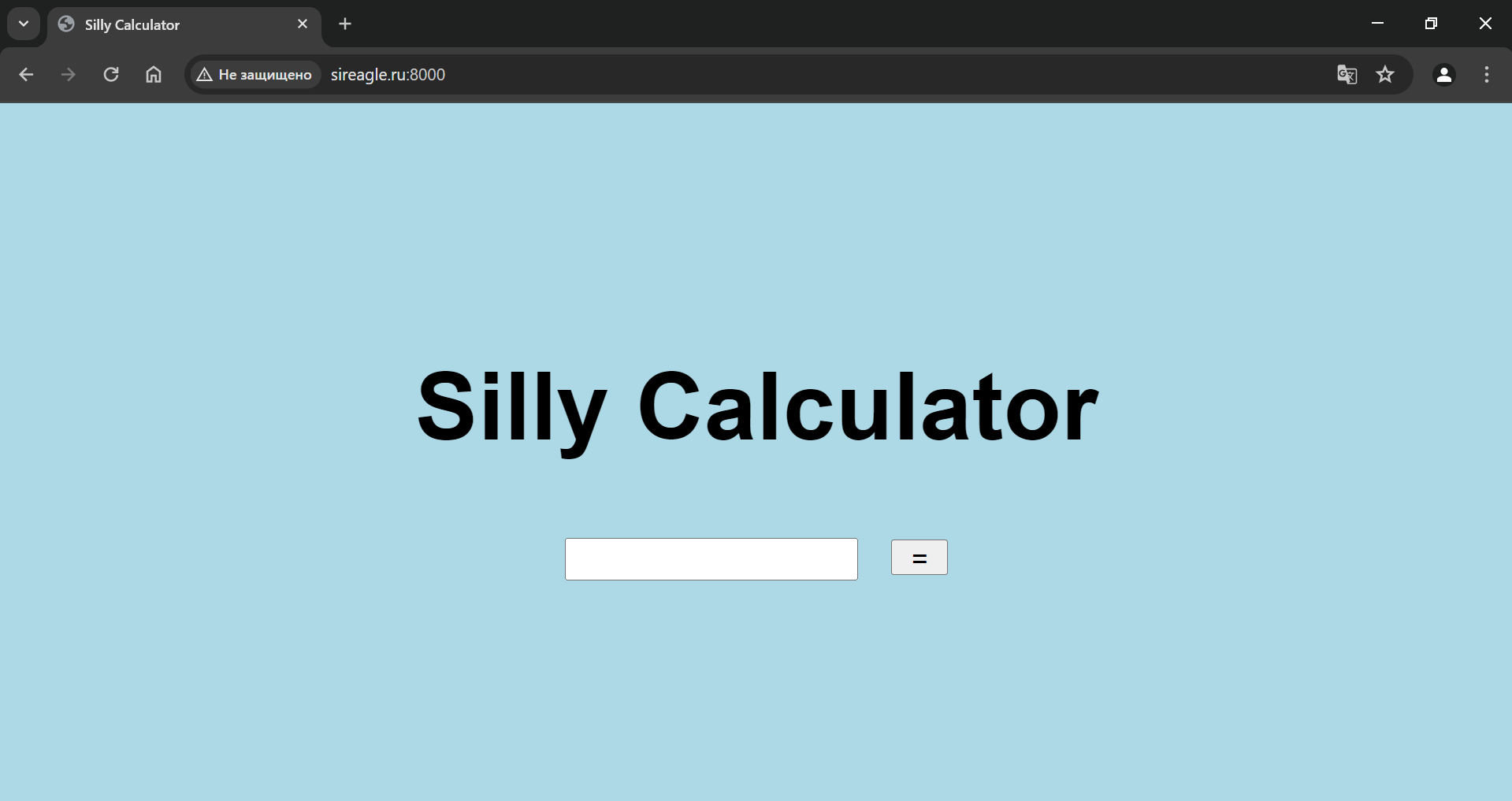
Запуск таска: **docker build -t task . ; docker run -d --name task -p \*внешний\_порт\*:80 task**

