

Week 7: Transport Layer

Assignment Project Exam Help

Internet Technologies COMP90007

<https://powcoder.com>

Add WeChat powcoder

Lecturer: Tom Drummond

Semester 2, 2021

Connection Release

- **Asymmetric** Disconnection

- Either party can issue a DISCONNECT, which results in DISCONNECT TPDU and transmission end in both directions

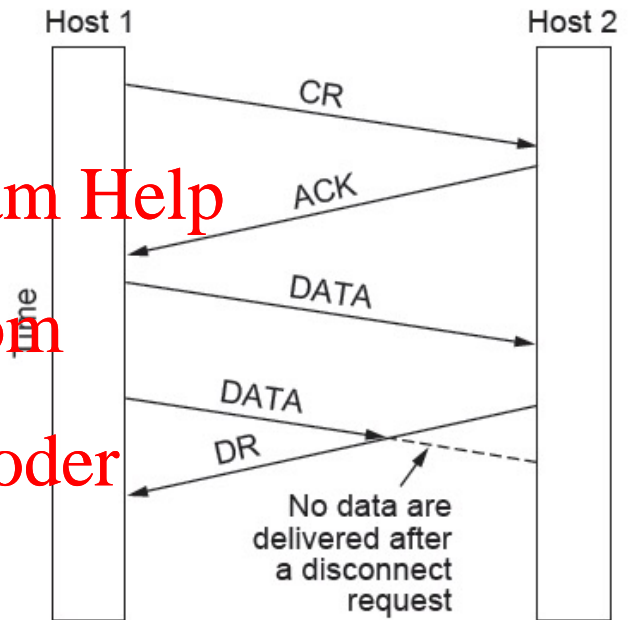
<https://powcoder.com>

- **Symmetric** Disconnection

- Both parties issue DISCONNECT, closing only one direction at a time – allows flexibility to remain in receive mode

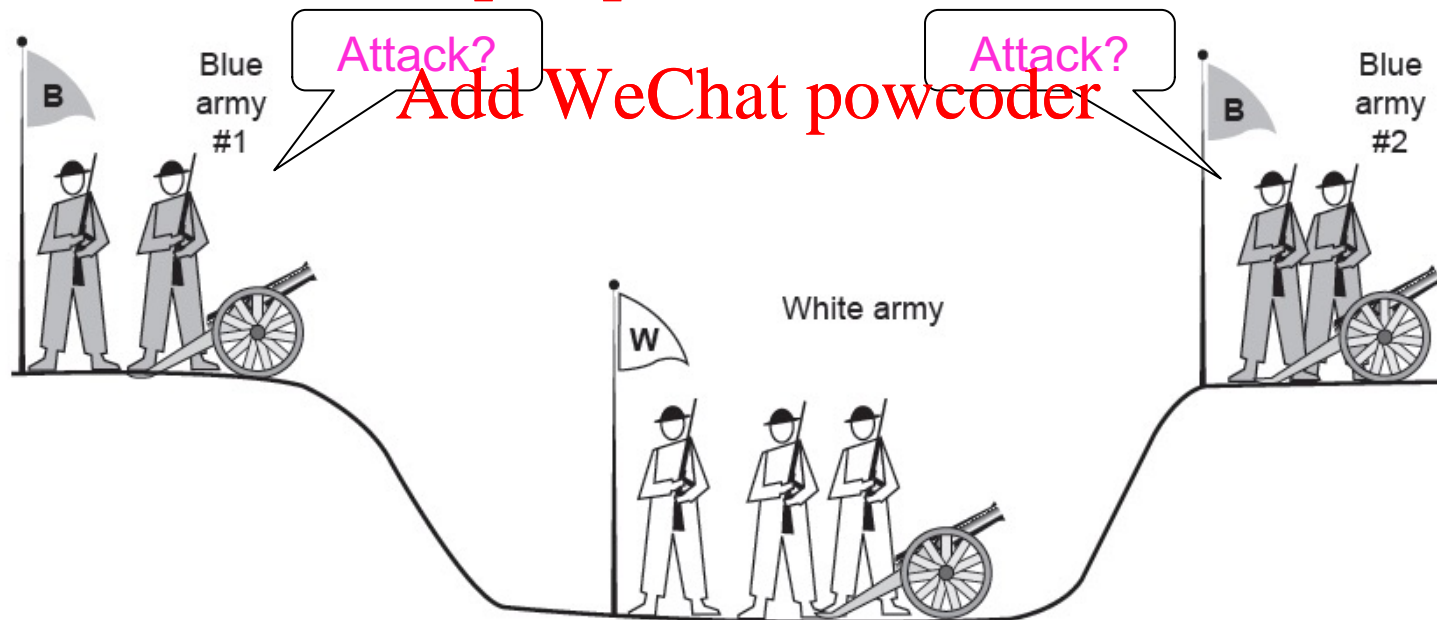
Connection Release (Cont.)

