Cross-Origin Resource Sharing (CORS)

https://campus.barracuda.com/doc/96768789/

Cross-Origin Resource Sharing (CORS) is an HTTP header-based security mechanism that enables a web page of one origin (domain, scheme, or port) to access the resources of another origin. For example: Assume www.ap.abc.com and www.ab.xyz.com are two applications on different domains. If the JavaScript from the application www.ap.abc.com wants to refer to the content from www.ab.xyz.com, it can be permitted only if CORS headers are set to allow www.ab.xyz.com.

On the Barracuda WAF, the CORS policy configuration can be offloaded from the server. Administrators can specify the domain(s), method(s), and header(s) that need to be allowed to access the response from the server when a request is sent to the configured URL. To enable CORS Protection, **Override CORS** must be set to **Yes**.

After **Override CORS** is set to **Yes**, any CORS-related headers set by back-end servers are dropped by the Barracuda WAF as per the configured settings.

Configure CORS Protection

1. Go to the **BOT MITIGATION > Bot Mitigation** page.
2. Select **Edit** under **Options** next to the URL policy for which you want to configure CORS protection.
3. On the **Edit URL Policy** page, scroll down to the **CORS Protection** section and do the following:
    1. **Override CORS** - Set to **Yes** to override the CORS response headers that are returned by the back-end server. When set to **No**, the response is sent to the client without any CORS related changes.
    2. **CORS Allow Origin** – Specify the origin that needs to be allowed to access the response from the server.
    3. **CORS Allow Methods** – Specify the HTTP method(s) that needs to be allowed to access the response. Use a comma as the delimiter for setting multiple values. For example: GET, POST
    4. **CORS Allow Headers** – Specify the header(s) that needs to be allowed to access the response. Use a comma as the delimiter for setting multiple values. For example: X-Custom-Header, Upgrade-Insecure-Requests
    5. **CORS Allow Credentials**
        1. Select **True** if you want the credentials, like cookies and the authorization header, to be sent with the request.
        2. Select **Do not include** to not send the credentials in the request.
    6. **CORS Max Age** – Specify the time in seconds for the results of a preflight request to be

    cached. The results here refer to the content of the **Access-Control-Allow-Methods** and **Access-Control-Allow-Headers** headers.
    7. **CORS Expose Headers** – Specify the response header(s) that needs to be visible to the client.
4. Click **Save**.