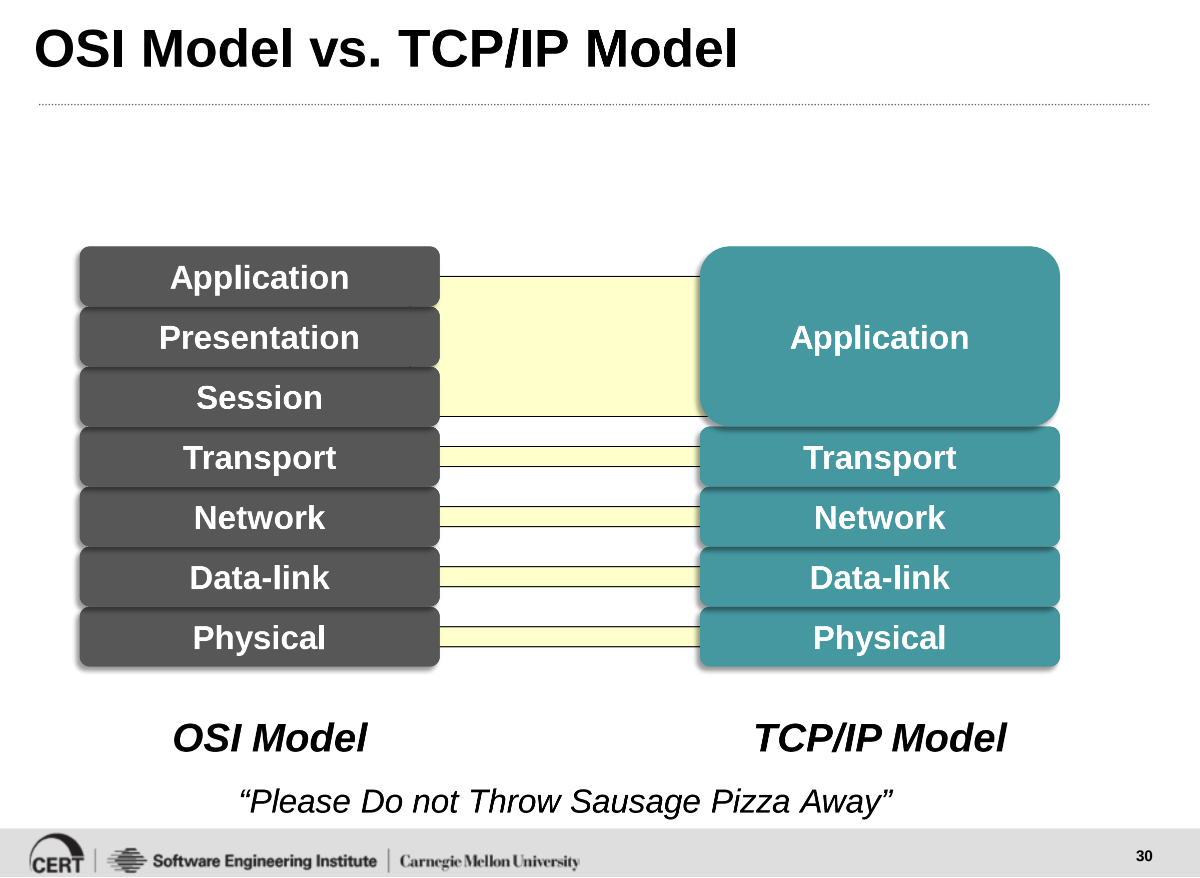
TCP/IP Model Overview

Table of Contents

OSI Model vs. TCP/IP Model



OSI Model vs. TCP/IP Model

Application

Presentation

Session

Transport

Network

Data-link

Physical

Application

Transport

Network

Data-link

Physical

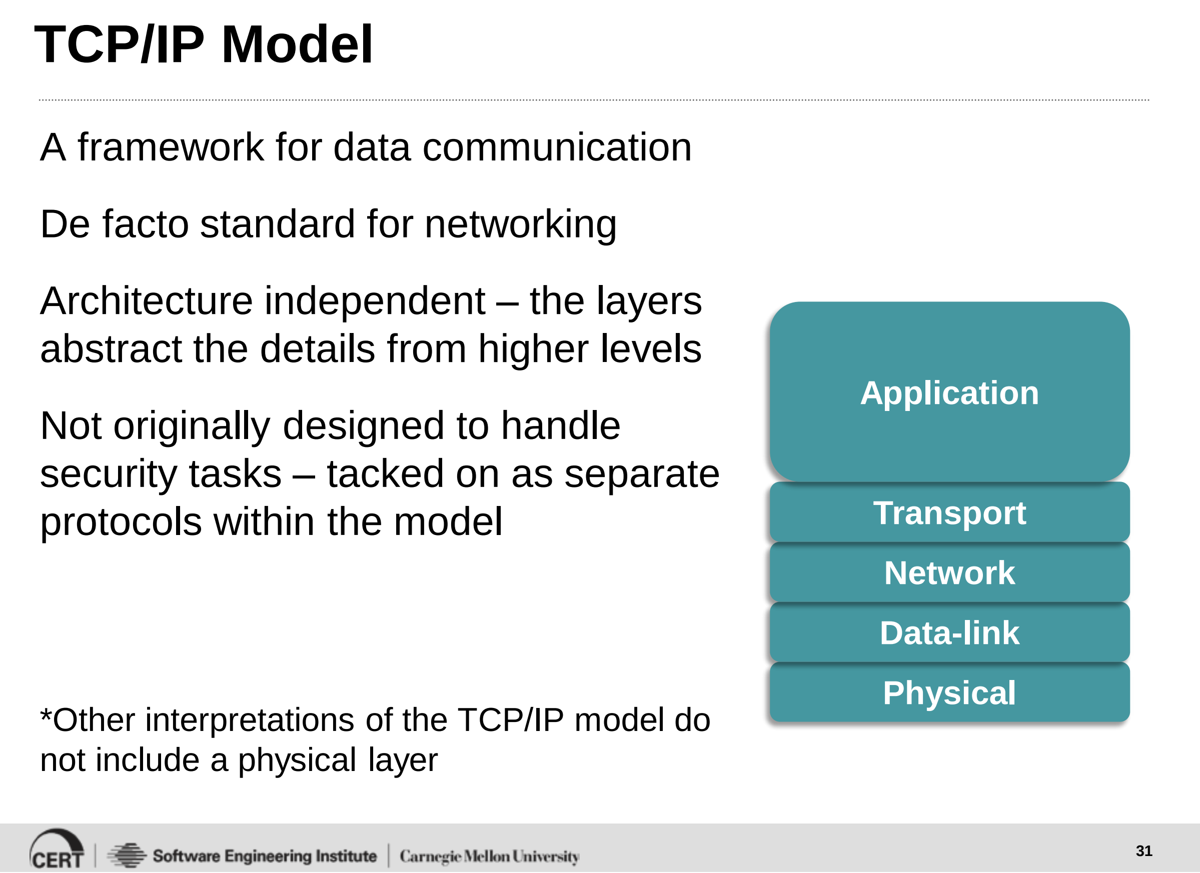
OSI Model

TCP/IP Model

“Please Do not Throw Sausage Pizza Away”

**030 Okay so now let's map to the other model that's out there.

TCP/IP Model



TCP/IP Model

A framework for data communication

De facto standard for networking

Architecture independent – the layers abstract the details from higher levels

Not originally designed to handle security tasks – tacked on as separate protocols within the model

*Other interpretations of the TCP/IP model do not include a physical layer

Application

Transport

Network

Data-link

Physical

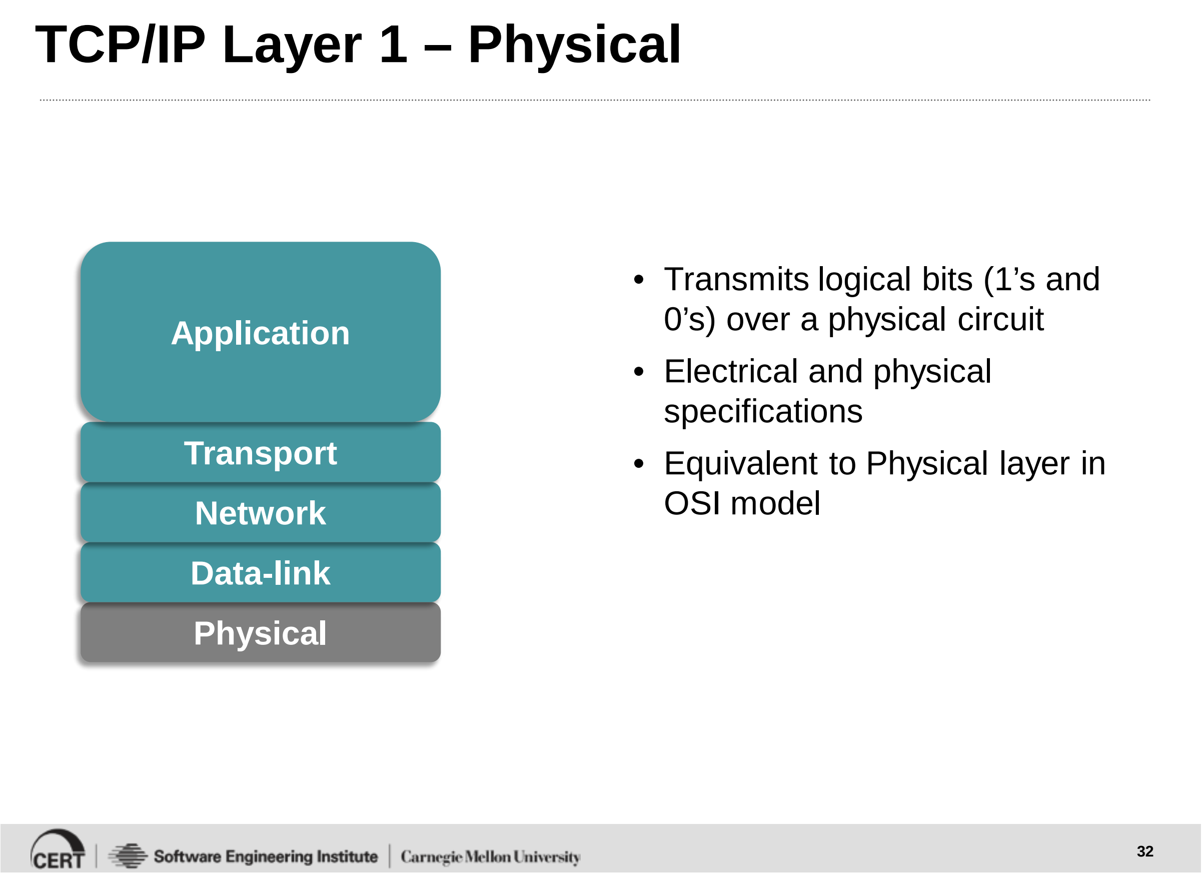
CERT | Software Engineering Institute | Carnegie Mellon University 31

**031 So let's talk about the whole model first before we actually dig any deeper.

It is another framework for data Communication.

