Running QGIS Server in Production

Speaker notes

This talk reports on our experience (at Oslandia) with running QGIS Server in Production. I will describe the options you have to running QGIS Server, and discuss the advantages and drawbacks of each option, based on my experience.

Éric Lemoine

Software Engineer at Oslandia

eric.lemoine@oslandia.com
@elemoine
@erilem

Speaker notes

I am not a core QGIS developer, but I often set up QGIS Server for clients in production contexts.

Agenda

- introduction to QGIS Server
- CGI and FastCGI
- how to run QGIS Server
- a word about Docker

Introduction to QGIS Server

Speaker notes

This is not an official logo, but it might be in the future. It was created by one of my colleagues, Sylvain Beorchia.

QGIS Server

- OGC Server
- competes with MapServer and GeoServer
- supports WMS, WFS, WFS-T and WMTS
- includes lots of vendor specificities (e.g. `GetPrint`)

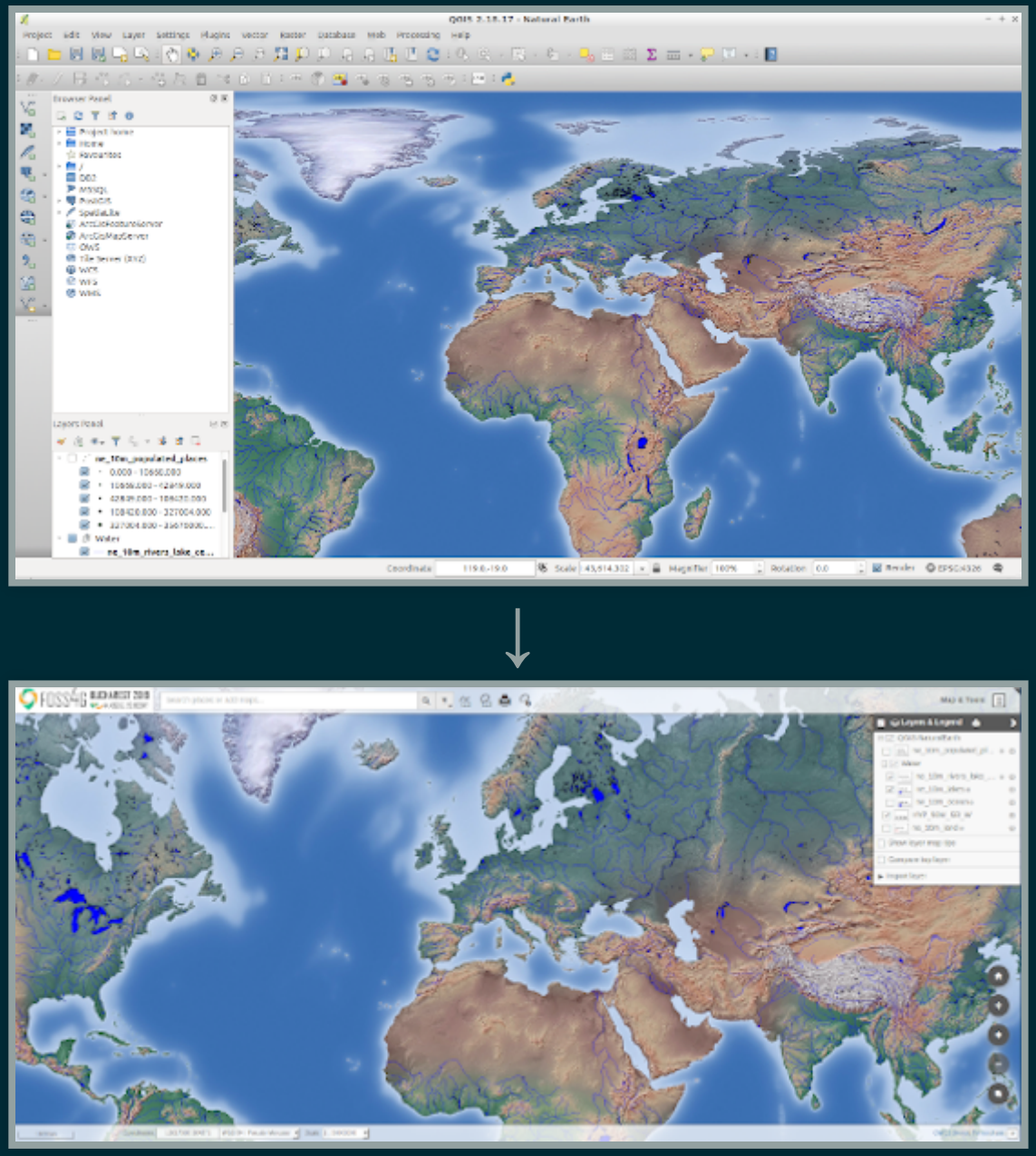
Speaker notes

QGIS Server is an opensource OGC Server, just like MapServer and GeoServer.

« Publish your QGIS projects to the web »

Speaker notes

You design and create maps using QGIS Desktop, and you can then use QGIS Server to expose your maps and geographic features through OGC services.

