上面需要用sudo，或者root

启动

```
service nginx start  
service nginx stop  
service nginx restart
```

增加dns

修改sudo vi /etc/resolv.conf 文件

添加

```
nameserver 10.1.18.237
```

nginx配置

nginx配置文件位于/data/nginx-server/conf/nginx.conf

```
#user  nobody;
worker_processes  1;

error_log  logs/error.log;

events {
    worker_connections  1024;
}

http {
    include      mime.types;
    default_type  application/octet-stream;

    log_format  main  '$remote_addr - $remote_user [$time_local]
"$request" '
                    '$status $body_bytes_sent "$http_referer" '
                    '"$http_user_agent" "$http_x_forwarded_for"
"$upstream_addr" "$cookie_userId" "$http_userId" "$http_jwt" '
                    '"$request_time" "$upstream_response_time"
"$request_body"';

    access_log  logs/access.log  main;

    sendfile      on;
    #tcp_nopush    on;

    keepalive_timeout  65;

    gzip on;
    gzip_buffers 32 4k;
    gzip_comp_level 6;
    gzip_min_length 1000k;
    gzip_types text/plain application/javascript application/x-javascript
text/css application/xml text/javascript application/x-httpd-php image
/jpeg image/gif image/png application/json;
    gzip_vary on;

    client_max_body_size 800m;
    client_header_timeout 10m;
    client_body_timeout 10m;
```

```

proxy_connect_timeout    60s;
proxy_read_timeout       10m;
proxy_send_timeout       10m;

add_header X-Frame-Options SAMEORIGIN;
add_header X-Content-Type-Options nosniff;

    map $http_upgrade $connection_upgrade {
    default upgrade;
    '' close;
    }

server {
    listen            80;
    server_name       localhost;

    #charset koi8-r;

    #access_log   logs/host.access.log  main;

    location / {
        root      html;
        index     index.html index.htm;
    }

}

server {
    listen            443 ssl;
    server_name       10.1.17.140;

    ssl_certificate    /data/cert-internal/pax.crt;

    ssl_certificate_key /data/cert-internal/pax_nopass.key;

    ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
    ssl_ecdh_curve X25519:P-256:P-384;
    ssl_prefer_server_ciphers on;
    ssl_session_cache shared:SSL:50m;
    ssl_session_timeout 5m;
    ssl_session_tickets on;
    add_header Strict-Transport-Security max-age=15768000;

    keepalive_timeout 65;

    location /opra-api-ws/ {
        proxy_set_header Host opragateway-ca-opra-test.apps.ocp.
acca;

        proxy_pass http://opragateway-ca-opra-dev.apps.ocp.acca/;

```

```

        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
        proxy_read_timeout 1d;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For
$proxy_add_x_forwarded_for;

    }
    location /opra-api/ {
        proxy_set_header Host opragateway-ca-opra-dev.apps.ocp.
acca;

        proxy_pass http://opragateway-ca-opra-dev.apps.ocp.acca/;
        proxy_read_timeout 600s;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For
$proxy_add_x_forwarded_for;
        if ($request_method = 'OPTIONS') {
            add_header 'Access-Control-Allow-Origin' '*';
            add_header 'Access-Control-Allow-Headers' 'Content-
Type, userId, jwt, struserid, authorization,userid';
            add_header 'Access-Control-Allow-Methods' 'GET,
POST, OPTIONS, PUT, DELETE, PATCH';
            return 200;
        }
    }

    location /opra-ui {
        alias html/opra-ui;
        index index.html index.htm;
        try_files $uri $uri/ /index.html =404;

    }

        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;

    }

}

