server {

listen 80 default\_server;

#listen 443 ssl default\_server;

server\_name ~^(?<subdomain>.+)\.mubasher\.info$;

#root /var/www/nginx-afzal.com/html;

root /var/www/html/$subdomain/;

index index.html index.htm index.nginx-debian.html;

server\_name \_;

location / {

try\_files $uri $uri/ =404;

}

}

last part of the URL:

if ($request\_uri ~\* "([^/]\*$)" ) {

set $last\_path\_component $1;

}

You can then use the $last\_path\_component var anywhere you wish. Note this regex will return everything after the last / which could include in url arguments, so if you do not need them, you will have to modify the regex above accordingly.

Use regular expression submatches --

rewrite ^(.\*) http://host.example.com$1 permanent;

With [regular expressions](https://en.wikipedia.org/wiki/Regular_expression), the parentheses are [metacharacters](https://en.wikipedia.org/wiki/Metacharacter) that groups text into capture blocks. The first parenthesis will be grouped into $1, the second into $2, and so on.

For the example in the answer it would be more appropriate to use

$request\_uri i.e.:rewrite ^ http://example.com$request\_uri permanent; or better still location /whatever { return http://example.com$request\_uri; }.

server {

listen 80;

server\_name \*.mubasher.info ;

return 301 https://$http\_host$request\_uri$is\_args$args;

}

server {

listen 443 default\_server ssl;

server\_name ~^(?<subdomain>.+)\.fnn\.english\.mubasher\.com$;

server\_name ~^www\d+\.example\.net$;

server\_name fnn.english.mubasher.info almasryalyoum.english.mubasher.info masrawy.english.mubasher.info;

ssl on;

ssl\_certificate /etc/ssl/certs/example.com.crt;

ssl\_certificate\_key /etc/ssl/private/example.key;

root /var/www/html/$subdomain/;

}

provide subdomin as index as html

include /var/nginx/general/php;

include /var/nginx/general/upload;

include /var/nginx/general/error\_page\_50x;

}

The regular expressions used by nginx are compatible with those used by the Perl programming language

(PCRE). To use a regular expression, the server name must start with the tilde character:

server\_name ~^www\d+\.example\.net$;

otherwise it will be treated as an exact name, or if the expression contains an asterisk, as a

wildcard name (and most likely as an invalid one). Do not forget to set “^” and “$” anchors.

They are not required syntactically, but logically. Also note that domain name dots should be

escaped with a backslash. A regular expression containing the characters “{” and “}” should be quoted:

server\_name "~^(?<name>\w\d**{**1,3**}**+)\.example\.net$";

otherwise nginx will fail to start and display the error message:

directive "server\_name" is not terminated by ";" in ...

A named regular expression capture can be used later as a variable:

server {

server\_name ~^(www\.)?(**?<domain>**.+)$;

location / {

root /sites/**$domain**;

}

}

server {

server\_name ~^(www\.)?(.+)$;

location / {

root /sites/**$2**;

}

}

server\_name ~^(?<name>[\w-]+)\.example\.com$;

server\_name ~^www\d+\.example\.net$;

server {

server\_name ~^(?<name>\w+)\.example\.com$;

location /admin {

return 301 $scheme://$name.myurl.com/;

}

}

## Answer #1:

The answer is little bit simpler than that. Just get the substring with the subdomain and use it as a parameter for proxy\_pass:

server {

# this matches every subdomain of domain.

server\_name .domain;

location / {

set $new\_request\_uri "";

set $subdomain "";

if ($host ~\* "^(.+).domain$") {

set $subdomain $1;

# lets assume there are args...

set $new\_request\_uri "$request\_uri&subdomain=$subdomain";

}

# if there are no args add a question mark and the subdomain argument

if ($args = '') {

set $new\_request\_uri "$request\_uri?subdomain=$subdomain";

}

proxy\_pass http://127.0.0.1:8080$new\_request\_uri;

}

}

map $host $subdomain {

~^(?<sub>.+)\.[^\.]+\.[^\.]+$ $sub;

}

server {

listen 80 default\_server;

server\_name \_;

location / {

if ($subdomain) {

proxy\_pass http://$subdomain;

}

}

}

server {

charset utf8;

listen 80;

server\_name

domain.com,

sub1.domain.com,

sub2.domain.com,

sub3.domain.com,

sub4.domain.com,

sub5.domain.com;

# Default

if ($host ~ ^domain\.com) {

set $proxy\_uri $request\_uri;

}

# Rewrites

if ($host ~ (.\*)\.domain\.com) {

set $proxy\_uri $1$request\_uri;

}

location / {

expires 1s;

proxy\_pass http://node:8080$proxy\_uri; #node is an internally listed host (docker container)

proxy\_set\_header Host domain.com;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

proxy\_cache\_valid 200 1s;

}

}

upstream backend {

server node:8080;

keepalive 8;

}

I also had to slightly modify the proxy pass line to the following:

proxy\_pass http://backend$proxy\_uri;