Section 10: Networking

Frank Austin Nothaft

July 29, 2015

Contents

1 Warmup

1.1 Layering

In class last week, we introduced the concept of a network "stack." The stack we covered last week was a four layer stack. Can you draw out the different layers of this stack, and describe what that layer does? Can you think of common applications/technologies/protocols that map to a specific level of the stack?

1.2 Why layers?

As you may have noticed, the multilayered networking stack is a stark departure from the monolithic structure of an operating system kernel[1]. What is the advantage of breaking networking technologies into different stacks?

[1]Many academic projects have tried to implement non-monolithic OS kernels; this body of work is largely known as "microkernels". This work has seen a fair bit of adoption in the embedded world, but it has not been seen great industrial reception for enterprise workloads.

2 Problems

2.1 Lossy Networks and the End-to-end Principle

Typically, if we want a highly reliable connection between machines, we will use a layer 3 (transport) protocol that provides reliability (e.g., TCP). Why do we not make network reliability a function of a lower layer, such as IP?

In TCP, if we notice that a packet was dropped, what do we do? Why do we take this action? What does this assume/what is this trying to exploit?

Here, TCP assumes that packet losses due to error (e.g., network is interrupted, packet is corrupted) are fairly infrequent. What could happen if our physical layer was naturally lossy? For example, wireless links have packet loss rates that are orders of magnitude higher than wired links like Ethernet. How could you solve this problem? Does your solution agree with or contradict the end-to-end principle?

2.2 RPC

Remote Procedure Call (RPC) is an API for calling functions that reside on another computer, or in another process. When we are making an RPC call, how is the call executed?

RPC is used in a variety of contexts, with varying levels of success. What are the general reasons that RPC can be difficult to use?

What are a few examples of places where RPC is easy to use successfully?