### Задание 19.6.1 (HW-03)

**Персонализация:** Михайлов Юрий Никитич (uramihajlov481@gmail.com)

**Лабораторная работа:** Задание**№**19.6.1 (HW-03)**:**

Общая информация:

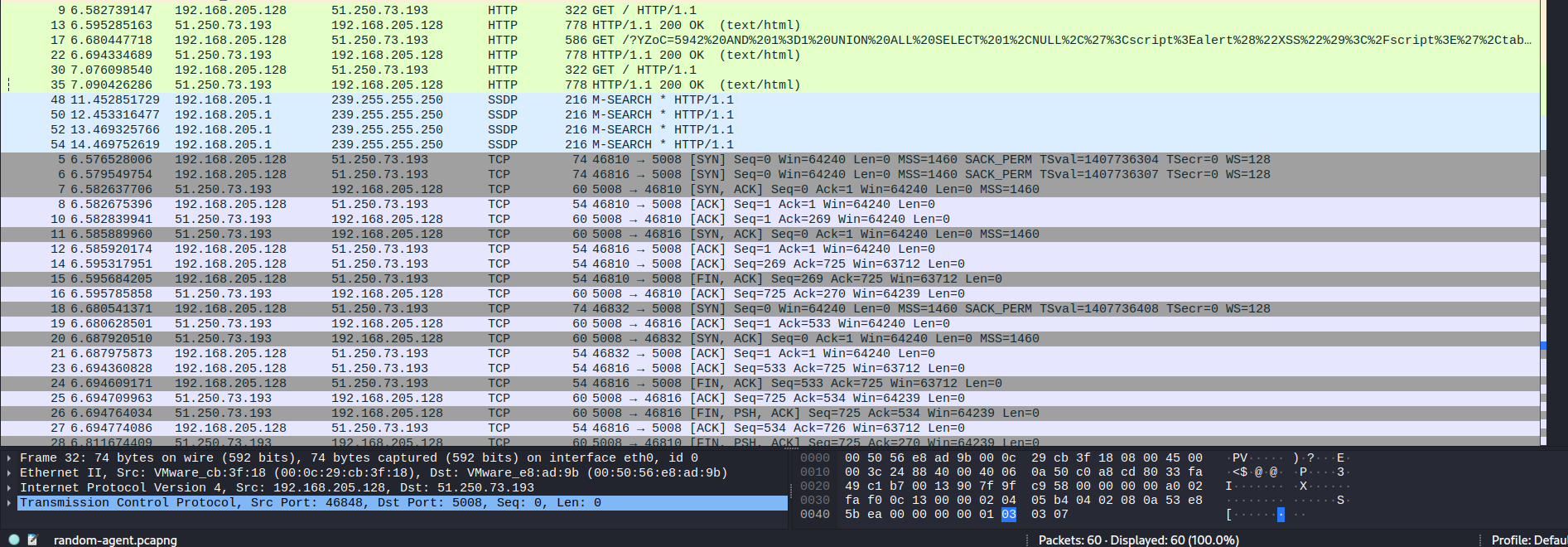
* Период тестирования: 28.03.2024
* Объект тестирования: http://51.250.73.193:5008
* Описание действий:

**Задачи:**Протестировать как можно больше ключей для sqlmap типа:

* **--random-agent**
* **--randomize**
* **--hpp**
* **--risk (от 1 до 3)**
* **--os-pwn**

**1) --random-agent**: Этот ключ случайным образом выбирает "поддельный" HTTP-агент для каждого запроса, чтобы обойти обнаружение ботов на целевом сайте.

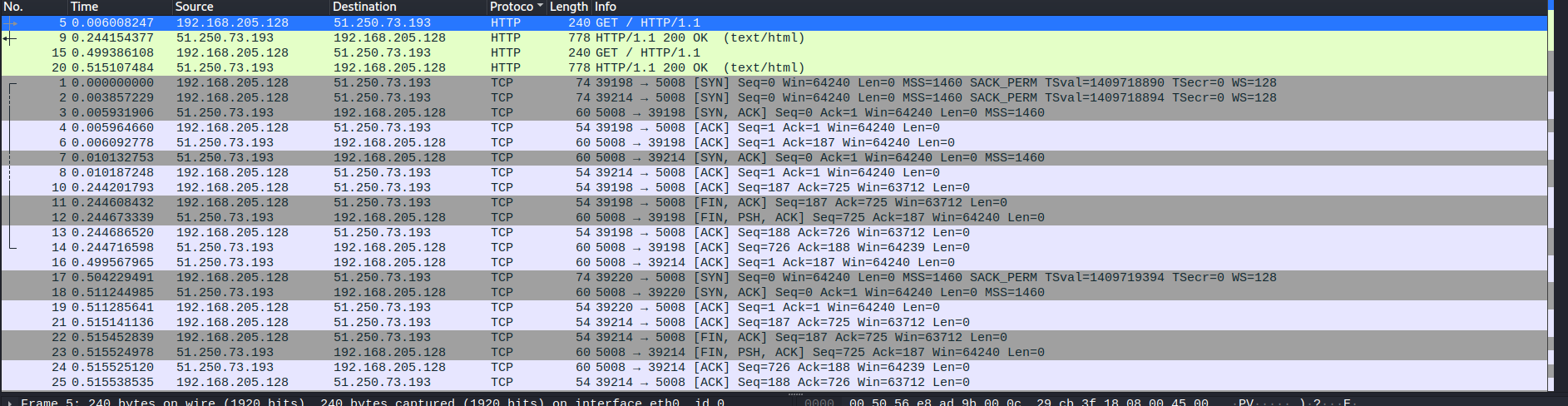
**sqlmap -u "http://51.250.73.193:5008" --random-agent**

