Communication

1. Naming
2. Routing
3. Reliability
4. Security
5. Performance:
   - Latency: 1st bit from sender to receiver
   - Bandwidth/throughput: total bits/second

Naming solved by naming service (eg. DNS, or ARP)

End points
Use for multiplexing the channel
How is the communication take place now?

Observations:
- there is a lot of copying: increase bus traffic, polluting the cache.
- A lot of kernel crossings, so it is very expensive, see context switch (trap, saving registers, change of stack...).

So when we have a faster network: from 3-10 Mbit to 100 Mbit, to 150 Mbit to Gigabit Ethernet, what we perceive is that LATENCY gets worse for small packages.

Possible reasons:
- Copying
- Inefficient implementation of communication protocols
- Lack of “right” mechanism: TCP/IP was design as WAN protocols, where IP is responsible for routing and TCP is responsible for reliability such that overcome the problems of a) linkerrors, b) line of different speeds, c) multi-crossing channel and congestion control. But with LAN networks the linkerrors are small, no multilinks, and no congestion \( \Rightarrow \) TCP/IP too much overhead for LAN.

UNET PAPER

Goals:
1. Avoid copying if possible
   - True zero copying: you don’t have to compact objects in your space, the interface will do it. So in here the communication segments span the whole user address space. Difficult of achieve at the receiver because we
need a pointer to the function that will be invoked upon receiving the message, so the function which is user code will know where the message should be copied to. It is not safe because user code is copying out of hardware buffer. Also the problem of where how the sender knows about the pointer of the handling function. Possibilities are handshake, indirect table.

- zero copying

2. get the kernel out of the critical path
   - need at setup time because of protection

3. leave function ability of protocol to user level: different applications need different communication protocol, but the kernel can provides only a generic one. Eg. Active messaging: integrate communication with application via handler function.
   - Not 100% true because how many application patterns exist???

Engineering of UNET

Endpoints are queues in communication segments which are in user virtual space, so no copy through kernel, and protection is provided via virtual address space.

Communication segment

Multiplexing is achieved by tagging the channels. Cart has to deal with virtual address when has a receive because communication segment is in virtual address space. Needs some kind of cache, and translation is aided by the kernel.

A Survey of Messaging Software Issues and Systems for Myrinet based clusters

- Standard Message Passing
- Handler carrying message – Active massage
- Pipelining (store and forward), DMA vs PIO
- Zero copying vs. true copying
- Multiplexing and protection
- Remote memory write
- Arrival notification: poll vs. interrupt
- Reliability

Remote memory write
- Shared memory model
- Is incorporated into Ethernet
- 2 copies, so there is consistency problem, so if you know the pattern well then it is useful.
- Too low level, inconvenient for high level application

**RPC**

- Procedural model for distributes systems
- Sequential/blocking model

Issues:
1. Naming/Location
   - Dynamically locate a server that is willing to serve my request.
   - So need a name server: find that name server and attach the handler to the server.
2. Not true procedure call (parameter passing)
   - Because of network failure you may return to the application even though it is not completed, and the function may still running on the server.
   - Idempotence:
     1. exact-one
     2. at most one
     3. whenever
     if invoked multiple times does not change the value, so has no side effect.
3. Reliability
4. Performance
5. Security
6. Hardware incompatibility