Increasing Asynchronous Message Throughput

Jason Roberts

@robertsjason | dontcodetired.com

Overview

Blocking in actors

Slow down message throughput

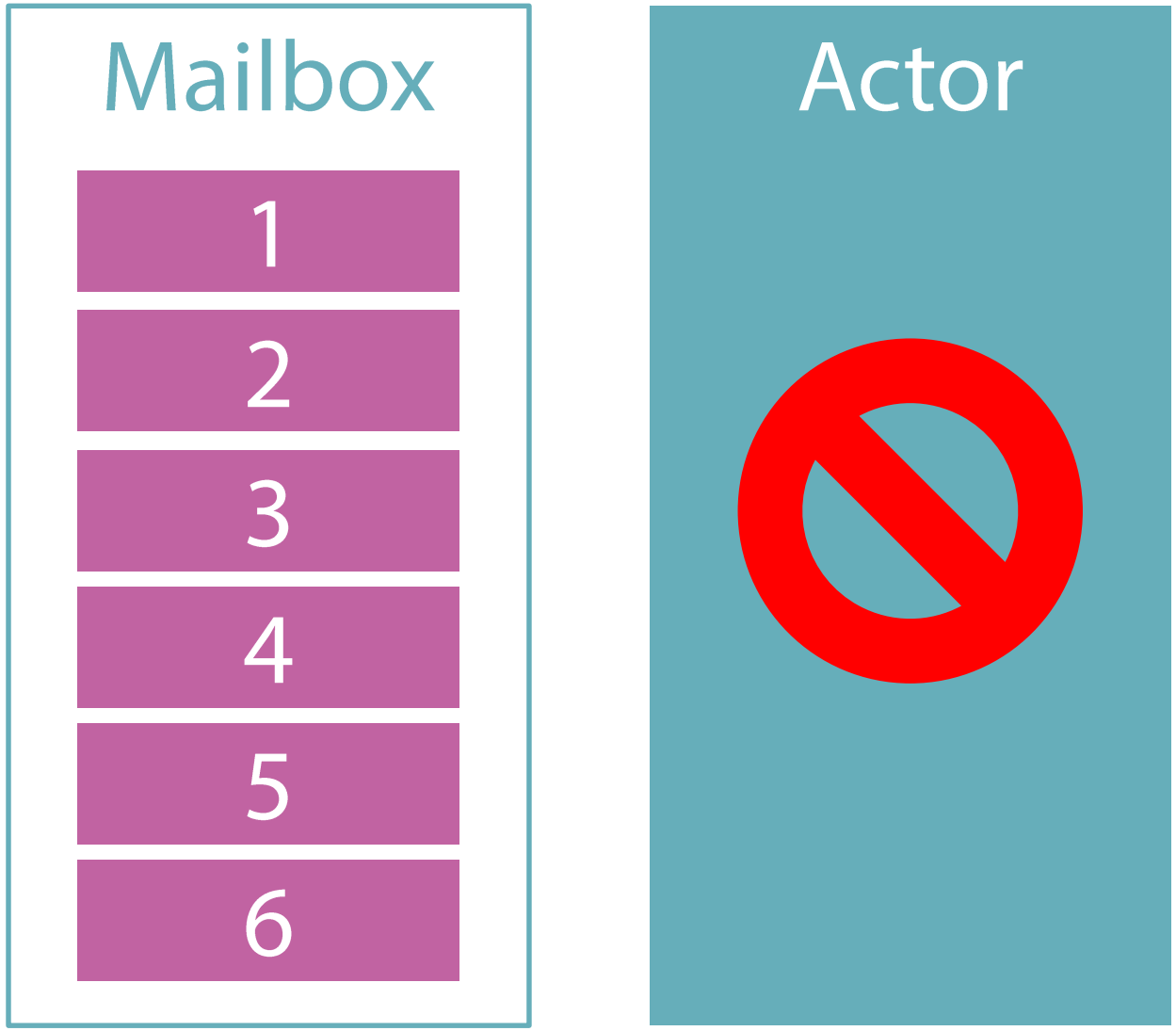
Potential for deadlocked actors

Using async/await in actor code

PipeTo()