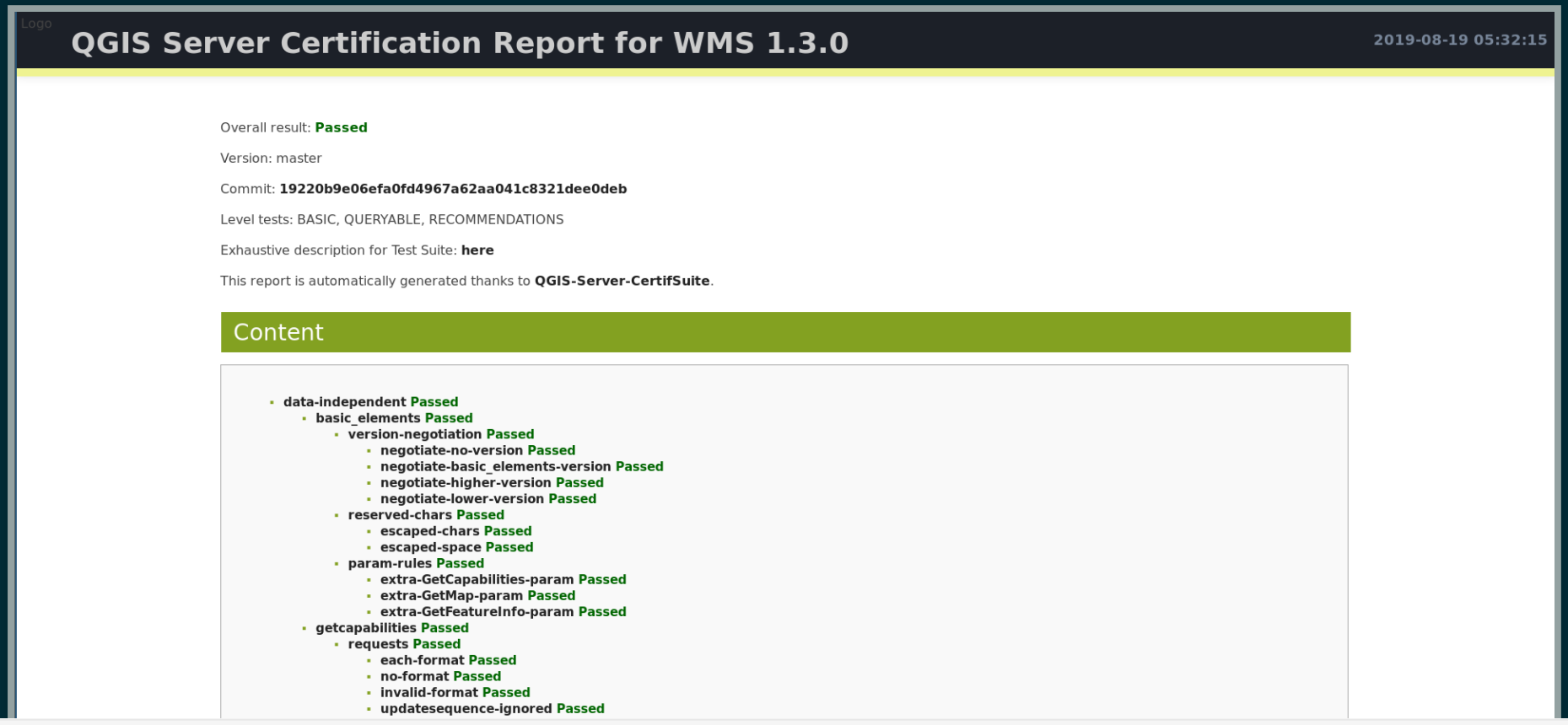


From QGIS 2 to 3

QGIS Server's changed a lot between version 2 and 3!

- multi-thread rendering as in QGIS Desktop
- WFS 1.1 support (WFS 3 now supported!)
- full Python bindings for the server API
- server services as plugins
- OGC Compliancy Platform

OGC Compliancy


Logo

QGIS Server Certification Report for WMS 1.3.0

2019-08-19 05:32:15

Overall result: **Passed**

Version: master

Commit: **19220b9e06efa0fd4967a62aa041c8321dee0deb**

Level tests: BASIC, QUERYABLE, RECOMMENDATIONS

Exhaustive description for Test Suite: **here**

This report is automatically generated thanks to **QGIS-Server-CertifSuite**.

Content

- data-independent Passed
  - basic_elements Passed
    - version-negotiation Passed
      - negotiate-no-version Passed
      - negotiate-basic_elements-version Passed
      - negotiate-higher-version Passed
      - negotiate-lower-version Passed
    - reserved-chars Passed
      - escaped-chars Passed
      - escaped-space Passed
    - param-rules Passed
      - extra-GetCapabilities-param Passed
      - extra-GetMap-param Passed
      - extra-GetFeatureInfo-param Passed
  - getcapabilities Passed
    - requests Passed
      - each-format Passed
      - no-format Passed
      - invalid-format Passed
      - updatesequence-ignored Passed

Speaker notes

this screenshot shows an example of a report page generated every day on the platform.

CGI and FastCGI

Speaker notes

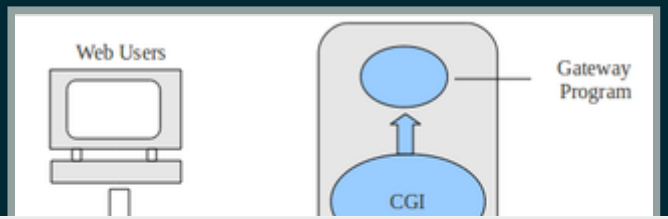
Let's now get into more technical stuff. To understand how to run QGIS Server requires some understanding of the CGI and FastCGI standards.

QGIS Server is a CGI/FastCGI program

CGI

« Common Gateway Interface »

A standard protocol for web servers to execute programs that generate web pages dynamically



Speaker notes

CGI is very old.

The HTTP headers and the query parameters are passed to the CGI program through environment variables.

CGI (cont'd)

« One process per request »

A CGI application process is created at the start of a request and torn down at the end

→ High overhead!

CGI (cont'd)

CGI is dead-simple, but unappropriate for high-end applications

Speaker notes

CGI is certainly not appropriate for QGIS Server.

Here comes FastCGI!

Instead of creating a new process for each request, FastCGI uses persistent processes to handle a series of requests

Speaker notes

FastCGI came after CGI, but it's still very old. Yet very much used for web applications written in C or C++.

FastCGI (cont'd)

FastCGI is much more efficient than CGI

→ Make sure you use FastCGI for QGIS Server! Critical to get decent performance

Speaker notes

Think about the case where the program opens and manages database connections.

There's some compatibility between FastCGI and CGI. FastCGI applications are also CGI applications.

