취약점 보고서			
성 명	강준영	**소 속**	교하고등학교 2학년
이메일	juneyoungdev@gmail.com	**전화번호**	010 2075 8000
주 소	경기도 고양시 일산서구 가좌3로 45, 212동 804호		

<취약점 개요>

취약점 제목	o Personal Information Exposure through URL Query Parameter Tampering and SQL Injection Attack Possibilities in Specific Webpage, https://coding.edaily.co.kr
취약점 요약	o Personal Information Exposure. o URL Query Parameter Tampering. o SQL Injection. (Only Possibilities)
취약한 버전	o Current Version
취약점 발생환경	o All Types of Browser (Chrome, Safari, Firefox, etc.)

<취약점 상세 설명>

1. 취약점 발견 방법

o 웹페이지 접속 후 회원가입이 완료되면, 본인의 이름과 휴대전화 번호가 표시되는 페이지가 표시

o 해당 페이지의 URL GET Parameter로 'seq' 존재 (https://coding.edaily.co.kr/member/regist_ok.php?seq=nJk)

o 해당 Parameter를 조작하면 ASCII에 기반한 특정한 규칙으로 참가자 정보 또는 Query Error 페이지가 표시

o [첨부 1: 올바른 Parameter 입력할 경우 표시되는 개인정보]

o [첨부 2: 잘못된 Parameter 입력할 경우 표시되는 Query Error]

2. 취약점 발생원인

o member/register_ok.php의 URL Parameter 'seq' 값으로 특정 문자열을 대입하면 약 4개~5개의 연속된 문자열 구간에 한 명의 개인정보가 표시

o 표시되는 개인정보는 참가자의 이름과 휴대전화 번호

o 특정 문자열이란, 현재까지 검증해 본 결과, 세 자리의 문자열로 이루어져 있으며, 맨 앞에서부터 Alphabet lowercase/uppercase, Alphabet lowercase/uppercase, Alphabet lowercase/uppercase+digits (예시: Aa0 또는 ABc)

o 괄호 안은 아스키 코드의 순서쌍이라 하고, 입력 Query에 문자열은 없다고 했을 때, nIY(110, 73, 89) 부터 nJb(110, 74, 98)까지 DB Query Error 페이지에서는 '6+'(54, 43)라는 값으로 인식, nJc(110, 74, 99) 부터 nJf(110, 74, 102)까지 '6,'(54, 44)라고 인식하고, 여기에 특정한 규칙이 있음을 확인

o 4~5개씩 끊어지며 특정 문자열을 순차적으로 입력하면 그에 대응되는 또다른 문자열이 발생하고, 이 또한 ASCII Code의 순서에 따른 문자임을 확인. 이때의 새로운 문자열은 특수문자를 포함한 문자임을 확인 [첨부 3]

3. 취약점 증명 / 검증

o 본 취약점을 검증하기 위해 간단한 Python 코드를 작성 [첨부 3: 검증을 위한 Shell Script & Python Files]

o Shell Script에는 특정 문자열을 생성하여 Excel로 저장하는 write_excel.py와 생성된 Excel (.xls)에서 Excel (.xlsx)로 확장자를 바꾸는 pyexcel 명령어, 그리고 Excel의 특정 row(특정 문자열의 특정 범위)에서 문자열을 추출하여 URL에 대입하고 Parsing하는 get.py 실행하는 내용이 포함

o 결론적으로 참가자 개인정보가 노출되는 경우 참가자 이름과 참가자 휴대전화번호가 JSON Type 저장되고, Query Error가 표시되는 경우 빈 JSON이 생성하여 하나의 File(result.json)로 저장

o 파일 실행 방법: cd report, chmod +x shell.sh, ./shell.sh

o get.py에서 읽어내는 Excel Sheet의 범위(즉, 특정 문자열의 범위)를 A44083부터 A44268까지(즉, nJa ~ nL9) 시행하였으므로 테스트 케이스는 총 186개이지만, 실제로 더 넓은 범위에서 시행할 경우 더 많은 양의 데이터가 수집될 것으로 예측

