

Running Streamlit on Amazon EC2 with HTTPS



Steven Munn · [Follow](#)

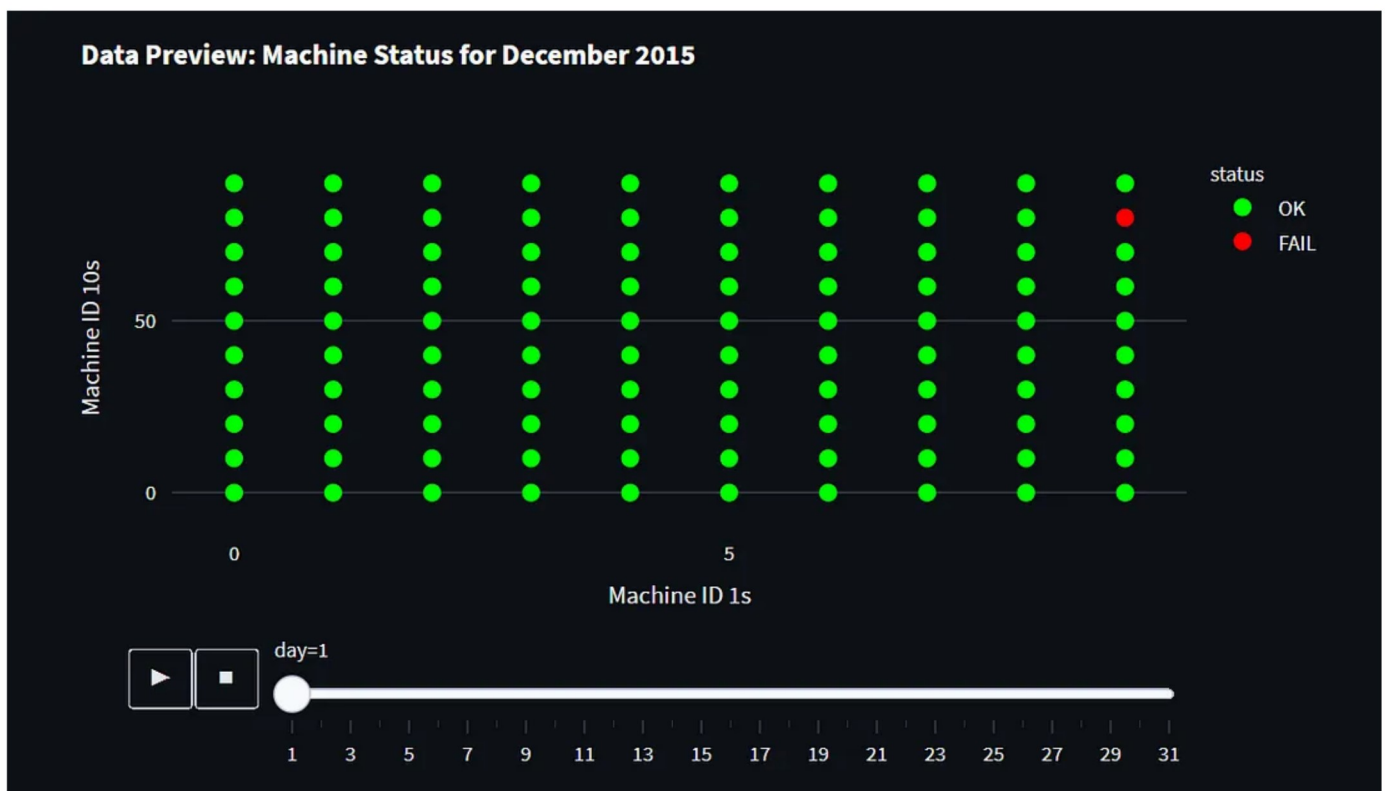
3 min read · Jun 28, 2023



5



2



My streamlit app <https://streamlit.bigmunny.com/>

After building and deploying an [streamlit](#) app to [Amazon EC2](#), I decided to share a quick write up with some challenges I encountered in case someone may find this useful. *NOTE: These instructions are for an Amazon EC2 t2.micro instance with Amazon Linux 2.*

First, I will talk about which instance type to chose. Second, I will explain how to setup streamlit as a system service (far more stable than using something like screen). Third, I will mention some tips for obtaining CA signed certificates. Finally, I will share my reverse proxy setup to enable https connections.

Instance Types

Streamlit seems to be quite RAM intensive. I have had issues with running the server on t2.nano instances. This may be possible with further optimization for lightweight apps, but I recommend provisioning at least a t2.micro instance at least initially. t2.micro instances are required to be EBS-back, so do not worry. If it's too much or too little compute, you can change it later just by restarting the instance (see [here](#)).

