Сокеты

Основы Веб-программирования

Кафедра Интеллектуальных Информационных Технологий, ИнФО, УрФУ

Ресурсы

https://ru.wikipedia.org/wiki/Сокеты_Беркли

http://lecturesnet.readthedocs.org/net/low-level/index.html

Сокет

Сокет — абстрактный объект, представляющий конечную точку соединения.

В ОС объект сокета представлен файловым дескриптором и может принимать те же операции что и файл (открытие - чтение/запись - закрытие).

Дополнительно сокеты имеют операции для связи их с адресом и подготовки к обмену данными.

Типы

- UNIX - для взаимодействия процессов на локальной машине.
- INET - для взаимодействия по сети.
- WebSocket - протокол напоминающий механизм сокетов, предназначен для связи браузера с сервером (в обе стороны).

INET типы

- Stream socket - TCP.
- Datagram socket - UDP.
- Raw socket - формируем пакеты вручную.

Системный вызов socket

Пример INET сокета типа TCP:

Код 1: Си

```c
#include <sys/types.h>
#include <sys/socket.h>
int sock_fd = socket(AF_INET, SOCK_STREAM, 0);
```

Код 2: Python

```python
import socket
sock_obj = socket.socket(socket.AF_INET,
                         socket.SOCK_STREAM, 0)
```

Основные функции

socket	Создать новый сокет и вернуть файловый дескриптор
send	Отправить данные по сети
receive	Получить данные из сети
close	Закрыть соединение
bind	Связать сокет с IP-адресом и портом
listen	Объявить о желании принимать соединения. Слушает порт и ждет когда будет установлено соединение
accept	Принять запрос на установку соединения
connect	Установить соединение

Клиент. Установка соединения

Со стороны клиента используется функция **connect**, которая инициирует установление связи на сокете:

Код 3: Си

```c
#include <sys/types.h>
#include <sys/socket.h>
int connect(int sockfd,
            const struct sockaddr *serv_addr,
            socklen_t addrlen);
```

Код 4: Python

```python
server_address = ('192.168.1.100', 8080)
sock_obj.connect(server_address)
```

Клиент. Установка соединения

Со стороны клиента используется функция **connect**:

Код 5: Си

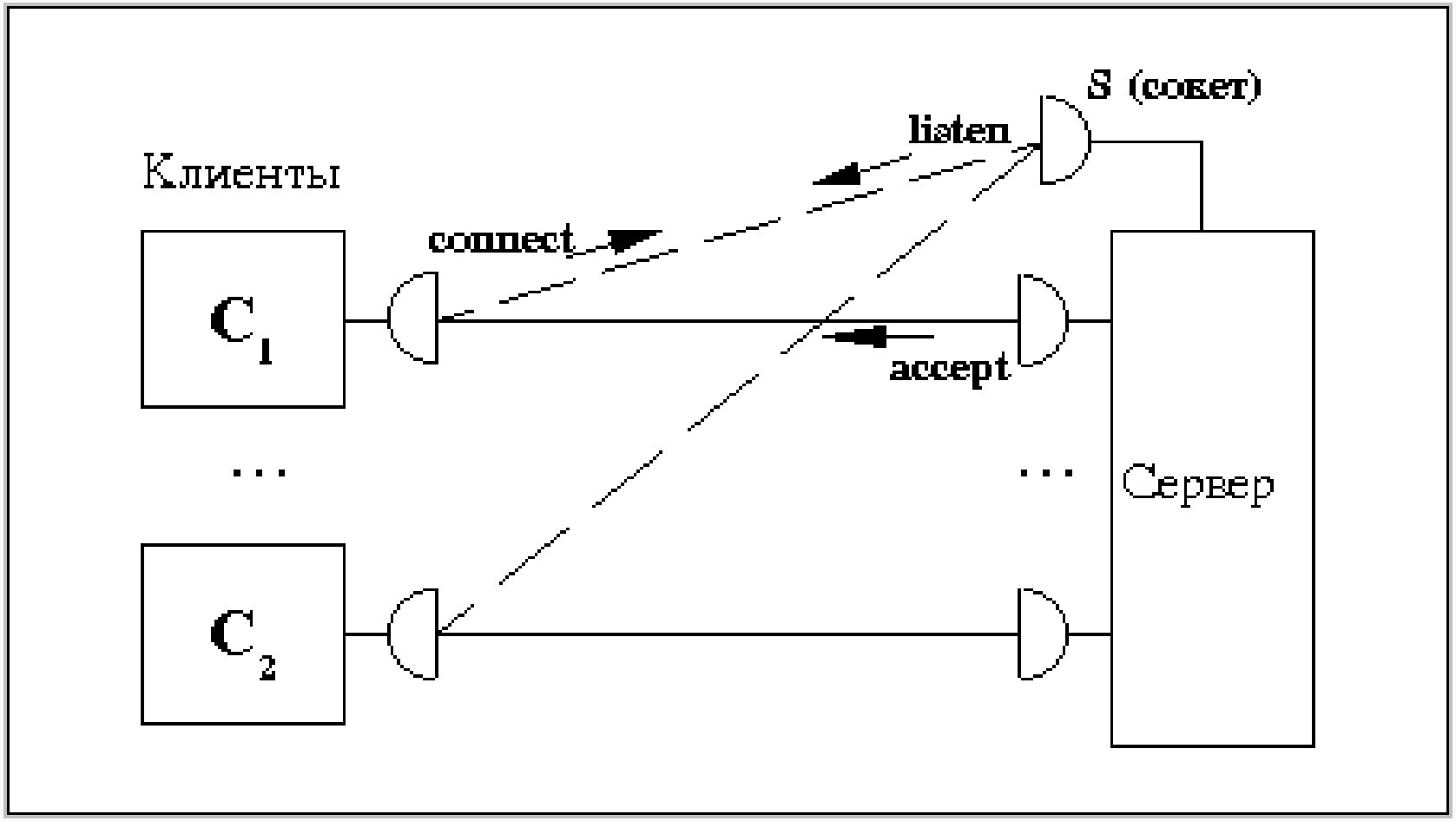
```c
#include <sys/types.h>
#include <sys/socket.h>
int connect(int sockfd,
            const struct sockaddr *serv_addr,
            socklen_t addrlen);
```

