Artigo
Danusa Calixto · Ago. 11, 2022 12min de leitura

ECP com Docker

Olá, comunidade,

Este é o terceiro artigo da série sobre a inicialização de instâncias da IRIS com Docker. Desta vez, focaremos no **E**nterprise **C**ache **P**rotocol (ECP).

De maneira bastante simplificada, o ECP permite configurar algumas instâncias da IRIS como servidores de aplicação e outras como servidores de dados. As informações técnicas detalhadas podem ser encontradas na documentação oficial.

O objetivo deste artigo é descrever o seguinte:

- Como programar a inicialização de um servidor de dados e como programar a inicialização de um ou mais servidores de aplicação.
- Como estabelecer uma conexão criptografada entre os nós com Docker.

Para fazer isso, geralmente, usamos algumas das ferramentas já abordadas nos artigos anteriores sobre Web Gateway e espelhamento (*Mirror*), que descrevem instrumentos como OpenSSL, envsubst e Config-API.

Requisitos

O ECP não está disponível com a versão da Comunidade da IRIS. Portanto, é necessário o acesso à Central de Suporte (WRC) para fazer o download de uma licença de contêiner e conectar ao registro containers.intersystems.com.

Preparando o sistema

O sistema precisa compartilhar alguns arquivos locais com os contêineres. É necessário criar determinados usuários e grupos para evitar o erro "acesso negado".

```
sudo useradd --uid 51773 --user-group irisowner
sudo useradd --uid 52773 --user-group irisuser
sudo groupmod --gid 51773 irisowner
sudo groupmod --gid 52773 irisuser
```

Se você ainda não tem a licença "iris.key", faça o download na WRC e adicione ao diretório do início.

Recupere o repositório de amostra

Todos os arquivos necessários estão disponíveis em um repositório público, exceto a licença "iris.key", então comece por clonar:

```
git clone https://github.com/lscalese/ecp-with-docker.git
```

```
cd ecp-with-docker
```

Certificados SSL

Para criptografar as comunicações entre os servidores de aplicação e o servidor de dados, precisamos de certificados SSL.
Um script pronto para uso ("gen-certificates.sh") está disponível. No entanto, fique à vontade para modificá-lo, para a consistência das configurações do certificado com sua localização, empresa etc.

Execute:

```
sh ./gen-certificates.sh
```

Os certificados gerados estão agora no diretório "./certificates".

Arquivo	Contêiner	Descrição
./certificates/CAServer.cer	Servidor de aplicação e servidor de dados	Certificado do servidor da autoridade
./certificates/appserver.cer	Servidor de aplicação	Certificado para a instância do servidor de aplicação da IRIS
./certificates/appserver.key	Servidor de aplicação	Chave privada relacionada
./certificates/dataserver.cer	Servidor de dados	Certificado para a instância do servidor de dados da IRIS
./certificates/dataserver.key	Servidor de dados	Chave privada relacionada

Construa a imagem

Primeiro, faça login no registro docker da InterSystems. A imagem de base será baixada do registro durante a construção:

```
docker login -u="YourWRCLogin" -p="YourICRToken" containers.intersystems.com
```

Se você não sabe seu token, faça login em https://containers.intersystems.com/ com a conta da WRC.

Durante essa construção, adicionaremos alguns utilitários de software à imagem de base da IRIS:

- **gettext-base**: permitirá substituir as variáveis do ambiente nos arquivos de configuração com o comando "envsubst".
- **iputils-arping**: é necessário se quisermos espelhar o servidor de dados.
- **ZPM**: gerente de pacotes ObjectScript.

