

## 京峰教育 第十次作業 – 0421 課程

作業 1：設置訪問控制, 只允許本機查看 nginx 的 status 狀態信息, 其他人均拒絕。

1.1、默認 nginx 並無編譯 nginx 的 http\_stub\_status\_module 狀態模塊, 查看編譯參數 ./configure --help | grep status 如下圖一(a), 圖 x(b)

```
[root@NGINX-170 nginx-1.16.1]# ./configure --help | grep status
--with-http_stub_status_module      enable ngx_http_stub_status_module
```

圖一(a)、http\_stub\_status\_module 默認無編譯

```
[root@NGINX-170 nginx-1.16.1]# nginx -V
nginx version: JFWS/1.1
built by gcc 4.8.5 20150623 (Red Hat 4.8.5-39) (GCC)
configure arguments: --prefix=/usr/local/nginx
```

圖一(b)、編譯時並無編譯 http\_stub\_status\_module 模塊

1.2、熱部署nginx的status模塊, 重新預編譯nginx參數, 使用./configure --prefix=/usr/local/nginx --with-http\_stub\_status\_module, 增加--with-http\_stub\_status\_module參數, 如下圖二

```
[root@NGINX-170 nginx-1.16.1]# ./configure --prefix=/usr/local/nginx --with-http_stub_status_module
```

圖二、./configure預編譯加入http\_stub\_status\_module模塊

1.3、編譯並安裝, 如圖三

```
[root@NGINX-170 nginx-1.16.1]# make && make install
```

圖三、編譯並安裝

1.4、直接進行更新, 但執行完命令, 產生錯誤, 如圖四

```
[root@NGINX-170 nginx-1.16.1]# make upgrade
/usr/local/nginx/sbin/nginx -t
nginx: the configuration file /usr/local/nginx/conf/nginx.conf syntax is ok
nginx: configuration file /usr/local/nginx/conf/nginx.conf test is successful
kill -USR2 `cat /usr/local/nginx/logs/nginx.pid`
sleep 1
test -f /usr/local/nginx/logs/nginx.pid.oldbin
make: *** [upgrade] Error 1
```

圖四、直接更新錯誤

1.5、實際查看Makefile文件,發現執行make upgrade命令,最後一個命令並沒有執行,因為在檢查/usr/local/nginx/logs/nginx.pid.oldbin沒有存在,如圖五

```
upgrade:
    /usr/local/nginx/sbin/nginx -t

    kill -USR2 `cat /usr/local/nginx/logs/nginx.pid`
    sleep 1
    test -f /usr/local/nginx/logs/nginx.pid.oldbin

    kill -QUIT `cat /usr/local/nginx/logs/nginx.pid.oldbin`
```

圖五、查看Makefile文件

1.6、嘗試手動更新也依舊失敗, kill -USR2 `cat /usr/local/nginx/logs/nginx.pid`, 也沒有拉起一個master新進程, 查看/usr/local/nginx/logs/error.log發現啟動nginx失敗, 是因為使用環境變量啟動nginx, 並非絕對路徑啟動nginx, 如圖六

10690#0: execve() failed while executing new binary process "nginx" (2: No such file or directory)

这告诉我们, 找不到nginx这个命令。这是因为我们一开始启动时就是依赖的是环境变量, 但这时候nginx需要的是一个绝对的路径来找到nginx命令所在, 所以出错了。

解决方法是: 启动时要使用绝对路径。如:

```
1. /usr/bin/nginx -c /etc/nginx.conf
```

圖六、錯誤原因

1.7、在使用絕對路徑啟動nginx後, 用kill -USR2 `cat /usr/local/nginx/logs/nginx.pid`啟動新的master進程, 如圖七

```
[root@NGINX-170 nginx-1.16.1]# kill -USR2 `cat /usr/local/nginx/logs/nginx.pid`
[root@NGINX-170 nginx-1.16.1]# ps -ef | grep nginx
root      7015   6941   0 08:51 pts/1    00:00:00 tailf /usr/local/nginx/logs/error.log
root      7597     1   0 08:52 ?        00:00:00 nginx: master process /usr/local/nginx/sbin/nginx
nginx     7598   7597   0 08:52 ?        00:00:00 nginx: worker process
root     11196   7597   0 08:56 ?        00:00:00 nginx: master process /usr/local/nginx/sbin/nginx
nginx     11197  11196   0 08:56 ?        00:00:00 nginx: worker process
root     11226   6378   0 08:56 pts/0    00:00:00 grep --color=auto nginx
```