Executing blocking code in actors can reduce the throughput of messages and in severe cases, where an operation never completes, actor deadlock may occur

Blocked Actors



Blocking Demo

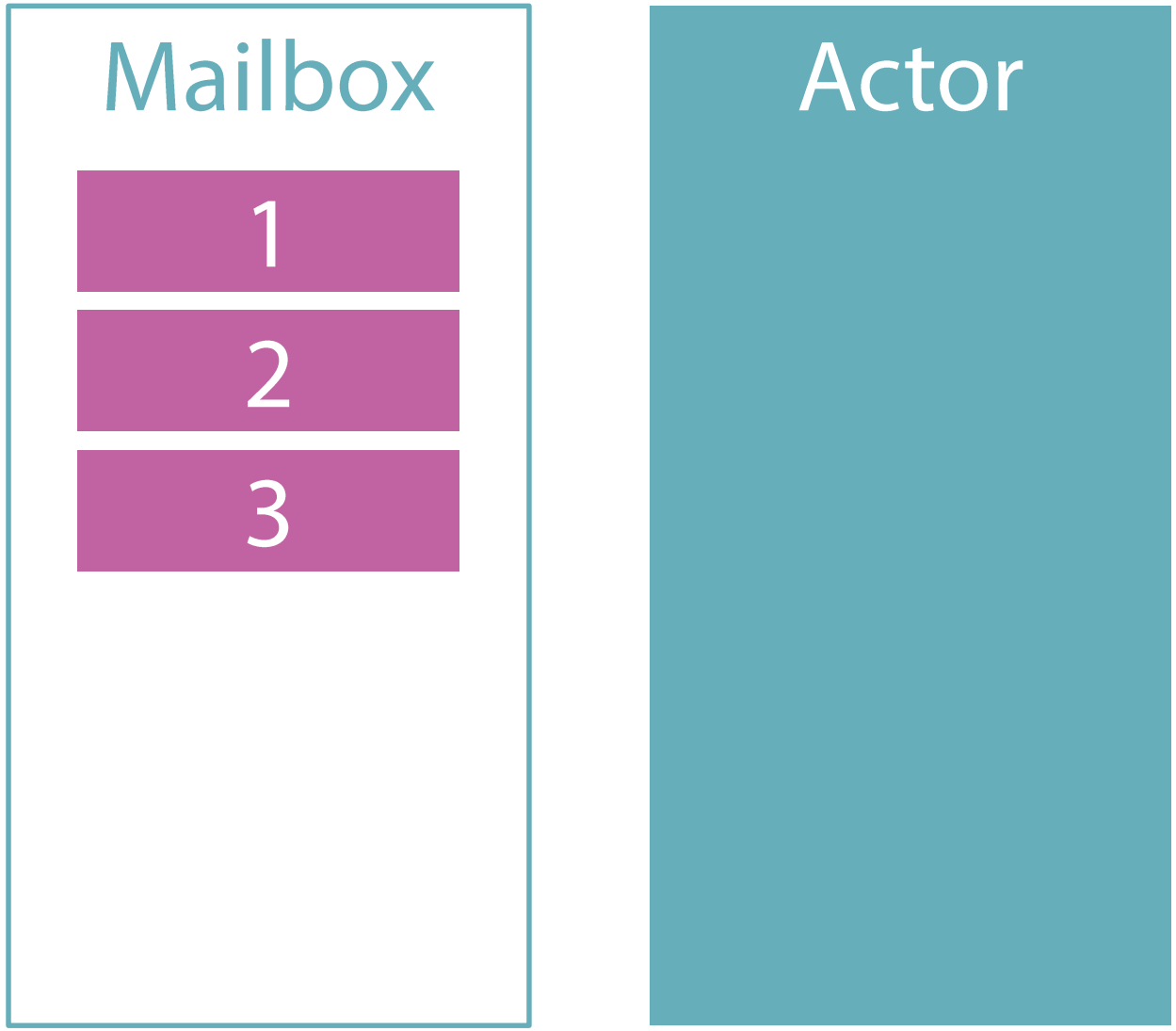
Deadlocking an Actor With Code That Never Ends

Using Async/Await in Actors

PipeTo()

