NTNU | Kunnskap for en bedre verden

TTM4175 – Week 36

Networking II

Ports, Layers, Client-Server Architecture, Web Servers

Goals

- Recognize the importance of **ports and layers**
  - Run and interact with a basic web server
  - Investigate the protocol stack
  - Observe HTTP traffic
- Learn basic **Docker** principles
  - Images and containers
  - The Dockerfile
  - Basic commands for managing containers

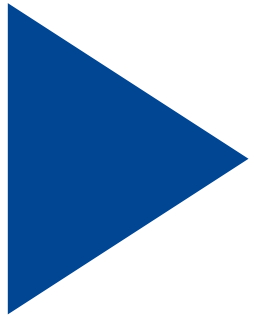
Recap of Preparation Material

Readings

Layered architecture

Network application architectures

Processes communicating



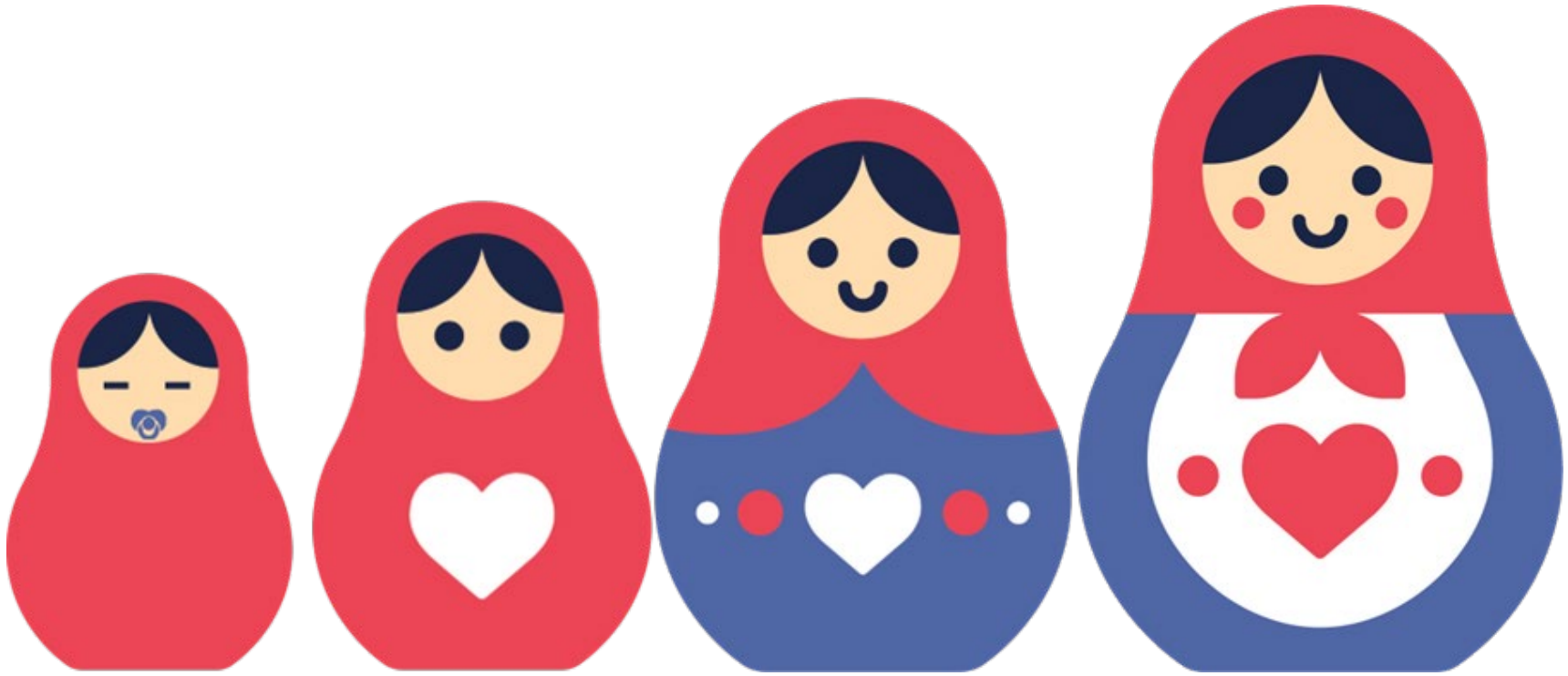
Videos

Client-server architecture

What is a server?

Docker, web servers (optional)

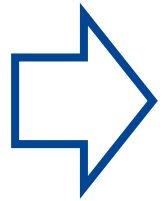
Layers and Encapsulation



Protocol Layers and Reference Models

Networks are complex, with many “pieces”

- Hosts
- Routers
- Links of various media
- Applications
- Protocols
- Hardware, software



Any hope of *organizing* the structure of networks and our discussion of them?

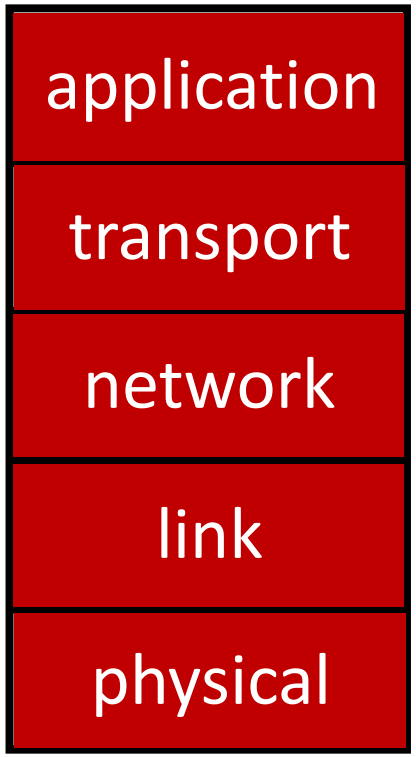
Why Layering?

