**Отчет по пентесту-номер задания 16.x модуля 16**

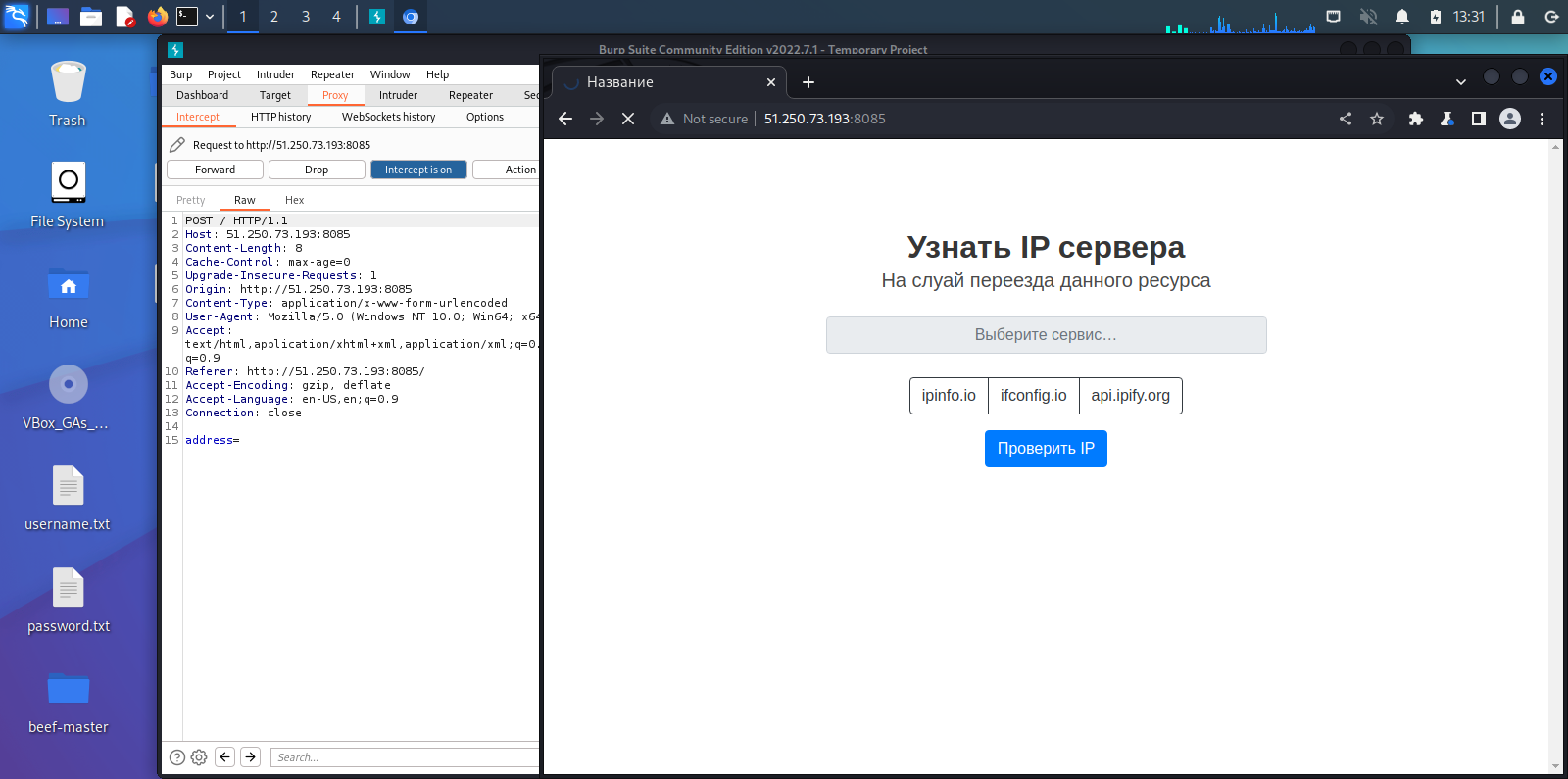
**Персонализация:** Ермакова Наталья Олеговна (natalysoliton@mail.ru)

# Задание 16.8.1

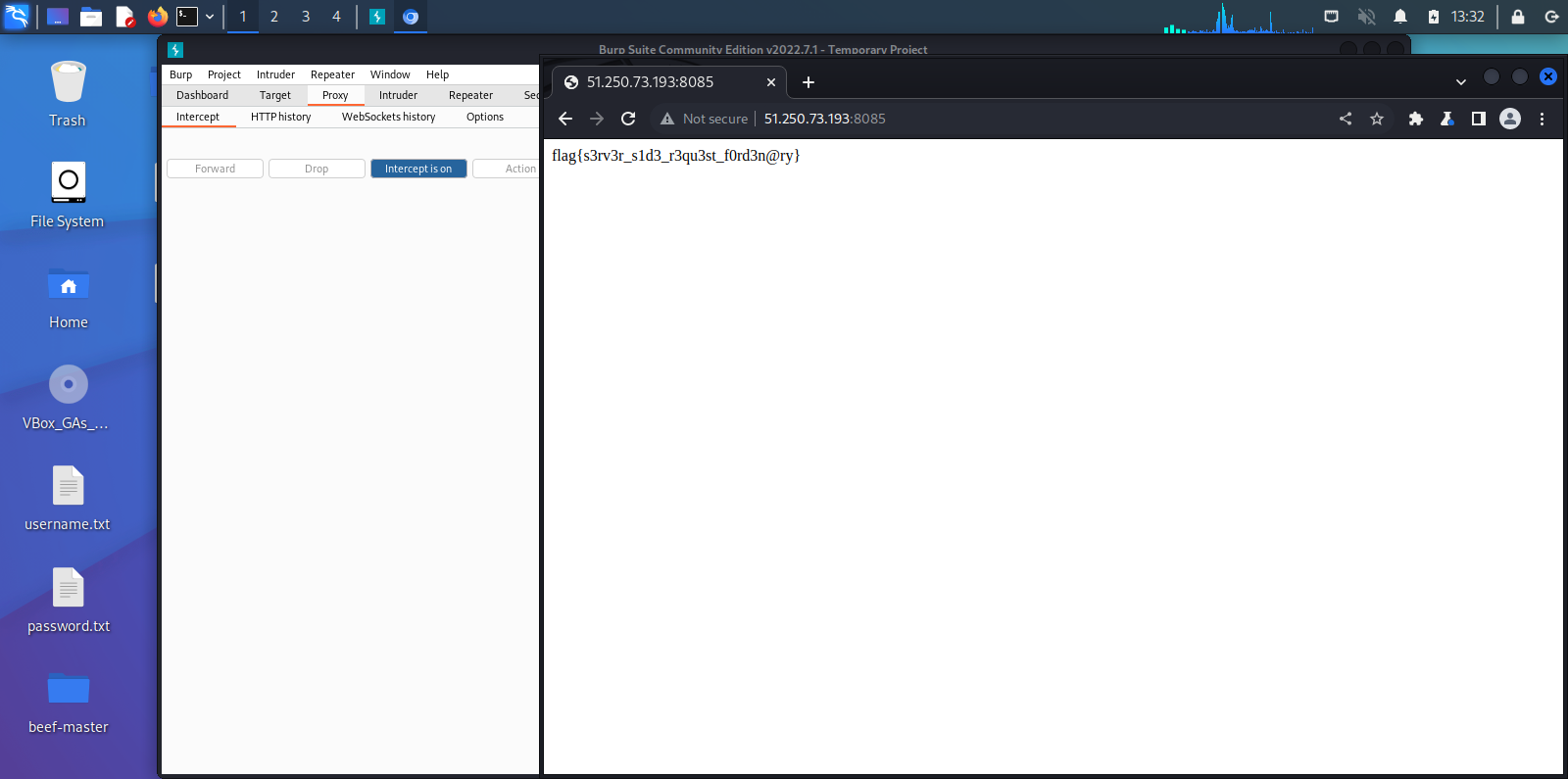
В нашей компании работает очень ленивый админ, и он очень любит упрощать себе жизнь, сначала мы не видели в этом ничего плохого, но потом различных сервисов стало слишком много, мы не хотим лишать его этих маленьких радостей, но надо подумать о безопасности. Проверьте, так ли безопасен его новый сервис: [http://51.250.73.193:8085](http://51.250.73.193:8085/). В отчёте прикрепите скриншоты выполнения работы.

* Общая информация:
  + Период тестирования: 16.12.2022
  + Объект тестирования: [http://51.250.73.193:8085](http://51.250.73.193:8085/)
* Описание действий:

Открыла веб-страницу и Burp Suite, включила интерсепт, на сайте нажала на чинюю кнопку.



Записала произвольный текст: address=text в Burp Suit, нажала Forward, видим флаг:



flag{s3rv3r\_s1d3\_r3qu3st\_f0rd3n@ry}

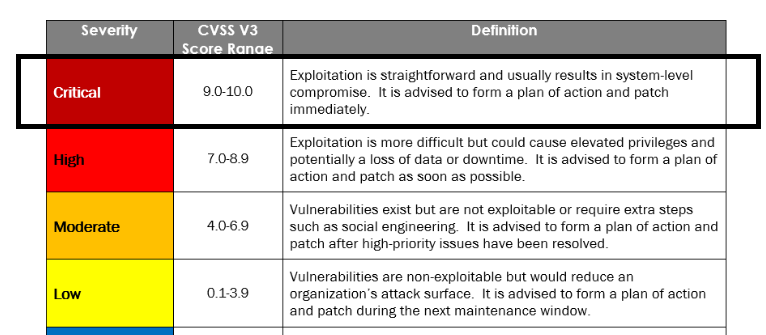
**Вопросы для самопроверки:**

* использовала Burp Suite
* совет по защите кода от уязвимости SSRF:

1. Ограничить доступ к внутренней инфраструктуре для потенциально подверженных SSRF серверов:
   1. ввести межсервисную аутентификацию;
   2. использовать внешний прокси.
2. Ограничить поддерживаемые схемы.
3. Отключить поддержку перенаправлений или проверять каждый шаг.
4. Валидировать доменные имена.
5. Разобраться, как используемая библиотека обрабатывает адреса.

Самый надёжный способ избежать подделки запросов на стороне сервера (SSRF) — это занести в белый список DNS-имя или IP-адрес, к которому ваше приложение должно получить доступ. Если подход с использованием белого списка не подходит и необходимо полагаться на черный список, важно правильно проверять вводимые пользователем данные. Например, не разрешать запросы к частным (немаршрутизируемым) IP-адресам (подробно описано в RFC 1918).

* Мне кажется уязвимость высокой опасности 9.0.10.0 по шкале:

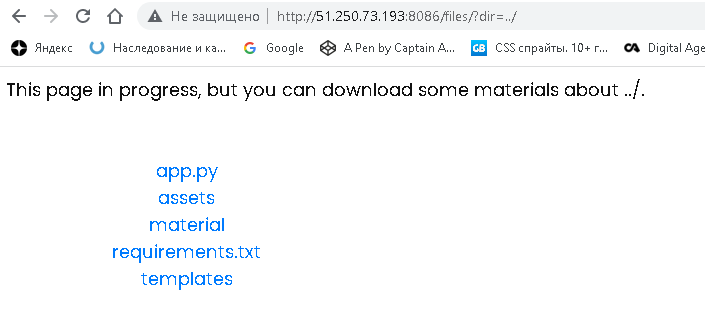


# Задание 16.8.2

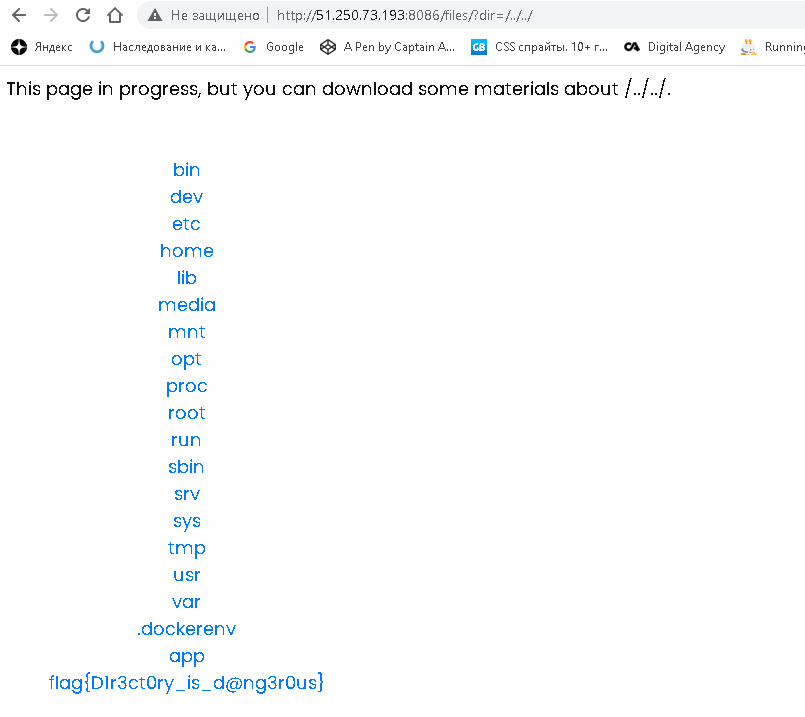
Мы основали свой стартап по предоставлению услуг веб-дизайна, а также обучения различным современным технологиям. Обучающее направление у нас пока на ранней стадии, так что посмотрите, пожалуйста, не допустили ли мы ошибок на странице предоставления учебных материалов: [http://51.250.73.193:8086](http://51.250.73.193:8086/). В отчёте прикрепите скриншоты выполнения работы.

* Общая информация:
  + Период тестирования: 16.12.2022
  + Объект тестирования: [http://51.250.73.193:8086](http://51.250.73.193:8086/)
* Описание действий:

Исследую скрытые директории:



В этих файлах нет флага. Иду глубже (http://51.250.73.193:8086/files/?dir=/../../):



Виден флаг: [flag{D1r3ct0ry\_is\_d@ng3r0us}](http://51.250.73.193:8086/files/?filename=flag%7BD1r3ct0ry_is_d%40ng3r0us%7D&dir=%2F..%2F..%2F)

**Вопросы для самопроверки:**

* использовала ручное исследование веб-страницы
* совет по защите кода от уязвимости Directory Traversal:

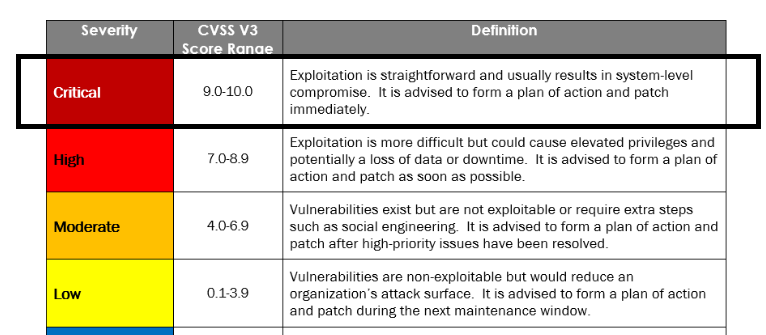
Эти атаки основаны на двух уязвимых элементах: коде веб-приложения и конфигурации веб-сервера.

Уязвимые веб-приложения используют непроверенные пользовательские входные данные в именах файлов и путях. Передача необработанных имён файлов и путей всегда является плохой идеей не только по соображениям безопасности (помимо path traversal, это может привести к Cross-Site Scripting), но и потому, что это делает приложения хрупкими и более трудными в обслуживании. Современные приложения обычно избегают этого, используя сопоставление URL-адресов для отделения URL-адресов от базовых файлов. Лучшее решение заключается в хранении файлов в центральной базе данных, а не непосредственно в файловой системе веб-сервера, и определении собственных имен ресурсов, используемых для доступа к ним.

Чтобы уменьшить уязвимость на стороне веб-сервера, убедитесь, что вы используете современное программное обеспечение веб-сервера. Процесс веб-сервера также должен работать с минимальными необходимыми привилегиями и иметь доступ только к каталогам, которые действительно нужны веб-сайту или приложению. Для систем Linux/UNIX вы можете рассмотреть возможность запуска веб-сервера в chroot jail, чтобы любые атаки Path Traversal не увенчались успехом.

Chroot jail – операция изменения корневого каталога диска для запущенного процесса и его дочерних процессов. Программа, запущенная в таком окружении не может получить доступ к файлам вне нового корневого каталога.

* Уязвимость высокой опасности 9.0.10.0 по шкале:



## 

## 

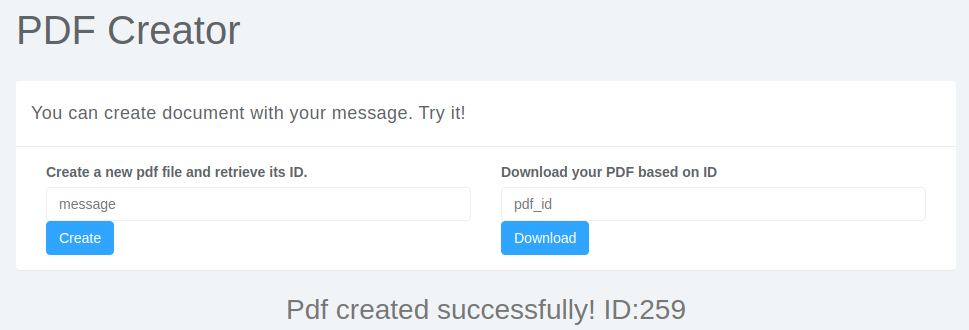
## 

# Задание 16.8.3

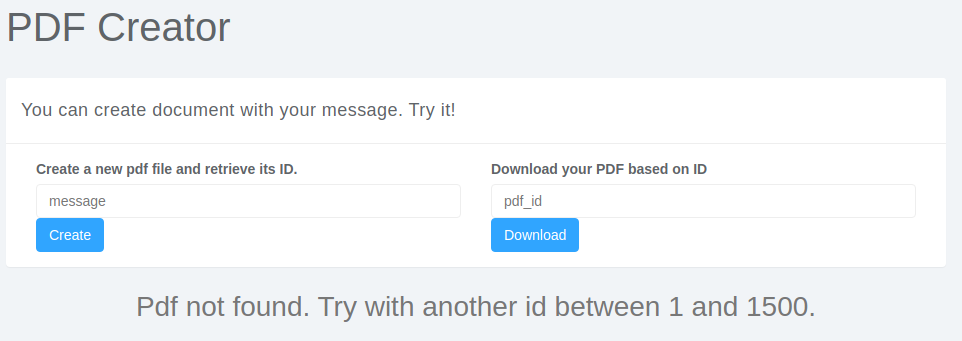
Открылся новый сервис по генерации *pdf*-документов, теперь каждый наш клиент может сгенерировать себе документ с любой надписью, которую только пожелает, мы думаем, людям это понравится. Надписи на футболках ведь востребованы: [http://51.250.73.193:8087](http://51.250.73.193:8087/). В отчёте прикрепите скриншоты выполнения работы.

* Общая информация:
  + Период тестирования: 16.12.2022
  + Объект тестирования: [http://51.250.73.193:8087](http://51.250.73.193:8087/)
* Описание действий:

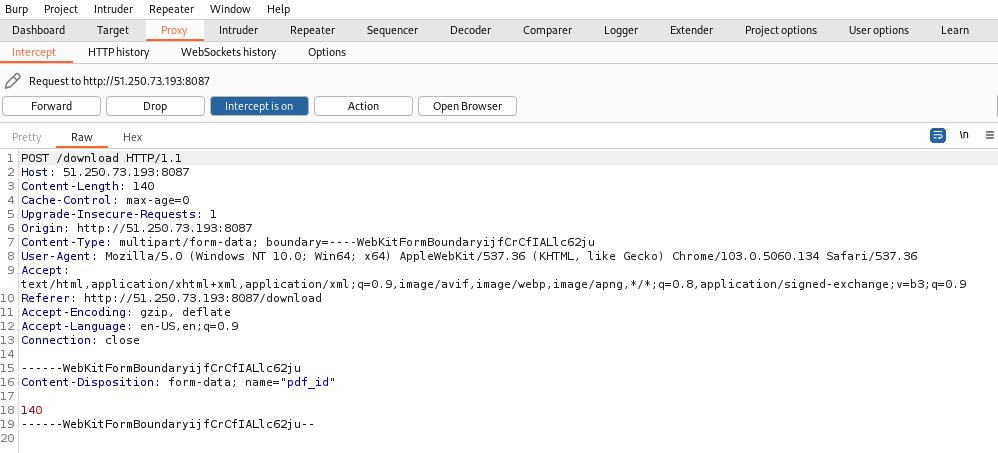
Перешла на сайт [http://51.250.73.193:8087](http://51.250.73.193:8087/), ввела произвольное название в поле создания пдф:



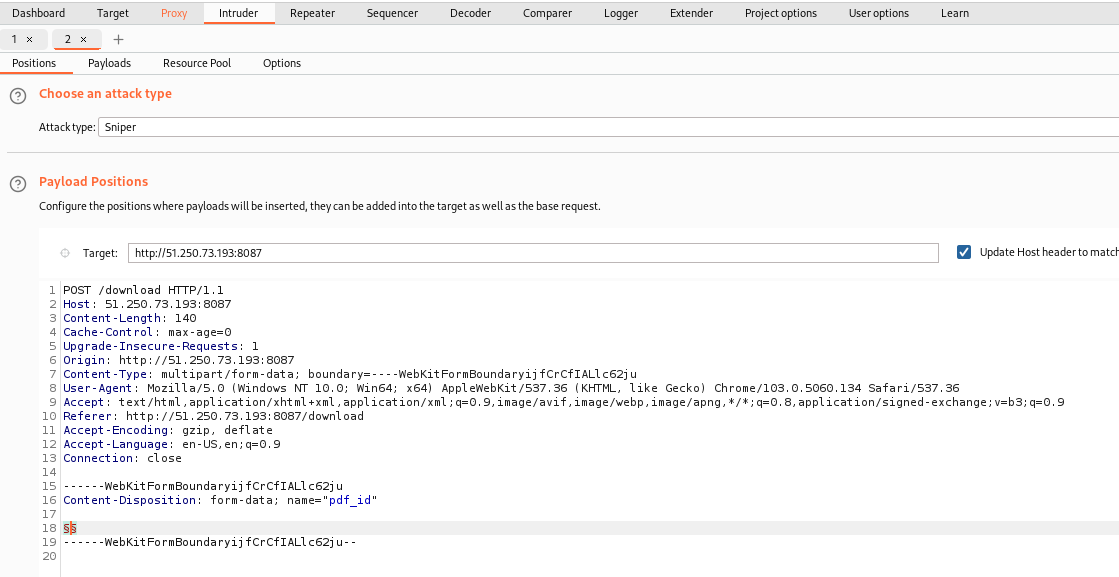
В поле pdf\_id ввела произвольное число - 120:



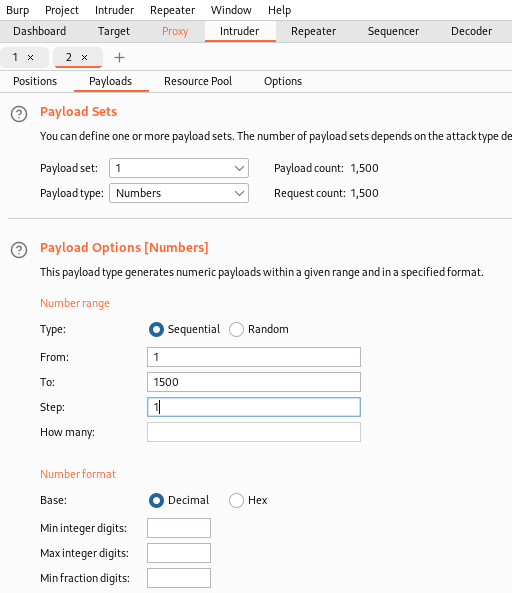
Посмотрела Intercept в Burp Suite при введении произвольного числа 140 в поле pdf\_id:



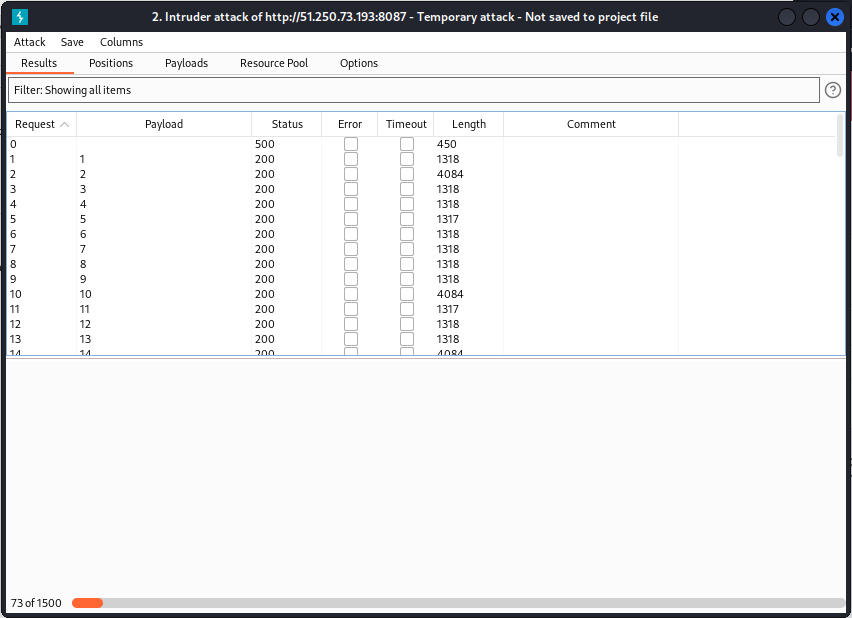
Отправила в Intruder, убрала число 140.



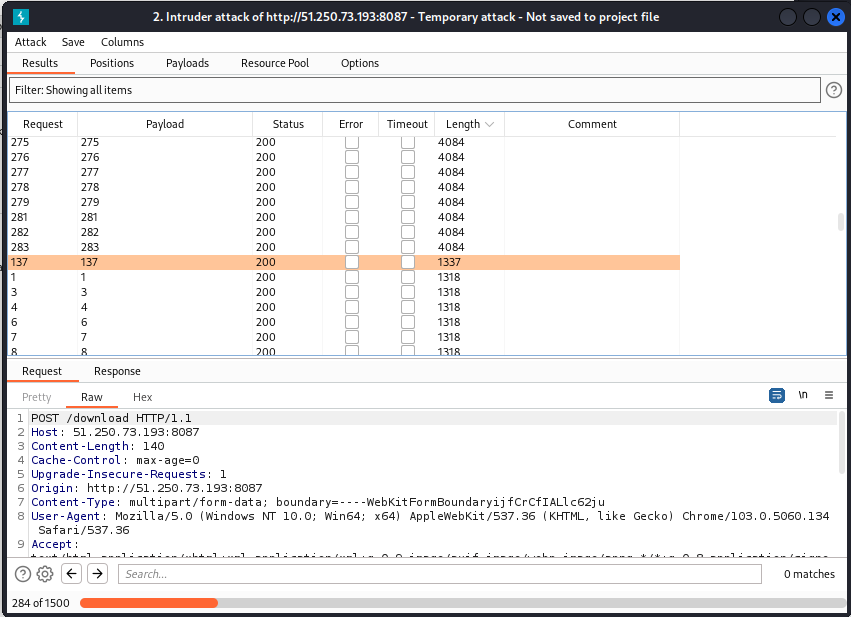
Сделаю автоперебор от 1- 1500:



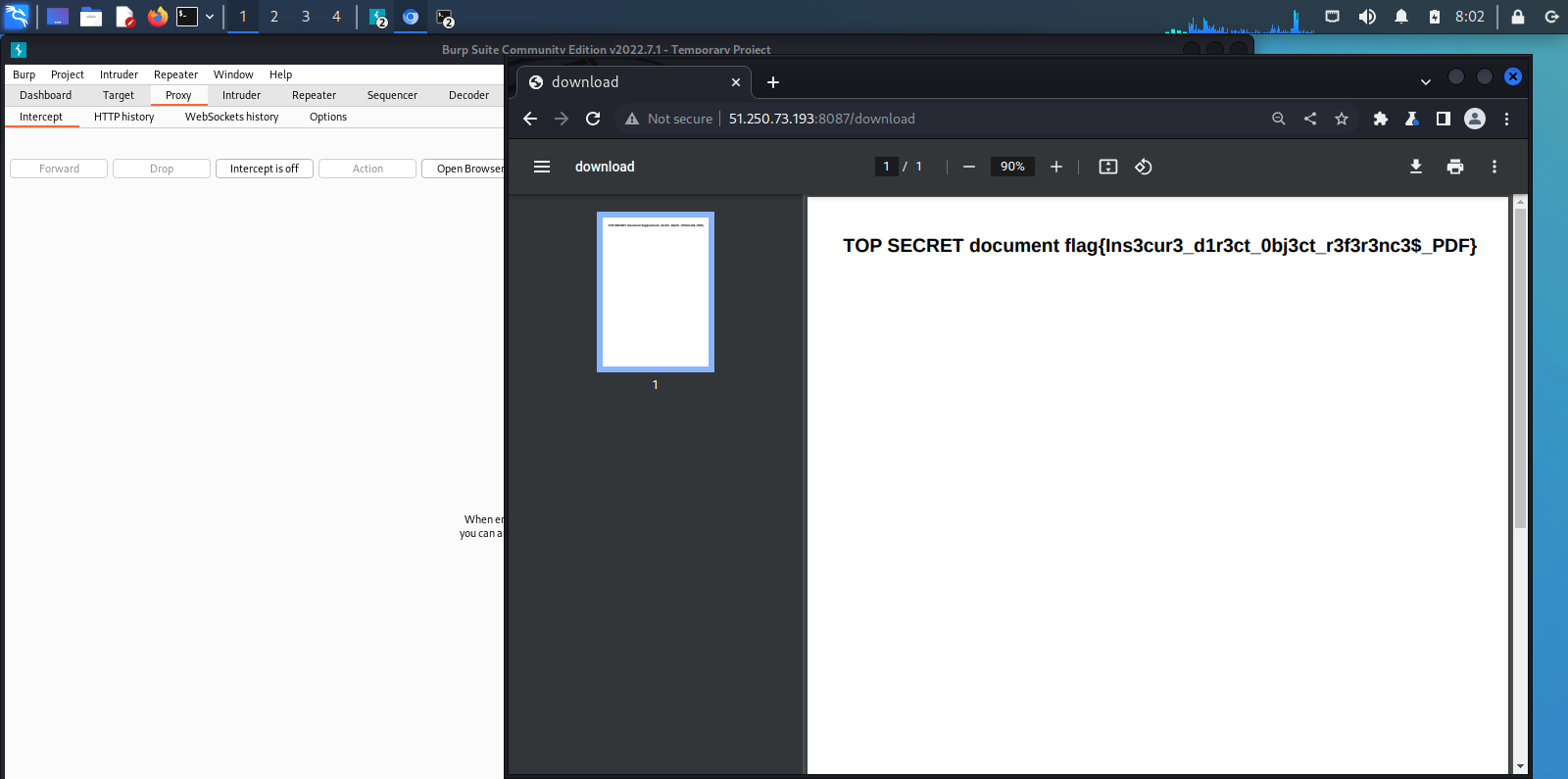
Стартуем атаку.



Так как перебор идет долго, попробую пока остановится от 1-280, отсортировать Length и посмотреть уникальное значение.



Пока вижу неповторяющееся число -1337, попробую скопировать ссылку для запроса (через контекстное меню Request in browser->In original session->copy link). Отключаю интерсепт, вставляю ссылку в браузер.



Получаю флаг: flag{Ins3cur3\_d1r3ct\_0bj3ct\_r3f3r3nc3$\_PDF}.

**Вопросы для самопроверки:**

* использовала Burp Suite
* совет по защите от IDOR уязвимости:

Первый подход заключается в замене их косвенными ссылками на объекты, которые затем внутренне сопоставляются с реальными объектами. Это может означать использование временной справочной карты для каждого сеанса, заполненной только значениями, действительными для конкретного пользователя и связанными со случайными, не последовательными ключами.

Использование защищённых хешей вместо реальных ссылок на объекты — это ещё один способ затруднить злоумышленникам манипулировать пользовательскими значениями.

Оба этих метода эффективны для сокрытия внутренних деталей реализации, но не решают основную проблему контроля доступа. Более надёжный подход к устранению уязвимостей IDOR заключается в обеспечении надлежащего управления сеансами и проверками контроля доступа пользователей на уровне объектов. Таким образом, даже если злоумышленник сумеет обнаружить внутреннюю ссылку на объект и попробует использовать её, то он не получит несанкционированного доступа.

* уязвимость высокой опасности 9.0.10.0 по шкале:

