웹 어플리케이션 보안성 검증방법

노 시 춘* · 전 익 수** · 김 귀 남***

요 약

오늘날 인터넷 상에서 web 시스템 취약점을 이용한 해킹이 점차 증가하고 있는 추세이다. 이 같은 위협에 대처하기 위해 web을 통한 위협의 종류와 그 대응 방안을 소개하고 web 어플리케이션 보안성 검증방법을 제시한다.

Verification Method of Insuring Web Application Security

SiChoon Noh* · IkSoo Jun** · Kuinam J. Kim***

ABSTRACT

Now a days the threatenings of using internet web system's vulnerability are rapidly increasing. To protect the threat we introduce the sorts of threats and present the verification method of web application security.

* 경기대학교 대학원 정보보호기술공학과
** ㈜코코넛 보안 컨설팅
*** 경기대학교 정보보호기술공학과

1. 서 론

이미 웹 서버는 인터넷 비즈니스를 위한 가장 핵심적인 인프라 중의 하나로서의 역할을 수행하고 있다. 이미 많은 사용자들은 인터넷 전용선, ADSL, Cable 등과 같은 통신 수단을 이용하여 웹 서버의 유용한 정보를 획득하고 간편한 금전 거래나 물품 구매 등 다양할 활동을 수행하고 있다. 이와 같은 활동을 위해서 사용자는 자신의 중요한 개인 정보를 웹 서버에 등록해야 하는데, 이때 입력되는 개인 정보는 인터넷 비즈니스의 가장 중요한 정보이며, 이 정보의 잘못된 사용은 사용자의 금전적인 피해 및 이를 이용한 제2의 피해가 발생할 수 있는 위험성을 항상 지니고 있다. 근래의 해킹사고 피해 사례를 보면, 과거에 흔히 볼 수 있었던 해킹 사례인 홈 페이지 위, 변조 등을 넘어서, 해킹을 통한 개인 정보의 획득 및 그에 대한 오,남용 문제가 더욱 심각하게 대두되고 있다. 이러한 문제는 인터넷 비즈니스에 심각한 영향을 미치므로 문제 해결을 위해 많은 방안이 제시되고 있는데, 본 글에서는 "웹 어플리케이션 보안성 검증"이라는 매우 실질적이며, 유용한 방법에 대해 소개하여 웹 어플리케이션을 구축, 운영하는 독자에게 실질적인 정보를 제공하고자 한다.

2. 웹 보안 위협

웹은 1990년 Berners-Lee에 의하여 처음으로 제안되었다. 그 후 Explorer, Navigator 등과 같은 웹 브라우저의 편리한 사용자 인터페이스가 등장하여 전문가들뿐만 아니라 일반인들의 인터넷 사용을 증가시켰고, 인터넷의 사용 규모를 계속해서 증가시키고 있다. 이와 같은 웹 사용자의 급속한 증가는 인터넷을 사용한 전자상거래까지 발전시키고 있으나 현재로서는 웹 보안이 웹 사용 및 발전의 커다란 걸림돌이 되고 있다. 또한, 유닉스의 취약점과 웹 서버, 웹 브라우저, 자바 CGI(Common Gateway Interface)등의 응용 프로그램에 존재하는 오류를 이용하거나 IP 위장(spoofing), TCP 용량 초과 공격(SYN Flooding), ICMP 폭탄 등과 같은 네트워크 프로토콜의 구조적 결함을 이용하는 방법 등이 웹 서버 해킹에 널리 이용되고 있다.

웹의 사용으로 발생할 수 있는 보안 위협 요소들을 이해하는 것은 이들에 대한 올바른 대책을 세우는데 있어 매우 중요하다. 웹 사용자들에게 잘 알려진 일반적인 위협에는 무결성에 대한 위협, 비밀성에 대한 위협, 서비스 거부에 대한 위협, 인증에 대한 위협등이 있다.

2.1 무결성에 대한 위협

무결성에 대한 공격이 성공하게 되면 다른 유형의 공격이 가능하게 되기 때문에 가장 치명적인 위협이다. 데이터, 프로그램, 전송중인 메시지, 메모리 등에 대한 악의적인 수정 등이 무결성에 대한 공격의 결과로 이루어진다.

이러한 위협에 대한 대책으로는 공격자가 호스트에 접근하지 못하도록 하고, 전송중인 메시지에 대해서는 메시지 인증 코드라 부르는 암호기술을 이용한 메시지 체크섬(checksum)을 이용하여 무결성 훼손 여부를 검증하는 방법이 있다.