Approach to designing and discussing complex systems

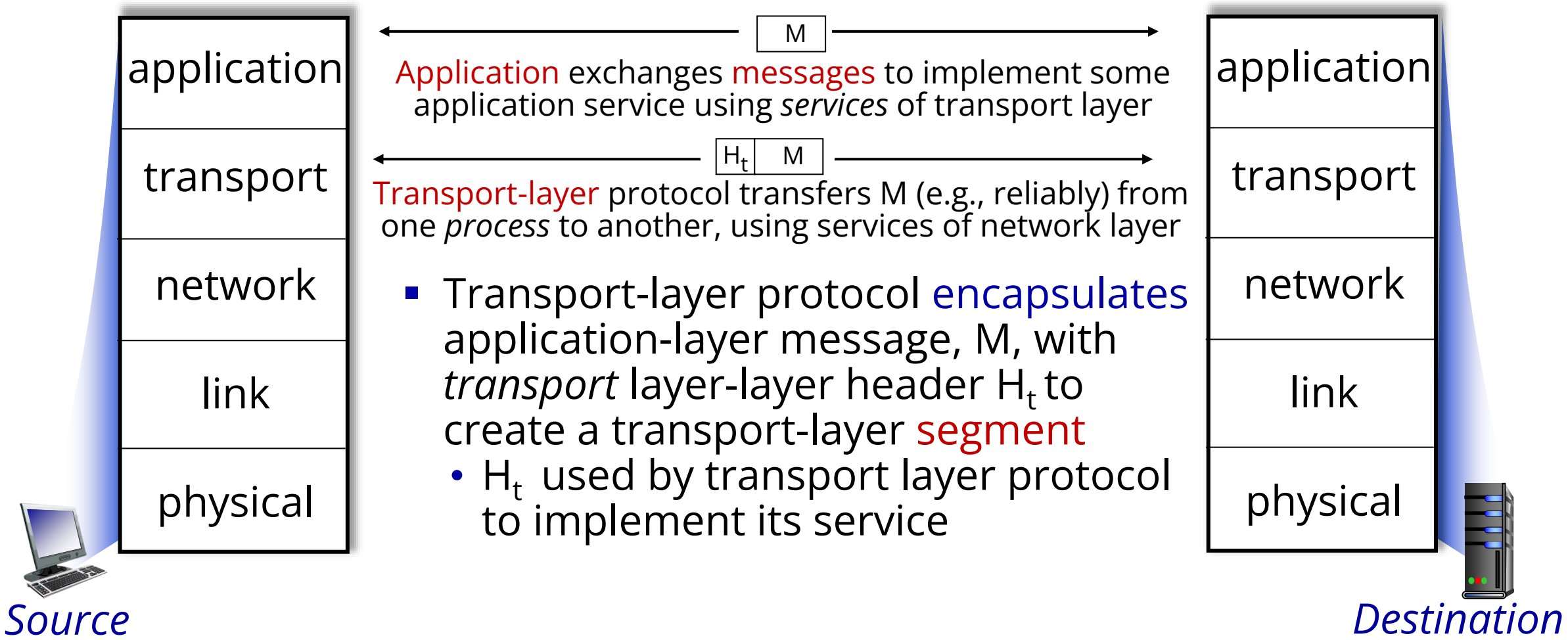
- Explicit structure allows identification, relationship of system's pieces
  - Layered *reference model* for discussion
- Modularization eases maintenance, updating of system
  - Changes in layers' service *implementation* transparent to rest of system
  - Airline example: change in gate procedure doesn't affect rest of system

Layered Internet Protocol Stack

- *Application:* supports network applications
  - HTTP, IMAP, SMTP, DNS, ..
- *Transport:* process-to-process data transfer
  - TCP, UDP
- *Network:* routing of datagrams from source to destination
  - IP, routing protocols
- *Link:* data transfer between neighboring network elements
  - Ethernet, 802.11 (WiFi), PPP
- *Physical:* bits “on the wire”

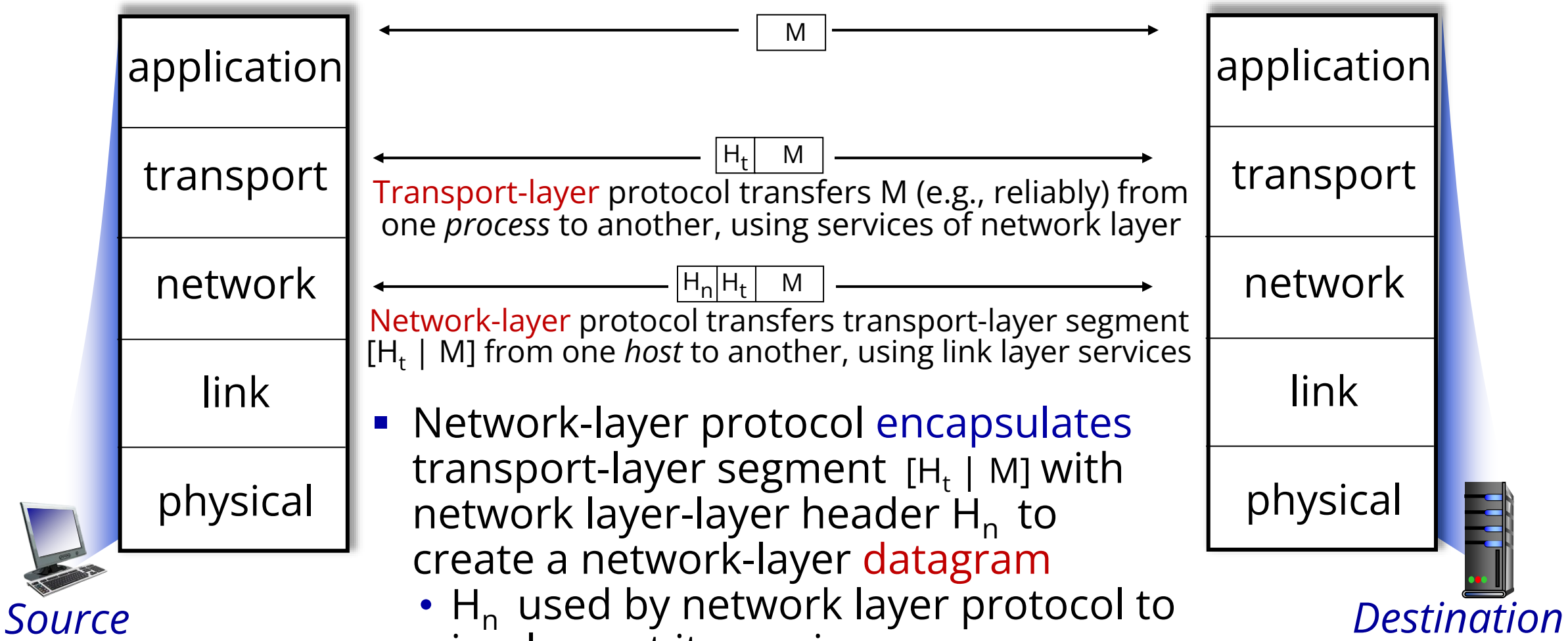


Services, Layering, and Encapsulation



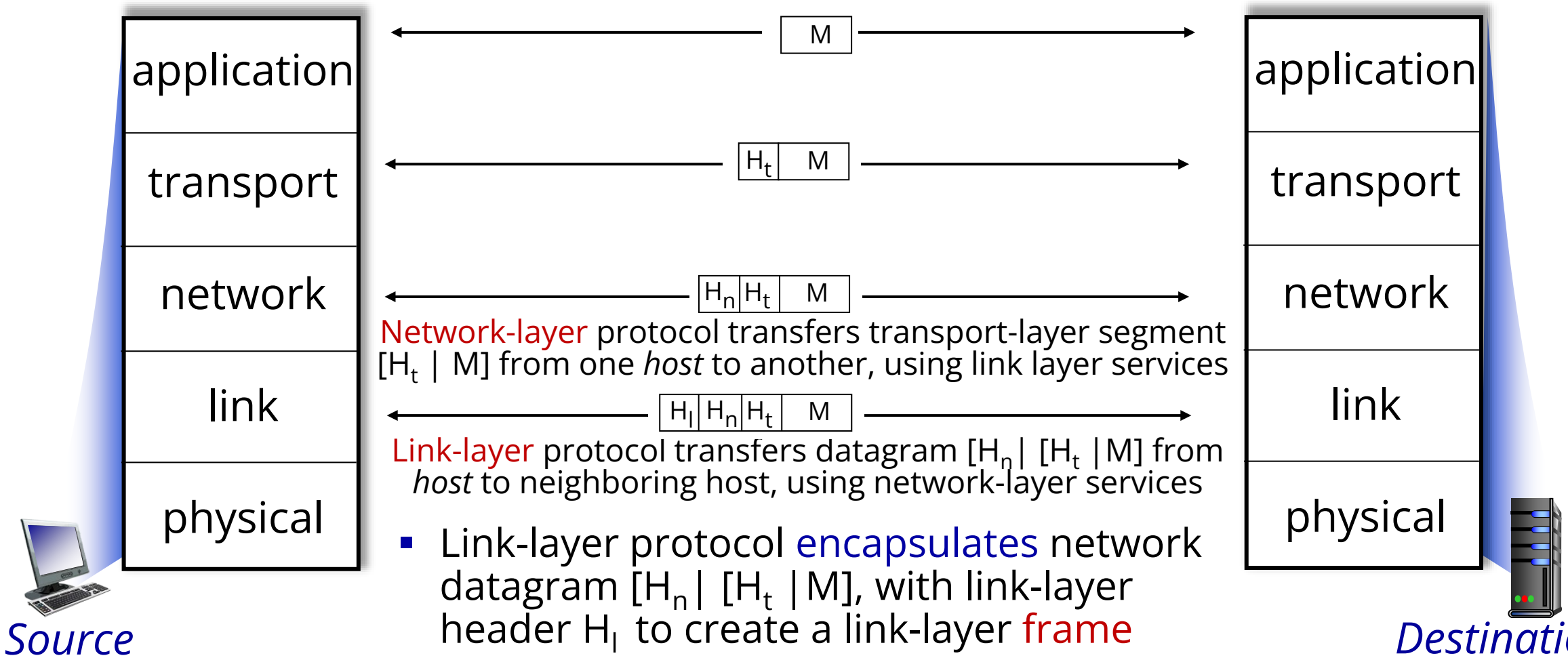
- Transport-layer protocol encapsulates application-layer message, M, with *transport* layer-layer header H_t to create a transport-layer segment
  - H_t used by transport layer protocol to implement its service

