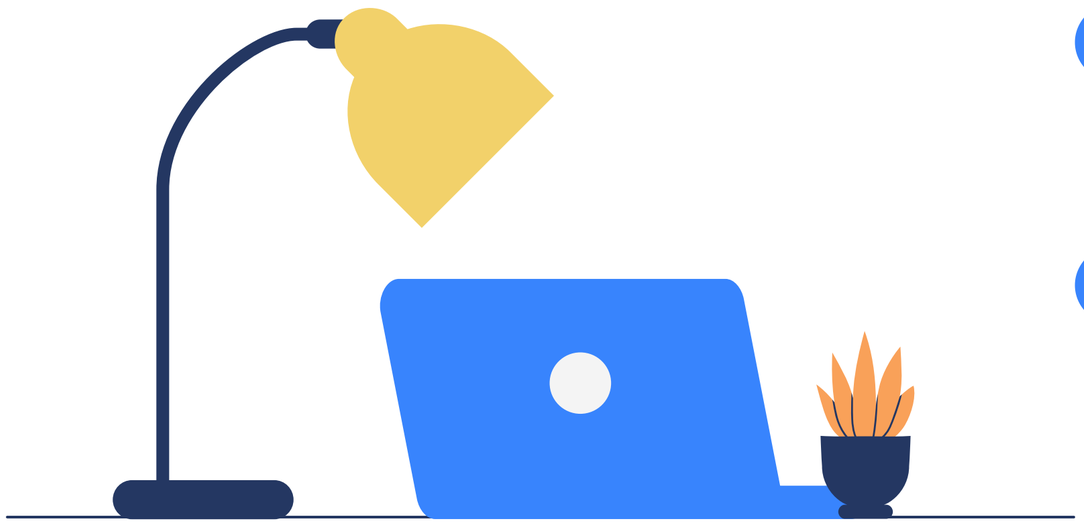
WEB SERVER

проект школы 21

ОСНОВЫ

СЕРВЕР И КЛИЕНТ

ЧТО ЭТО?
И КАК ЭТО РАБОТАЕТ?



- В интернете информация часто хранится на разных машинах(серверах в физическом смысле).
- Для обмена информации между машинами используют программы-клиенты и программы-серверы.
- Клиенты отправляют запросы, чтобы им отправили нужную информацию
- Серверы принимают запросы, обрабатывают их и отправляют в ответ нужную ифнормацию.
- Чтобы сервер и клиент поняли друг друга им нужно руководствоваться одинаковыми правилами. Их еще называют протоколами.

01

ОТПРАВКА ЗАПРОСА

Программа клиент отправляет запрос.

02

ОБРАБОТКА ЗАПРОСА

Программа-сервер получает запрос. Понимает из него, что от нее хотят. Совершает то, что нужно. И готовит ответ о том, как все прошло

03

ОТПРАВКА ОТВЕТА

Программа-сервер отправляет ответ.

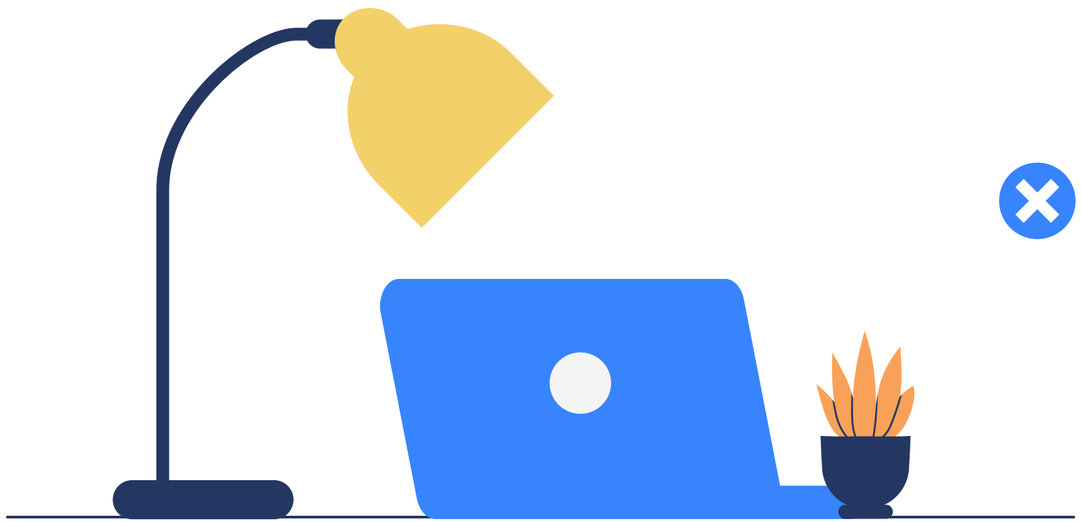
04

ОБРАБОТКА ОТВЕТА

Программа-клиент получает ответ. Обрабатывает его. И если надо отправляет новый запрос.

