작성자 : 이승용
강연자 : NBP 황성호 과장
해킹 사례로 본 정보보안 가이드
2016년 7월 25일

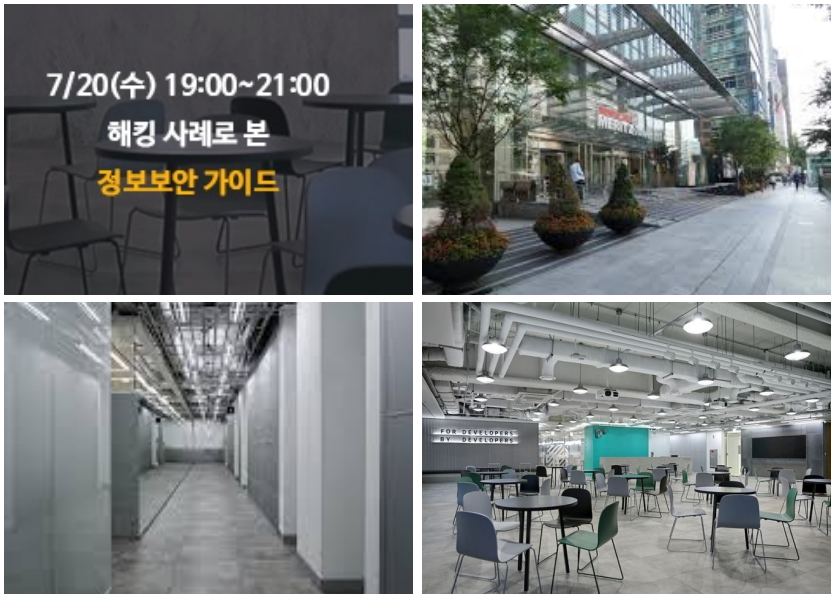
해킹 사례로 본 정보보안 가이드 강연 요약

2016년 7월 20일 수요일 19시 부터 21시 까지 진행한 '해킹 사례로 본 정보보안 가이드' 강연의 필기를 정리하였습니다. 강연을 참석하게 된 계기는 강연 신청 시 사전 질문을 등록할 수 있어서 강연 내용 뿐만 아니라 궁금했던 평소 궁금했던 것 까지 얻어 갈 수 있지 않을 까하는 기대감에 퇴근 후 강연 장으로 곧장 향했습니다. 강연 자는 네이버 비즈니스 플랫폼 IT 보안대응실 '황성호' 님이 나오셨습니다.

- 참석 목표 : 개발 보안에 관련된 내용을 듣고자 함.
- 참석 신청서 제출 시 전달한 사전 질문 : 네이버는 개발 보안 시에 개발 프로세스에 맞춰서 보안 관련 테스트 케이스를 어떻게 도출하고 정의 하는 지 궁금합니다. 주로 프로그램 개발 후 해당 프로그램에 대한

‘대응 방법을 미리 마련해 두시고 피해 최소화에 집중하세요.’

-NBP 황성호



취약성 검토를 수행하는 데 초기 소프트웨어 개발 프로세스에서 보안 문제를 확인하고 해결할 수 있는 방법이 알고 싶습니다.

현재 진행하고 있는 이베이 프로젝트에서 ‘Fortify’를 이용한 소스코드 진단을 수행하게 되었습니다. 책에서 ‘하나의 버그를 고치기 위해서 한줄을 수정하면 세 개의 버그가 새롭게 생겨난다’ 는 말을 본적이 있습니다. 그래서 네이버에서 비용을 최소화하기 위한 개발 초기 단계에서의 보안을 위한 노력을 수행하고 있는 지와 하고 있다면 어떻게 하고 있는 지가 궁금했습니다. 결론 부터 말하자면 질문이 선정 되지 않아 답변을 들을 수 없었습니다. 여러 분야의 청중을 생각해서 기술적인 내용을 많이 포함하지 않을 것으로 보입니다. 강연자가 말한 핵심 내용은 다음과 같습니다.

- 한 줄 요약: 대응 방법을 미리 마련해두시고 피해 최소화에 집중하세요.

아래는 강연을 들으며 필기한 내용입니다. 필기 내용은 IT 보안의 전반적인 내용으로 기술의 깊이가 있는 내용이 아니라는 것을 알려 드립니다.