圖七、啟動一個新進程

## 1.8、優雅退出老進程, 使用命令kill -WINCH 7597, 如圖八

```
[root@NGINX-170 nginx-1.16.1]# kill -WINCH 7597
[root@NGINX-170 nginx-1.16.1]# ps -ef | grep nginx
root      7015   6941   0 08:51 pts/1    00:00:00 tailf /usr/local/nginx/logs/error.log
root      7597     1   0 08:52 ?        00:00:00 nginx: master process /usr/local/nginx/sbin/nginx
root     11196   7597   0 08:56 ?        00:00:00 nginx: master process /usr/local/nginx/sbin/nginx
nginx     11197  11196   0 08:56 ?        00:00:00 nginx: worker process
root     12016   6378   0 08:59 pts/0    00:00:00 grep --color=auto nginx
```

圖八、優雅退出老進程

## 1.9、清除掉舊的進程, 使用命令 kill -QUIT 7597, 完成更新, 如圖九(a)、圖九(b)

```
[root@NGINX-170 nginx-1.16.1]# kill -QUIT 7597
[root@NGINX-170 nginx-1.16.1]# ps -ef | grep nginx
root     11196     1   0 08:56 ?        00:00:00 nginx: master process /usr/local/nginx/sbin/nginx
nginx    11197  11196   0 08:56 ?        00:00:00 nginx: worker process
root     13227   6378   0 09:02 pts/0    00:00:00 grep --color=auto nginx
```

圖九(a)、清除掉舊的 master 進程

```
[root@NGINX-170 nginx-1.16.1]# /usr/local/nginx/sbin/nginx -V
nginx version: JFWS/1.1
built by gcc 4.8.5 20150623 (Red Hat 4.8.5-39) (GCC)
configure arguments: --prefix=/usr/local/nginx --with-http_stub_status_module
```

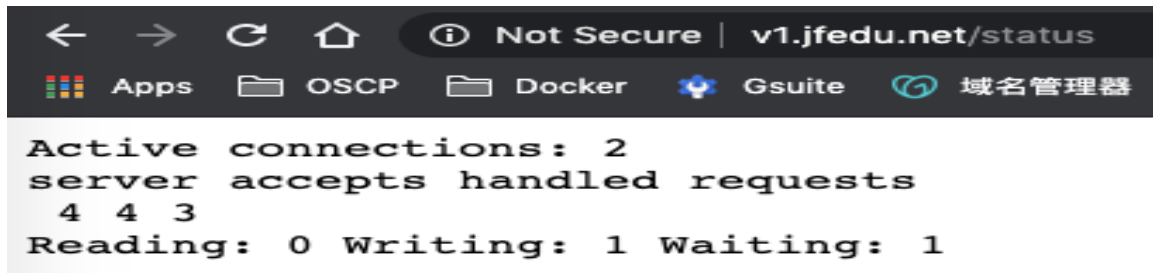
圖九(b)、查看已編譯 http\_stub\_status\_module 模塊

## 1.10、配置/usr/local/nginx/conf/vhost/v1.jfedu.net.conf 內 location 模塊, 當 URI 匹配到/status 的時候, 可以看到 nginx 的接受連線的狀態, 如圖十(a), 圖十(b)

```
server {
    listen      80;
    server_name v1.jfedu.net;
    access_log  logs/v1.access.log  init-log;
    error_log   logs/v1.error.log;
    #allow 127.0.0.1/32;
    #deny all;
    location / {
        root    html/v1;
        index   index.html index.htm;
    }

    location /status {
        stub_status;
    }
    error_page   500 502 503 504   /50x.html;
    location = /50x.html {
        root    html;
    }
}
```