Services, Layering, and Encapsulation



- Network-layer protocol encapsulates transport-layer segment [H_t | M] with network layer-layer header H_n to create a network-layer datagram
  - H_n used by network layer protocol to implement its service

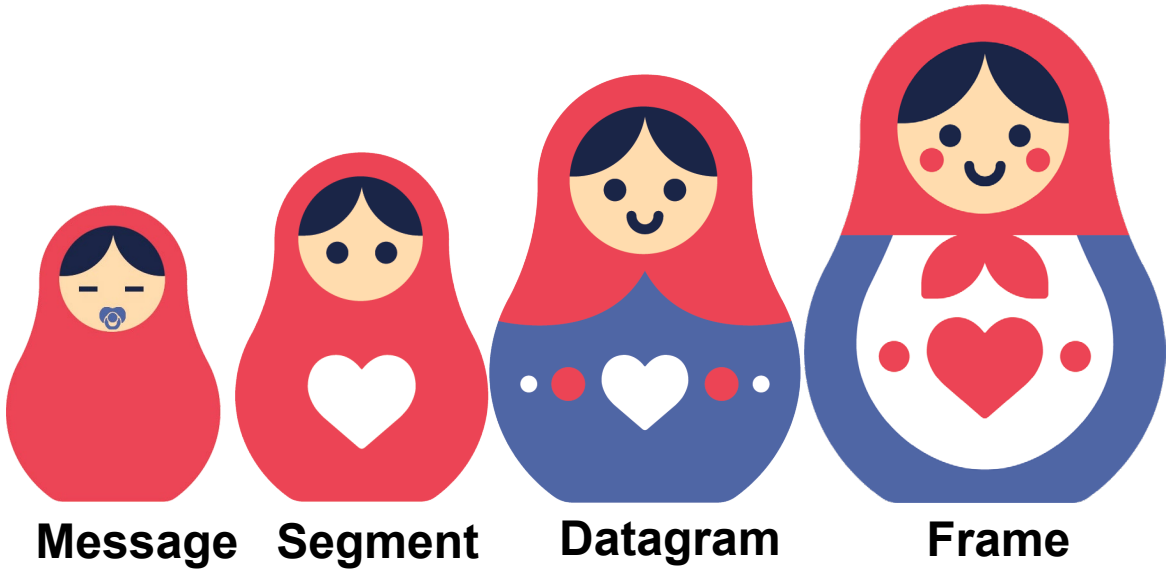
Services, Layering, and Encapsulation



- Link-layer protocol encapsulates network datagram [H_n| [H_t |M], with link-layer header H_l to create a link-layer frame

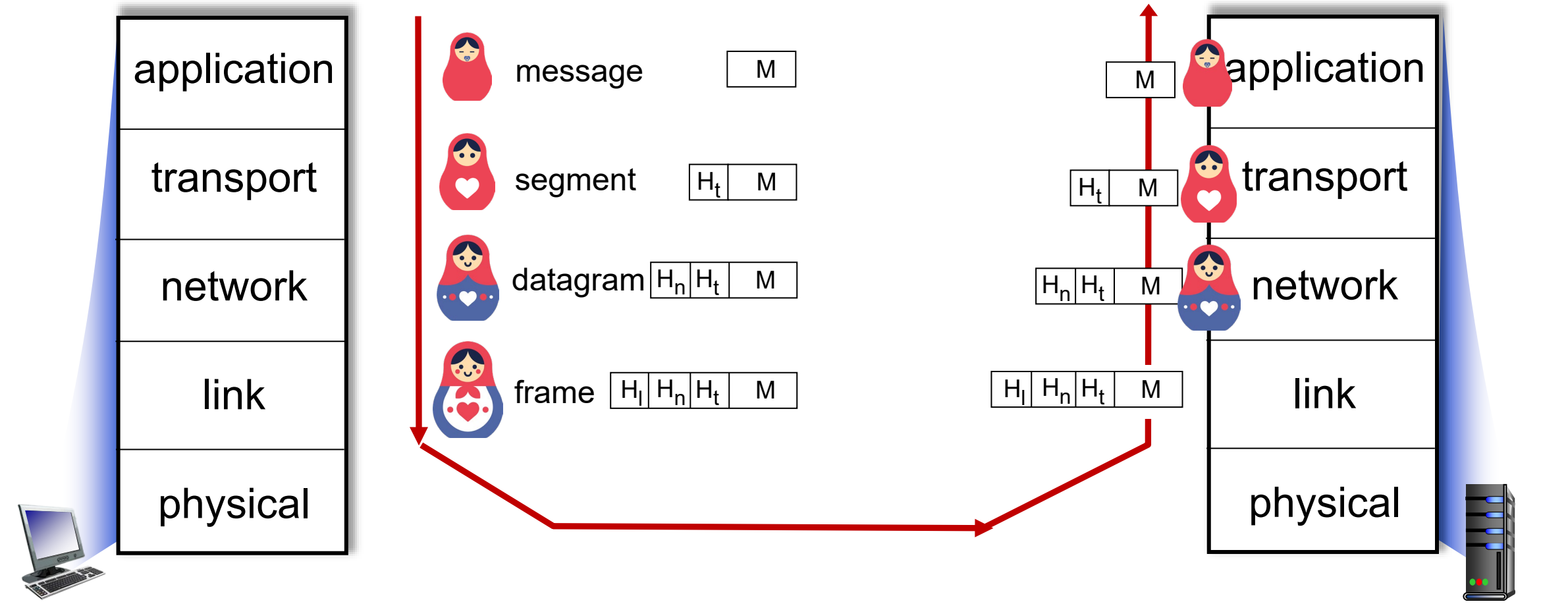
Encapsulation

Matryoshka dolls (stacking dolls)