2.2 비밀성에 대한 위협

비밀성에 대한 공격은 사용자 컴퓨터나 서버에 저장되어 있는 정보를 노출시킨다. 대부분의 컴퓨터 사용자들은 자신의 컴퓨터에 전화번호, 금융 정보, 스케줄 등의 개인 자료를 저장하며, 동시에 브라우저를 사용하여 접속을 한다. 그러나, 웹의 사용으로 이들 정보가 노출될 위험을 안고 있음을 대부분의 사용자들은 모르고 있다. 또한, 브라우저 자체도 사용자 컴퓨터에 프라이버시가 유지되어야 할 정보를 저장하고 있다. 예를 들

어, 웹 접속 속도를 높이기 위해 최근에 방문한 사이트를 로컬 캐쉬에 유지하도록 설정해 놓고 있는데, 이 캐쉬에 저장된 정보가 유출되면 사용자의 접속 습관에 대한 정보의 프라이버시가 훼손되는 결과를 낳는다. 웹 서버는 보통 패스워드를 사용하여 서비스를 제공 받을 사용자를 인증하는데 브라우저와 서버간의 메시지 트래픽이 암호화되어 있지 않으면 네트워크 도청에 의해 패스워드가 노출되고, 이를 획득한 자는 몰래 서비스를 이용할 수 있게 된다. 사용자 패스워드를 알아내지 못한 경우에라도 정당한 사용자가 서버에 요청한 정보가 브라우저로 전송될 때 중간에서 가로채서 그 내용을 볼 수도 있다.

2.3 서비스 거부에 대한 위협

서비스 거부 공격은 매우 방어하기 어려운 위협으로 시스템 자원에 대한 접근을 방해하는 악성 행위를 말한다. 공격자가 DNS 시스템의 IP 주소 매핑을 바꾸어 놓으면 호스트로 송수신되는 패킷은 바뀐 주소로 송신되어, 결국 네트워크에서 분리된 것과 같은 결과를 낳게 된다. 좀 더 쉬운 서비스 거부 공격으로는 무의미한 패킷을 무차별적으로 한 호스트에 전송하여 이 호스트의 CPU가 전송되어 온 이들 쓰레기 메시지들을 처리하느라고 대부분의 시간을 소비하게 만들어 시스템의 서비스가 마비되도록 하는 방법이 있다. 사실, 많은 서버들은 동시 접속 수를 제한하며, 공격자가 모든 가능한 접속을 연속적으로 사용하면 합법적인 사용자의 접속은 거부되게 된다. 특히, 정보의 안전한 전송을 위해 암호 연산을 수행하는 서버의 경우 서비스 거부 공격에 취약하다. 공개키 연산에 필요한 계산량은 엄청나므로 만일 어떤 호스트가 수신된 메시지를 처리하기 전에 전자 서명을 확인하는 암호 연산을 수행한다고 할 때, 가짜로 서명된 메시지를 연속적으로 보내어 호스트가 서명 확인에 모든 시간을 낭비하도록 할 수 있다.

2.4 웹 브라우저의 기능 확대로 인한 위협

초기의 웹 브라우저는 상업적 성공을 위하여 또는 보안사의 결함을 개선하기 위하여 계속적으로 새로운 기능들로 개선되어 왔다. 전자 우편이나 뉴스의 송수신 기능을 브라우저에 통합한 것도 이러한 기능 개선의 한 예이다. 사실, 이러한 브라우저의 성능 개선을 누가 먼저 해내는가가 그 동안 개발되었던 수 많은 웹 브라우저의 상업적 성공으로 귀결되었고, 여기서 살아남은 브라우저는 넷스케이프사의 네비게이터와 마이크로소프트사의 인터넷 익스플로러가 대표적이다.

대부분의 웹 사용자들은 전자우편이나 뉴스를 별도의 전용 프로그램을 이용해 다루지 않고, 웹 브라우저로 처리하는 경우에 보안상의 위험이 발생할 수 있다. 예를 들어, 어떤 해커가 자바의 알려지지 않은 보안 결점을 발견하여 이를 이용하여 악의적인 행위를 실행하는 자바 애플릿을 작성하고, 이를 부르는 HTML 페이지를 만들어서 전자우편으로 송신했다고 가정할 때, 메시지를 받은 자가 웹 브라우저로 전자우편을 보기 위해 무슨 내용인지도 모른 채 자신의 컴퓨터 상에 임의의 프로그램을 실행하는 것과 마찬가지이다.

웹 기술이 날로 발전함에 따라 보다 강력한 기능을 가진 범용 브라우저가 만들어져 궁극적으로 개인용 컴퓨터 운영체제를 대체하게 될지도 모른다. 이렇게 되면 브라우저로부터 직접 실행되게 되며, 이는 곧 내용도 모른 채 프로그램을 실행시키는 것과 같은 보안상 아주 위험한 상황을 초래한다.