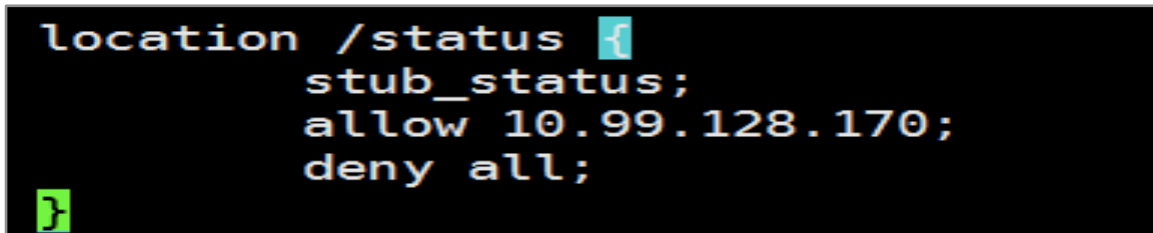
圖十(a)、/status 匹配 URI



```
Active connections: 2
server accepts handled requests
 4 4 3
Reading: 0 Writing: 1 Waiting: 1
```

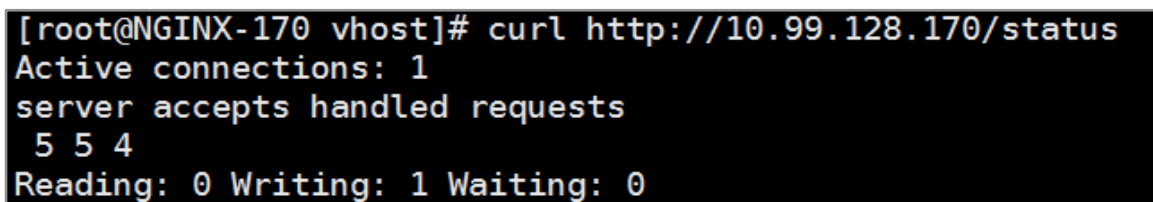
圖十(b)、nginx 的連線狀態

1.11、使用 allow 命令允許只有本機 IP 可以查看 nginx status, 其他 IP 皆禁止查看 status, 如圖十一(a)、圖十一(b)、圖十一(c)



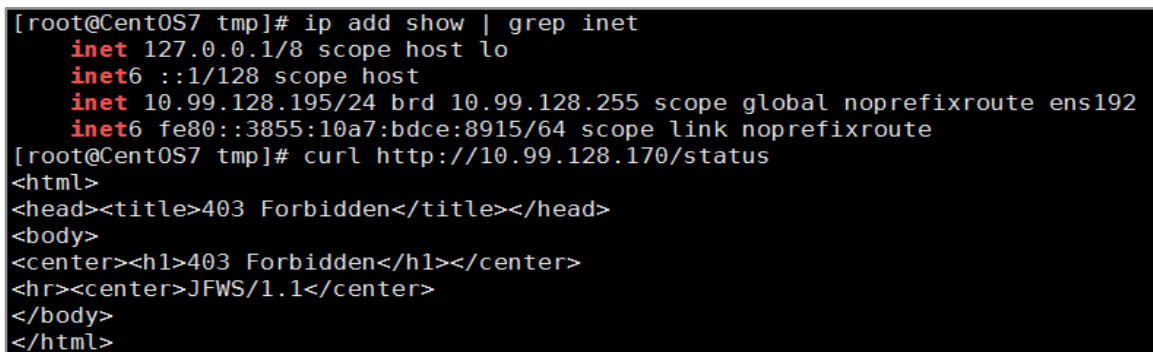
```
location /status {
    stub_status;
    allow 10.99.128.170;
    deny all;
}
```

圖十一(a)、允許本機查看 nginx status



```
[root@NGINX-170 vhost]# curl http://10.99.128.170/status
Active connections: 1
server accepts handled requests
 5 5 4
Reading: 0 Writing: 1 Waiting: 0
```

圖十一(b)、本機(10.99.128.170)查看 nginx status



```
[root@CentOS7 tmp]# ip add show | grep inet
inet 127.0.0.1/8 scope host lo
inet6 ::1/128 scope host
inet 10.99.128.195/24 brd 10.99.128.255 scope global noprefixroute ens192
inet6 fe80::3855:10a7:bdce:8915/64 scope link noprefixroute
[root@CentOS7 tmp]# curl http://10.99.128.170/status
<html>
<head><title>403 Forbidden</title></head>
<body>
<center><h1>403 Forbidden</h1></center>
<hr><center>JFWS/1.1</center>
</body>
</html>
```