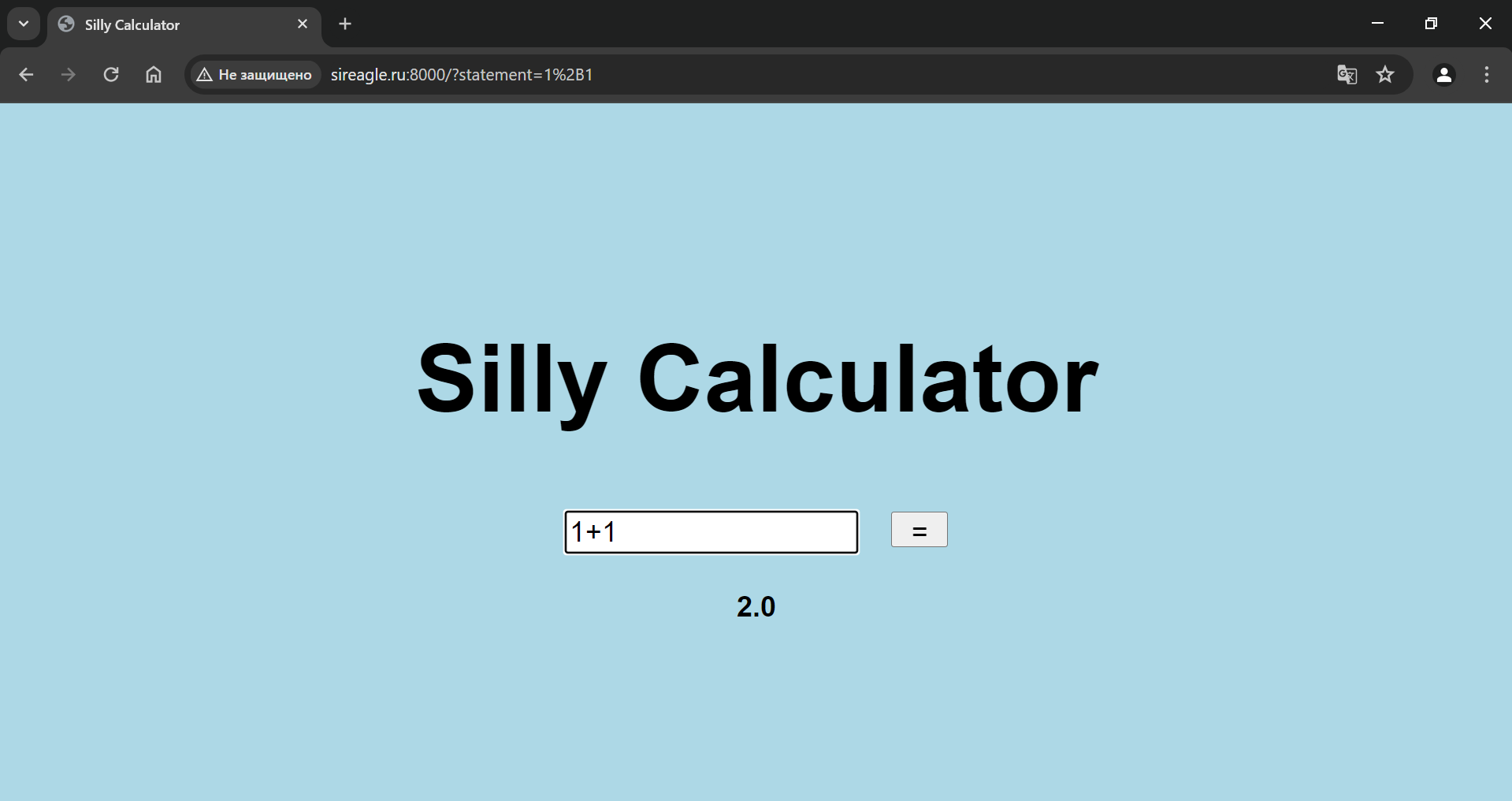
Флаг задаётся в файле Dockerfile, по необходимости можно заменить на свой.