4. 취약점 악용 시나리오

o 본 취약점은 무차별 대입으로 다량의 Query를 생성하여 요청을 보내 참가자의 개인정보를 수집할 가능성이 우려

o 이 외에도 기본적인 오류 구문들 및 일부 테이블의 구조가 출력되는 것으로 보아 SQL Injection의 가능성이 존재

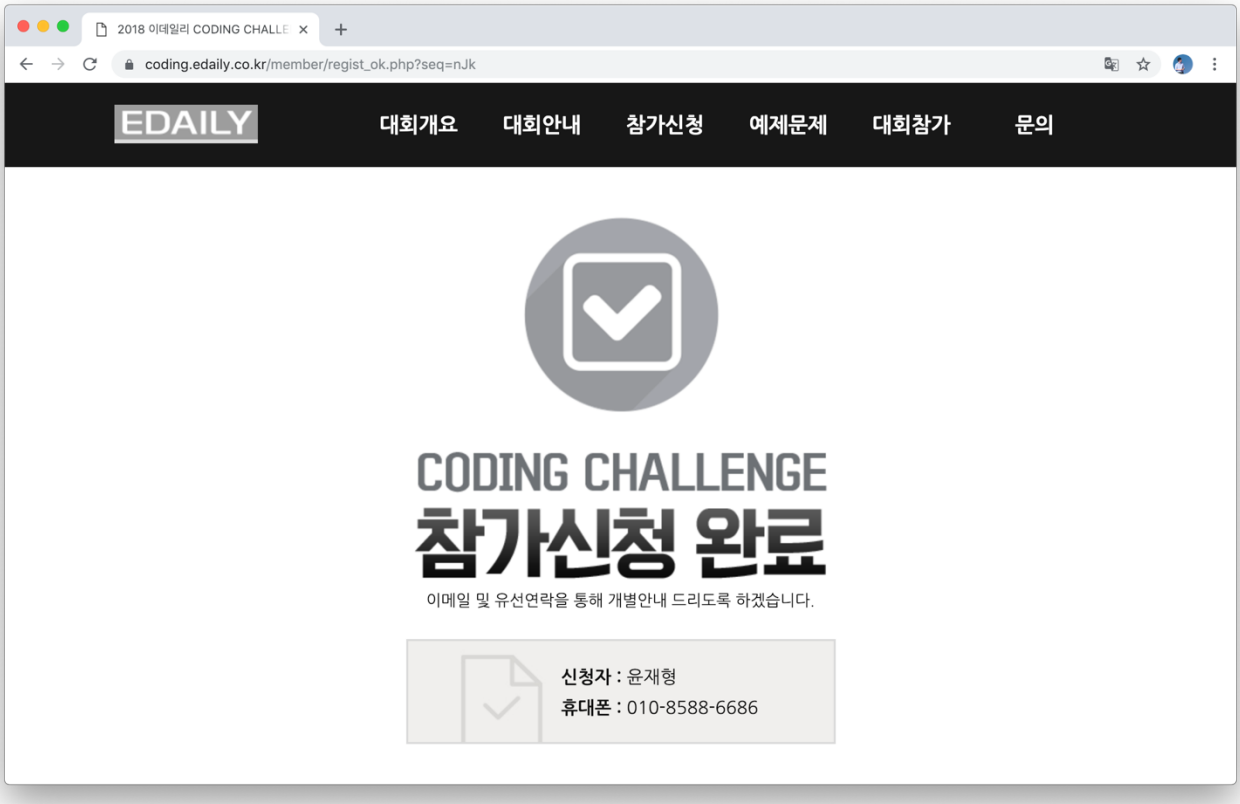
5. 조치 방안

o 사용자의 정보를 저장하는 방법에 있어 새로운 방법의 고안이 필요

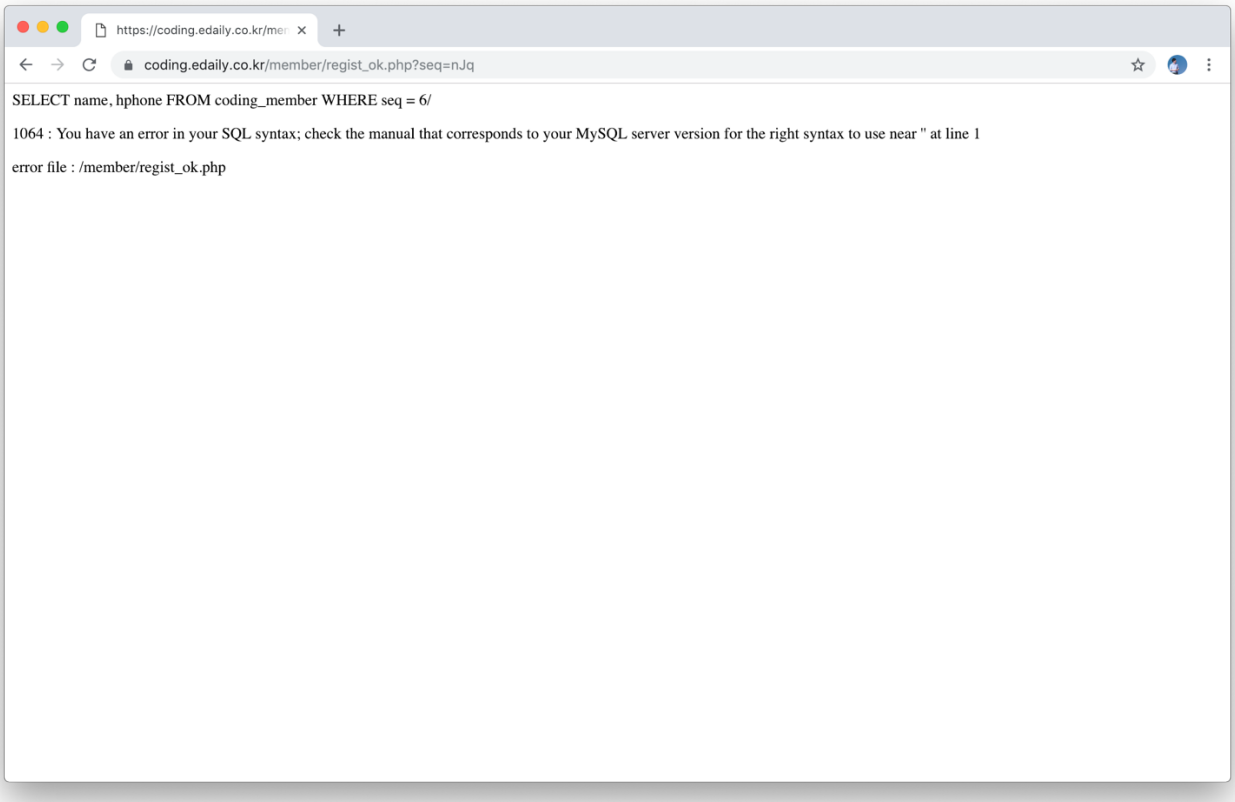
6. 기타

o 없음

[첨부 1]



[첨부 2]



[첨부 3]
https://drive.google.com/file/d/1uEjI6mlRiF3_gYNNJODSdPMVAnKQL_Km/view?usp=sharing