

---

## Nginx: HTTP and HTTPS on a single port

Let's say you have a website `www.site.com`. That site sits on a server that you connect to with a client - your browser. The browser communicates with the client on a protocol, in this case it's HTTP. But in nowadays internet communication in plain HTTP (which is not secured at all) is abandoned. Everyone use HTTPS which is HTTP with SSL layer so the communication is encrypted.

Each protocol uses a different port - HTTP uses port 80, HTTPS port 443. Since HTTP is abandoned on production use (but it's good enough for local development where you don't need encryption) here comes the case where you want any request that comes over HTTP to your site redirect to HTTPS. This is just a simple redirect with a separated server directive

```
server {  
    listen 80;  
    server_name www.site.com;  
    return 301 https://$host$request_uri;  
}
```

Now you have 2 access points (HTTP on port 80 + HTTPS on port 443) where one is just an alias to another. This also means that your server has to be open to listen on those 2 ports which is quite a standard.

Now imagine you have a testing server where you stack up your sites. Each site is complex and comes with it's own proxy (like Docker application with bundled Nginx). In such case you cannot put all those proxies from all those sites on the same ports 80 or 443. The solution is to reserve a set of ports for each site and bind them to docker proxy to port 80 and 443. So app1 will sit on port 2000 and 2001 which is gonna be binded to 80 or 443 respectively. App2 will sit on ports 2002 and 2003. This means your docker compose for each of the sites will looks like:

```
services:  
  # ...  
  nginx:  
    # ...  
    ports:  
      - 2000:80  
      - 2001:443
```

Now each site blocks 2 ports on your server and you use pretty much the same code (see above in first code example) in each proxy config. If you drop the extra config from proxy, remove port binding for port 80 and try to access your site on `http://site.com:2001` you get the "The plain HTTP request was sent to HTTPS port" error:

---

# 400 Bad Request

The plain HTTP request was sent to HTTPS port

---

nginx

**Figure 1:** nginx 400 error

This error means that your client send an unencrypted HTTP request which landed on proxy that expects encrypted HTTPS request. Basically you mix those two protocols together.

So the question here is how to handle both HTTP and HTTPS on a single port? Is that even possible? Well it's not possible to mix up the communication but you can simulate the original idea with HTTP -> HTTPS redirect in a nifty way. Luckily Nginx has a smart solution and that's custom error code 497 which can be handled with `[error_code]` ([http://nginx.org/en/docs/http/nginx\\_http\\_core\\_module.html](http://nginx.org/en/docs/http/nginx_http_core_module.html)) directive. If we add a redirect with a proper code and an URL we end up with the same effect that handles our original ( lately duplicate) configuration:

```
error_page 497 301 =307 https://$host:2001$request_uri;
```

This code catches 497 and 301 codes and replaces them with 307 code and URL where client is redirected to. Note the variables that are used and also the hard-coded port number. You cannot use `$server_port` here, because that will be 443 since the incoming request comes from 2001 port (on the machine) to 443 port (on the proxy).