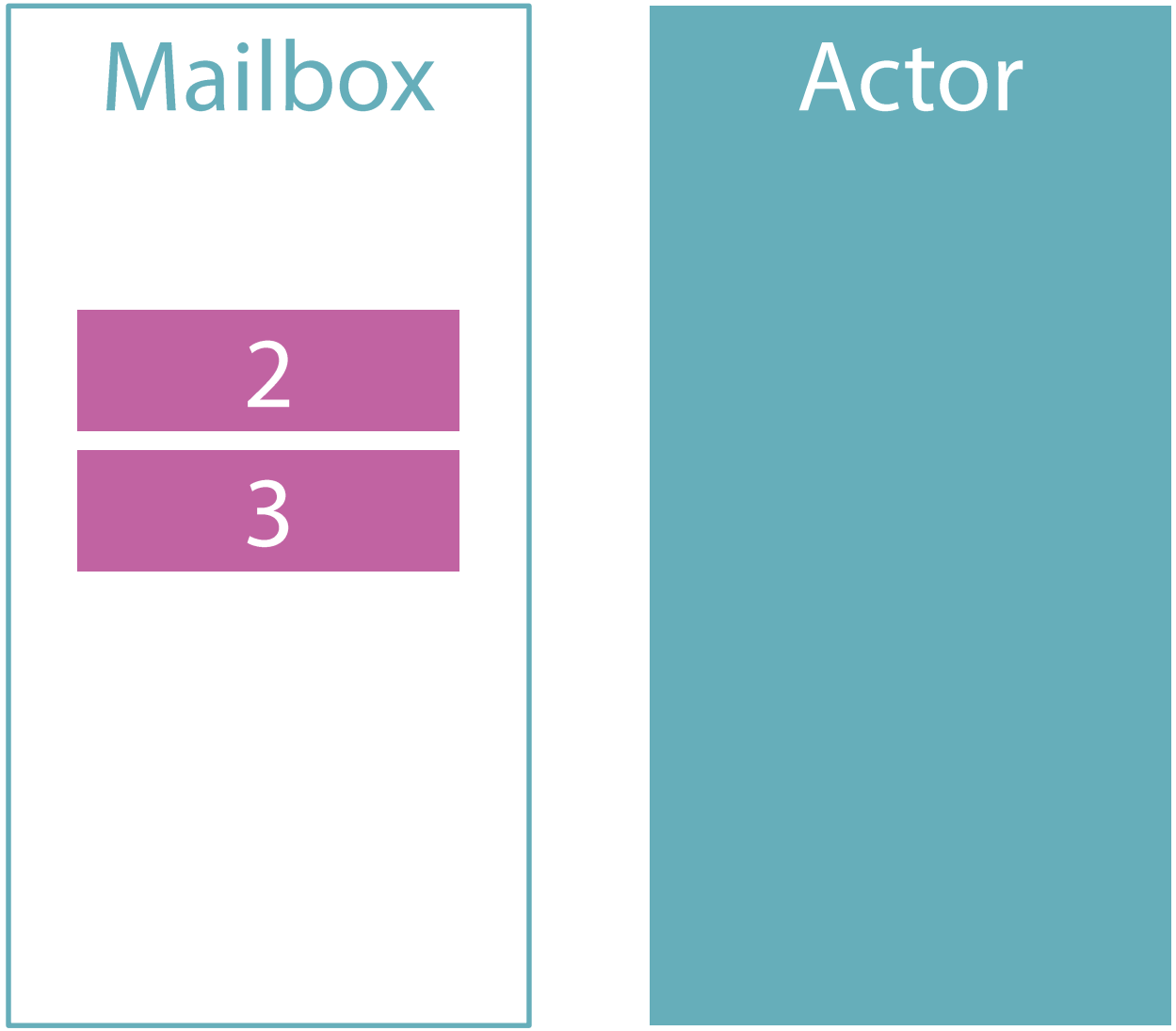
PipeTo() is an Akka.NET extension method that enables an actor to make multiple simultaneous asynchronous calls and send (pipe) the results to an actor in the form of a message.

