

# The Greatest Network Simulation Architecture Of All Time

Hongjian Lan

Yamei Ou

Samuel Richerd

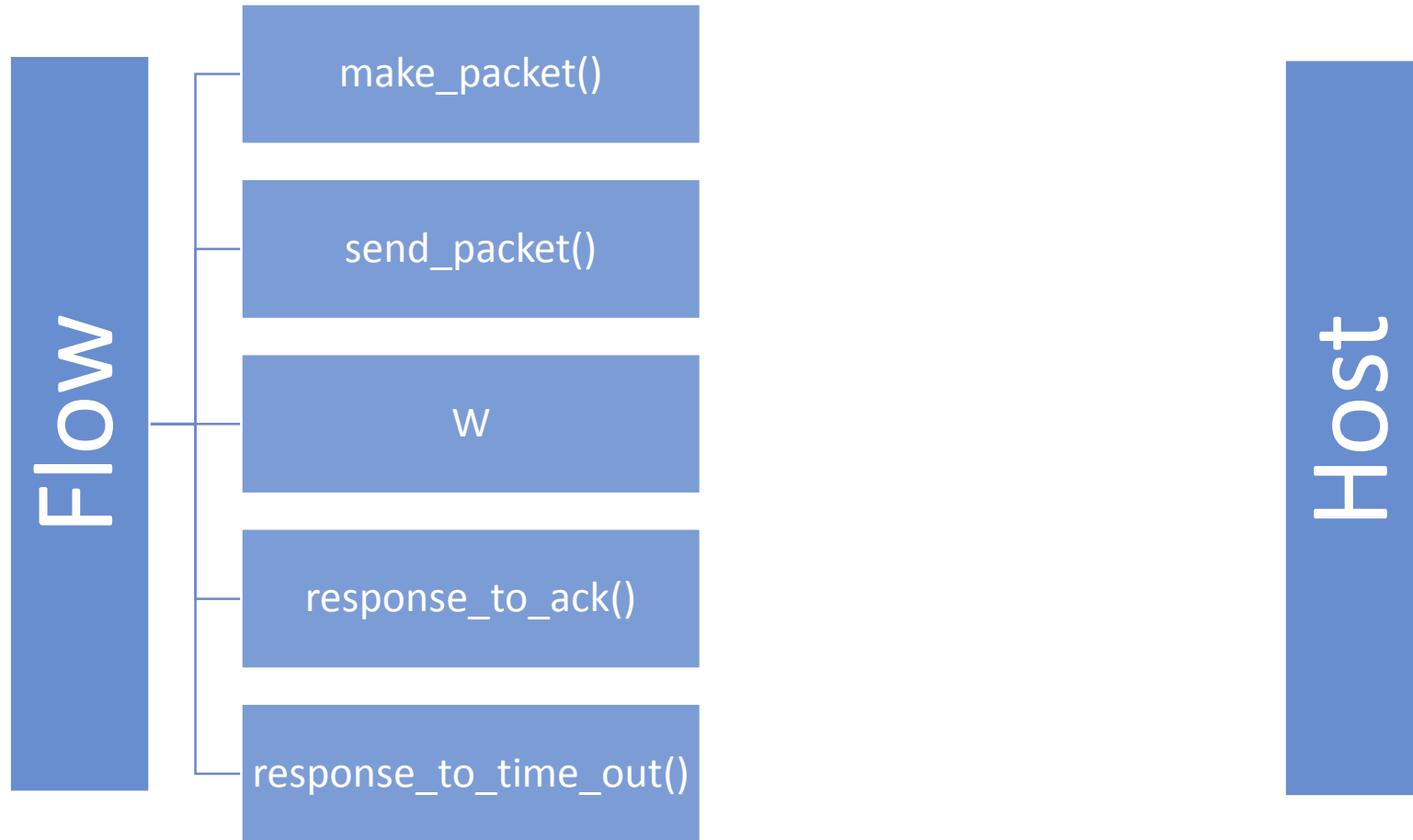
Jan Van Bruggen

Junlin Zhang

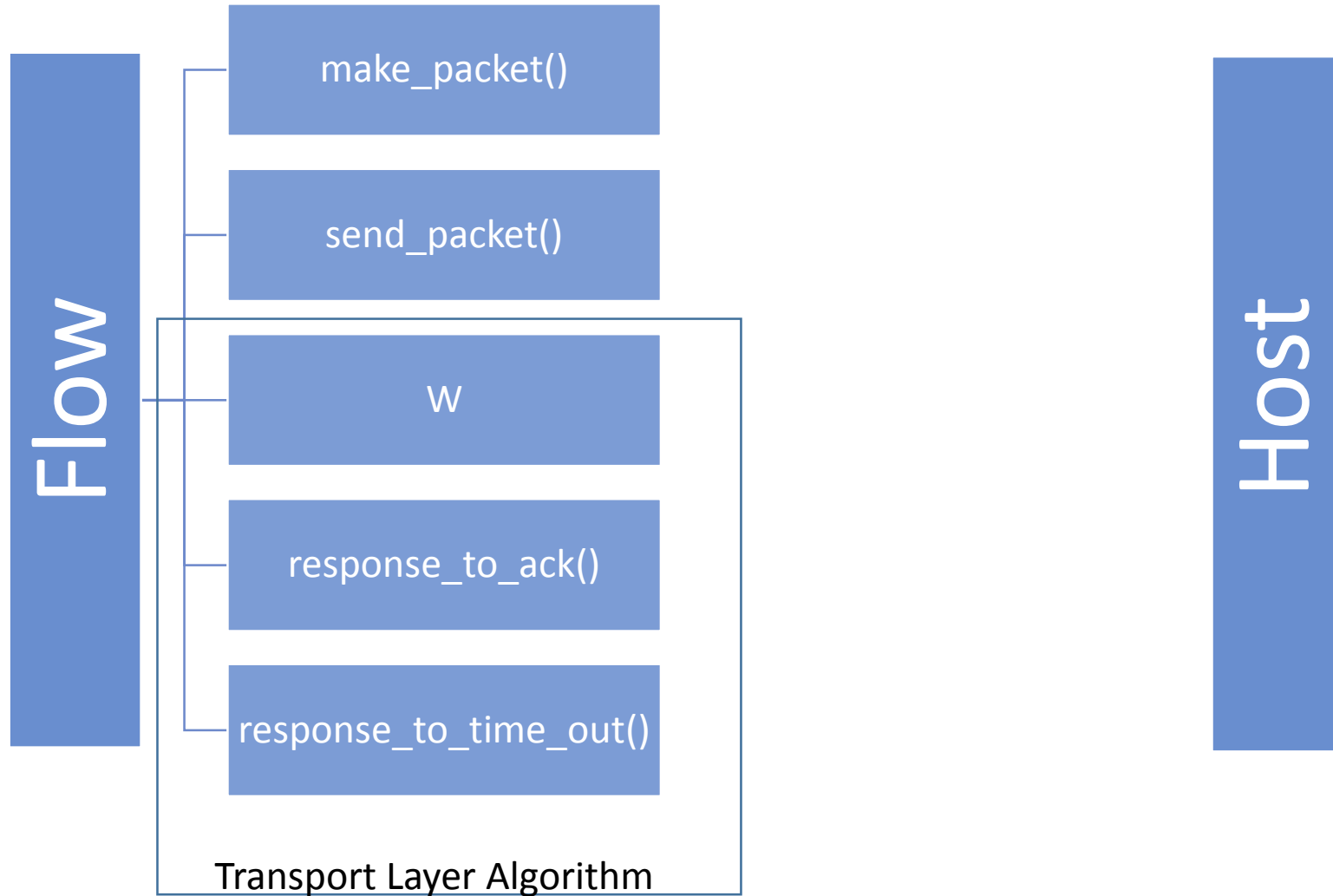
