

Virtual Private Networks

Instructor: Khaled Diab

Outline

- What is VPN?
- Overview of TLS VPN
- TLS VPN Details
- Building a VPN

What is VPN?

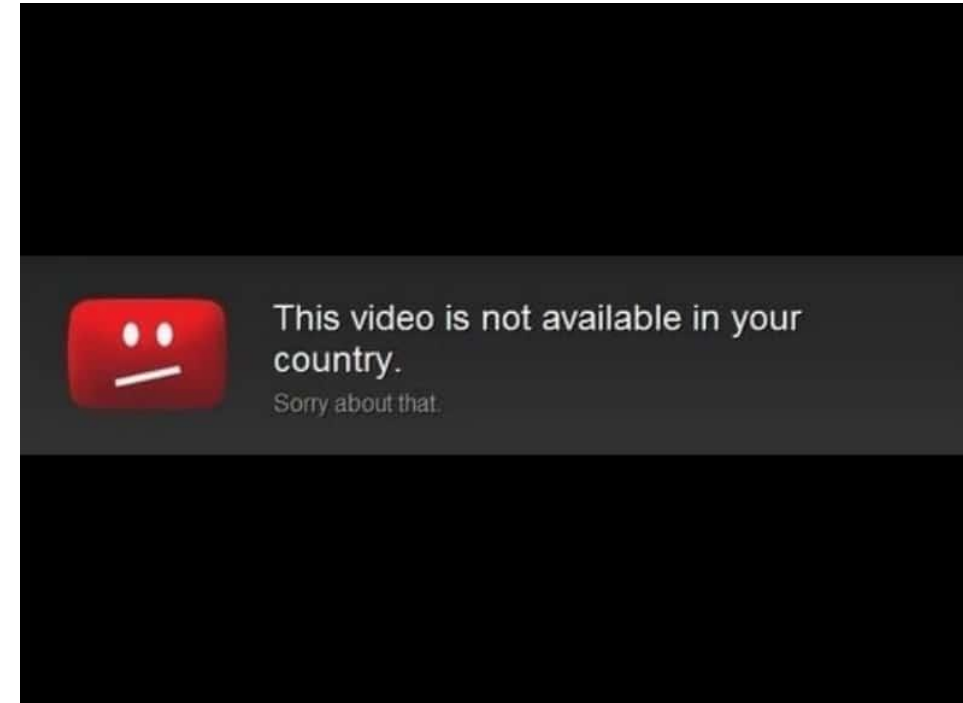
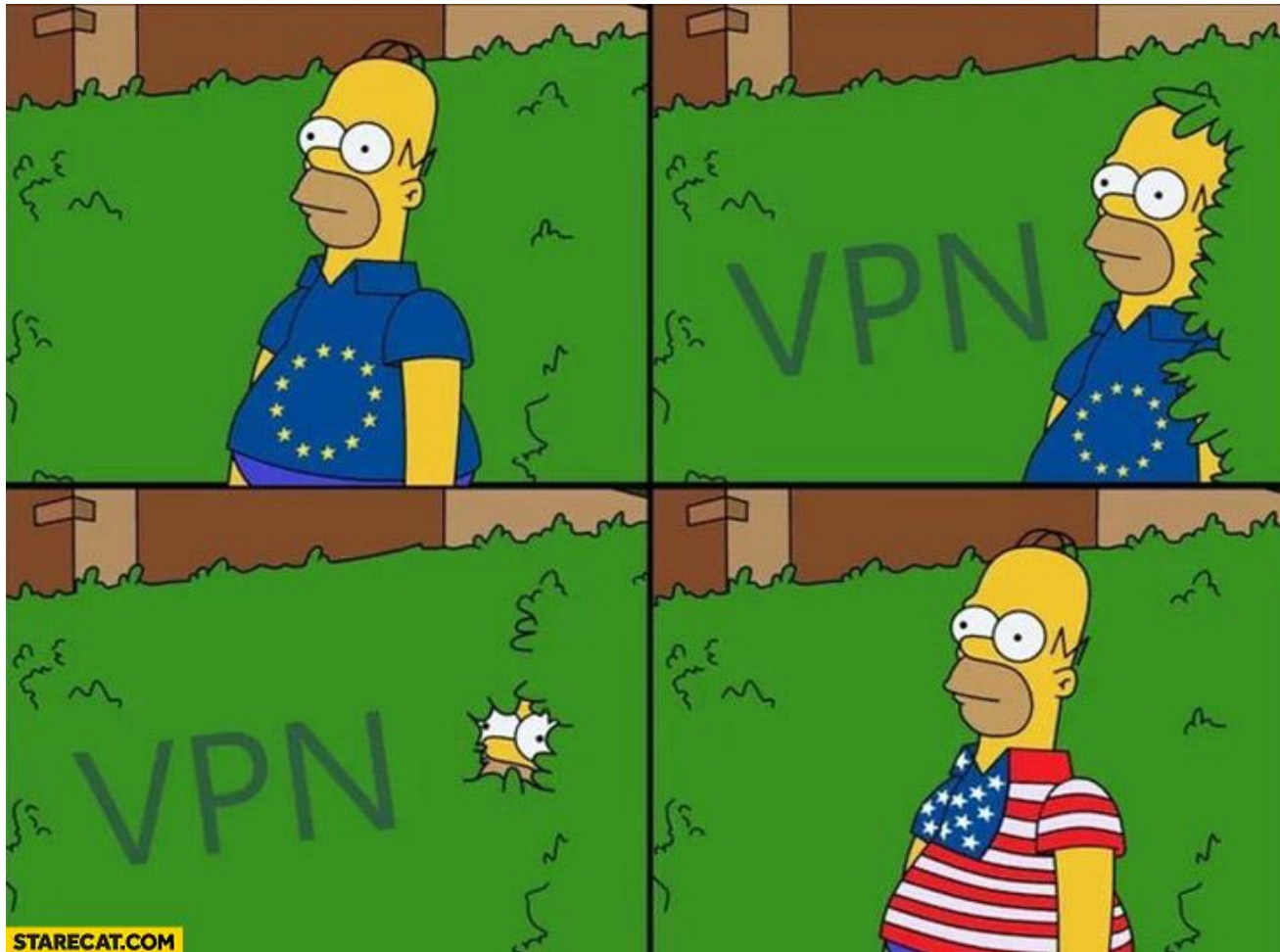
VPN: Use Cases

