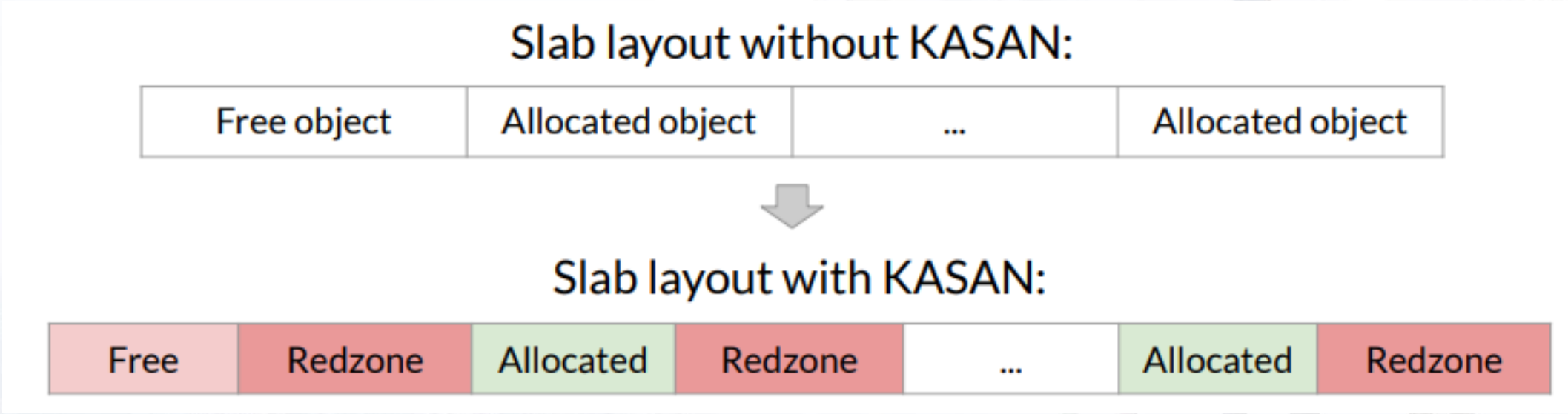
ITMO Advanced OS 2024

Aleksei Romanovskii, PhD,
SPb Research Center (CBG OS Lab)

Lesson 2024.11.06

Allocation hooks

- KASAN need to keep shadow up-to-date
- This requires tracking of alloc/free events
- KASAN adds hooks to kernel allocators
  - SLUB/SLAB, page_alloc, vmalloc (grep code for "kasan_")



Quarantine for freed memory

- When memory is freed, it's typically immediately reallocated
  - Detecting use-after-free is hard

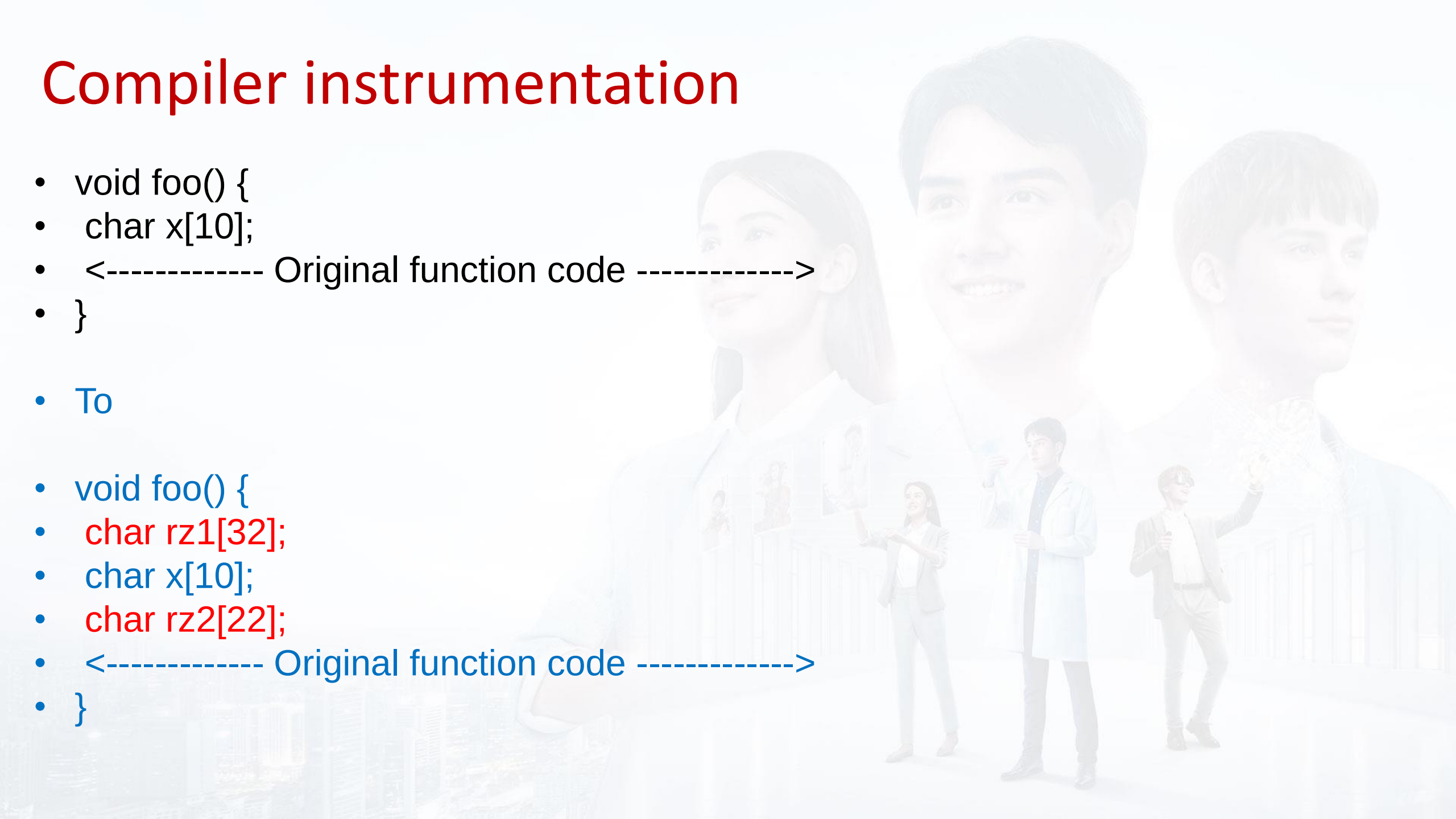
- KASAN implements quarantine for slab objects
  - Freed objects are not returned to allocator immediately
  - Instead, they are put into a delayed reuse queue
  - Higher chance to detect use-after-free

Compiler instrumentation

- void foo() {
- char x[10];
- <---------- Original function code ---------->
- }

- To

- void foo() {
- char rz1[32];
- char x[10];
- char rz2[22];
- <---------- Original function code ---------->
- }