Now it was never designed at the very beginning to handle the security. It just-- it wasn't- that wasn't an issue as it was birthed. But now that is an issue. And so we've revved the protocol from version 4 to version 6. There is no version 5 of TCP/IP.

TCP/IP Layer 1 – Physical



TCP/IP Layer 1 – Physical

Application

Transport

Network

Data-link

Physical

- Transmits logical bits (1’s and 0’s) over a physical circuit
- Electrical and physical specifications
- Equivalent to Physical layer in OSI model

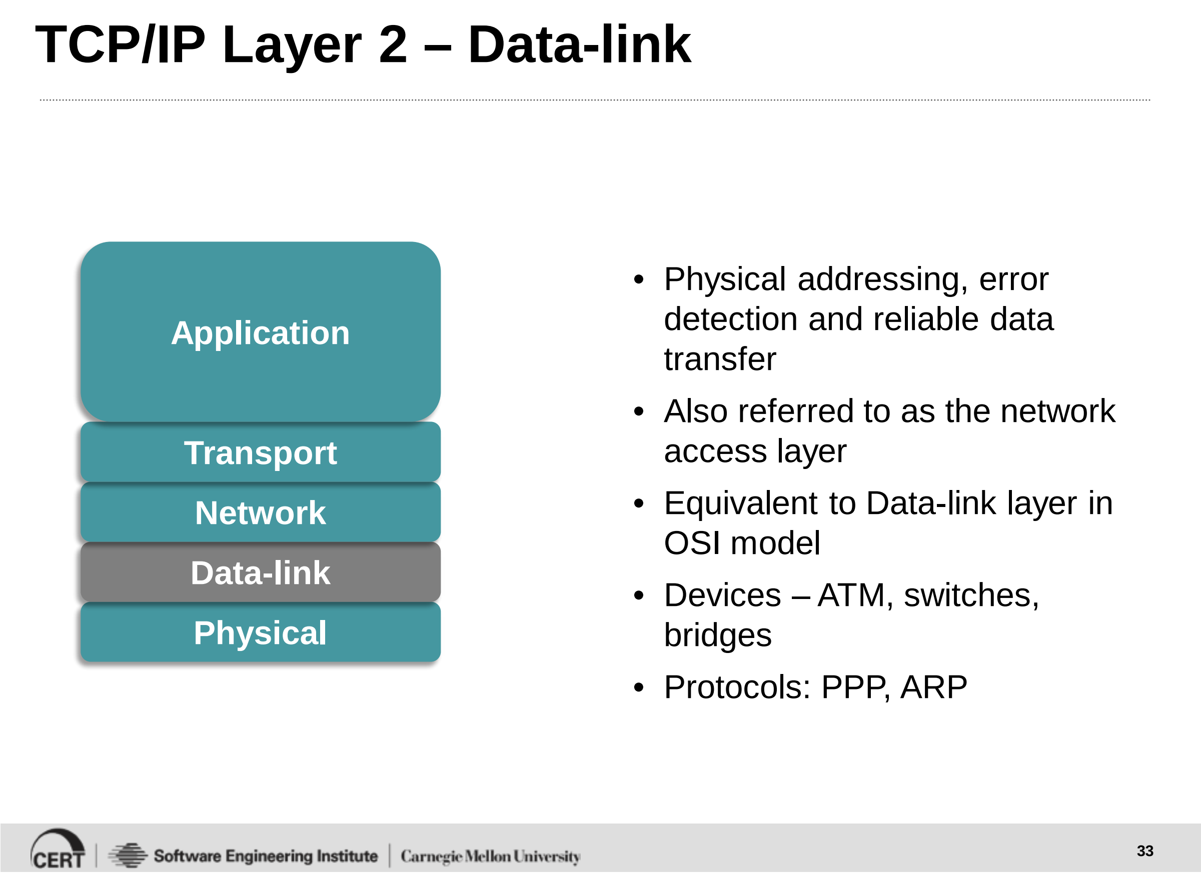
CERT | Software Engineering Institute | Carnegie Mellon University

32

**032 So let's look at it.

Physical layer; same thing: Transmitting the bits; electrical signals; Physical specification. So identical to the OSI.

TCP/IP Layer 2 – Data-link

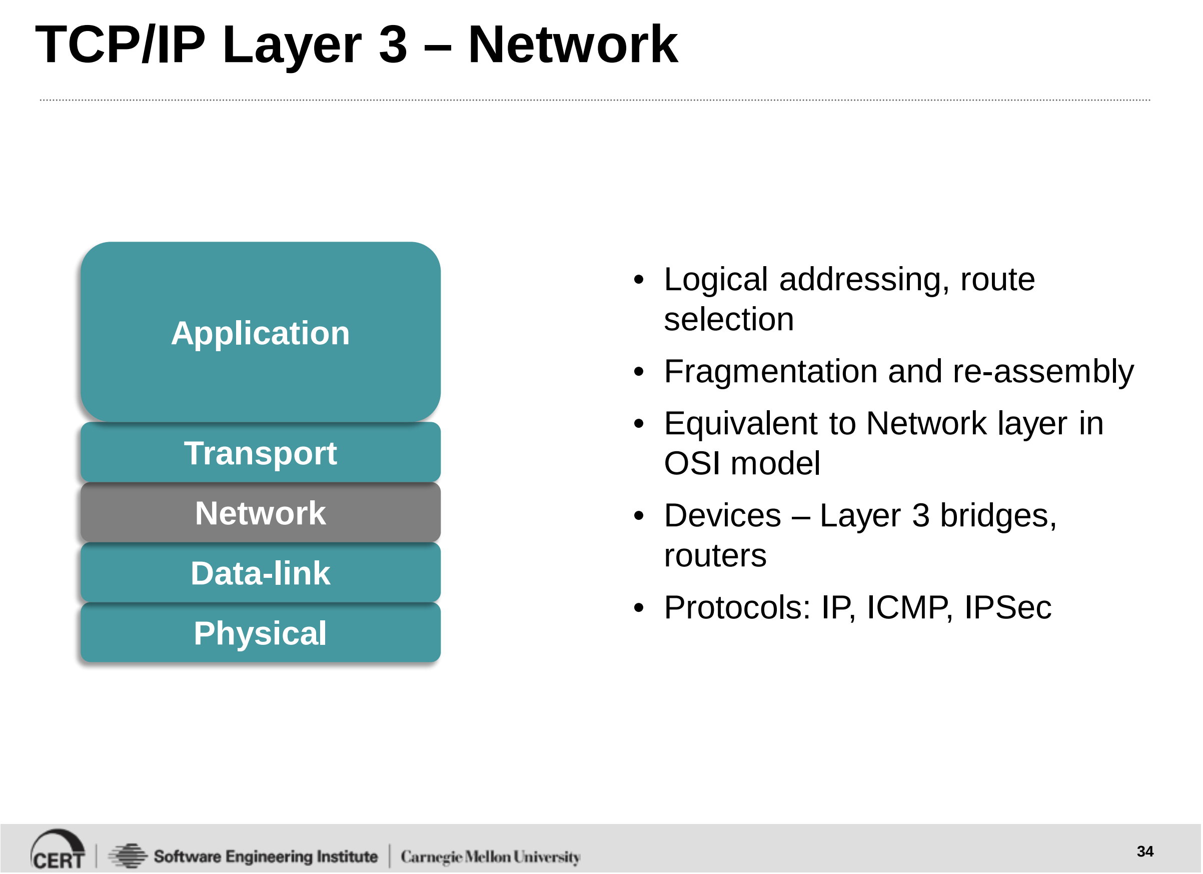


TCP/IP Layer 2 – Data-link

- Physical addressing, error detection and reliable data transfer
- Also referred to as the network access layer
- Equivalent to Data-link layer in OSI model
- Devices – ATM, switches, bridges
- Protocols: PPP, ARP

**033 Data-link layer; same thing. Now we're talking about having Physical addressing that was burned into the network card itself, that MAC address. We're using that over again. And we also pay attention to ARP; and that allows us to map the Data-link to the Network layer.