- 강연 주제: 안전한 Service 운영을 위한 보안 교육
- 강연 목표: Service 최소한의 노력으로 최대의 보안 효과 누리기

외부 위협 유형

네이버에서는 공격을 위협, 취약성, 위험의 3 가지 유형으로 분류합니다.

위협은 사건(이벤트)으로 부정적인 영향을 줄 수 있는 이벤트입니다. 취약성(취약점)은 속성이나 상태로 정적으로 영향이 미비한 약한 취약점 입니다. 위험은 침해 당할 가능성(확률)이 존재하는 것으로 확인 결과 실제로 시스템에 영향이 있는 것입니다.

최다 발생 위협

네이버로 들어오는 공격 유형 중 가장 큰 비중을 차지하는 공격은 데이터베이스 공격 입니다. 다음은 최다 빈도를 기록하는 공격들 입니다.

1. Mysql Login Success

: 계정 설정 및 계정 정책 방안

데이터 베이스를 노리는 공격이 가장 높은 빈도로 발생하고 있습니다. 데이터베이스 보안에서 기술적인 부분은 대부분 해결되었다고 볼 수 있으나, 항상 문제가 되는 것이 기본 계정으로 인한 문제입니다. 계정 설정 및 계정 정책 방안을 수립하고 운영하는 것이 침해사고를 예방하는 데 많은 도움이 됩니다.

2. WP Pingback

: 디도스 공격 예방 (시큐어 코딩, 취약점 패치)

디도스 공격은 완벽한 해법이 존재하지 않기 때문에 평소 훈련에 의한 빠른 대응이 중요합니다. 소규모 서비스는 디도스의 피해를 최소화하는 데 목표를 두고 대응 방안을 세우는 것이 현실 적입니다.

3. Wget Execute Success

: Application 설정 (웹 어플리케이션 설정, 에러 출력 설정, Trace 정보 출력)

각 어플리케이션의 설정 만 완벽히 해두어도 공격의 대부분을 방어 해낼 수 있습니다. 서버 증축과 같이 새롭게 어플리케이션을 설치해야 하는 경우를 대비해서 미리 설정 자동화 스크립트를 작성해 두거나 표준 config 파일을 만들어서 사용하게 합니다.

4. Account

: Application Logic(SQL Injection, File Upload)

SQL Injection 공격은 XSS 다음으로 많이 발생하는 공격으로 항상 경계하고 위협에서 나기 위해 노력합니다. File Upload 공격의 경우 잘 발생되지 않지만 그 피해가 매우 크기 때문에 차단과 탐지를 위해서 노력을 기울이고 있습니다. 이 두 공격을 비롯한 어플리케이션 로직 공격 방어는 파라미터 검증에 있습니다. 각 파라미터의 용도를 확실히 하고 그에 따른 허용 문자를 정의 하여 개발하게 합니다.

5. Intrusion

: Netbios, wget, webshell, Malware

멀웨어나 웹쉘이 발견되었다는 것은 이미 공격에 성공했을 수 있습니다. 그에 따른 영향을 파악하는 데 집중합니다.

6. Vulnerability

: patch, update(Apache, Tomcat, Struct 취약점)

현실적으로 최신 버전을 유지한다는 것은 불가능하지만 반드시 패치를 수행해야 하는 경우를 대비해서 프로세스를 최적화하는 것이 중요합니다. 네이버도 꾸준한 노력을 통해서 프로세스를 개선했고 그 결과 이제는 전 서버를 패치 하는 데 3 일이면 가능하게 되었습니다.

계정 관리

설치 시 제공되는 기본 계정을 변경 없이 사용하거나, 유추 가능한 문자열로 비밀번호를 설정하여 계정 탈취 위협 발생. 대응 책으로 안전한 비밀번호의 사용 권고하지만 생각보다 쉽지 않다. 업무의 효율성을 위해서 쉽게 만들고 여러 사람이 공유하게 된다.

1. 패스워드는 최소 9자 이상
2. 대문자 소문자 특수 문자
3. 흔히 사용하는 문자와 숫자 조합은 피해라
4. 60 ~ 90일 간격으로 변경해라
5. 잘못된 패스워드로 인증 시 계정 잠금을 해라
6. 이메일 혹은 제 3자에게 공유하지마라

최근 사내 보안 이슈

1. 젠킨스 보안

외부로 오픈 되어 있지 않아 패스워드를 설정하지 않은 상태로 운영하는 젠킨스 서버들이 있었습니다. 젠킨스에 원격 커맨드 명령을 통해서 소스코드를 내려받을 수 있는 취약점이 존재하여서 이를 이용하여 웹으로 침투 후 젠킨스로 소스코드 무단 다운로드가 가능하여 사내 젠킨스 서버를 모두 재 점검하는 등의 조치를 한적이 있습니다.

