3. 웹서비스 시 주의해야 할 관리상의 오류 분석 및 대책

1. 개 요

웹 어플리케이션의 보안사고 예방 및 재발방지를 위해 한국정보보호진흥원을 비롯하여, 수많은 기관이나 단체에서 보안 취약성에 대한 가이드 또는 기술지원이 제공되고 있다.
하지만, 기술적으로나 관리적으로 어렵고 깊이가 있는 보안 취약성에 대해서만 관심과 자원이 집중되는 경향이 있으나, 보안사고는 가장 취약한 부분에서 발생하기 때문에 아무리 많은 비용과 노력을 들이더라도 사소한 실수로 인해 치명적인 사고가 발생하곤 한다. 본 분석에서는 단순한 실수 또는 쉽게 범할 수 있는 실수들로 인해 초래되는 보안 취약성에 대해 운용 환경상의 문제와 개발상의 문제로 나누어 살펴보고 그 심각성에 대해 알리고자 한다.

2. 보안 취약성 노출 사례 및 대책

가. 운영 환경상의 보안 취약성 노출 사례 및 대책

운영 환경상의 보안 취약성 노출은 관리자나 시스템 운영자가 장비 또는 각종 서버들의 설정 오류나 관리상의 오류로 인해 보안 취약성이 노출되는 사례들을 분석하고 대책을 살펴본다.

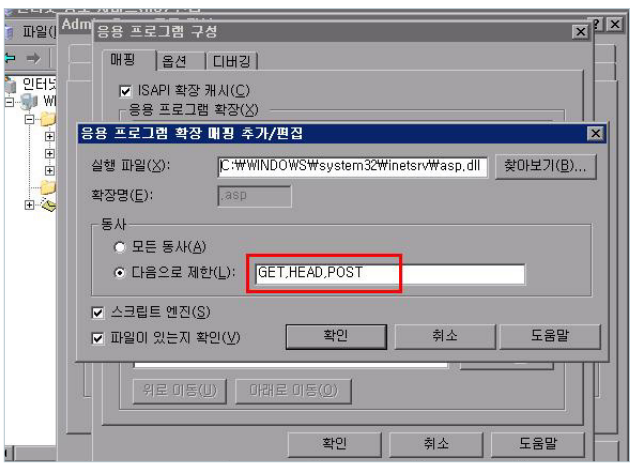
1) 웹 서비스 데몬(daemon)에서 불필요한 메소드 허용

APACHE 또는 IIS등과 같은 웹 서비스용 데몬에서 HTTP서비스에 대해 불필요한 메소드(method)들을 활성화 해 놓은 경우가 있다. 웹 사이트마다 사용하는 메소드만을 사용하여 필요 없는 메소드들로 인한 보안 취약성 노출에 주의해야 한다. 특히, HTTP 메소드들 중에서도 PUT, DELETE등은 외부에서 매우 손쉽게 웹 사이트의 변조가 가능할 정도로 매우 위험한 기능을 제공하고 있으므로, 충분한 검토가 필요하다.

```
HTTP/1.1
Content-Length: 0
Server: Microsoft-IIS/5.0
Date: Thu, 24 Apr 2008 04:40:36 GMT
Location: http://www.[    ].net
Allow: OPTIONS, TRACE, GET, HEAD, DELETE, PUT, COPY, MOVE,
PROPFIND, PROPPATCH, SEARCH, LOCK, UNLOCK
Connection: close
```

〈위험하게 운용중인 HTTP 메소드〉

이에 대한 대책으로는 서비스에 필요한 메소드만 활성화해야 한다. 실제로 대부분의 경우 GET, POST, HEAD 정도만 사용하고 있다.