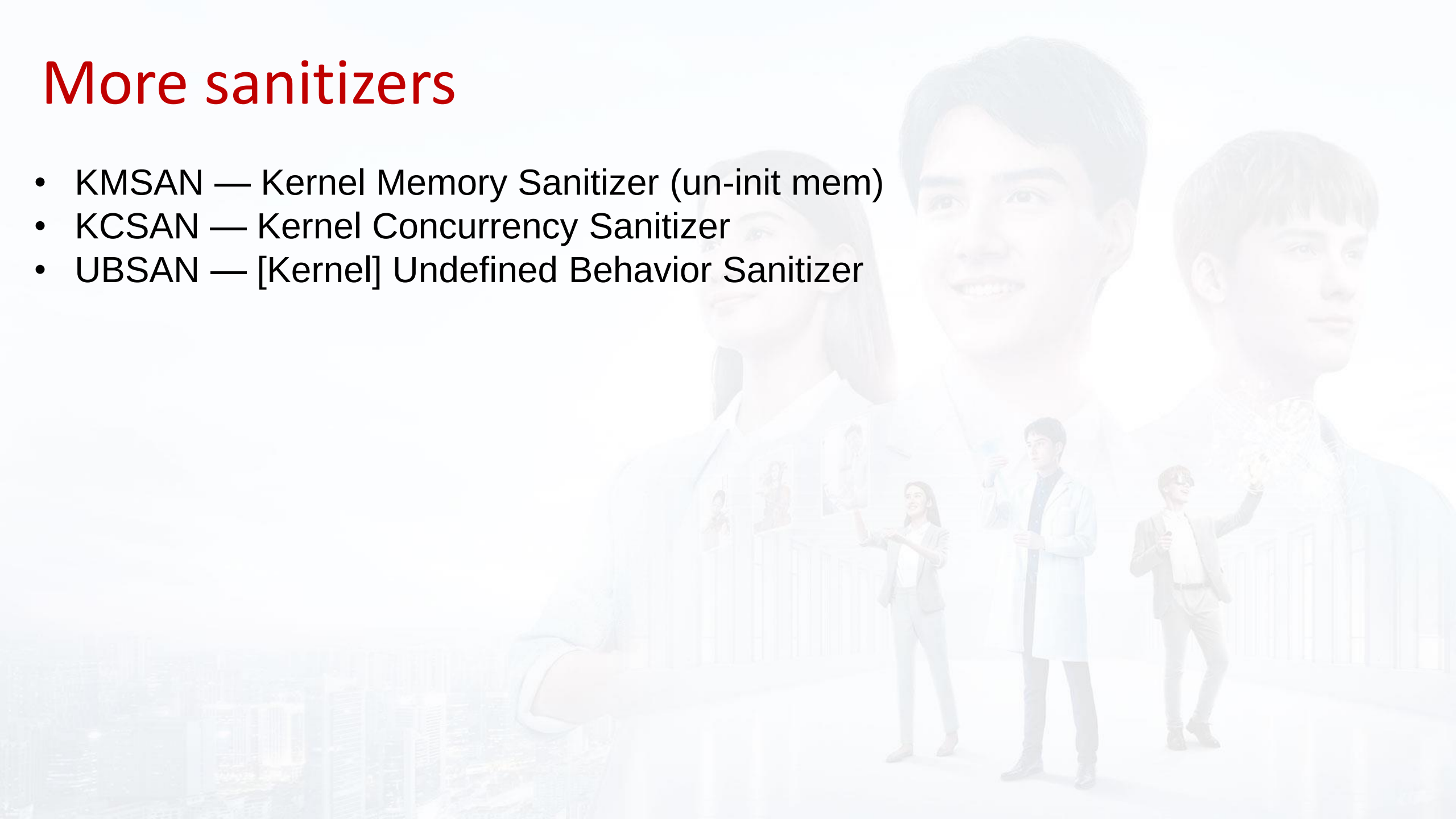


Generic KASAN summary

- Dynamic memory corruption detector for the Linux kernel
- Finds out-of-bounds, use-after-free, and double/invalid-free bugs
- Supports slab, page_alloc, vmalloc, stack, and global memory
- Requires compiler support: implemented in both Clang and GCC
- google.github.io/kernel-sanitizers/KASAN
- Relatively fast: ~x2 slowdown
- RAM impact: shadow (1/8 RAM) + quarantine (1/32 RAM) + ~x1.5 for slab
- Basic usage: enable and run tests

More sanitizers

- KMSAN — Kernel Memory Sanitizer (un-init mem)
- KCSAN — Kernel Concurrency Sanitizer
- UBSAN — [Kernel] Undefined Behavior Sanitizer



Linux networking

- A bit of history: Networking concepts
- Networking overview
- TCP/IP
- Protocols in TCP/IP suite
- TCP/IP addressing
- IPv4 address classes
- Address Resolution Protocol
- Network Stack: Linux kernel view
- Network Stack: Protocol view
- Network data units
- Sending and receiving network data
- Network data “send-receive”
- Flow of data between two computers (TCP/IP)
- Network devices
- IP services: routing
- IP services: fragmentation, timeouts, options
- Socket implementation
- Socket types in detail
- Socket domains (families) and protocols
- Socket use example: Client
- Socket use example: Client-Server
- Socket use example: Server
- Differences between client and server
- Network stack in Linux kernel (recall)

A bit of history: Networking concepts

- In 1962, Paul Baran described, in a RAND Corporation report, a hypothetical military network having to resist a nuclear attack. Small standardized "message blocks", bearing source and destination addresses, were stored and forwarded in computer nodes of a highly redundant meshed computer network.
- In 1967, Donald Davies published a seminal article in which he introduced core network to deal with datagram permutations (due to dynamically updated routing preferences) and to datagram losses (unavoidable when fast sources send to a slow destinations)
- In 1970, Lawrence Roberts and Barry D. Wessler published an article about ARPANET, the first multi-node packet-switching network.
- In 1973, Louis Pouzin presented his design for Cyclades, the first real size network providing to user applications a reliable virtual circuit service (the equivalent of an Internet TCP connection)
- In 1981, the Defense Advanced Research Projects Agency (DARPA) issued the first specification the Internet protocol (IP). It introduced a fragmentation. With fragmentation, some parts of the global network may use large packet size (typically local area networks for processing power minimization), while some others may impose smaller packet sizes (typically wide area networks for response time minimization).
- In 1999, the Internet Engineering Task Force (IETF) standardized the use of the already largely deployed Network address translation (NAT) whereby each public address can be shared by several private devices.
  - With it, the forthcoming Internet Address exhaustion was delayed, leaving enough time to introduce IPv6, the new generation of Internet packets supporting longer addresses.
- In 2015, the IETF upgraded its weak "informational" recommendation of 1998, that datagram switching nodes perform active queue management (AQM), to make it a stronger and more detailed "best current practice" recommendation.