2. 깃 보안

소스 저장소에 소스를 업로드 시 Public Git으로 업로드하여 서버 정보와 소스를 노출할 뻔한 적이 있습니다.

3. 구글 독 보안

문서 공유 및 작성 시 구글 문서를 사용하는 데 작성 시 Public으로 작성하여 내부 정보를 노출의 위험이 발생한 적이 있습니다. 문서 관리 환경이 변화함에 따라 새로운 표준안이 필요로 되고 있습니다.

DDoS 공격

디도스는 완벽한 해법이 아직 존재하지 않습니다. 패킷 자체는 잘못된 패킷이 아니기 때문에 공격성 판단이 어렵기 때문입니다. 네이버에서 내린 결론은 '즉시 대응이 답이다' 입니다. 네이버는 평소에 디도스 유사 패턴 발생 시 방어하기 위한 각 부서의 행동 가이드라인이 정해져 있습니다. 디도스 방어의 최선책은 비싼 장비 구입이 아니라 이와 같은 사전 준비와 훈련에 있습니다.

이 밖에도 평소에 각 서비스의 최대 트래픽이나 최대 커넥션 수를 파악해두고 서버 모니터링 도구를 통해서 안전 수치를 위협하는 통신량의 발생 시 알림을 발생시키고 서버 설정을 변경하는 등의 대비책이 필요합니다. AWS의 경우 아마존 서버를 보호하기 위해서 막지만, 서버 보안 자체는 지원해주지 않습니다. 현재 서버 이용자의 각자 책임이라는 정책을 가지고 있습니다. KISA를 활용하는 것이 가장 좋다.

어플리케이션 로직

이제는 모바일이 웹 보다 더 큰 플랫폼이 되었다고 해도 과언이 아니지만 모바일 자체 취약점의 경우 대부분 특정 동작을 분석해서 크랙하기 위함이 크다고 볼 수 있습니다. 외부로 향하는 취약점이 아니기 때문에 네이버에서는 아직까지 웹 취약점 방어에 주안점을 두고 있습니다. 발생 빈도와 중요도를 기반으로 어플리케이션 로직 공격을 알아보겠습니다.

1. XSS

가장 많이 들어오는 공격으로 실제로 확인해 보면 방학을 맞은 중학생 부터 전문 해커까지 다양한 사람들이 시도할 만큼 널리 퍼진 공격입니다. 하지만 실제 영향은 생각했던 것 보다 클 수 있으므로 항상 경계하고 있습니다. 실제 사례로 XSS으로 가로챈 인증 정보를 이용하여 네이버 페이에서 상품을 구입한 사례가 있습니다.

대응 방법 :

- 사용자 입력 값 한정 : 파라미터로 들어오는 사용자 값에 대한 철저한 검증
- 특수문자가 필요한 경우 : HTML Entity 처리, URL Encoding
- 추가로 할 수 있는 것 : X-XSS-PROTECTION Header or HTTP Only Cookie

2. SQL INjection

두 번째로 많이 발생하는 공격은 SQL Injection 공격입니다. 해당 공격은 주로 자동화 도구를 이용하여 발생하며 최근에는 SQL Injection을 고려한 개발이 정착화되어서 잘 발생하지는 않지만, 발생할 경우 피

해가 매우 크기 때문에 새로운 서비스 개발이나 업데이트가 발생할 때는 내부에서 꼼꼼하게 확인하게 됩니다.

대응방법 :

- Sysadmin (SA) 권한 미 사용
- 허용하지 않는 형식의 값 제한
- 모든 입력 값은 인용부호 (' 또는 ")로 묶음
- 불필요한 프로시저 사용 제한
- 입력 값에 대한 검증 로직 추가 (‘, “, \,), , , ;, union, =)
- 프로시저 또는 PreparedStatement 사용

3. HTTP Method

클라이언트단과 서버단(RESTful)을 완벽히 구분해서 개발하는 요즘의 개발 추세에 더 많이 발생합니다. 예를 들어 PUT 메소드와 SQL 쿼리를 매칭하여 사용하는 경우나 클라이언트 단에서 확인을 위해 Trace 메소드를 이용하여 상세 서버 정보를 확인하던 개발 환경 그대로 운영을 시작하는 등의 이슈가 있었습니다.

4. 서버 설정

- **Directory Listing**
- **서버 정보 노출 (ex. /phpinfo.php, /server-status, /jkmanager)**
- **오류 페이지 노출**

간단한 서버 설정으로 예방할 수 있는 취약점 입니다. 사전 교육을 통해서 서버 증설 시 실수하지 않게 하며 표준 템플릿을 구성하여 모든 서버가 동일한 보안 수준을 유지할 수 있게 합니다.

5. 인증 우회

시큐어 코딩을 하지 않은 경우 보다는 설계 상의 실수로 인해 많이 발생되는 취약점 입니다. 서버에서 발행한 세션만 참고하고 사용자 입력 값으로 권한을 판단하지 않습니다.

6. File Upload 취약점

침해사고의 40% 이상을 차지하는 치명적인 취약점 입니다. 파일 업로드 로직에서 파라미터 검증을 통해서 막습니다. 혹시나 서버 내 이미 업로드 된 웹쉘은 존재하지 않는 지 정기적으로 확인하고 웹쉘이 존재한다면 이미 침해사고가 발생된 것으로 유출된 정보 파악과 2차 침투를 막기 위한 대비를 합니다.

결론

공격할 빌미를 주지 않는 것이 가장 중요합니다. 특수한 목적을 가지고 접근한 경우를 제외하고는 공격자에게는 수많은 대상이 존재합니다. 보통은 서버의 정확한 설정만으로도 현재 서비스로 시도되는 공격의 반을 줄일 수 있습니다. 공격의 여지를 제공하여 대상이 되는 일이 없도록 합니다.

시스템 취약점

실제로 시스템 취약점을 이용하여 들어오는 공격은 기술의 깊이가 깊어서 공격 여부를 알고도 막지 못하는 경우가 있습니다. 이와 같은 시스템 취약점의 방어는 신속한 패치가 가장 적절한 대처 방법입니다. 신규 시스템 취약점을 모니터링하고 그 중에서 서비스와 관련된 시스템 취약점이 있다면 패치를 수행합니다. 하지만 서비스의 규모가 큰 경우 패치 완료 까지 많은 시간이 소요됩니다. 패치를 위한 업무 프로세스를 개선하여 (네이버는 전 서버의 패치를 완료하는 데 까지 3일의 시간이 소요된다고 합니다.)

Unknown Communication

명확히 목적이 정해져 있어서 사용되는 통신 혹은 프로그램이 예측 가능한 PC, Server의 경우 비교적 관리가 쉬운 편입니다. 하지만 그외 업무용 PC의 경우 정확히 판단할 수 없는 네트워크 트래픽이나 알지 못하는 프로그램이 설치 되는 등의 일이 발생하는 데 이를 어떻게 처리할 것인지에 대한 명확한 가이드 라인을 세우는 것이 필요합니다. 현재 네이버 또한 이 문제로 많은 회의를 진행하고 있으며 평소에는 보지 못한 로그가 발생하면 해당 정보를 확인하면서 최대한 파악해 나가고 있습니다.

For Safety Service

서비스의 안전을 위한 방법은 끝이 없지만, 적은 비용으로 최대 효과를 누릴 수 있는 4 가지 방안을 알려드리겠습니다.

1. 기술적 보안에 대한 관심

개발자들이 관심을 가지면 대부분 쉽게 해결할 수 있습니다.

2. 업데이트 확인 / 패치

주기적인 패치로 대부분의 시스템 취약점 공격을 막아낼 수 있습니다.

3. 아이디 / 패스워드 관리

회사 내 원칙을 세우고 관리하여 취약한 계정을 없애는 것이 중요합니다.

4. 서비스에서 사용하는 언어 / 프로토콜 / 시스템에 대한 정확한 이해

PC 나 Server의 용도를 명확히 규정하고 그에 필요한 구성요소를 제외한 불필요한 프로토콜이나 서비스를 사용 중지시키고 KISA에서 공지한 표준 서버 설정에 따라 서비스를 안전하게 운영할 수 있습니다.