圖十一(c)、非本機(10.99.128.195)查看 nginx status 返回 403

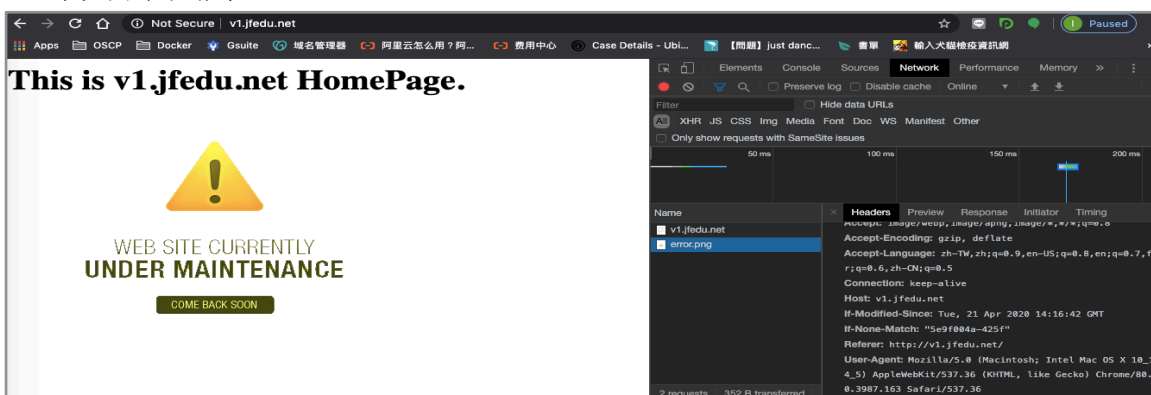
作業 2：設置防盜鏈，截圖驗證設置成功(出現圖裂)。

2.1、配置/usr/local/nginx/conf/vhost/v1.jfedu.net.conf 內新增防盜鏈模塊，可以利用命令 `valid_referers none blocked url` 實現防盜鏈，如果不是合法的 url，則返回 403，如圖十二，其中 `none` 表示：referer 值為空，`blocked` 表示：由防火牆進行偽裝，`url` 表示：接受該 url 過來的 referer，實際在使用的時候 `none`，`blocked` 都可以拿掉，剩下要允許的域名即可

```
location ~* \.(png|jpg)$ {  
    root html/v1;  
    valid_referers none blocked v1.jfedu.net;  
    if ($invalid_referer) {  
        return 403;  
    }  
}
```

