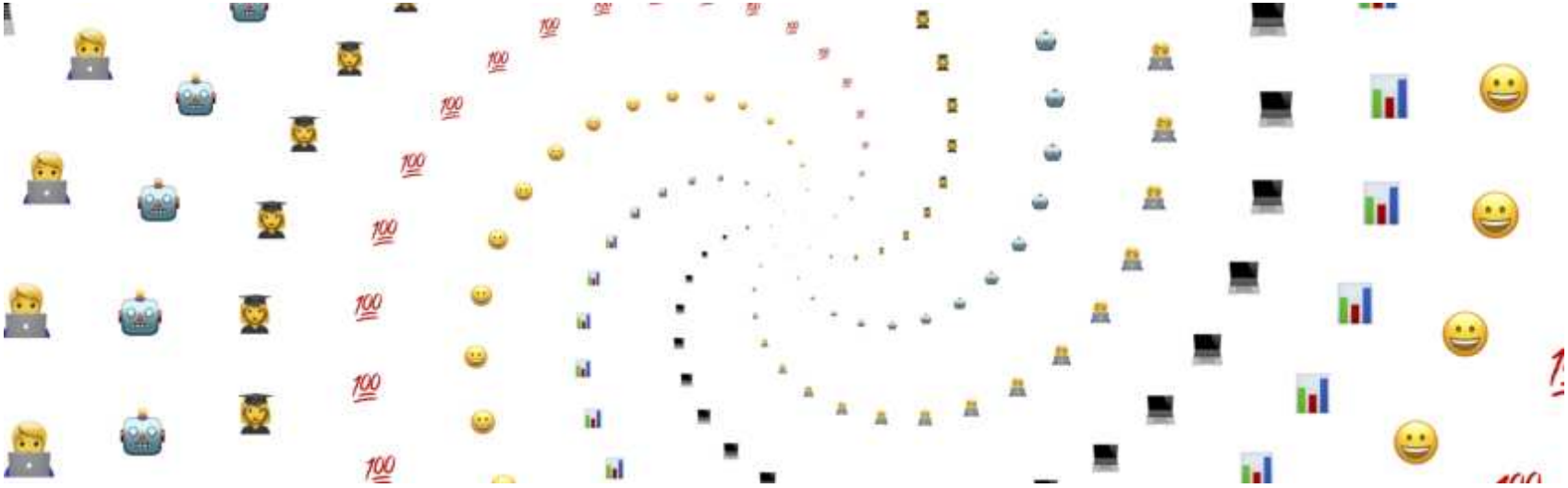
Chat-Programmierung mit Sockets

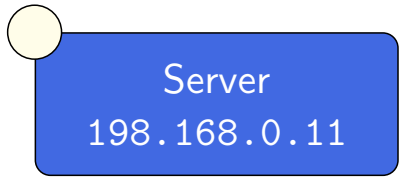
Cyril Blum

Fachschaft Informatik
Kantonsschule im Lee



Sockets

Client-to-Server



Client:

Server:
1. Socket erstellen (`.socket()`)

Sockets

Client-to-Server

Client:

Server:
1. Socket erstellen (`.socket()`)
2. Socket binden (`.bind()`)

Sockets

Client-to-Server

Client:

Server:

1. Socket erstellen (`.socket()`)
2. Socket binden (`.bind()`)
3. Warten auf Verbindungen (`.listen()`)

Sockets

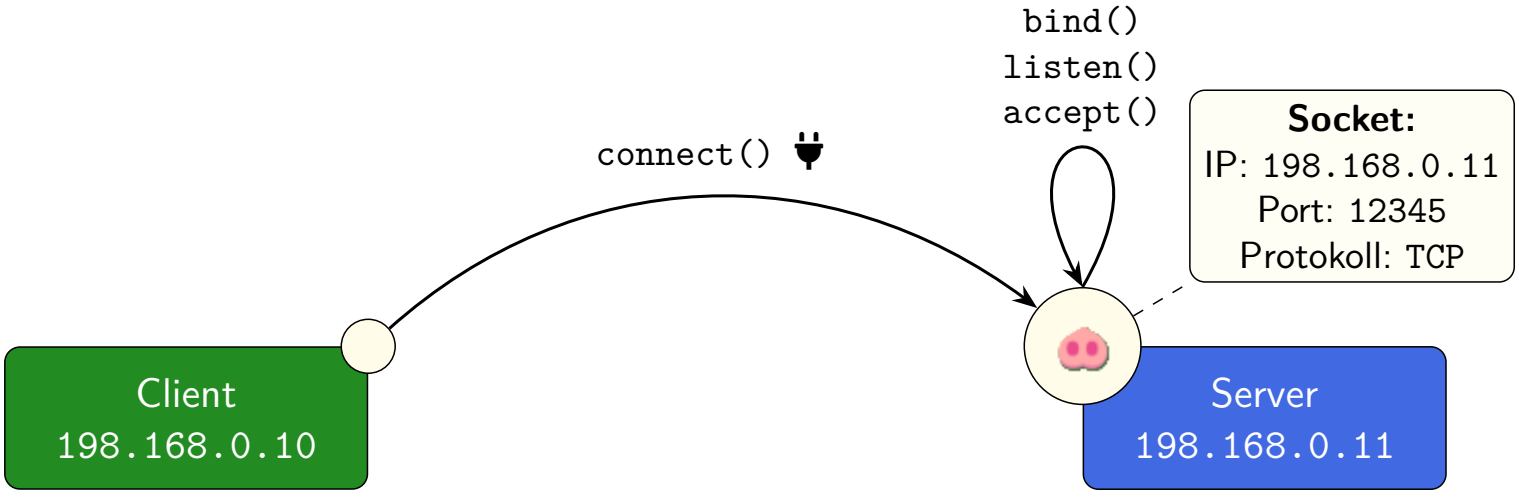
Client-to-Server

Client:
1. Socket erstellen (`.socket()`)

Server:
1. Socket erstellen (`.socket()`)
2. Socket binden (`.bind()`)
3. Warten auf Verbindungen (`.listen()`)