# Flows

Flows generate Packets and control congestion

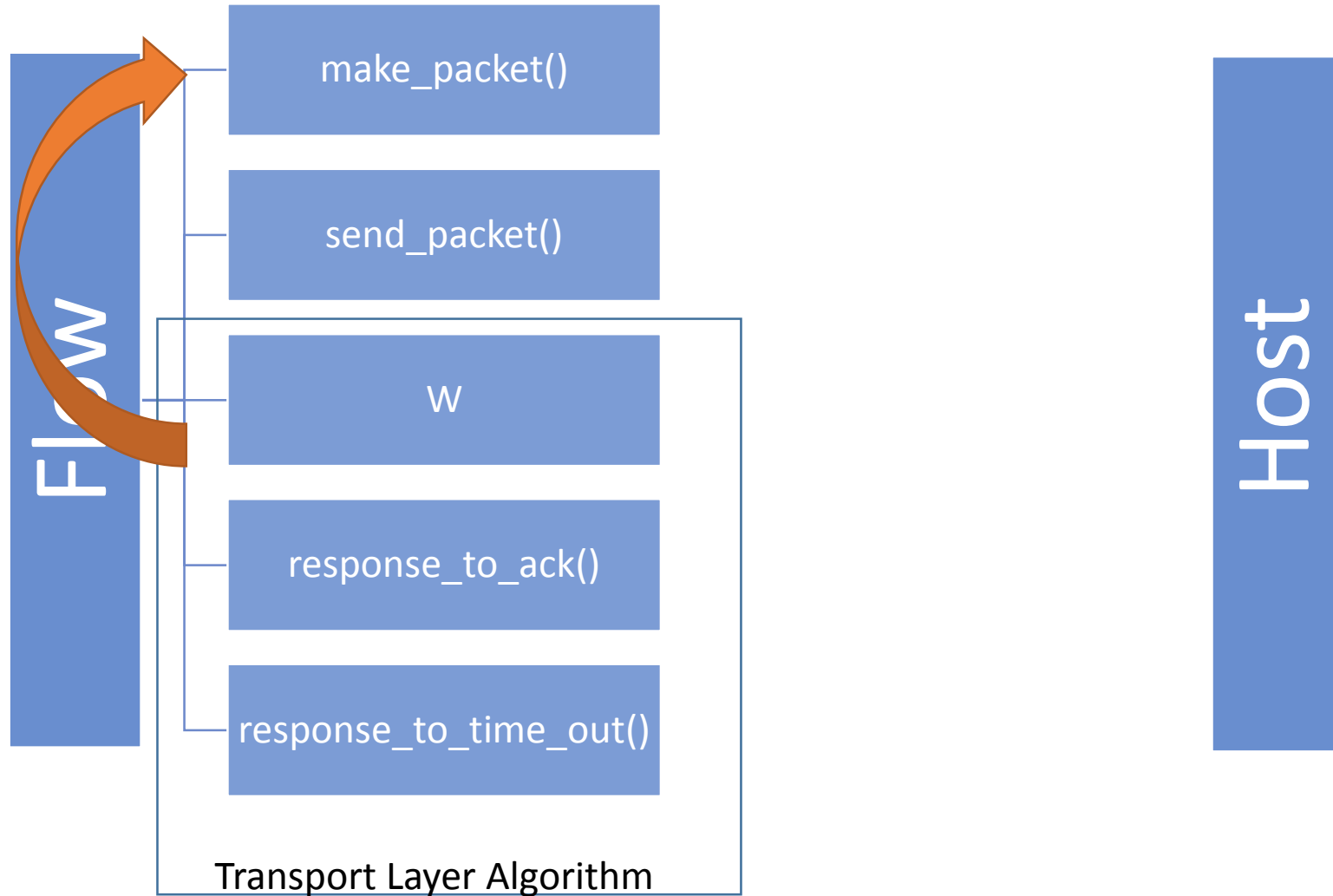
# Flow



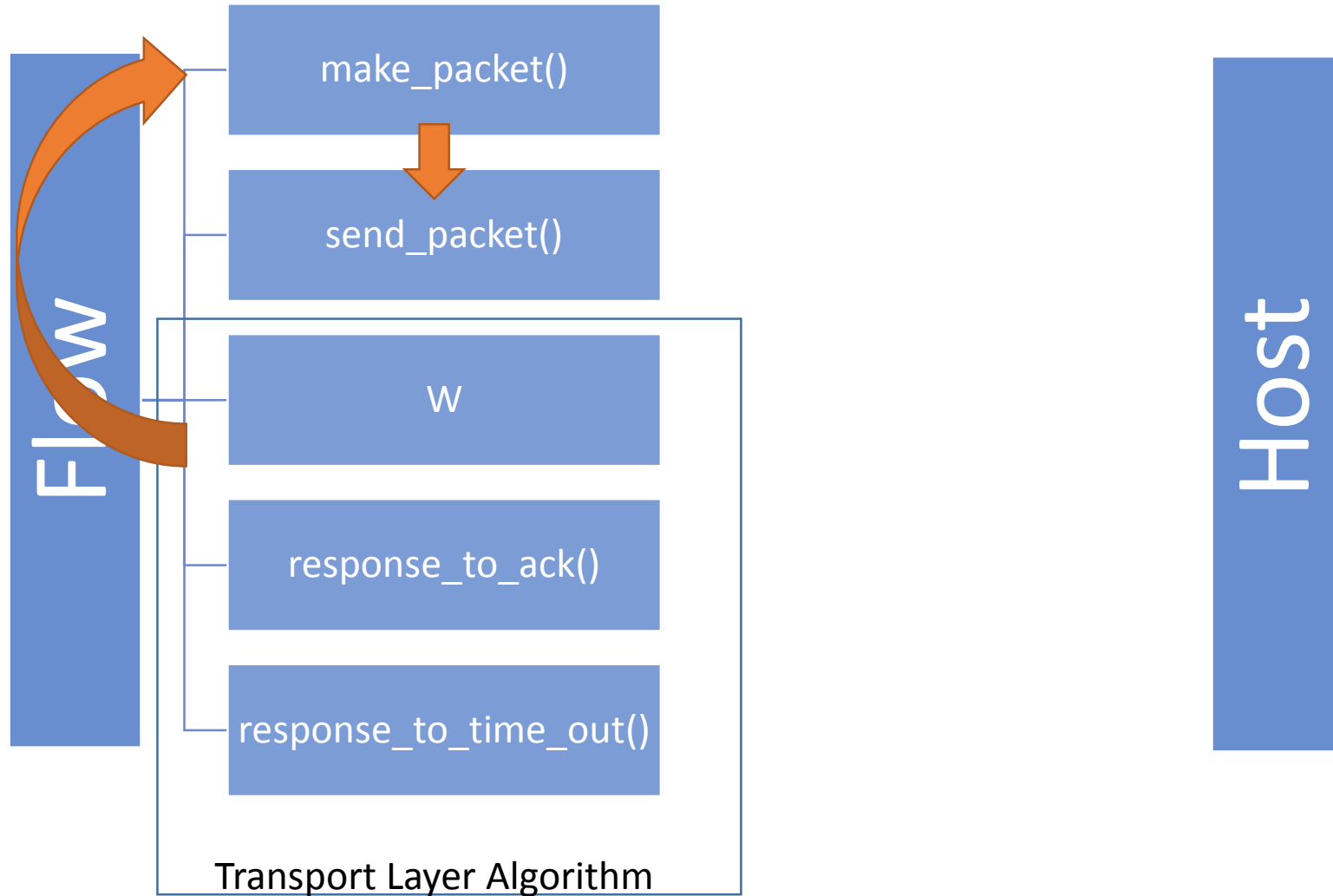
# Flow



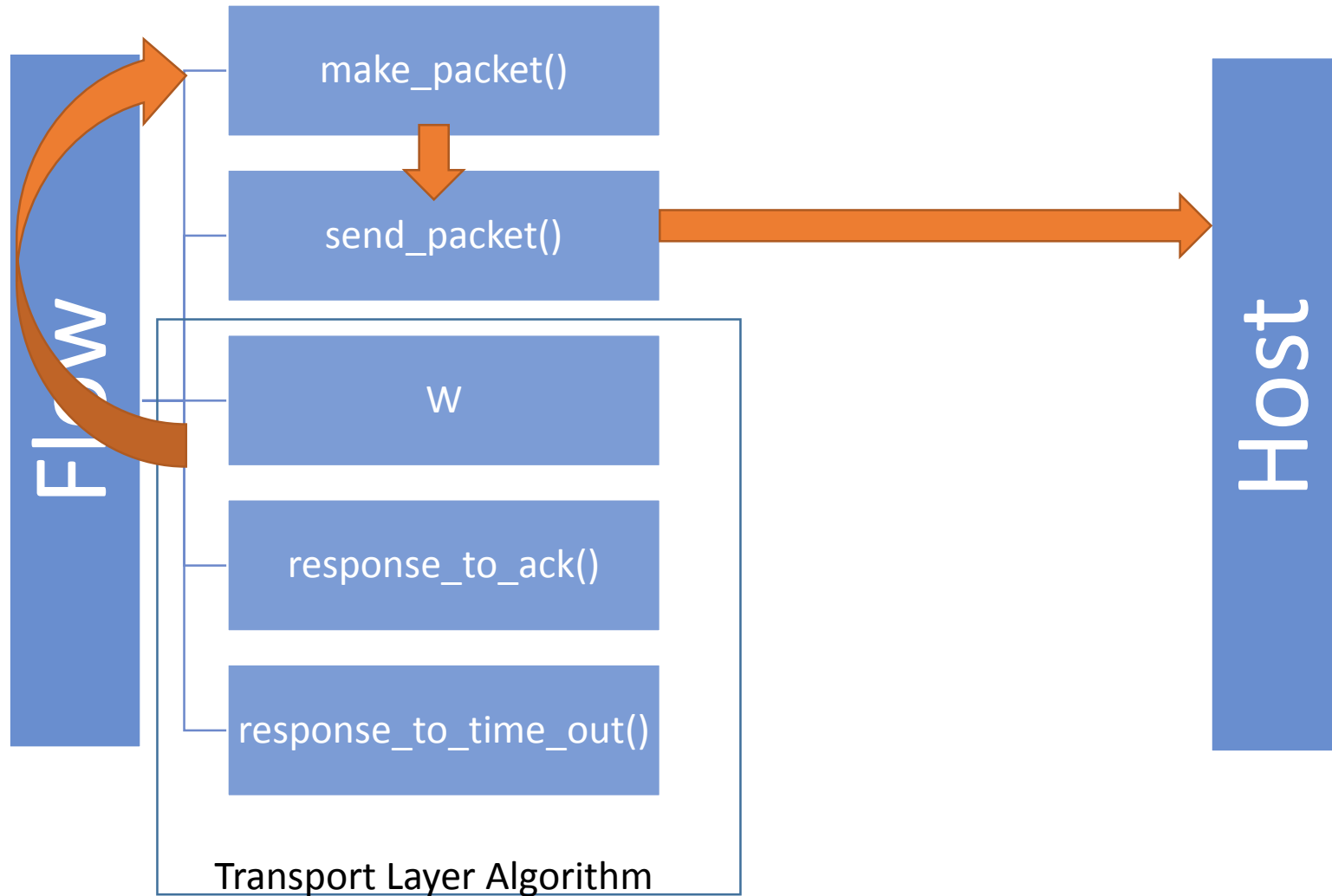