웹 브라우저에 의한 또다른 보안 위협은 웹 브라우저와 외부 표시기(external viewer)에 존재할 수 있는 보안 구멍(security hole)으로, 이는 치명적인 위협 요소로 작용할 수도 있다.

2.5 불법적인 웹 서버 설치에 의한 위협

웹 환경에서의 가장 커다란 보안 위협은 불법적인 웹 서버의 설치를 통한 공격이다. 즉, 누구나 쉽게 웹 서버를 설치할 수 있으므로 악성 서버를 설치하여 이 서버에 접근을 시도하는 사용자의 정보를 빼내거나 사용자의 시스템을 손상시키는 것이다. 이러한 위협은 악의적인 마음으로 장난스럽게 쓰이는 경우부터, 그럴 듯한 서버를 구축하여 사이버 공간상의 물건 구입이 가능한 것처럼 가장하고 사용자의 신용 카드 정보를 알아내어 컴퓨터 범죄에 악용하는 경우에 까지 이르고 있다.

3. 웹 보안 대책

웹 보안 위협에 대하여 완벽하지는 않지만 방어할 수 있는 여러 가지 대책이 있다. 기본적으로 웹 서버와 브라우저는 네트워크에 클라이언트-서버 관계로 접속되어 있으므로, 클라이언트 측면에서의 보안 대책, 서버 측면에서의 보안대책, 그리고 서버와 클라이언트 사이의 전송 보안 대책으로 구분할 수 있다.

3.1 클라이언트 측면에서의 보안 대책

클라이언트 측의 웹 요소는 브라우저이다. 브라우저는 사용자가 설정할 수 있는 옵션을 가지고 있다.

이 옵션에서는 클라이언트 컴퓨터의 보안과 관계된 것들이 많은데, 예를 들면 자바 스크립트 실행 가부를 선택하는 옵션, 쿠키(cookie)의 저장 가부를 선택하는 옵션 등이 있다. 이들 옵션은 대부분 보안성을 고려하지 않은 편익성 위주로 초기 설정이 되어 있다. 보안 취약성이 발견된 브라우저는 비교적 빨리 보완판으로 버전업(version-up) 되거나 패치(patch)가 발표되지만, 일반 사용자들은 이 새로운 보완판으로 자신의 브라우저를 신속히 업그레이드 하거나 패치를 설치하지 않고 그대로 사용하는 경향이 많다.

보안이 중요하다고 깨달은 때는 이미 보안 사고로 손실을 보고 후회하는 시점이다. 대부분의 사용자들은 자신에게는 보안 사고가 발생하지 않을 거라는 막연한 생각을 가지고 보안을 심각하게 고려하지 않는다. 따라서, 다음과 같은 보안 대책을 항상 적용하여야 한다.

(1) 방문 링크의 기록 유지
(2) 신뢰할 수 있는 도움 프로그램(helper program)
(3) 전자우편 자동 접속을 위한 암호 저장 금지
(4) 성능 향상을 위한 로컬 캐쉬의 내용 삭제
(5) 프락시 서버 설정 파일에 대한 불법 접근 방지
(6) 팝업 윈도우에 대한 "YES" 대답 성향 근절
(7) 쿠키에 의한 침해 예방을 위하여 쿠키 파일 삭제 또는 쿠키 경고 발생
(8) 정보 전송시 보안 경고 박스 표시
(9) 사용자 인증 강화
(10) 사용자 ID와 패스워드 보안 강화

3.2 서버 측면에서의 보안 대책

웹 서비스를 제공하기 위하여 웹 서버를 구축할 때, 관리자는 웹 서비스가 공격에 노출되지 않도록 서버 프로그램의 설정에 주의를 기울여야 하며, 서버가 설치된 호스트의 보안성을 강화하여야 한다.

특히, CGI 스크립트의 보안 관리에 주의하여야 하며, 자바 애플릿 실행을 감시하여야 한다. 뿐만 아니라 웹 서버에 대한 접근 통제와 무결성 보장도 필요하다. 이를 위해서는 다음과 같은 보안 대책을 적용 하여야 한다.