Sockets

Client-to-Server

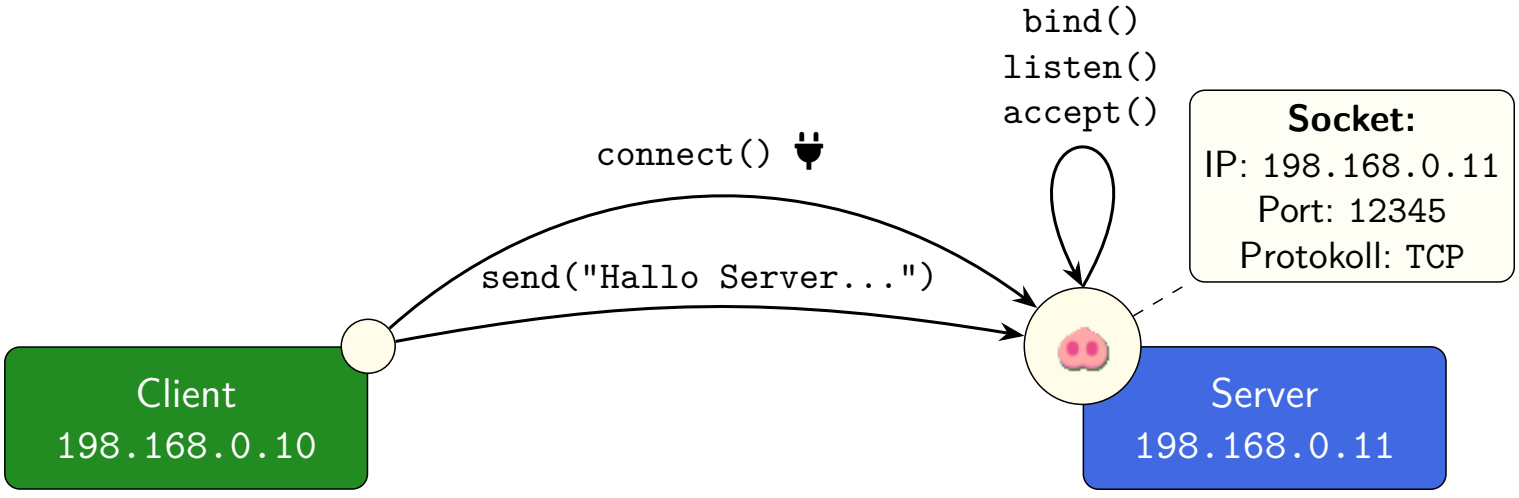


Client:
1. Socket erstellen (`.socket()`)
2. Mit Server-Socket verbinden (`.connect()`)

Server:
1. Socket erstellen (`.socket()`)
2. Socket binden (`.bind()`)
3. Warten auf Verbindungen (`.listen()`)
4. Accept Verbindung (`.accept()`)

Sockets

Client-to-Server

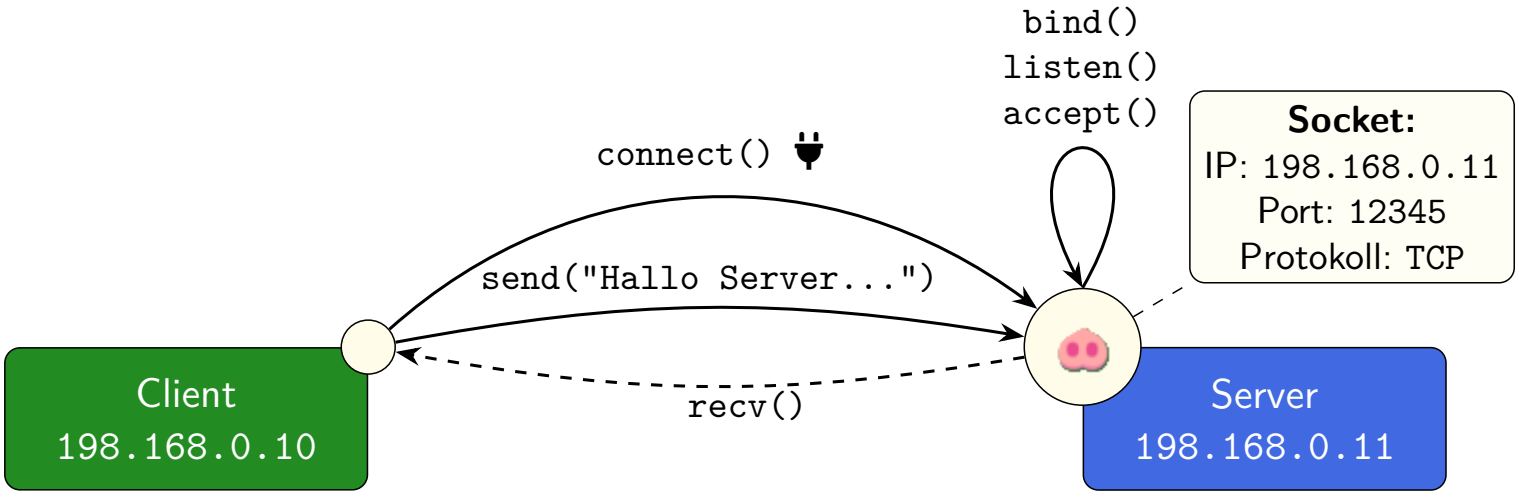


Client:
1. Socket erstellen (`.socket()`)
2. Mit Server-Socket verbinden (`.connect()`)
3. Nachricht senden (`.send()`)

Server:
1. Socket erstellen (`.socket()`)
2. Socket binden (`.bind()`)
3. Warten auf Verbindungen (`.listen()`)
4. Accept Verbindung (`.accept()`)