# Flow



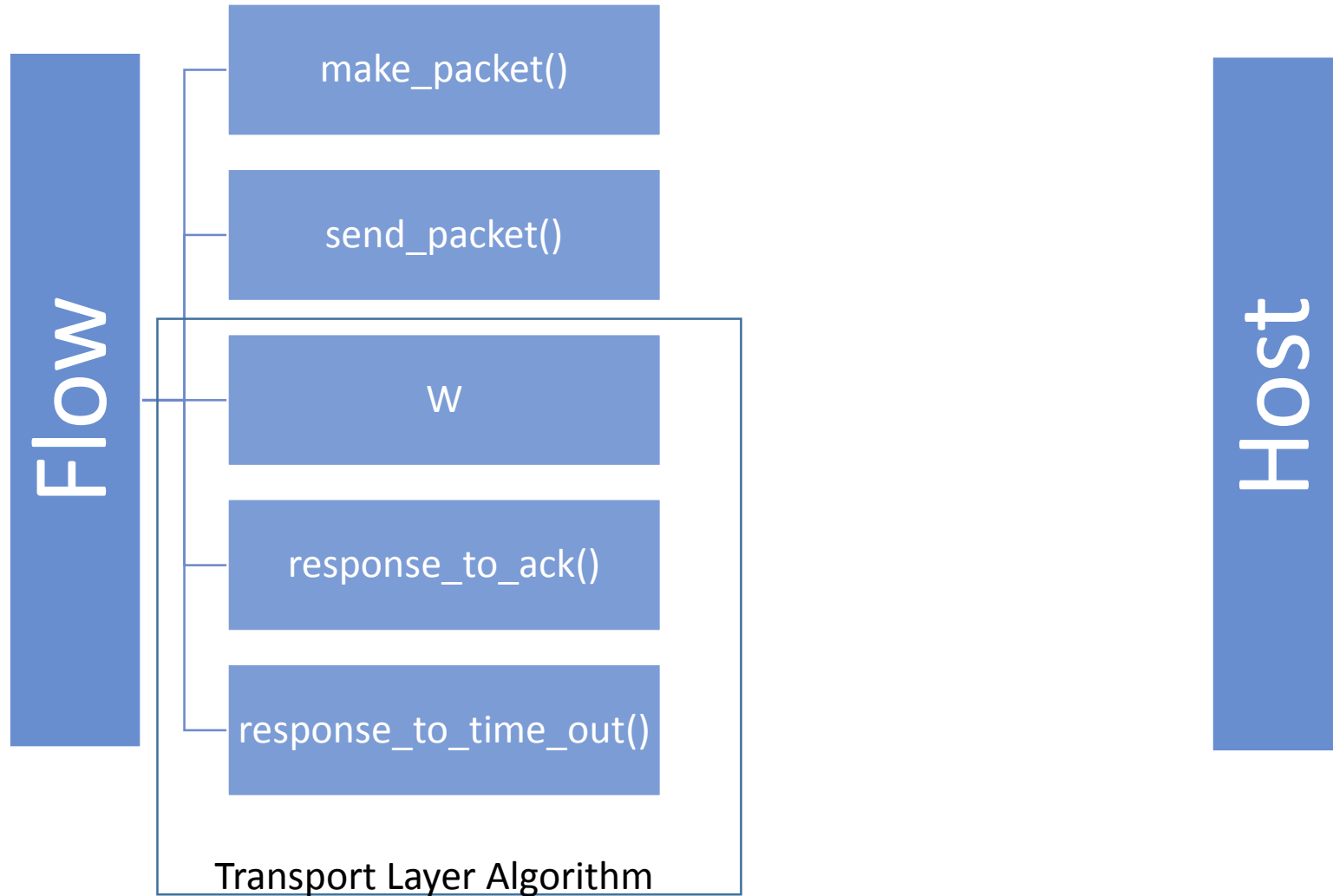
# Flow



# Flow

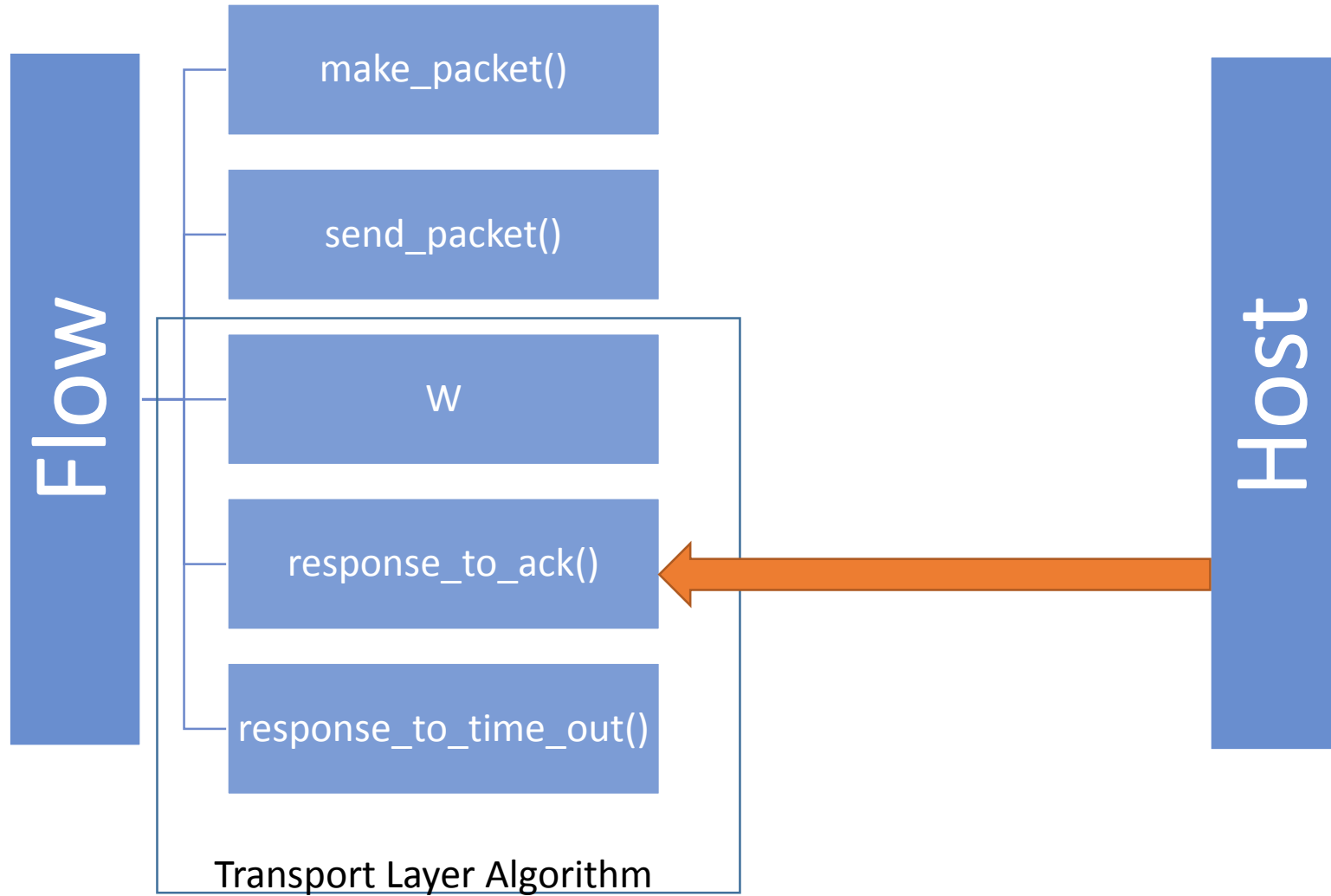


# Flow

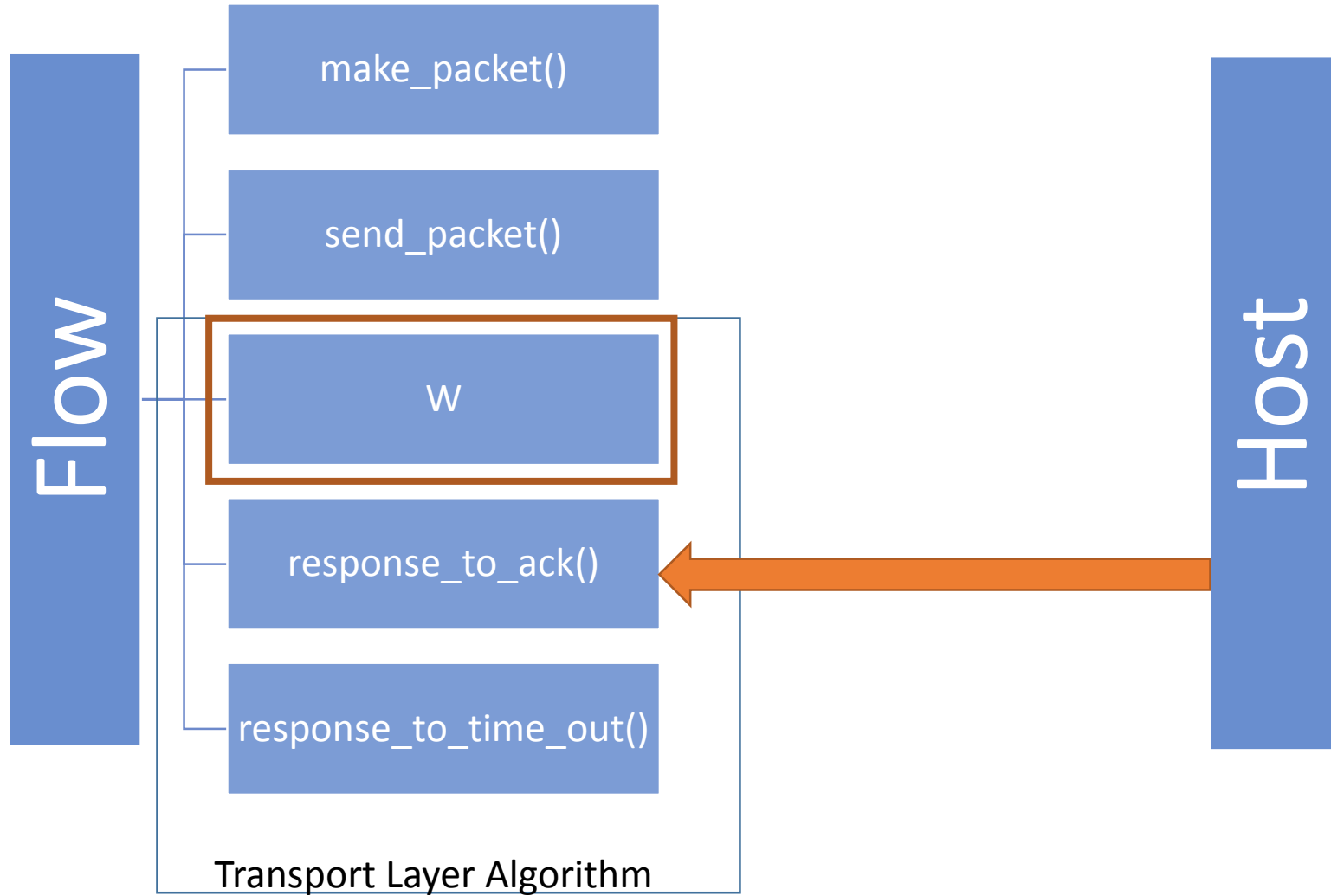




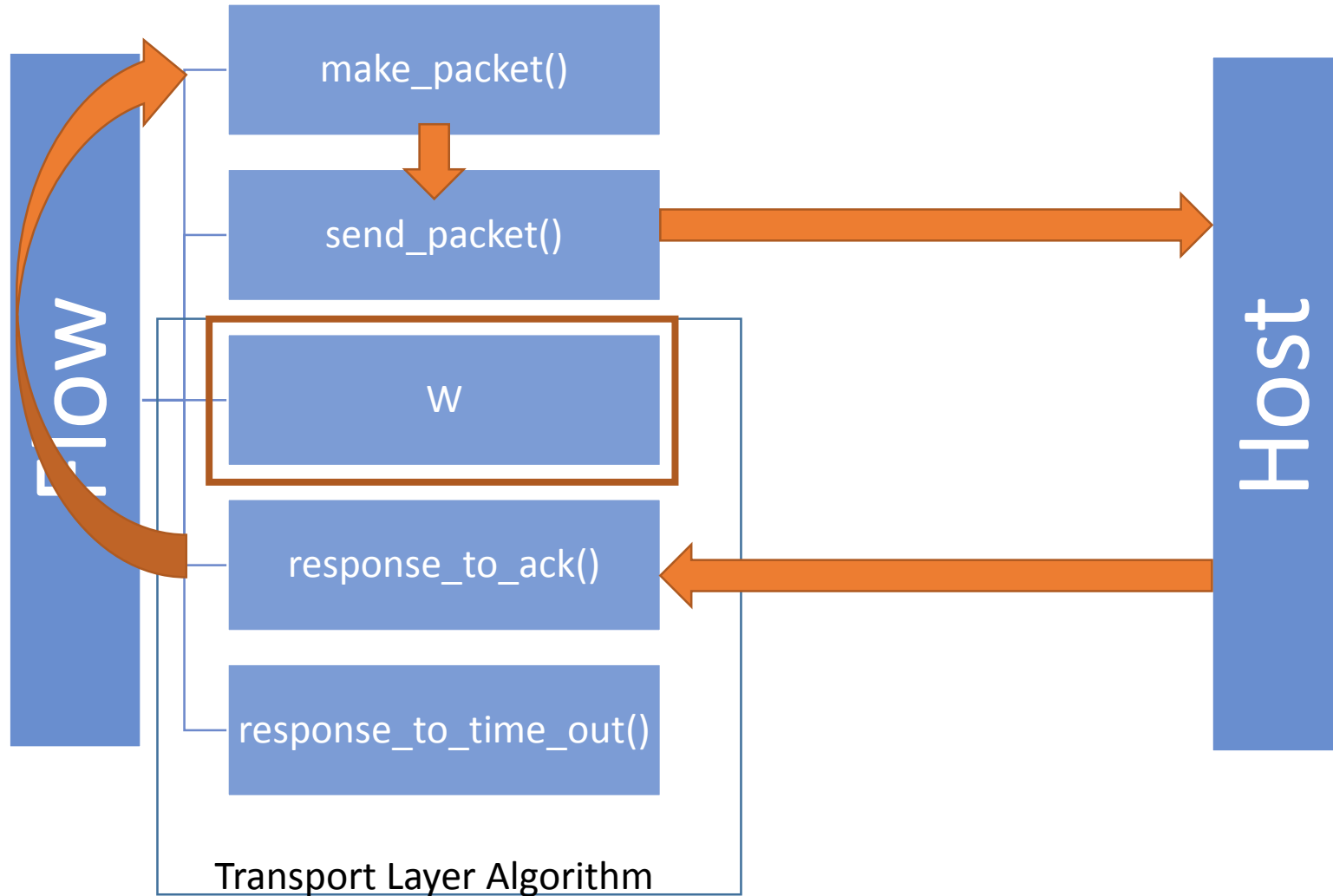