PipeTo()



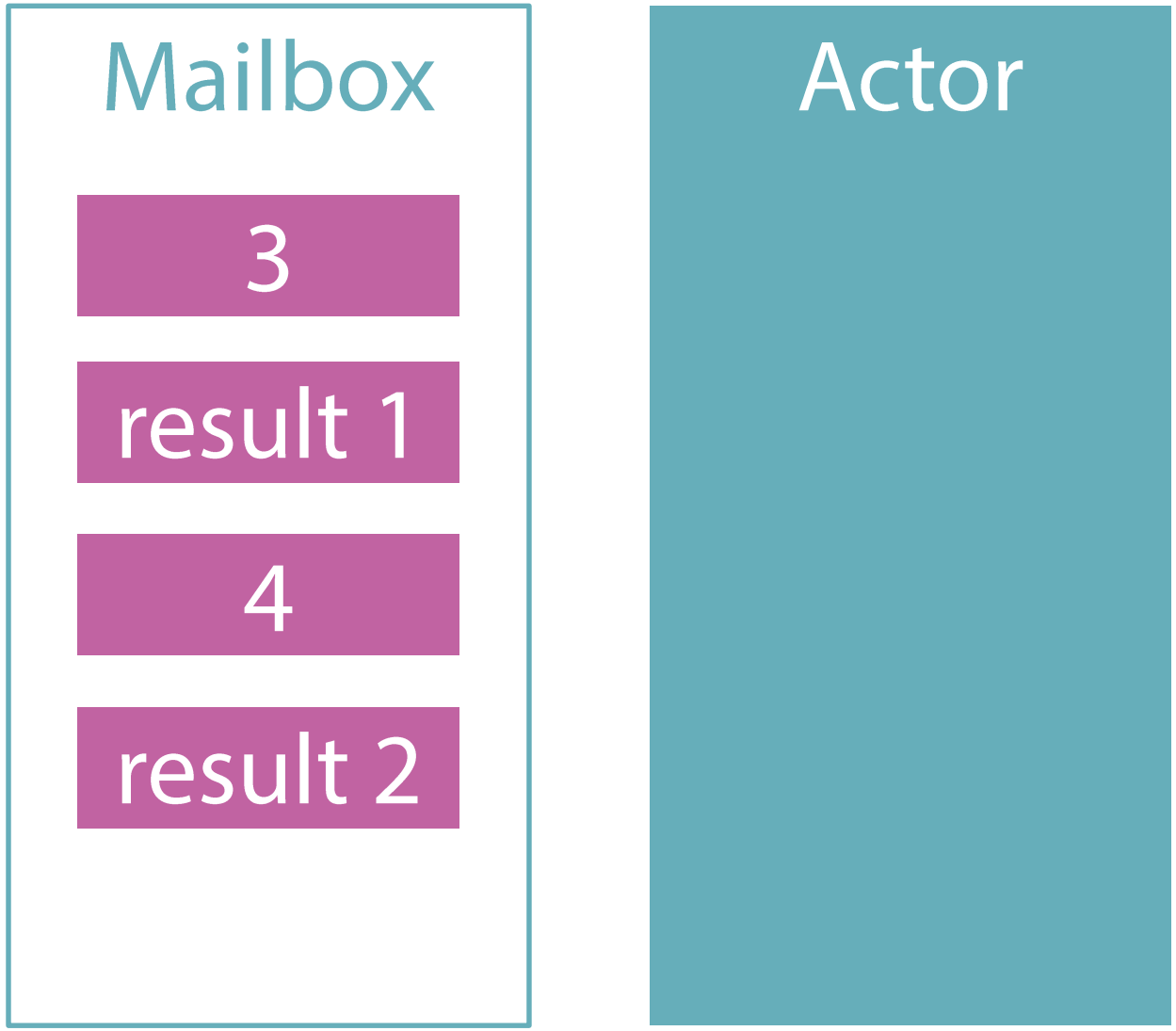
Start asynchronous operation

PipeTo()