Sockets

Client-to-Server

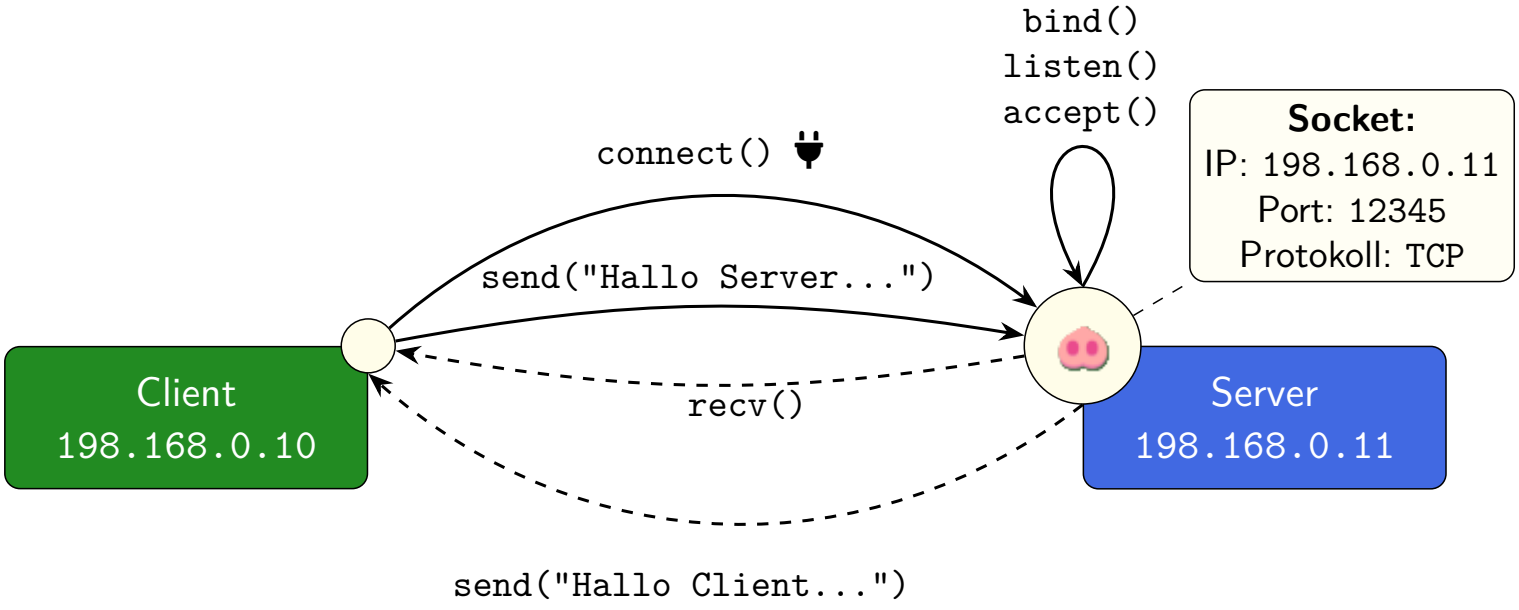


Client:
1. Socket erstellen (`.socket()`)
2. Mit Server-Socket verbinden (`.connect()`)
3. Nachricht senden (`.send()`)

Server:
1. Socket erstellen (`.socket()`)
2. Socket binden (`.bind()`)
3. Warten auf Verbindungen (`.listen()`)
4. Accept Verbindung (`.accept()`)
5. Nachricht empfangen (`.recv()`)

Sockets

Client-to-Server

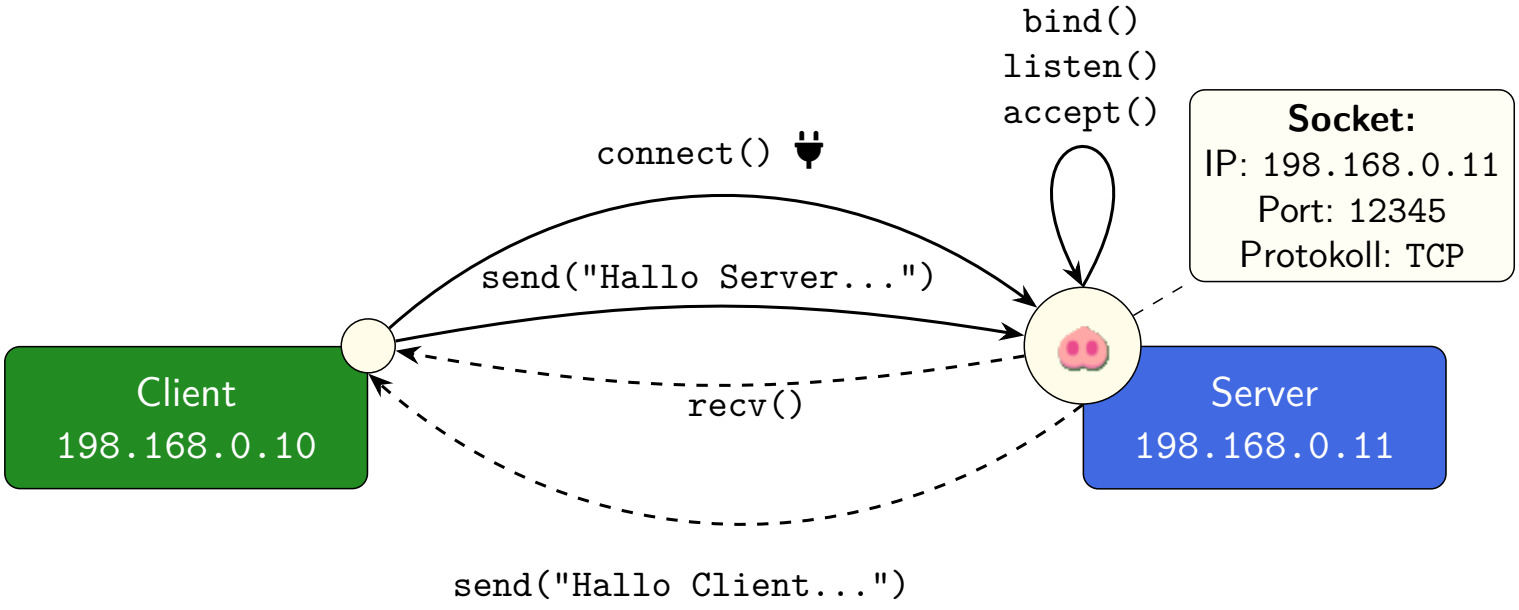


Client:
1. Socket erstellen (`.socket()`)
2. Mit Server-Socket verbinden (`.connect()`)
3. Nachricht senden (`.send()`)

Server:
1. Socket erstellen (`.socket()`)
2. Socket binden (`.bind()`)
3. Warten auf Verbindungen (`.listen()`)
4. Accept Verbindung (`.accept()`)
5. Nachricht empfangen (`.recv()`)
6. Antwort senden (`.send()`)

