Cross Origin Resource Sharing

Systems Analysis & Design

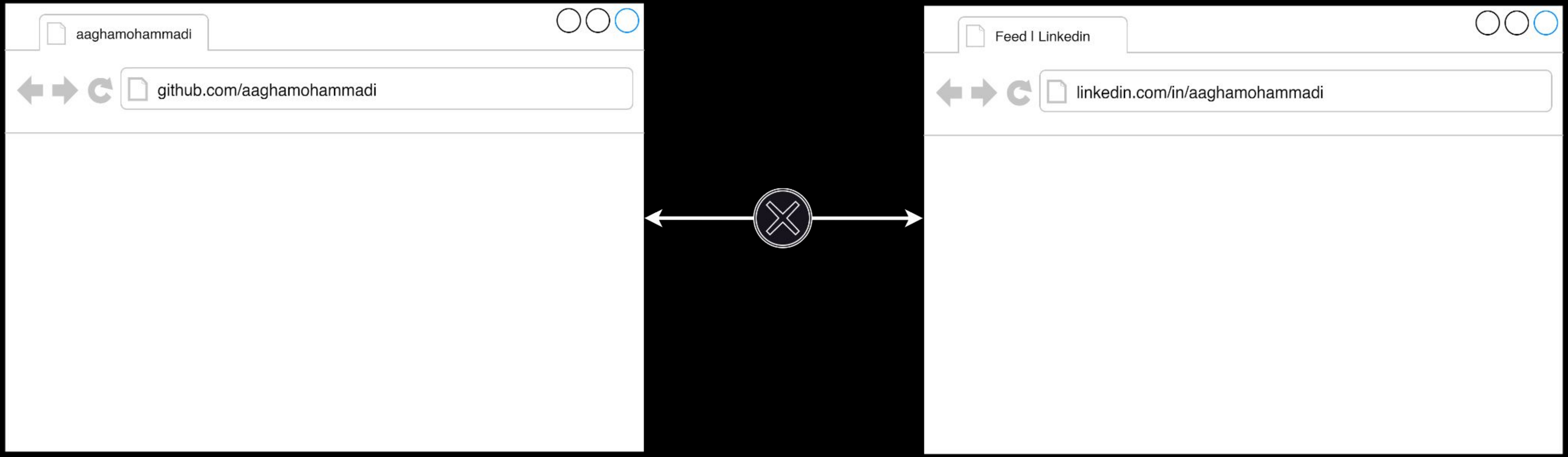
Learning Objectives

By the end of this session, you will have acquired the following information:

- Same-Origin Policy (SOP)
- What Is an Origin?
- Cross-Origin Resource Sharing (CORS)
- CORS Headers
- CORS Vulnerabilities

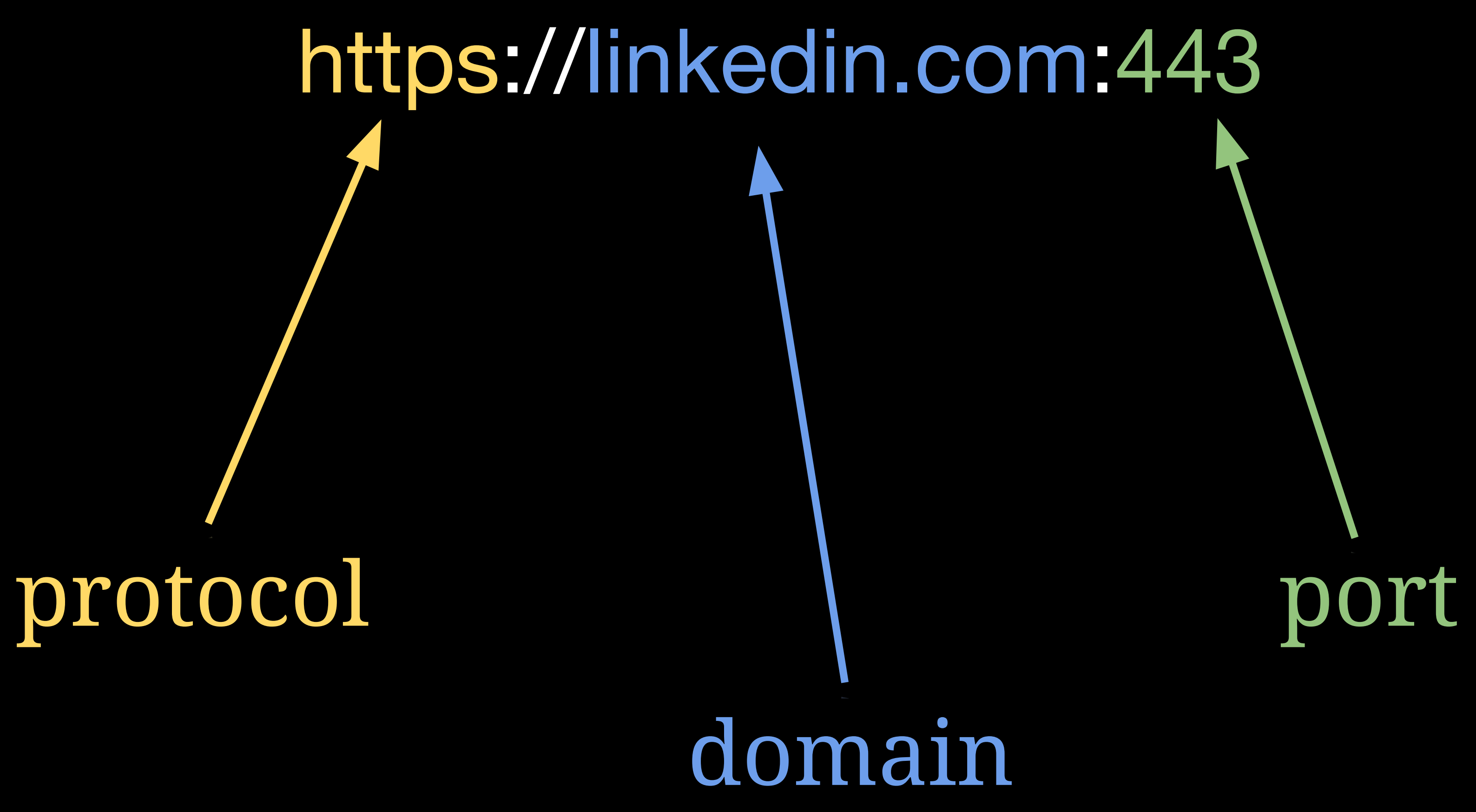
Same-Origin Policy (SOP)

- The Same-Origin Policy (SOP) is a rule enforced by browsers to control access to data between web applications.
  - It doesn’t prevent writing between web applications, but rather, it prevents reading between them.
  - Access is determined based on the origin.



What Is an Origin?

The origin is defined by the protocol, domain, and port of the URL that is used to access it.



https://linkedin.com:443

protocol

domain

port

Consider the URL: http://linkedin.com/in/aaghamohammadi.

URL	Permitted?	Reason
http://linkedin.com/	Yes	Same protocol, domain, and port.
http://linkedin.com/login/	Yes	Same protocol, domain, and port.
https://linkedin.com/	No	Different protocol and port.
http://business.linkedin.com/	No	Different domain.
http://linkedin.com:8080/	No	Different port.

What happens when business.linkedin.com tries to access resources from the linkedin.com origin?

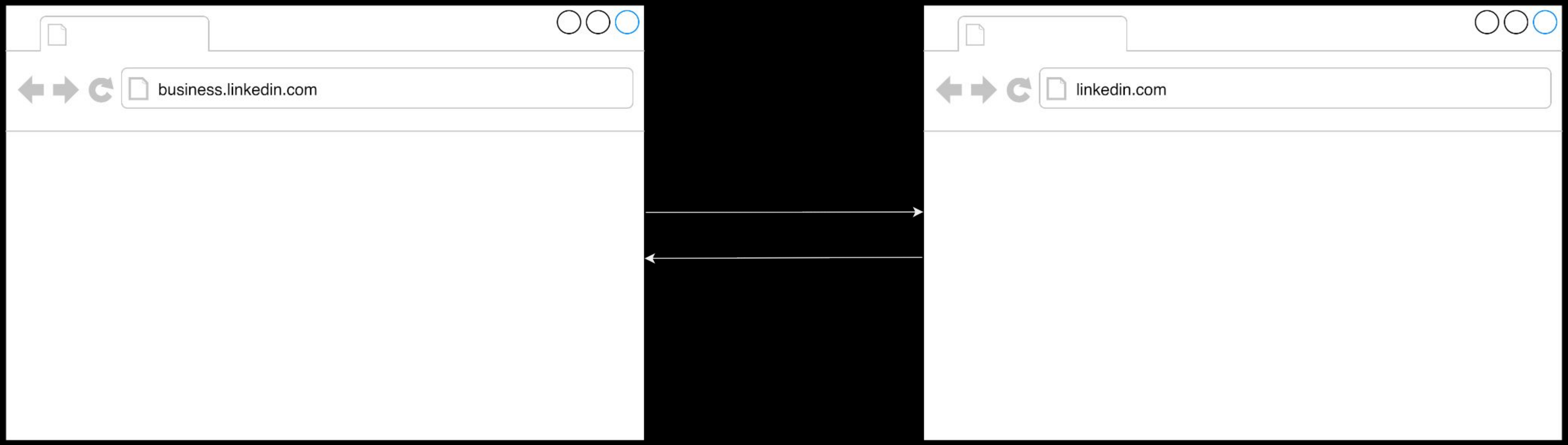
Access to XMLHttpRequest at https://linkedin.com from origin https://business.linkedin.com has been blocked by **CORS policy**. The ‘Access-Control-Allow-Origin’ header is not present on the requested resource.