HTTP

НЕМНОГО ПРО ПРОТОКОЛ И ЕЩЕ НЕМНОГО ПРО HTTP-СЕРВЕР



- HTTP - это один из таких протоколов
- Вообще HTTP - это гипертекстовый протокол и слово текст тут стоит понимать буквально. В основе этого протокола лежит обмен абсолютно человекочитаемыми сообщениями.
- Представь, что ты зашел в браузер. Ты вбиваешь какой-то адрес и у тебя открывается какая-то страница.
  Как она попала к тебе на компьютер?
- Она попала к тебе с другой машины, на которой хранится эта страничка и работает HTTP-server (скорее всего NGINX или apache), ответственный за то, чтобы отправить ее тебе.

КАК ВЫГЛЯДЯТ ЗАПРОС И ОТВЕТ

по-разному, но как-то так

	ЗАПРОС		ОТВЕТ
-ШАПКА-	POST / HTTP/1.1/r/n	-ШАПКА-	HTTP/1.1 403 Forbidden/r/n
-ЗАГОЛОВКИ-	Host: first_server/r/n	-ЗАГОЛОВКИ-	Server: Apache/r/n
	Content-length: 12/r/n		Conten-length: 170
	/r/n		/r/n
-ТЕЛО-	1234567890/r/n	-ТЕЛО-	<html>/r/n
			<head><title>403 Forbidden</title></head>/r/n
			<body bgcolor="white">/r/n
			403 Forbidden /r/n
			<hr> nginx/0.8.54 /r/n
			</body>/r/n
			</html>/r/n

ТИПЫ ЗАПРОСОВ

GET

Когда мы хотим получить какую-то информацию с сервера.

PUT

Когда мы хотим изменить какую-то информацию на сервере.

POST

Когда мы хотим загрузить какую-то информацию на сервер.

DELETE

Когда мы хотим удалить какую-то информацию с сервера.

СТАТУС-КОД

Это код в ответе. Он говорит нам о том, как прошла обработка запроса

100-199	200-299	300-399	400-499	500-599
Информационные	Успешные	Перенаправление	Клиентские ошибки	Серверные ошибки

CGI И COOKIES

CGI

- на сервере не всегда изначально есть нужная информация
- порой ее нужно сформировать, используя данные из запроса
- это делается с помощью специальной программы, которая находится на сервере
- эта программа запускается при принятии запроса с определенным url
- фактически это работает через execve, причем данные передаются через переменные среды
- cookies - это заголовок, который позволяет серверу понять с какой сессией он работает

ТЕХНИЧЕСКАЯ СТОРОНА

КАК УСТАНАВЛИВАЕТСЯ СОЕДИНЕНИЕ?

ЧЕРЕЗ СОКЕТЫ.

ЧТО ТАКОЕ СОКЕТЫ?

ЕСЛИ КОРОТКО - ЭТО ЧТО-ТО ВРОДЕ ПАЙПОВ, НО НЕ ДЛЯ ПРОЦЕССОВ, А ДЛЯ ПРОГРАММ. В ТОМ ЧИСЛЕ И НА РАЗНЫХ МАШИНАХ.

КАК МЫ ПОЛУЧАЕМ СОКЕТЫ?

ЗАВИСИТ ОТ ТОГО КЛИЕНТ ЭТО ИЛИ СЕРВЕР.

ЧТО ПРОИСХОДИТ В СЛУЧАЙ С КЛИЕНТОМ?

МЫ СОЗДАЕМ СОКЕТ С ПОМОЩЬЮ SOCKET И ПОТОМ ПОДКЛЮЧАЕМСЯ ПО НУЖНОМУ НАМ АДРЕСУ С ПОМОЩЬЮ ФУНКЦИИ CONNECT.

А С СЕРВЕРОМ?

ТУТ СЛОЖНЕЕ. СНАЧАЛА МЫ СОЗДАЕМ СОКЕТ. ПОТОМ ПРИВЯЗЫВАЕМ ЕГО К ОПРЕДЕЛЕННОМУ АДРЕСУ ЧЕРЕЗ BIND. ПОТОМ ПЕРЕВОДИМ ЕГО В СЛУШАЮЩИЙ РЕЖИМ. ПРИ ПОПЫТКАХ ПОДКЛЮЧИТЬСЯ ПО ДАННОМУ АДРЕСУ СЛУШАЮЩИЙ СОКЕТ ПРИ ЧТЕНИИ ИЗ НЕГО БУДЕТ ВОЗВРАЩАТЬ НАМ FD СОКЕТА, ОТВЕЧАЮЩЕГО ЗА ВНОВЬ ОБРАЗОВАННОЕ СОЕДИНЕНИЕ. ФУХ...