- Asymmetric vs Symmetric connection release types
- **Asymmetric** release may result in data loss hence symmetric release is more attractive
- **Symmetric** release works well where each process has a set amount of data to transmit and knows it has been sent



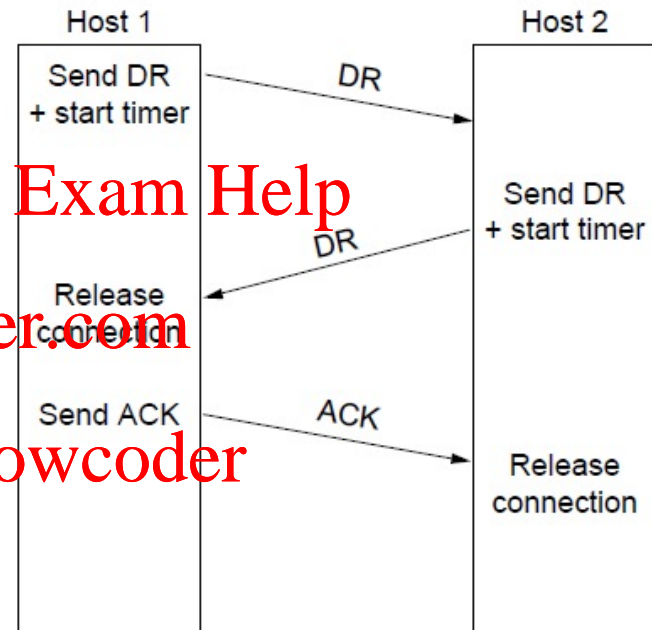
Generalizing the Connection Release Problem

- How do we decide the importance of the last message? Is it essential or not?
- No protocol exists which can resolve this ambiguity
 - Two-army problem shows pitfall of agreement