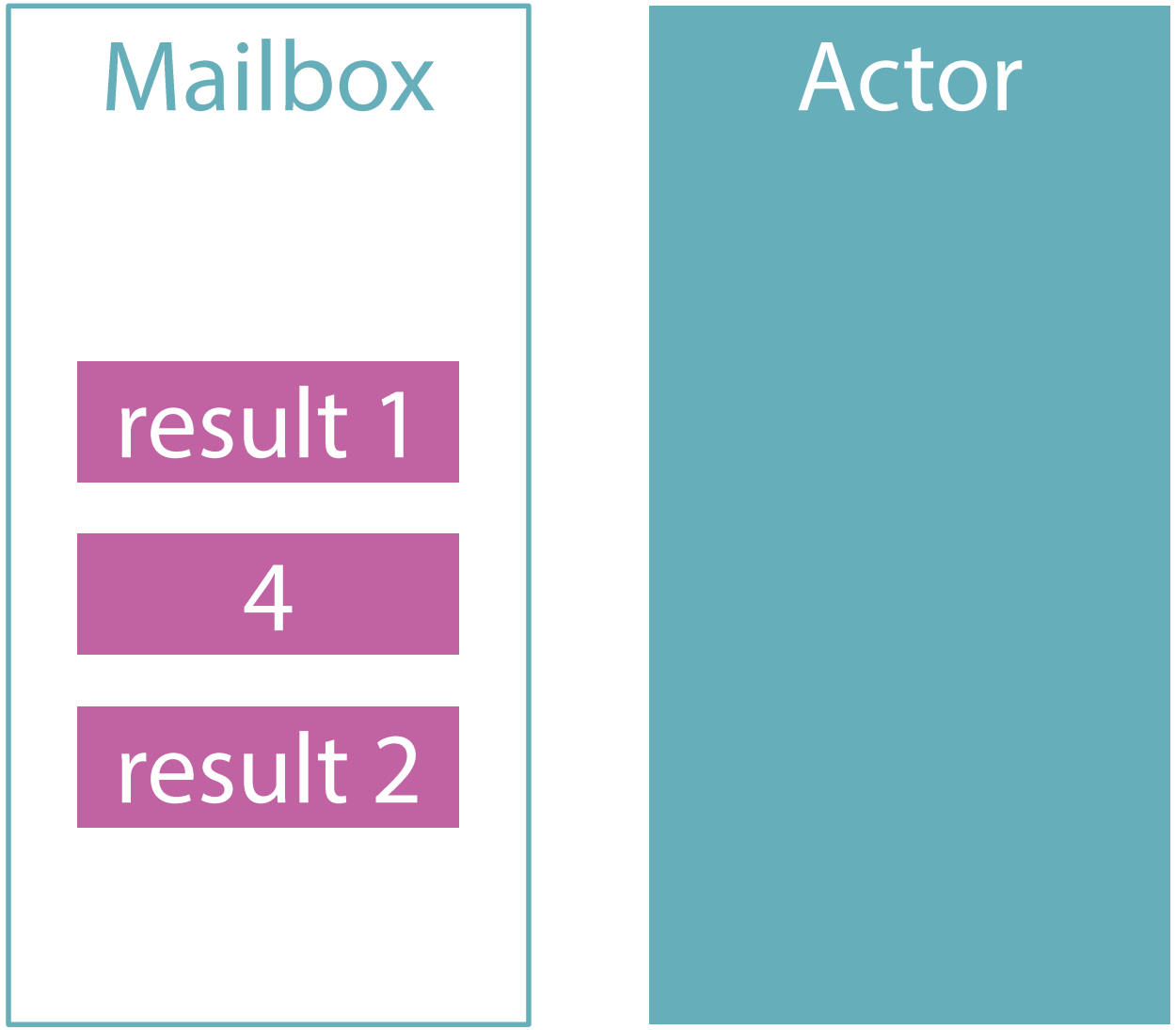


Start asynchronous operation

PipeTo()



PipeTo()



Result of asynchronous operation

The advantage of using the PipeTo pattern over async/await is that asynchronous results can be piped to other actors, and not just the originating actor. PipeTo is also more conceptually in-line with the Actor Model due to asynchronous results being treated like any other message.

Refactoring To Use PipeTo

PipeTo()

~~Async~~ void HandleSendPayment

Receive<PaymentReceipt>

Summary

Blocking in actors

Slow down message throughput

Deadlocked actors

Using async/await in actor code

Capture the sender for later use

.PipeTo(Self, Sender)