Networking overview

- Network is a system of interconnected devices that can communicate sharing information and resources, such as files, printers, applications, and Internet connection.
- Each of these devices has a unique Internet Protocol (IP) address to send and receive messages between two or more devices using a set of rules called protocol(s)
- TCP/IP is a suite of protocols for communication between computers on the Internet. It stands for **T**ransmission **C**ontrol **P**rotocol / **I**nternet **P**rotocol.

TCP/IP

- TCP/IP specifies how data is exchanged over the internet by providing end-to-end communications
  - how data should be broken into packets, addressed, transmitted, routed and received at the destination.
- TCP/IP requires little central management and is designed to make networks reliable with the ability to recover automatically from the failure of any device on the network.
- TCP
  - takes care of the communication between application software (i.e. browser) and network software
  - defines how applications can create channels of communication across a network
  - is responsible for breaking data down into so called IP packets before they are sent, and for assembling the packets when they arrive
- IP
  - takes care of the communication with other computers
  - defines how to address and route each packet to make sure it reaches the right destination.

Protocols in TCP/IP suite

- TCP (Transmission Control Protocol) communication between applications
- UDP (User Datagram Protocol) simple communication between applications
- IP (Internet Protocol) communication between computers
- ICMP (Internet Control Message Protocol) for errors and statistics
- DHCP (Dynamic Host Configuration Protocol) for dynamic addressing
- Hypertext Transfer Protocol (HTTP) handles the communication between a web server and a web browser.
- HTTP Secure handles secure communication between a web server and a web browser.
- FTP - File Transfer Protocol handles transmission of files between computers.
- NTP - Network Time Protocol is used to synchronize the time (the clock) between computers.
- SSL - Secure Sockets Layer is used for encryption of data for secure data transmission.
- etc.

TCP/IP addressing

- **IP Addresses**
  - Each computer must have an IP address before it can connect to the Internet.
  - Each IP packet must have an address before it can be sent to another computer.
  - This is an IP address: 192.68.20.50
  - This might be the same IP address: www.itmo-global.com
  - TCP/IP uses 32 bits for addressing or 4 computer bytes.
  - TCP/IP V6 uses 128 bits addressing.
- **Domain Names**
  - A name is much easier to remember than a 12 digit number.
  - Names used for TCP/IP addresses are called domain names.
  - www.itmo-global.com is domain name
  - When you address a web site, the name is translated to a number by a Domain Name Server using a distributed database of domain name and IP address bindings.
  - DNS servers contain information on some segment of the DNS and make that information available to clients who are called resolvers.
  - DNS servers are responsible for translating domain names into TCP/IP addresses.
  - When a new domain name is registered together with a TCP/IP address, DNS servers all over the world are updated with this information.

IPv4 address classes (mostly historical)

- The first four bits of an IP address determine the class of the network.
- The class specifies how many of the remaining bits belong to the prefix (aka Network ID) and to the suffix (aka Host ID).
- The first three classes, A, B and C, are the primary network classes.

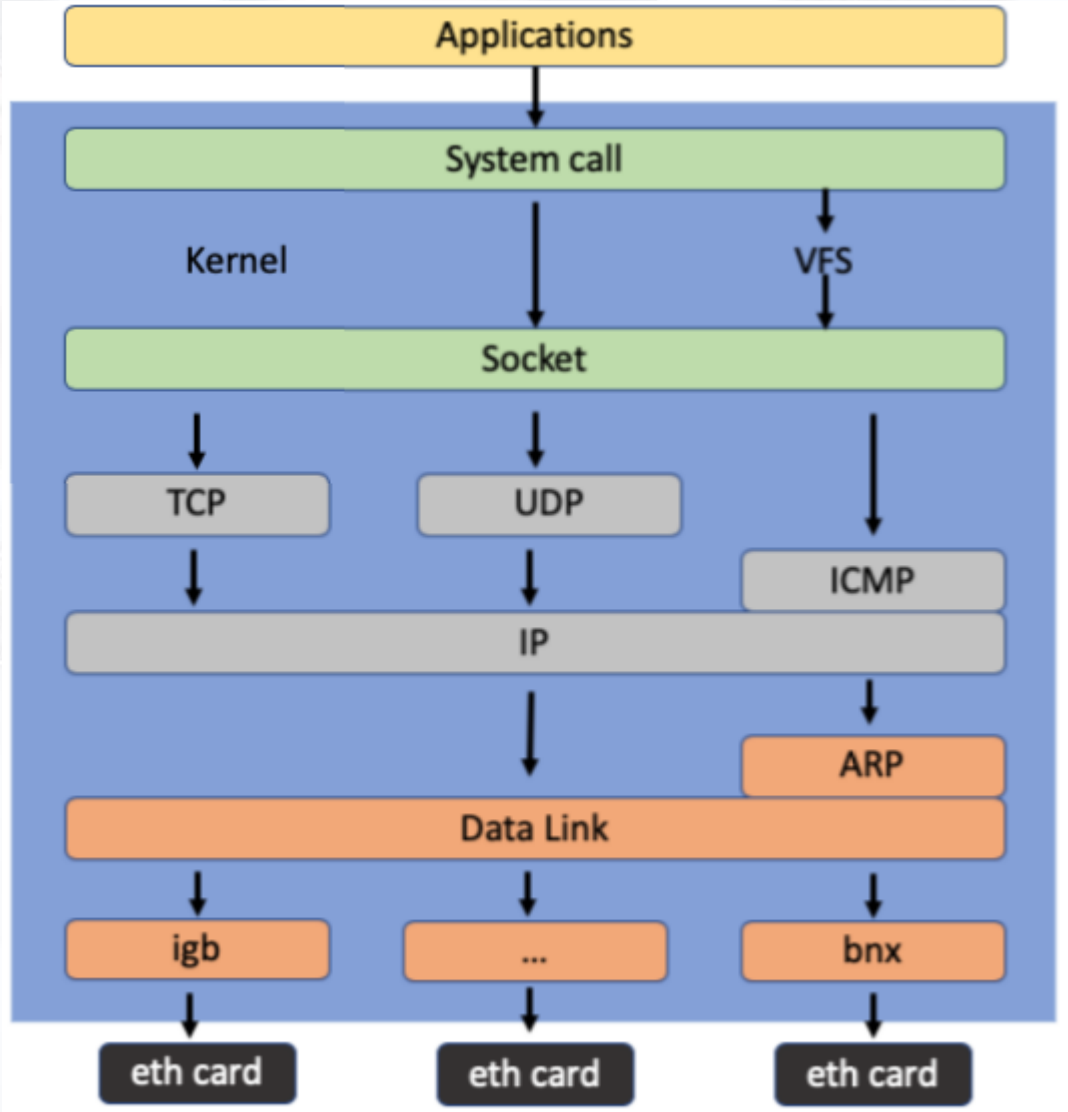
Class	First 4 Bits	Number Of Prefix Bits	Max # Of Networks	Number Of Suffix Bits	Max # Of Hosts Per Network
A	0xxx	7	128	24	16,777,216
B	10xx	14	16,384	16	65,536
C	110x	21	2,097,152	8	256
D	1110	Multicast			
E	1111	Reserved for future use.			