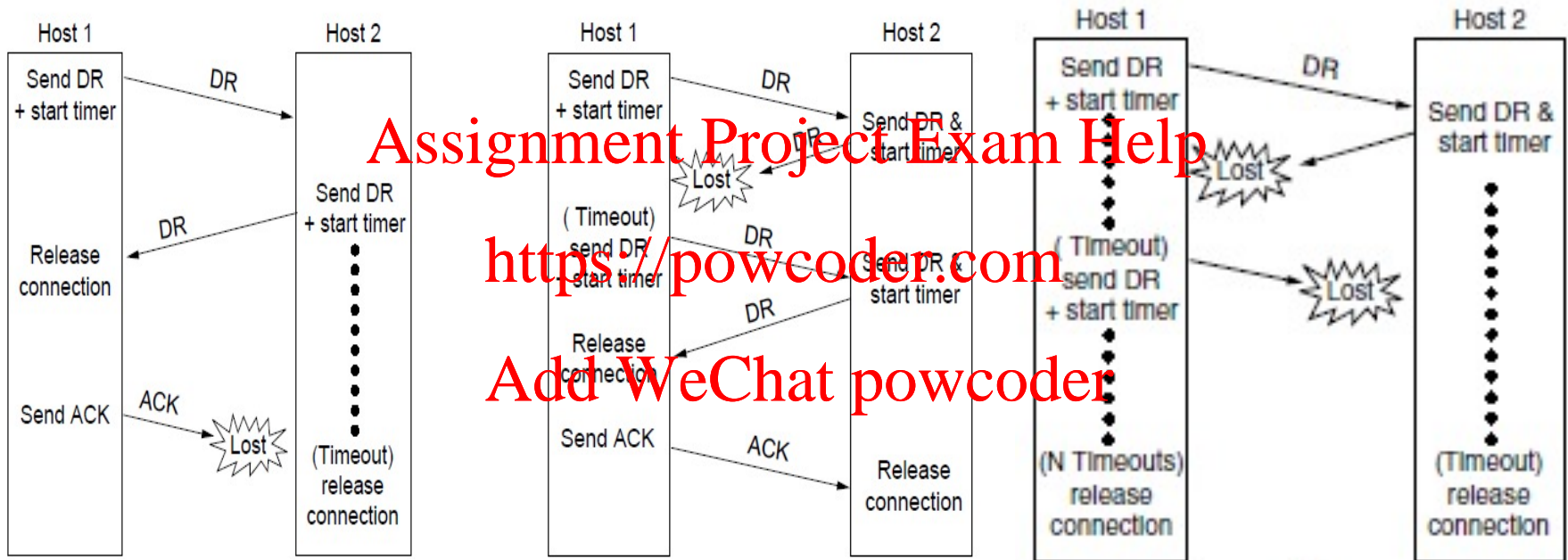
Strategies for Connection Release

- 3 way handshake
- Finite retry
- Timeouts
- Normal release sequence, initiated by transport user on Host 1
 - DR=Disconnect Request
 - Both DRs are ACKed by the other side



Connection Release (Error Cases)

- Error cases are handled with timers and retransmission



Final ACK
lost, Host 2
times out

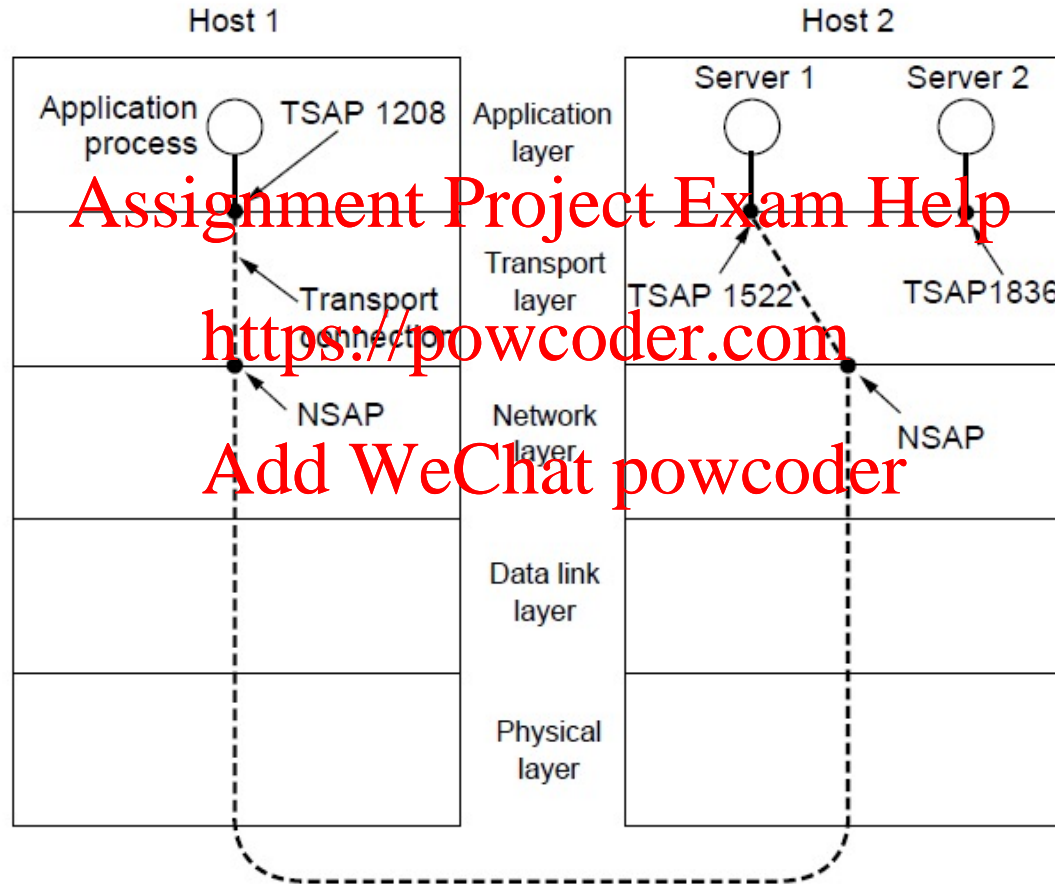
Lost DR causes retransmissions

Extreme: Many
lost DRs cause
both hosts to
timeout

Addressing

- Specification of **remote process to connect to** is required at application and transport layers
- Addressing in transport layer is typically done using **Transport Service Access Points** (TSAPs)
 - on the Internet, a TSAP is commonly referred to as a port (e.g. **port** 80)
- Addressing in the network layer is typically done using **Network Service Access Points** (NSAPs)
 - on the Internet, the concept of an NSAP is commonly interpreted as simply an **IP address**

