<1p>

안녕하세요. SQL Injection 공격과 방어를 주제로 발표할 손예은, 최유진입니다.

<2p>

목차는 다음과 같습니다.

<3p>

저희는 다른 수업에서 사례로 많이 들었던 뽐뿌 사건에 관심을 가지고 주제를 선정하게 되었습니다. 뽐뿌 사건은 2015년 웹 취약점을 악용한 데이터베이스 공격으로 개인정보 유출이 발생한 사건인데요. 이 공격에서는 웹 어플리케이션의 허점을 악용해 SQL 문장이 실행되게 함으로써 데이터베이스를 비정상적으로 조작하는 공격인 SQL Injection이 사용되었습니다. 그래서 저희는 SQL Injection 공격을 기반으로 하여 sqlmap으로 일반 칼리 리눅스 환경에서 진행한 공격과 docker 컨테이너를 각각 공격자와 방어자로 설정한 공격 실습을 진행하였습니다.

<4p>

먼저 공격 구성도입니다. 공격자 컴퓨터를 kali로 설정하고 타겟 url을 설정하여 데이터 베이스 정보를 획득하는 공격 시나리오를 구성했습니다.

<5p>

다음으로 docker 컨테이너를 이용한 공격입니다.

도커 컨테이너에서 공격 컨테이너와 타깃 컨테이너를 각각 생성 후, 공격 컨테이너에서 네트워크 공격을 실행하고 타깃 컨테이너에 공격을 발생시키는 시나리오를 구성하였습니다.

<6p>

우선, 첫 번째 공격 실습입니다. 타겟 url은 실습 가능한 예시 url로 다음과 같습니다. 1번부터 4번까지 과정을 거쳐 테이블과 컬럼 정보를 확인하고 타겟 url에 가입되어있는 사람의 이름과 비밀번호, 이메일, 휴대전화번호 등을 확인할 수 있었고, 앞서 말씀드린 참고 사례에서와 같은 유출 공격을 수행할 수 있었습니다.

<7p>

다음으로, docker 컨테이너를 통한 공격 과정입니다. 저희는 도커 컨테이너 간 네트워크 공격을 통해 타겟 컨테이너의 ip주소에 접근하여 sql 인젝션을 하고자했습니다. 타겟 컨테이너는 web_server이고 공격자 컨테이너는 kali입니다. 저희가 슬라이드와 같은 공격을 통해 얻은 웹 서버의 ip 주소는 172.17.0.3 입니다.

<8p>

공격 컨테이너의 ip 주소를 알아낸 후 wireshark를 통해 tcp 프로토콜의 패킷들이 전송되는 것을 확인하여 공격이 제대로 이루어졌음을 알 수 있었습니다. 또한, 해당 ip주소에 접근하는 것도 성공적으로 이뤄졌습니다. 이후, 얻어낸 ip주소를 타겟 url을 export하고, sql injection 공격 코드를 코드의 페이로드부분에 삽입하여 공격을 실행했습니다.

<9p>

타겟 컨테이너에 sql 취약점을 이용해 or 1=1인 sql 쿼리문을 넣고 확인한 결과입니다. 공격이 성공한 것을 볼 수 있습니다.

<10p>

다음은 실습에 대한 방어입니다. 저희는 업데이트와 방화벽 설치, 차단 대상 설정, iptables rules 교체, 대상 포트 차단 등 방어에 대해 여러 방면으로 고려하고 시도해봤는데요, 그 중에서 가장 방어 성공에 근접했다고 느낀 방어에 대해 말씀드리겠습니다.

<11p>

저희가 시도한 방법은 iptables를 사용해서 방화벽을 설정하여 접근제어를 시도한 방법입니다. 슬라이드에서와 같이 차례로 코드를 입력해주고 etc 밑에 iptables 밑에 방화벽 rules를 재복원햇습니다.

<12p>

앞선 방법으로 방화벽 rules가 제대로 설정되지 않아 firewall_rules를 iptable에 실습 당일 날짜의 rules로 재설정 이후 재복원해주어 다시 시도했습니다.

<13p>

먼저 정책 리스트를 확인해보면 172.17.0.3에서 80포트가 accept 된 것을 알 수 있습니다. 이것을 차단하기 위해 아래와 같은 명령어를 수행합니다.

저희는 공격 컨테이너에서 사용하는 80포트에서 타겟 url인 172.17.0.3에 접근하는 것을 reject하도록 시도했습니다. 그러나 화면에서 보시는 것과 같이 설정 전후의 차이가 없이 모두 ACCEPT로 나오는 것을 볼 수 있습니다. 이후, 재시도 끝에 ACCEPT를 DROP으로 바뀌었으나 재부팅을 했더니 ACCEPT로 다시 돌아왔습니다. 저희의 방어가 성공하지 못한 이유는, iptables 설정을 재부팅 이후 지속시키도록 설정하지 못하게 한 것이 한 부분일 것 같습니다. 명령어나 다른 설정을 통해

재부팅 초기화를 고려하지 못한 부분이 아쉬웠습니다. 또한, 초기화된 설정을 확인하고 재시도 했을 때에 공격 컨테이너가 처음에는 접근을 못하다가 타겟 url 접근에 시간이 지연되고 나면 다시 접근이 가능해졌는데, 타겟 url이 3초 이상과 같은 일정 시간이 지나면 해당 포트가 접근하지 못하도록 설정하는 부분에 대한 것을 고려하지 못한 것도 방어에 성공하지 못한 이유가 아닐까 짐작해봤습니다.

<14p>

이상 저희의 발표를 마치고 Q&A 시간을 가지겠습니다. 감사합니다.