Multiplexing & Demultiplexing in UDP

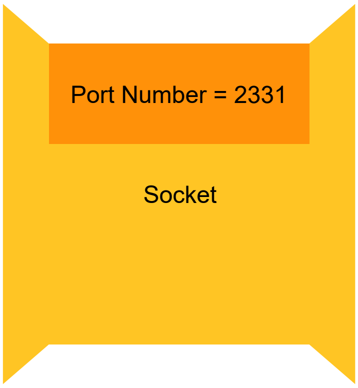
Connectionless refers to multiplexing and demultiplexing with UDP. Let's dive right in.

WE'LL COVER THE FOLLOWING

Ports

Here’s a quick refresher on what ports are because that needs to be crystal clear in order for you to understand multiplexing and demultiplexing.

- **Sockets**, which are gateways to applications, are identified by a combination of an **IP address** and a 16-bit **port number**. That means $2^{16} = 65536$ port numbers exist. However, they start from port 0 so they exist in the range of $0 - 65535$.



Sockets have associated port numbers!

- Out of these, the port numbers $0 - 1023$ are **well-known** and are reserved for certain standard protocols. Port 80, for instance, is reserved for HTTP whereas port 22 is reserved for SSH.

- Refer to page 16 of RFC 1700 for more details regarding what port number is assigned to what protocol.

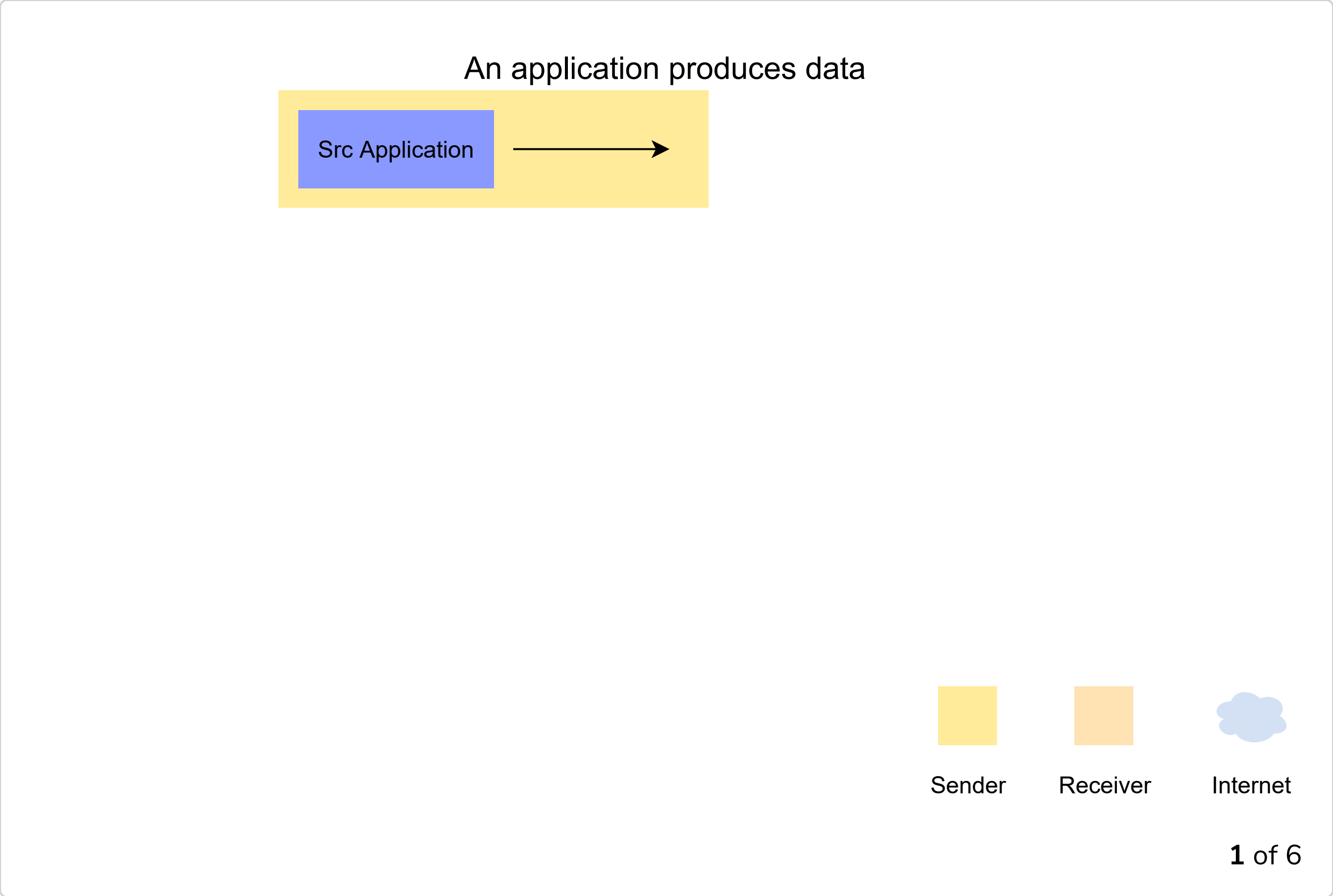
Multiplexing & Demultiplexing in UDP

- When a datagram is sent out from an application, the port number of the associated **source** and **destination** application is appended to it in the UDP protocol header.

- When the datagram is received at the receiving host, it sends the datagram off to the relevant application’s socket based on **destination port number**.

- If the source port and source IP address of two datagrams are different but the destination port and IP address are the same, the datagrams will still get sent to the same application.

Here are some slides to give you a quick overview of what happens.