(1) 웹 서버 호스트의 불필요한 기능을 삭제하여 최소한의 기능만을 수행

(2) 수퍼 유저의 권한을 제한
(3) 인가된 사용자만이 인가된 데이터, 프로그램 및 장비에 접근 허가
(4) 사용자 인증 및 로그를 통한 책임 추적
(5) 주기적 자동 감사
(6) 보안 취약점의 신속한 공고 및 보안
(7) 복구에 대비한 백업
(8) 웹 서버의 구성 파일(inetd.cofig, httpd.conf, access.acl)의 옵션 설정
(9) 웹 서버 디렉토리 접근 통제
(10) 웹 서버에 대한 접근 기록 유지
(11) CGI의 보안 취약성을 고려한 서비스 구현
(12) JAVA 보안 취약서을 고려한 서비스 구현

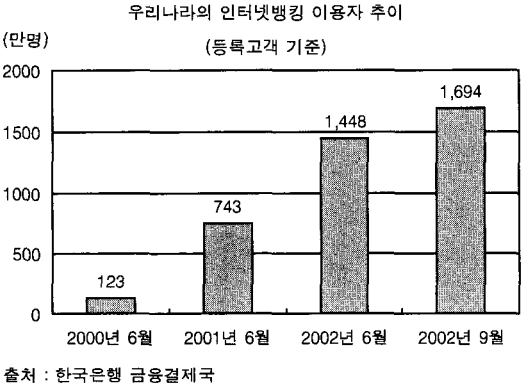
3.3 서버와 클라이언트 사이의 전송 보안 대책

브라우저가 서버에게 요청한 웹 페이지와 웹 페이지 내에 포함된 form에 정보를 기록한 다음 서버에게 리턴하는 경우, 서버와 클라이언트 간에는 데이터가 이동한다. 도청이나 변조를 방지하기 위하여 전송중인 데이터를 네트워크 레벨 또는 트랜스포트 레벨에서 암호화 시켜야 한다. 이를 위하여 제안된 대표적인 표준으로는 S-HTTP (Secure Hyper-Text Transfer Protocol), SSL (Secure Socket Layer), TLS(Transport Layer Security)등이 있다. 서버와 클라이언트 사이의 전송 보안을 위해서는 SSL을 이용한다.

4. 웹 어플리케이션 보안성 검증

4.1 필요성

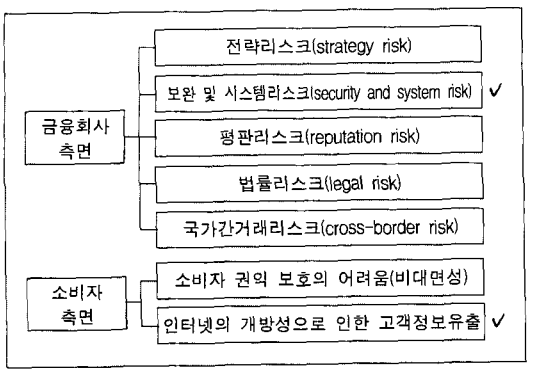
인터넷 뱅킹, 인터넷을 통한 증권거래 및 온라인 보험 가입과 같은 인터넷을 통한 금융 서비스는 전통적인 점포 창구를 통한 대면 업무에 비해 이용의 편리성, 높은 시스템 안정성, 빠른 거래처리속도 및 저렴한 비용 등의 장점으로 인해 서비스 이용자가 급격히 증가하고 있다. 현재 온라인 증권거래는 증권사의 서비스 채널 가운데 가장 큰 부분을 차지(50% 초과)하고 있으며, 인터넷 뱅킹의 경우는 가장 비중이 높은 서비스 채널은 아니지만 서비스 증가율은 가히 폭발적이다. 더욱이 최근 신행된 은행권 주 5일 근무 등의 영향으로 이러한 추세는 꾸준히 지속될 것으로 전망된다.



(그림 1) 인터넷 뱅킹 이용자 추이

위와 같은 인터넷을 통한 전자금융업무는 기존의 금융업무의 틀을 유지하면서 인터넷과 IT 기술이라는 새로운 접근 채널이 추가된 형태이기 때문에 위협요소(risk factors) 또한 전통적인 위험요소는 그대로 존재하면서 인터넷과 IT 기술에 대하여 그 위험이 확대된 개념으로 이해하여야 한다. 전자금융업무로 인해 확대된 위협요소는 금융회사 측면과 소비자(사용자) 측면으로 구분이 가능하며, (그림 2)와 같이 나타낼 수 있다

(그림 2)의 위협 요소 가운데 보안 및 시스템 리스크는 해킹으로 인한 고객정보 유출, 시스템 손상 등의 가능성과 금융회사의 전산 시스템 설계 및 운영상 오류 동 시스템 개발 아웃소싱에 따른 문제로 표현 할 수 있으며, 이러한 문제로 인한 사고 사례는 다음과 여러 곳에서 찾아 볼 수 있다.