Решение:

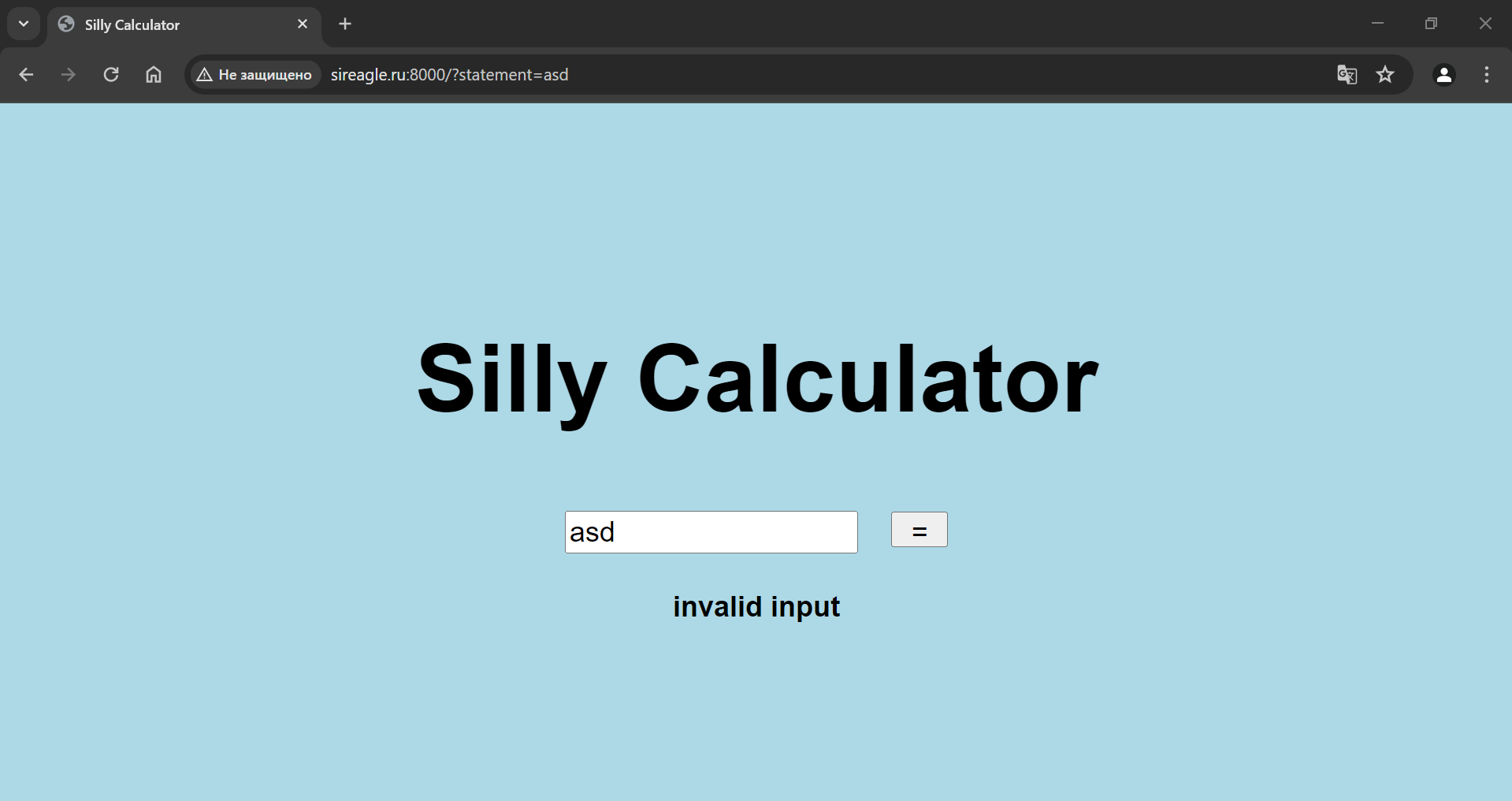
Встречает нас незамысловатый сайт.