```

nginx开启监听端口，ssl使用443端口，需要加上ssl key，key可以自己生成。具体过程可以在网上查询。也可以使用已有的。拷贝过来即可。

启动自动收集日志的脚本

这个脚本用来打包nginx的日志，默认按月放置到文件夹中。需要将下面的脚本加到crontab中，定时执行。

```
#i/bin/bash

base_path='/data/nginx-server/logs'
bin_path='/data/nginx-server/sbin'
log_path=$(date -d yesterday +"%Y%m")
day=$(date -d yesterday +"%d")

mkdir -p $base_path/$log_path
mv $base_path/access.log $base_path/$log_path/access_$day.log

#echo $base_path/$log_path/access_$day.log
#kill -USR1 `cat /data/nginx-server/logs/nginx.pid`
$bin_path/nginx -s reload
```

执行

```
crontab -e

01 00 * * * /data/software/scripts/runlog.sh

crontab
sudo service crond start
```

前端代码打包部署

安装jenkins

略

修改前端代码的发布文件

artifact_deploy_pom.xml，以ca为例，增加发布的路径和服务器登录信息

```
<uatnginx.home>/data/nginx-server/html</uatnginx.home>
<uatca.server.address>scp://appadm:App1adm2@10.1.17.14</uatca.server.
address>
```

增加uatcdeploy的profile，Jenkins发布时需要用到

```

<profile>
  <id>uatcdeploy</id>
  <activation>
    <activeByDefault>true</activeByDefault>
  </activation>
  <build>
    <plugins>
      <plugin>
        <artifactId>maven-assembly-plugin</artifactId>
        <version>2.5.3</version>
        <configuration>
          <descriptors>
            <!-- -->
            <descriptor>assembly.xml</descriptor>
          </descriptors>
          <archive>

          </archive>
        </configuration>
        <executions>
          <execution>
            <!-- -->
            <id>make-assembly</id>
            <!-- package -->
            <phase>install</phase>
            <goals>
              <!-- -->
              <goal>single</goal>
            </goals>
          </execution>
        </executions>
      </plugin>
      <!-- Deploy Web/Schedule/Execution Subsystem to DEV Environment
-->
      <plugin>
        <groupId>com.github.goldin</groupId>
        <artifactId>copy-maven-plugin</artifactId>
        <version>0.2.5</version>
        <executions>
          <execution>
            <id>deploy-subsystem-archive</id>
            <phase>install</phase>
            <goals>
              <goal>copy</goal>
            </goals>
            <configuration>
              <resources>
                <resource>
                  <targetPath>${uatca.server.address}:${tmp.home}/<
/targetPath>
                  <file>${project.basedir}/target/${jar.name}</file>
                </resource>
              </resources>

```

```

        </configuration>
      </execution>
    </executions>
  </plugin>
  <plugin>
    <groupId>com.github.goldin</groupId>
    <artifactId>sshexec-maven-plugin</artifactId>
    <version>0.2.5</version>
    <executions>
      <execution>
        <id>restart-tomcat</id>
        <phase>install</phase>
        <goals>
          <goal>sshexec</goal>
        </goals>
        <configuration>
          <location>${uatca.server.address}:${uatnginx.home}/<
/location>
          <commands>
            <command>rm -rf ${cloud-front.path}</command>
            <command>tar -xzvf ${tmp.home}/${jar.name} -C ${tmp.home}
/ </command>
            <command>mv ${tmp.home}/dist/${cloud-front.path}
${uatnginx.home}/${cloud-front.path}</command>
          </commands>
        </configuration>
      </execution>

    </executions>
  </plugin>
</plugins>
</build>
</profile>

```

配置jenkins任务

以国航为例

opra-ui-uat-ca

GeneralSource Code ManagementBuild TriggersBuild EnvironmentPre StepsBuildPost Steps构建设置

Post-build Actions

Description

opra-ui-uat-ca
ca的uat环境
opra前端代码发布，所有模块都在这一个项目里面
从master分支发布

[纯文本] Preview

GitLab Connection

☐ This build requires lockable resources

☐ Throttle builds

☐ Prepare an environment for the run

☒ 丢弃旧的构建

StrategyLog Rotation

Days to keep builds1

if not empty, build records are only kept up to this number of days

Max # of builds to keep2

if not empty, only up to this number of build records are kept

SaveApply

opra-ui-uat-ca

GeneralSource Code ManagementBuild TriggersBuild EnvironmentPre StepsBuildPost Steps构建设置

Post-build Actions

Advanced...