Sockets

Client-to-Server

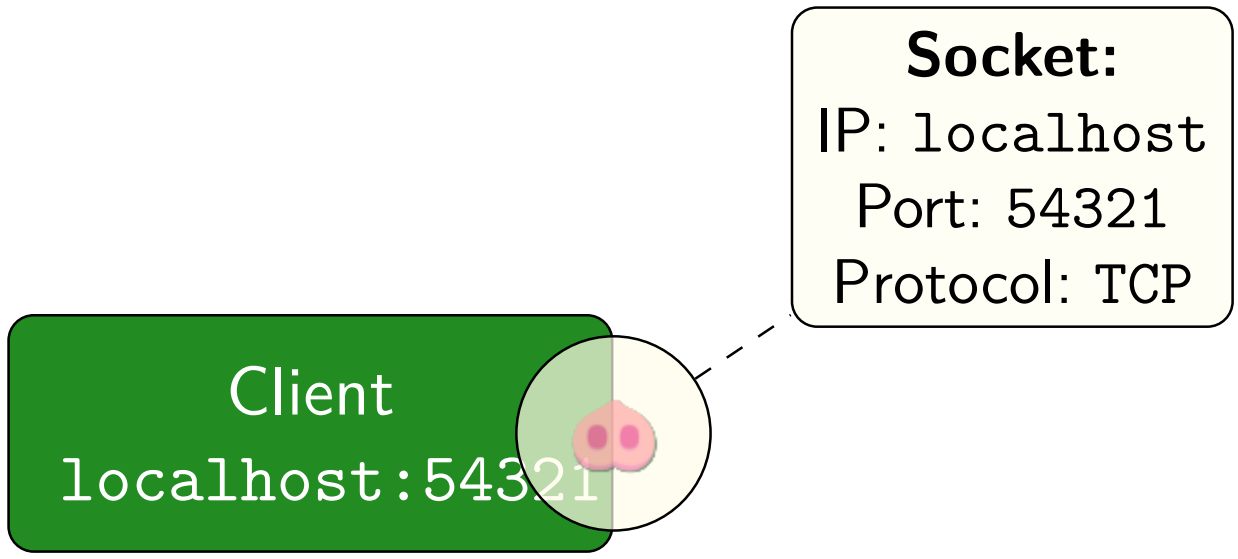


Client:
1. Socket erstellen (`.socket()`)
2. Mit Server-Socket verbinden (`.connect()`)
3. Nachricht senden (`.send()`)
4. Antwort empfangen (`.recv()`)

Server:
1. Socket erstellen (`.socket()`)
2. Socket binden (`.bind()`)
3. Warten auf Verbindungen (`.listen()`)
4. Accept Verbindung (`.accept()`)
5. Nachricht empfangen (`.recv()`)
6. Antwort senden (`.send()`)

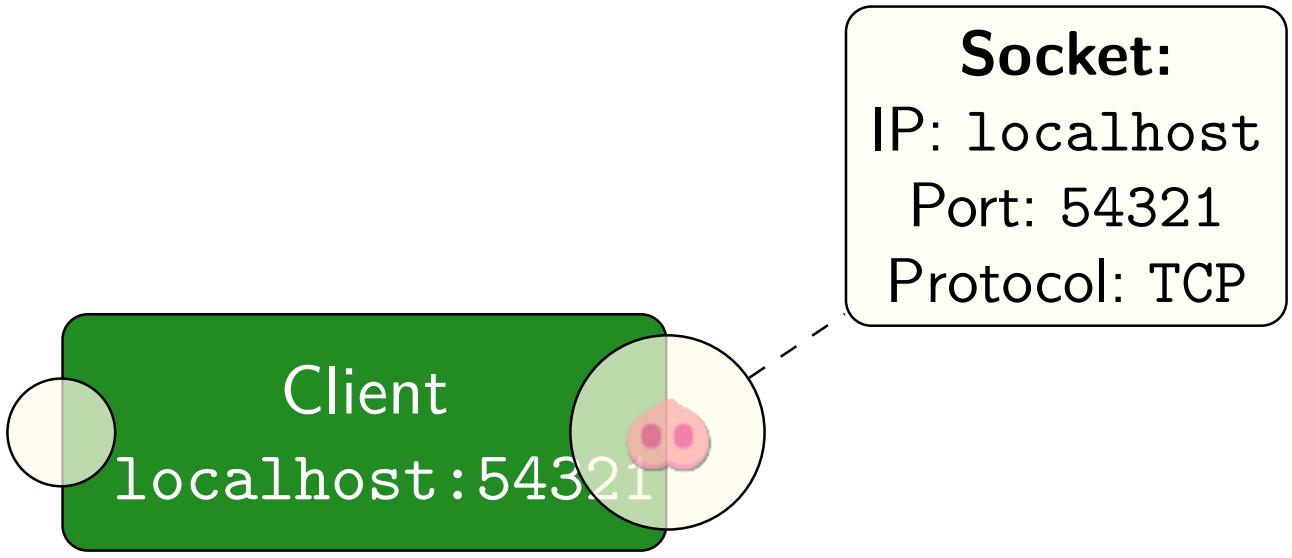
Sockets

Client-to-Client (`localhost`)



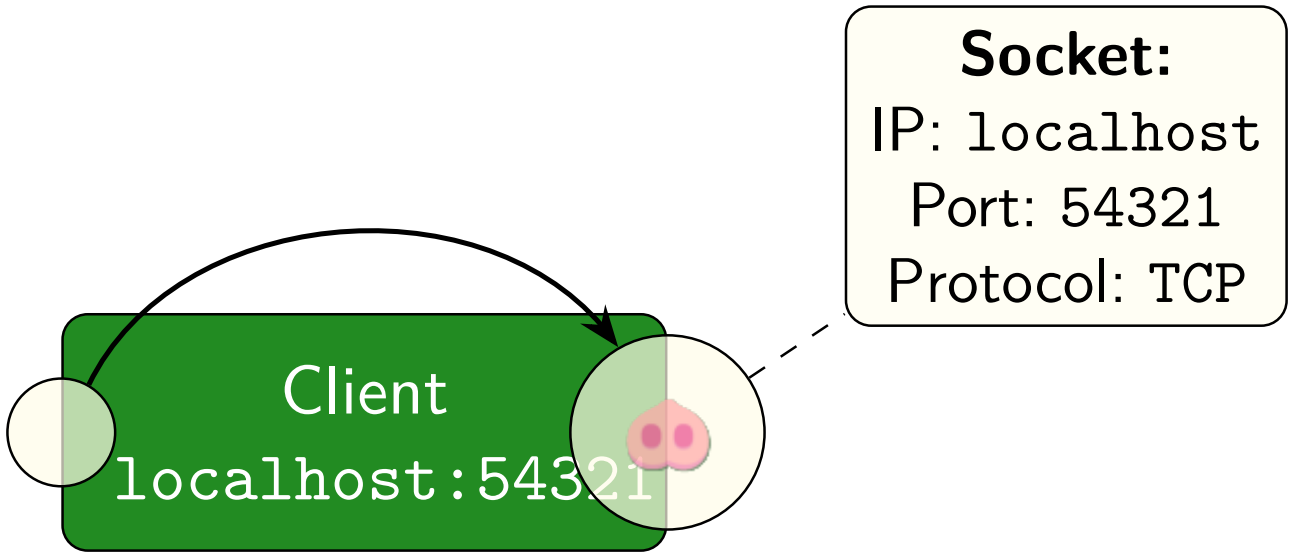
Sockets

Client-to-Client (`localhost`)



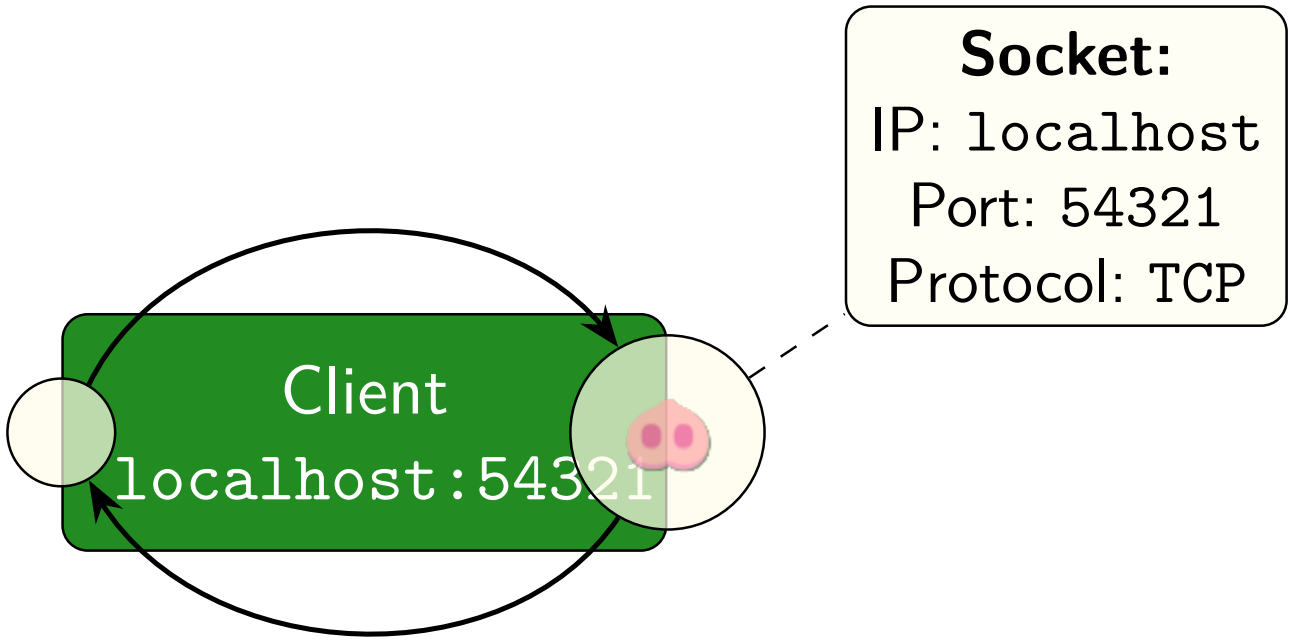
Sockets

Client-to-Client (`localhost`)



Sockets

Client-to-Client (`localhost`)



Auftrag (Skript Netzwerke)

Client-to-Server über `localhost`

2.1 - 2.4

Auftrag (Skript Netzwerke)

Client-to-Server über LAN

2.5 - 2.9

Sockets

Client-to-Client (Relay-Server)

Client 1 (Alice)
`51.154.56.61`

Client 2 (Bob)
`51.154.56.61`

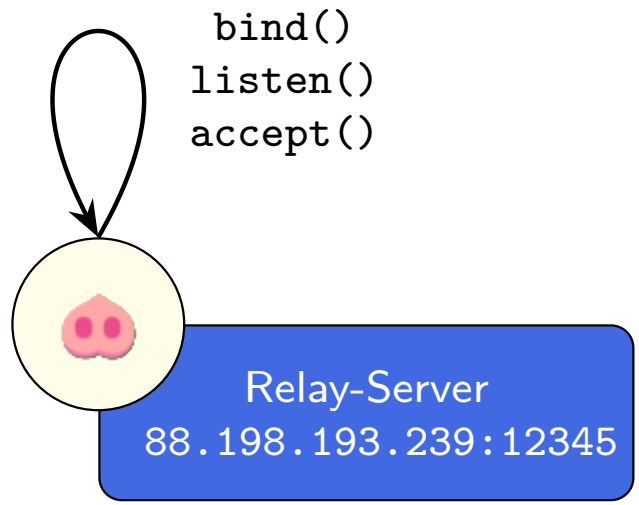
Relay-Server
`88.198.193.239:12345`

Sockets

Client-to-Client (Relay-Server)

