

- [\[Doc\]](#) Crypto ( )
- [\[Doc\]](#) TLS/SSL
- [\[Doc\]](#) HTTPS
- [\[Point\]](#) XSS
- [\[Point\]](#) CSRF
- [\[Point\]](#)
- [\[Point\]](#) Sql/Nosql

## Crypto

Node.js [crypto](#) , OpenSSL HMAC .

Node.js , ( Python) . ( ), .

“ ?

, . / . ( md5) DBA

## TLS/SSL

, . SSL (Secure Socket Lay

1. , ;
2. ;
3. , .

:

- SSL
- ,
- SSL

1999 , SSL , . IETF SSL / . (Transport Layer Security, T

## HTTPS

, , ( / ), , , /

(Public Key Infrastructure, PKI) , (Certificate Service, DS) .

RA , CA , DS . , (

TLS/SSL OpenSSL TLS/SSL public/private key. public/private keylet's ,  
Encrypt HTTPS

PKI HTTPSTTPS , .CA 12306), CA ( / , .

CA , / , CA .

## XSS

(Cross-Site Scripting, XSS) , CSS XSS. ,

Cookie / / . , js (Cookie ), img.:

“ Html XSS? ?

```
<script>alert(' xss');</script>
```

url

```
<table background="javascript: alert( /xss/)"></table>  

```

, , , Tab

```

```

(URL , Unicode , HTML , ESCAPE )

```
<img%20src=%22javascript: alert(' xss' );%22>  

```

## CSP

, , CSP .

Node.js , hashes :

```
const crypto = require('crypto');

function getHashByCode(code, algorithm = 'sha256') {
  return algorithm + '-' + crypto.createHash(algorithm).update(code, 'utf8').digest("base64");
}

getHashByCode(' console.log("hello world");'); // 'sha256-
wxWy1+9Lmiu0eDwtQyZNmWpT0jqCUikqaqVLJdtdh/0='
```

CSP :

```
content-security-policy: script-src 'sha256-wxWy1+9Lmiu0eDwtQyZNmWpT0jqCUikqaqVLJdtdh/0='
```

```
<script>console.log('hello geemo')</script> <!-- -->
<script>console.log('hello world');</script> <!-- -->
```

[CSP Policy Directives](#),

## CSRF

(Cross-Site Request Forgery, CSRF, [https://www.owasp.org/index.php/Cross-Site\\_Request\\_Forgery\\_\(CSRF\)\\_Prevention\\_Cheat\\_Sheet](https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF)_Prevention_Cheat_Sheet)) . A ( ) cookie /

Q . A post , id. B CSRF . Q .

(xss), , email , , , .

CSRF , (Cross-Site Request) / . , , :

```
a. public.com
b. public.com
c. public.com
...
```

, c.public.com xss , CSRF .

, , , .

:

1. A ( ) http header origin
2. CSRF token

## 1.

CSRF , header:

- Origin Header
- Referer Header

## 2.CSRF token

, , token session , token , token , .

(Man-in-the-middle attack, MITM) , , ,

wifi, / wifi . , wifi wifi .

MITM, PKI / TLS , wifi HTTPS . HTTP ,

, , ; , , ; .

## SQL/NoSQL

, . eval, new Function , .

## SQL

Sql . / , sql :

```
SELECT * FROM users WHERE userna = 'myName' AND password = 'mySecret';
```

```
' ; DROP TABLE users; --' , : ,
```

```
SELECT * FROM users WHERE userna = 'myName' AND password = '' ; DROP TABLE users; --';
```

, ( ), ( ), ( , ) . :

- / ( )
- ( , 12306 )

- SQL
- ( limit, order by )
- ...

# NoSQL

:

```
let {user, pass, age} = ctx.query;

db.collection.find({
  user, pass,
  $where: `this.age >= ${age}`
})
```

age . GET/POST ( ?nodes[0]=alan , qs ,

Revision #1

Created 3 August 2021 02:28:04 by

Updated 3 August 2021 02:29:32 by