# Flow



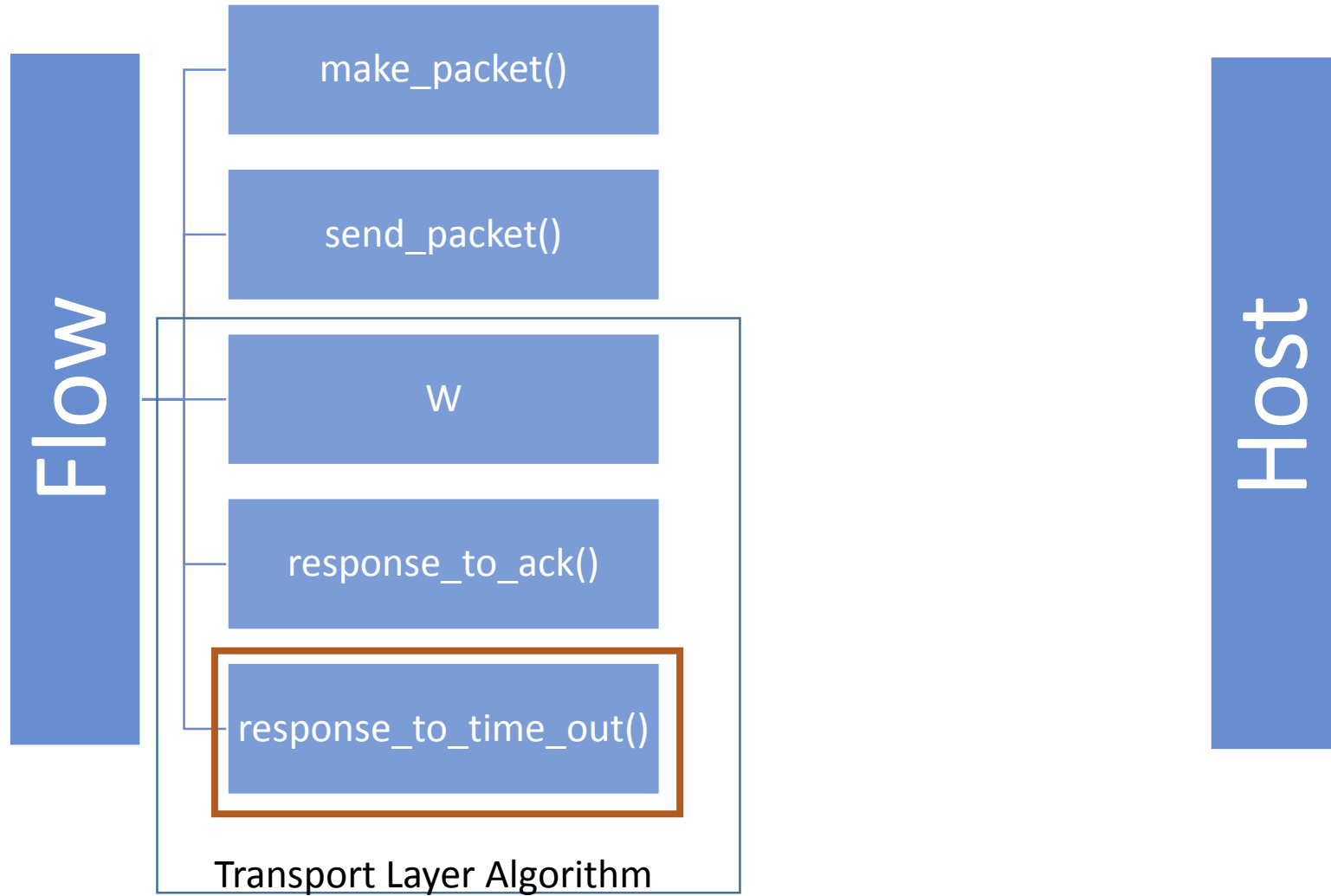
# Flow



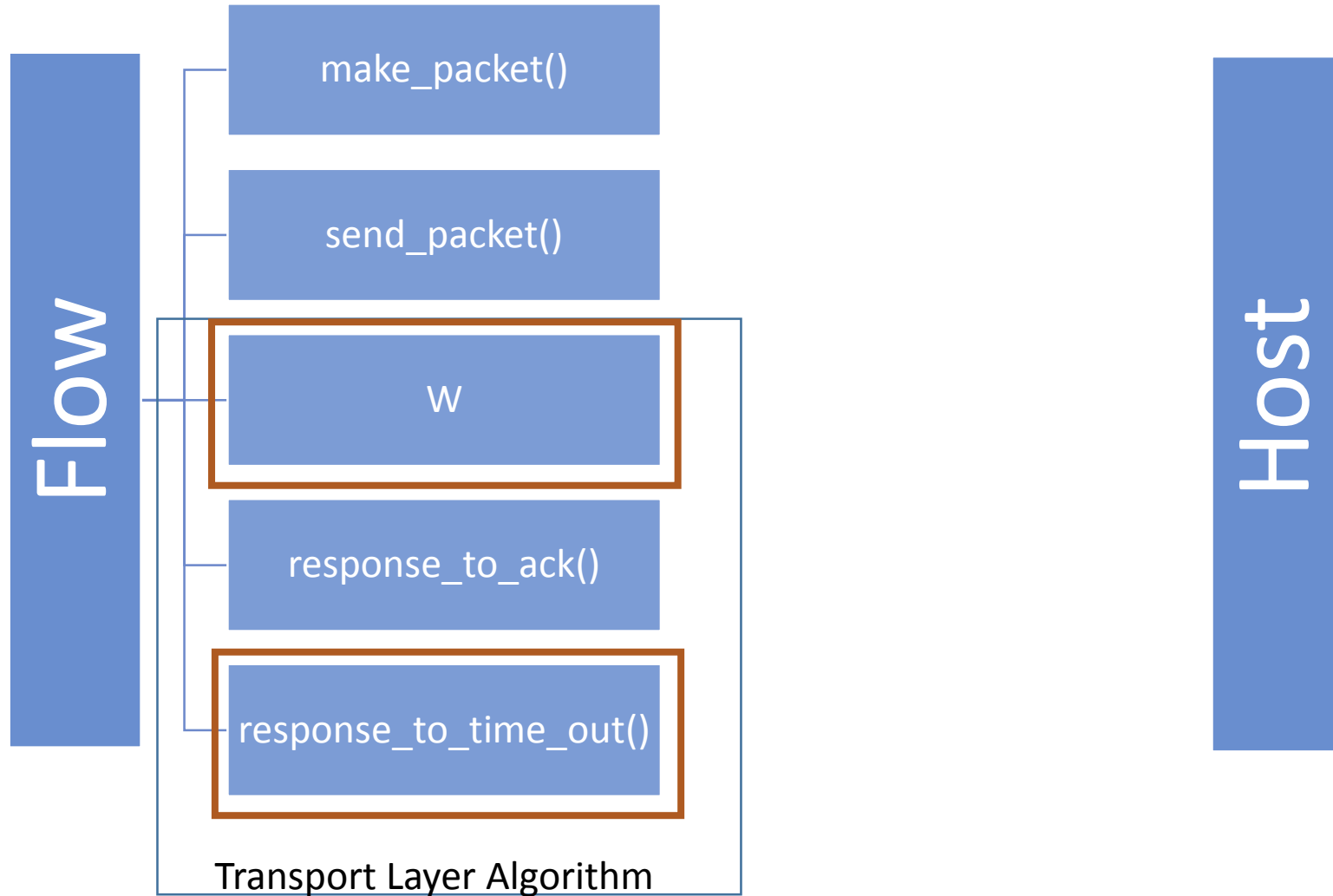
# Flow



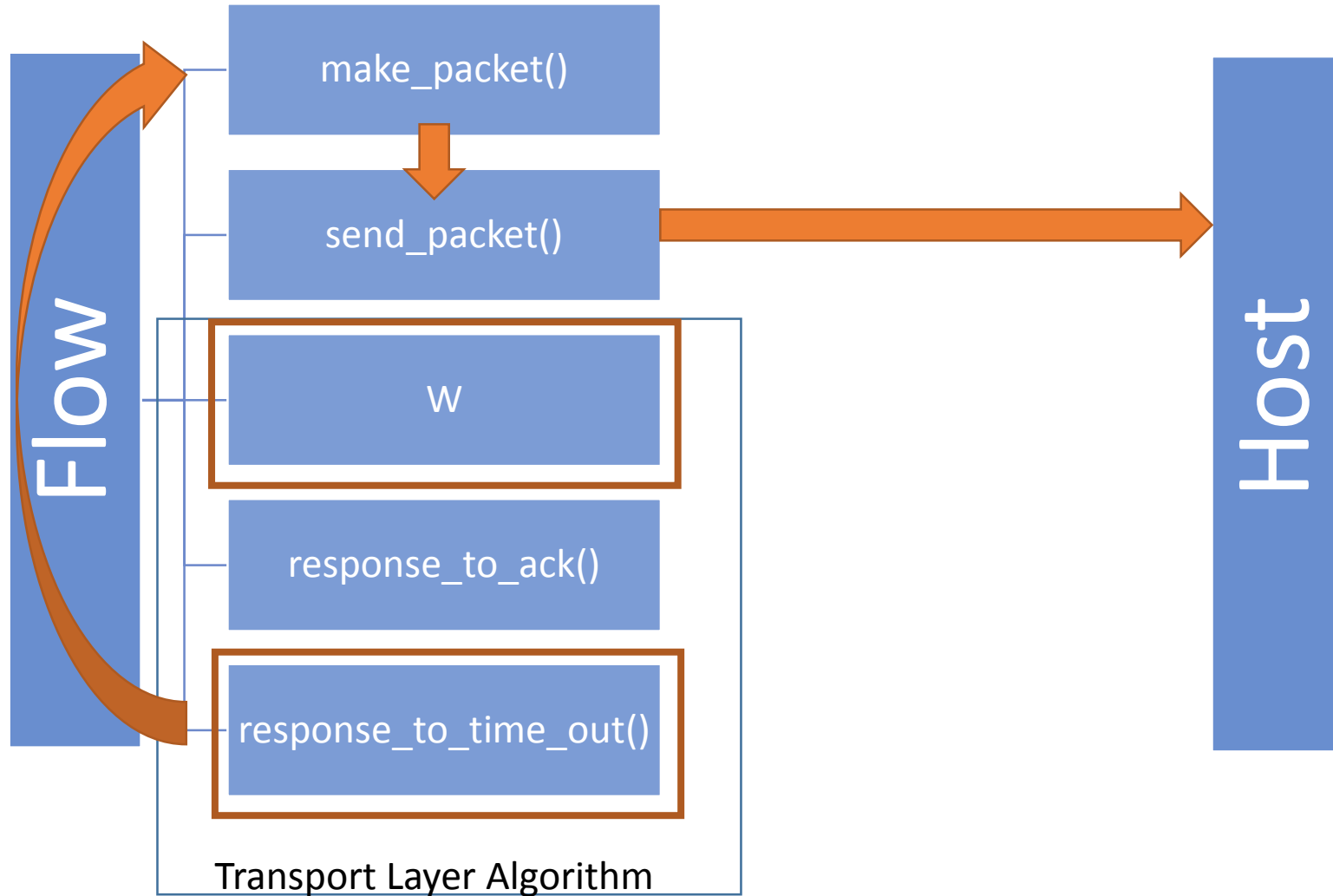
# Flow



# Flow



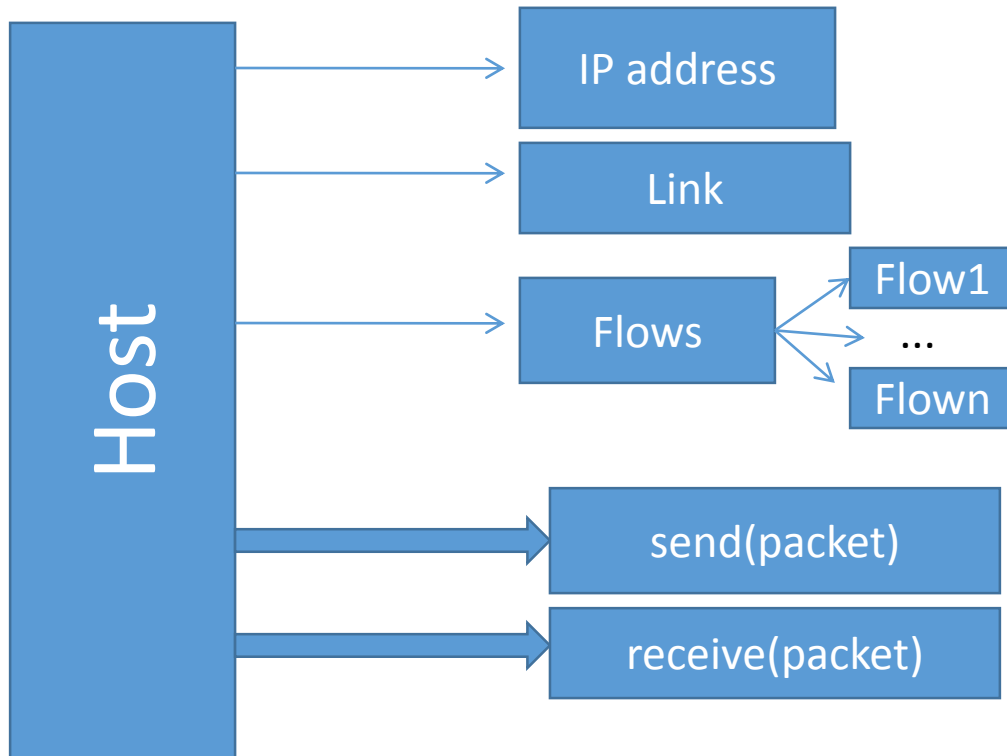