TCP/IP Layer 3 – Network



TCP/IP Layer 3 – Network

Application

Transport

Network

Data-link

Physical

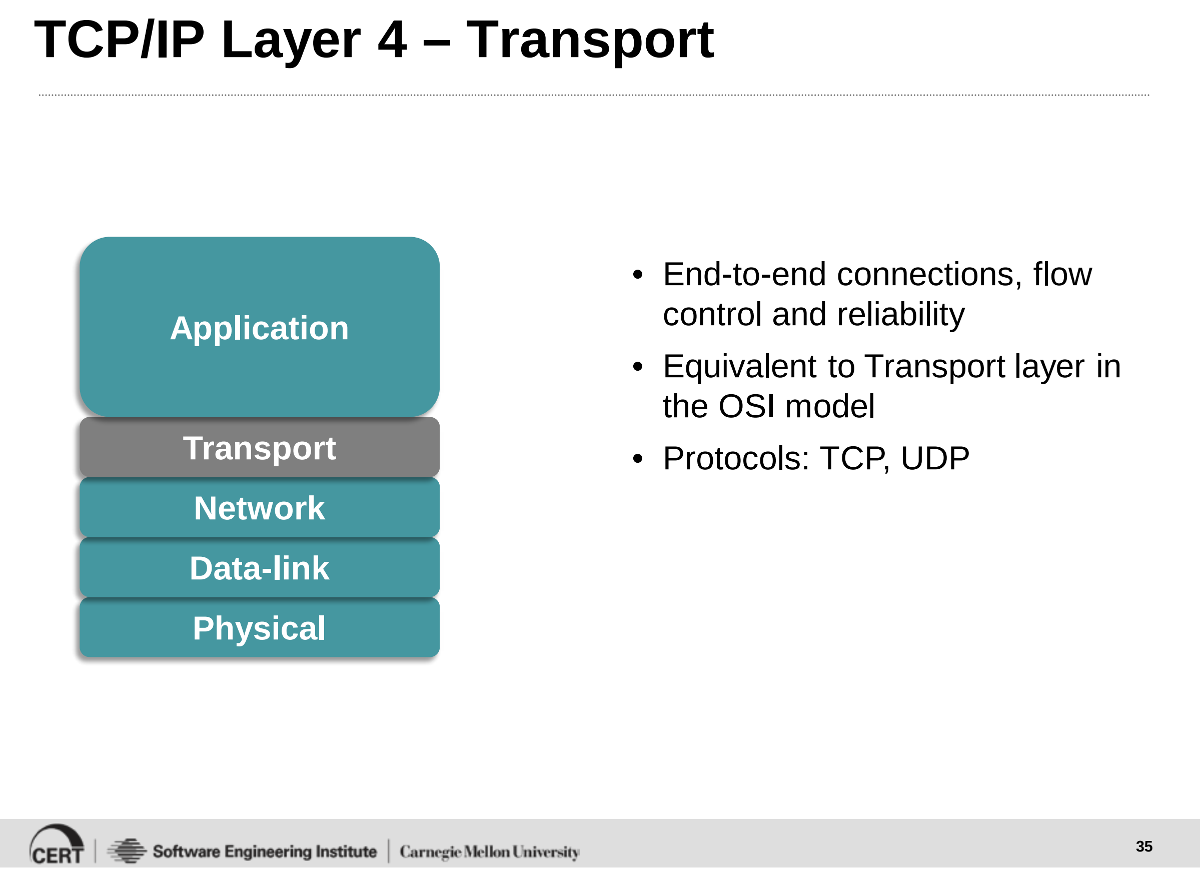
- Logical addressing, route selection
- Fragmentation and re-assembly
- Equivalent to Network layer in OSI model
- Devices – Layer 3 bridges, routers
- Protocols: IP, ICMP, IPSec

34

**034 TCP Network layer. Same thing all over again. Same kind of logical addressing.

Routed protocols versus routing protocols. Routed protocols means the data is being transmitted from host to host. Routing protocols means that when we're talking about the routers that are in the way that we route around the congestion. We'll talk about those in a bit.

TCP/IP Layer 4 – Transport



TCP/IP Layer 4 – Transport

Application

Transport

Network

Data-link

Physical

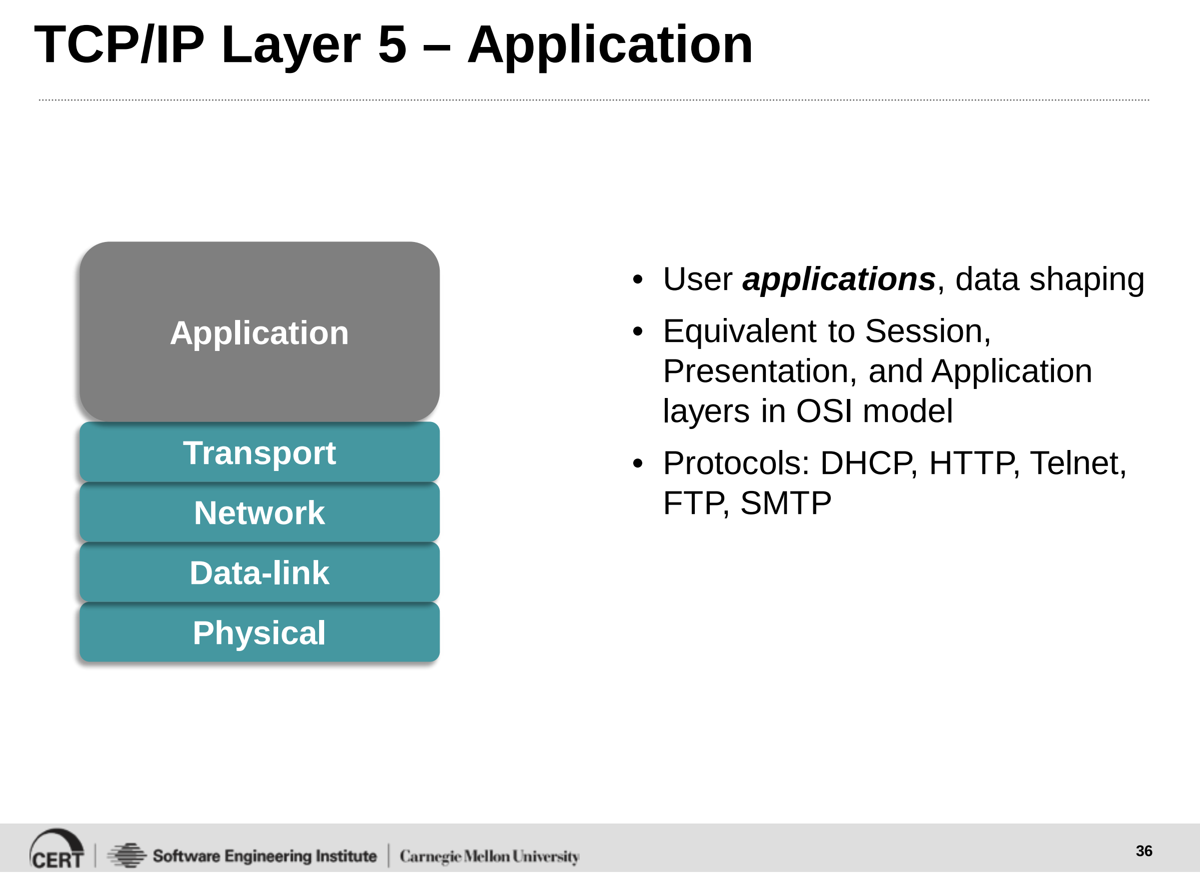
- End-to-end connections, flow control and reliability
- Equivalent to Transport layer in the OSI model
- Protocols: TCP, UDP

CERT | Software Engineering Institute | Carnegie Mellon University

35

**035 Transport Layer. Again the same thing. At the Transport layer for TCP there are definitely only two protocols: TCP and UDP.

TCP/IP Layer 5 – Application



TCP/IP Layer 5 – Application

- User ***applications***, data shaping
- Equivalent to Session, Presentation, and Application layers in OSI model
- Protocols: DHCP, HTTP, Telnet, FTP, SMTP

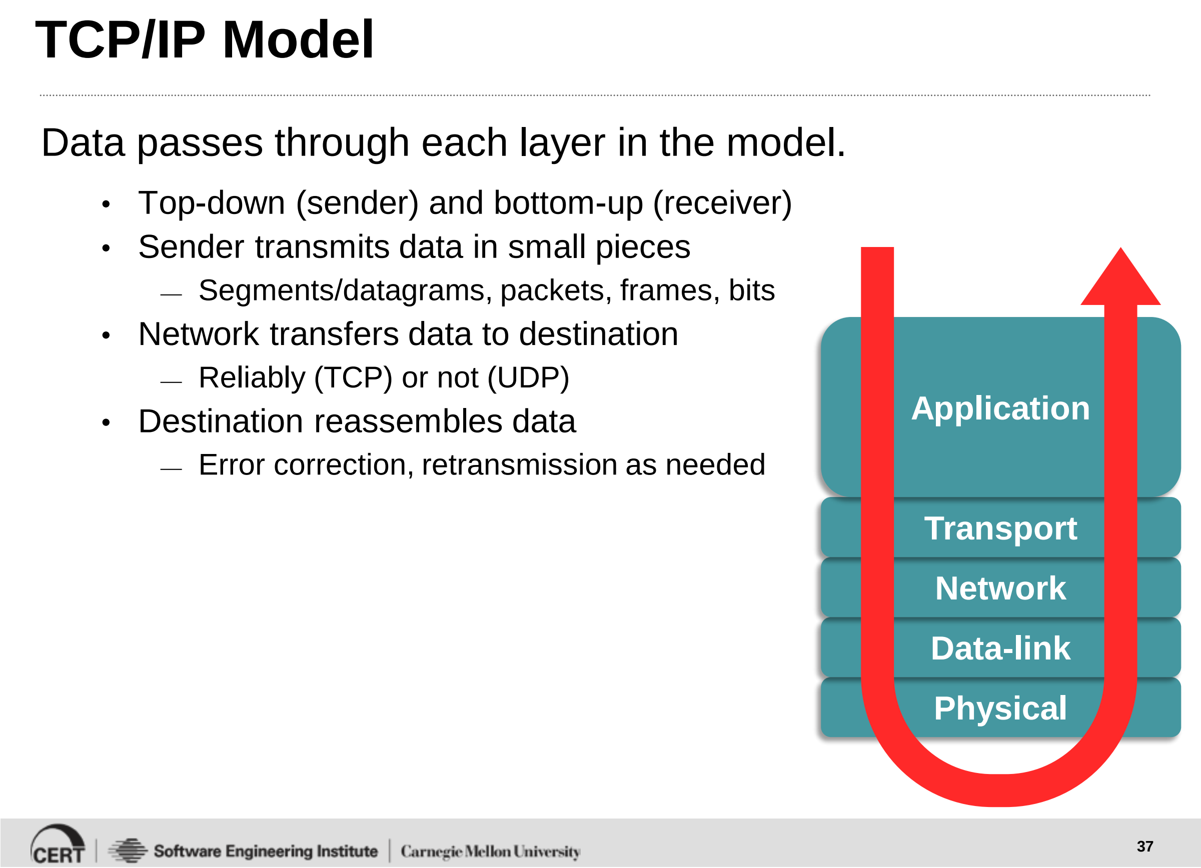
36

**036 At the Application layer it all got jammed together-- Session, Presentation and Application-- when we look at the protocols that are there.