(그림 2) 전자금융거래의 위협요소

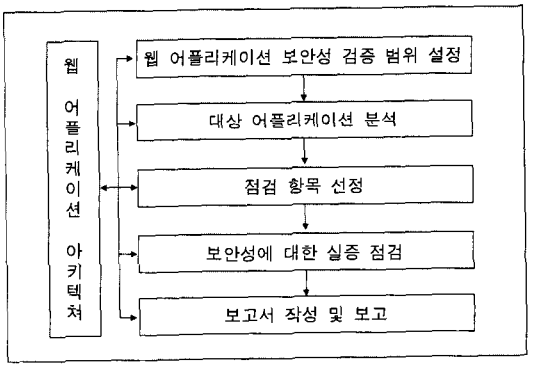
[사례 1] 경찰청 사이버 테러 대응센터는 고객의 사이버 선물계좌를 해킹하여 고객의 주식을 저가에 팔도록 조작한 뒤 자신이 사들여 고가에 매도주문을 내고 고객계좌에 다시 파는 방식으로 11억원을 챙긴 이모(29)씨 등 2명을 컴퓨터 등 사용 사기혐의로 구속(2001. 2. 14)

[사례 2] 서울경찰청 사이버 범죄 수사대는 웹(자바)기반 증권거래시스템에 대한 해킹 프로그램을 제작한 뒤 증권회사 고객 200여명의 계좌번호와 비밀번호를 파악하여 매수주문과 부당매매를 통해 시세를 조작한 강모(29)씨를 증권거래법, 정보통신망이용촉진 및 정보보호 등에 관한 법률위반 혐의로 구속영장 신청(2001. 8. 17)

위에서 제시한 예의 경우 인터넷 거래 시스템의 논리적, 기술적인 취약점을 이용한 대표적인 해킹 사례로 볼 수 있다. 두 사례 모두 사전에 웹 어플리케이션 보안성 검증을 거쳤다면 예방이 가능했을 것이다. 웹 어플리케이션 개발 및 운영 시에는 예상되는 오류 및 취약점에 대해서 금융감독 기관 기준에 따라 개발된 시스템인지 여부에 대한 검증 작업과, 시스템 개발 시 간과하기 쉬운 취약점을 실질적인 방법으로 직접 점검하고 그에 대한 대책을 세우는 작업이 반드시 요구된다. 많은 보안 담당자들이 방화벽 또는 IDS를 지나치게 의존하여 모든 보안 문제를 해결해 주리라 과신하고 있으나, 보안을 고려하지 않고 개발된 웹 어플리케이션을 통해서 정상적인 통로(예, 80 port)를 이용한 해킹이 가능하다는 것을 반드시 기억해야 한다.

4.2 검증 방법

웹 어플리케이션 보안성 검토는 웹 어플리케이션의 구조에 대한 정확한 이해 및 서비스 및 데이터 흐름의 정확한 이해를 바탕으로 체계화된 점검항목과 다양한 보안 기술의 결합을 통해 이루어진다.



(그림 3) 웹 어플리케이션 보안성 검증 서비스 흐름

4.2.1 웹 어플리케이션 아키텍처

웹 어플리케이션은 HTTP 프로토콜을 사용하여 사용자와 다른 시스템과 통신하기 위한 클라이언트/서버 소프트웨어를 지칭한다. 사용자의 편리성을 위하여 클라이언트에서는 대부분 Internet Explorer나 Netscape Navigator와 같은 브라우저를 사용하거나 자동화된 브라우저로 동작하는 HTTP 에이전트를 사용한다. 아직까지 웹 어플리케이션을 위한 많은 제품들이 수시로 개

발되고 있는 실정으로 웹 어플리케이션 구성을 위한 전체적인 구조에 대해서 혼돈이 많은 상태이지만 (그림 4)과 같이 표현된 내용은 가장 일반적으로 볼 수 있는 3단계의 논리적인 구조를 나타내고 있다.

- 표현 계층(Presentation Tier) : 사용자나 시스템에게 데이터를 표현하는 것을 담당
- 어플리케이션 계층(Application Tier) : 비즈니스 로직, 사용자 입력 처리, 결정 등을 위한 웹 어플리케이션의 엔진에 해당하는 부분
- 데이터 계층(Data Tier) : 어플리케이션에서 사용하는 임시 및 영구 저장소 역할을 위한 계층