ТО ЕСТЬ У НАС МОЖЕТ БЫТЬ МНОЖЕСТВО ОДНОВРЕМЕННЫХ СОЕДИНЕНИЙ МЕЖДУ ДВУМЯ ПРОГРАММАМИ?

ДА. НО ПРИДЕТСЯ ПОЗНАКОМИТЬСЯ С МУЛЬТИПЛЕКСИРОВАНИЕМ.

А МОЖЕМ ЛИ МЫ НА ОДНОМ СЕРВЕРЕ ОДНОВРЕМЕННО ПРИНЯТЬ НЕСКОЛЬКО СОЕДИНЕНИЙ ОТ РАЗНЫХ КЛИЕНТОВ?

ДА. НО ПРИДЕТСЯ ПОЗНАКОМИТЬСЯ С МУЛЬТИПЛЕКСИРОВАНИЕМ.

ПРЕДСТАВЬ ЧТО НЕСКОЛЬКО ПОЛЬЗОВАТЕЛЕЙ ПЫТАЮТСЯ СКАЧАТЬ С ТВОЕГО САЙТА ПРОСТУЮ HTML СТРАНИЦУ. А ОДИН - ТЯЖЕЛЫЙ ФАЙЛ. КАКАЯ ВОЗНИКНЕТ ПРОБЛЕМА?

МОЖНО ЛИ РЕШИТЬ ЭТУ ПРОБЛЕМУ С ПОМОЩЬЮ МНОГОПОТОЧНОСТИ?

ТОЛЬКО ПРИ МАЛЕНЬКОМ КОЛИЧЕСТВЕ СОЕДИНЕНИЙ.

А ВОТ СЕЙЧАС Я РАССКАЖУ ПРО НЕБЛОКИРУЮЩИЙ I/O И SELECT!

ЧЕК-ЛИСТ

01

Select

Находится в бесконечном цикле в мэйн

02

read/write

Все чтения и записи являются неблокирующими

03

Select и read/write

Все файловые дескрипторы на чтение и запись проверяются в select одновременно

04

errors

При чтении и записи проверяются возвращаемый результат, если меньше или равен 0, то сокет закрывается

ПРОВЕРИМ РАЗНЫЕ ТИПЫ ЗАПРОСОВ

curl -i -X POST -H 'Content-Type: multypart/form-data' -F 'filename=@path' ip+URL

curl -i -X DELETE ip+URL

GET

Возвращает нужный файл

POST

Загружает нужный файл

PUT

У нас его нет. Сервер просто должен не крашиться.

DELETE

Удаляет нужный файл

ПРОВЕРИМ ОДНОВРЕМЕННУЮ РАБОТУ РАЗНЫХ ПОРТОВ

8080

ПОКАЗЫВАЕТ ДЕФОЛТНЫЙ ФАЙЛ ДЛЯ ДАННОГО ПОРТА

9090

ТОЖЕ ПОКАЗЫВАЕТ ДЕФОЛТНЫЙ ФАЙЛ НО УЖЕ ДЛЯ ДАННОГО ПОРТА

01

Файл Ошибки

Можно задать дефолтный файл, который будет возвращаться при определенной ошибке

02

Автоиндексация

В случай отсутствия индексирующего файла при запросе на директорию будет возвращаться листинг фалов директории

03

Перенаправление

При совпадении url с url блока location будут применяться конфигурация данного блока

04

Размер тела и разрешенные методы

Можно ограничить тело запроса и разрешить использовать только определнные методы для определнных url.

А ТЕПЕРЬ ОДИН ПОРТ. НО РАЗНЫЙ СЕРВЕР-НЭЙМ.

curl --header 'Host: first_server' http://localhost:8080

curl --header 'Host: second_server' http://localhost:8080

FIRST_SERVER

ПОКАЗЫВАЕТ ДЕФОЛТНЫЙ ФАЙЛ ДЛЯ ДАННОГО СЕРВЕР-НЭЙМА

SECOND_SERVER

ТОЖЕ ПОКАЗЫВАЕТ ДЕФОЛТНЫЙ ФАЙЛ НО УЖЕ ДЛЯ ДАННОГО СЕРВЕР-НЭЙМА

ПРОВЕРКА СКОРОСТИ И УТЕЧЕК

- siege -b ip:port
- Activity Monitor
- lsof -i

CGI И КУКИЗ

Тут все покажет Вова