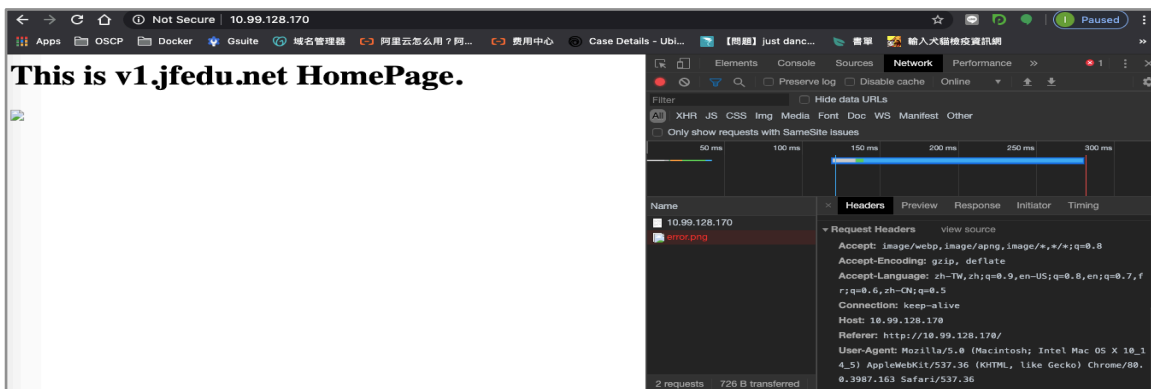
圖十二、配置防盜鏈

2.2、實際由 v1.jfedu.net 域名訪問，如圖十三，所帶的 referer:http://v1.jfedu.net，成功顯示圖片



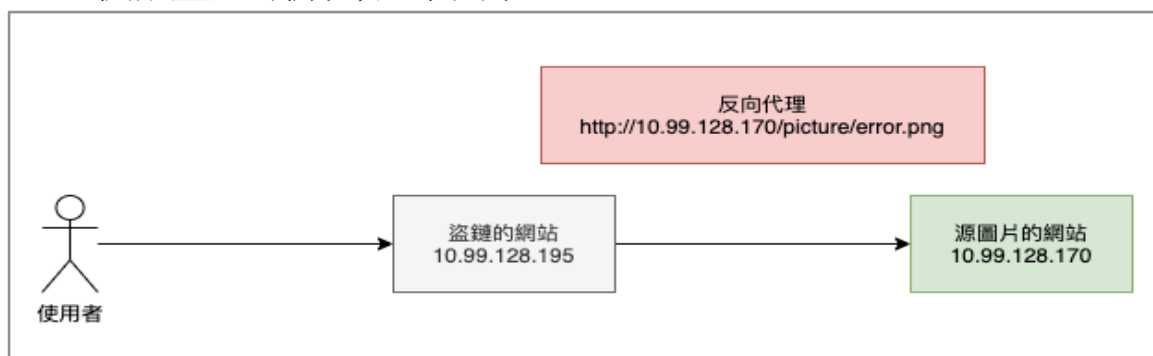
圖十三、由 v1.jfedu.net 成功訪問圖片

2.3、由 ip 進行訪問，如圖十四，所帶的 referer:http://10.99.128.170，無法獲得圖片，產生圖裂



圖十四、由 ip 進行訪問圖片，產生圖裂

## 2.4、模擬盜鏈的情境, 如下圖十五



圖十五、盜鏈情境

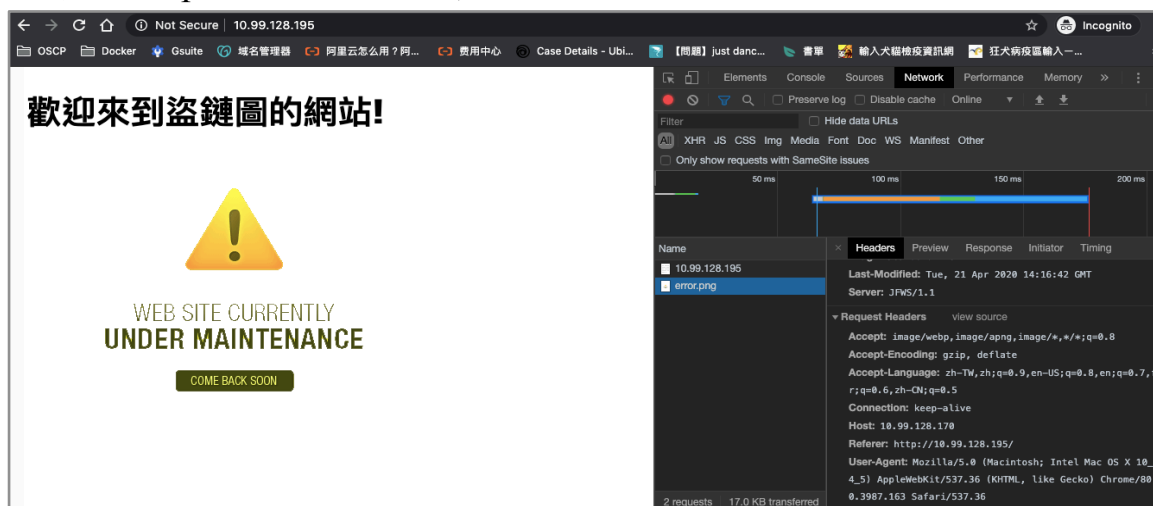
## 2.5、在準備盜鏈的 nginx 服務器上編寫 html 代碼, 引用 http://10.99.128.170 的圖片, , 如圖十六

```
<!DOCTYPE html>
<html>
<head>
<title>盜鏈網站!</title>
<style>
  body {
    width: 35em;
    margin: 0 auto;
    font-family: Tahoma, Verdana, Arial, sans-serif;
  }
</style>
</head>
<body>
<h1>歡迎來到盜鏈圖的網站!</h1>

</body>
</html>
```