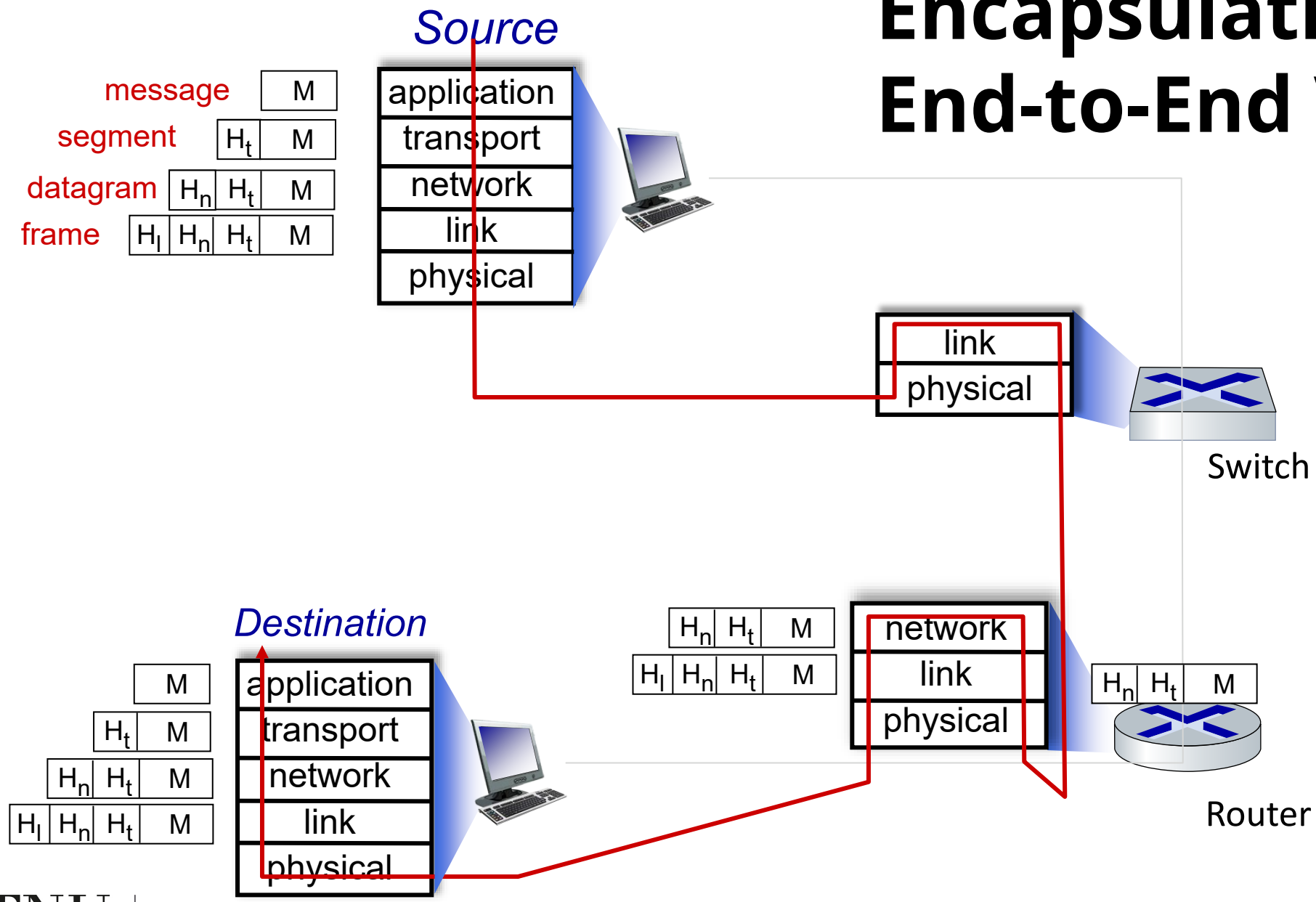


Credit: https://dribbble.com/shots/7182188-Babushka-Boi

Services, Layering, and Encapsulation



Encapsulation: an End-to-End View



Sockets and Ports

Some Network Apps

- Social networking
- Web
- Text messaging
- E-mail
- Multiplayer network games
- Streaming stored video (YouTube, Hulu, Netflix)
- P2P file sharing
- Voice over IP (e.g., Skype)
- Real-time video conferencing (e.g., Zoom)
- Internet search
- Remote login
- …

Processes Communicating

Process: program running within a host

- Within same host, two processes communicate using inter-process communication (defined by OS)
- Processes in *different* hosts communicate by exchanging messages

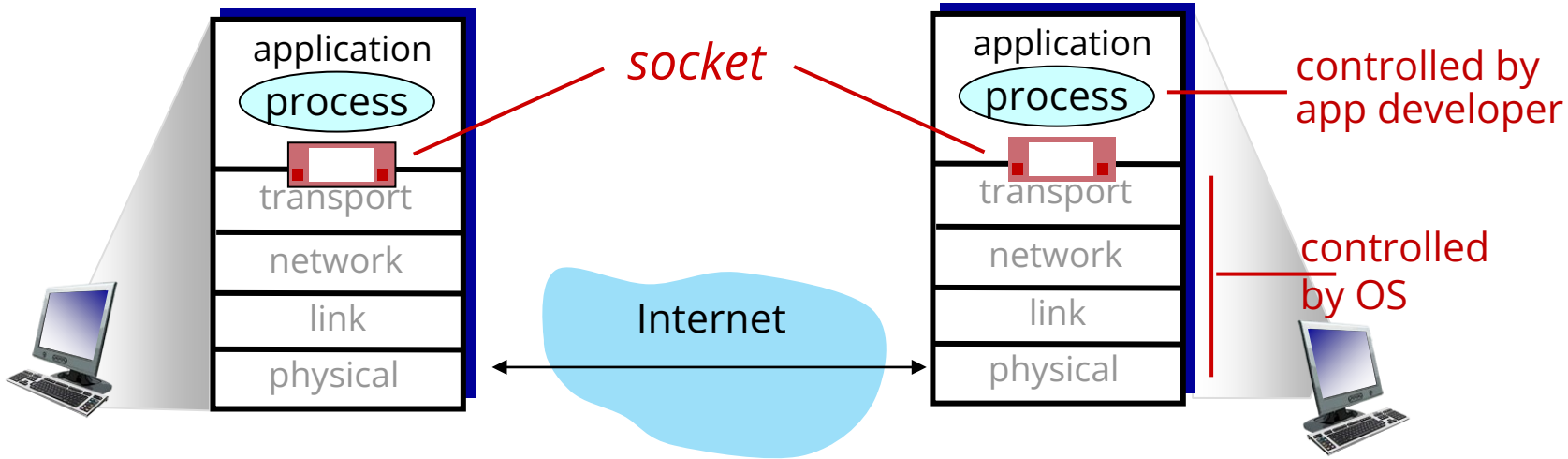
Clients, servers

Client process: process that *initiates* communication

Server process: process that *waits* to be contacted

Sockets

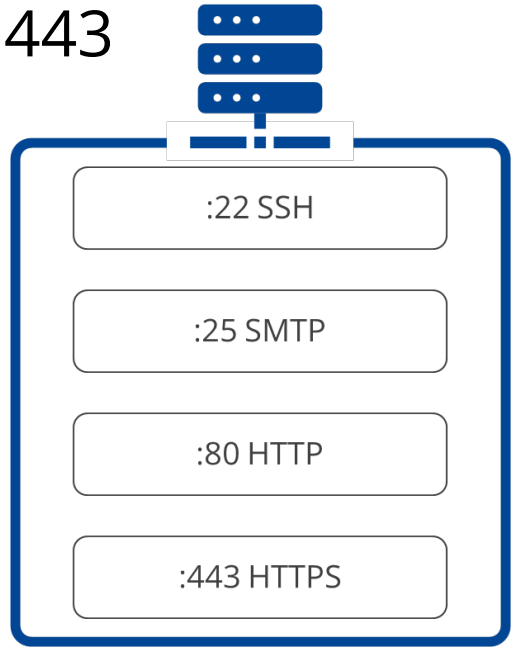
- Process sends/receives messages to/from its socket
- Socket analogous to door
  - Sending process shoves message out door
  - Sending process relies on transport infrastructure on other side of door to deliver message to socket at receiving process
  - Two sockets involved: one on each side