These are the same protocols that were in the OSI. But the problem was is we didn't quite know what layer to put Dynamic Host Configuration protocol. We didn't quite know where to put SMTP; because it's not really an end-user mail protocol, unless you're POPing or IMAPing in. I mean, yes you can SMTP back. But we didn't quite know where to put those in the OSI model because they kind of bled into all the different layers.

And so that's why TCP/IP, from an implementation standpoint, is actually a better model; because those abstractions that had to be done at layer after layer after layer, those three layers there, could all be satisfied by one executable or one DLL. And so it makes it a little bit easier for us to wrap them all together into the Application layer; and we leave all that stuff there.

TCP/IP Model



TCP/IP Model

Data passes through each layer in the model.

- Top-down (sender) and bottom-up (receiver)
- Sender transmits data in small pieces
  - Segments/datagrams, packets, frames, bits
- Network transfers data to destination
  - Reliably (TCP) or not (UDP)
- Destination reassembles data
  - Error correction, retransmission as needed

CERT | Software Engineering Institute | Carnegie Mellon University 37

**037 Now when we talk about TCP model-- and you could say the exact same thing for the OSI model; and so what we're going to do is we're going to leave OSI behind and we're going to only focus on TCP/IP pretty much for the rest of this session.

When you've passed the data down or up the stack, what happens is encapsulation or de-encapsulation.

As we move up and down the layers, we call the name of the protocol data unit at that layer something distinct. So at the Physical layer it's called bits; at the Data-link layer-- let's see it's-- the way to do this is to talk about bits, frames, at the Data-link layer.

Very rarely do we hear that; because people don't talk about the frames anymore: Well there's a problem with the frames.

The problem is is that everybody says one universal word when they're talking about non-application layer protocols, in most cases; they say the word 'packet'. And so we accept that from the rest of the world; but we want to be a little bit more specific here.

And what happens with these protocol data units is at the frame level we do these kinds of activities. If we have a frame problem, we know it's not an end-user issue; it's something either wrong with the stack or it's something wrong with our resolution.

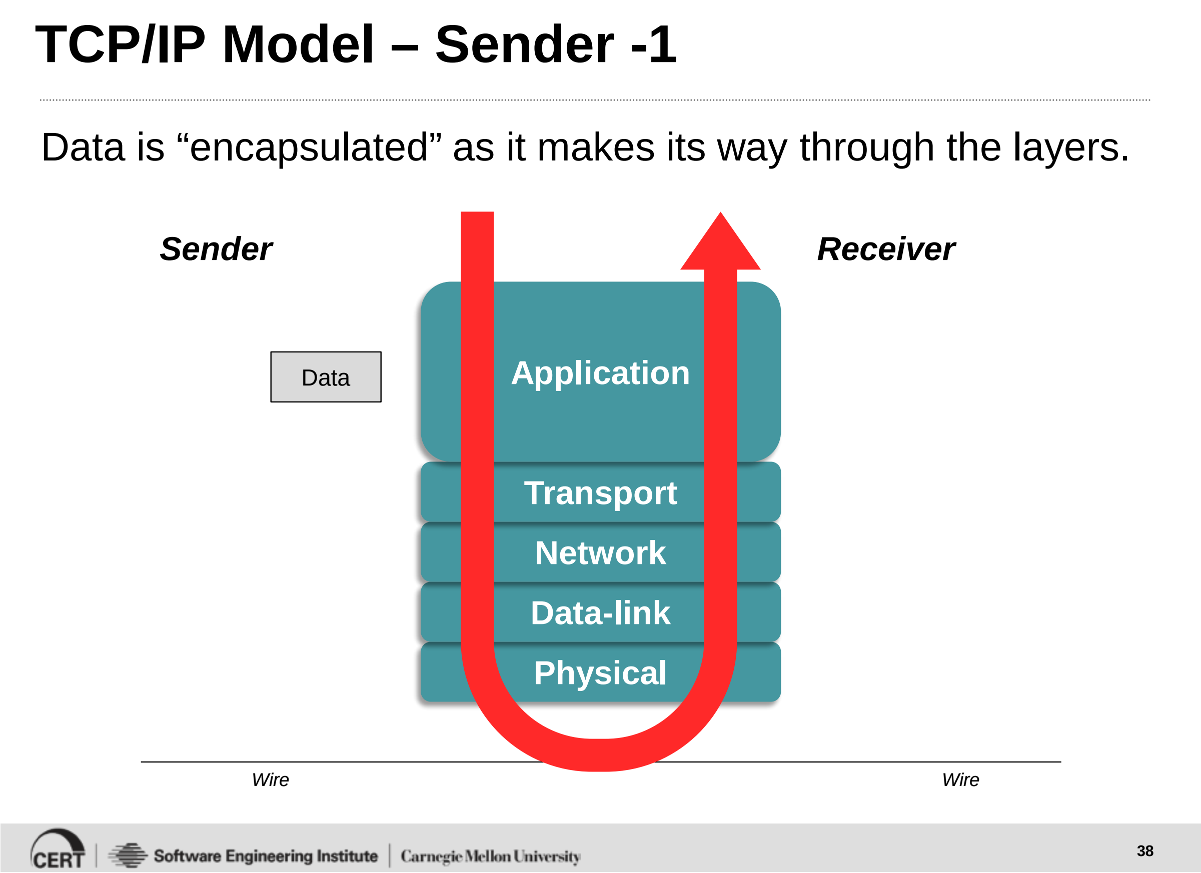
Now what happens in TCP model is we pass through each one of these layers; and each one of these protocol data units, or names for things at that layer, does its particular purpose; and then it's set it and forget it for the other layers.

When I do all my frame work at the Data-link layer, the Physical layer knows it needs to give it to me in a certain format. But that's pretty much it; and it de-encapsulates and hands that frame to me.

I look at this frame and I take my action. I de-encapsulate; and then I pass that up to the Network layer.

So as we send and transmit segments, packets, data, frames, bits, when we start doing this transmission back and forth we use those constructs.

TCP/IP Model – Sender -1



TCP/IP Model – Sender -1

Data is “encapsulated” as it makes its way through the layers.