An application produces data

Src Application

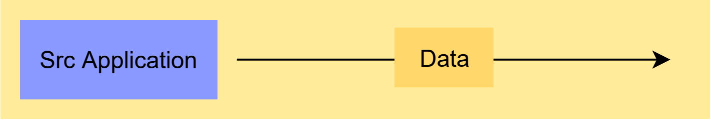
Sender

Receiver

Internet

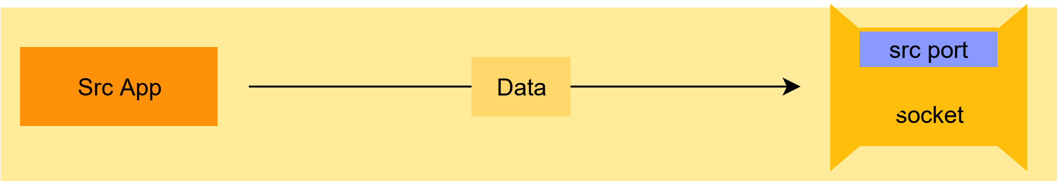
1 of 6

An application produces data



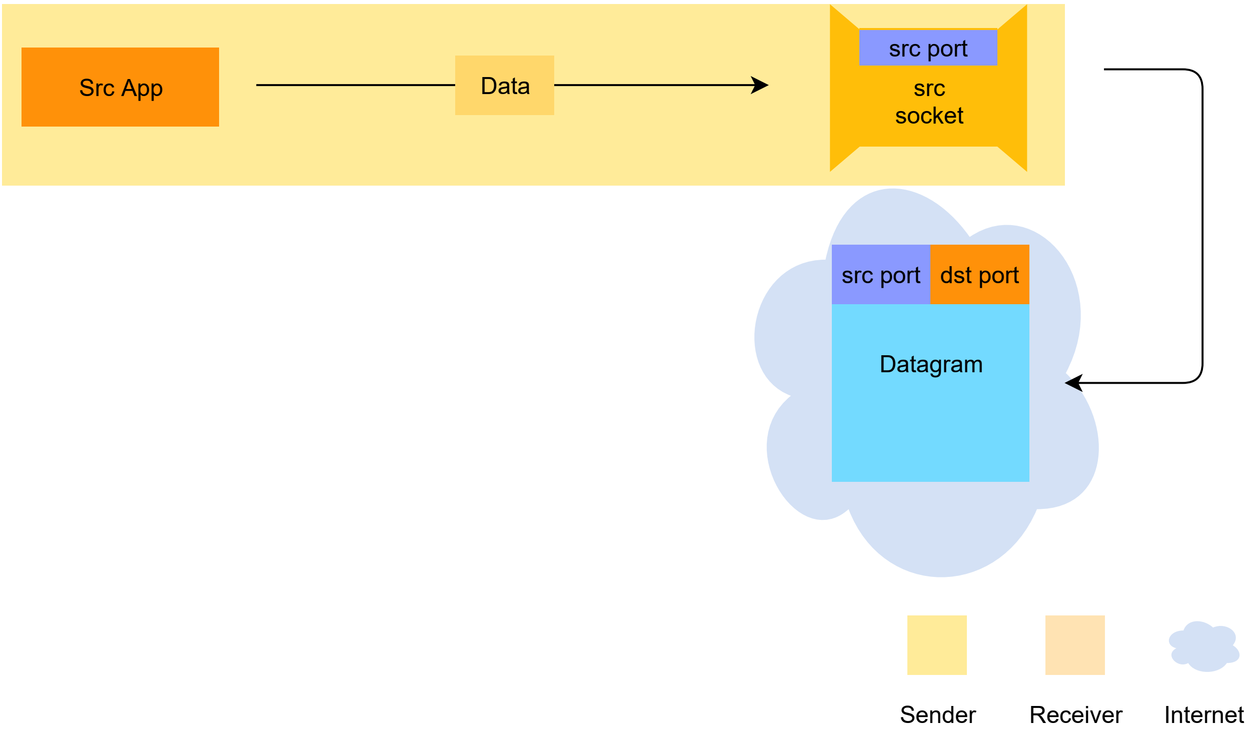
Sender Receiver Internet

It writes the data out to its socket and it turns into a datagram



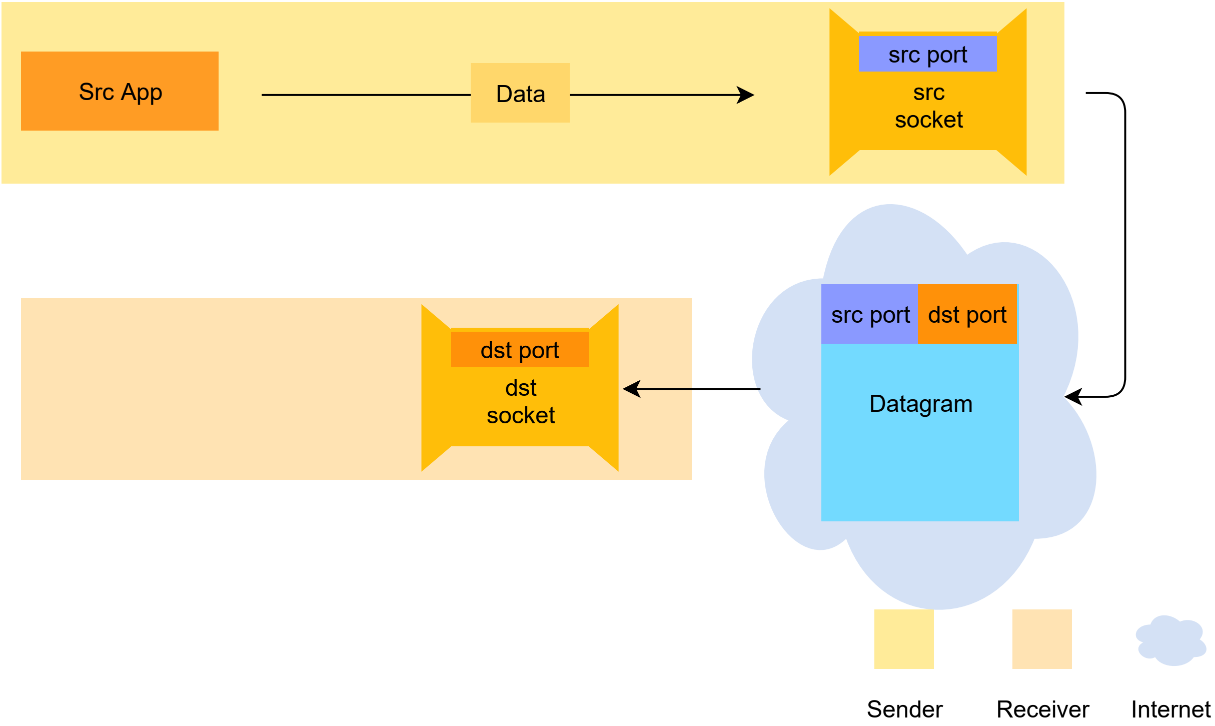
Sender Receiver Internet

It writes data out to its socket and turns it into a datagram



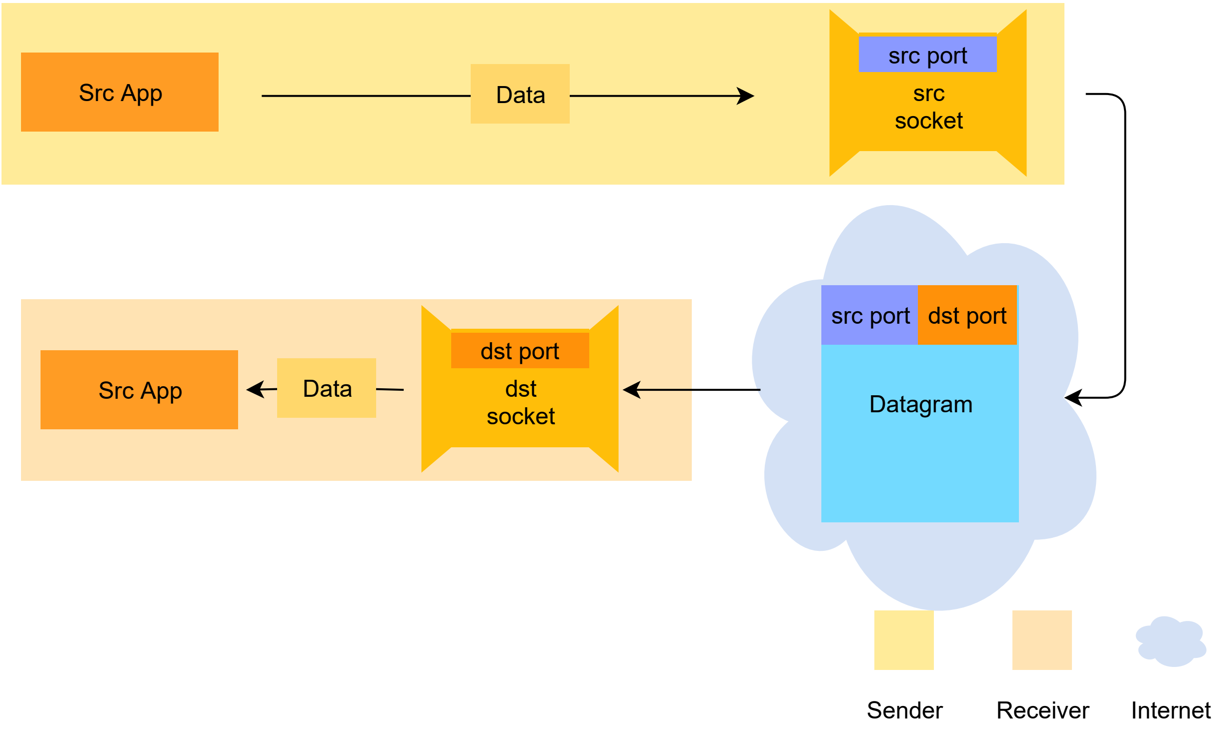
4 of 6

The datagram is received by the receiving end's socket



5 of 6

The data is then received by the application



6 of 6

On Port Assignment in UDP

It’s far more common to let the port on the client-side of an application be assigned dynamically instead of choosing *a* particular port. This is because for communication, both parties must be able to identify each other. Since the client *initiates* communication to the server, it must know the port number of the application on the server. However, the server doesn’t need to know the client application’s port number in advance. When the first datagram from the client reaches the server, it will carry the client port number, which the server can use to send datagrams back to the client.

However, server-side applications generally do not use dynamically allocated ports! This is because they are running well-known protocols like HTTP and need to be bound to specific ports.

Quick Quiz!

1 Ports 0 - 1023 are termed as ephemeral ports.

COMPLETED 0%

1 of 3 ‹ ›

Let’s look at the principles of reliable data transfer in the next lesson. This is key to building a good foundation for later lesson on TCP!