Address Resolution Protocol (ARP)

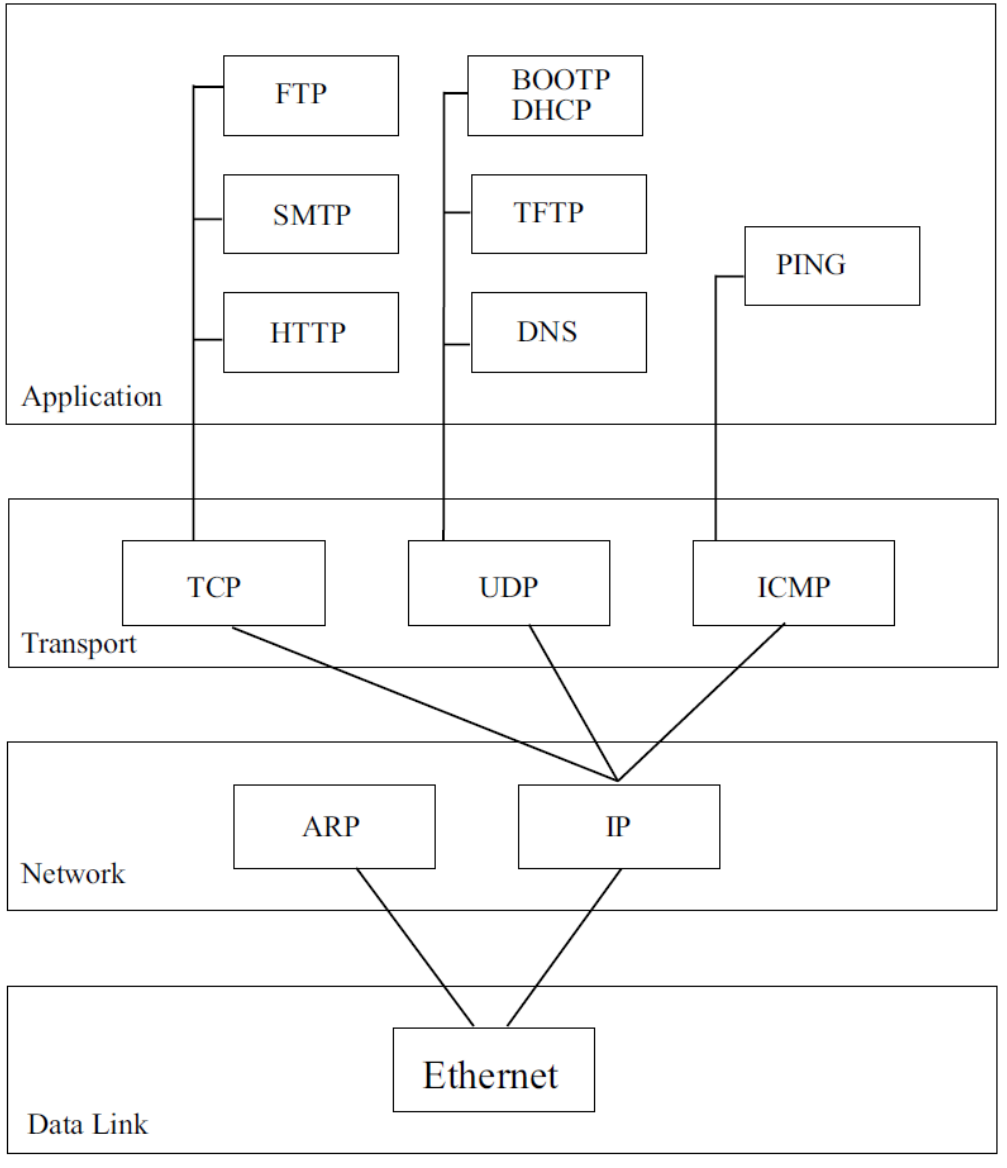
- ARP is used to translate IP addresses to physical ones.
- The network hardware does not understand the software-maintained IP addresses.
- ARP translates the 32-bit IP address to a physical address that matches the addressing scheme of the underlying hardware (for Ethernet, the 48-bit MAC address).
- There are three general addressing strategies:
  - Table lookup
  - Translation performed by a mathematical function
  - Message exchange
- ARP employs the third strategy, message exchange.
- ARP defines a request and a response.
  - A request message is placed in a hardware frame (e.g., an Ethernet frame), and broadcast to all computers on the network.
  - Only the computer whose IP address matches the request sends a response.

Network stack: Linux kernel view

- Modern Linux network model, so called TCP/IP model (stack), comprises 5 layers:
  - Application (session: sockets and files)
  - Transport (TCP, UDP, etc)
  - Network (IPv4/6, ARP)
  - Data link (e.g.Ethernet)
  - Physical
- The top level application needs to interact with the socket interface through system calls
- Below the socket is the transport layer, network layer
- The bottom layer is the network card driver and the physical network card device



Network stack: Protocol view



Network data units

- Packet is a basic unit of information to transfer over network
  - Packet is TCP protocol data unit at transport layer, but often is used as more general term
  - A packet consists of control information and user data; the latter is also known as the payload.
  - Control information provides data for delivering the payload (e.g., source and destination network addresses, error detection codes, sequencing information, etc.).
- Packet is a TCP unit of data, while datagram is a UDP unit of data.
  - But… packet and datagram may be (often) used as synonyms
- Packets may be divided into fragments to fit into MTU (Maximum Transmitted Unit)
  - Fragment is the protocol data unit at the network layer
- Fragments are later packed into frames
  - Frame is the protocol data unit at the data link layer