Protect yourself from hackers in untrustworthy Wi-Fi hotspots



VPN: Use Cases

Bypass geographic restrictions



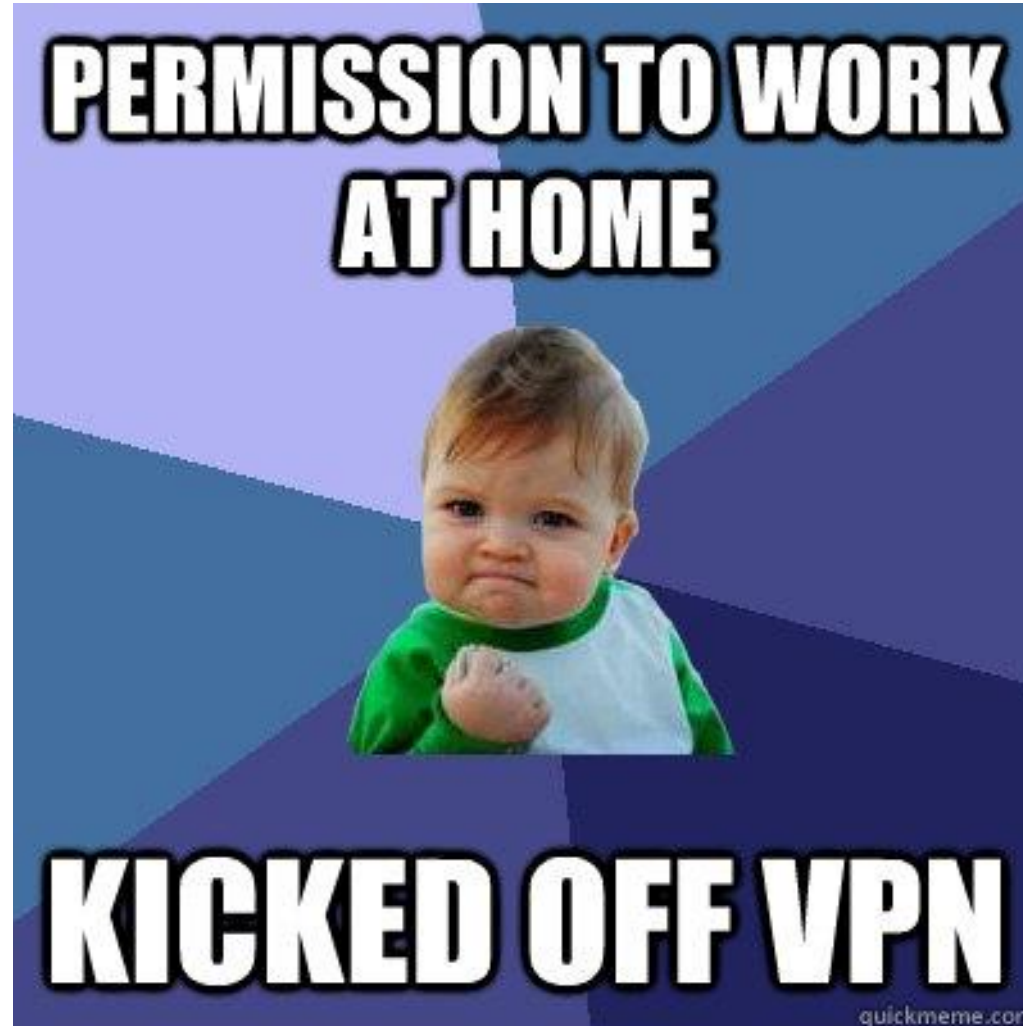
VPN: Use Cases

Bypassing egress filtering at firewalls



VPN: Use Cases

Extend private network (e.g., enterprise, home, etc.)



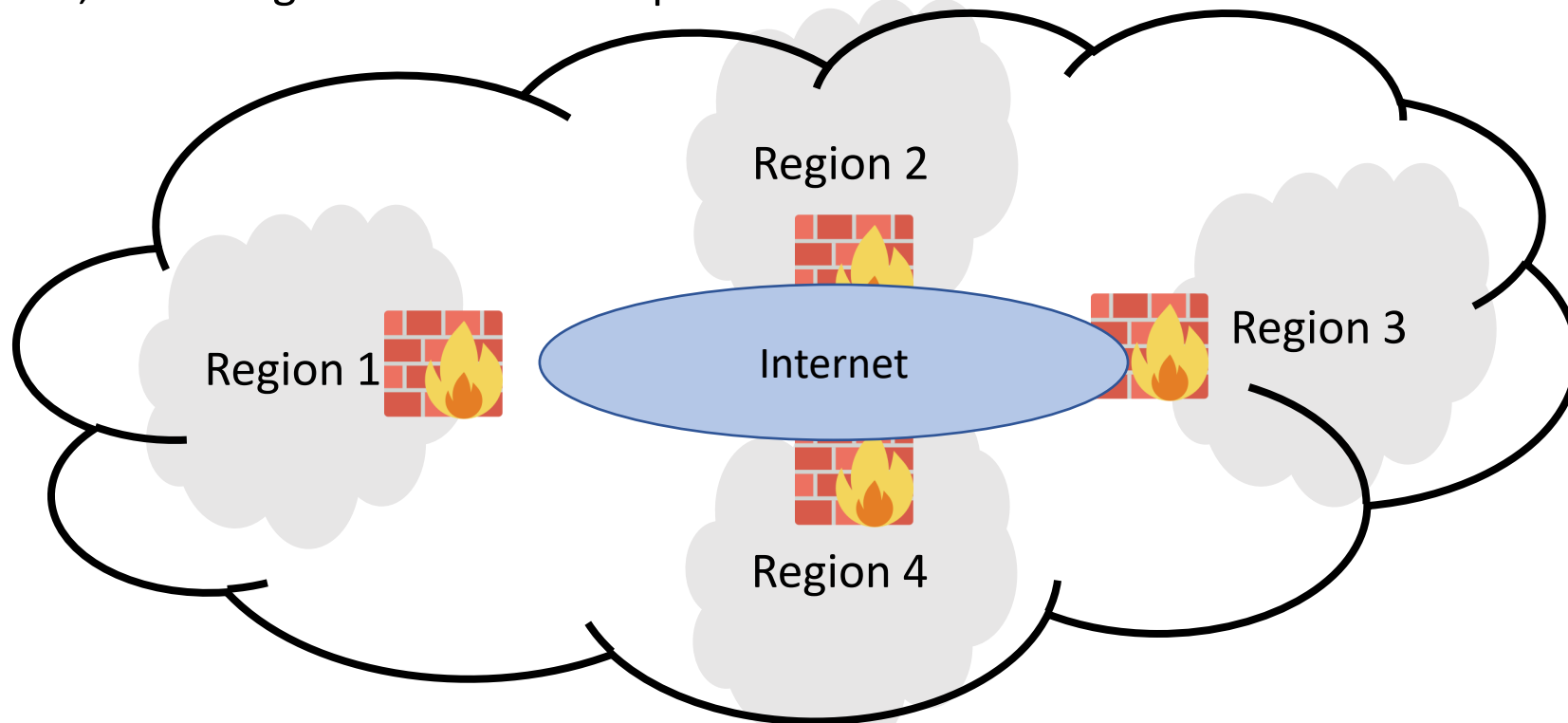
Many VPN Options



And other commercial VPN software

Motivation

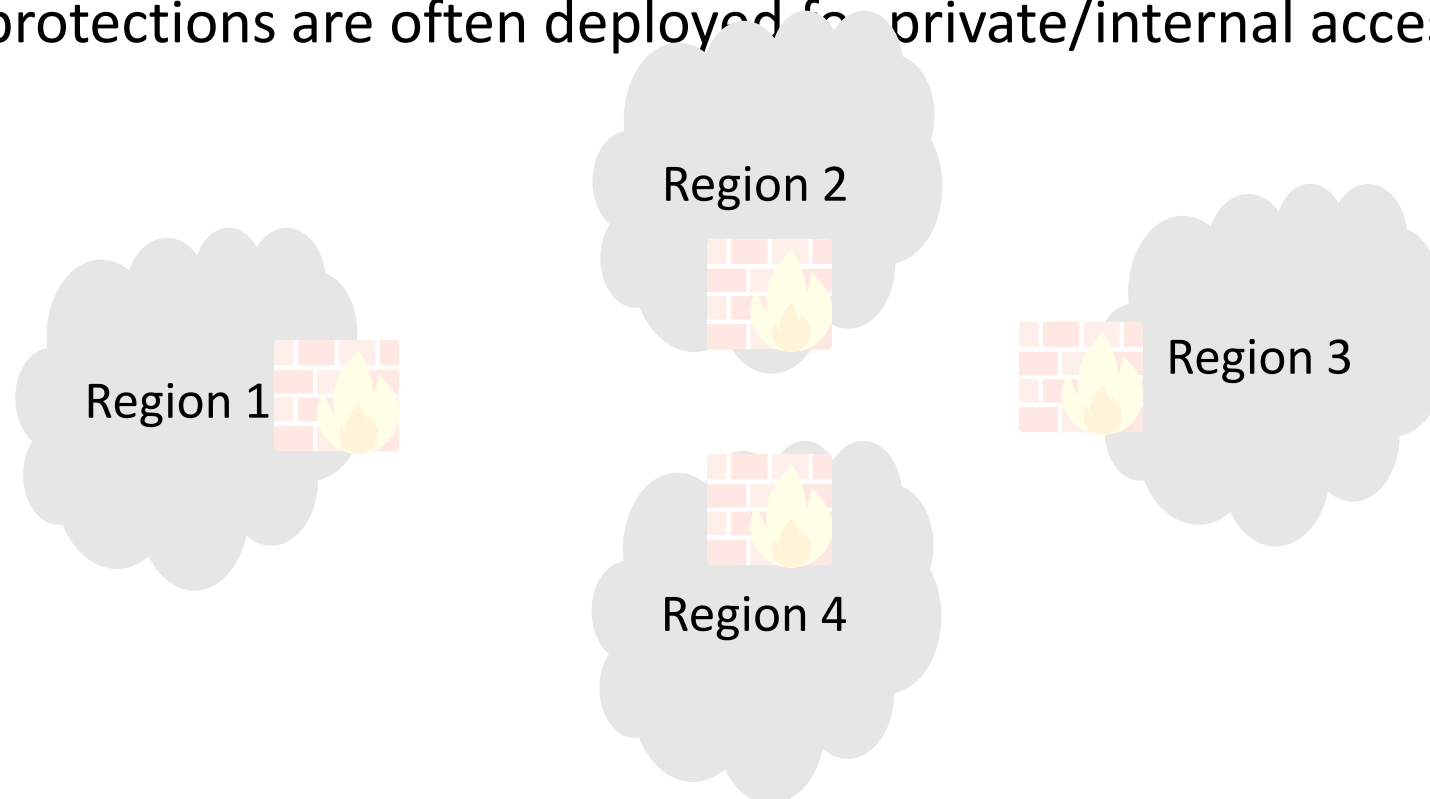
- As enterprises grow:
 - Their private networks deployed to different geographical regions
 - Employees need while travelling or at home
 - i.e., accessing resources inside private networks



Should act as a single **private** network

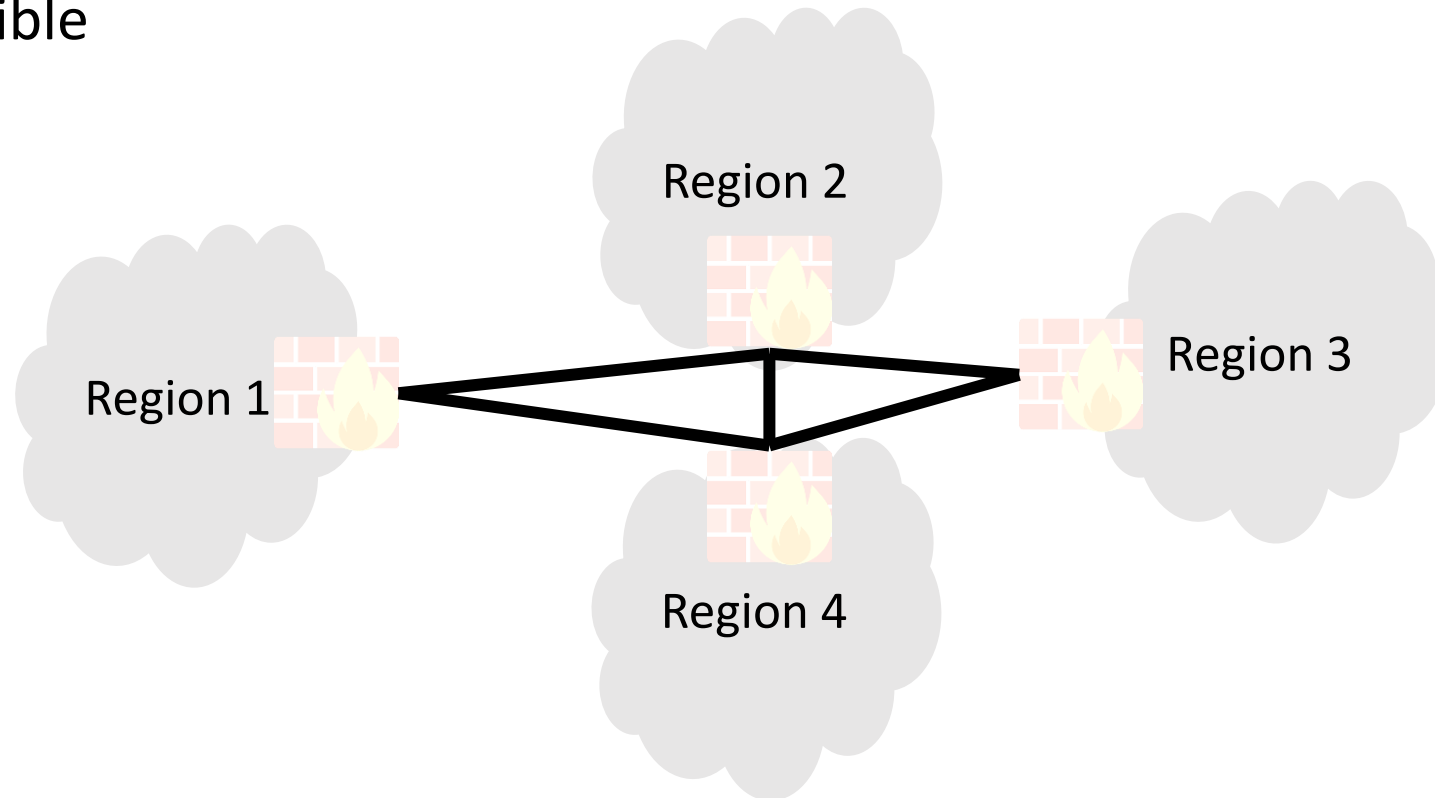
Option #1

- Relax firewall policies to allow external networks/users
- Drawbacks:
 - Increasing the attack surface and risks
 - Simple protections are often deployed for private/internal accesses



Option #2

- Lease/own dedicated links between sites
- Drawbacks:
 - Expensive
 - Not flexible



Other Options?

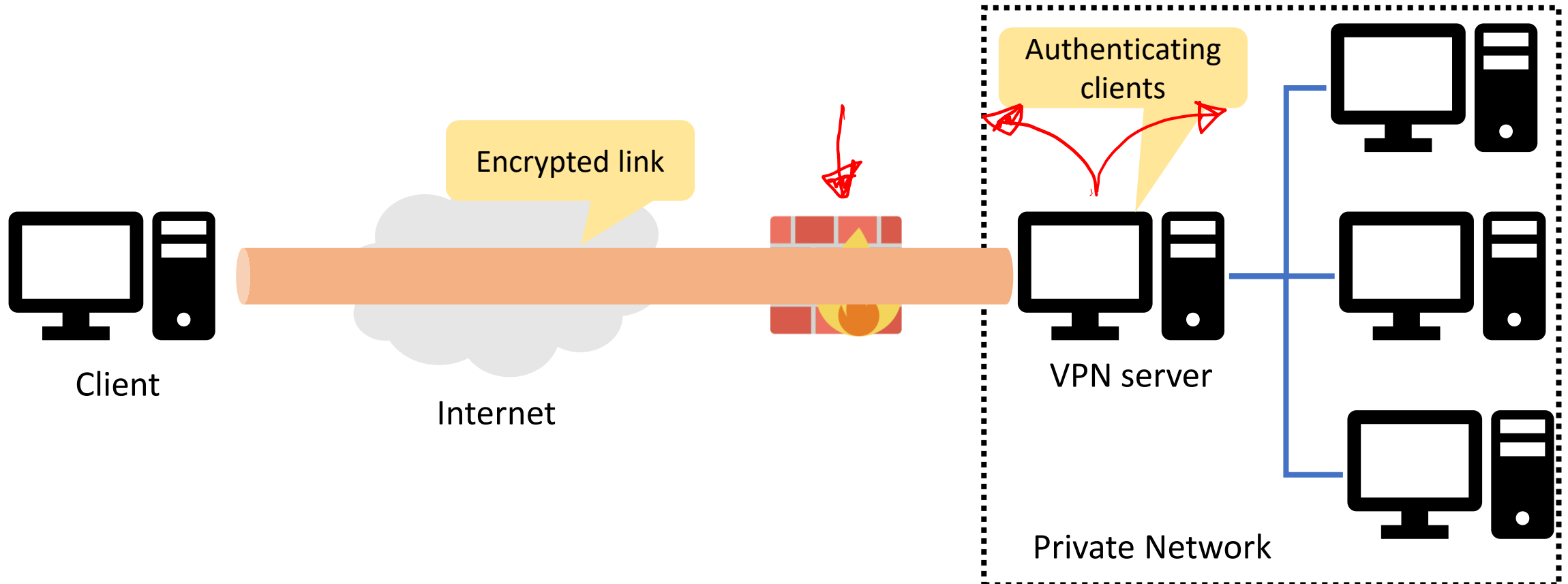
- Instead, we need to think:
 - of what protection guarantees are made by private networks
 - whether these guarantees are achieved if a host is outside the private network

Guarantees of a Private Network

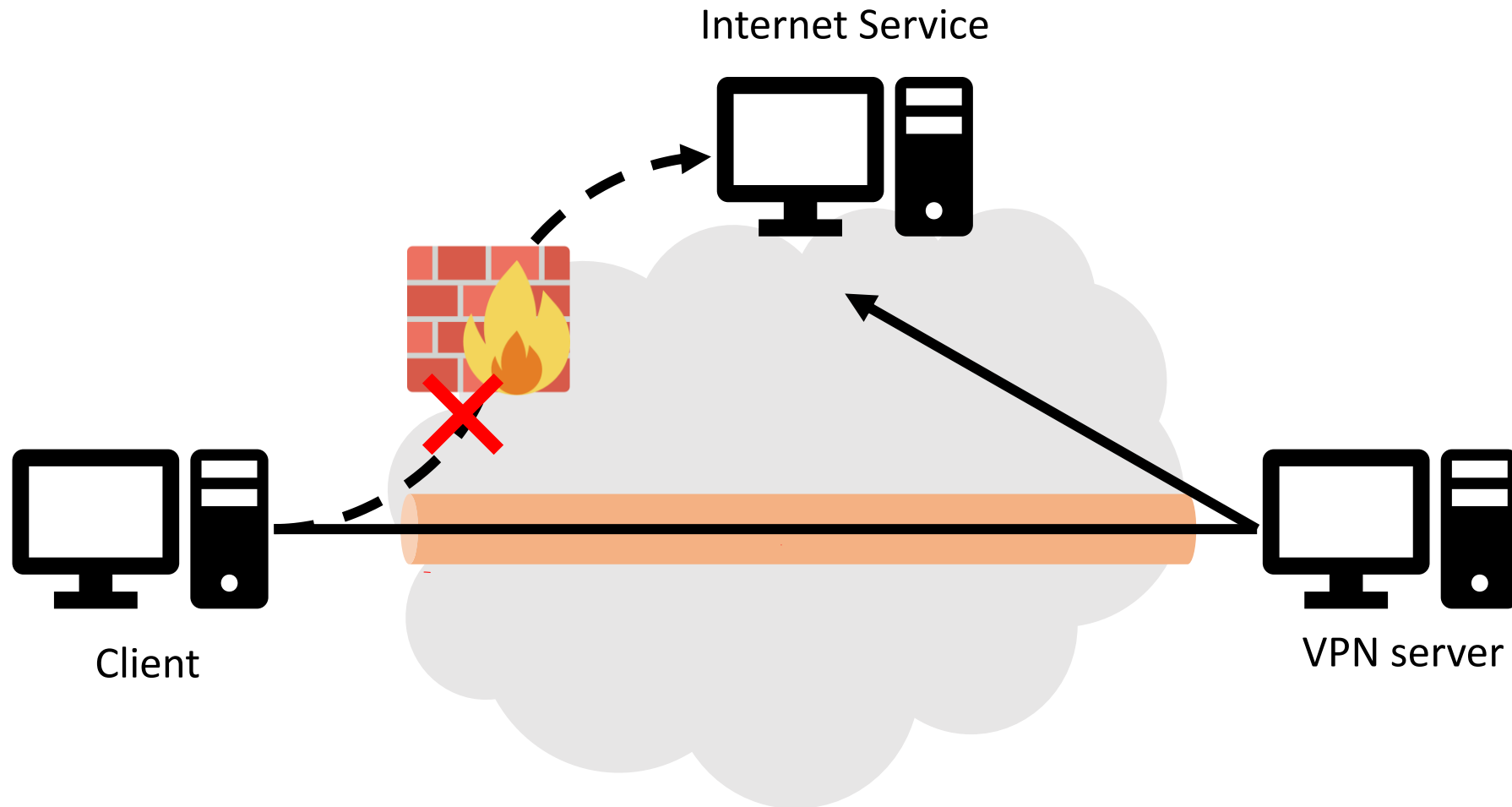
- User authenticated
 - Verified identity
- Content protected
 - Content of communication cannot be seen from the outside
- Integrity preserved
 - Outsiders cannot inject fake data

What is a VPN?

- A private network consisting of hosts from both inside and outside
 - Virtual → because this network isn't **physically** private

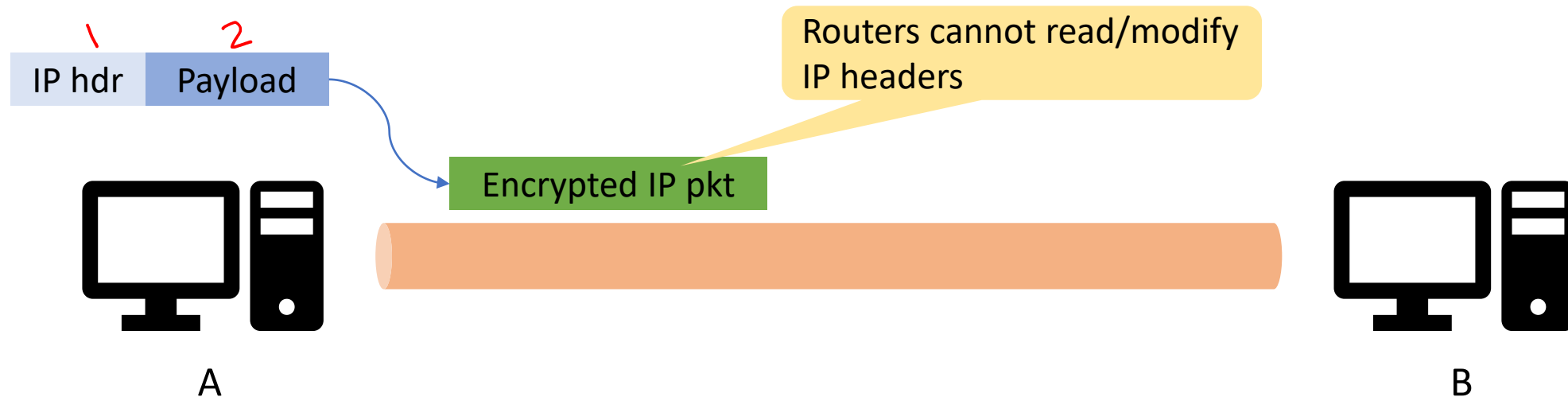


What is a VPN?



What is a VPN?

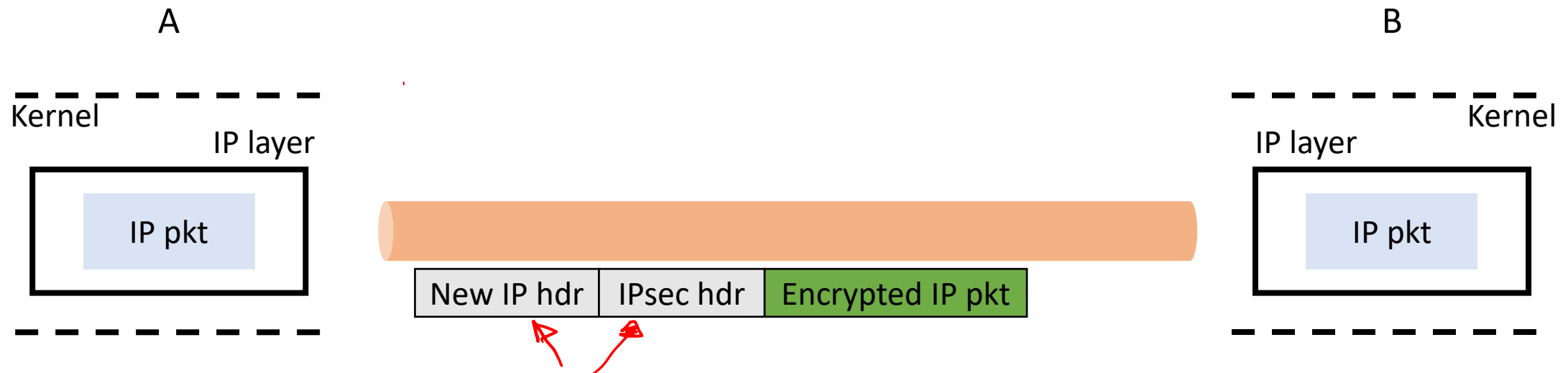
- Regardless of whether an application encrypts its data
→ IP packets need to be encrypted (including headers)



- Two techniques to implement IP tunneling:
 - TIPsec Tunneling (using IPsec Tunnel Mode)
 - LS Tunneling

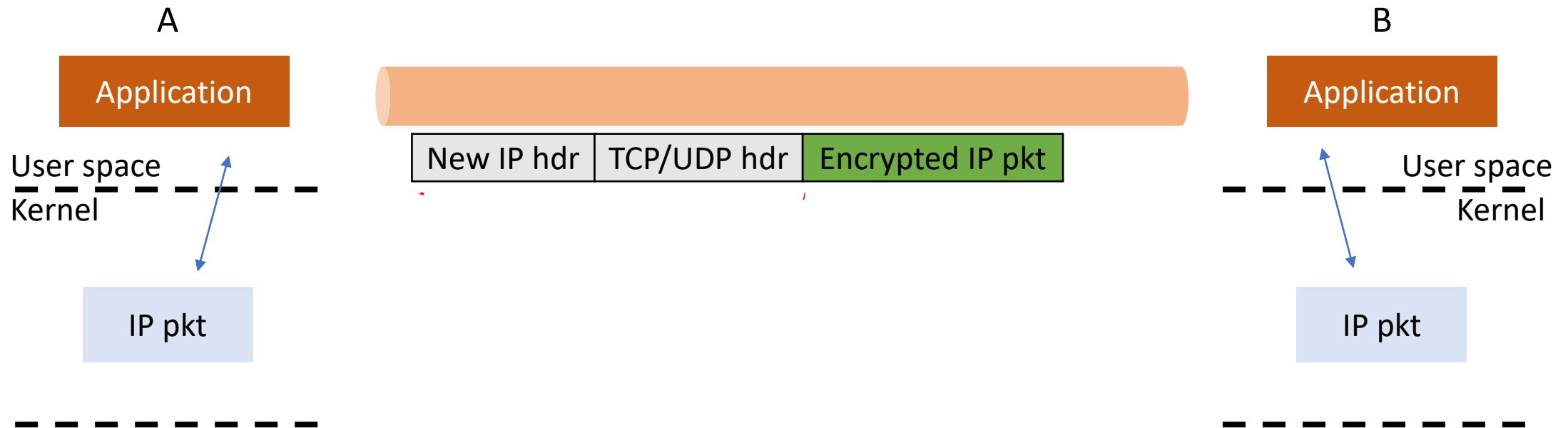
IPsec Tunneling: Tunnel Mode

- Encrypts the whole IP packet
- Encapsulates the encrypted IP packet with a new IP packet
- Operates at the kernel space



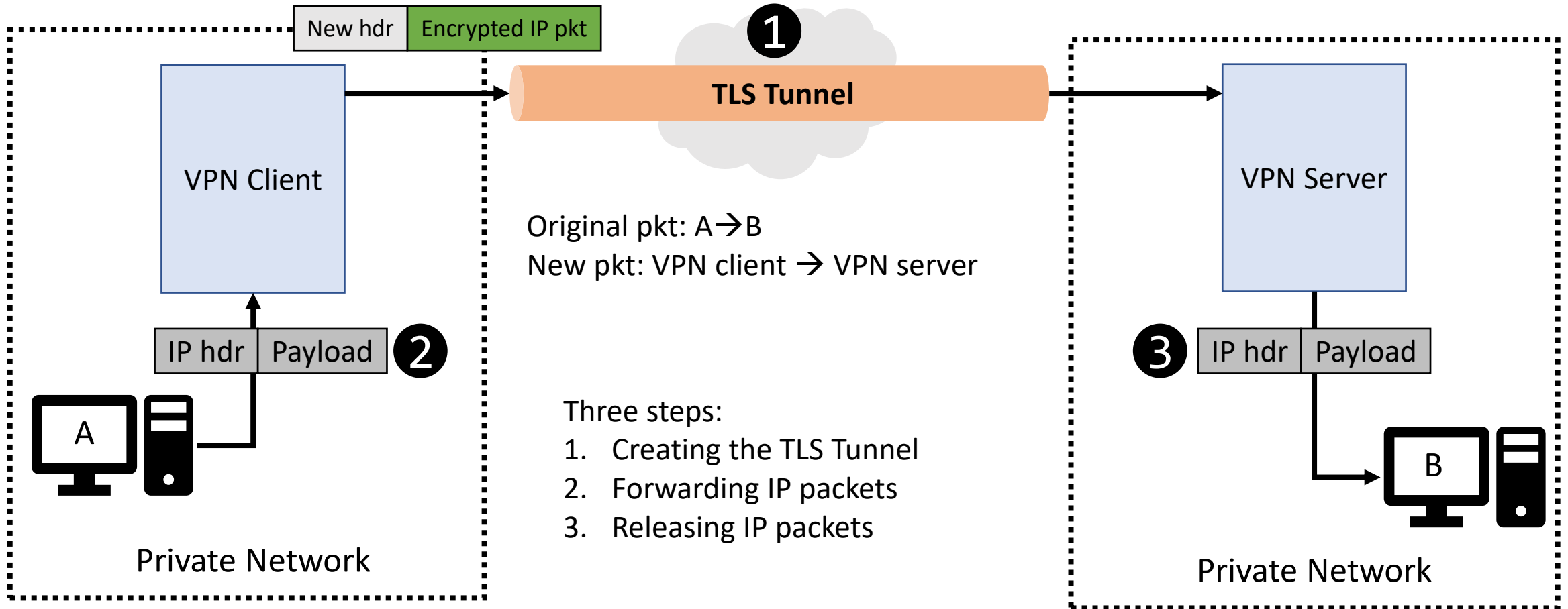
TLS Tunneling

- VPN-bound IP packets are handled by an application
- Encrypted using TLS protocol
- Operates at the application layer



Overview of TLS VPN

TLS-based VPN



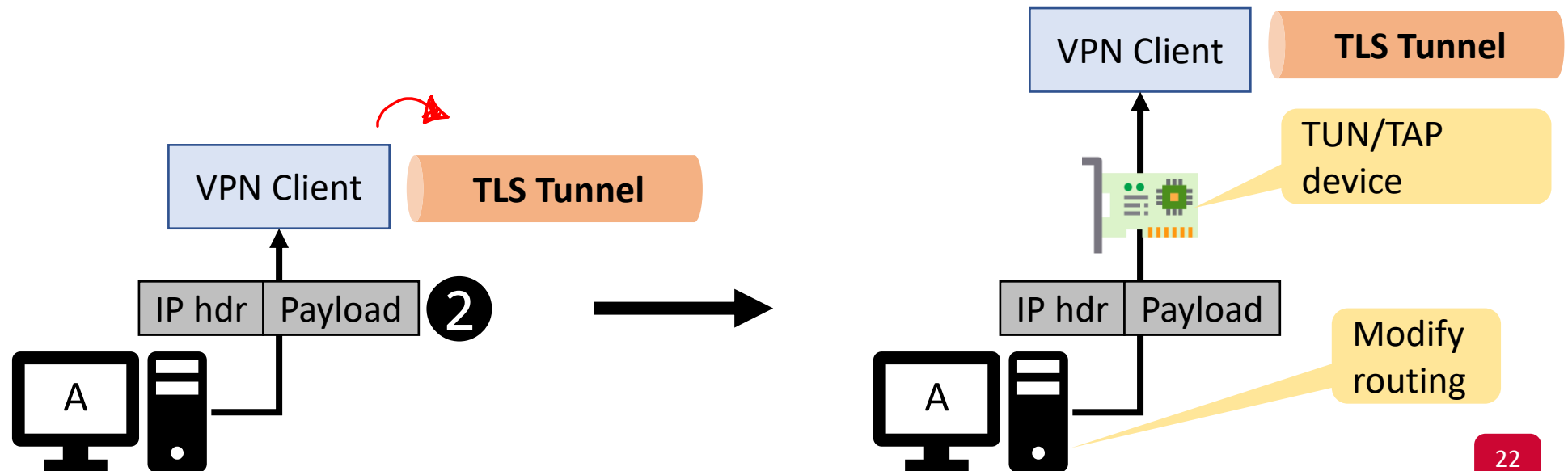
Creating a Tunnel



- This is a TLS channel
- It is built on top of a transport-layer protocol
- Before creating a channel, mutual authentication is needed:
 - Server authenticates client: e.g., using passwords
 - Client authenticates server: e.g., using certificates

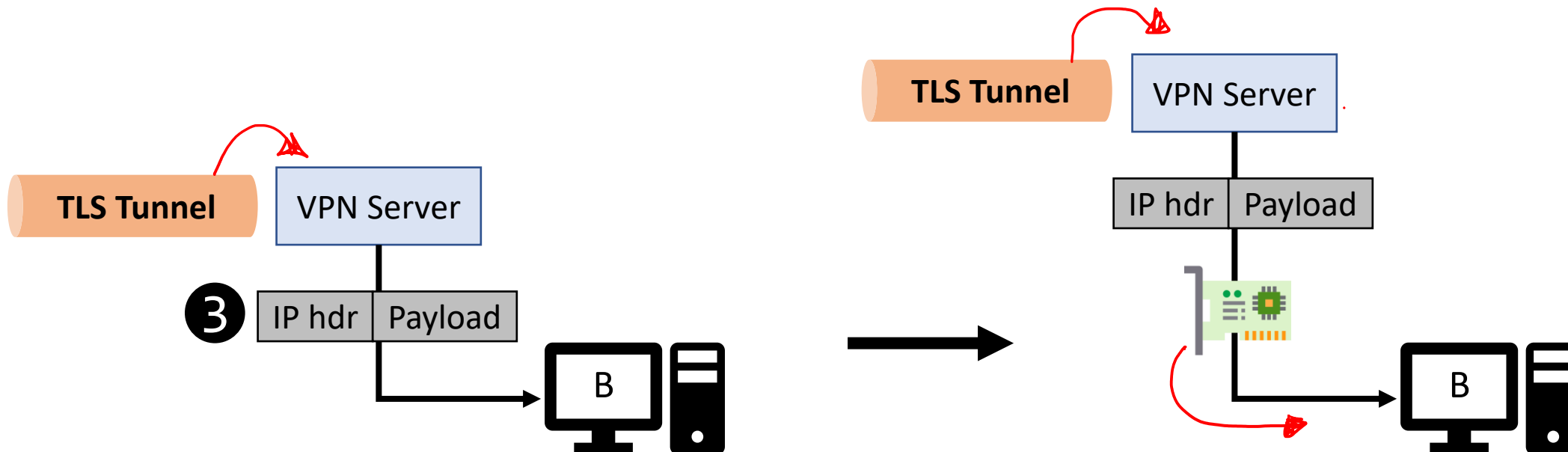
Forwarding IP Packets

- VPN Client needs to receive the whole IP pkt to encrypt it
 - The kernel removes these headers
- How can an application receive the whole pkt?
 - Create a TUN/TAP device
 - Modify routing table: All VPN-bound traffic goes to the new device



Releasing IP Packets

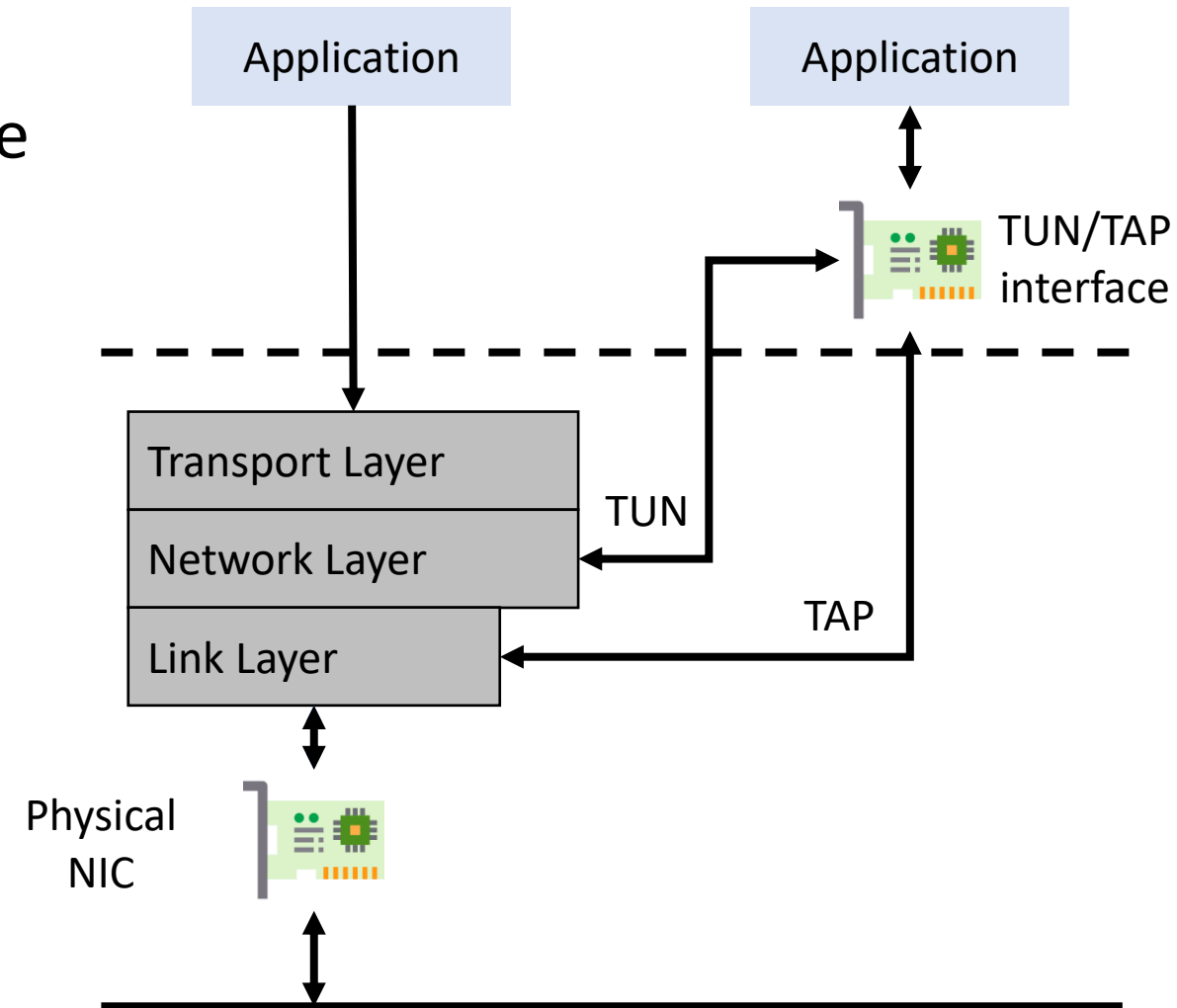
- VPN Server needs to release the original IP pkt after decrypting it
- How can an application send the whole pkt to the kernel?
 - Same idea as before



TLS VPN Details

Virtual Network Interfaces

- A virtual interface is a virtualized representation of a network interface
- TUN interface:
 - Works at the IP layer
 - Point-to-point is the default
 - Sending a pkt to a TUN interface will result in the pkt being delivered to the user-space program
- TAP interface:
 - Works at the Ethernet layer



Creating a TUN Interface

```
int createTunDevice()
{
    int tunfd;
    struct ifreq ifr;
    memset(&ifr, 0, sizeof(ifr));

    ifr.ifr_flags = IFF_TUN | IFF_NO_PI;
    tunfd = open("/dev/net/tun", O_RDWR);
    ioctl(tunfd, TUNSETIFF, &ifr);

    return tunfd;
}
```

No additional
info sent by
the driver

Create a TUN
device

Register the
device with the
kernel

Configuring the TUN Interface

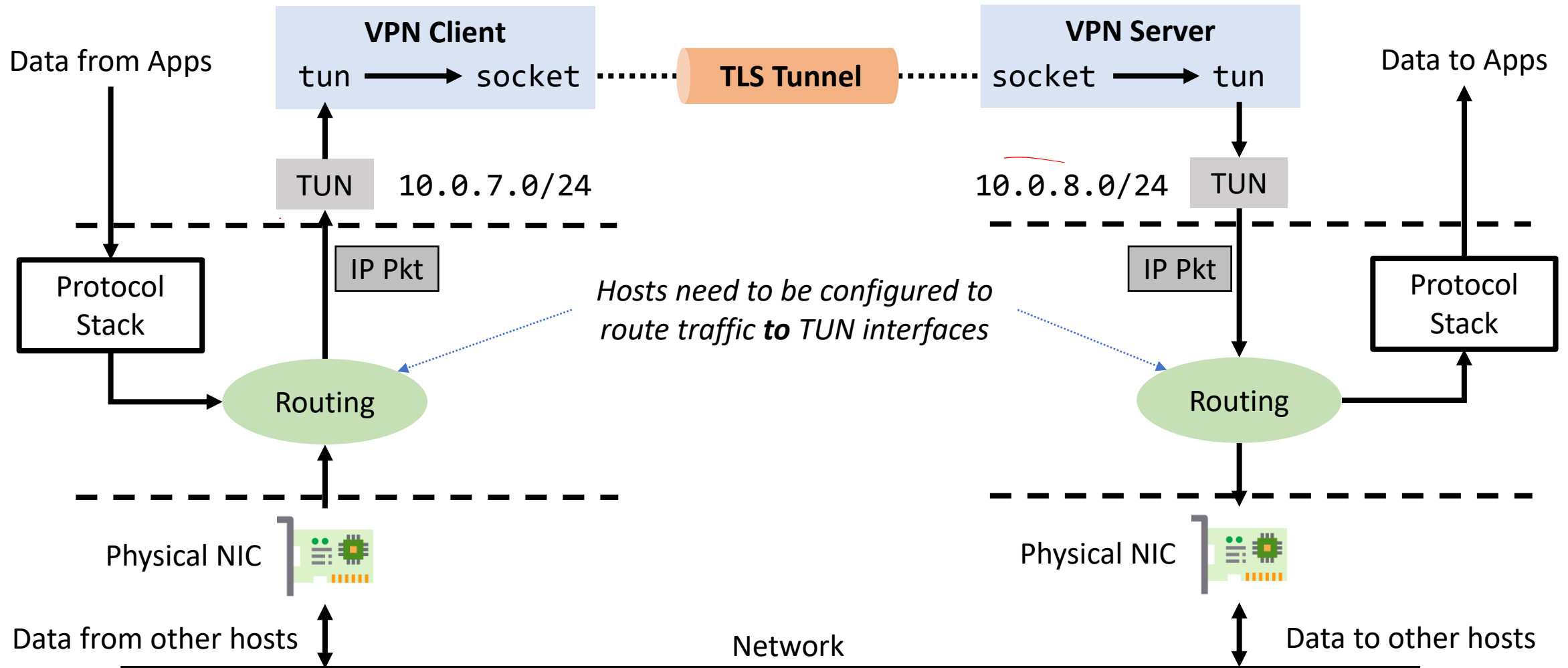
- We need to:
 - Specify what network the interface is connected to
 - Assign an IP address to the interface
 - Activate the interface

*\$ ifconfig -a
tun0*

IP
\$ sudo ifconfig tun0 10.0.7.99/24 up

net- 10.0.7.0/24

What is missing?



Routing Packets

- Routing is modified by configuring routing tables

- Traffic to 10.0.8.0/24 goes through tun0 @ client

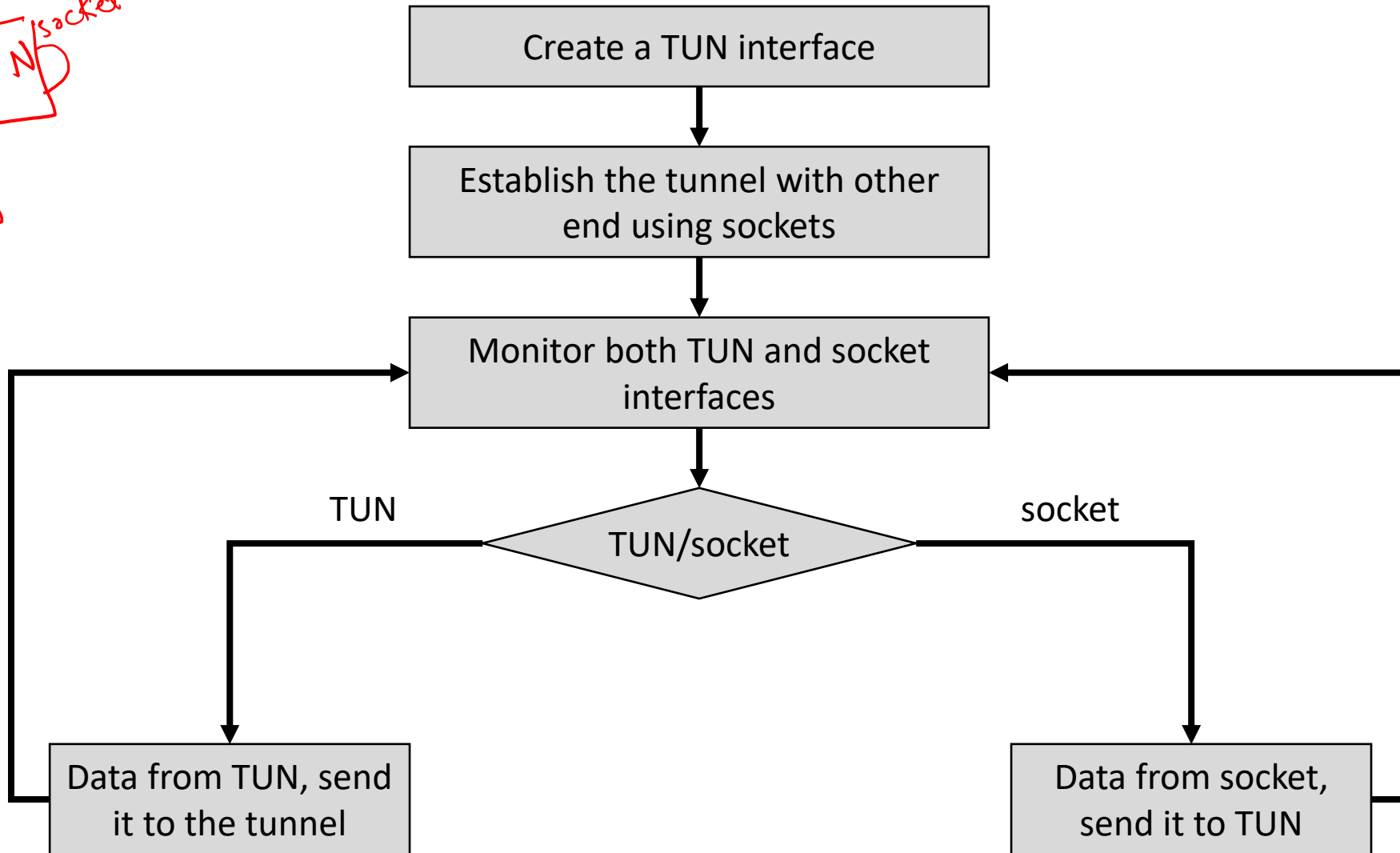
```
$ sudo route add -net 10.0.8.0/24 tun0
```

- Packets written to tun0 ^{10.0.7.0/24} ^{tun0 @ server} are received by our VPN application
 - Should be forwarded to the TLS tunnel
- Packets written to socket are received on the other side

Building a VPN

Overview of Our VPN Program



VPN/socket
TUN



Overview of Our VPN Program

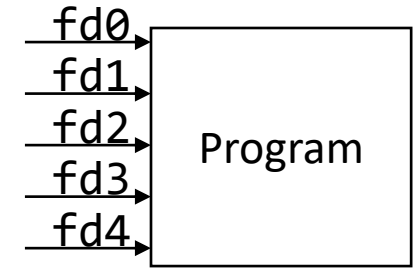


Establishing the IP Tunnel

- We build a simple UDP-based tunnel with no encryption.
- The server:
 - Creates a socket
 - Binds to a specific port
 - Receives data from the client 
- The client:
 - Creates a socket 
 - Sends data to the server

Monitoring File Descriptors

- Option #1: create a thread for each fd → inefficient
- Option #2: IO multiplexing allows:
 - Examining and blocking on multiple I/O streams
 - Notifying the program whenever any one of the streams is active so that it can process data on that stream
- We will use `select` system call for our VPN



select poll epoll

Monitoring File Descriptors

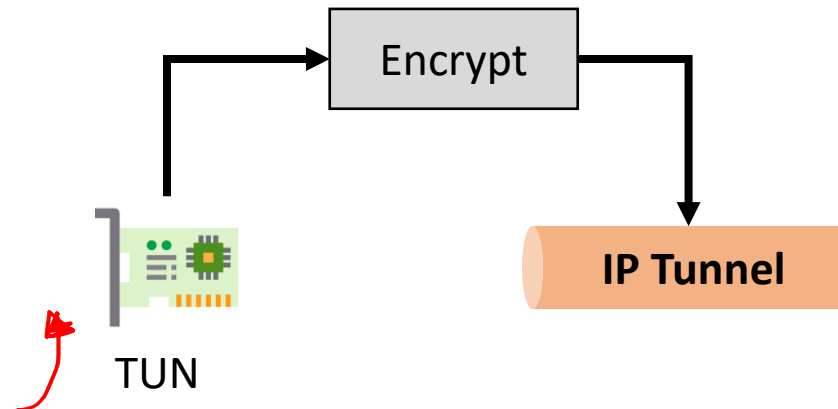
```
while (1) {  
    fd_set readFDSet;  
    FD_ZERO(&readFDSet);  
    FD_SET(sockfd, &readFDSet);  
    FD_SET(tunfd, &readFDSet);  
    select(FD_SETSIZE, &readFDSet, NULL, NULL, NULL);  
  
    if (FD_ISSET(tunfd, &readFDSet)) tunSelected(tunfd, sockfd);  
    if (FD_ISSET(sockfd, &readFDSet)) socketSelected(tunfd, sockfd);  
}
```

Register the fds

IO multiplexing

From TUN to Socket

- When the kernel sends an IP pkt to our VPN program

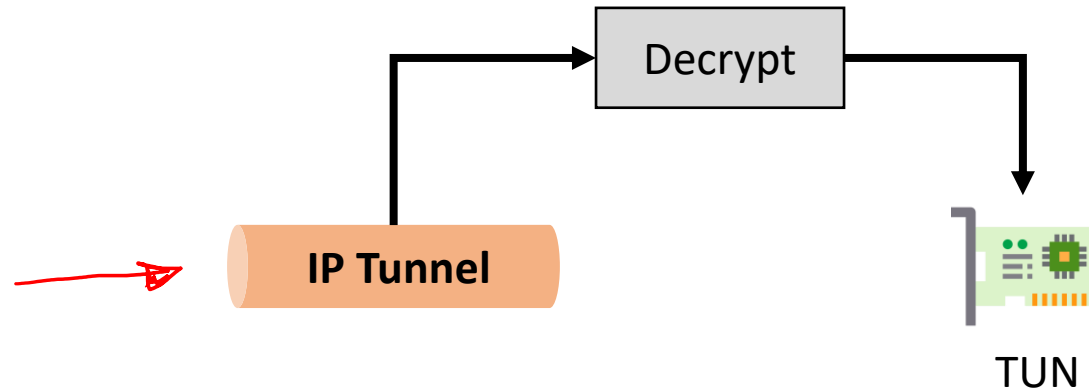


```
void tunSelected(int tunfd, int sockfd){  
    int len;  
    char buff[BUFF_SIZE];  
    bzero(buff, BUFF_SIZE);  
    len = read(tunfd, buff, BUFF_SIZE);  
    sendto(sockfd, buff, len, 0, (struct sockaddr *) &peerAddr,  
           sizeof(peerAddr));  
}
```

Pkt should be encrypted

From Socket to TUN


- When our VPN program sends an IP pkt to the kernel



```
void socketSelected (int tunfd, int sockfd){  
    int len;  
    char buff[BUFF_SIZE];  
    bzero(buff, BUFF_SIZE);  
  
    len = recvfrom(sockfd, buff, BUFF_SIZE, 0, NULL, NULL);  
    write(tunfd, buff, len);  
}
```

Pkt should be decrypted

Summary

- VPNs extend private networks to include hosts from the outside
- VPNs are implemented using IP tunneling
 - IPsec Tunnel Mode 
 - TLS Tunneling 