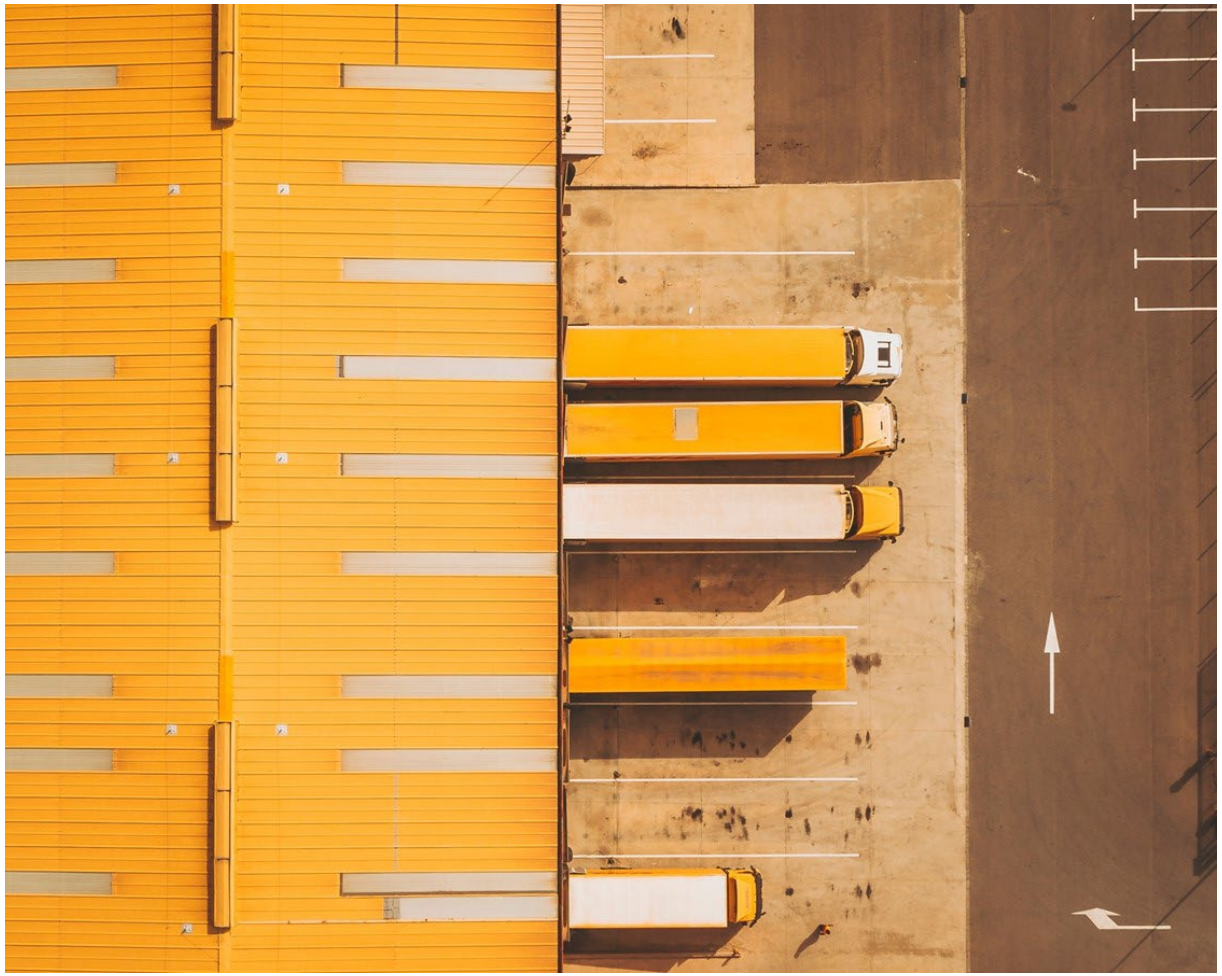
Addressing Processes

- To receive messages, process must have *identifier*
- Host device has unique 32-bit IP address
- *Q:* does IP address of host on which process runs suffice for identifying the process?
  - *A:* no, *many* processes can be running on same host

- *Identifier* includes both IP address and port numbers associated with process on host
- Example port numbers
  - HTTP(S) server: 80, 443
  - SSH: 22
  - Mail server: 25



Transport Protocols and Applications



Application-Layer Protocols Define

- Types of messages exchanged, e.g., request, response
- Message syntax
  - What fields in messages & how fields are delineated
- Message semantics
  - Meaning of information in fields
- Rules for when and how processes send & respond to messages

Open protocols

- Defined in RFCs, everyone has access to protocol definition
- Allow for interoperability
- Examples: HTTP, SMTP

Proprietary protocols:

- Examples: Skype, Zoom

What Transport Service Does an App Need?

Data integrity

- Some apps (e.g., file transfer, web transactions) require 100% reliable data transfer
- Other apps (e.g., audio) can tolerate some loss

Timing

- Some apps (e.g., Internet telephony, interactive games) require low delay to be "effective"

Throughput

- Some apps (e.g., multimedia) require minimum amount of throughput to be "effective"
- Other apps ("elastic apps") make use of whatever throughput they get

Security

- Encryption, data integrity, …

Transport Service Requirements of Common Apps

Application	Data loss	Throughput	Time-sensitive?
File transfer/download	no loss	elastic	no
E-mail	no loss	elastic	no
Web documents	no loss	elastic	no
Real-time audio/video	loss-tolerant	audio: 5Kbps-1Mbps video:10Kbps-5Mbps	yes, 10's msec
Streaming audio/video	loss-tolerant	same as above	yes, few secs
Interactive games	loss-tolerant	Kbps+	yes, 10's msec
Text messaging	no loss	elastic	yes and no

Internet Transport Protocols Services

TCP service

- *Reliable transport* between sending and receiving process
- *Flow control:* sender won't overwhelm receiver
- *Congestion control:* throttle sender when network overloaded
- *Connection-oriented:* setup required between client and server processes
- *Does not provide:* timing, minimum throughput guarantee, security

UDP service

- *Unreliable data transfer* between sending and receiving process
- *Does not provide:* reliability, flow control, congestion control, timing, throughput guarantee, security, or connection setup

Internet Applications and Transport Protocols

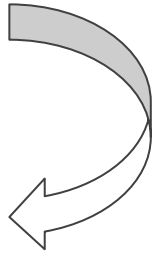
Application	App. layer protocol	Transport protocol
File transfer/download	FTP [RFC 959]	TCP
E-mail	SMTP [RFC 5321]	TCP
Web documents	HTTP [RFC 7230, 9110]	TCP
Internet telephony	SIP [RFC 3261], RTP [RFC 3550], or proprietary	TCP or UDP
Streaming audio/video	HTTP [RFC 7230], DASH	TCP
Interactive games	WOW, FPS (proprietary)	UDP or TCP


docker

Slide adapted from Jérôme Petazzoni, “Introduction to Docker and Containers”, QCON SF 2017, https://qconsf2017intro.container.training, https://www.youtube.com/playlist?list=PLBAFXs0YjviLgqTum8MkspG_8VzGl6C07