Running streamlit as a System Service

These instructions/types are valid and tested on my Amazon Linux 2 t2.micro EC2 instance only. Instructions for AL2023 might be a bit different in practice, but similar in spirit.

The biggest problem with running streamlit in the terminal or with screens is that if the server crashes, it will not restart automatically.

Running streamlit as a system service is the best solution I have found. To do this, run,

```
$ cd /etc/systemd/system  
$ emacs streamlit.service # pick text editor of choice..
```

and fill the file with,

```
[Unit]
Description=Streamlit
After=network-online.target

[Service]
User=ec2-user
Group=ec2-user
Restart=always
ExecStart=/path/to/python/virtualenv/bin/streamlit run path/to/repo/Home.py

[Install]
WantedBy=multi-user.target
```

This specifies the commands to run the server and it also requires the system to restart the app if it crashes.

After saving the file, kill all other streamlit instances and execute the following:

```
$ sudo systemctl enable streamlit.service
$ sudo systemctl start streamlit
$ sudo systemctl status streamlit -l
```

This will start the service and show its status. If there are any issues, try

```
$ sudo journalctl -xe
```

to get the logs for the service.

Obtaining CA Signed Certificates

Please note, although I think it is possible to obtain signed certificates

without a domain name (e.g. www.bigmunny.com), this will not be straightforward. I recommend looking into nip.io and seeing if [LetEncrypt](https://letsencrypt.org/) will not hit you with a rate limit for nip.io.

If you do have a domain, then this process will use certbot to obtain certificates from [Letsencrypt](https://letsencrypt.org/).

Instructions for installing certbot can be confusing. For Amazon Linux 2, I have come to the conclusion that installing via python and pip is the only viable method:

```
# Create a virtualenv only for certbot
$ sudo python3 -m venv /opt/certbot/
# Upgrade pip
$ sudo /opt/certbot/bin/pip install --upgrade pip
# Install certbot
$ sudo /opt/certbot/bin/pip install certbot
# link certbot so you can run it anywhere
$ sudo ln -s /opt/certbot/bin/certbot /usr/bin/certbot
# Get the certificates
$ sudo certbot certonly --standalone -d <your-domain>
```

Setting up nginx for HTTPS

First of all, I strongly recommend using nginx as the reverse proxy rather than the Apache server. Support of websockets was much easier with nginx, and I have never quite been able to get everything to work with httpd (although, I suspect it is possible).

To setup nginx on Amazon Linux 2, I ran,

```
# Install nginx
$ sudo amazon-linux-extras install nginx1
# Start the service
$ sudo systemctl enable nginx
$ sudo systemctl start nginx
# Go to the config file directory
$ cd /etc/nginx/
```

```
# Edit the config file (chose your favorite text editor..)
$ emacs nginx.conf
```

This is a fairly large file, but the structure will look something like,

```
user nginx;
# stuff

events {
    # more stuff
}

http {
    server {
        # we will change this to 443 for ssl
        listen 80;
        # ...
        error_page 404 /404.html;
        location = /404.html {
        }

        error_page 500 502 503 504 /50x.html;
        location = /50x.html {
        }
        # <--- here is where we want to add our config
    }
}
```

The github gist below shows the config file I used,

```
1  user nginx;
2  worker_processes auto;
3  error_log /var/log/nginx/error.log;
4  pid /run/nginx.pid;
5
6  # Load dynamic modules. See /usr/share/doc/nginx/README.dynamic.
7  include /usr/share/nginx/modules/*.conf;
8
9  events {
10     worker_connections 1024;
11 }
12
13 http {
14     log_format main '$remote_addr - $remote_user [$time_local] "$request" '
15                     '$status $body_bytes_sent "$http_referer" '
16                     '"$http_user_agent" "$http_x_forwarded_for"';
17
18     access_log /var/log/nginx/access.log main;
19
20     sendfile on;
21     tcp_nopush on;
22     tcp_nodelay on;
23     keepalive_timeout 65;
24     types_hash_max_size 4096;
25
26     include /etc/nginx/mime.types;
27     default_type application/octet-stream;
28
29     # Load modular configuration files from the /etc/nginx/conf.d directory.
30     # See http://nginx.org/en/docs/nginx_core_module.html#include
31     # for more information.
32     include /etc/nginx/conf.d/*.conf;
33
34     server {
35         listen 80;
36         return 301 https://$host$request_uri;
37     }
38
39     server {
40         listen 443 ssl;
41         server_name # server domain name #;
42         root /usr/share/nginx/html;
43
44         ssl_certificate # certificate #;
45         ssl_certificate_key # key #;
46
47
48     # Load configuration files for the default server block.
```

```

49     include /etc/nginx/default.d/*.conf;
50
51     error_page 404 /404.html;
52     location = /404.html {
53     }
54
55     error_page 500 502 503 504 /50x.html;
56     location = /50x.html {
57     }
58     location / {
59         proxy_ssl_certificate    #CA pem#;
60         proxy_ssl_certificate_key    # key #;
61
62         proxy_pass http://localhost:8501/;
63         proxy_set_header    Host    $host;
64         proxy_set_header    X-Real-IP $remote_addr;
65         proxy_set_header    X-Forwarded-For $proxy_add_x_forwarded_for;
66         proxy_set_header    X-Forwarded-Proto $scheme;
67         proxy_buffering    off;
68         proxy_http_version 1.1;
69         # Also requires websocket:
70         proxy_set_header Upgrade $http_upgrade;
71         proxy_set_header Connection "upgrade";
72         proxy_read_timeout 86400;
73     }
74 }
75 }