Код 6: Python

```python
server_address = ('192.168.1.100', 8080)
sock_obj.connect(server_address)
```

Сервер. Установка соединения

Сервер слушает порт и при запросе от клиента устанавливает соединение.



Сервер. Ждем соединения.

Сервер слушает порт и ожидает запрос на соединение от клиента:

Код 7: Си

```c
#include <sys/socket.h>
int listen(int sockfd, int backlog);
```

Код 8: Python

```python
sock_obj.listen(5)  # fd = 5
```

Сервер. Подтверждаем соединение.

accept - подтверждает запрос клиента и устанавливает соединение.

Код 9: Си

```c
#include <sys/types.h>
#include <sys/socket.h>
int accept(int sockfd, struct sockaddr *cliaddr,
           socklen_t *addrlen);
```

Код 10: Python

```python
conn, addr = sock_obj.accept()
```

Передача данных

В Python вызов методов **send**, **recv** без дополнительных параметров, аналогичен системным вызовам **read**, **write**.

Код 11: Си

```c
#include <sys/types.h>
#include <unistd.h>
ssize_t read(int fd, void *buf, size_t nbytes);
ssize_t write(int fd, const void *buf, size_t nbytes);
```

Код 12: Python

```python
conn.send('Hello World!')
data = conn.recv(BUFFER_SIZE)
```

Сервер TCP на Pyhton

Создаем TCP сокет, который слушает порт 5005 на локальной машине.

Код 13: Python

```python
import socket

TCP_IP = '127.0.0.1'
TCP_PORT = 5005
BUFFER_SIZE = 20  # Normally 1024,
                  # but we want fast response
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((TCP_IP, TCP_PORT))
s.listen(1)
```

Сервер TCP на Pyhton

После запроса клиента и установки с ним соединения, сервер в бесконечном цикле принимает от него данные, выводит их на экран и возвращает обратно. (Эхо-сервер)

Код 14: Python

```
conn, addr = s.accept()
print("Connection address: {}".format(addr))
while 1:
    data = conn.recv(BUFFER_SIZE)
    print("received data: {}".format(data))
    conn.send(data)  # echo
conn.close()
```

Клиент TCP

TCP соединение всегда можно установить при помощи утилиты **Telnet**.

Код 15: Telnet

```
$ telnet localhost 5005
```

Клиент TCP на Python

Создаем сокет и отправляем запрос на установку соединения с сервером.

Код 16: Python

```python
import socket

TCP_IP = '127.0.0.1'
TCP_PORT = 5005
BUFFER_SIZE = 1024
MESSAGE = "Hello, World!"

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((TCP_IP, TCP_PORT))
```

Клиент TCP на Python

Отправляем данные, получаем ответ сервера и закрываем соединение.

Код 17: Python

```
s.send(MESSAGE)
data = s.recv(BUFFER_SIZE)
s.close()

print("received data: {}".format(data))
```

Клиент HTTP на Python

Устанавливаем TCP соединение с **httpbin.org** и отправляем строку запроса.

Код 18: Python

```
sock_obj = socket.socket(socket.AF_INET,
                         socket.SOCK_STREAM)

sock_obj.connect(('httpbin.org', 80))
sock_obj.send("GET /ip HTTP/1.0\n\n")
```