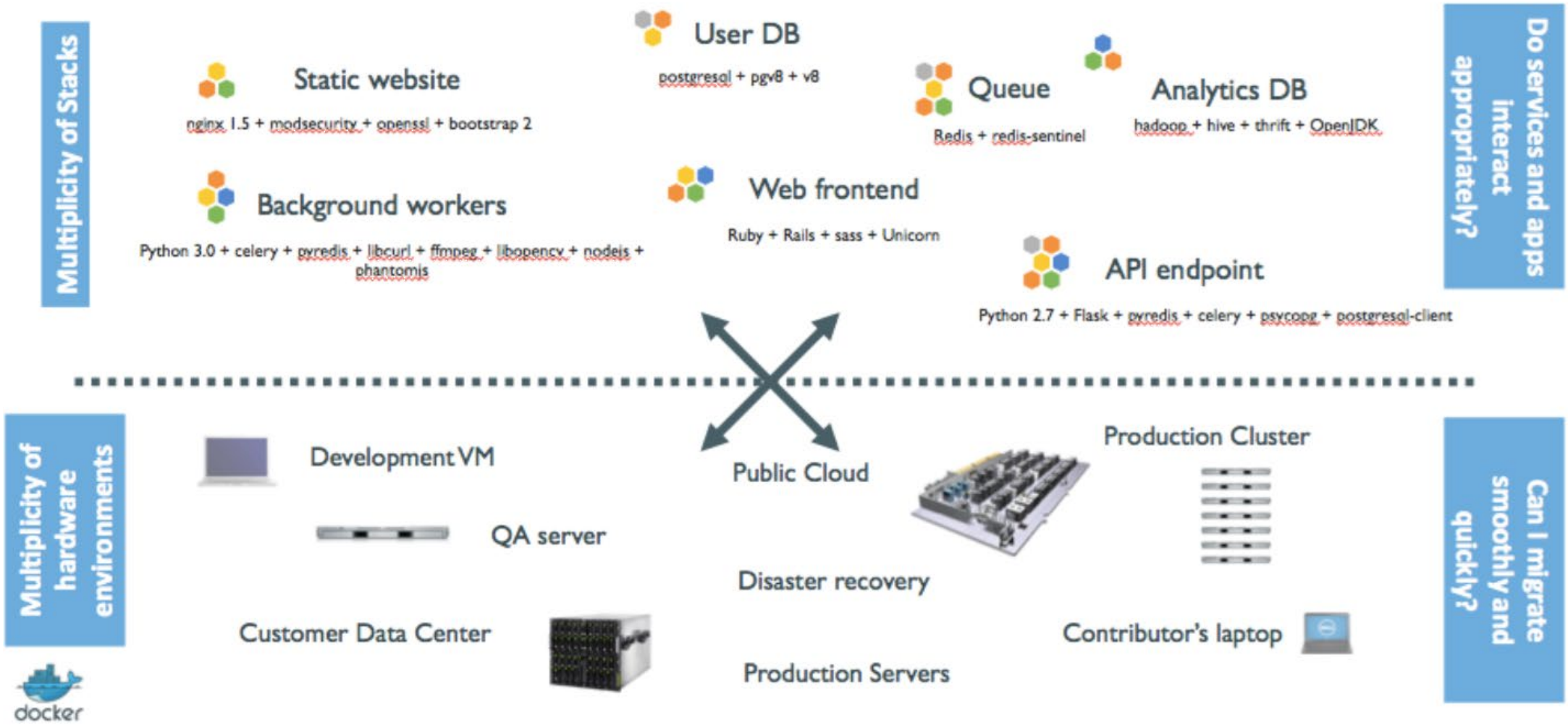
Docker – Motivation

- Software industry has changed
  - Before
    - Monolithic applications
    - Long development cycles
    - Single environment
    - Slowly scaling up
  - Now
    - Decoupled services
    - Fast, iterative improvements
    - Multiple environments
    - Quickly scaling out



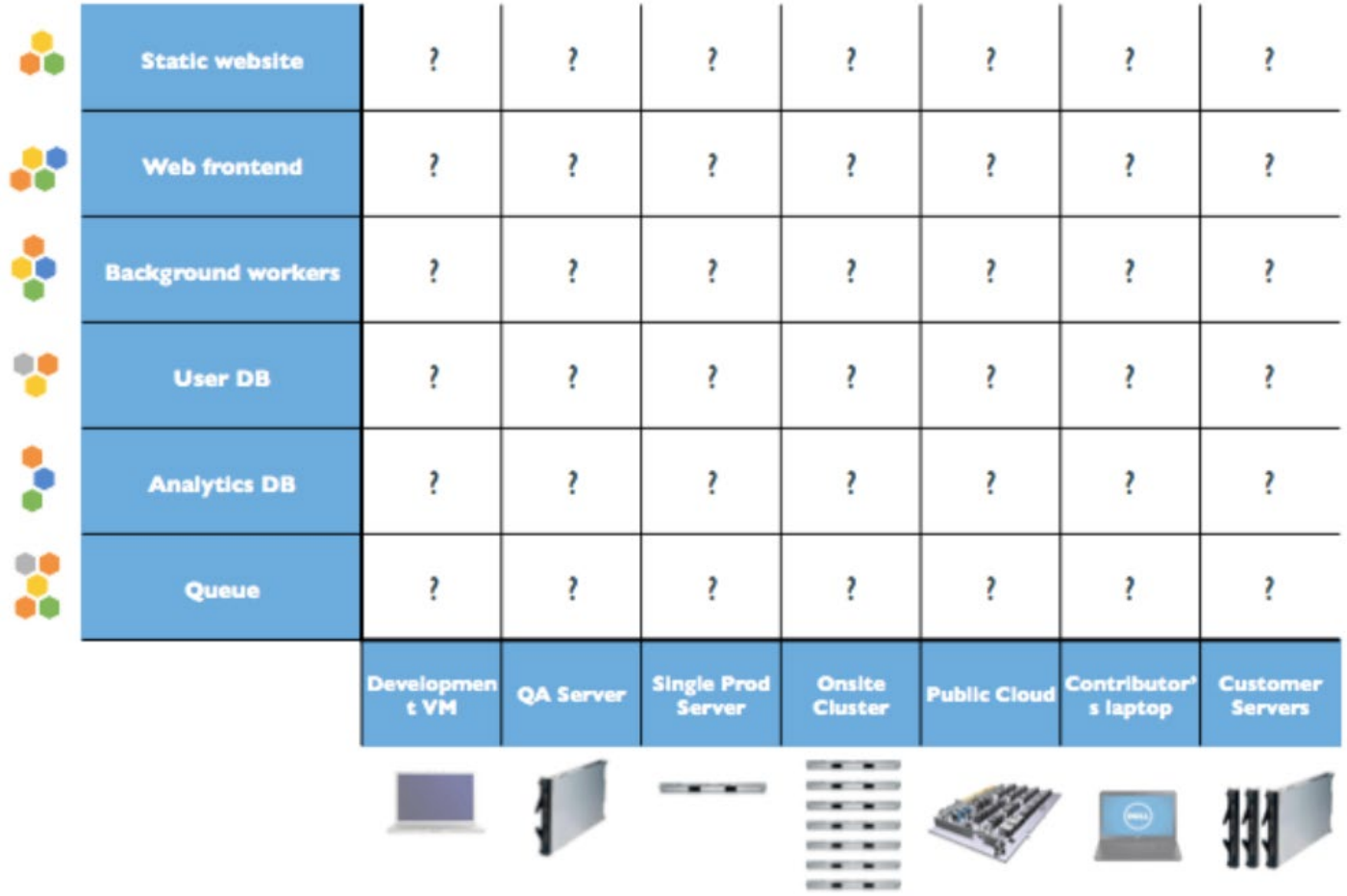
- Many different *stacks*
  - Languages
  - Frameworks
  - Databases
- Many different targets
  - Individual dev environments
  - Pre-production, QA, ..
  - Production

The Deployment Problem



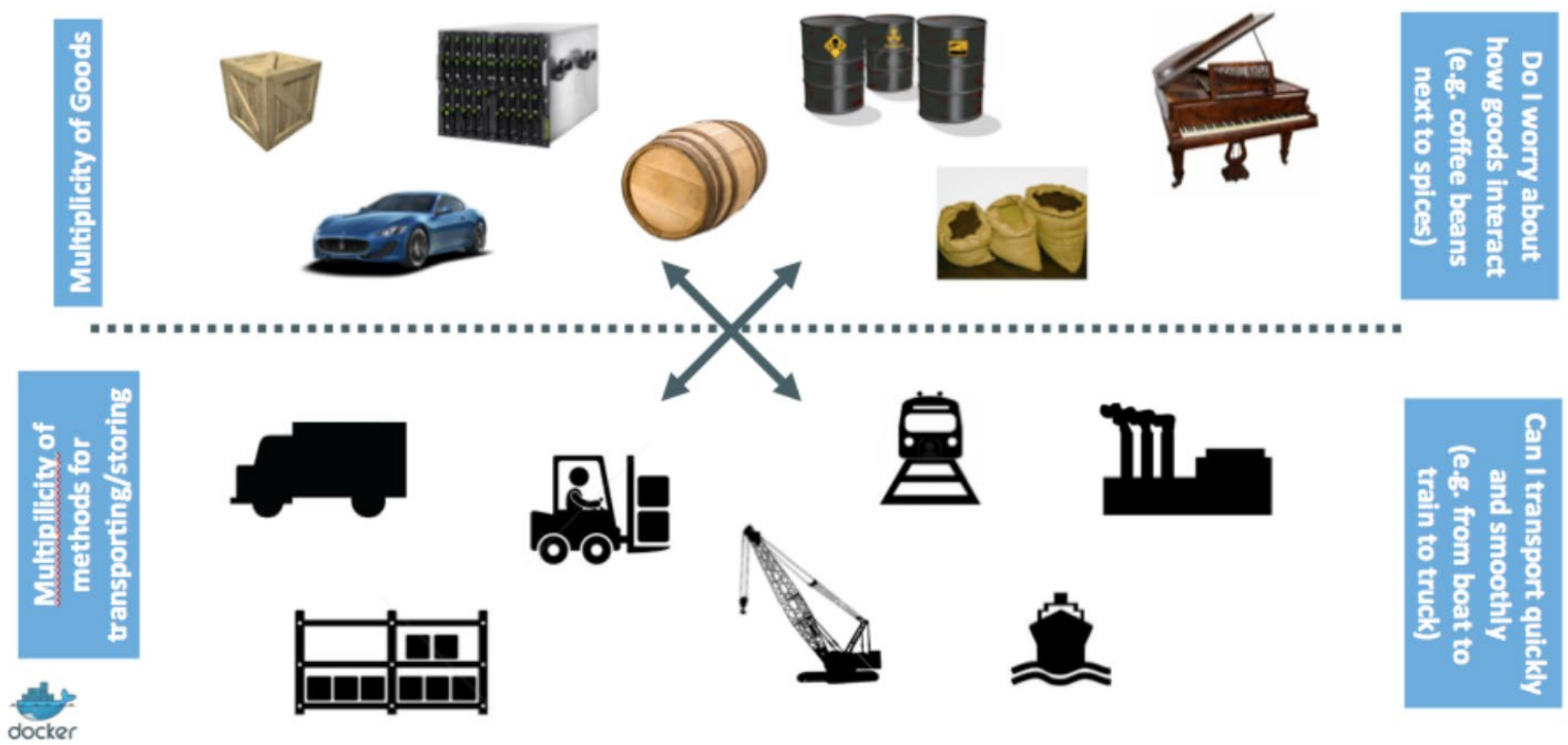
Slide adapted from Jérôme Petazzoni, "Introduction to Docker and Containers", QCON SF 2017, https://qconsf2017intro.container.training, https://www.youtube.com/playlist?list=PLBAFXs0YjviLgqTum8MkspG_8VzGl6C07

Multiplicative Complexity



Slide adapted from Jérôme Petazzoni, “Introduction to Docker and Containers”, QCON SF 2017, https://qconsf2017intro.container.training, https://www.youtube.com/playlist?list=PLBAFXs0YjviLgqTum8MkspG_8VzGl6C07

Shipping Analogy



Slide adapted from Jérôme Petazzoni, “Introduction to Docker and Containers”, QCON SF 2017, https://qconsf2017intro.container.training, https://www.youtube.com/playlist?list=PLBAFXs0YjviLgqTum8MkspG_8VzGl6C07

Shipping Analogy

Slide adapted from Jérôme Petazzoni, "Introduction to Docker and Containers", QCON SF 2017, https://qconsf2017intro.container.training, https://www.youtube.com/playlist?list=PLBAFXs0YjviLgqTum8MkspG_8VzGl6C07

