Безопасность приложений Node.js

github.com/HowProgrammingWorks

Тимур Шемсединов

Chief Software Architect at Metarhia
Lecturer at Kiev Polytechnic Institute

github.com/tshemsedinov

Аспекты безопасности для Node.js

- Конкурентная модель исполнения I/O
- Уязвимости path traversal повсюду
- SQL injection, XSRF, XSS и другие типичные
- Зависимости: загляните в node_modules
- Утечки ресурсов (приводящие к падению)
- Храним только хеши паролей с солью
- Управляем нагрузкой, иначе встречаем DoS

Инструменты безопасности для Node.js

- Линтер может кое-что найти github.com/nodesecurity/eslint-plugin-security
- Умеет искать npm audit и даже исправлять через npm audit fix
- Специальные инструменты: ~~nsp~~, snyk
- Githib имеет встроенный Security Alert

Конкурентная модель

- В одном процессе и одном потоке выполняется одновременно много запросов
- Все данные в памяти у них общие
- Права у процесса общие
- Переменные окружения и конфиги общие
- Соединения с БД, зависимости и сеть общие
- Падает тоже процесс целиком

Глобальное состояние

```
let groupName;

app.use((req, res, next) => {
  groupName = 'idiots'; next();
});

app.get('/user', (req, res) => {
  if (res.groupName === 'idiots') {
    res.end('I know you!');
  }
});
```

Миксины к req и res

```
app.use((req, res, next) => {
  res.groupName = 'idiots';
  next();
});

app.get('/user', (req, res) => {
  if (res.groupName === 'idiots') {
    res.end('I know you!');
  }
});
```

Специальное место res.locals

```
app.use((req, res, next) => {
  res.locals.groupName = 'idiots';
  next();
});

app.get('/user', (req, res) => {
  if (res.locals.groupName === 'idiots') {
    res.end('I know you!');
  }
});
```

Примешивание методов

```
app.get('/user/:id', (req, res, next) => {
  req.auth = (login, password) => { /* auth */ };
  next();
});

app.get('/user/:id', (req, res) => {
  if (req.auth(req.params.id, '111')) {
    res.end('I know you!');
  }
});
```

Соединение с БД на обработчиках

```
app.get((req, res, next) => {
  req.db = new Pool(config);
  next();
});

app.get('/user/:id', (req, res) => {
  req.db.query('SELECT * from USERS', (e, r) => {
  });
});
```

Потеря соединений при ошибках

```
const db = new Pool(config);

app.get('/user/:id', (req, res) => {
  req.db.query('SELECT * from USERS', (err, r) => {
    if (err) throw err;
    // Prepare data to reply client
  });
});
```

Зависимости

- Возможно, встроенные модули умеют
- Пакеты с очень похожими именами
- Зараженные пакеты
- Перед использованием смотрим код
- Смотрим вложенные зависимости
- Проверяем по базам уязвимостей
- Проверяем сообщество и поддержку

SQLI (SQL Injection)

Инъекция SQL

Привет! Смотри, что про тебя пишут:
http://bank-web-site.com/accounts?
name='marcus'%20--%20

“SELECT * from Accounts where name=” + name

XSRF (Cross-Site Request Forgery)

Межсайтовая подделка запросов

Привет! Смотри, что про тебя пишут:
http://payment-system.com/api/transfer?
amount=1000&destination=card-number

XSS (Cross-Site Scripting)

Межсайтовый скриптинг

Привет! Смотри, что про тебя пишут:
http://control-panel.com/help.php?q=
%3Cscript%3Ealert('Hello');%3C/script%3E

CSP (Content Security Policy)

В браузеры встроен еще один
уровень защиты от XSS и других типов атак

https://developer.mozilla.org/en-US/docs/Web/HTTP/CSP

На что еще обратить внимание

- httponly cookies
  https://www.owasp.org/index.php/HttpOnly
- HTTP Headers:
  - X-XSS-Protection
  - X-Frame-Options
  - X-Content-Type-Options
    etc.

OWASP (Open Web App. Security Project)

Открытый проект обеспечения безопасности веб-приложений

https://owasp.org/