****

**Мини вывод:** 3 запроса 3 ответа

**2) --randomize:** Этот ключ используется для случайного перемешивания значений в запросах и позволяет увеличить успешность обхода защиты от SQL-инъекций.

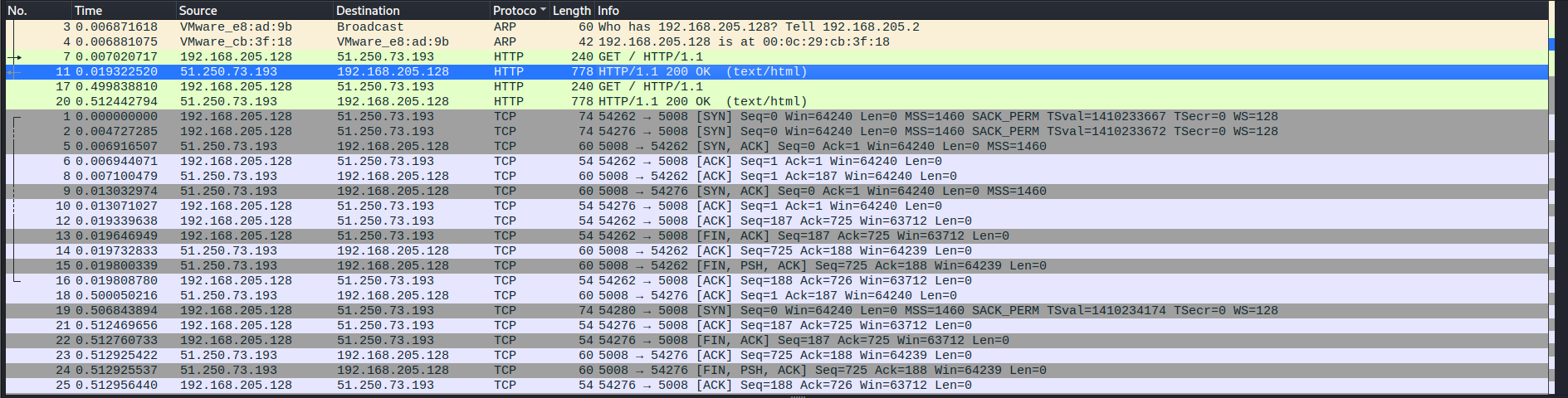
**sqlmap -u "http://51.250.73.193:5008" –randomize 8**

****

**Мини вывод:** 2 запроса 2 ответа

**3) --hpp:** Этот ключ используется для обхода защиты от атаки HTTP параметрами (HTTP Parameter Pollution), которая может применяться для предотвращения SQL инъекций.

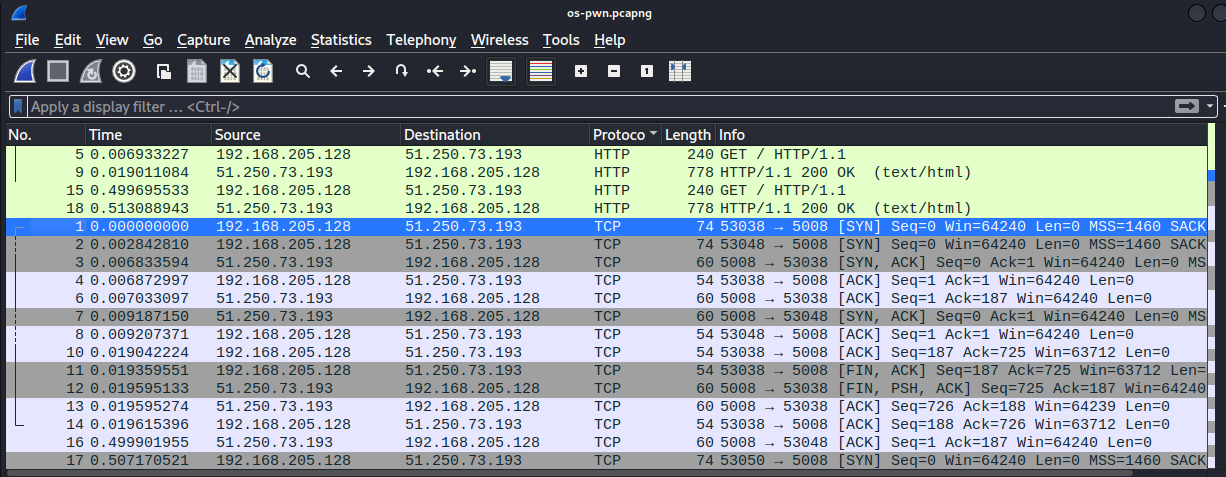
**sqlmap -u "http://51.250.73.193:5008" –hpp**

