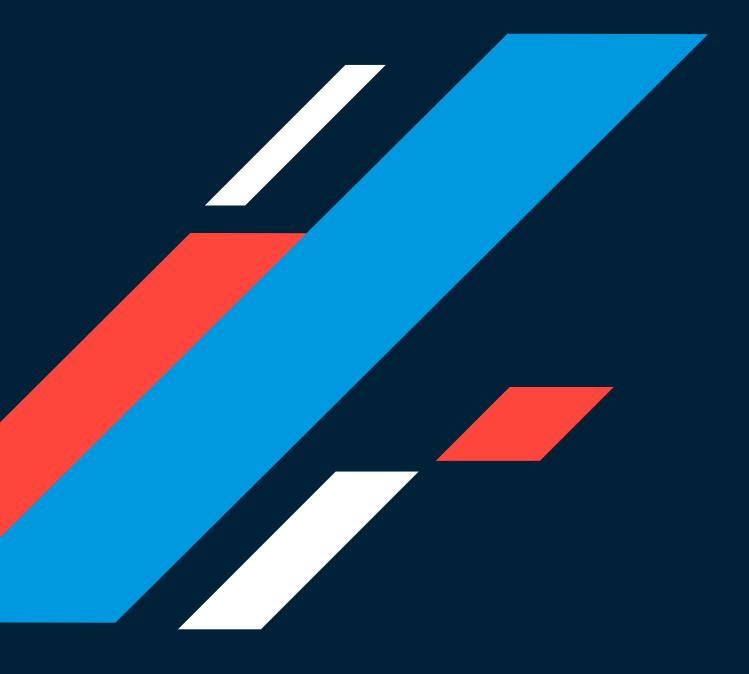


## NodeJS Security





## Agenda\_

- **01** Introduction
- 02 Application Security
- 03 Error & Exception handling
- 04 Server Security
- 05 Platform Security



## Application Security



## Application Security.

- Use flat promise chain.
- Set request size limits.
- Perform input validation.
- Perform output escaping



### Application Security.

- Perform application actively logging.
- monitor the event loop.
- Take precautions against brute forcing attack.
- use anti CSRF tokens.
- Remove unnecessary routes



## Application Security.

- Prevent HTTP parameter pollution.
- Only return what is necessary.



# Error and Exception handling



## Error and Exception handling.

- Handle uncaught Exception.
- Listen to errors when using EventEmitter.





- Setting cookie flag appropriately.
- Use Appropriate security headers .



HTTP/1.1 200 OK

X-Powered-By: Express

Content-Type: text/html; charset=utf-8

Content-Length: 20

ETag: W/"14-SsoazAISF4H46953FT6rSL7/tvU"

Date: Fri, 29 Jul 2022 10:47:59 GMT

Connection: keep-alive Keep-Alive: timeout=5



Content-Security-Policy: default-src 'self';base-uri 'self';block-all-mixed-content;font-src 'self' https: data:;form-action 'self';frame-ancestors 'self';img-src 'self' data:;object-src 'none';script-src 'self';script-src-attr 'none';style-src 'self' https: 'unsafe-inline';upgrade-insecure-requests

- X-DNS-Prefetch-Control: off
- Expect-CT: max-age=0
- X-Frame-Options: SAMEORIGIN
- Strict-Transport-Security: max-age=15552000; includeSubDomains
- X-Download-Options: noopen
- X-Content-Type-Options: nosniff
- X-XSS-Protection: 0



## Platform Security

### Platform Security.

- Keep your packages upto date.
- Don't use dangerous functions.
- Stay away from evil regexs.
- Run security linters.
- use strict mode

## THANK YOU.