TSAPs, NSAPs and Transport Layer Connections Illustrated



Types of TSAP Allocation

1. Static

- ❑ Well known services have standard allocated TSAPs/ports, which are embedded in OS

2. Directory Assistance – Port-mapper

- ❑ A new service must register itself with the portmapper, giving both its service name and TSAP

3. Mediated

- ❑ A process server intercepts inbound connections and spawns requested server and attaches inbound connection
- ❑ cf. Unix /etc/(x)inetd

Programming using Sockets

- Sockets widely used for interconnections
 - “Berkeley” sockets are predominant in internet applications
 - Notion of “sockets” as transport endpoints
 - Like the simple set plus SOCKET, BIND, and ACCEPT

Assignment Project Exam Help

Primitive	Meaning
SOCKET	Create a new communication endpoint
BIND	Associate a local address with a socket
LISTEN	Announce willingness to accept connections; give queue size
ACCEPT	Passively establish an incoming connection
CONNECT	Actively attempt to establish a connection
SEND	Send some data over the connection
RECEIVE	Receive some data from the connection
CLOSE	Release the connection

Recall Example Pseudo Code

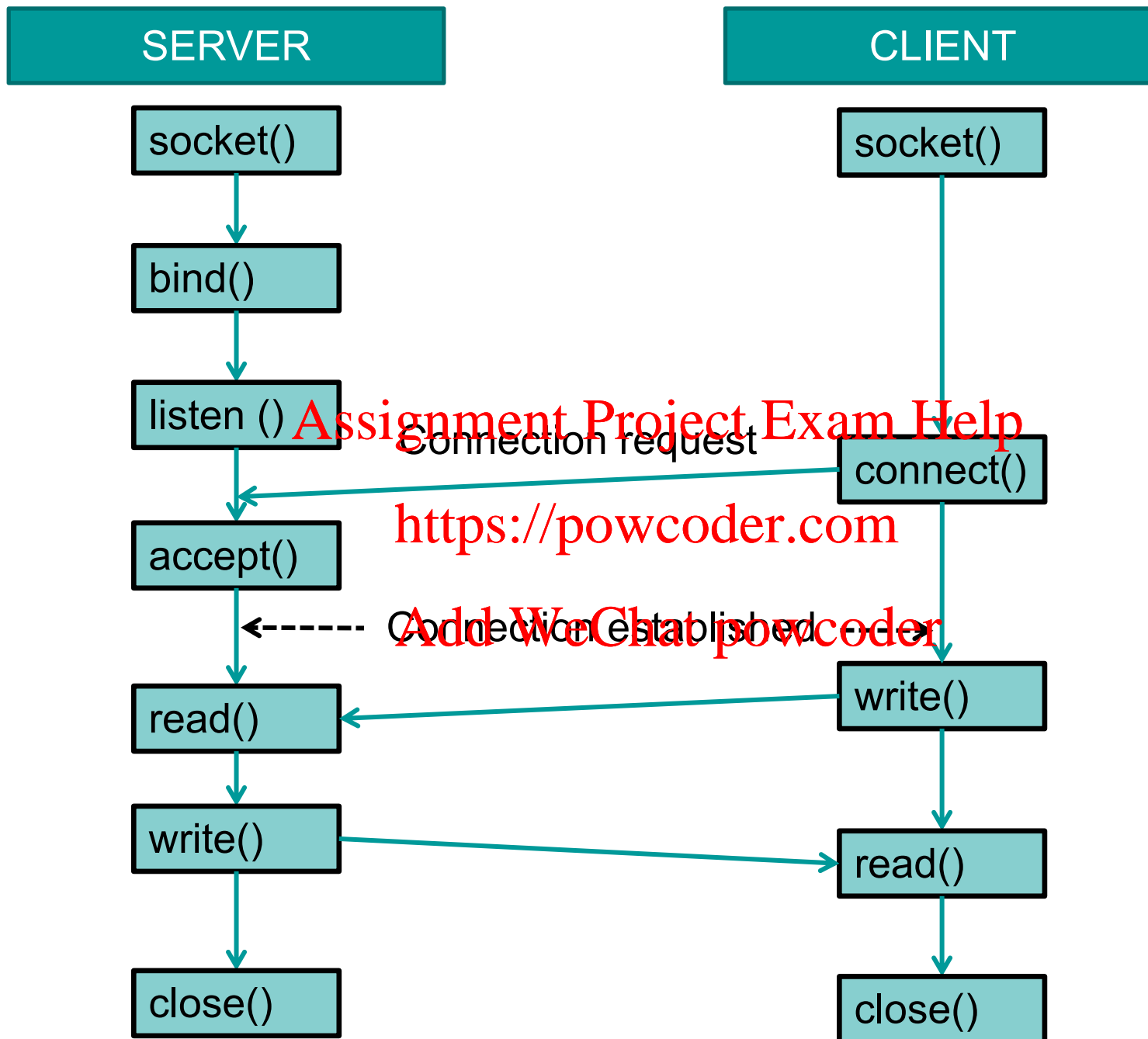
```
Socket A_Socket = createSocket("TCP");
```

```
connect(A_Socket, 128.255.16.0, 80);
```

```
send(A_socket, "My first message!");
```

```
disconnect(A_socket);
```