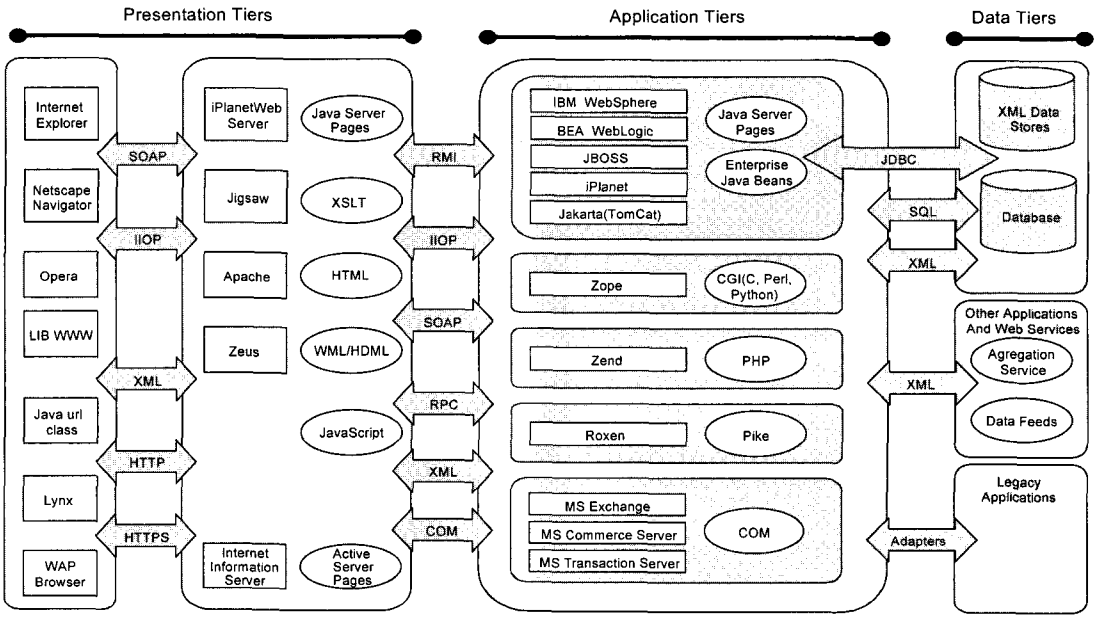
4.2.2 웹 어플리케이션 보안성 검증 범위 설정

웹 어플리케이션 보안성 검증은 관리적, 기술적, 물리적 보안성 검증으로 구분될 수 있으나, 물리적 보안성 검증은 큰 의미가 없어 주로 관리적 및 기술적 보안성 검증을 중심으로 수행된다. 관리적 보안성 검증은 프로젝트 검증과 보안 정책/지침/절차 검증으로 세분화 되어 질 수 있으며, 기술적 보안성 검증은 시스템 검증과 프로그램 검증으로 구분 할 수 있다. 이와 같은 기준을 기반으로 웹 어플리케이션 보안성 검증의 범위를 결정해야 한다.

4.2.3 대상 어플리케이션 분석

앞에서 설정된 검증 범위를 기반으로 하여 실제적인 보안성 검증 작업은 이루어 지며 그 첫 번째 단계는 대상 시스템에 대한 분석이다. 분석 단계는 보안성 검증을 위한 가장 중요한 단계로서 해당 서비스 흐름에 대한 이해, 웹 페이지에 구성되어 있는 컨텐츠의 상세 내역 및 상호 연결 관계와 정부기관 기준에 대한 요건 등의 내용을 파악해야 한다. .

1차적으로 컨설턴트는 해당 시스템 요건서, 아키텍처, 설계 관련 자료를 전달 받아 시스템 전



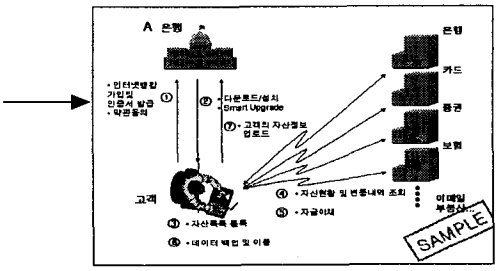
(그림 4) Web Application Architecture

반에 대한 업무 흐름 및 데이터 흐름에 대한 이해를 충분히 한 후, 시스템 상세 내용 정의를 위해 개발자 및 운영자와의 인터뷰, 웹 페이지의 상세 기능 분석을 통한 상세 서비스 흐름 파악을 수행한다. 또한, 관련 정부 기관의 서비스에 대한 요건을 정리하여 준거성 점검의 기본 자료로 활용하며, 대부분의 내용은 금감원에서 제공된다.

실제 이 과정에서 고객으로부터 제공 받아야 할 자료의 문서화 수준이 부족한 경우가 많으며, 대부분 컨설턴트가 일일이 모든 프로세스에 대한 실제적인 트레이스 및 그 과정에서의 데이터 흐름 파악이 필요하며 이 과정에 많은 시간이 소요된다.