Sender

Data

Application

Transport

Network

Data-link

Physical

Receiver

Wire

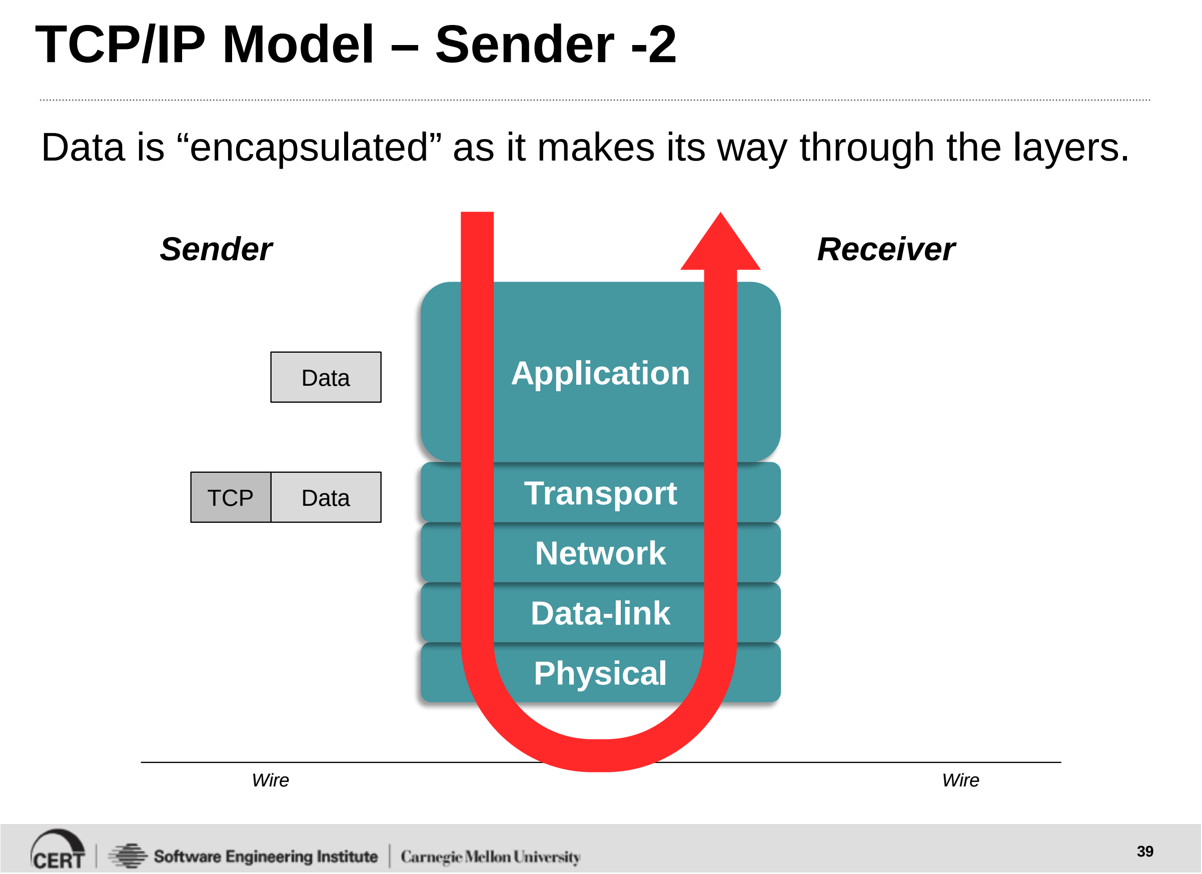
Wire

CERT | Software Engineering Institute | Carnegie Mellon University

38

**038 So as we pass down the
stack we're going to add information.
We're going to encapsulate.

TCP/IP Model – Sender -2



TCP/IP Model – Sender -2

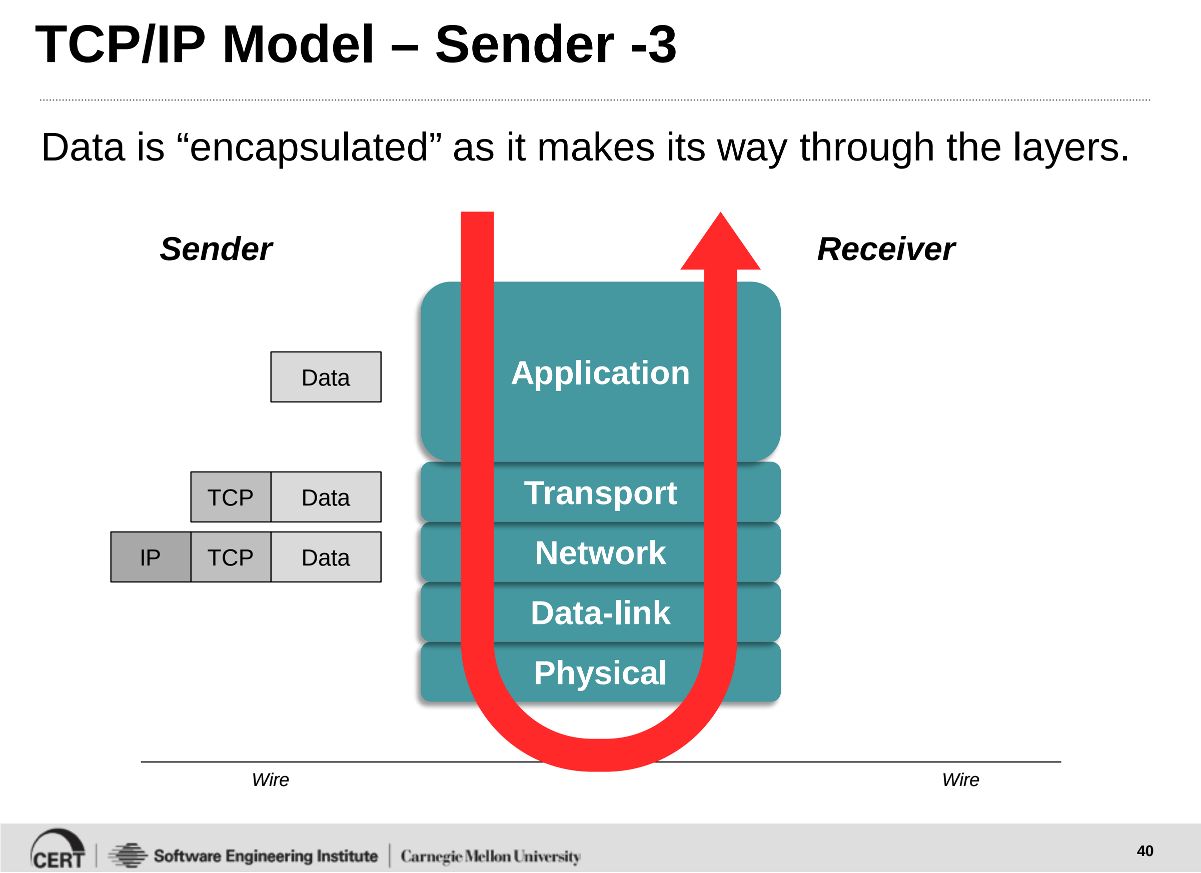
Data is “encapsulated” as it makes its way through the layers.

CERT | Software Engineering Institute | Carnegie Mellon University

39

**039 So from the Application layer we have pure data. At the Transport layer we add in the TCP header information that is valid for transmitting to the host on the other side.

TCP/IP Model – Sender -3



TCP/IP Model – Sender -3

Data is “encapsulated” as it makes its way through the layers.

CERT | Software Engineering Institute | Carnegie Mellon University 40

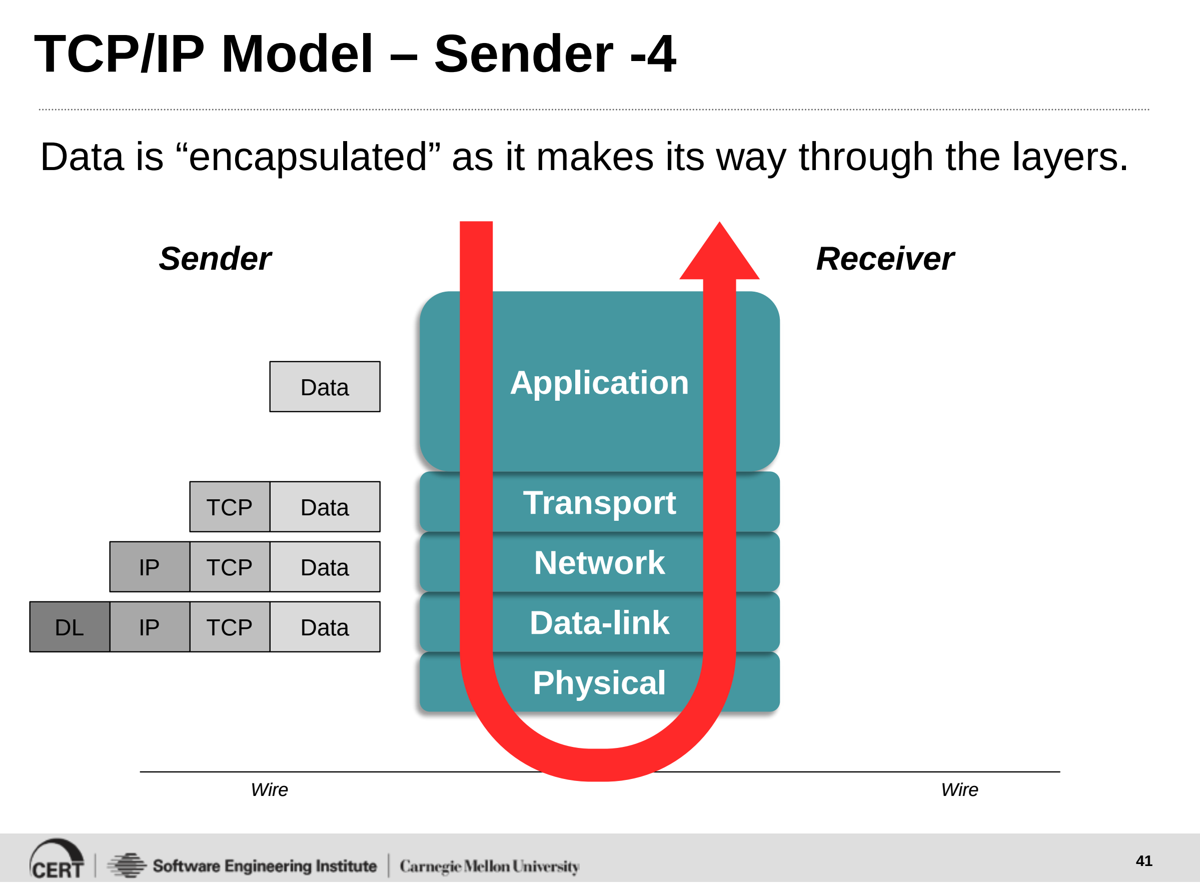
**040 At the Network layer we add in the IP header information.

Now at the Network layer, by the way, we don't know that there's TCP and data as separate things. We treat it all like data.

So you could- here you could cross off the word TCP and data at this layer and just write the word data; and that would be accurate because the IP layer, the Network layer, does not know anything about the preceding headers above it.

When it passes it down the stack again, this all becomes data.

TCP/IP Model – Sender -4



TCP/IP Model – Sender -4

Data is “encapsulated” as it makes its way through the layers.