****

**Мини вывод:** 2 запроса 2 ответа

**4) --os-pwn:** Этот ключ пытается взломать целевую систему, используя SQLMap, предполагая, что уязвимость в базе данных может позволить вам получить контроль над операционной системой.

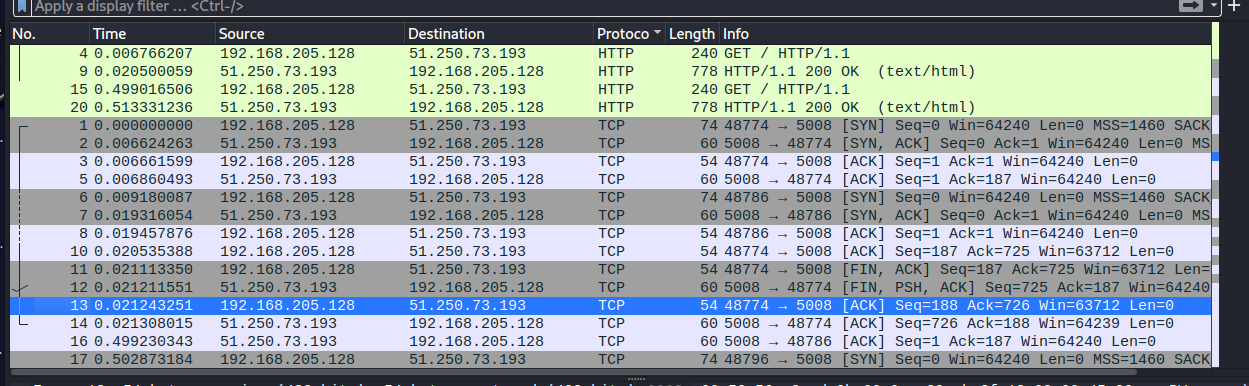
**sqlmap -u "http://51.250.73.193:5008" --os-pwn**

****

**Мини вывод:** 3 запроса 3 ответа

**5) --risk (от 1 до 3):** Этот ключ определяет уровень риска, который SQLMap готов принять при обнаружении уязвимостей SQL-инъекции. Чем выше уровень, тем более агрессивно будет проводиться тестирование, но и вероятность сбоя также возрастает.

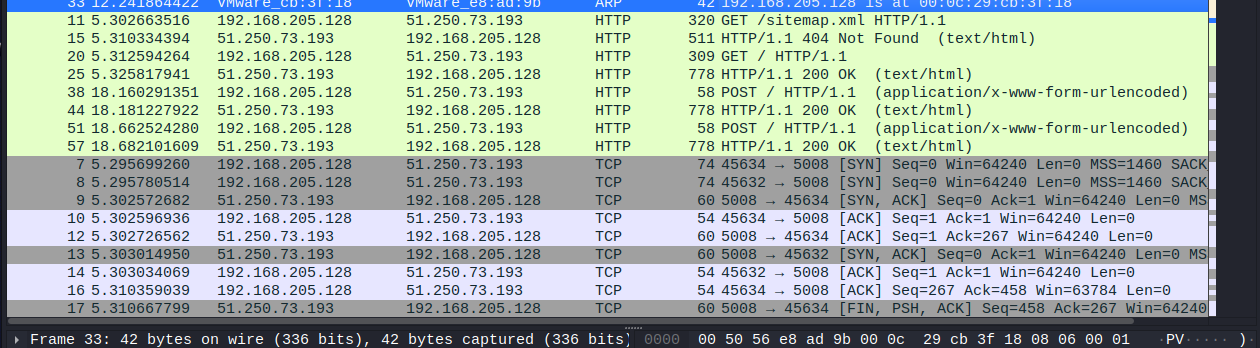
**sqlmap -u "http://51.250.73.193:5008" --risk 3**