圖十六、盜鏈服務器盜鏈圖片

## 2.6、實際訪問盜鏈的網頁 10.99.128.195, 注意到 request Headers 內的 Referer 字段為 http://10.99.128.195/, 如圖十七



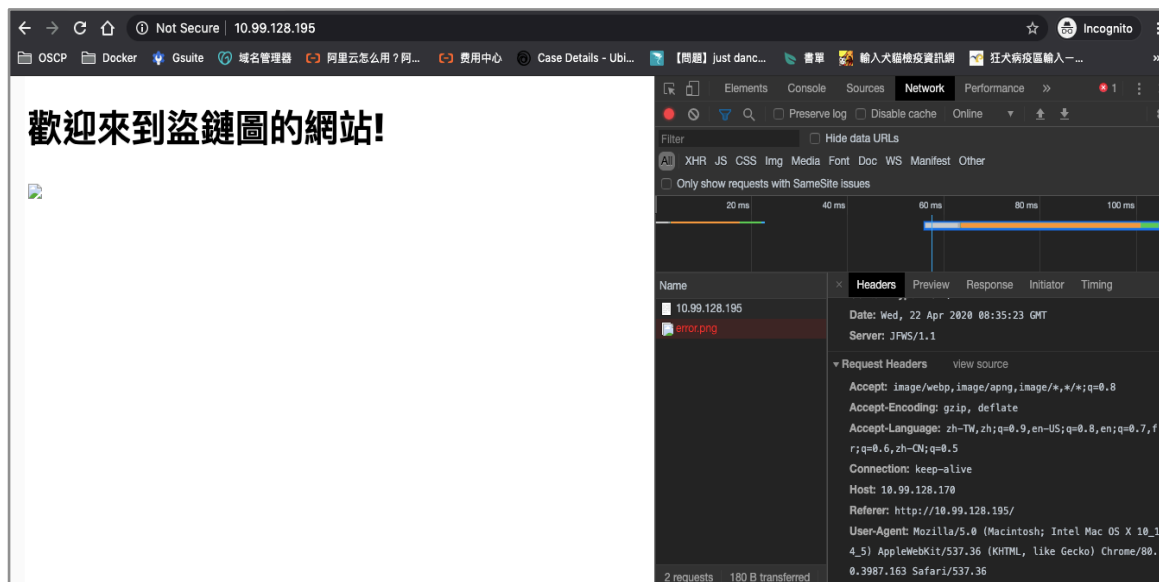
圖十七、訪問盜鏈網頁

2.7、前往圖片源服務器,更改/usr/local/nginx/conf/vhost/v1.jfedu.net.conf 配置,新增防盜鏈模塊,並只允許 v1.jfedu.net 域名過來的 request 允許存取.png or .jpg 圖片,如果不是該域名過來訪問圖片的,則返回 403 錯誤,如圖十八

```
location ~* \.(png|jpg)$ {
    root html/v1;
    valid_referers none blocked v1.jfedu.net;
    if ($invalid_referer) {
        return 403;
    }
}
```

圖十八、配置防盜鏈

2.8、再到盜鏈的網頁上訪問,發現已無法正常存取圖片,產生圖裂



圖十九、防盜鏈成功,產生圖裂

2.9、查看源服務器日誌, /usr/local/nginx/logs/v1.access.log, 直接返回 403, 如圖二十

```
10.4.9.202 - - [22/Apr/2020:04:38:04 -0400] "GET /picture/error.png HTTP/1.1" 403 551 "http://10.99.128.195/" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.163 Safari/537.36" "-"
10.4.9.202 - - [22/Apr/2020:04:38:13 -0400] "GET /picture/error.png HTTP/1.1" 403 551 "http://10.99.128.195/" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.163 Safari/537.36" "-"
```

圖二十、查看日誌/usr/local/nginx/logs/v1.access.log