Cross-Origin Resource Sharing (CORS)

- Cross-Origin Resource Sharing (CORS) is a mechanism that uses HTTP headers to define the origins that the browser permits for loading resources.
- CORS makes use of 2 HTTP headers:
  - Access-Control-Allow-Origin
  - Access-Control-Allow-Credentials

Access-Control-Allow-Origin Header

- The ‘Access-Control-Allow-Origin’ response header indicates whether the response can be shared with the requesting code from the given origin.



Request:

```
GET  /learning  HTTP/1.1
Host: linkedin.com
Origin: business.linkedin.com
```

Response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: business.linkedin.com
```

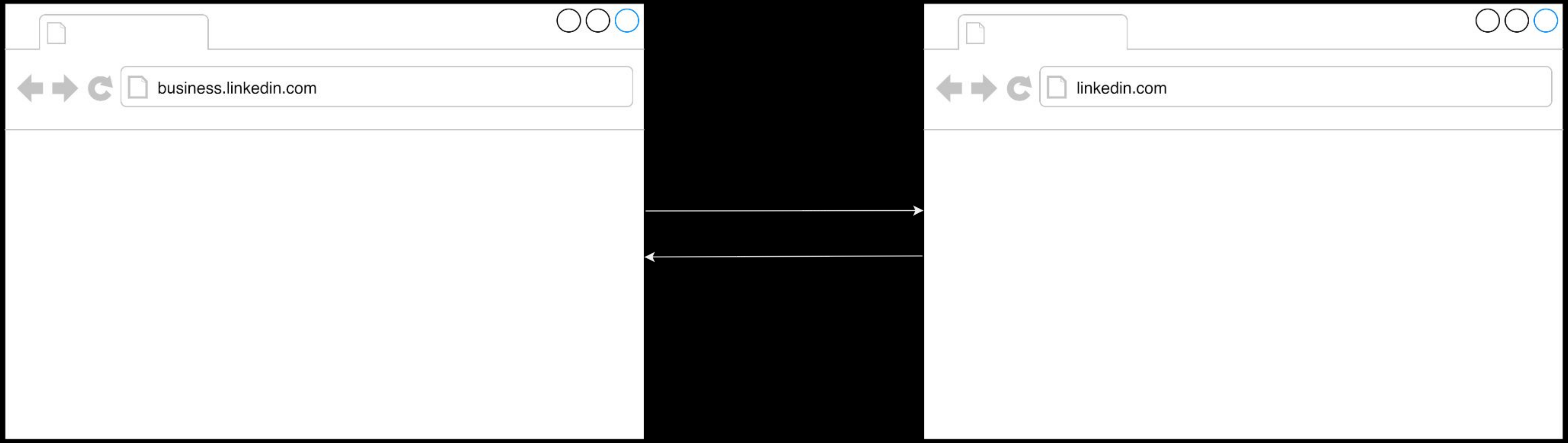
Access-Control-Allow-Origin Header

- The ‘Access-Control-Allow-Origin’ response header indicates whether the response can be shared with the requesting code from the given origin.

```
Access-Control-Allow-Origin: *
Access-Control-Allow-Origin: <origin>
```

Access-Control-Allow-Credentials Header

- The ‘Access-Control-Allow-Credentials’ response header allows cookies, or other user credentials, to be included in cross-origin requests.



```
Request:
GET  /learning  HTTP/1.1
Host: linkedin.com
Cookie: session=iW019U8YB73HZ4d7ShOxnGrQqcja7ah2
Origin: business.linkedin.com
```

```
Response:
HTTP/1.1 200 OK
Access-Control-Allow-Origin: business.linkedin.com
Access-Control-Allow-Credentials: true
```

Access-Control-Allow-Credentials Header

- The ‘Access-Control-Allow-Credentials’ response header allows cookies, or other user credentials, to be included in cross-origin requests.
- If the server is configured with the wildcard (‘*’) as the value of the ‘Access-Control-Allow-Origin’ header, then the use of credentials is not allowed.

```
Access-Control-Allow-Credentials: true
```

CORS Vulnerabilities

- CORS vulnerabilities arise from CORS configuration issues.
- Granting access to all domains that end in a specific string
  - Example: bank.com
  - Bypass: maliciousbank.com
- Granting access to all domains that begin with a specific string
  - Example: bank.com
  - Bypass: bank.com.malicious.com

Further Resources

- Release It!: Design and Deploy Production-Ready Software (pages: 223-242)