☐ 参数化构建过程

☐ Disable this project

☐ Execute concurrent builds if necessary

☒ Restrict where this project can be run

Label Expressionaraf2

Label araf2 is serviced by 1 node. Permissions or other restrictions provided by plugins may prevent this job from running on those nodes.

Advanced...

选择git代码库，分支等选项。

opra-ui-uat-ca

General **Source Code Management** Build Triggers Build Environment Pre Steps Build Post Steps 构建设置

Post-build Actions

Source Code Management

☐ 无
☐ CVS
☐ CVS Projectset
☒ Git

Repositories

Repository URL

Credentials

Branches to build

Branch Specifier (blank for 'any')

pre step: 先从私服下载operetta包，在执行前端代码打包，再替换各个航空公司的log文件（替换logo文件的时候根据不同的公司取不同文件夹，前端这个文件夹要放入相应航司的图片）

opra-ui-uat-ca

General Source Code Management Build Triggers Build Environment **Pre Steps** Build Post Steps 构建设置

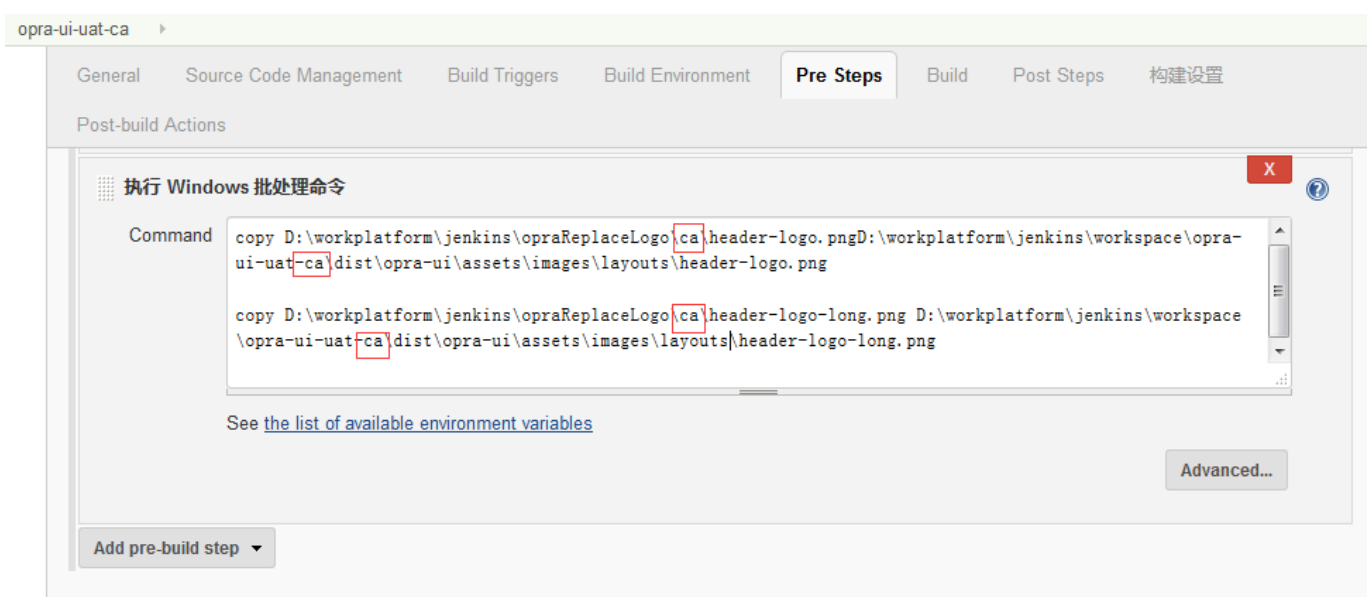
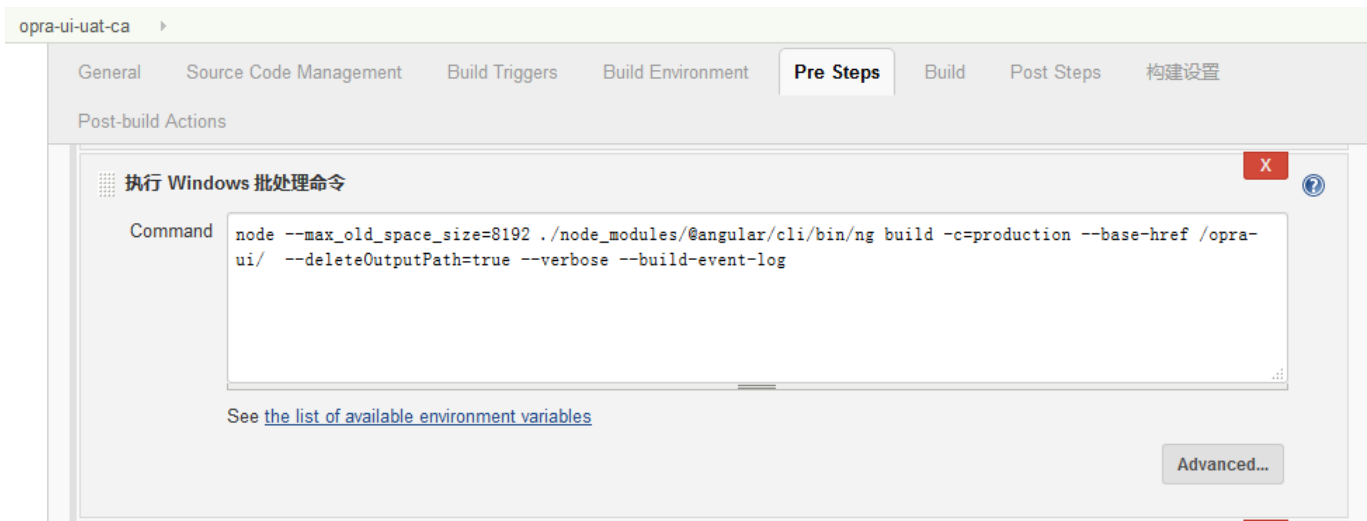
Post-build Actions

