

Article

[Lorenzo Scalese](#) · Apr 22 8m read

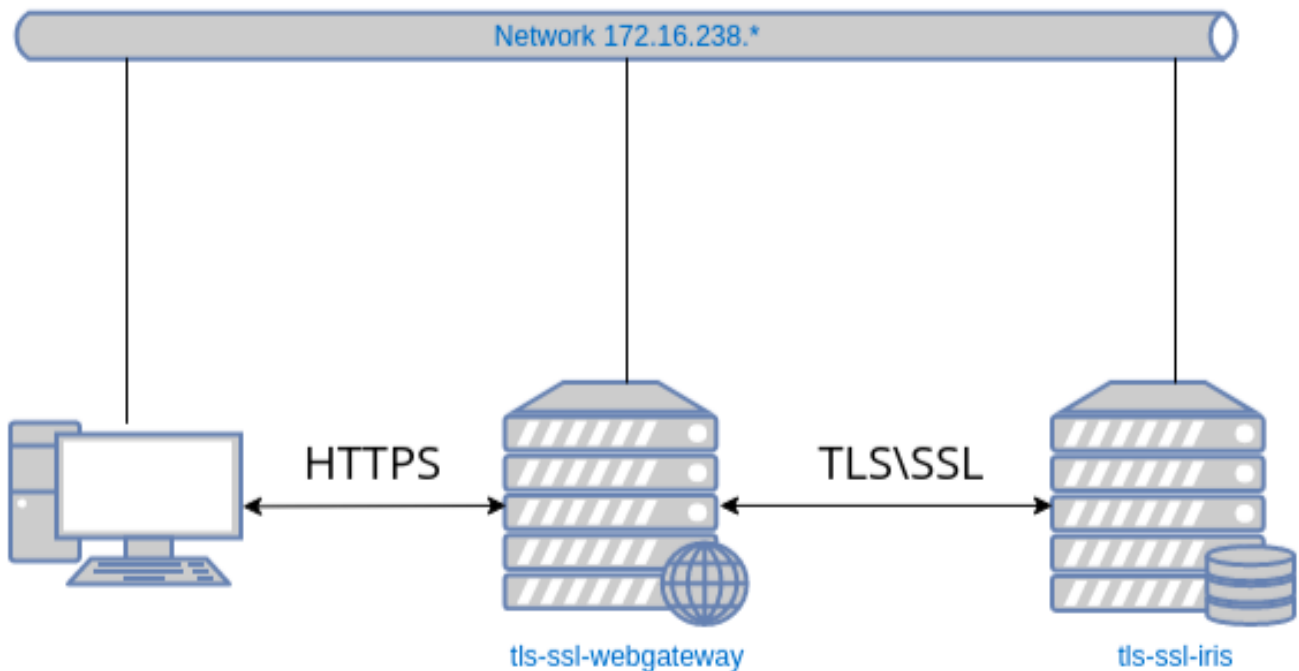
## Apache Web Gateway with Docker

## Apache Web Gateway with Docker

Hi, community.

In this article, we will programmatically configure an Apache Web Gateway with Docker using:

- HTTPS protocol.
- TLS / SSL to secure the communication between the Web Gateway and the IRIS instance.



We will use two images: one for the Web Gateway and the second one for the IRIS instance.

All necessary files are available in this [GitHub repository](#).

Let's start with a git clone:

```
git clone https://github.com/lscalese/docker-webgateway-sample.git
cd docker-webgateway-sample
```

### Prepare your system

To avoid problems with permissions, your system needs a user and a group:

- www-data

- irisowner

It's required to share certificates files with the containers. If they don't exist on your system, simply execute:

```
sudo useradd --uid 51773 --user-group irisowner
sudo groupmod --gid 51773 irisowner
sudo useradd -user-group www-data
```

## Generate certificates

In this sample, we will use three certificates:

1. HTTPS web server usage.
2. TLS / SSL encryption on Web Gateway client.
3. TLS / SSL encryption on IRIS Instance.

A script ready-to-use is available to generate them.

However, you should customize the subject of the certificate; simply edit the [gen-certificates.sh](#) file.