대상 어플리케이션 분석 과정의 주요 산출물은 다음과 같으며, 이는 통제 및 검증 항목 선정에 활용되는 매우 중요한 자료가 된다.

- 시스템의 개요
- 서비스 메뉴 구조도
- 각 항목별 기능 설명
- 업무 흐름도



- 데이터 흐름도
- 서비스 내부 구성도
- 주요 정보에 대한 정의
- 정부 기관 요건 정의

4.2.4 점검 항목 선정

시스템에 대한 분석 결과를 기반으로 적용 가능한 점검 항목을 선정하는 단계로 이 과정은 기준비된 컨설팅 제공 업체의 체크리스트 내용과 분석 결과를 종합하여 진행된다. 당사의 경우 웹 어플리케이션 보안성 검증을 위한 체크리스트를 대항목 11개와 통제 사항 100여 개로 구성되어 있으며, 11개의 대항목은 다음과 같다(A 은행 인터넷 뱅킹 시스템 적용 체크리스트 기준임).

- 어플리케이션 구조

1. 어플리케이션 구조 (ARCHITECTURE)						
	통제사항	S	P	N	N/A	Comments
일반적인 고려사항	1.1 Public networks으로 공개되는 웹 어플리케이션과 관리용 어플리케이션과의 아키텍처 수준의 보안이 구분되어 개발되어졌는가? -관리용 어플리케이션이 따로 구분되어 개발되어 있지 않을 경우 어플리케이션의 컨텐츠 갱신 절차에 대한 보안성 검토를 수행한다.					
	1.2 컨텐츠 표현, 사용자 세션의 보안과 제어, 데이터 저장 서비스 및 보호에 관련된 3계층으로 적절히 분리되어 개발되어졌는가? - 3계층으로 구분되어 개발되어질 경우 외부의 악의적인 침입자로부터의 공격으로부터 단계별 보호를 받을 수 있다.					SAMPLE

- 인 증
- 사용자 세션 관리
- 안전한 네트워크 전송
- 접근 권한 및 인가
- 이벤트 로그
- 데이터 타당성 검토
- 웹 공통 문제
- 사용자 개인정보보호
- 암 호
- 대상 시스템 중점점검 사항

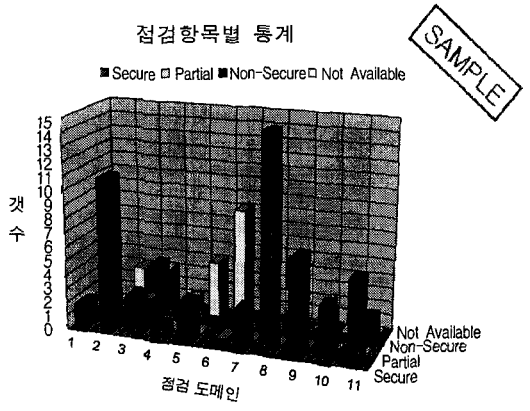
4.2.5 보안성에 대한 실증 점검

선택한 점검 항목을 대상으로 실증적인 점검이 진행된다. 이 과정에서는 웹 어플리케이션 구현상의 기술적인 취약점, 시스템 개발 요건에 따른 구현 여부 확인, 정부 기관 요건에 대한 부합 여부, 사용자 네트워크 및 내부 네트워크에 대한 보안 검증 등을 실제적인 방법으로 점검한다. 이때 사용하는 기법은 모의해킹에 사용되는 기술을 주로 사용된다. 하지만, 모의해킹이 해커의 관점에서 서버의 기술적인 취약점을 찾아 침입하는 것을 주로 한다면 웹 어플리케이션 보안성 검증의 경우는 모의해킹의 주요 부분을 모두 포

함하면서, 서비스 흐름 상의 논리적인 문제점 및 실증적인 방법을 통한 어플리케이션에 대한 감리에 대한 성격을 강하게 내포하고 있는 점이 차이점이라 할 수 있다. 실제 이 방법을 적용한 A은행의 인터넷 뱅킹 시스템의 경우를 보면 개발과정에서 고객의 보안 요구 사항이 제대로 구현되지 않은 경우와 미쳐 고려하지 않은 보안 취약점을 발견하여 사이트 오픈 전에 조치한 사례가 있다.

4.2.6 보고서 작성 및 보고

웹 어플리케이션 보안성 검증의 보고서 주용 내용은 상세 점검 내역, 통계 및 위험에 대한 대응책으로 구성된다. (그림 4)는 보고서에 포함되는 일부 내용을 나타낸 것이다(특정 사이트의 내용이 아님).