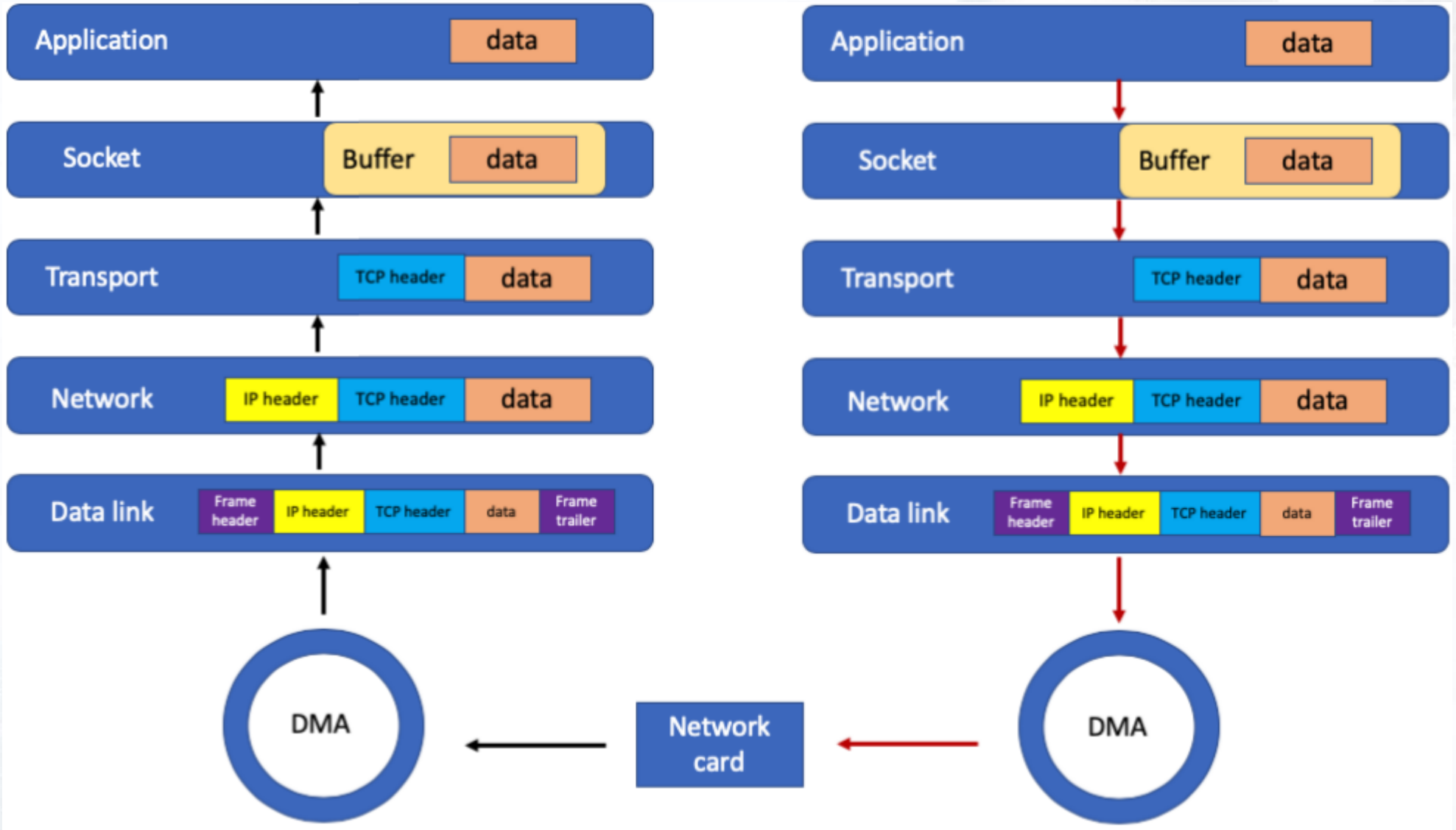
Sending network data

- Application calls socket API to send network packets. System call will be trapped in socket layer of kernel mode
- The socket layer will put the data packet into the socket send buffer
- Next, the network stack takes out the data packet from the socket send buffer, and then process it layer by layer according to TCP/IP stack
  - The **transport layer** adds TCP header to the packet
  - The **network layer** adds IP header to the packet, and perform fragmentation according to the MTU size
  - The fragmented network packets are sent to data link layer, which addressing for MAC address and add the frame header and tail, the frame will be put into the sending queue
    - frame is the protocol data unit at the data link layer
  - Then **data link layer** will trigger a soft interrupt notify the network card driver that there are new network frames in the packet queue
  - The network card driver reads the network frame from the packet sending queue through DMA and sends it through the physical network card

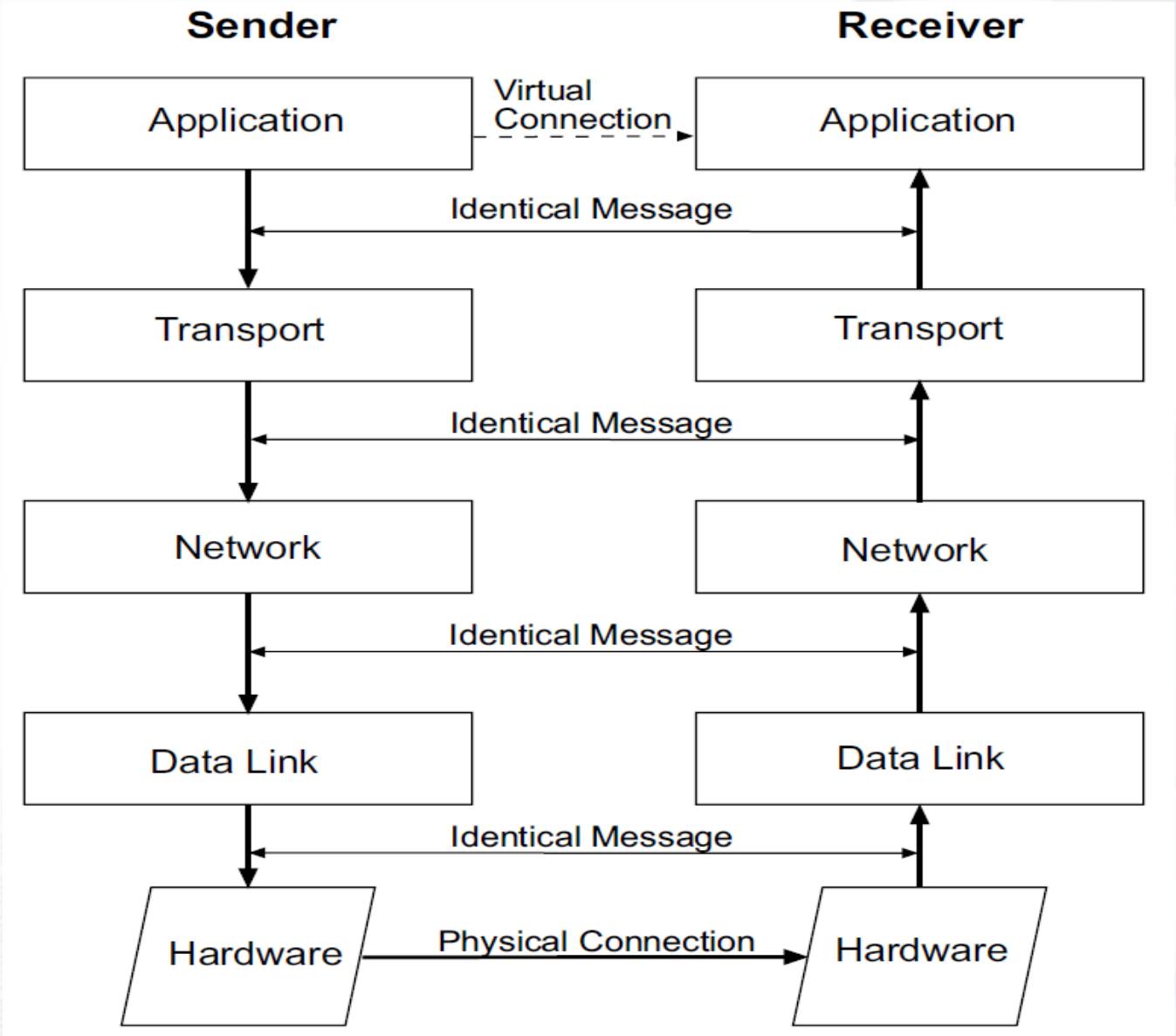
Receiving network data

- When a network frame arrives at the network card, the network card puts the network packet into the packet receiving queue through DMA (direct memory access)
- Then the card tells the interrupt handler that the network packet has been received through an interrupt.
- Next, the kernel protocol stack takes out the network frame from the buffer, and processes the network frame layer by layer from bottom to top through the network stack:
  - Check the validity of the packet at **data link layer**, determine the type of network protocol (IPv4 vs IPv6), then remove the frame header and trailer, hand it over to network layer
  - **Network layer** takes out the IP header and determines the next direction of the network packet, such as whether to hand it over to transport layer or forward it. If the packet stays in local machine, it will remove the IP header and pass it to transport layer
  - After **transport layer** takes out the TCP/UDP header, it finds the corresponding socket according to <source IP, source port, destination IP, destination port> quadruple as identifier, then copies the data to the receiving buffer of the socket
  - Finally, the application uses socket interface to read data.

Network data “send-receive”



Flow of data between two computers (TCP/IP)



Network devices

- When a data packet is sent from a computer, it arrives at an IP router.
  - All routers on an internet contain IP protocol software and use a routing table to determine where to send a packet next.
  - The destination IP address in the IP header contains the ultimate destination of the packet, but it might go through several other IP addresses (routers) before reaching that destination.
- A local network can be connected to the Internet by means of a gateway. The gateway is a computer that is connected both to the local network and to the Internet.
- Gateway computer checks IP address to determine where to forward the data packet.
- Repeaters are used to join network segments when the distance spanned causes electrical signals to weaken.
  - Repeaters are amplifiers that work at the bit level; they do not actively modify data;
- Bridges are used to connect two LANs together.
  - Bridges work at the frame level.
  - Bridges can detect and discard corrupted frames.
  - They can also perform frame filtering, only forwarding a frame when necessary.
  - Both of these capabilities decrease network congestion.

IP services: routing

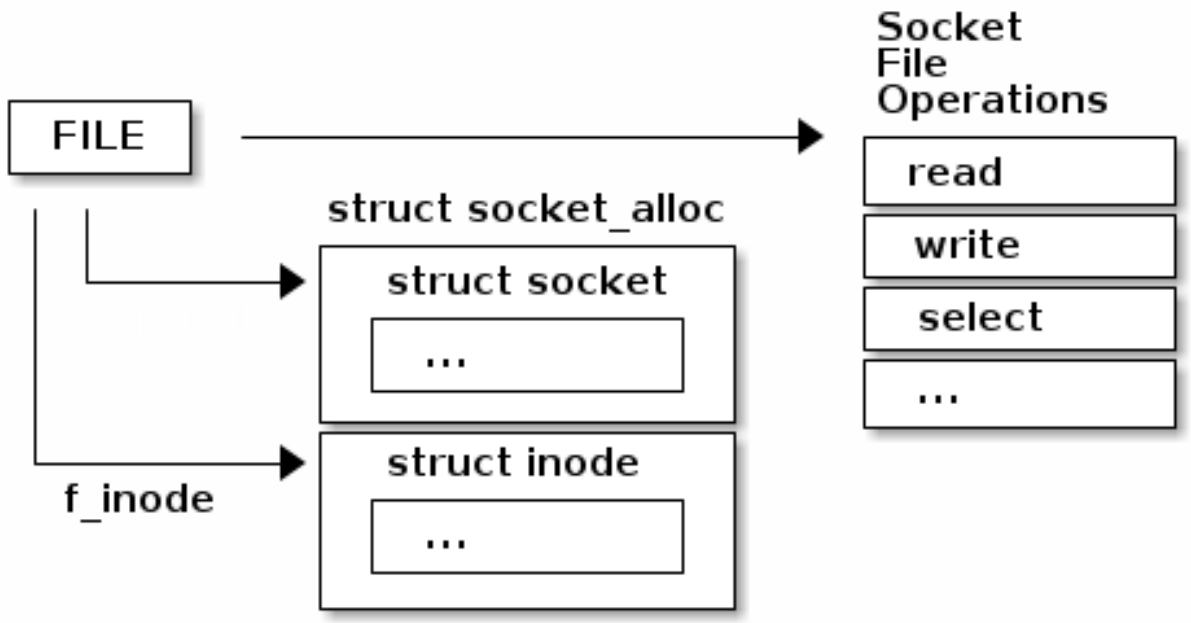
- Data packet travels from its source to its destination by means of routers.
- The router is responsible for the right addressing, depending on traffic volume, errors in the network, or other parameters.
- Routing table entries are created when TCP/IP initializes.
- Routing table entries provide needed information to each local host regarding how to communicate with remote networks and hosts.
- When IP receives a packet from a higher-level protocol, like TCP or UDP, the routing table is searched for the route that is the closest match to the destination IP address.
- The most specific to the least specific route is in the following order:
  - A route that matches the destination IP address (host route).
  - A route that matches the network ID of the destination IP address (network route).
  - The default route.
- If a matching route is not found, IP discards the packet.

IP services: fragmentation, timeouts, options

- Fragmentation: IP packets may be divided into smaller packets.
  - This permits a large packet to travel across a network which only accepts smaller packets.
  - IP fragments and reassembles packets transparent to the higher layers.
- Timeouts: Each IP packet has a Time To Live (TTL) field
  - TTL is decremented every time a packet moves through a router.
  - If TTL reaches zero, the packet is discarded.
- Options: IP allows a packet's sender to set requirements on the path the packet takes through the network (source route);
  - the route taken by a packet may be traced (record route) and packets may be labeled with security features.