Sender

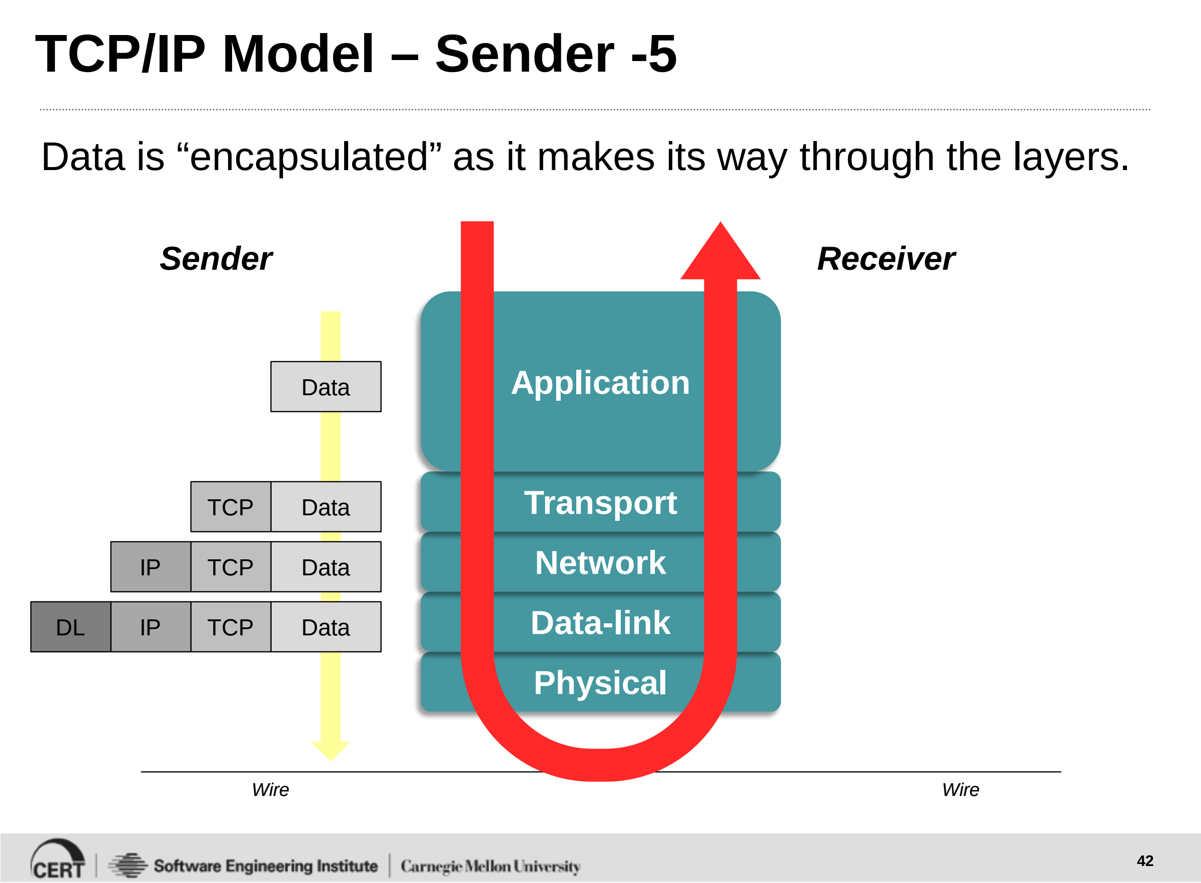
Receiver

			Data
		TCP	Data
	IP	TCP	Data
DL	IP	TCP	Data

CERT | Software Engineering Institute | Carnegie Mellon University 41

**041 And at the Data-link layer we add the header information of the source and destination MAC address; and we also add a trailer at this point, which is the Cyclical Redundancy Checking-- CRC or FCS if you're a Cisco person. So we actually do encapsulation on both sides.

TCP/IP Model – Sender -5



TCP/IP Model – Sender -5

Data is “encapsulated” as it makes its way through the layers.

Sender

Receiver

Data

TCP Data

IP TCP Data

DL IP TCP Data

Application

Transport

Network

Data-link

Physical

Wire

Wire

CERT | Software Engineering Institute | Carnegie Mellon University

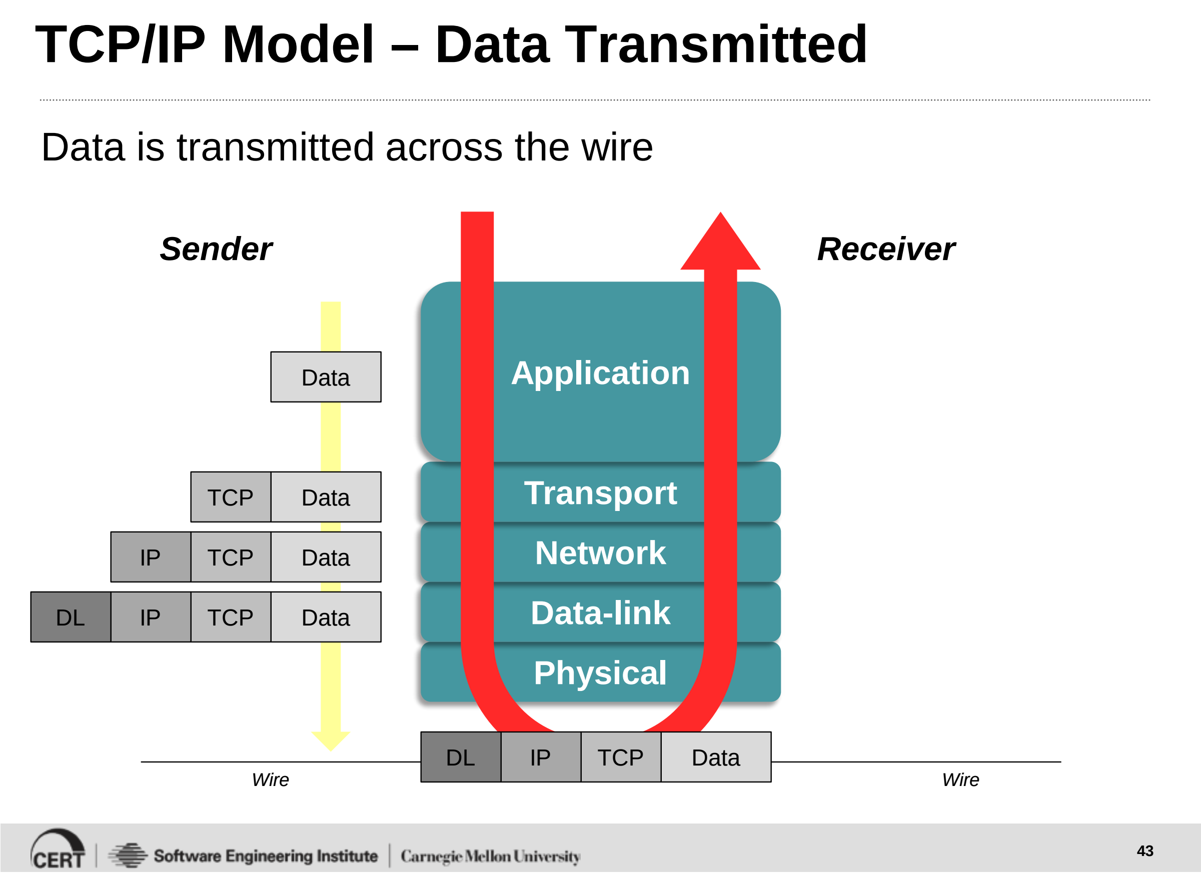
42

**042 At the Physical layer what we're doing is we're converting it into signal. We're converting it- for copper we're converting it into electronic charges of 1s and 0s, on's and off.

If we're converting it over to fiber, then what we're doing is is we're turning it into light pulses at that point. And that gets a little bit beyond- how the light pulses work gets beyond what we do here.

We pass it across the wire as a whole.