# Flow



# Host

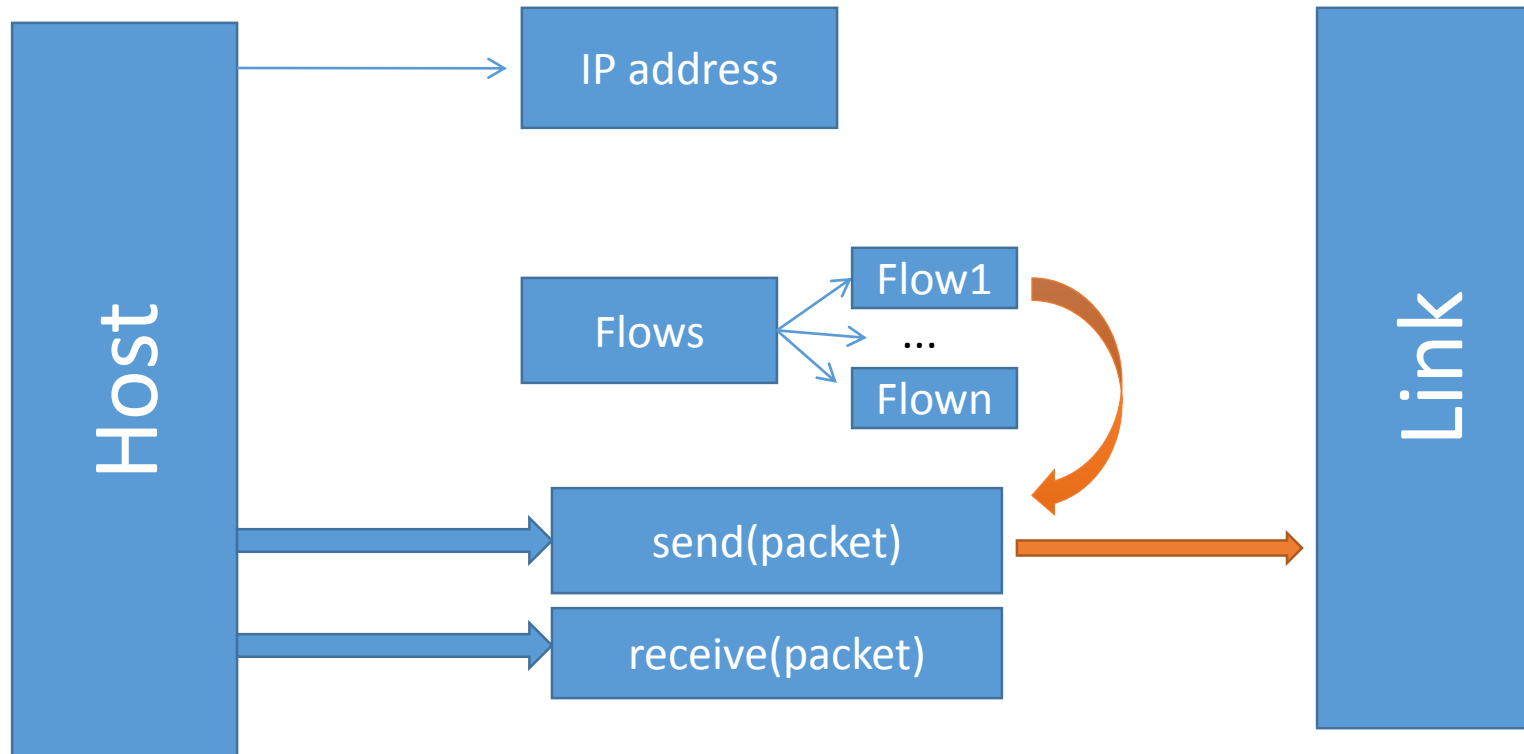
Hosts pass Packets from Flows to Links

# Host

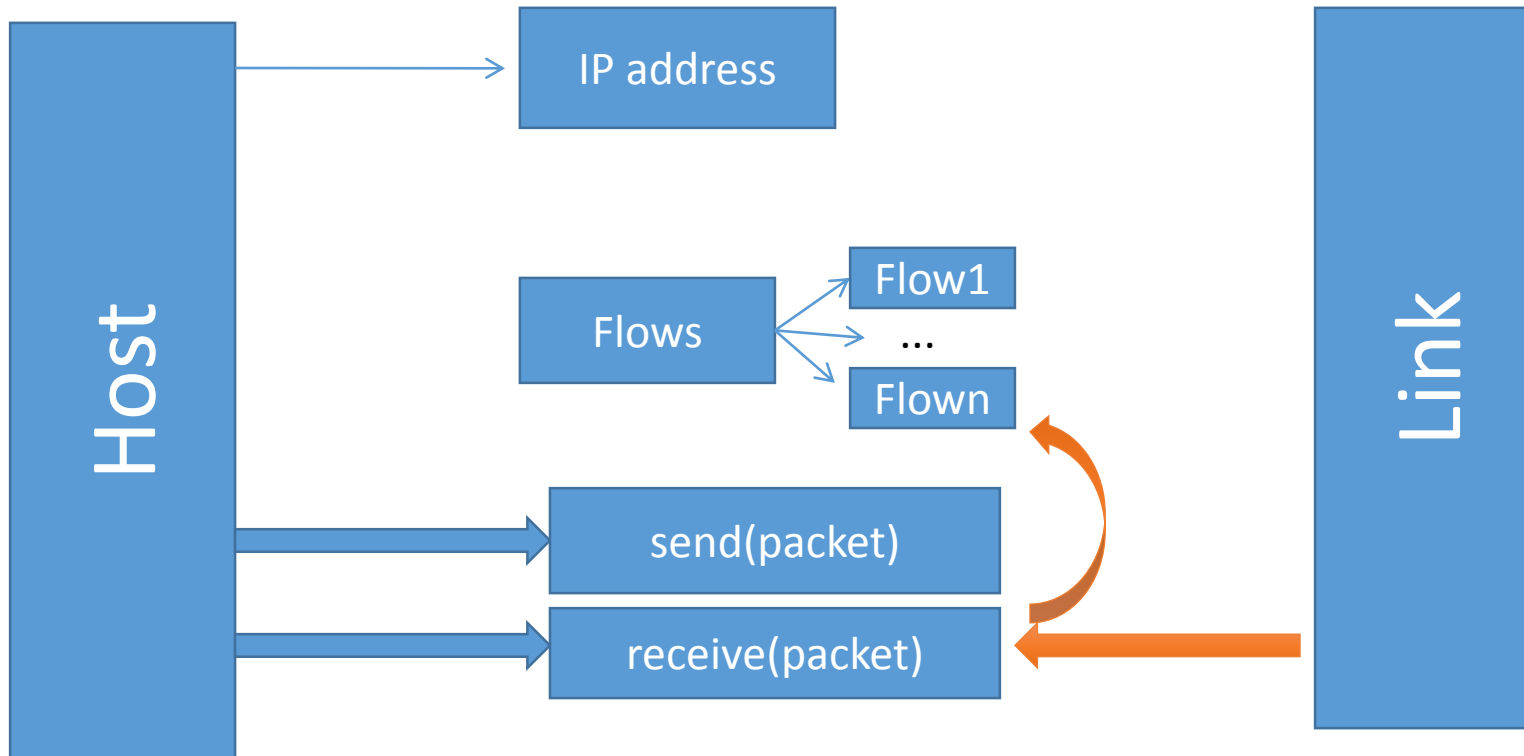




# Host



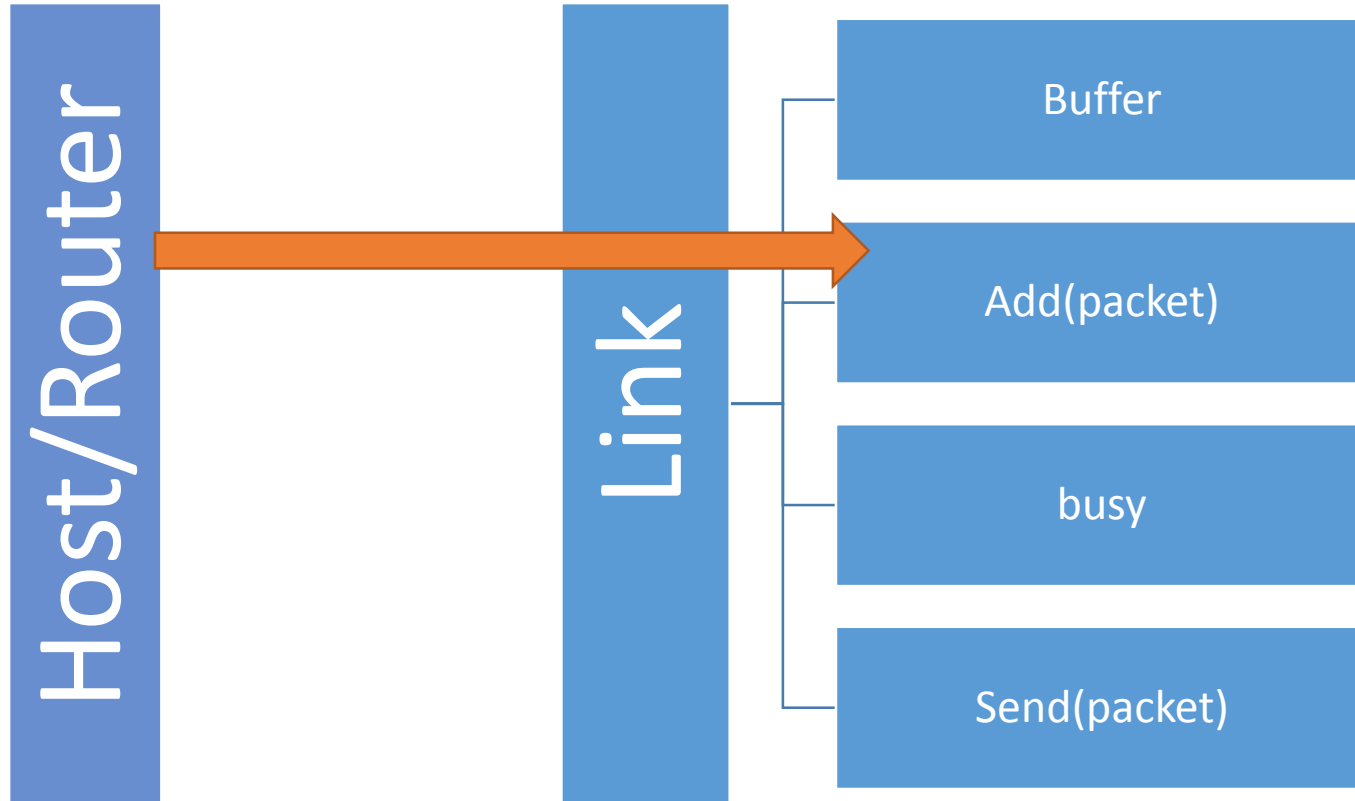
# Host



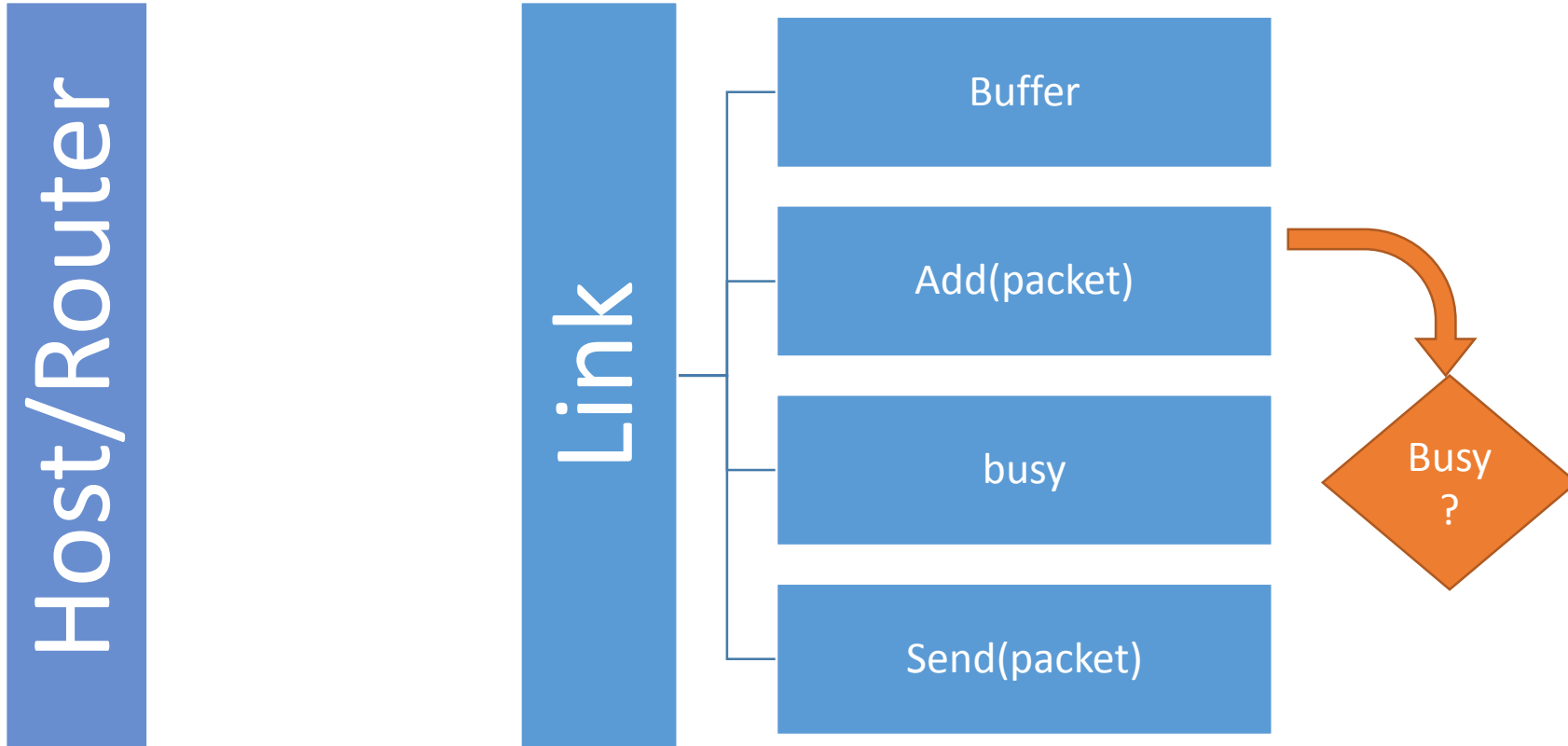