Sockets implementation

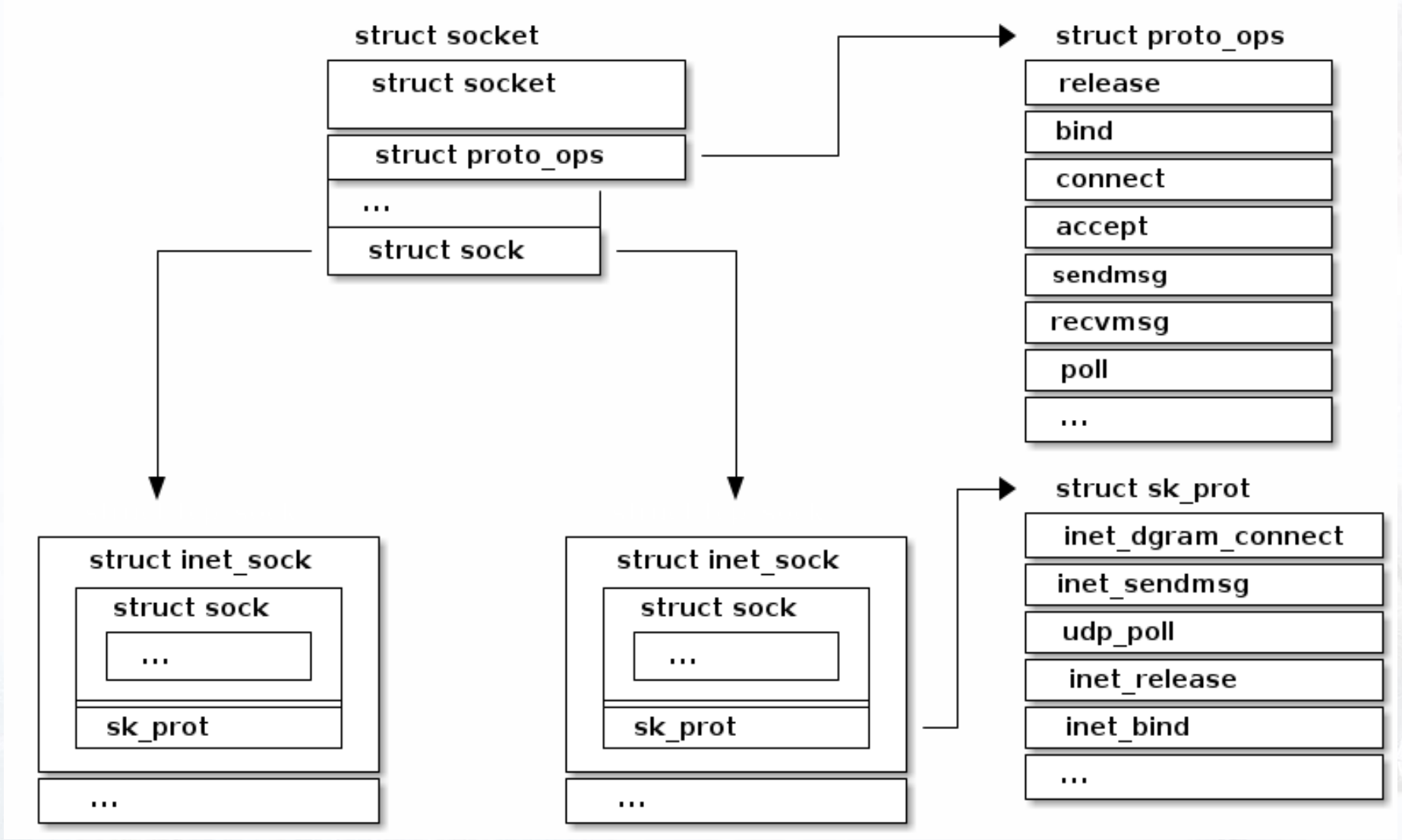
- A *socket* is one endpoint of a communication channel used by programs to pass data back and forth locally or across the Internet.
- Sockets have following primary properties controlling the way they send data:
  - the *address family* controls the OSI network layer protocol used (AQF_INET, AF_INET6, etc.)
  - the *socket type* controls the transport layer protocol (SOCK_STREAM, etc)



Socket types in detail

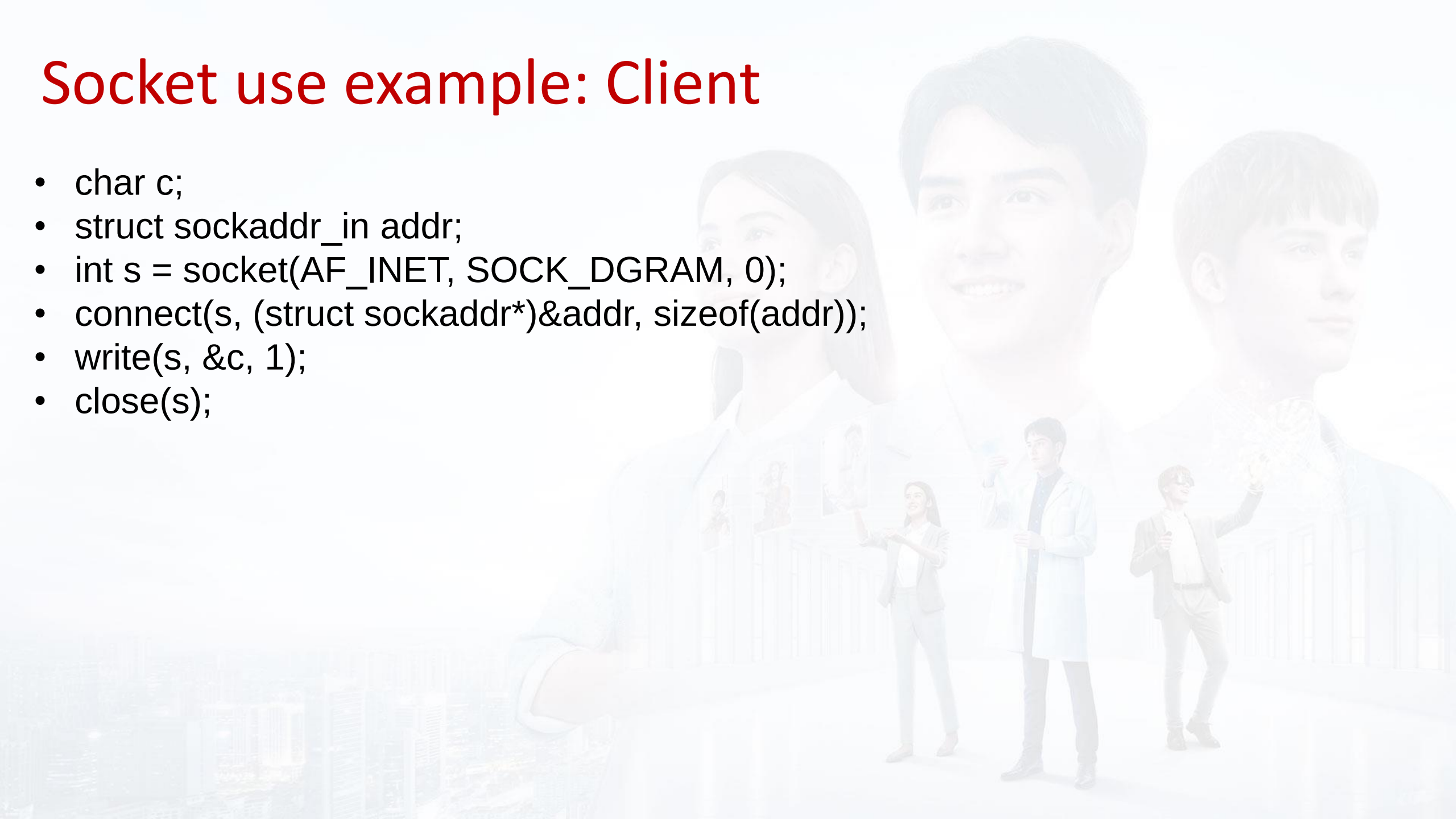
- SOCK_STREAM
  - Provides sequenced, reliable, two-way, connection-based byte streams.
  - An out-of-band data transmission mechanism may be supported.
- SOCK_DGRAM
  - Supports datagrams (connectionless, unreliable messages of a fixed maximum length).
- SOCK_SEQPACKET
  - Provides a sequenced, reliable, two-way connection-based data transmission path for datagrams of fixed maximum length;
  - A consumer is required to read an entire packet with each input system call.
- SOCK_RAW
  - Provides raw network protocol access.
- SOCK_RDM
  - Provides a reliable datagram layer that does not guarantee ordering.

Socket domains (families) and protocols

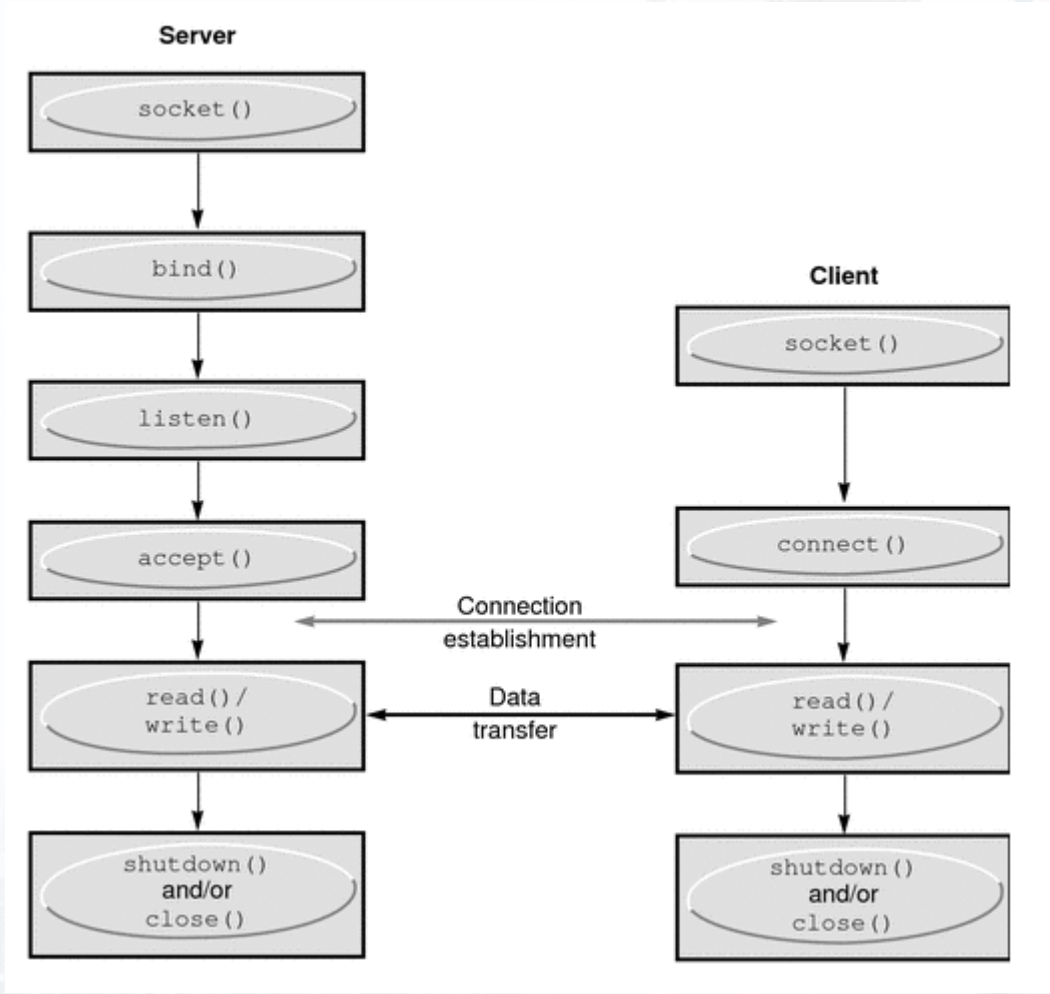


Socket use example: Client

- char c;
- struct sockaddr_in addr;
- int s = socket(AF_INET, SOCK_DGRAM, 0);
- connect(s, (struct sockaddr*)&addr, sizeof(addr));
- write(s, &c, 1);
- close(s);



Socket use example: Client-Server



Socket use example: Server

```
int sock, length;
struct sockaddr_in6 server;
int msgsock;
char buf[1024];
int rval;
sock = socket(AF_INET6, SOCK_STREAM, 0);
if (sock == -1)
    exit(1);
bzero (&server, sizeof(server));
server.sin6_family = AF_INET6; server.sin6_addr = in6addr_any; server.sin6_port = 0;
if (bind(sock, (struct sockaddr *) &server, sizeof server) == -1)
   exit(1);
length = sizeof server;
if (getsockname(sock,(struct sockaddr *) &server, &length) == -1)
    perror("getting socket name");
listen(sock, 5);
do {
    msgsock = accept(sock,(struct sockaddr *) 0,(int *) 0);
    if (msgsock == -1)
        perror("accept");
    else do {
        memset(buf, 0, sizeof buf);
        if ((rval = read(msgsock,buf, sizeof(buf))) == -1)
            perror("reading stream message");
        if (rval == 0)
            printf("Ending connection\n");
        else
            printf("-->%s\n", buf);
    } while (rval > 0);
    close(msgsock);
} while(TRUE);
```

Differences between client and server

Client Software	Server Software
An arbitrary application program that becomes a client when a remote service is desired. It also performs other local computations.	A special-purpose, privileged program dedicated to providing one service. It can handle multiple remote clients at the same time.
Actively initiates contact.	Passively waits for contact.
Invoked by a user and executes for one session.	Invoked when the system boots and executes through many sessions.
Capable of accessing multiple services as needed, but actively contacts only one remote server at a time.	Accepts contact from an arbitrary number of clients, but offers a single service or a fixed set of services.
Does not require special hardware or a sophisticated operating system.	Can require powerful hardware and a sophisticated operating system, depending on how many clients are being served.

Network stack in Linux kernel (recall)