TCP/IP Model – Data Transmitted



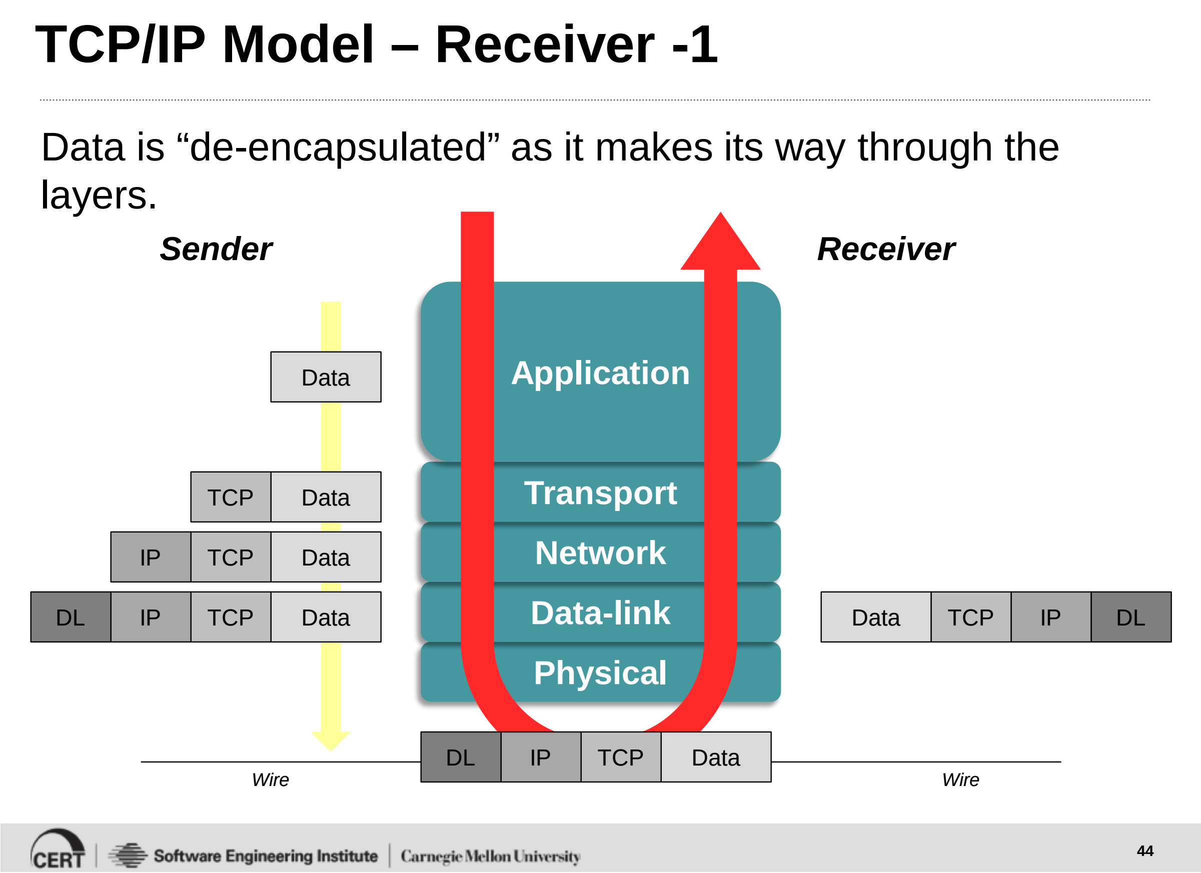
TCP/IP Model – Data Transmitted

Data is transmitted across the wire

Software Engineering Institute | Carnegie Mellon University 43

**043 So on the wire it looks like
all 1s and 0s.

TCP/IP Model – Receiver -1

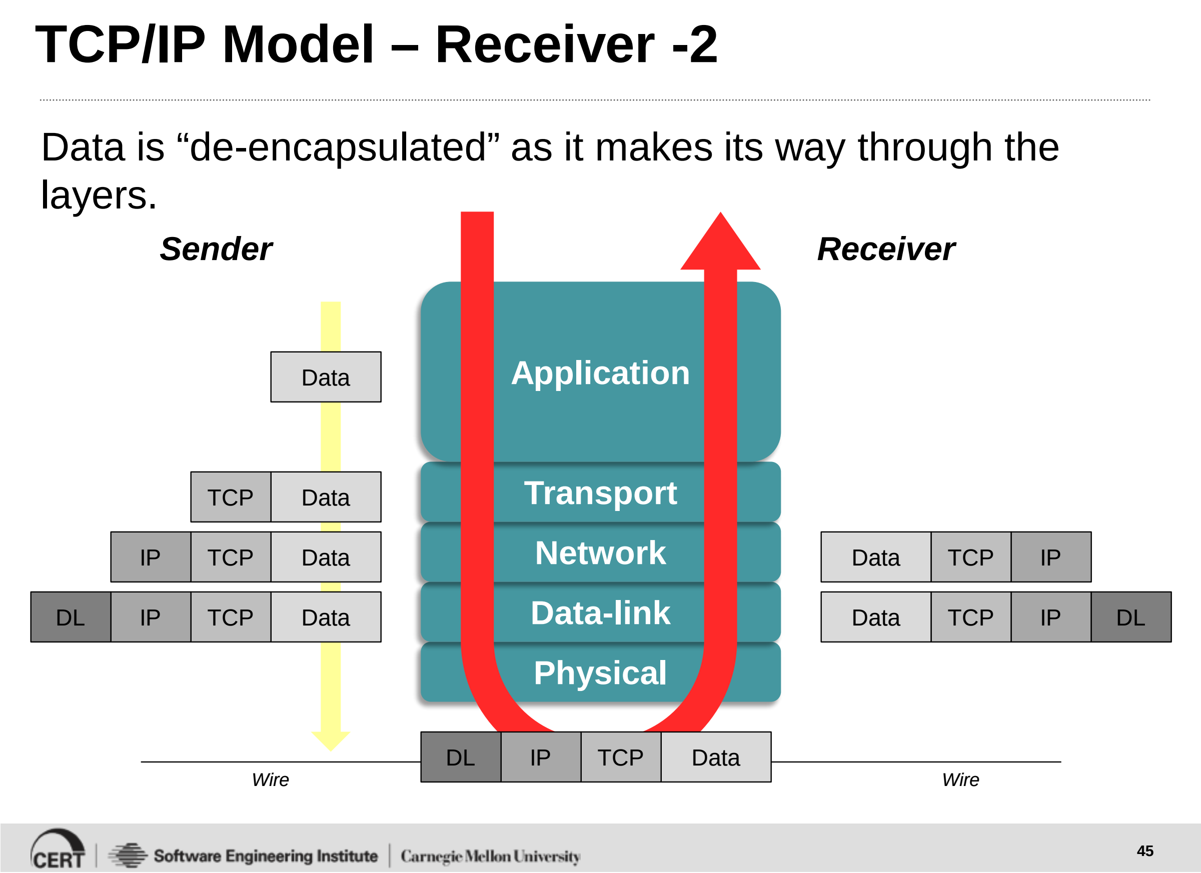


TCP/IP Model – Receiver -1

Data is “de-encapsulated” as it makes its way through the layers.

**044 But when it gets to the receiving host on the other side, it says: Oh I know what to do with that; that is destined for me based upon my MAC address; or based upon the broadcast. I'm going to take care of that and I'm going to rip off the Data-link layer information.

TCP/IP Model – Receiver -2



TCP/IP Model – Receiver -2

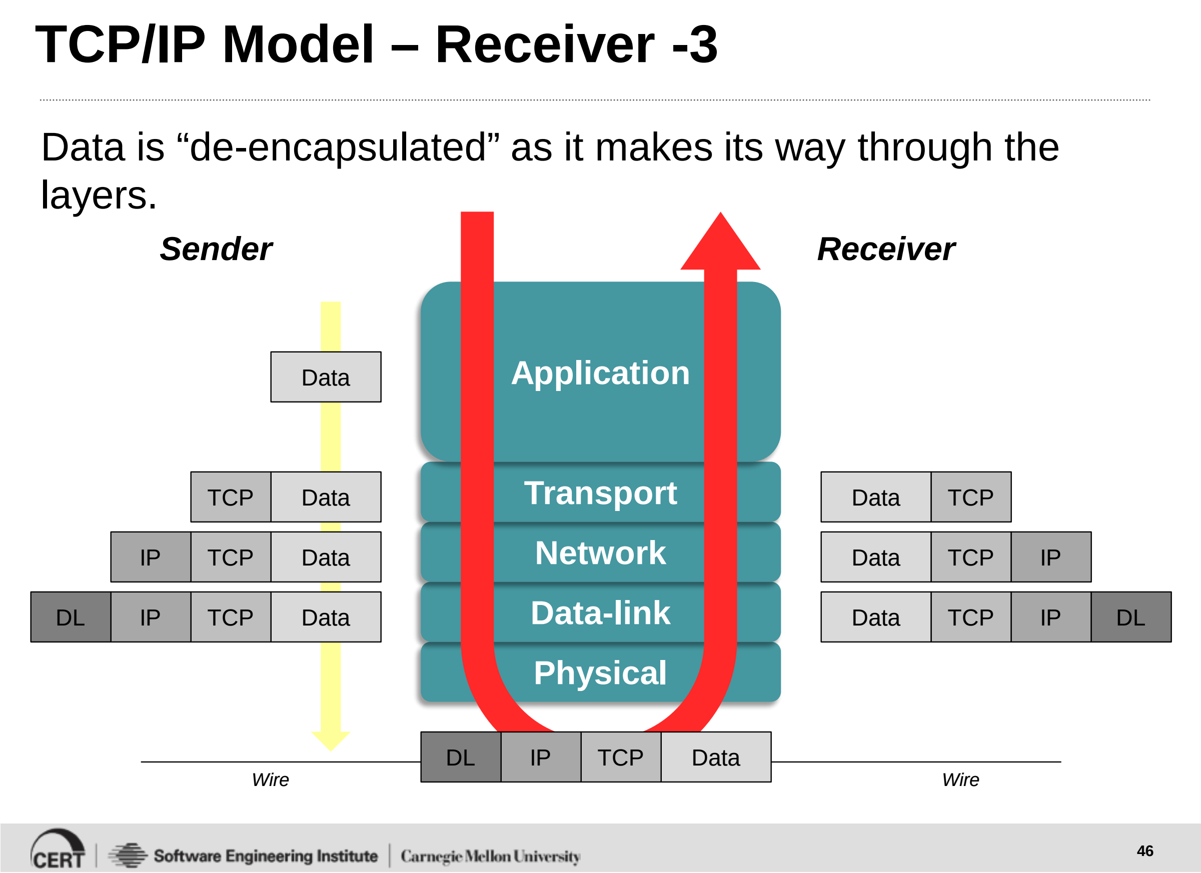
Data is “de-encapsulated” as it makes its way through the layers.

Sender

Receiver

**045 And I'm going to pass it up the stack; and I'll trust that the Network layer knows what to do with this. Because I knew what to do, my part, and I'm done my part; and I pass it up the stack.

TCP/IP Model – Receiver -3

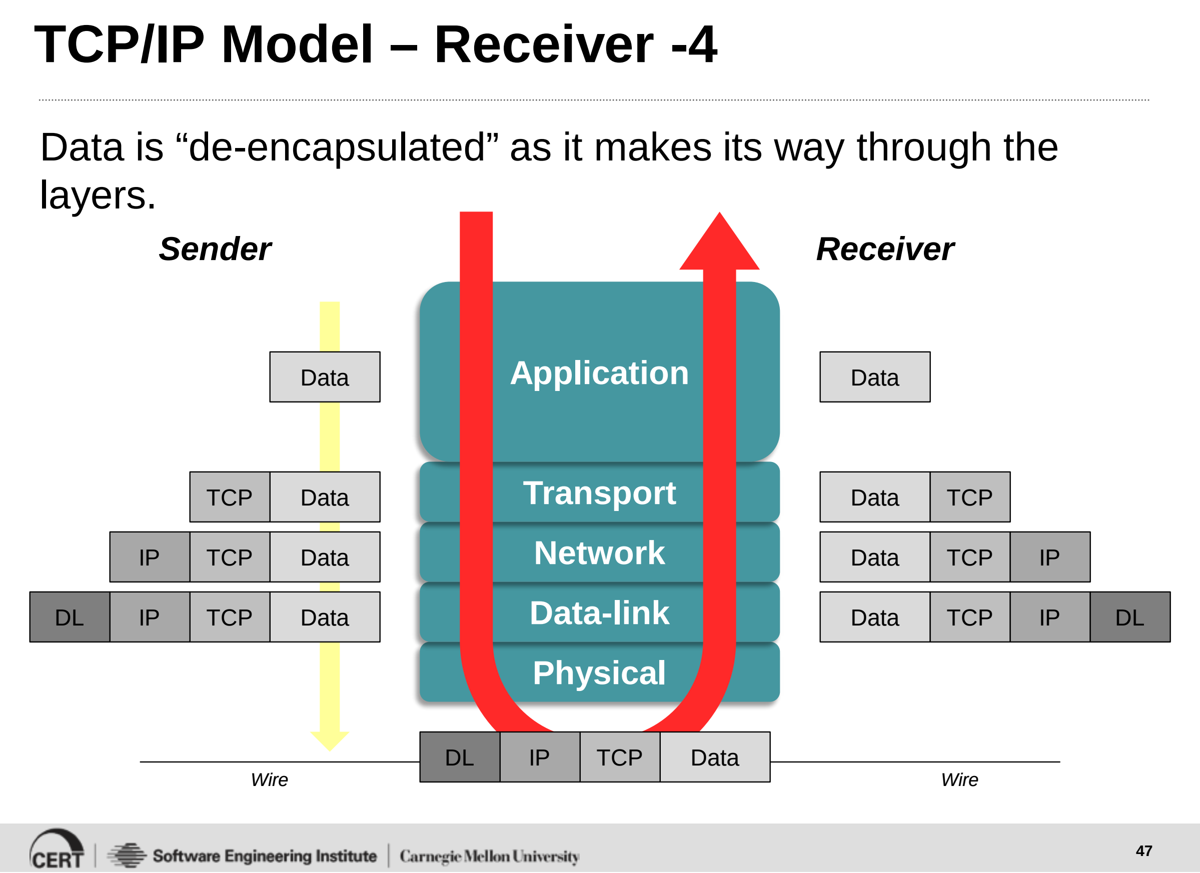


TCP/IP Model – Receiver -3

Data is “de-encapsulated” as it makes its way through the layers.