Если вводим какое-нибудь математическое выражение, то на выходе получаем ответ.

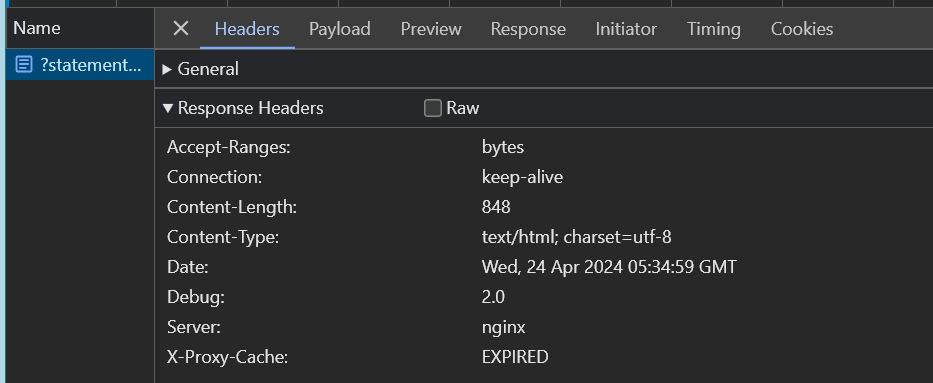


В ином случае – получаем ошибку.

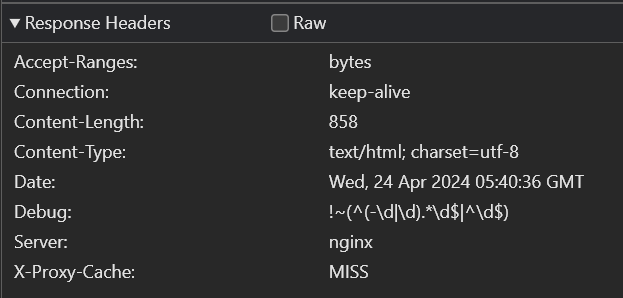


Непосредственно от вебки мы больше ничего интересного не получим.

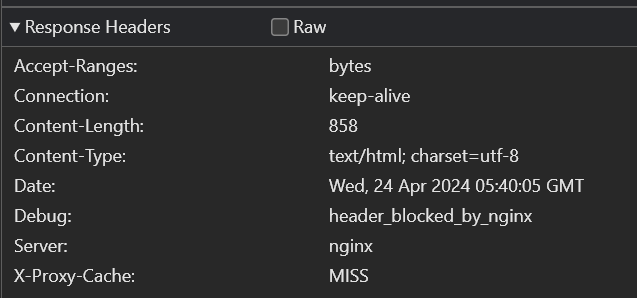
Если заглянем в заголовки ответов, то увидим интересный заголовок Debug, в который пробрасывается либо ответ, если ответ корректен:



Либо (глупая) регулярка, намекающая на валидацию запроса:

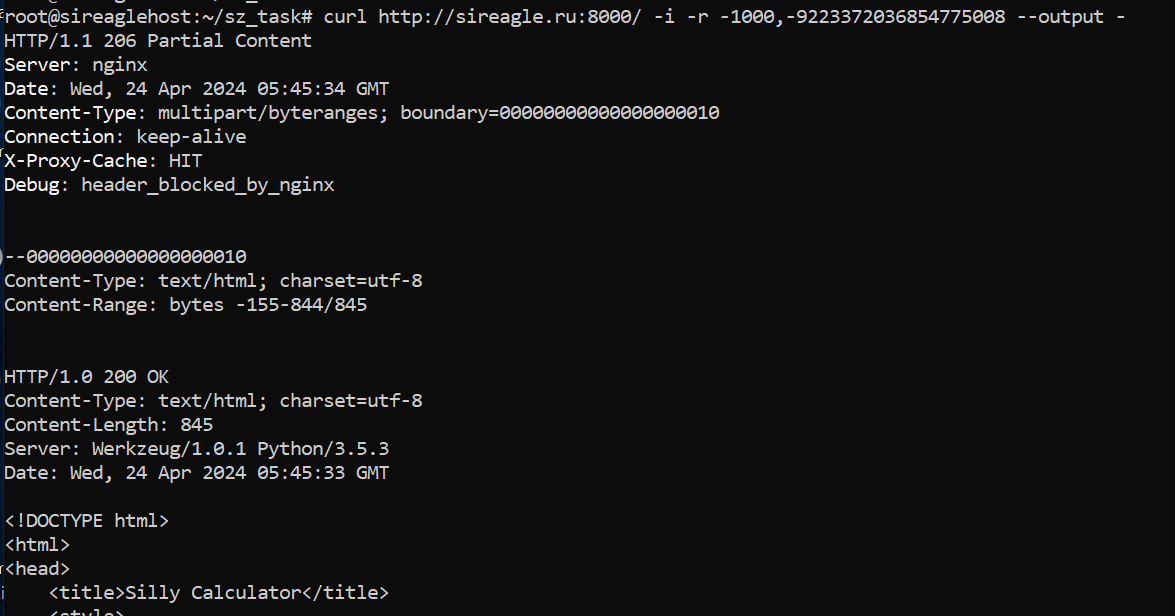


Либо ошибку «block by nginx», если запрос соответствует регулярке, но не является корректным математическим выражением:

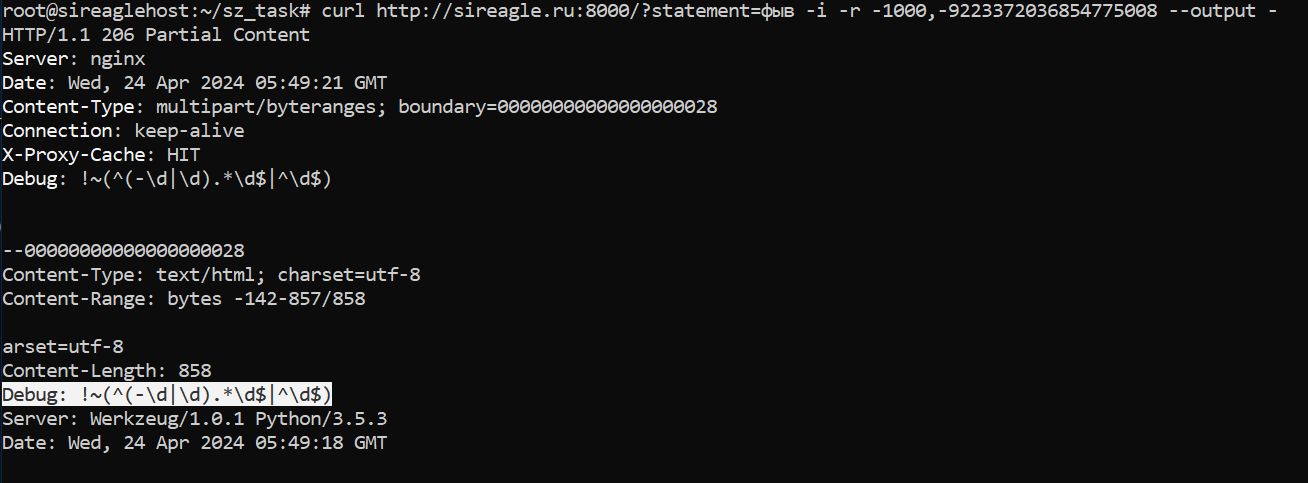


Из тех же заголовков получаем дополнительную информацию о приложении: перед ним стоит nginx неизвестной версии и на nginx настроено кэширование. Не так сложно будет по этим данным найти CVE-2017-7529, которая позволяет при помощи integer overflow получить оригинальный ответ сервера.