Pre Steps

执行 Windows 批处理命令

Command

[See the list of available environment variables](#)



artifact_deploy_pom.xml是前端代码中的发布文件，每发布一个环境需要在这个文件中增加对应的发布profile，这个profile会将打包好的代码上传到nginx-server的html对应的文件夹下。

opra-ui-uat-ca

General Source Code Management Build Triggers Build Environment Pre Steps **Build** Post Steps 构建设置

Post-build Actions

Build

Root POM

Goals and options

Advanced...

Post Steps

☐ Run only if build succeeds ☐ Run only if build succeeds or is unstable ☒ Run regardless of build result

Should the post-build steps run only for successful builds, etc.

Add post-build step

Jenkins完成之后点立即构建发布代码



Jenkins

Jenkins > OPRA > opra-ui-uat-ca

 [Back to Dashboard](#)

 [Status](#)

 [Changes](#)

 [Workspace](#)

 [立即构建](#)

 [Delete Maven project](#)

 [Configure](#)

 [模块](#)

 [Rename](#)

Maven project opra-ui-uat-ca

opra-ui-uat-ca
ca的uat环境
opra前端代码发布，所有模块都在这一个项目里面
从master分支发布

 [工作区](#)

 [最新修改](#)

 **Build History** [trend](#)

 [RSS for all](#)  [RSS for failures](#)

Permalinks