**046 Same thing with the Transport layer. The IP layer is ripped off; and now it makes TCP decisions.

TCP/IP Model – Receiver -4



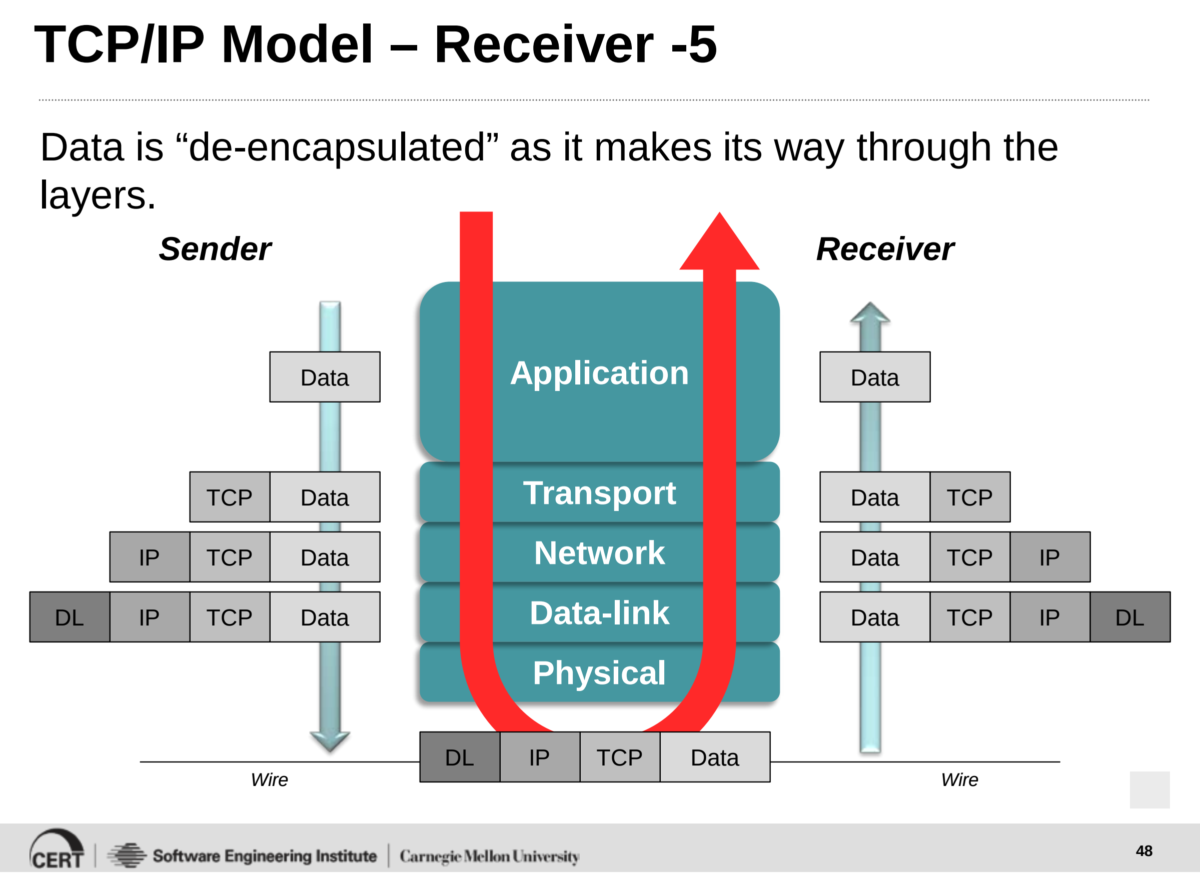
TCP/IP Model – Receiver -4

Data is “de-encapsulated” as it makes its way through the layers.

CERT | Software Engineering Institute | Carnegie Mellon University 47

**047 Finally it gets to the top of the food chain, where it's now data at that point, and now we've stripped off all the header information; and the Application is going to make probably some sort of presentation actions that occur at that point.

TCP/IP Model – Receiver -5

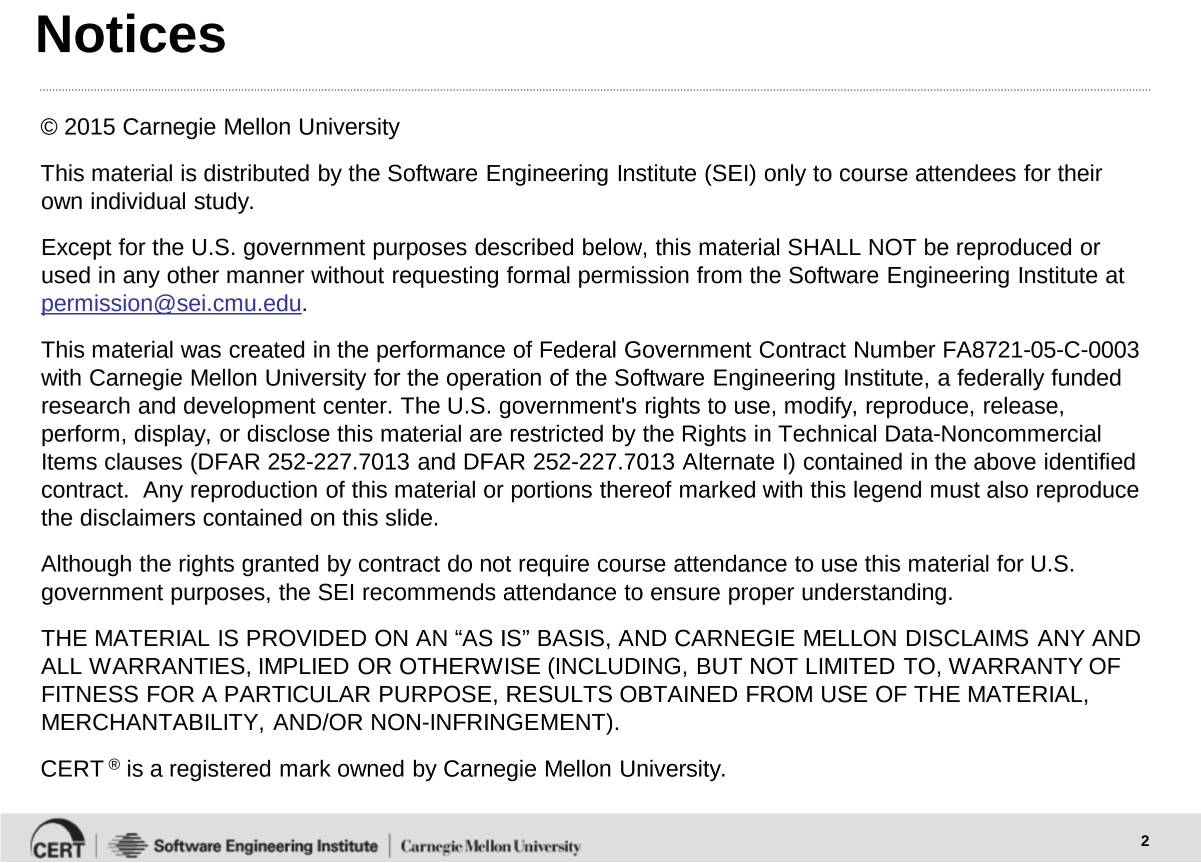


TCP/IP Model – Receiver -5

Data is “de-encapsulated” as it makes its way through the layers.

**048 So we've got a sender and a receiver; an encapsulation as we go down, a de-encapsulation as we go up.

Notices



Notices

This material is distributed by the Software Engineering Institute (SEI) only to course attendees for their own individual study.

Except for the U.S. government purposes described below, this material SHALL NOT be reproduced or used in any other manner without requesting formal permission from the Software Engineering Institute at permission@sei.cmu.edu.

This material was created in the performance of Federal Government Contract Number FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The U.S. government's rights to use, modify, reproduce, release, perform, display, or disclose this material are restricted by the Rights in Technical Data-Noncommercial Items clauses (DFAR 252-227.7013 and DFAR 252-227.7013 Alternate I) contained in the above identified contract. Any reproduction of this material or portions thereof marked with this legend must also reproduce the disclaimers contained on this slide.

Although the rights granted by contract do not require course attendance to use this material for U.S. government purposes, the SEI recommends attendance to ensure proper understanding.

THE MATERIAL IS PROVIDED ON AN “AS IS” BASIS, AND CARNEGIE MELLON DISCLAIMS ANY AND ALL WARRANTIES, IMPLIED OR OTHERWISE (INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE, RESULTS OBTAINED FROM USE OF THE MATERIAL, MERCHANTABILITY, AND/OR NON-INFRINGEMENT).

CERT ® is a registered mark owned by Carnegie Mellon University.