# Link

Links send Packets or store them in a Buffer

# Link

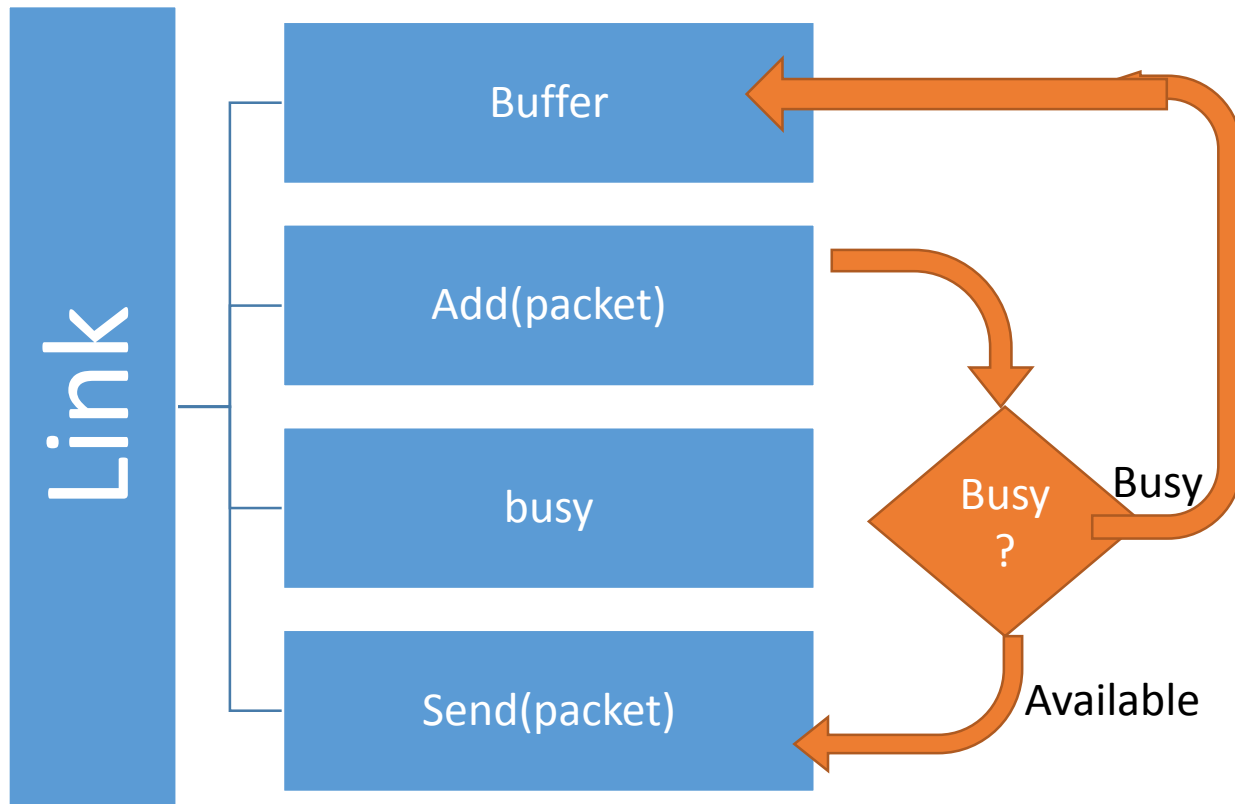


# Link



Host/Router

# Link



# Router

Update routing tables

Route Packets to the correct Link

# Router

## Update router table

- Send special packets to communicate with neighbor router, then decide what to update.
- Router table is updated using period timers.
- Implement Bellman-Ford algorithm first.

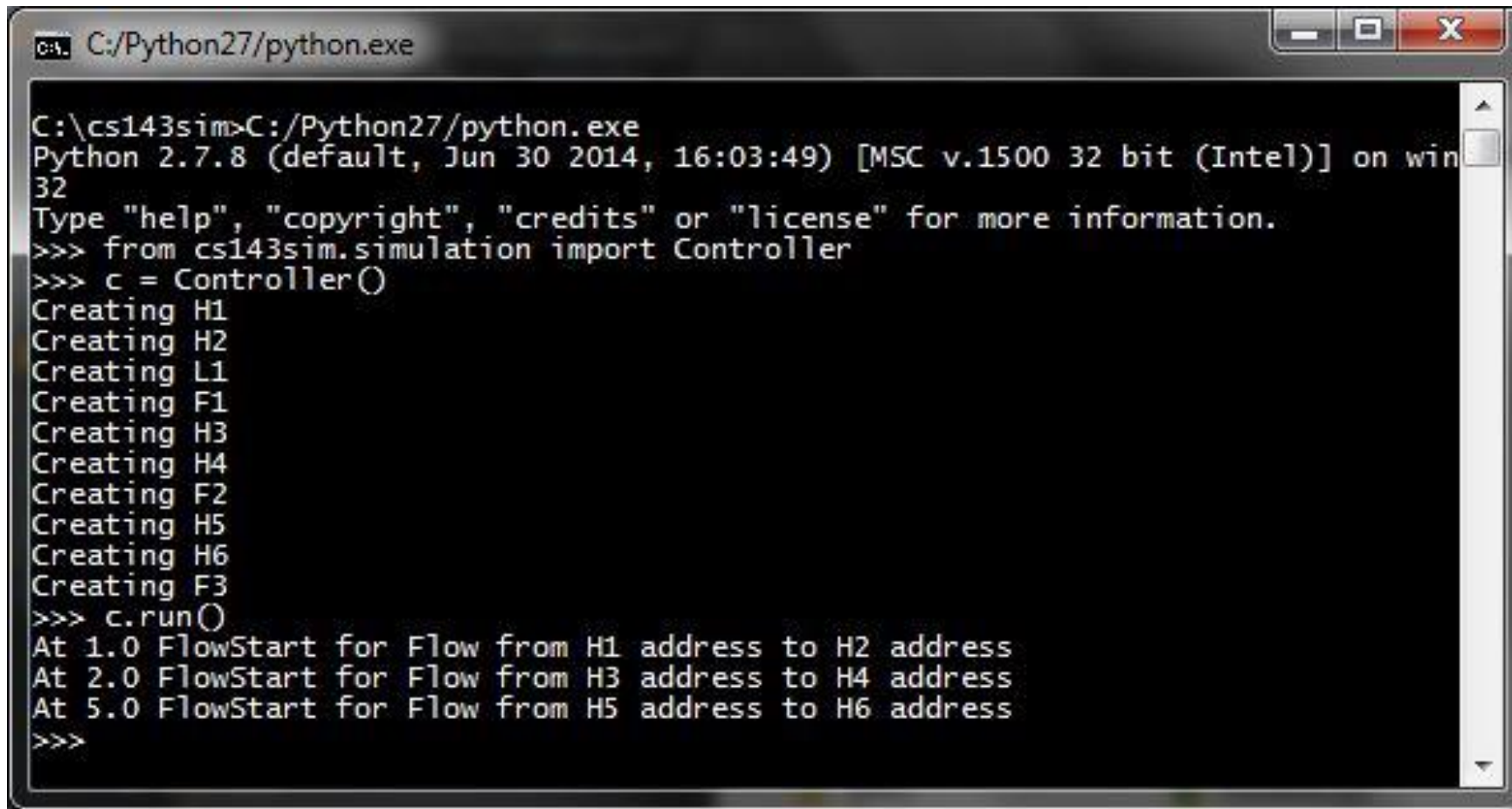


# Router

Send packet to the next link

- The routers decide which paths to forward current packages by looking up router table.
- Not involved in CCA, instantaneously forward arriving packets, the link buffer decides whether to drop it.

# Successful Build 0.1



```
C:\Python27/python.exe

C:\cs143sim>C:/Python27/python.exe
Python 2.7.8 (default, Jun 30 2014, 16:03:49) [MSC v.1500 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> from cs143sim.simulation import Controller
>>> c = Controller()
Creating H1
Creating H2
Creating L1
Creating F1
Creating H3
Creating H4
Creating F2
Creating H5
Creating H6
Creating F3
>>> c.run()
At 1.0 FlowStart for Flow from H1 address to H2 address
At 2.0 FlowStart for Flow from H3 address to H4 address
At 5.0 FlowStart for Flow from H5 address to H6 address
>>>
```


# Version Control

Simulator for operation of an abstract communication network (Caltech CS/EE 143, Fall 2014) — Edit

93 commits   2 branches   0 releases   5 contributors

branch: master   **cs143sim** / +

Additions from our pre-Jianchi group meeting: added event callbacks, ...

 **jvanbrug** authored 22 hours ago   latest commit 92be9b96bc

cs143sim	Additions from our pre-Jianchi group meeting: added event callbacks, ...	22 hours ago
docs	Added simulation Controller to docs	2 days ago
tests	Additions from our pre-Jianchi group meeting: added event callbacks, ...	22 hours ago
LICENSE	Initial commit	21 days ago
README.rst	Moved all documentation to ReadTheDocs.org! (goodnight)	10 days ago
requirements.txt	Mistake in requirements.txt	10 days ago
runtests.py	Added tests and runtests.py script	6 days ago
setup.py	Moved all documentation to ReadTheDocs.org! (minor fixes)	10 days ago
temp_routing_algorithms.py	Minor changes	9 days ago

**README.rst**

## cs143sim

Simulator for operation of an abstract communication network (Caltech CS/EE 143, Fall 2014)

Visit the documentation site [here!](#)

**Code**

Issues 0

Pull Requests 1

Wiki

Pulse

Graphs

Settings

HTTPS clone URL

<https://github.com/>

You can clone with HTTPS, SSH, or Subversion.

Clone in Desktop

Download ZIP

# Documentation

CS 143 Simulation

Search docs

Architecture

Packets

**Flows**

Hosts

Links

Routers

Monitoring

Input/Output

SimPy

Source Code

Course Materials

reStructuredText Help

Docs » Architecture » Flows

Edit on GitHub

## Flows

This is a summary of the roles, attributes, and design of flows.

### Abstract

In this case, flows are in charge of transport layer. There are three main part of transport layer:

- Error Control
- Congestion Control
- Flow Control

They are all features of TCP protocol. Different versions of TCP protocol employ different strategies.

### Error Control: How to retransmit

Three basic strategies

- Stop-and-wait
- Go Back N
- Selective Acknowledgments

### Congestion Control: What's window size; Avoid saturating routers

# Documentation

CS 143 Simulation

Search docs

Architecture

Source Code

Actors

Events

Simulation

Course Materials

reStructuredText Help

[Docs](#) » [Source Code](#) » [Actors](#)

[Edit on GitHub](#)

## Actors

This module contains all actor definitions.

<code>Buffer</code> (capacity)	Representation of a data storage container
<code>Flow</code> (source, destination, amount)	Representation of a connection between access points
<code>Host</code> (address)	Representation of an access point
<code>Link</code> (source, destination, delay, rate, ...)	Representation of a physical link between access points or routers
<code>Packet</code> (source, destination, number, ...)	Representation of a quantum of information
<code>Router</code> (address)	Representation of a data router

## Buffer

```
class cs143sim.actors.Buffer(capacity) \[source\]
```

Representation of a data storage container


Buffers store data to be linked while `Link` is busy sending data.

Parameters: **capacity** (*int*) – maximum number of bits that can be stored

Variables:

- capacity** (*int*) – maximum number of bits that can be stored
- packets** (*list*) – `Packets` currently in storage

# Documentation

 CS 143 Simulation

Architecture

Source Code

Actors

Events

Simulation

Course Materials

reStructuredText Help

[Docs](#) » [Source Code](#) » [Events](#)

[Edit on GitHub](#)

## Events

This module contains all event definitions.

<code>FlowStart</code> ( <code>env</code> , <code>delay</code> , <code>flow</code> )	A <code>Flow</code> begins generating packets
<code>LinkAvailable</code> ( <code>env</code> , <code>delay</code> , <code>link</code> )	A <code>Router</code> finishes sending a
<code>PacketReceipt</code> ( <code>env</code> , <code>delay</code> , <code>receiver</code> , <code>packet</code> )	A <code>Host</code> or a <code>Router</code>
<code>UpdateRoutingTable</code>	

## Flow Start

```
class cs143sim.events.FlowStart(env, delay, flow) \[source\]
```

A `Flow` begins generating packets

- Parameters:
- `env` - SimPy simulation `Environment`
  - `delay` (*float*) - time until `Flow` starts
  - `flow` - `Flow` that starts

## Link Available

```
class cs143sim.events.LinkAvailable(env, delay, link) \[source\]
```

A `Router` finishes sending a `Packet` on `Link`

# Responsibilities

Hongjian Lan

Hosts, Packet Routing

Yamei Ou

Routing Tables

Samuel Richerd

I/O, Graphs, Packets

Jan Van Bruggen

Monitoring, Team  
Manager

Junlin Zhang

Flows, Congestion  
Control

# Schedule

<u>Time</u>	<u>Progress</u>	<u>Obstacles</u>
Week 6:	Test Case 0	Packet Sending
Week 7:	All Test Cases, no CCA	Routing Tables
Week 8:	All Test Cases	Congestion Control
Week 9:	Output Graphs	Monitoring Hooks
Week 10:	Bells & Whistles	Algorithm Toggles