This is the structure of OpenSSL subj argument:

1. C: Country code
2. ST: State
3. L: Location
4. O: Organization
5. OU: Organization Unit
6. CN: Common name (basically the domain name or the hostname)

Feel free to change these values.

```
# sudo is needed due chown, chgrp, chmod ...
sudo ./gen-certificates.sh
```

If everything is ok, you should see two new directories `./certificates/` and `~/webgateway-apache-certificates/` with certificates:

File	Container	Description
<code>./certificates/CA_Server.cer</code>	<code>webgateway,iris</code>	Authority server certificate
<code>./certificates/iris_server.cer</code>	<code>iris</code>	Certificate for IRIS instance (used for mirror and wegateway communication encryption)
<code>./certificates/iris_server.key</code>	<code>iris</code>	Related private key
<code>~/webgateway-apache-certificates/apache_webgateway.cer</code>	<code>webgateway</code>	Certificate for apache webserver
<code>~/webgateway-apache-certificates/apache_webgateway.key</code>	<code>webgateway</code>	Related private key
<code>./certificates/webgateway_client.cer</code>	<code>webgateway</code>	Certificate to encrypt communication between webgateway and IRIS
<code>./certificates/webgateway_client.key</code>	<code>webgateway</code>	Related private key

Keep in mind that if there are self-signed certificates, web browsers will show security alerts. Obviously, if you have a certificate delivered by a certified authority, you can use it instead of a self-signed one (especially for the Apache server certificate).

## Web Gateway Configuration files

Take a look at the configuration files.

### CSP.INI

You can see a CSP.INI file in the `webgateway-config-files` directory. It will be pushed into the image, but the content can be modified at runtime. Consider this file as a template.

In this sample the following parameters will be overridden on container startup:

- `Ip_Address`
- `TCP_Port`
- `System_Manager`

See [startUpScript.sh](#) for more details. Roughly, the replacement is performed with the `sed` command line.

Also, this file contains the SSL / TLS configuration to secure the communication with the IRIS instance:

```
SSLCC_Certificate_File=/opt/webgateway/bin/webgateway_client.cer
SSLCC_Certificate_Key_File=/opt/webgateway/bin/webgateway_client.key
SSLCC_CA_Certificate_File=/opt/webgateway/bin/CA_Server.cer
```

These lines are important. We must ensure the certificate files will be available for the container. We will do that later in the `docker-compose` file with a volume.

### 000-default.conf

This is an Apache configuration file. It allows the use of HTTPS protocol and redirects HTTP calls to HTTPS. Certificate and private key files are setup in this file:

```
SSLCertificateFile /etc/apache2/certificate/apache_webgateway.cer
SSLCertificateKeyFile /etc/apache2/certificate/apache_webgateway.key
```

## IRIS instance

For our IRIS instance, we configure only the minimal requirement to allow the SSL / TLS communication with the Web Gateway; it involves:

1. `%SuperServer SSL Config`.
2. Enable `SSLSuperServer` security setting.
3. Restrict the list of IPs that can use the Web Gateway service.

To ease the configuration, `config-api` is used with a simple JSON configuration file.

```
{
```

```

"Security.SSLConfigs": {
  "%SuperServer": {
    "CAFile": "/usr/irissys/mgr/CA_Server.cer",
    "CertificateFile": "/usr/irissys/mgr/iris_server.cer",
    "Name": "%SuperServer",
    "PrivateKeyFile": "/usr/irissys/mgr/iris_server.key",
    "Type": "1",
    "VerifyPeer": 3
  }
},
"Security.System": {
  "SSLSuperServer":1
},
"Security.Services": {
  "%Service_WebGateway": {
    "ClientSystems": "172.16.238.50;127.0.0.1;172.16.238.20"
  }
}
}

```

There is no action needed. The configuration will be automatically loaded on container startup.

## Image tls-ssl-webgateway

### dockerfile

```

ARG IMAGEWEBGTW=containers.intersystems.com/intersystems/webgateway:2021.1.0.215.0
FROM ${IMAGEWEBGTW}
ADD webgateway-config-files /webgateway-config-files
ADD buildWebGateway.sh /
ADD startUpScript.sh /
RUN chmod +x buildWebGateway.sh startUpScript.sh && /buildWebGateway.sh
ENTRYPOINT ["/startUpScript.sh"]

```

By default the entry point is /startWebGateway, but we need to perform some operations before starting the webserver. Remember that our CSP.ini file is a template, and we need to change some parameters (IP, port, system manager) on starting. startUpScript.sh will perform these changes and then execute the initial entry point script /startWebGateway.

## Starting containers

### docker-compose file

Before starting containers, the docker-compose.yml file must be modified:

- **\*\*SYSTEM\_MANAGER\*\*** must be set with the IP authorized to have an access to Web Gateway Management <https://localhost/csp/bin/Systems/Module.cxw>  
Basically, it's your IP address (It could be a comma-separated list).
- **\*\*IRIS\_WEBAPPS\*\*** must be set with the list of your CSP applications. The list is separated by space, for example: IRIS\_WEBAPPS=/csp/sys /swagger-ui. By default, only /csp/sys is exposed.

- Ports 80 and 443 are mapped. Adapt them to other ports if they are already used on your system.

```
version: '3.6'
```

```
services:
```

```
webgateway:
  image: tls-ssl-webgateway
  container_name: tls-ssl-webgateway
  networks:
    app_net:
      ipv4_address: 172.16.238.50
  ports:
    # change the local port already used on your system.
    - "80:80"
    - "443:443"
  environment:
    - IRIS_HOST=172.16.238.20
    - IRIS_PORT=1972
    # Replace by the list of ip address allowed to open the CSP system manager
    # https://localhost/csp/bin/Systems/Module.cxx
    # see .env file to set environment variable.
    - "SYSTEM_MANAGER=${LOCAL_IP}"
    # the list of web apps
    # /csp allow to the webgateway to redirect all request starting by /csp to the i
ris instance
    # You can specify a list separate by a space : "IRIS_WEBAPPS=/csp /api /isc /swa
gger-ui"
    - "IRIS_WEBAPPS=/csp/sys"
  volumes:
    # Mount certificates files.
    - ./volume-
apache/webgateway_client.cer:/opt/webgateway/bin/webgateway_client.cer
  - ./volume-
apache/webgateway_client.key:/opt/webgateway/bin/webgateway_client.key
  - ./volume-apache/CA_Server.cer:/opt/webgateway/bin/CA_Server.cer
  - ./volume-
apache/apache_webgateway.cer:/etc/apache2/certificate/apache_webgateway.cer
  - ./volume-
apache/apache_webgateway.key:/etc/apache2/certificate/apache_webgateway.key
  hostname: webgateway
  command: ["--ssl"]

iris:
  image: intersystemsdc/iris-community:latest
  container_name: tls-ssl-iris
  networks:
    app_net:
      ipv4_address: 172.16.238.20
  volumes:
    - ./iris-config-files:/opt/config-files
    # Mount certificates files.
    - ./volume-iris/CA_Server.cer:/usr/irissys/mgr/CA_Server.cer
    - ./volume-iris/iris_server.cer:/usr/irissys/mgr/iris_server.cer
    - ./volume-iris/iris_server.key:/usr/irissys/mgr/iris_server.key
  hostname: iris
  # Load the IRIS configuration file ./iris-config-files/iris-config.json
  command: ["-a", "sh /opt/config-files/configureIris.sh"]
```

```
networks:
```

```
app_net:
  ipam:
    driver: default
    config:
      - subnet: "172.16.238.0/24"
```

Build and start:

```
docker-compose up -d --build
```

Containers `tls-ssl-iris` and `tls-ssl-webgateway` should be started.

## Test Web Access

### Apache default page

Open the page <http://localhost>.

You will be automatically redirected to <https://localhost>.

The browsers show security alerts. This is the standard behaviour with a self-signed certificate, accept the risk and continue.

### Web Gateway management page

Open <https://localhost/csp/bin/Systems/Module.cxw> and test the server connection.

### Management portal

Open <https://localhost/csp/sys/utilhome.csp>

Great! The Web Gateway sample is working!

## IRIS Mirror with Web Gateway

In the previous article, we built a mirror environment, but the Web Gateway was a missing piece. Now, we can improve that.

A new repository [iris-mirroring-with-webgateway](#) is available including Web Gateway and a few more improvements:

1. Certificates are no longer generated on the fly but in a separate process.
2. IP Addresses are replaced by environment variables in docker-compose and JSON configuration files. Variables are defined in the '.env' file.
3. The repository can be used as a template.

See the repository [README.md](#) file to run an environment like this:

[#Best Practices](#) [#DevOps](#) [#Web Gateway](#) [#InterSystems IRIS](#)