```

nginx.conf hosted with ❤ by GitHub

[view raw](#)

Conclusion

Please let me know if you have any questions, comments, or concerns about this setup.

Streamlit App Deployment

Aws Ec2

Nginx

Streamlit



Written by Steven Munn

Follow



2 Followers

Machine Learning Software Engineer in the San Francisco Bay Area 🇺🇸 Experience across industries such as manufacturing, aviation, automation 🧰

Recommended from Medium



Manyi

Hosting Streamlit App on AWS EC2

If you have created a Streamlit app and like to deploy it on AWS, for your own use or for...

★ Aug 3 🖱 6



Christopher Adamson

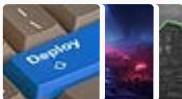
Implementing Session Affinity with ALB Sticky Sessions

In web applications, maintaining client-server session affinity is crucial for ensuring a...

★ Jul 28 🖱 5



Lists



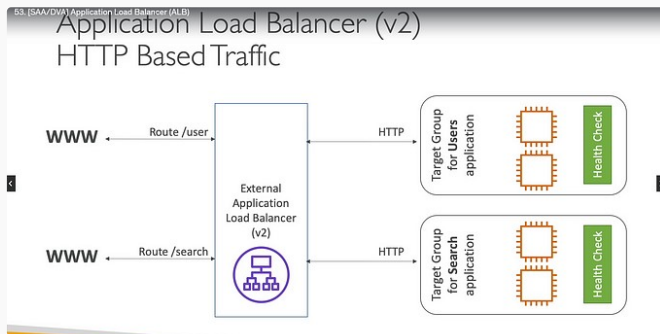
Predictive Modeling w/ Python


20 stories · 1622 saves



Natural Language Processing

1780 stories · 1386 saves




 Nagarjun Nagesh

Analyzing the Architecture of Application Load Balancer (v2) fo...

In this blog, we will delve into the architecture depicted in the provided image, which...

★ Jul 9 🖱 6 📖 + ...

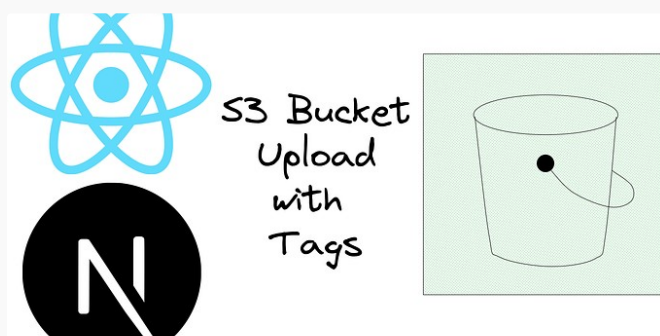


 XeusNguyen

Create Atlantis with ECS Fargate for automation provision...

Hi @all, this week is so busy for me, and I am on my way to learning something new about...

★ Aug 9 🖱 1 📖 + ...




 Peng Cao in AWS Tip

Streamlined AWS S3 File Uploads with Tagging via Pre-signed URLs...

Uploading files to AWS S3 is crucial for many web applications, but it can be complex for...

★ May 13 🖱 8 📖 + ...



 Benjamin Franklin. S

Streamlit with Nginx: A Step-by-Step Guide to Setting up Your Dat...

Streamlit is an open-source app framework that is perfect for machine learning and data...

Jun 29 🖱 1 📖 + ...

See more recommendations