***... there is also a server component for this client
that runs on another host...***



Let's Look at the Code from the book (in a specific language)

Example from the book has more details but the essence is the same... This is the case in most languages...

Assignment Project Exam Help

```
s = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP);  
if (s < 0) fatal("socket");  
memset(&channel, 0, sizeof(channel));  
channel.sin_family= AF_INET;  
memcpy(&channel.sin_addr.s_addr, h->h_addr, h->h_length);  
channel.sin_port= htons(SERVER_PORT);
```

```
c = connect(s, (struct sockaddr *) &channel, sizeof(channel));
```

<https://powcoder.com>

Add WeChat powcoder

Socket Example – Server Side

Server code. . .

```
memset(&channel, 0, sizeof(channel));  
channel.sin_family = AF_INET;  
channel.sin_addr.s_addr = htonl(INADDR_ANY);  
channel.sin_port = htons(SERVER_PORT);
```

Assignment Project Exam Help

<https://powcoder.com>

```
s = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);  
if (s < 0) fatal("socket failed");  
setsockopt(s, SOL_SOCKET, SO_REUSEADDR, (char *) &on, sizeof(on));
```

Add WeChat powcoder

```
b = bind(s, (struct sockaddr *) &channel, sizeof(channel));  
if (b < 0) fatal("bind failed");
```

**Assign
address**

```
l = listen(s, QUEUE_SIZE);  
if (l < 0) fatal("listen failed");
```

**Prepare for
incoming
connections**

. . .

Server Code Contd

```
while (1) {  
    sa = accept(s, 0, 0);  
    if (sa < 0) fatal("accept failed");  
    read(sa, buf, BUF_SIZE);  
    /* Get and return the file. */  
    fd = open(buf, O_RDONLY);  
    if (fd < 0) fatal("open failed");  
    .....
```

Block waiting for the next connection

Read (receive) request

The server can also create a new thread to handle the connection on the new socket and go back to waiting for the next connection on the original socket...

An Example on Multi-Threading

```
ServerSocket serverSocket = new ServerSocket([parameters]);
```

```
While (true) {  
    Socket socket = serverSocket.accept();  
    MultiThreadMyServer server = new MultiThreadMyServer();  
    server.setMyService([some more parameters]);  
    server.setSocket(socket);  
    new Thread(server).start();  
    ....  
}
```

- *(Code from OO Programming with Java; Chp. 14)*

Looking under the hood for Transport Layer Services...

- The **most basic** is actually connectionless:
 - Called: User Datagram Protocol (UDP)
 - Does not add much to the Network Layer functionality
 - TCP we just does the real-deal for this layer, *reliability*...
 - For UDP: Just remove connection primitives to use it in a program
- **UDP good for?:**
 - It is used for apps like video streaming/gaming regularly
- **The reliability issue is left to?:**
 - the application layer... retransmission decisions as well as congestion control

New Code: UDP Client...

```
public static void main(String args[]) {  
    ....  
    Assignment Project Exam Help  
DatagramSocket mySocket = new  
        https://powcoder.com  
        DatagramSocket(),  
    mySocket.Add WeChat powdersend([data, address, etc  
        parameters]);  
    ...  
}
```

Server Side: UDP Example Contd

```
public static void main(String args[]) {  
    ....  
    DatagramSocket server = new  
        DatagramSocket(port);  
    while (true) {  
        server.receive([parameters]);  
        ...  
    }  
}
```

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder