



Exploring Caddy + Let's Encrypt + HTTP/2 “Serve The Web Like It's 2016”

by Fernando Álvarez
GoMAD - Madrid Go UG - 16/03/2016

Summary

A blurred night city street with bokeh lights from cars and buildings. The image is dark with many out-of-focus light sources in warm and cool tones, creating a bokeh effect. The lights are arranged in a way that suggests a perspective view down a street.

Summary

- Who Am I

Summary

- **Who Am I**
- **Caddy server**

Summary

- **Who Am I**
- **Caddy server**
- **HTTP/2 (aka h2)**

Summary

- **Who Am I**
- **Caddy server**
- **HTTP/2 (aka h2)**
- **Let's Encrypt + ACME**

Summary

- **Who Am I**
- **Caddy server**
- **HTTP/2 (aka h2)**
- **Let's Encrypt + ACME**
- **Internals: code!**

Summary

- **Who Am I**
- **Caddy server**
- **HTTP/2 (aka h2)**
- **Let's Encrypt + ACME**
- **Internals:** code!
- **Demo**

Summary

- **Who Am I**
- **Caddy server**
- **HTTP/2 (aka h2)**
- **Let's Encrypt + ACME**
- **Internals:** code!
- **Demo**
- **Future plans, Q&A ...**



Who Am I



Fernando Álvarez

- **Madrid->Berlin->Madrid**
- **Infrastructure @ BeBanjo**
- **Go, Bash, Ruby, Docker, Chef**
- **Ubuntu @ Server & Desktop**
- **@fern4lvarez**

Disclaimer



Caddy

<https://caddyserver.com>



POWER SERVER

Server Power	ENABLED
Status	OK
Data Path	OK
Access	OK
Backup	ENABLED

Facts

- **Web server written in Go**
- **Announced in April 2015**
- **Authored by Matt Holt, and more**
- **Active development, 0.8.2 as of now**
- **Releases for all main platforms**
- **Sustained by sponsors and donations**

Facts

- **Web server written in Go**
- **Announced in April 2015**
- **Authored by Matt Holt, and more**
- **Active development, 0.8.2 as of now**
- **Releases for all main platforms**
- **Sustained by sponsors and donations**
- ***NOT* a replacement of nginx, etc.**

Main features



POWER SERVER

Server Power	ENABLED
Status	OK
Data Rate	100 MB/s
Memory	16 GB
Backup	ENABLED

Main features

- Simple design, easy to use



Main features

- Simple design, easy to use
- Configured by project, not by system

Main features

- Simple design, easy to use
- Configured by project, not by system
- Extensible via “Directives”

Main features

- Simple design, easy to use
- Configured by project, not by system
- Extensible via “Directives”
- Uses HTTP/2 by default

Main features

- Simple design, easy to use
- Configured by project, not by system
- Extensible via “Directives”
- Uses HTTP/2 by default
- Enables TLS automatically

Main features

- Simple design, easy to use
- Configured by project, not by system
- Extensible via “Directives”
- Uses HTTP/2 by default
- Enables TLS automatically
- Support for Go Templates

Use cases

- **Serving static files**
- **Reverse proxy**
- **Load balancer**
- **TLS termination**
- **FastCGI server**
- **Blogging platform**

Use cases

“Caddy aims to cover the 90% of the priorities and workflows of web developers and site owners in 2016”

Matt Holt, creator of Caddy

Caddyfile

```
1 www.example.org  
2 basicauth / user password  
3 proxy / localhost:8080
```

Caddyfile

```
1 www.example.org
2 # new lines or comments are ignored
3 log /var/log/example/access.log
4 errors {
5     log /var/log/example/error.log
6     404 404.html # Not Found
7     500 500.html # Internal Server Error
8 }
9 ext .html .htm
10 proxy / localhost:8001 localhost:8001 localhost:8002 {
11     policy round_robin
12     health_check /health
13     proxy_header Host {host}
14     proxy_header X-Real-IP {remote}
15     proxy_header X-Forwarded-Proto {scheme}
16 }
17
```


Directives

basicauth, bind, browse, cors, errors, ext, fastcgi, git, gzip, header, hugo, import, internal, ipfilter, jsonp, log, mailout, markdown, mime, prometheus, proxy, redir, rewrite, root, search, shutdown, startup, templates, tls, websocket

The background of the slide is a dark, deep blue space filled with a complex network of glowing points and lines. The points, or nodes, are small spheres in various colors including white, light blue, and magenta. They are interconnected by thin, translucent lines that create a web-like structure. Some lines are more prominent than others, and the overall effect is one of a dynamic, interconnected digital space.

HTTP/2

<https://http2.github.io/>

History

- **HTTP/0.9 (1991)**
- **HTTP/1.0 (1996)**
- **HTTP/1.1 (1999)**
- **SPDY (2012)**
- **HTTP/2 (2015)**

What's new?

Compatible with HTTP/1.1

- HTTP/2 contains HTTP/1.1
- “It works!”

Better performance

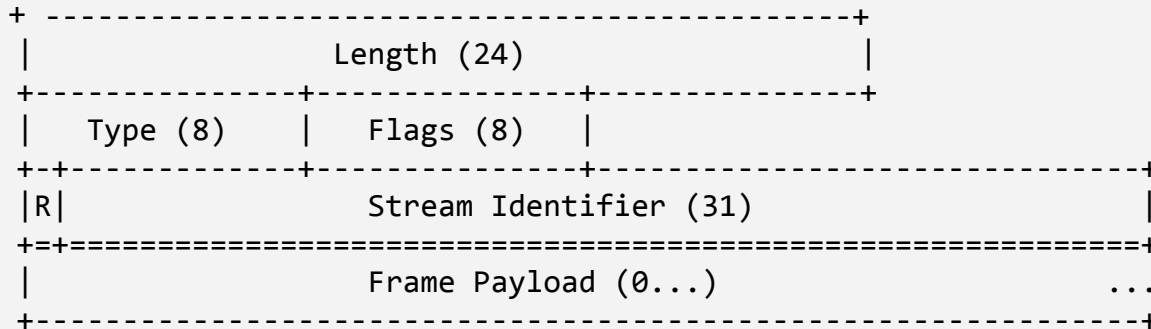
- **HTTP/1.1 uses 6 concurrent conn**
- **HTTP/2 muxes unlimited conn**

Better performance

- **HTTP/1.1 uses 6 concurrent conn**
- **HTTP/2 muxes unlimited conn**
- **Binary protocol**

Better performance

- HTTP/1.1 uses 6 concurrent conn
- HTTP/2 muxes unlimited conn
- Binary protocol
- Uses frames as entity on the wire



Better performance

- **HTTP/1.1 uses 6 concurrent conn**
- **HTTP/2 muxes unlimited conn**
- **Binary protocol**
- **Uses frames as entity on the wire**
- **Header compression using HPACK**

Better performance

- **HTTP/1.1 uses 6 concurrent conn**
- **HTTP/2 muxes unlimited conn**
- **Binary protocol**
- **Uses frames as entity on the wire**
- **Header compression using HPACK**
- **Server push**

The background of the slide is a dark blue and black space filled with a complex network of glowing white and light blue nodes connected by thin, translucent lines. The nodes are scattered across the frame, with some clusters and many isolated points, creating a sense of a vast, interconnected digital or physical network.

TLS only!

- **Fallbacks to HTTP/1.1 when no TLS**

HTTP/2 and Go

- github.com/bradfitz/http2
- golang.org/x/net/http2
- [net/http](https://golang.org/pkg/net/http/) (go1.6)

HTTP/2 and Go

- github.com/bradfitz/http2
- golang.org/x/net/http2
- [net/http](https://golang.org/pkg/net/http/) (go1.6)
- **Used by default! (by Caddy too)**

HTTP/2 and Go

- github.com/bradfitz/http2
- golang.org/x/net/http2
- [net/http](https://golang.org/pkg/net/http/) (go1.6)
- Used by default! (by Caddy too)
- Thank you, Brad Fitzpatrick!

A black and white photograph of a wooden door. The door has a vertical grain and is secured with a metal handle and lock mechanism. The handle is a long, cylindrical metal piece with a small, rounded knob at the end. It is mounted on a metal plate that is secured with several screws. The background is slightly blurred, focusing attention on the door hardware.

Let's Encrypt

<https://letsencrypt.org/>

No one likes...

- **Get certificates via impossible forms**
- **Receive validation emails**
- **.csr? .crt? bundle? chain?**
- **Complex server configuration**
- **Deal with expired certificates**
- **Pay overpriced costs**

Let's Encrypt

- Free Certificate Authority
- EFF, Mozilla, IdenTrust, Akamai...

Let's Encrypt

- **Free Certificate Authority**
- **EFF, Mozilla, IdenTrust, Akamai...**
- **Make certificates management easy**

Let's Encrypt

- **Free Certificate Authority**
- **EFF, Mozilla, IdenTrust, Akamai...**
- **Make certificates management easy**
- **Automate installation, renewal, etc.**

Let's Encrypt

- **Free Certificate Authority**
- **EFF, Mozilla, IdenTrust, Akamai...**
- **Make certificates management easy**
- **Automate installation, renewal, etc.**
- **Beta phase**

Let's Encrypt

- **Free Certificate Authority**
- **EFF, Mozilla, IdenTrust, Akamai...**
- **Make certificates management easy**
- **Automate installation, renewal, etc.**
- **Beta phase**
- **Certificates lifetime of 90 days**

Let's Encrypt

- **Free Certificate Authority**
- **EFF, Mozilla, IdenTrust, Akamai...**
- **Make certificates management easy**
- **Automate installation, renewal, etc.**
- **Beta phase**
- **Certificates lifetime of 90 days**
- **No support for wildcard certs**



ACME

**Automated Certificate Management
Environment**

<https://letsencrypt.github.io/acme-spec/>

ACME

- Protocol designed and used by LE
- Automate interactions with CA

ACME

- **Protocol designed and used by LE**
- **Automate interactions with CA**
 - **Domain validation**
 - **Certs installation, renewal, revocat.**

ACME

- **Protocol designed and used by LE**
- **Automate interactions with CA**
 - **Domain validation**
 - **Certs installation, renewal, revocat.**
- **REST API**

Caddy's Automatic HTTPS

- TLS enabled by default
- Let's Encrypt certificates

Caddy's Automatic HTTPS

- **TLS enabled by default**
- **Let's Encrypt certificates**
- **Speaks to ACME API**
- **Uses Sebastian Erhart's `lego/acme`**
 - **`github.com/xenolf/lego/acme`**

Criteria

- **Host is no localhost, no IP**
- **DNS correctly configured**
- **Feature not explicitly disable**
- **Binding to ports 80 and 443**
- **Port 80 not used**

Caddy vs nginx

<http://www.cyberciti.biz/faq/how-to-configure-nginx-with-free-lets-encrypt-ssl-certificate-on-debian-or-ubuntu-linux/>

Caddy vs nginx

```
⇒ echo my.host.name > Caddyfile  
⇒ caddy -agree -email=my@host.name
```


Internals

- github.com/mholt/caddy



Internals

- github.com/mholt/caddy
- Server startup + TLS activation

Internals

- github.com/mholt/caddy
- Server startup + TLS activation
- Directives (Middleware)

Internals

- github.com/mholt/caddy
- Server startup + TLS activation
- Directives (Middleware)
- Web server

Demo



Future plans

- **Self-signed certificates**
- **Limit of conn per upstream on proxy**
- **Change config via API**

Future plans

- Self-signed certificates
- Limit of conn per upstream on proxy
- Change config via API
- Ask Caddy on Twitter! @caddyserver

A nighttime photograph of a cityscape. In the foreground, a multi-lane highway with a concrete overpass is visible, with cars and streetlights. To the right, a dark, wooded area slopes down towards a body of water. In the background, several tall city buildings are lit up, and a bridge with lights is visible in the distance across the water. The sky is a dark, hazy blue.

Thank you
Questions?