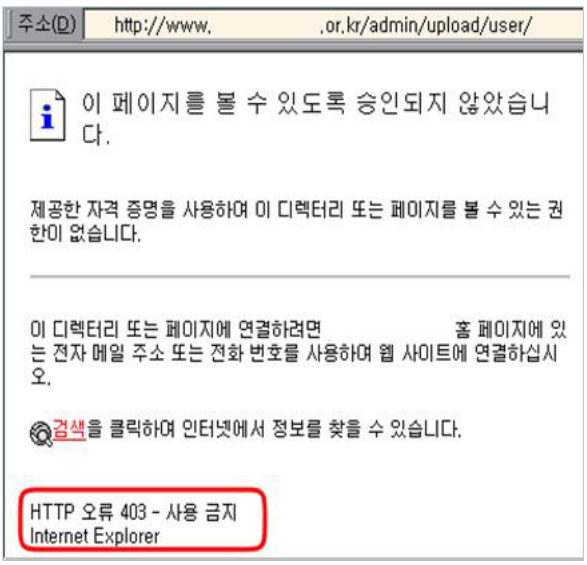
〈IIS에서의 HTTP 메소드 선택〉

2) 디렉토리 정보 출력으로 인한 서버 내부 정보 노출

디렉토리 리스팅은 웹 서버에서 디렉토리의 정보를 나열하여 개발이나 정보공유가 쉽게 가능하도록 제공하는 기능이다. 그러므로 이러한 디렉토리 리스팅은 반드시 필요한 경우가 아니라면 서비스 개시 전에 반드시 해제하여 서비스해야 한다.

또한, 디렉토리 리스팅 기능을 제한한 뒤에도 에러페이지 설정이 적절하게 되어 있지 않아 HTTP 오류 코드 "403 FORBIDDEN"이 표시되는 것도 공격자에게 디렉토리 존재에 대한 식별 정보가 노출 되는 것이다. 그러므로 미리 마련 해 둔 에러페이지를 표시하여 공격자가 디렉토리의 존재여부를 식별 할 수 없도록 해야 한다.

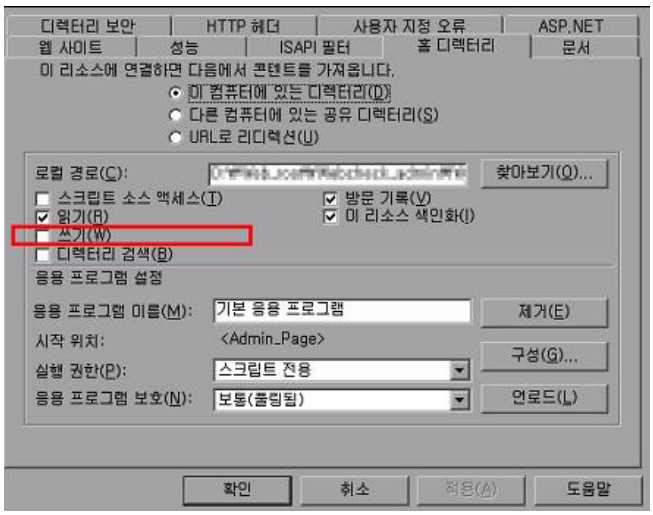
IIS의 경우에는 IIS 관리자 메뉴의 사용자 지정 오류탭을 이용하고, 아파치의 경우에는 conf 파일의 ErrorDocument 설정을 이용한다. 단, 에러페이지를 설정해도 HTTP 응답에 403 코드가 포함되어 있을 수 있으므로, 에러 페이지 설정 시 유의해야 한다. IIS의 경우엔 사용자 지정 오류탭에서 파일 지정이 아닌 URL로 설정하면 HTTP 응답코드가 200으로 반환되므로 공격자가 디렉토리의 존재여부를 식별 할 수 없다.



〈HTTP 오류코드 403으로 인한 디렉토리 정보 노출〉

3) 디렉토리 권한 설정 오류

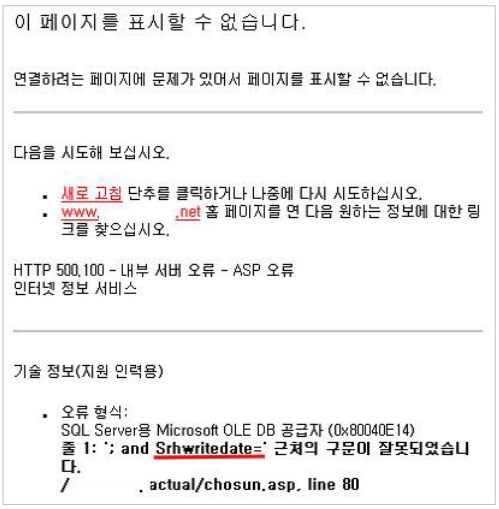
개발상의 편의를 위해서 웹 서비스용 디렉토리에 쓰기권한을 설정 해 두거나 WebDAV 등의 기능을 사용한 경우에는 반드시 서비스 개시 전에 해당 기능을 비 활성화 해야 한다.
디렉토리에 쓰기권한이 설정되어 있거나 WebDAV 기능이 활성화 되어 있다면 홈페이지 변조사고에 대해 매우 위험한 상태로 노출되어 있는 것이다.



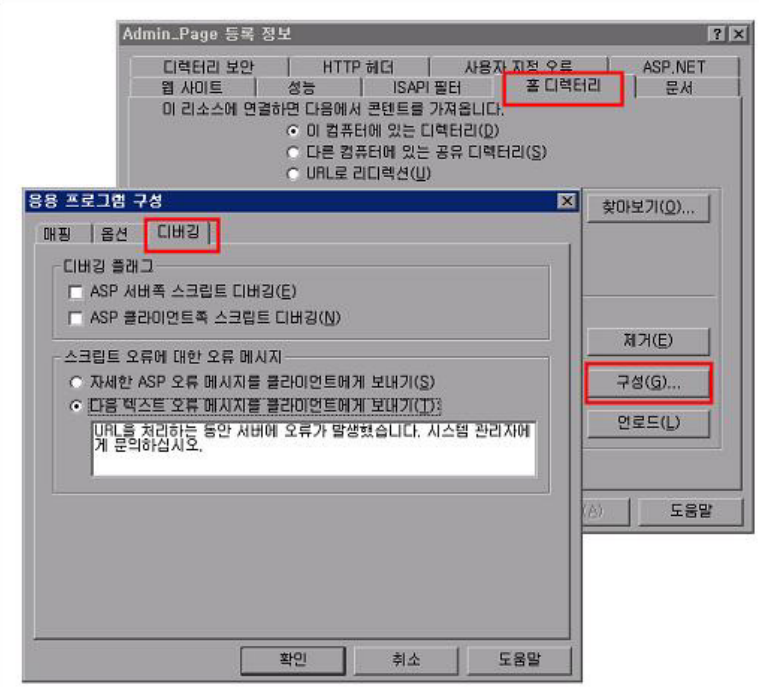
〈IIS에서의 디렉토리 쓰기권한 해제〉

4) 서비스 오류에 대한 상세한 정보 제공

일반적으로 기본적인 웹 서버 데몬의 설정은 오류가 발생 했을 때, 자세하게 오류의 내용을 사용자에게 표시하게 된다. 그러나 이러한 오류정보만으로도 공격자에게 유용한 정보가 될 수 있으므로, 오류정보 또한 지정된 에러페이지로 표시하여 공격자에게 불필요한 정보를 노출하지 않아야 한다.



〈오류 메시지를 이용한 DB Column 정보 수집〉



〈IIS에서의 오류 메시지 권장 설정〉

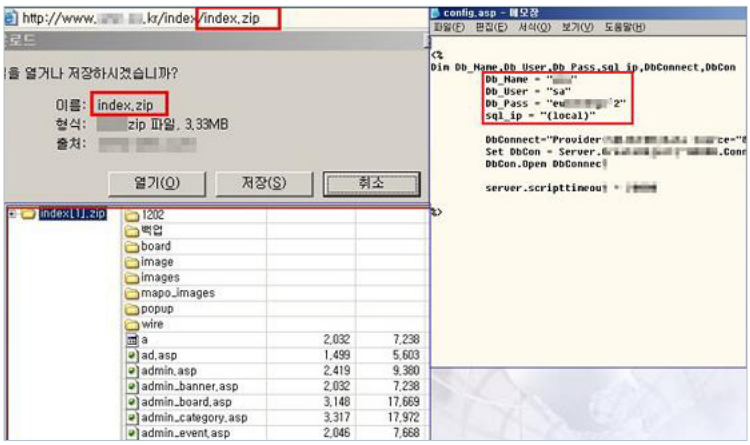
나. 개발상의 보안 취약성 노출 사례 및 대책

개발상의 보안 취약성 노출 사례에서는 개발자나 관리자가 웹 어플리케이션을 구현하면서 노출되는 보안 문제들을 분석하고 대책을 살펴본다.

1) 부주의한 디렉토리 압축파일 관리로 인한 보안 취약성 노출

일부 개발자나 관리자의 경우 웹 어플리케이션의 디렉토리나 특정 파일을 실제 웹 서비스 디렉토리 내에서 압축하여 백업 또는 전송 한 뒤 삭제 등의 관리조치가 이루어 지지 않아, 결과적으로 웹 어플리케이션의 소스코드나 환경설정 정보가 유출되는 사례가 많이 발견된다. 실제로 공격자들은 "/inc/inc.zip" 등과 같이 주요 디렉토리의 압축파일 존재여부를 확인해 본다.

아래의 그림은 특정 사이트에서 발견된 사례이며, 관리자가 부주의하게 디렉토리 압축파일을 남겨놓아 해당 웹 페이지 소스코드와 DB환경 설정 정보가 노출된 사례이다. 그러므로 실제 웹 서비스 디렉토리에서는 반드시 필요한 파일만 존재하도록 관리 하여야 하며, 특히 압축이나 관리상의 파일 관리는 웹 서비스용 디렉토리가 아닌 곳에서 이루어 져야 한다.

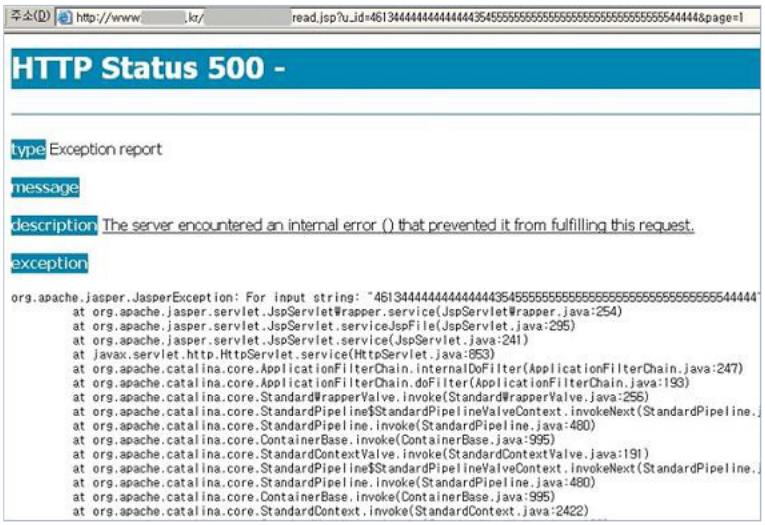


〈디렉토리 압축파일 관리 부주의로 인한 내부 정보 노출〉

2) 구현상의 자료형 설정 부주의로 인한 오류정보 노출

자바와 같은 일부 웹 개발 언어에서는 변수의 자료형(data type)에 대해 매우 상세하고 정확하게 정의하여 사용하도록 하고 있으며, 이에 따라 적절한 자료형이 아닌 경우에 대해서는 오류가 발생하게 된다. 공격자는 이러한 자료형의 오류를 고의로 유발하여 오류 메시지를 통해 서버의 내부정보를 획득 할 수 있으므로 주의 하여야 한다.

이에 대한 대책으로는 변수의 자료형을 정확히 정의하고, 입력 단계에서 처리할 수 있는 자료형인지에 대한 검증 절차가 필요하다. 아래의 화면은 정수형의 자료형에 대해 입력값의 크기 또는 길이(length)를 검증하는 절차 부재로 인해 발생한 오류화면이다.

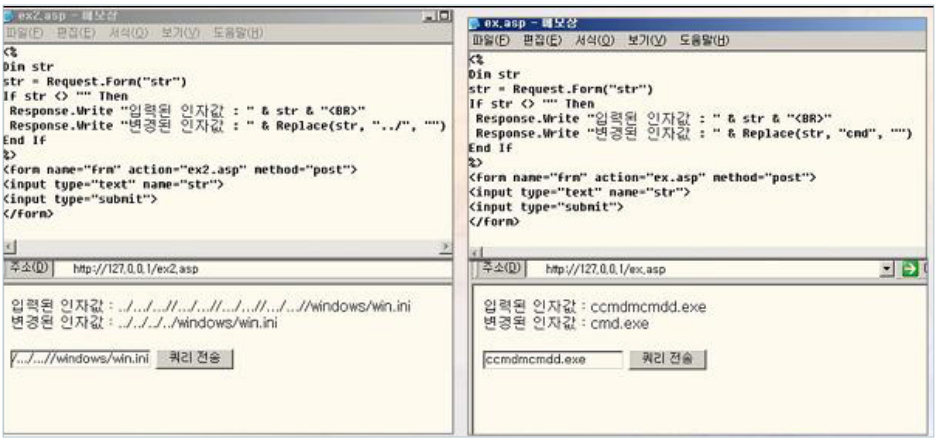


〈과도한 크기의 자료입력으로 유발시킨 오류 화면〉

3) 부적절한 입력값 필터링 문제

일반적으로 알려진 공격 문자열을 필터링 하기 위해 충분히 검토되지 않은 상태에서 적용하게 되면 충분히 우회 할 수 있는 경우가 발생할 수 있다. 아래의 화면은 보안수준 향상을 위해 개발자가 상위 디렉토리 경유 차단(../../../)과 공격 명령어(cmd)에 대해 필터링하였으나, 문자열 필터링을 우회할 수 있는 경우를 고려하지 않아 공격자가 우회할 수 있게 되었다.

이러한 경우, 가장 좋은 입력값 검증 방법은 처리 가능한 입력값의 범위와 유형을 정의를 한 후 그 외의 입력값에 대해서는 모두 거부하는 것이 바람직하다.

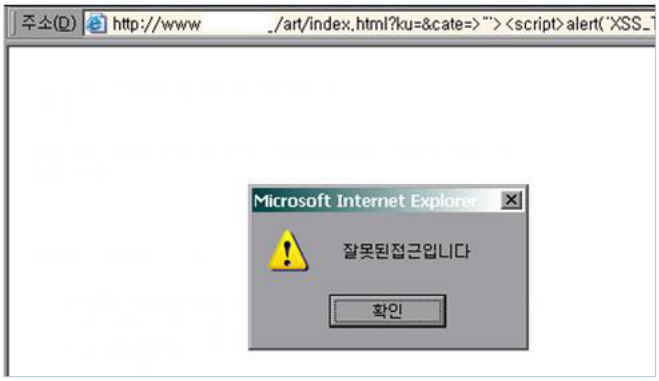


〈부적절한 문자열 필터링으로 인한 우회 공격 가능 사례〉

4) 부적절한 예외 처리 문제

입력받은 자료에 대해 처리할 수 없거나 거부해야 하는 경우에는 예외처리 실행 전에 반드시 해당 변수를 비우거나 초기화 한 후에 이전 페이지로 돌리거나(redirect) 에러 메시지를 표시해야 한다. 그렇지 않으면, 예외처리는 실행되지만 실제의 변수값은 그대로 서버에 전달될 우려가 있다.

아래의 화면은 실제 한 웹사이트에서 XSS 문자열을 입력할 경우 화면상에서는 적절하게 조치된 것으로 보이나 실제 소스를 살펴보면 XSS 문자열을 저장했던 변수가 초기화 되지 않은 상태에서 서버에 전달되어 그대로 XSS 공격이 반영된 사례이다.



〈XSS 공격 문자열에 대한 예외처리 화면〉

```
<script>alert('잘못된접근입니다');top.location.href = '/index.html';</script><html>
<head>
<meta http-equiv="content-type" content="text/html; charset=euc-kr" />
<title> ▮ ▮ ▮ ㄴ ' ▮ (  )</title>
<link href="../css/style.css" rel="stylesheet" type="text/css">
</head>
<body>
<table width="960" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td><img src="images/default_t1.gif" width="960" height="80"></td>
  </tr>
  <tr>
    <td><table width="100%" border="0" cellspacing="0" cellpadding="0">
      <tr>
        <td height="50" valign="middle" align="right"><a href="index.html?ku=&cate=>"'><script>alert('XSS
      </tr>
    </table></td>
  </tr>
  <tr>
...이하 생략...
```

〈예외 처리 후에도 서버측에 전달된 XSS 문자열 예시〉

[보완 코드의 예]

```
boolean getSUonlyNum(String str) {

    boolean isCheck = true;

    // NULL, 공백, 문자열 길이 점검
    if(str == null || str == "" || str.length()>5) str = "1";
    str = getReplace(str, " ", "");
    for (int i = 0; i < str.length(); i++)
    {
        char iChr = str.charAt(i);

        if( !(iChr>=48&&iChr<=57) ) {   // 숫자..
            str = "1";  // 초기화
            return false;
        }
    }

    return true;
}
```

〈적절한 예외처리를 구현한 소스코드 예시〉

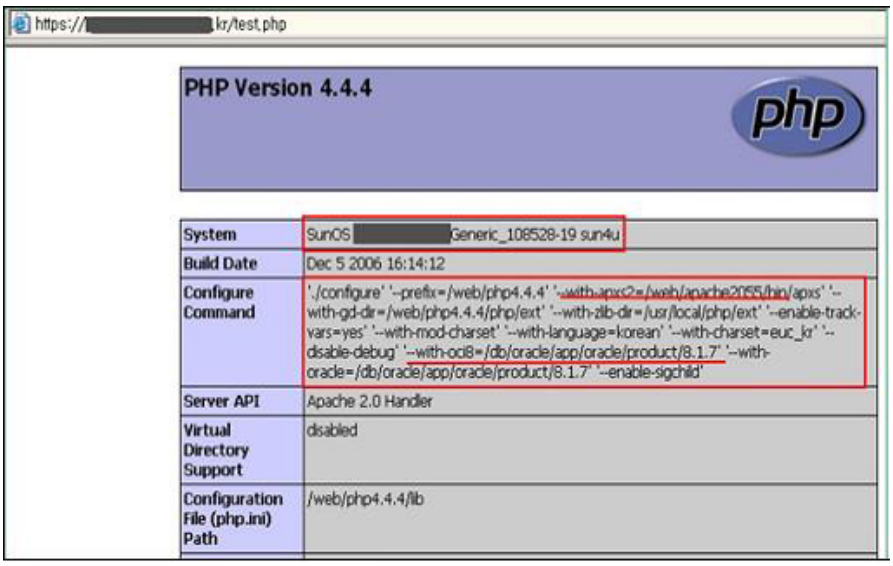
5) 부주의한 파일 확장자 변경으로 인한 소스 코드 노출

개발자나 관리자가 임시로 백업하려는 의도에서 웹 서비스용 파일의 확장자를 변경한 경우에 사전 정의된 확장자가 아니라면 소스코드가 그대로 노출될 우려가 있다. 그러므로 반드시 백업이나 임시파일은 웹 서비스용 디렉토리에서 수행되면 안되며, 파일 확장자 또한 임의로 변경하지 않아야 한다. 또한, 웹 서버 설정에서 처리 가능한 확장자를 지정하고 그 외의 확장자에 대해서는 처리 거부설정을 하는 것이 더욱 더 안전하다.

〈DB설정 파일의 확장자 변경으로 인한 정보 노출〉

6) 테스트파일 관리 부주의로 인한 정보 노출

개발 단계에서나 장애처리에서 흔히 사용되는 테스트 파일들은 반드시 목록화하여 관리하고, 실제 웹 서비스 디렉토리에 남겨져서는 안된다. 아래의 화면은 테스트파일 관리 부주의로 인한 서버 정보 노출 사례이다.



〈테스트 파일 관리 부주의로 인한 서버 정보 노출〉

3. 결론

끝없는 인터넷의 확장에 따라, 수많은 서비스들이 웹 사이트를 통해 이루어지고 있다. 이러한 최근의 정보통신 환경에서 성공적인 사업을 위한 위험관리도 많은 자원이 투입되고 있으며, 잠재적인 위험을 감소시키기 위한 보안수준 제고에도 많은 관심과 노력이 기울여 지고 있다.

하지만, 앞서 살펴본 사례들과 같이 아무리 훌륭한 보안장비나 인력을 투입하더라도 관리자나 개발자의 사소한 실수로 인해 웹 사이트가 심각한 위험에 노출될 수 있고, 이러한 위협은 결코 기술적인 깊이가 깊은 보안 취약점보다 위험도면에서 떨어지지 않는 경우가 많다. 그러므로 개발 단계 뿐만 아니라 운영면에서도 체계적인 관리 절차와 방법을 수립하고 지켜져야 할 것이다.