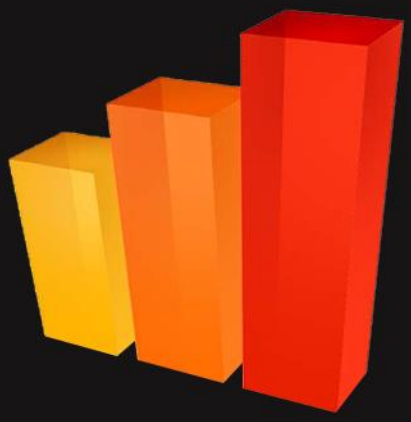
netsparker®
web application security scanner



넷스파커 보고서

꼭 필요한 보고서를 제공

넷스파커는 사용자의 요구에 맞는 다양한 보고서를 제공 합니다. 넷스파커가 제공하는 보고서는 아래와 같습니다.

임원용 요약보고서

점검 결과의 가장 핵심적인 부분만 추려서 제공 합니다.

PCI 컴플라이언스 보고서

PCI DSS 컴플라이언스에 위배되는 항목이 있는지에 대해 보고서를 제공합니다.
가장 상세한 컴플라이언스 입니다.

상세 보고서

점검 결과를 상세하게 기술하는 보고서를 제공합니다.

HIPAA 컴플라이언스 보고서

HIPAA 컴플라이언스에 위배되는 항목이 있는지에 대해 보고서를 제공합니다.

비교 보고서

서로 다른 스캔 결과를 비교하여 보고서를 작성합니다.
추적 점검을 할 때 유용합니다.

URL 목록 보고서

크롤링 되고 스캔된 URL 목록을 제공합니다.

OWASP TOP 10 컴플라이언스 보고서

OWASP TOP 10 컴플라이언스에 위배되는 항목이 있는지에 대해 보고서를 제공합니다.

취약점 목록 보고서

점검에서 나타난 취약점들의 목록에 대한 보고서를 제공합니다.

OWASP top10 보고서 예시

OWASP TOP 10 2013 스캔 보고서 요약

대상 URL	http://php.testsparker.com/
스캔 날짜	2016-02-05 오후 12:03:57
보고 날짜	2016-02-05 오후 12:23:56
스캔 시간	00:07:43

총 요청 5344

평균 속도 11.53 요청/초

28 식별됨

17 확정됨

6 취급

2 정보

스캔 설명

설명 이 보고서는 OWASP Top Ten 2013 등급에 기반하여 생성되었습니다.

OWASP Top 10 2013에 규정된 취약점 중 28 개의 취약점 개가 이 웹사이트에서 발견되었습니다.

OWASP TOP 10 2013에 따른 취약점

OWASP A1 - 인젝션

URL	엄격	취약점
/artist.php	치명적	블라인드 SQL 인젝션
/artist.php	치명적	부울 기반 SQL 인젝션
/hello.php	치명적	원격 코드 평가 (PHP)
/nsbackup.php	치명적	블라인드 커맨드 인젝션
/nsbackup.php	치명적	커맨드 인젝션
/process.php	치명적	RFI(Remote File Inclusion)

OWASP A3 - 크로스 사이트 스크립팅 (XSS)

URL	엄격	취약점
/hello.php	중요	크로스 사이트 스크립팅
/process.php	중요	RFI(Remote File Inclusion)를 통한 크로스 사이트 스크립팅
/products.php	중요	크로스 사이트 스크립팅

취약점 상세 내용과 조치 권고 예시

1. 블라인드 SQL 인젝션

1 합계

치명적

확정됨

1

Netsparker가 사용자에 의해 입력 된 데이터가 백엔드 데이터베이스에서 일반 데이터가 아닌 SQL 명령으로 해석 될 때 발생하는 블라인드 SQL 인젝션을 식별하였습니다.

이것은 매우 일반적인 취약점이며 공격이 성공하면 매우 치명적인 영향을 미칠 수 있습니다.

Netsparker가 백엔드 데이터베이스에 테스트 SQL 쿼리를 실행하여 취약점을 **확정** 하였습니다. 시행한 테스트에서, SQL 인젝션이 분명하지는 않았지만, 페이지에서 온 인젝션 시험에 기반한 다른 응답들을 통해 SQL 인젝션을 식별하고 확정 할 수 있었습니다.

영향

백엔드 데이터 베이스, 데이터베이스 연결 세팅, 운영체계에 따라 공격자가 다음의 공격 중 하나 이상을 성공적으로 수행할 수 있습니다:

- 데이터베이스에서 임의의 데이터 또는 테이블을 읽기, 업데이트 및 삭제
- 기본 운영 시스템에 명령을 실행

취할 조치

1. 솔루션에 대한 대책을 참조하십시오.
2. 만약 데이터베이스 액세스 계층 (DAL)를 사용하지 않는 경우, 사용하는 것을 고려하십시오. 문제를 집중시키는 데 도움이 됩니다. 또 ORM(*object relational mapping*)을 사용할 수도 있습니다. ORM 시스템 대부분은 매개 변수화 된 쿼리만을 사용하기 때문에 전체 SQL 인젝션 문제를 해결할 수 있습니다.
3. 동적으로 생성 된 모든 SQL 쿼리를 찾아 매개 변수화된 쿼리로 변환하십시오. *(DAL / ORM을 사용하기로 결정한 경우, 모든 기존 코드를 이 새로운 라이브러리를 사용하도록 변경하십시오.)*
4. 이 리소스에 이전에 발견되지 않은 공격이 있는지 확인하려면 웹 로그 및 어플리케이션 로그를 사용하십시오.

대책

SQL 인젝션 기반 취약점의 위협을 완화하기위한 강력한 방법은 매개 변수화 된 쿼리(*prepared statements*)를 사용하는 것입니다. 대부분의 현대 언어는 이를 위해 내장된 라이브러리를 제공합니다. 가능하다면, 문자열 연결이 있는 SQL 쿼리 또는 동적 SQL 쿼리를 작성하지 마십시오.

성공적인 공격에 필요한 기술

SQL 인젝션 취약점을 공격하기 위해 자유롭게 사용할 수있는 툴이 굉장히 많습니다. 종속성이 많은 복잡한 분야입니다. 그러나, 이 분야에서 사용할 수있는 다수의 리소스가 이 이슈에 대한 공격자의 인식과, 이슈를 발견하고 이를 활용하는 능력 모두를 끌어올렸다는 사실은 주목해야 합니다. SQL 인젝션은 가장 일반적인 웹 애플리케이션의 취약점 중 하나입니다.

외부 참조

- OWASP SQL 인젝션
- SQL 인젝션 위키

대책 참조

- SQL 인젝션 방지 치트 시트
- SQL 인젝션 방지를 위한 가이드

취약점 확정 예시

7.1. /hello.php 확정됨

http://php.testsparker.com/hello.php?name='%22--%3E%3C/style%3E%3C/scRipt%3E%3CscRipt%3Enetsparker(0...

매개변수

매개변수	형식	값
name	GET	'"--></style></scRipt><scRipt>netsparker(0x00018E)</scRipt>

응답

```
...
o.php(24) : eval()'d code</b> on line <b>1</b><br />
<br />
<b>Parse error</b>: syntax error, unexpected '"' in <b>C:\AppServ\www\hello.php(24) : eval()'d code</b> on line <b>1</b><br />
20$str = 20 '"--></style></scRipt><scRipt>netsparker(0x00018E)</scRipt>;20 </p>

<div style="clear: both;"> </div>
<div class="entry">


</div>
</div>
<div style="clear: both;"> </div>
</div>
<!-- end #content -
...
```