

## Title: - Security Misconfiguration (Cross origin Resource Sharing)

**Description:** - The Access-Control-Allow-Origin header is included in the response from one website to a request originating from another website, and identifies the permitted origin of the request. A web browser compares the Access-Control-Allow-Origin with the requesting website's origin and permits access to the response if they match.

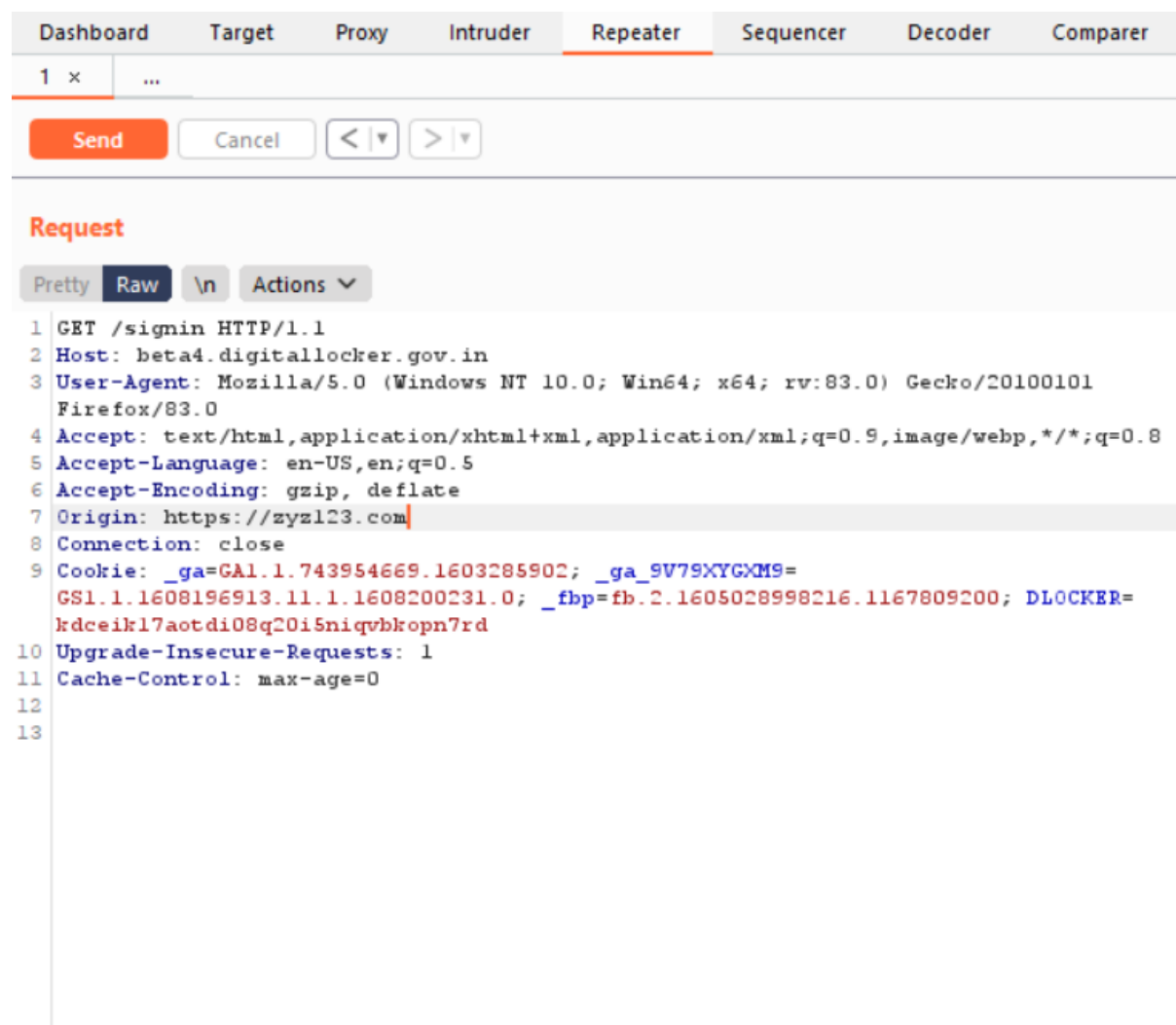
The header Access-Control-Allow-Origin supports wildcards. For example:

Access-Control-Allow-Origin: \*

**Note:** - wildcards cannot be used within any other value. For example, the following header is not valid:

Access-Control-Allow-Origin: [https://\\*.normal-website.com](https://*.normal-website.com)

### Request



The screenshot shows a web security tool interface with tabs for Dashboard, Target, Proxy, Intruder, Repeater, Sequencer, Decoder, and Comparer. The Repeater tab is active. Below the tabs, there is a 'Send' button and a 'Cancel' button. The main area displays a request in raw format, with the 'Raw' tab selected. The request is as follows:

```
1 GET /signin HTTP/1.1
2 Host: beta4.digitallocker.gov.in
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:83.0) Gecko/20100101
  Firefox/83.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Origin: https://zyzl23.com
8 Connection: close
9 Cookie: __ga=GA1.1.743954669.1603285902; __ga_9V79XYGXM9=
  GS1.1.1608196913.11.1.1608200231.0; _fbp=fb.2.1605028998216.1167809200; DLOCKER=
  kdceik17aotdi08q20i5niqvbkopn7rd
10 Upgrade-Insecure-Requests: 1
11 Cache-Control: max-age=0
12
13
```

### Response

```
Response
Pretty Raw Render \n Actions v
1 HTTP/1.1 200 OK
2 server: envoy
3 date: Thu, 17 Dec 2020 10:17:46 GMT
4 content-type: text/html; charset=UTF-8
5 vary: Accept-Encoding
6 expires: Thu, 19 Nov 1981 08:52:00 GMT
7 cache-control: no-store, no-cache, must-revalidate
8 pragma: no-cache
9 x-frame-options: SAMEORIGIN
10 x-xss-protection: 1; mode=block
11 x-content-type-options: nosniff
12 strict-transport-security: max-age=31536000; env=HTTPS;
13 access-control-allow-origin: *
14 access-control-allow-methods: POST, GET, OPTIONS, DELETE, PUT
15 access-control-allow-headers: x-requested-with, Content-Type, origin, authorizat:
16 x-envoy-upstream-service-time: 68
17 connection: close
18 Content-Length: 47997
19
20 <!DOCTYPE html>
21 <html xml:lang="en" lang="en">
22
23   <head>
24     <meta charset="utf-8">
25     <meta name="viewport" content="width=device-width, initial-scale=1.0, minimu
26     <title>
      DigiLocker
    </title>
27   <!-- global style css -->
```

### Impact: -

Fortunately, from a security perspective, the use of the wildcard is restricted in the specification as you cannot combine the wildcard with the cross-origin transfer of credentials (authentication, cookies or client-side certificates). Consequently, a cross-domain server response of the form:

Access-Control-Allow-Origin: \*

It is not permitted as this would be dangerously insecure, exposing any authenticated content on the target site to everyone.

Given these constraints, some web servers dynamically create Access-Control-Allow-Origin headers based upon the client-specified origin. This is a workaround for CORS constraints that is not secure.