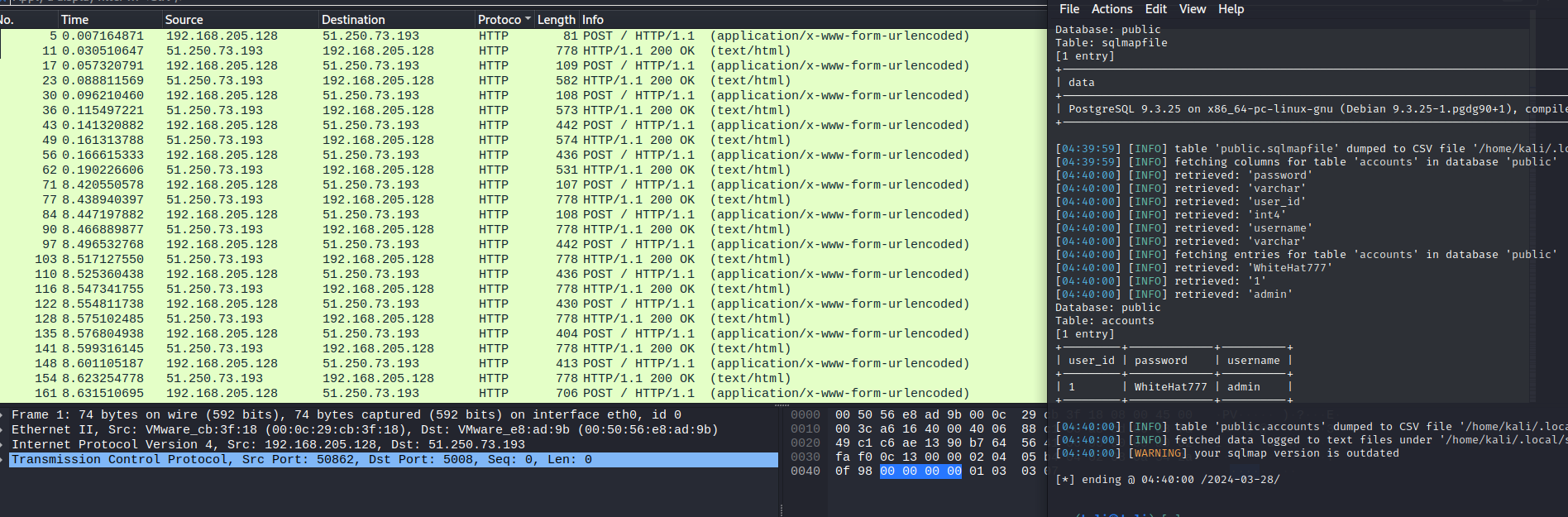
****

**Мини вывод:** 2 запроса 2 ответа

**Итог:**

Использование всего одного ключа малоэффективно. Я считаю, что нужно уметь их комбинировать, чтобы добиться каких-либо результатов.

Сразу можем заметить разницу в результате используя больше ключей  
**sqlmap -u "http://51.250.73.193:5008" --random-agent --randomize 5 --hpp --risk 3 --os-pwn**  


Ну и на последок я возьму готовый ключ решения этой задачи и посмотрю как это выглядит со стороны wireshark **sqlmap -u "http://51.250.73.193:5008" --data="username=test&password=test" --dbms=postgresql --technique=E –dump**

**Выводом для себя я приму факт того, что количество не всегда означает качество, поэтому нужно подбирать свои ключи под каждую ситуацию.**

1. **--random-agent:**
   * Положительные стороны: Позволяет случайным образом выбирать HTTP User-Agent для каждого запроса, что помогает обойти обнаружение ботов.
   * Ограничения: Может вызвать подозрение, если целевой сайт активно мониторит трафик.
2. **--randomize:**
   * Положительные стороны: Случайное перемешивание значений в запросах помогает обойти защиту от SQL-инъекций.
   * Ограничения: Требует указания количества случайных значений, что может быть неудобно при первоначальной настройке.
3. **--hpp:**
   * Положительные стороны: Обход защиты от атаки HTTP параметрами помогает предотвратить SQL инъекции.
   * Ограничения: Может быть неэффективен, если целевой сайт применяет другие методы защиты.
4. **--risk (от 1 до 3):**
   * Положительные стороны: Устанавливает уровень риска, что позволяет более агрессивно тестировать на уязвимости.
   * Ограничения: Более высокий уровень риска может привести к увеличению ложных срабатываний и ненужных запросов к серверу.
5. **--os-pwn:**
   * Положительные стороны: Пытается взломать целевую систему, используя уязвимости в базе данных для получения контроля над операционной системой.
   * Ограничения: Высокий риск использования, может привести к блокировке доступа к целевой системе и незаконным действиям.