Dockerfile:

```
ARG IMAGE=containers.intersystems.com/intersystems/iris:2022.2.0.281.0

# Não é necessário fazer o download da imagem do WRC. Ela será extraída do ICR no mom
ento da construção.

FROM $IMAGE

USER root

# Installe iputils-
```

```
arping para ter um comando de arping.  É necessário para configurar o IP Virtual.
# Baixe a última versão do ZPM (incluso somente na edição da comunidade).
RUN apt-get update && apt-get install iputils-arping gettext-base && \
   rm -rf /var/lib/apt/lists/*

USER ${ISC_PACKAGE_MGRUSER}

WORKDIR /home/irisowner/demo

RUN --mount=type=bind,src=.,dst=. \
   iris start IRIS && \
       iris session IRIS < iris.script && \
   iris stop IRIS quietly
```

Não há nada especial neste Dockerfile, exceto a última linha. Ela configura a instância do servidor de dados da IRIS para aceitar até 3 servidores de aplicação. Atenção: essa configuração exige a reinicialização da IRIS. Atribuímos o valor deste parâmetro durante a construção para evitar a reinicialização do script depois.

Inicie a construção:

```
docker-compose build –no-cache
```

Arquivo de configuração

Para a configuração das instâncias da IRIS (servidores de aplicação e dados), usamos o formato de arquivo SON config-api. Você verá que esses arquivos contêm variáveis de ambiente "${variablename}". Os valores são definidos nas seções do "ambiente" do arquivo "docker-compose.yml" que veremos ainda neste documento. Essas variáveis serão substituídas logo antes do carregamento dos arquivos usando o utilitário "envsubst".

Servidor de dados

Para o servidor de dados, vamos:

- Permitir o serviço ECP e definir a lista de clientes autorizados (servidores de aplicação).
- Criar a configuração "SSL %ECPServer" necessária para a criptografia de comunicações.
- Criar uma base de dados "myappdata". Ela será usada como uma base de dados remota dos servidores de aplicação.

(data-serer.json)[https://github.com/lscalese/ecp-with-docker/blob/master/config-files/dat...

```
{
   "Security.Services" : {
       "%Service_ECP" : {
           "Enabled" : true,
           "ClientSystems":"${CLIENT_SYSTEMS}",
           "AutheEnabled":"1024"
       }
   },
   "Security.SSLConfigs": {
       "%ECPServer": {
           "CAFile": "${CA_ROOT}",
           "CertificateFile": "${CA_SERVER}",
           "Name": "%ECPServer",
```

```
            "PrivateKeyFile": "${CA_PRIVATE_KEY}",
            "Type": "1",
            "VerifyPeer": 3
        }
    },
    "Security.System": {
        "SSLECPServer":1
    },
    "SYS.Databases":{
        "/usr/irissys/mgr/myappdata/" : {}
    },
    "Databases":{
        "myappdata" : {
            "Directory" : "/usr/irissys/mgr/myappdata/"
        }
    }
}
```

Esse arquivo de configuração é carregado na inicialização do contêiner do servidor de dados pelo script "initdatasrv.sh". Todos os servidores de aplicação conectados ao servidor de dados precisam ser confiáveis. Esse script validará automaticamente todas as conexões dentro de 100 segundos para limitar as ações manuais no portal de administração. É claro que isso pode ser melhorado para aprimorar a segurança.

Servidor de aplicação

Para os servidores de aplicação, vamos:

- Ativar o serviço ECP.
- Criar a configuração do SLL "%ECPClient" necessária para a criptografia da comunicação.
- Configurar as informações da conexão para o servidor de dados.
- Criar a configuração da base de dados remota "myappdata".
- Criar um mapeamento global "demo.*" no espaço de nome "USER" para a base de dados "myappdata". Isso permitirá o teste da operação do ECP mais tarde.

app-server.json:

```
{
    "Security.Services" : {
        "%Service_ECP" : {
            "Enabled" : true
        }
    },
    "Security.SSLConfigs": {
        "%ECPClient": {
            "CAFile": "${CA_ROOT}",
            "CertificateFile": "${CA_CLIENT}",
            "Name": "%ECPClient",
            "PrivateKeyFile": "${CA_PRIVATE_KEY}",
            "Type": "0"
        }
    },
    "ECPServers" : {
        "${DATASERVER_NAME}" : {
            "Name" : "${DATASERVER_NAME}",
            "Address" : "${DATASERVER_IP}",
```

```
            "Port" : "${DATASERVER_PORT}",
            "SSLConfig" : "1"
        }
    },
    "Databases": {
        "myappdata" : {
            "Directory" : "/usr/irissys/mgr/myappdata/",
            "Name" : "${REMOTE_DB_NAME}",
            "Server" : "${DATASERVER_NAME}"
        }
    },
    "MapGlobals":{
        "USER": [{
            "Name" : "demo.*",
            "Database" : "myappdata"
        }]
    }
}
```

O arquivo de configuração é carregado na inicialização de um contêiner do servidor de aplicação pelo script "initappsrv.sh ".

Inicializando os contêineres

Agora, podemos inicializar os contêineres:

- 2 servidores de aplicação.
- 1 servidor de dados.

Para fazer isso execute:

docker-compose up –scale ecp-demo-app-server=2

Veja mais detalhes no arquivo docker-compose:

```
# As variáveis são definidas no arquivo .env
# para mostrar o arquivo docker-compose revolvido, execute
# docker-compose config

version: '3.7'

services:
  ecp-demo-data-server:
    build: .
    image: ecp-demo
    container_name: ecp-demo-data-server
    hostname: data-server
    networks:
      app_net:
    environment:
      # Lista dos clientes ECP permitidos (servidor de aplicação).
      - CLIENT_SYSTEMS=ecp-with-docker_ecp-demo-app-server_1;ecp-with-docker_ecp-demo-
app-server_2;ecp-with-docker_ecp-demo-app-server_3
      # Path authority server certificate
      - CA_ROOT=/certificates/CA_Server.cer
```

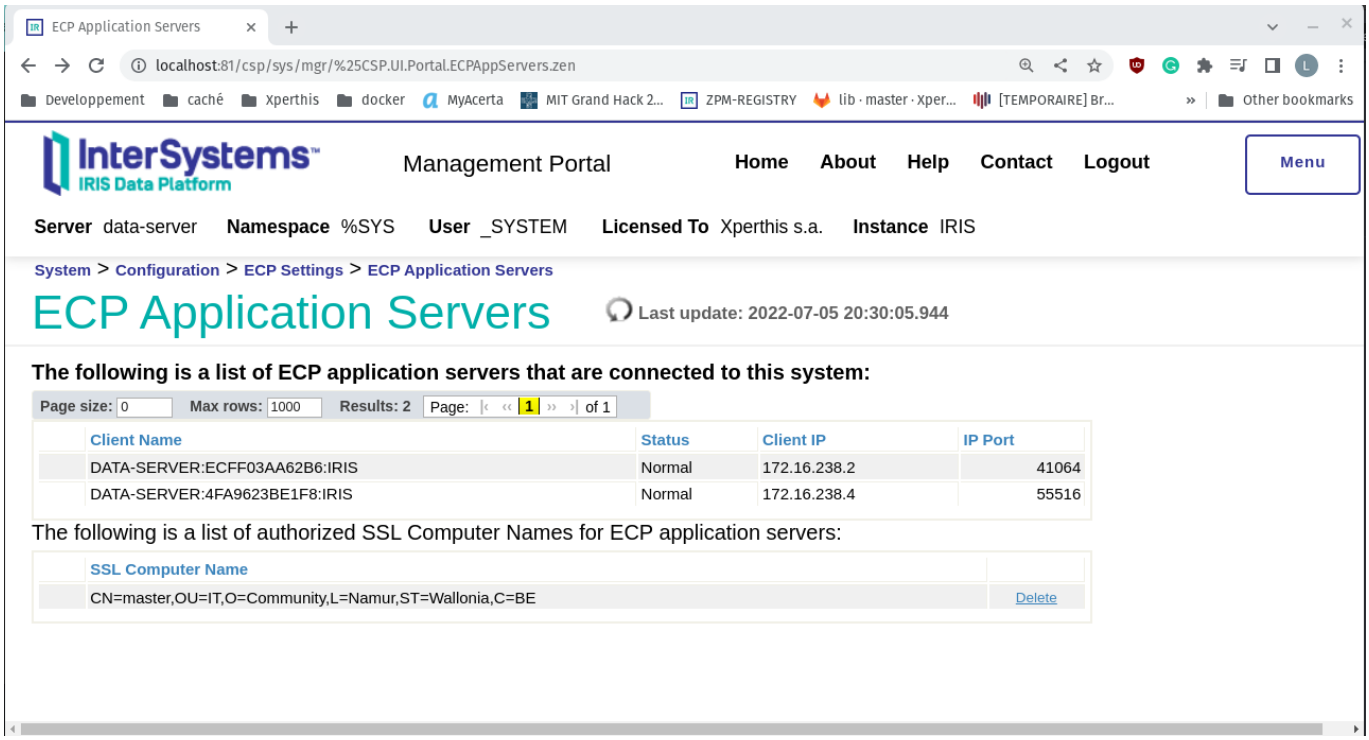
```
    # Path to data server certificate
    - CA_SERVER=/certificates/data_server.cer
    # Path to private key of the data server certificate
    - CA_PRIVATE_KEY=/certificates/data_server.key
    # Path to Config-API file to initiliaze this IRIS instance
    - IRIS_CONFIGAPI_FILE=/home/irisowner/demo/data-server.json
  ports:
    - "81:52773"
  volumes:
    # Script após o início - inicialização do servidor de dados.
    - ./init_datasrv.sh:/home/irisowner/demo/init_datasrv.sh
    # Montar certificados (ver gen-certificates.sh para gerar certificados)
    - ./certificates/app_server.cer:/certificates/data_server.cer
    - ./certificates/app_server.key:/certificates/data_server.key
    - ./certificates/CA_Server.cer:/certificates/CA_Server.cer
    # Montar arquivo de configuração
    - ./config-files/data-server.json:/home/irisowner/demo/data-server.json
    # Licença da IRIS
    - ~/iris.key:/usr/irissys/mgr/iris.key
  command: -a /home/irisowner/demo/init_datasrv.sh


ecp-demo-app-server:
  image: ecp-demo
  networks:
    app_net:
  environment:
    # Nome do host ou ID do servidor de dados.
    - DATASERVER_IP=data-server
    - DATASERVER_NAME=data-server
    - DATASERVER_PORT=1972
    # Caminho do certificado do servidor de autoridade
    - CA_ROOT=/certificates/CA_Server.cer
    - CA_CLIENT=/certificates/app_server.cer
    - CA_PRIVATE_KEY=/certificates/app_server.key
    - IRIS_CONFIGAPI_FILE=/home/irisowner/demo/app-server.json
  ports:
    - 52773
  volumes:
    # Após o início do script - inicialização do servidor de aplicação.
    - ./init_appsrv.sh:/home/irisowner/demo/init_appsrv.sh
    # Montar certificados
    - ./certificates/CA_Server.cer:/certificates/CA_Server.cer
    # Caminho para a chave privada do certificado do servidor de dados
    - ./certificates/app_server.cer:/certificates/app_server.cer
    # Caminho para a chave privada do certificado do servidor de dados
    - ./certificates/app_server.key:/certificates/app_server.key
    # Caminho para o arquivo Config-API inicializar a instância da IRIS
    - ./config-files/app-server.json:/home/irisowner/demo/app-server.json
    # Licença da IRIS
    - ~/iris.key:/usr/irissys/mgr/iris.key
  command: -a /home/irisowner/demo/init_appsrv.sh
networks:
app_net:
  ipam:
    driver: default
    config:
      # A variável APP_NET_SUBNET é definida no arquivo .env
      - subnet: "${APP_NET_SUBNET}"
```

Vamos testar!

