Web App Segurtasun Finkapena - SOP

Helburua

Web aplikazioa segurtasun estandarren arabera finkatzeko prozedura, OWASP Top 10 arauak kontuan hartuta.

Aurrebaldintzak

- Web aplikazioaren iturburu kodea eskuragarria
- OWASP ZAP edo antzeko tresnak instalatuta
- SSL/TLS ziurtagiriak eskuragarriak

1. Segurtasun Burua (Security Headers) Konfigurazioa

1.1 Helmet Middleware Ezarri

```
npm install helmet
```

1.2 Content Security Policy (CSP) Konfiguratu

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

app.use(helmet.contentSecurityPolicy({
  directives: {
    defaultSrc: ["'self'"],
    styleSrc: ["'self'", "'unsafe-inline'"],
    scriptSrc: ["'self'"],
    imgSrc: ["'self'", "data:"],
    connectSrc: ["'self'"],
    fontSrc: ["'self'"],
    objectSrc: ["'none'"],
    mediaSrc: ["'self'"],
    frameSrc: ["'none'"],
  },
}));
```

1.3 HSTS (HTTP Strict Transport Security) Gaitu

```
app.use(helmet.hsts({
  maxAge: 31536000,
  includeSubDomains: true,
  preload: true
}));
```

2. Sarreren Balidazioa (Input Validation)

2.1 OWASP Validator Erabili

```
npm install express-validator
```

2.2 Adibidea - Produktu Sorrera

```
const { body, validationResult } = require('express-validator');

app.post('/api/products', [
  body('name').trim().isLength({ min: 2, max: 100 }).escape(),
  body('price').isFloat({ min: 0 }).toFloat(),
  body('description').optional().trim().isLength({ max: 500 }).escape()
], (req, res) => {
  const errors = validationResult(req);
  if (!errors.isEmpty()) {
    return res.status(400).json({ errors: errors.array() });
  }
  // Prozesatu produktua
});
```

2.3 SQL Injection Prebentzioa

- Beti parameterized queries erabili
- Ez eraiki SQL kateak dinamikoki
- Erabili ORM seguruak (Sequelize, TypeORM, Mongoose)

3. Sesioen Kudeaketa

3.1 Cookie Seguruak Konfiguratu

```
app.use(session({
  secret: process.env.SESSION_SECRET,
  resave: false,
  saveUninitialized: false,
  cookie: {
    secure: true, // HTTPS soilik
```

```
    httpOnly: true, // JavaScript atzigarria ez
    sameSite: 'strict', // CSRF prebentzioa
    maxAge: 3600000 // 1 ordu
  }
}));
```

3.2 JWT Token Seguruak

```
const jwt = require('jsonwebtoken');

const token = jwt.sign(
  { userId: user.id },
  process.env.JWT_SECRET,
  { expiresIn: '1h' }
);
```

4. Pasahitz Segurtasuna

4.1 Hashing Konfiguratu

```
npm install bcryptjs
```

4.2 Pasahitzak Gorde

```
const bcrypt = require('bcryptjs');

const hashPassword = async (password) => {
  return await bcrypt.hash(password, 10);
};
```

4.3 Pasahitz Politika

- Gutxienez 8 karaktere
- Maiuskula, minuskula, zenbaki eta karaktere bereziak
- Ez erabili ohiko pasahitzak
- Behin betiko blokeoa 5 saiakera okerren ondoren

5. XSS (Cross-Site Scripting) Prebentzioa

5.1 Output Encoding

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

```
const safeOutput = xss(userInput);
```

5.2 Content Security Policy (CSP)

Ikusi atala 1.2

6. CSRF (Cross-Site Request Forgery) Prebentzioa

6.1 CSRF Tokenak

```
npm install csurf
```

6.2 Inplementazioa

```
const csrf = require('csurf');

const csrfProtection = csrf({ cookie: true });

app.get('/form', csrfProtection, (req, res) => {
  res.render('send', { csrfToken: req.csrfToken() });
});
```

7. Rate Limiting

7.1 Ezarri

```
npm install express-rate-limit
```

7.2 Konfigurazioa

```
const rateLimit = require('express-rate-limit');

const limiter = rateLimit({
  windowMs: 15 * 60 * 1000, // 15 minutu
  max: 100 // Gehienez 100 eskaeri
});

app.use('/api/', limiter);
```

8. Fitxategi Igoera Segurua

8.1 Balidazioak

- Fitxategi mota balidatu (MIME type eta luzapena)
- Tamaina muga ezarri
- Izena berrizendatu (timestamp + random string)
- Direktorio egokia (web rootetik kanpo)

8.2 Adibidea

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

const upload = multer({
  dest: 'uploads/',
  limits: { fileSize: 2 * 1024 * 1024 }, // 2MB
  fileFilter: (req, file, cb) => {
    const allowedTypes = ['image/jpeg', 'image/png'];
    if (!allowedTypes.includes(file.mimetype)) {
      return cb(new Error('Fitxategi mota ez da baimendua'));
    }
    cb(null, true);
  }
});
```

9. HTTPS Konfigurazioa

9.1 Let’s Encrypt Erabili

```
sudo apt install certbot python3-certbot-nginx
sudo certbot --nginx -d zabala-gailetak.com -d www.zabala-gailetak.com
```

9.2 Auto-Renewal

```
sudo certbot renew --dry-run
```

10. Monitorizazioa eta Log-ak

10.1 Winston Logger

```
npm install winston
```

10.2 Segurtasun Gertaerak Erregistratu

```
logger.info('Login attempt', {
  userId: user.id,
```

```
  ip: req.ip,
  userAgent: req.headers['user-agent']
});

logger.warn('Failed login attempt', {
  username: req.body.username,
  ip: req.ip
});

logger.error('Security violation', {
  type: 'SQL Injection attempt',
  ip: req.ip
});
```

11. Pentesting-a

11.1 OWASP ZAP Erabili

```
docker run -t owasp/zap2docker-stable zap-baseline.py -t https://zabala-gailetak.com
```

11.2 Buruzko Proba

- SQL Injection probak
- XSS probak
- CSRF probak
- Auth bypass probak
- Rate limiting probak

12. Berrikuspen eta Eguneratzeak

- Hilabetero dependentziak eguneratu
- Hiru hilabetero segurtasun audit-ak egin
- Urtero pentesting osoa egin
- Gertaera berrien arabera konfigurazioa eguneratu

Erreferentziak

- OWASP Top 10: https://owasp.org/www-project-top-ten/
- OWASP ASVS: https://owasp.org/www-project-application-security-verification-standard/
- OWASP Testing Guide: https://owasp.org/www-project-web-security-testing-guide/