Executing QGIS Server as a FastCGI program

Web Servers

Many web servers support FastCGI

Apache and NGINX are very popular web servers (for good reasons)

Speaker notes

Apache and NGINX are very flexible, they can be configured to work in many different ways, and they include lots of modules of various types.

Apache vs NGINX for FastCGI

Apache (with mod_fcgid) manages the application processes (a.k.a. the FastCGI processes)

NGINX does not!

Speaker notes

when using NGINX you have to set up some mechanism for creating the application processes, and the socket for the FastCGI communication between NGINX and the application.

Apache with mod_fcgid

Still a very good choice! Many advantages:

- quite easy to set up
- mod_fcgid manages the FastCGI processes
- multiple processes for concurrent requests
- allows for robust setups (e.g. `FcgidMaxRequestsPerProcess`)

Apache with mod_fcgid (cont'd)

```
<VirtualHost *:80>

 ...

 FcgidInitialEnv DISPLAY ":99"
 FcgidInitialEnv QGIS_SERVER_LOG_LEVEL "0"
 FcgidInitialEnv QGIS_SERVER_LOG_STDERR "1"

 <Location /qgisserver>
  SetHandler fcgid-script
  FcgidWrapper /usr/lib/cgi-bin/qgis_mapserv.fcgi virtual
  Options +ExecCGI -MultiViews +FollowSymLinks
  Require all granted
 </Location>

</VirtualHost>
```

Apache with mod_proxy_fcgi

Another FastCGI module for Apache

mod_proxy_fcgi doesn't manage the FastCGI processes

```
<VirtualHost *:80>

 <Location /qgisserver>
  ProxyPass "fcgi://localhost:4000/"
 </Location>

</VirtualHost>
```

Speaker notes

here you just indicate to mod_proxy_fcgi how to reach the FastCGI process. Creating the FastCGI process is not the responsibility of Apache.

NGINX

Supports FastCGI through ngx_http_fastcgi_module

NGINX doesn't manage the FastCGI processes

NGINX (cont'd)

```
location /qgisserver {
    gzip            off;
    include         fastcgi_params;
    fastcgi_pass    unix:/var/run/qgisserver.socket;
}
```

How to manage the FastCGI processes then?

The easiest: `fcgiwrap`

- packaged by all common Linux distros
- the package includes a Systemd service for it

But don't use it! It's slow!

(`fcgiwrap` runs the wrapped programs as CGI programs)

A better choice: `spawn-fcgi`

- a thin wrapper around your FastCGI program
- requires more work on your side
- in part. you will need to write a Systemd service for it

Perfect for running QGIS Server in Docker!

See https://github.com/Oslandia/docker-qgis

Speaker notes

A wrapper is required for creating the socket used for the communication between the web server and the FastCGI program.

Another good choice: rely on Systemd

- the "thin" wrapper is Systemd in this case
- one component less! (less is better)

Check out the Oslandia blog post for more information:

https://oslandia.com/en/2018/11/23/deploying-qgis-server-with-systemd/

To sum it up!

- Apache and NGINX both work well (no surprise)
- Apache mod_fcgid is easy, flexible, and robust
- don't use fcgiwrap for managing FastCGI processes
- use spawn-fcgi or Systemd directly

Docker

- there are many Docker images for QGIS Server
- often based on Debian
- very useful to run QGIS Server on any distro

`docker-qgis` (by Oslandia)

https://github.com/Oslandia/docker-qgis

- based on Debian
- uses `spawn-fcgi`
- exposes a FastCGI TCP socket
- doesn't include any web server
- very flexible / well documented
- we want to make it official! (QEP)

Thank you!

Oslandia Blog Posts on QGIS Server

- OGC Certification work for WFS 1.1.0
- QGIS Server: continuous integration platform
- Deploying QGIS Server with Systemd
- QGIS 3 and performance analysis

Éric Lemoine

Software Engineer at Oslandia

eric.lemoine@oslandia.com
@elemoine
@erilem