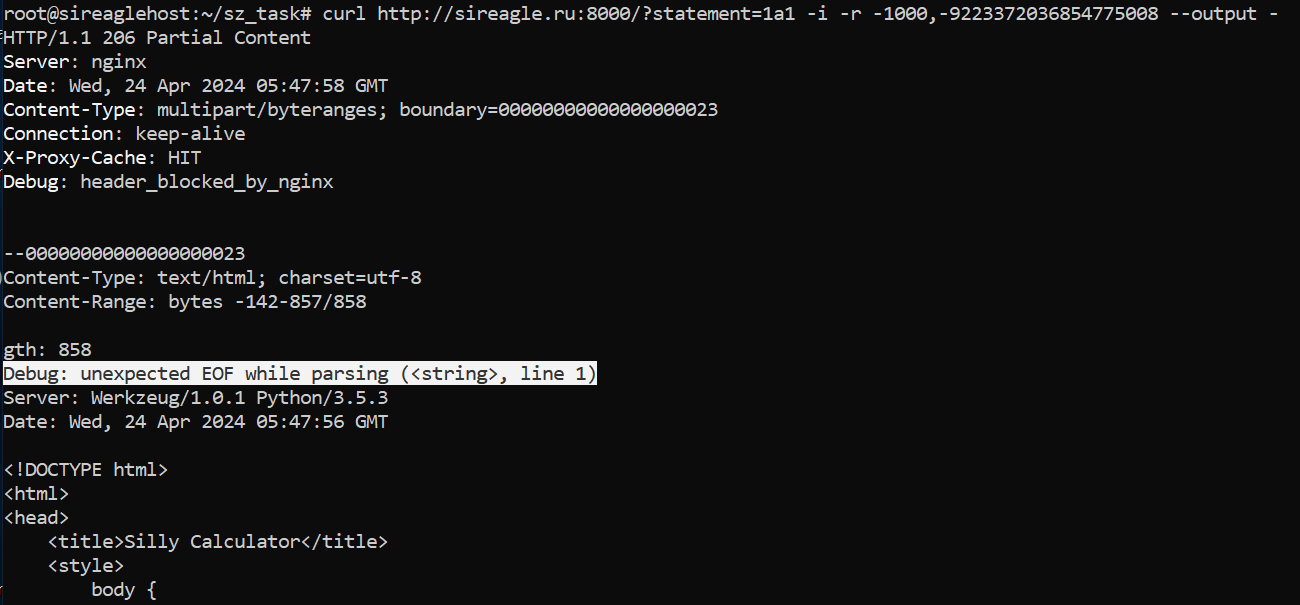
Отправив, например, обычный курл (чтобы ответ попал в кэш, по конфигурации он чистится каждые 5 секунд), а следом курл с указанием заголовку Range с особым диапазоном (его надо будет частично подбирать), получаем информацию о бэкенде.



Видим, что имеем дело с Python, заголовок Debug пока не светится. Отправим какой-нибудь GET-параметр. Если он не соответствует руглярке, мы её также увидим:

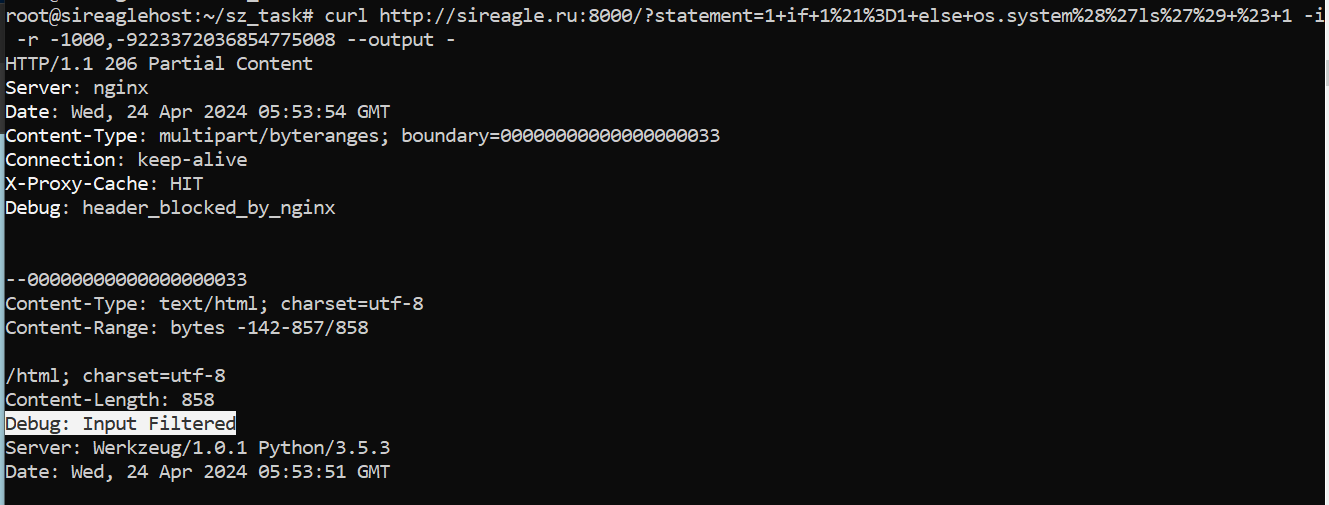


Если запрос соответствует регулярке, но не является корректным математическим выражением, то получим следующую ошибку:

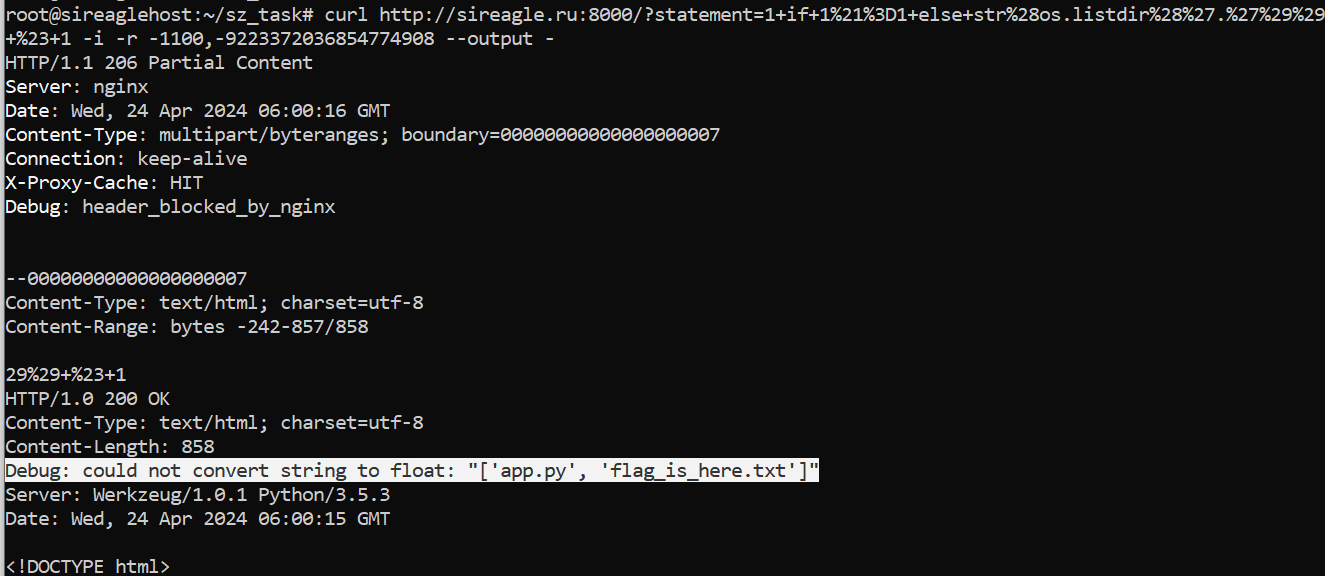


На этом этапе уже почти очевидно, что используется функция eval() для вычисления выражение. Данная функция уязвима к инъекции, поэтому начинаем тестить разный пейлоады. Учитывая, что нам всё ещё необходимо удовлетворять регулярке, необходимо придумать какой-то хитрый запрос, который начинается и заканчивается на цифру и при этом который обработает eval(). Например, что-то такое: «1 if 1!=1 else \*инъекция\* # 1».

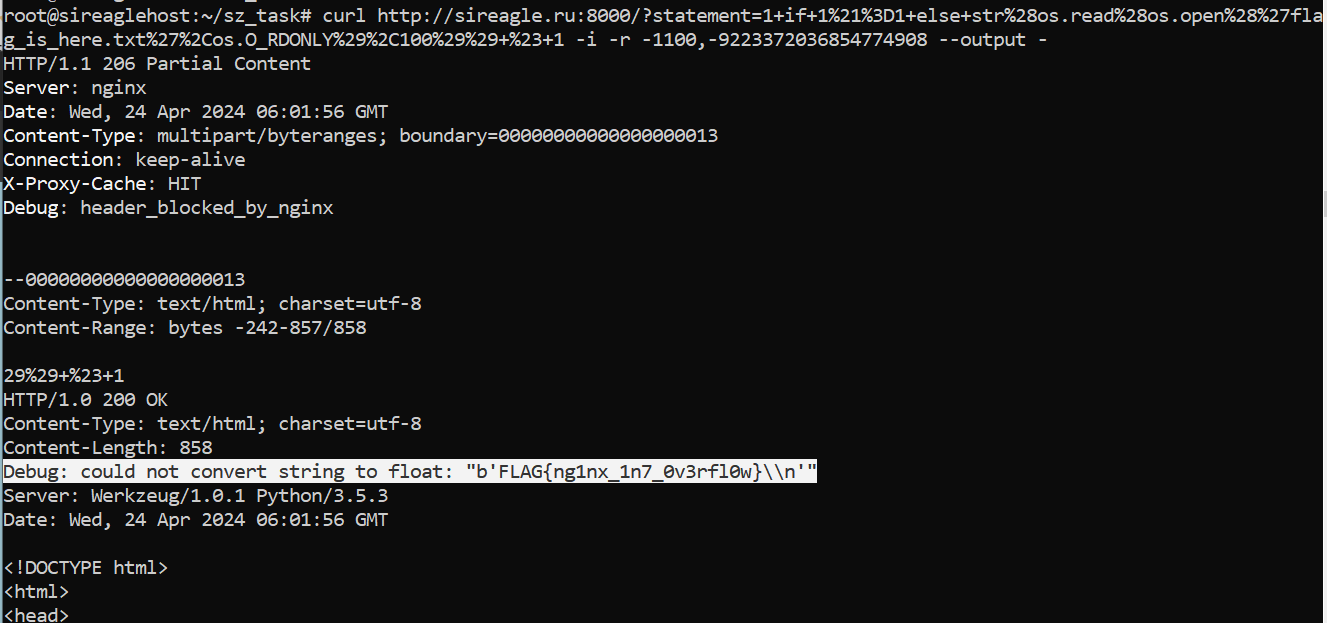
Ввод у нас дополнительно фильтруется, например, недоступны «system», «sh», «rm», «\_\_», «exec», но для получения флага они и не потребуются.



Для получения файлов в текущей директории используем: «1 if 1!=1 else str(os.listdir('.')) # 1»



Для чтения файла: «1 if 1!=1 else str(os.read(os.open('flag\_is\_here.txt',os.O\_RDONLY),100)) # 1»



Получаем флаг «FLAG{nginx\_1n7\_0v3rfl0w}»