## Complete Code, Commands, and Configuration Summary (Week 1 to Week 3)

This document outlines all key code snippets, commands, and configuration changes implemented across three weeks while working on the Damn Vulnerable NodeJS Application (DVNA).

---

### Week 1: Vulnerability Discovery & Exploitation

Commands Used:

```bash

# Start DVNA server

npm start

# Delete old database

rm -f dvna.sqlite

# List open ports

nmap -sV localhost -p 9090

nmap -A localhost

# Launch OWASP ZAP (manual)

# Or access ZAP via browser at http://127.0.0.1:8080

# Launch Nikto

nikto -h http://localhost:9090

```

Manual Exploits:

- XSS Payload Example:

```html

<script>alert('XSS')</script>

```

- Session Hijacking (Firefox):

- Open DevTools → Storage → Cookies → copy `connect.sid`

- Use in another browser session to impersonate user

---

### Week 2: Fixing Vulnerabilities and Improving Security

1. Input Validation

# Install validator:

```bash

npm install validator

```

# Example Changes in `authHandler.js`:

```js

const validator = require('validator');

if (!validator.isEmail(email)) {

return res.status(400).send('Invalid email');

}

if (!validator.isLength(password, { min: 6 })) {

req.flash('danger', 'Password must be at least 6 characters');

return res.redirect('/register');

}

```

---

2. Password Hashing

# Install bcrypt:

```bash

npm install bcrypt

```

# In `passport.js`:

```js

const bCrypt = require('bcrypt');

function createHash(password) {

return bCrypt.hashSync(password, bCrypt.genSaltSync(10), null);

}

```

---

3. HTTP Header Security

# Install helmet:

```bash

npm install helmet

```

# In `server.js`:

```js

const helmet = require('helmet');

app.use(helmet());

```

---

### Week 3: Logging, Testing & Final Security Checks

1. Logging with Winston

# Install winston:

```bash

npm install winston

```

# Create `core/logger.js`:

```js

const winston = require('winston');

const logger = winston.createLogger({

level: 'info',

format: winston.format.simple(),

transports: [

new winston.transports.Console(),

new winston.transports.File({ filename: 'security.log' })

]

});

module.exports = logger;

```

# In `appHandler.js`:

```js

const logger = require('./logger');

logger.warn("/ping endpoint received: " + req.body.address);

```

# In `server.js`:

```js

const logger = require('./core/logger');

logger.info('DVNA server started');

```

---

2. Penetration Testing

```bash

# Scan open ports

nmap -sV localhost -p 9090

nmap -A localhost

# Basic vulnerability scan

nikto -h http://localhost:9090

```

---

3. Reset Password Secure Flow Fix

**In `authHandler.js`:**

const md5 = require('md5');

const token = md5(login);

const resetLink = `/resetpw?login=${login}&token=${token}`;

if (req.body.token === md5(req.body.login)) {

// Valid token, proceed to reset

}

```---

**Summary**

By Week 3, DVNA was significantly hardened:

- Inputs were validated with `validator`

- Passwords were hashed with `bcrypt`

- Secure headers added via `helmet`

- Logging introduced with `winston`

- Manual tests validated the security fixes