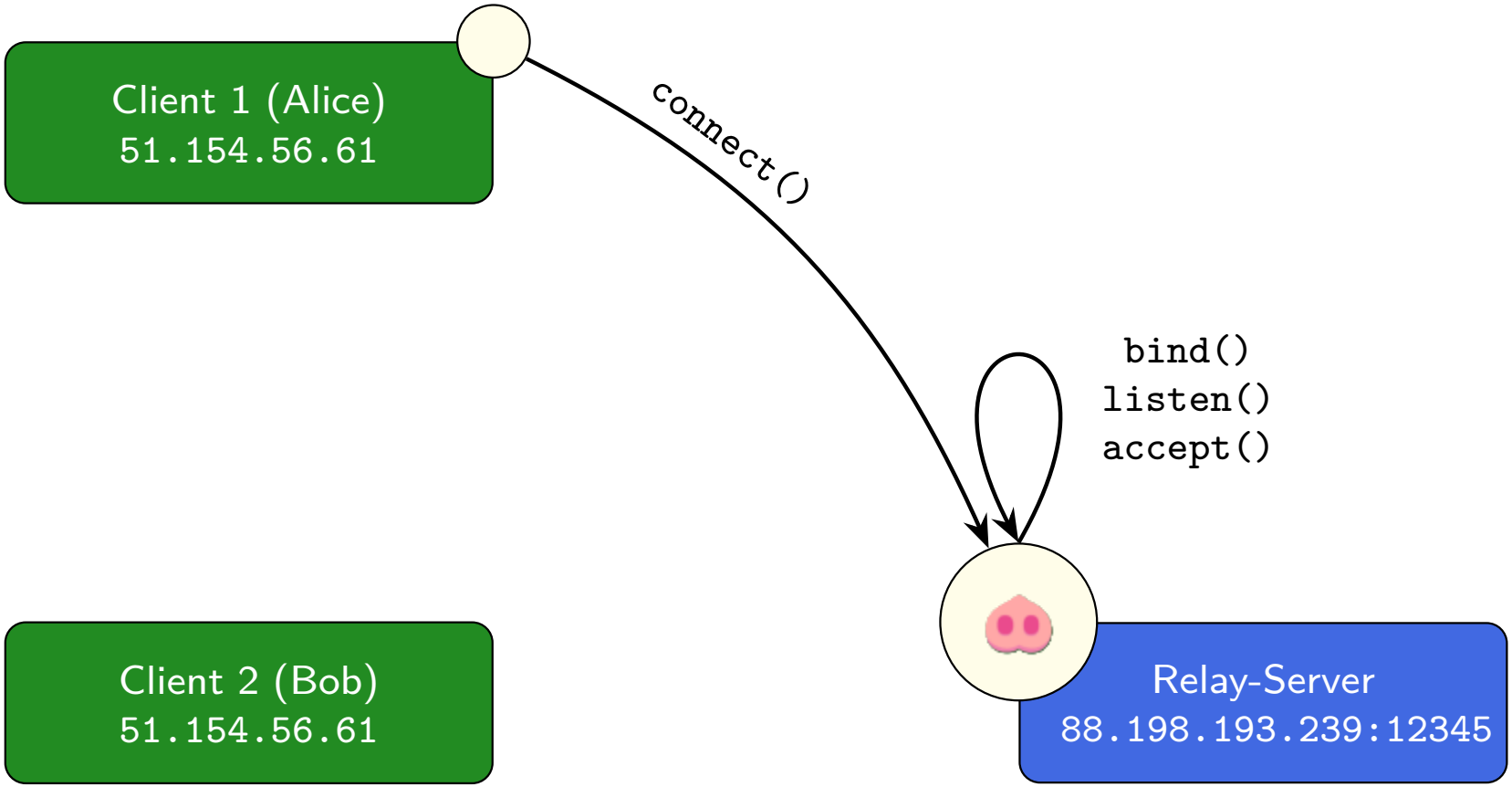
Client 1 (Alice)
51.154.56.61

Client 2 (Bob)
51.154.56.61



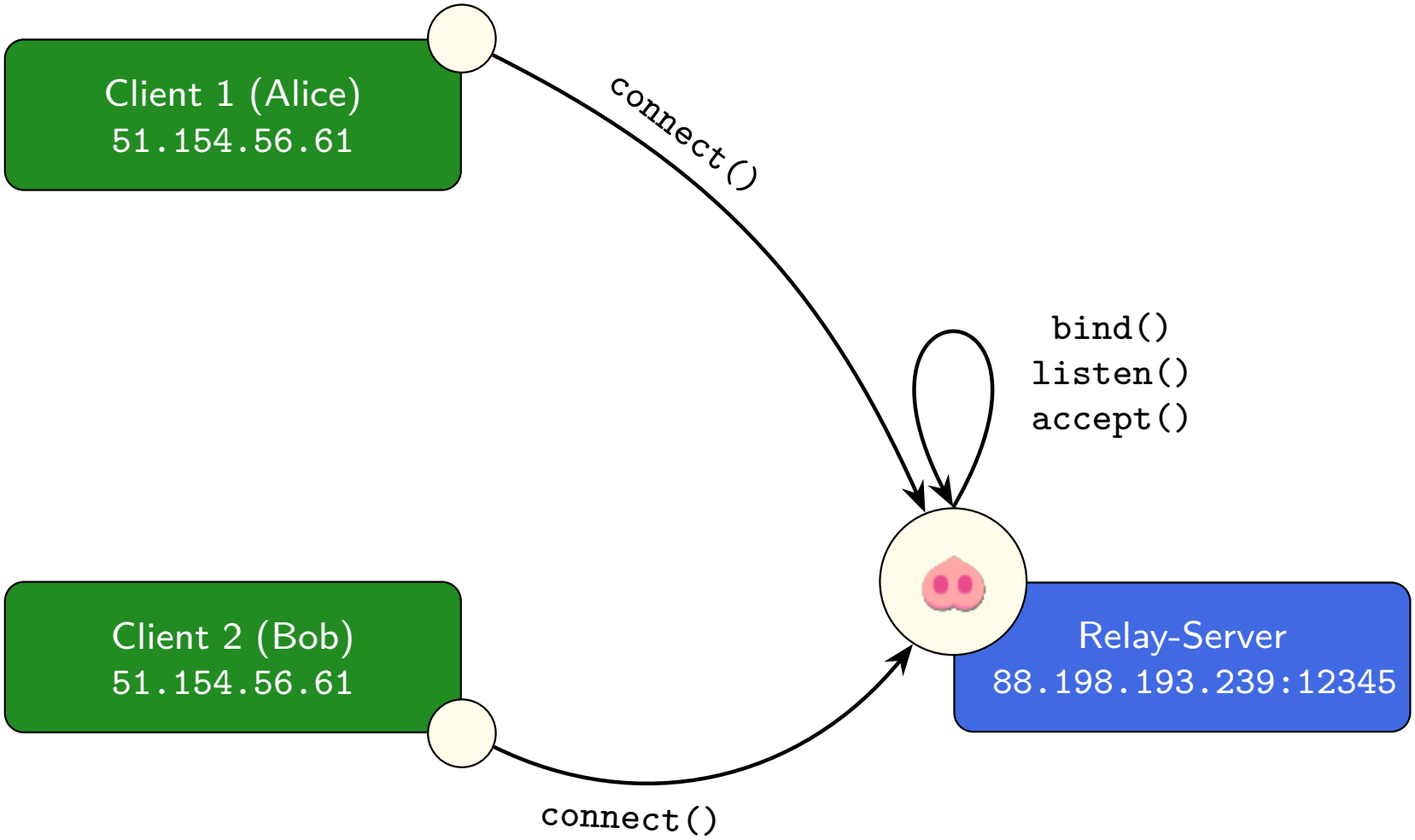
Sockets

Client-to-Client (Relay-Server)



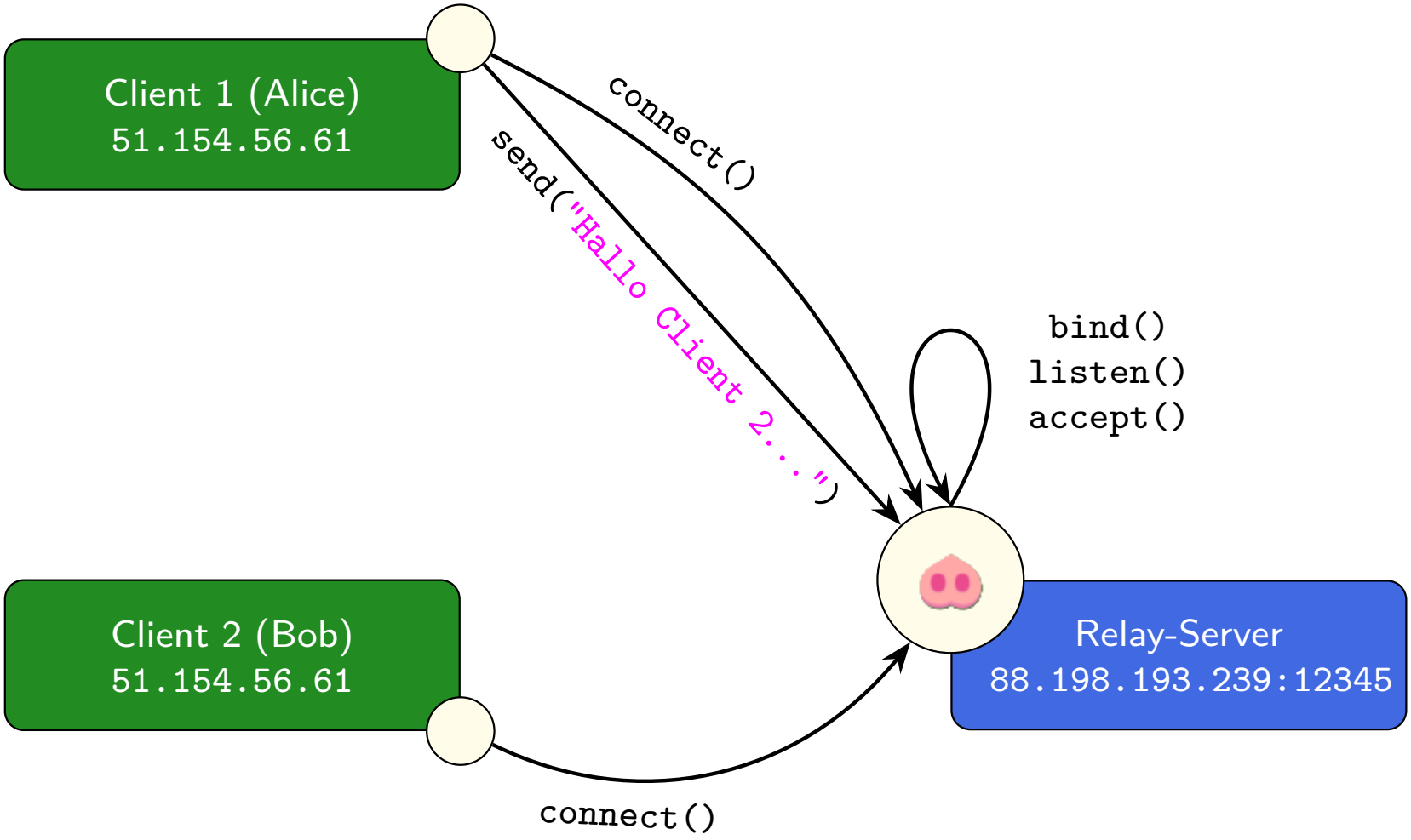
Sockets

Client-to-Client (Relay-Server)



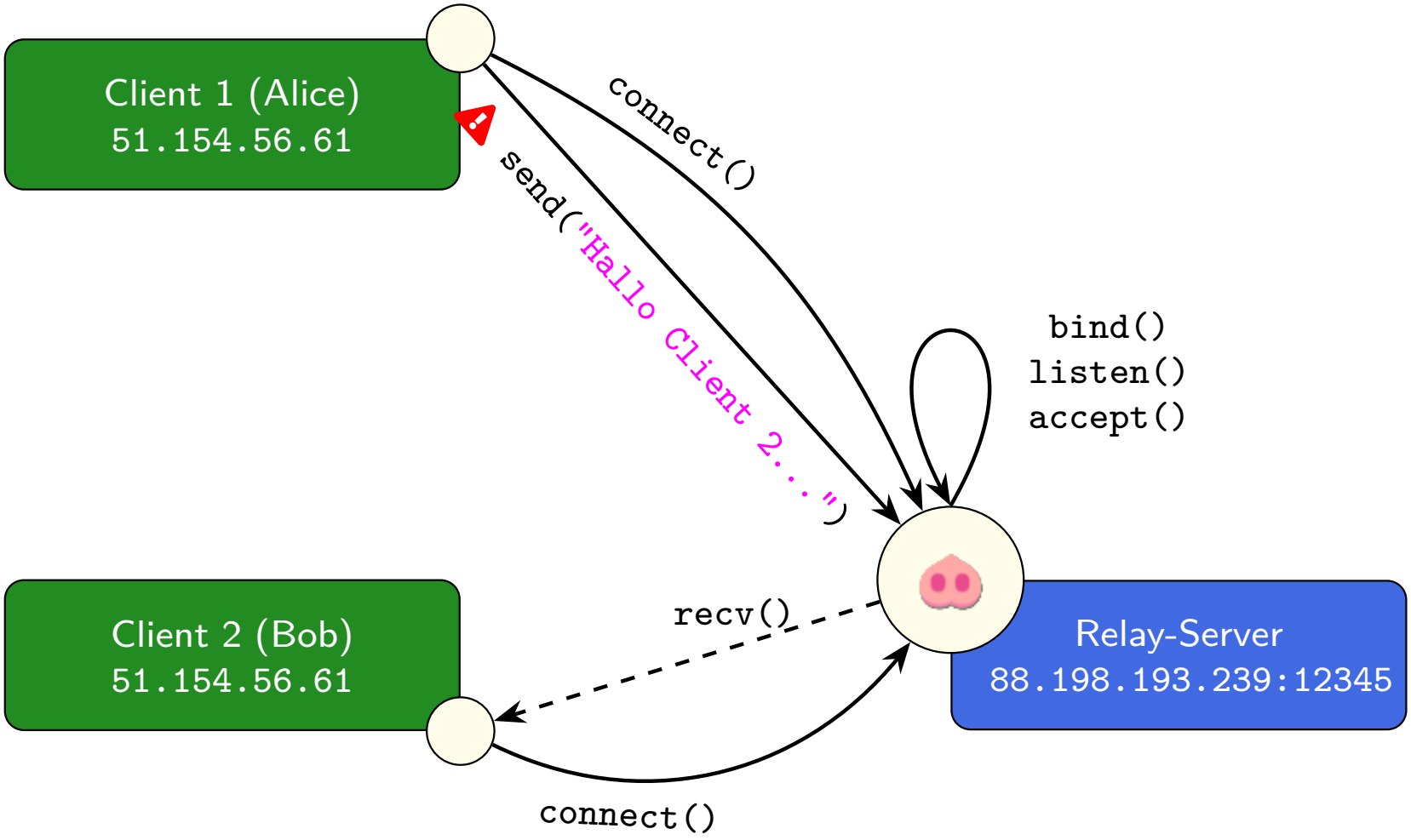
Sockets

Client-to-Client (Relay-Server)



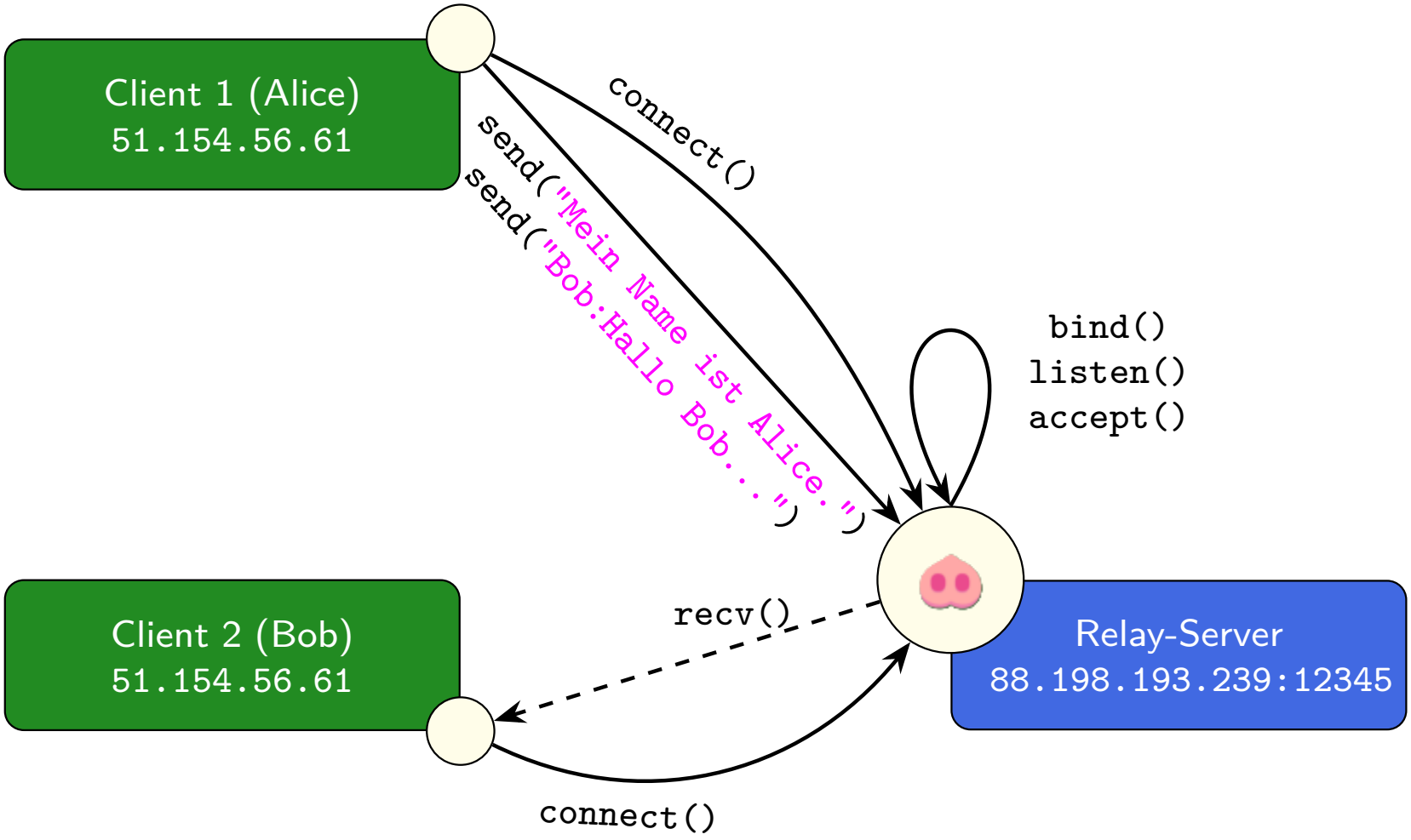
Sockets

Client-to-Client (Relay-Server)



Sockets

Client-to-Client (Relay-Server)



Client-to-Client über Relay-Server

Einige wichtie Neuerungen im Code:

- Server...

Client-to-Client über Relay-Server

Einige wichtie Neuerungen im Code:

- Server...
  - muss Verbindungsanfragen von mehreren Clients akzeptieren (z.B. mit **Threads**)

Client-to-Client über Relay-Server

Einige wichtie Neuerungen im Code:

- Server...
  - muss Verbindungsanfragen von mehreren Clients akzeptieren (z.B. mit **Threads**)
  - muss Nachrichten von einem Client empfangen und an den anderen weiterleiten

Client-to-Client über Relay-Server

Einige wichtie Neuerungen im Code:

- Server...
  - muss Verbindungsanfragen von mehreren Clients akzeptieren (z.B. mit **Threads**)
  - muss Nachrichten von einem Client empfangen und an den anderen weiterleiten
- Clients...

Client-to-Client über Relay-Server

Einige wichtie Neuerungen im Code:

- Server...
  - muss Verbindungsanfragen von mehreren Clients akzeptieren (z.B. mit **Threads**)
  - muss Nachrichten von einem Client empfangen und an den anderen weiterleiten
- Clients...
  - Müssen mit dem Relay-Server kommunizieren, anstatt direkt miteinander

Client-to-Client über Relay-Server

Einige wichtie Neuerungen im Code:

- Server...
  - muss Verbindungsanfragen von mehreren Clients akzeptieren (z.B. mit **Threads**)
  - muss Nachrichten von einem Client empfangen und an den anderen weiterleiten
- Clients...
  - Müssen mit dem Relay-Server kommunizieren, anstatt direkt miteinander
  - Verwenden nicht die IP-Adresse des anderen Clients, sondern die des Relay-Servers, um sich zu verbinden

Client-to-Client über Relay-Server

Einige wichtie Neuerungen im Code:

- Server...
  - muss Verbindungsanfragen von mehreren Clients akzeptieren (z.B. mit **Threads**)
  - muss Nachrichten von einem Client empfangen und an den anderen weiterleiten
- Clients...
  - Müssen mit dem Relay-Server kommunizieren, anstatt direkt miteinander
  - Verwenden nicht die IP-Adresse des anderen Clients, sondern die des Relay-Servers, um sich zu verbinden
  - Adressieren ihre Nachrichten an den Namen des anderen Clients, damit der Relay-Server weiss, an wen er die Nachricht weiterleiten soll

Client-to-Client über Relay-Server

Einige wichtie Neuerungen im Code:

- Server...
  - muss Verbindungsanfragen von mehreren Clients akzeptieren (z.B. mit **Threads**)
  - muss Nachrichten von einem Client empfangen und an den anderen weiterleiten
- Clients...
  - Müssen mit dem Relay-Server kommunizieren, anstatt direkt miteinander
  - Verwenden nicht die IP-Adresse des anderen Clients, sondern die des Relay-Servers, um sich zu verbinden
  - Adressieren ihre Nachrichten an den Namen des anderen Clients, damit der Relay-Server weiss, an wen er die Nachricht weiterleiten soll
  - Verwenden ebenfalls **Threads**, um gleichzeitig Nachrichten vom Relay-Server empfangen und senden zu können

Relay-Server

Client-to-Client über Relay-Server, im LAN / WAN

- LAN: 2.10 - 2.14
- WAN: 2.15 - 2.17

Tipps:

- Bearbeiten Sie nur `client.py`
- Passwort für `Mein sicheres Netzwerk`: `CVW-1234`
- IP-Adresse des Relay-Servers: `192.168.0.`