(그림 4) 보고서 내용 Sample

5. 결 론

많은 보안 컨설팅 산출물은 매우 일반적이고, 광범위하여 고객의 실질적인 보안수준 향상에 큰 도움을 주지 못하고 있는 점에 대한 지적이 많다. 하지만, A 은행 컨설팅 프로젝트의 일부분으로 진행했던 웹 어플리케이션 보안성 검증이라는 새로운 접근 방식은 매우 실질적이며, 구체적인 대안 제공이 가능하다는 큰 장점을 가지고 있다.

웹 어플리케이션은 인터넷 기반 비즈니스의 가장 핵심적인 자산이라는 것은 누구나 인지하고 있는 사항이다. 하지만, 보안을 고려하지 않은 시스템 개발, 개발 요건에 대한 실증적인 검증 및 정부 기관의 요건 반영과 같은 매우 중요한 내용을 검증하지 않고 웹 서버를 운영하는 경우가 대부분이다. 또한 방화벽과 IDS만 운영하면 이러한 문제점이 전부 해결된다고 생각하는 보안 담당자가 매우 많다. 하지만, 웹 어플리케이션 구현상의 오류는 해커들로 하여금 정상적인 서비스를 위해 방화벽에서 열어둔 80 포트를 이용하여 고객의 중요한 정보를 빼낼 수 있음을 잊지 말아야 한다. 따라서, 성공적인 인터넷 서비스를 위해서 웹 어플리케이션 보안성 검증은 실질적이며 절실히 요구되는 대안이라고 할 수 있다. 최근 금융감독 기관에서도 인터넷 뱅킹 시스템, 홈 트레이딩 시스템 등과 같은 전자금융 시스템에 대한 어플리케이션 보안성 점검의 필요성이 대두되고 있어, 향후 보안 점검의 새로운 항목으로 자리잡을 것으로 생각된다.

참 고 문 헌

[1] Matt Bishop, "Computer Security : art and science", Person Education, Inc. 2002.

[2] http://www.mailencrypt.com.

[3] http://www.ziplip.com.

[4] 배상우, "Application Hacking", 제3회 사이버테러 정보전 컨퍼런스, STG Security, 2003.

[5] 박성철, "안전한 프로그래밍 가이드", 제5회 해킹방지 워크샵, ㈜인젠, 2001.

[6] 이임영, "전자상거래 보안입문", 생능출판사, 2001.

[7] Festa, Paul, *Hotmail Flaw Exposes Pass-*

word, 19 Mar. 2001, http://news.cnet.com/news/0-1004-200-332525.html.

[8] Holmströrn, Ursula, *User-centered Design of Security Software*, Human Factors in Telecommunications, Copenhage, Denmark. May 1999, 18 Oct. 2000, http://www.tcm.hut.fi/Research/TeSSA/Papers/Holmstrom/htf99.ps.

[9] Oppliger, Rolf, *Security Technologies for the World Wide Web*, Boston : Artech House, 2000.

[10] Raskin, Jef, *The Humane Interface : New Directions for Designing Interactive System*, Reading : Addison Wesley Longman, Inc. 2000.

[11] Tiwana, Amrit, *Web Security*, Boston : Digital Press. 1999.

[12] Whitten, Alma, J. D. Tygar, *Usability of Security : A Case Study*, Technical Report CMU-CS-98-115, Carnegie Mellon University, School of Computer Science. December 1998, 18 Oct. 2000, http://reports-archive.adm.cs.cmu.edu/anon/1998/CMU-CS-98-155.pdf.

[13] Whitten, Alma, and J. D. Tygar, "Why Johnny Can't Encrypt : A Usability Evaluation of PGP 5.0", Proceedings of the 8th USENIX Security Symposium, August 1999, 12 Sept. 2000, http://www.cs.cmu.edu/~alma/johnny.pdf.

노 시 춘

1992년 고려대학교 경영대학원 경영정보학과 석사

2003년 경기대학교 대학원 정보보호기술공학과 박사과정

1980~현재 KT IT본부 수도권 전산국 부장

전 익 수

1988년~1995년 성균관대학교 정보공학과(학사)

1995년~2001년 LG CNS 네크워크엔지니어

2001년~현재 코코넛 보안 컨설팅팀장

김 귀 남

미국 캔자스대학 수학과(응용수학사)

미국 콜로라도주립대학 통계학과(통계학 석사)

미국 콜로라도주립대학 기계·산업공학과(기계·산업공학과박사)

현재 경기대학교 정보보호기술공학과 주임교수