Acesso ao portal de administração do servidor de dados

Os contêineres foram inicializados. Vamos conferir o status do servidor de dados.
A porta 52773 é mapeada para a porta local 81, então você pode acessar por este endereço:
http://localhost:81/csp/sys/utilhome.csp

Entre com o login e a senha padrão e acesse System -> Configuration -> ECP Params. Clique em "ECP Application Servers". Se tudo estiver funcionando, você verá 2 servidores de aplicação com o status "Normal". A estrutura do nome do cliente é "nome do servidor de dados":"hostname do servidor de aplicação":"nome da instância IRIS". No nosso caso, não configuramos os hostnames do servidor de aplicação, então temos hostnames gerados.



Acessando o portal de administração dos servidores de aplicação

Para se conectar ao portal de administração dos servidores de aplicação, primeiro, você precisa ter o número da porta. Já que usamos a opção "--scale", não conseguimos definir as portas no arquivo docker-compose. Então, precisamos recuperá-las com o comando docker ps:

```
docker ps -a
CONTAINER ID   IMAGE      COMMAND                  CREATED          STATUS
          PORTS
               NAMES
a1844f38939f   ecp-demo   "/tini -- /iris-main…"   25 minutes ago   Up 25 minutes (un
healthy)   1972/tcp, 2188/tcp, 53773/tcp, 54773/tcp, 0.0.0.0:81->52773/tcp, :::81->52
773/tcp         ecp-demo-data-server
4fa9623be1f8   ecp-demo   "/tini -- /iris-main…"   25 minutes ago   Up 25 minutes (un
healthy)   1972/tcp, 2188/tcp, 53773/tcp, 54773/tcp, 0.0.0.0:49170->52773/tcp, :::491
70->52773/tcp   ecp-with-docker_ecp-demo-app-server_1
ecff03aa62b6   ecp-demo   "/tini -- /iris-main…"   25 minutes ago   Up 25 minutes (un
healthy)   1972/tcp, 2188/tcp, 53773/tcp, 54773/tcp, 0.0.0.0:49169->52773/tcp, :::491
```

```
69->52773/tcp   ecp-with-docker_ecp-demo-app-server_2
```

Neste exemplo, as portas são:

- 49170 para o primeiro servidor de aplicação http://localhost:49170/csp/sys/utilhome.csp
- 49169 para o segundo servidor de aplicação http://localhost:49169/csp/sys/utilhome.csp

Teste de leitura/escrita na base de dados remota

Vamos realizar alguns testes de leitura/escrita no terminal.
Abra um terminal IRIS no primeiro servidor de aplicação:

```
docker exec -it ecp-with-docker_ecp-demo-app-server_1 iris session iris
Set ^demo.ecp=$zdt($h,3,1) _ “ write from the first application server.”
```

Agora, abra um terminal no segundo servidor de aplicação:

```
docker exec -it ecp-with-docker_ecp-demo-app-server_2 iris session iris
Set ^demo.ecp(2)=$zdt($h,3,1) _ " write from the second application server."
zwrite ^demo.ecp
```

Você deverá ver as respostas dos dois servidores:

```
^demo.ecp(1)="2022-07-05 23:05:10 write from the first application server."
^demo.ecp(2)="2022-07-05 23:07:44 write from the second application server."
```

Por fim, abra um terminal IRIS no servidor de dados e realize a leitura do global demo.ecp:

```
docker exec -it ecp-demo-data-server iris session iris
zwrite ^["^^/usr/irissys/mgr/myappdata/"]demo.ecp

^["^^/usr/irissys/mgr/myappdata/"]demo.ecp(1)="2022-07-05 23:05:10 write from the fir
st application server."
^["^^/usr/irissys/mgr/myappdata/"]demo.ecp(2)="2022-07-05 23:07:44 write from the sec
ond application server."
```

Isso é tudo por hoje. Espero que tenha gostado deste artigo. Não hesite em deixar seus comentários.

#DevOps #Implantação #InterSystems IRIS

URL de origem:https://pt.community.intersystems.com/post/ecp-com-docker