New Shipping Ecosystem

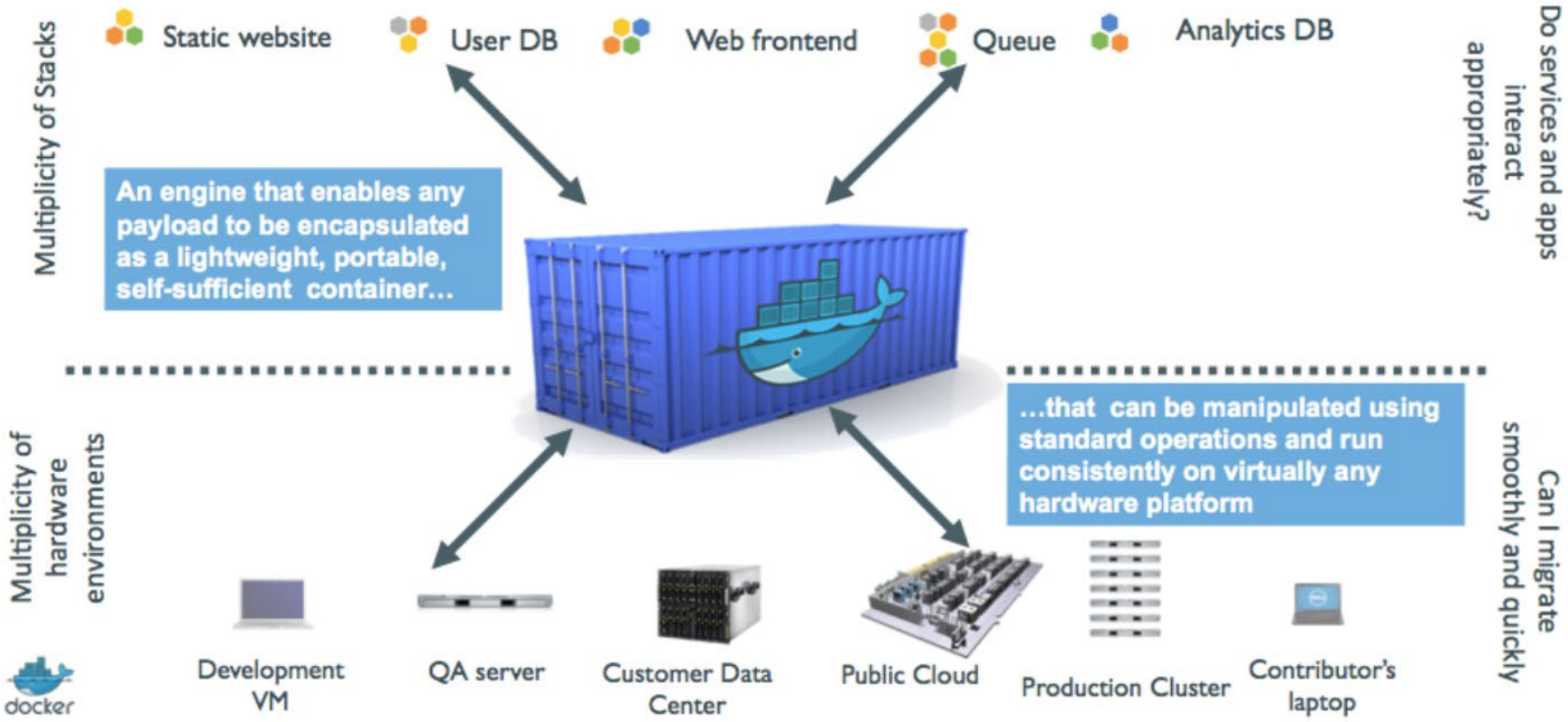
- 90% of all cargo now shipped in a standard container
- Order of magnitude reduction in cost and time to load and unload ships
- Massive reduction in losses due to theft or damage
- Huge reduction in freight cost as percent of final goods (from >25% to <3%)

→ massive globalization

- 5000 ships deliver 200M containers per year

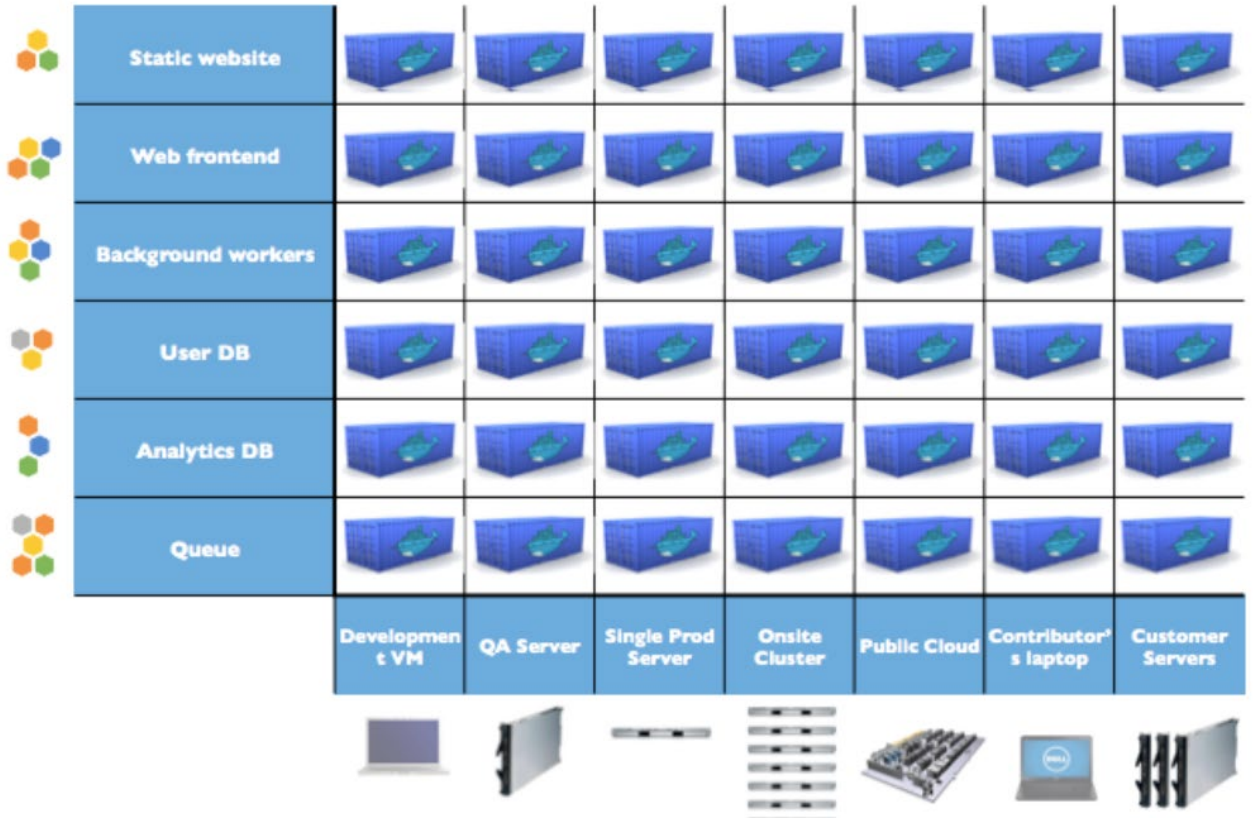
Slide adapted from Jérôme Petazzoni, "Introduction to Docker and Containers", QCON SF 2017, https://qconsf2017intro.container.training, https://www.youtube.com/playlist?list=PLBAFXs0YjviLgqTum8MkspG_8VzGl6C07

Container System for Applications



Slide adapted from Jérôme Petazzoni, "Introduction to Docker and Containers", QCON SF 2017, https://qconsf2017intro.container.training, https://www.youtube.com/playlist?list=PLBAFXs0YjviLgqTum8MkspG_8VzGl6C07

Eliminating Complexity



Slide adapted from Jérôme Petazzoni, “Introduction to Docker and Containers”, QCON SF 2017, https://qconsf2017intro.container.training, https://www.youtube.com/playlist?list=PLBAFXs0YjviLgqTum8MkspG_8VzGl6C07

Docker – Pros and Cons

+ Escape dependency / “works on my machine” issues
+ Easier onboarding using pre-made images / environments
+ Untainted tests with fresh clean state on each run
+ Standardized formats, APIs, abstractions
- Security and performance isolation can be tricky
- Additional complexity layer(s) for orchestration

Slide adapted from Jérôme Petazzoni, “Introduction to Docker and Containers”, QCON SF 2017, https://qconsf2017intro.container.training, https://www.youtube.com/playlist?list=PLBAFXs0YjviLgqTum8MkspG_8VzGl6C07

https://www.gns3.com/assets/custom/gns3/images/logo-colour.png
https://en.wikipedia.org/wiki/File:Docker_logo.svg

Lab Program Today

- Run a web server in your local GNS3 network
- Learn about
  - Hypertext Transfer Protocol (HTTP)
  - Ports, layers, and packets
- Get hands-on experience with Docker
  - Images vs containers
  - Running and attaching to containers
  - Building your own images

Next Week: Networking Lab III

- Topics: routing, DNS, services
- Goals
  - Recognize the role of routing in networking
  - Use `ip route` for managing routes
  - Retrieve basic DNS information
  - Deploy basic network services
- Preparation material & BB announcement on Monday
- ! Remember the reflections after the lab