Communication

Problems:
1. Naming
2. Routing
3. Reliability
4. Security
5. Performance
   - Latency - time from when the 1st bit leaves the sender to the time it arrives at the receiver
   - Bandwidth & Throughput (bits/second)

Logical address is translated to physical.
ARP - protocol to discover physical address.
To send data:
- Sender copies memory into the kernel.
- Kernel copies memory into a different location.
  
  "Thu: The target region of memory has to be pinned as well. This is assuming that DMA is used to transfer the data. If the processor is moving the data via PIO, then it should be able to deal with a page fault and so the memory does not have to be pinned."
- Kernel negotiates the data transfer with hardware.
- Data is sent to the hardware.

To receive data:
- Hardware raises an interrupt.
- Data is copied to pinned memory in kernel.
- Kernel copies the data to a different location and waits for a receive call.
- When receiver sends a receive call, data is copied to receiver.

Kernel copying is expensive because several things have to happen:
- Trap
- Save context/registers/stacks
  
  "Thu: Copying is expensive because it involves memory operations over the bus. Multiple copies per message can make the bus a performance bottleneck. It can also affect your cache. Finally, depending on whether memory bandwidth is keeping pace with processor performance or not, just the copying itself, involving touching memory is expensive.

  Entering the kernel is also expensive for the reasons cited above but this is separate from why copying is expensive.""

Software bloat (software instructions taking too long)  
  "Thu: not “instructions.” I assume that this “bloat” is referring to the thickness of the protocols. So, it’s the number of instructions, particularly when they are meant to handle events that aren’t as common on LANs (e.g., dropping of packets)."
- Copying
• Inefficient implementation of communication protocol
• "Right" mechanism

WAN protocol is different from LAN. On LAN there are no line errors and link speed is fixed.
[[Thu: There are always the possibility of errors. On LANs, the possibility is much lower.]]

IP:
• Internetworking
• Routing

TCP (has congestion control):
• Reliability
• Lines errors
• Lines of different speed (bottleneck link)
• Multi-crossing channel

U-Net Paper
• Avoid copying if at all possible
  o True 0 copying vs. 0 copying
  o 0 copying - there is no copying after buffer is handed out to kernel
  o True 0 copying - sending data from where it is defined (no buffer)
• Get the kernel out of the way in the critical path (important for small messages)
  o Keep kernel at set-up time for protection
  o Mechanism for ensuring no violation of protection
  o Tightly coupled parallel distributed
• Leave functionality of protocol to user-level (receivers are slower than senders)
  o Applications have different needs
  o Kernel can do only general functions
  o Damage vs. benefits. (Not as reliable)
• Active message
  o Integrate communication with computation
  o Kernel does not know where message should go
  o Message includes a pointer to a function that should be invoked
    ▪ True 0 copying on the receiver
    ▪ Function will get the rest of the data from NIC
    ▪ Kernel only reads a few bytes to get the function pointer
  o How to know address of the function?
    ▪ Single point of program (virtual address is the same on all machines)
      [[Thu: SPMD model; single program multiple data.]]
    ▪ Indirection (table that maps function pointers)
**Communication Channel**

- Kernel allocates communication segment. This is done for performance for tightly-coupled parallel/distributed programming.
- Virtual address space allows user access to device in save way
- For true 0 copying memory has to be added/freed for the receiver
- Kernel must demultiplex "channel tag ID" to find the right receive queue
- Report queue is used for error reporting (ex: queue if send queue is full)
- All queues are constantly monitored to see when there is new data
  - NIC monitors the send queue (for performance send queue should be on the NIC card)
  - User process monitors the receive queue (for performance receive queue should be in user memory)
- NIC knows only the physical address
- User program knows only the virtual address
- NIC has to translate between virtual and physical memory
  - NIC has a cache of virtual address mappings
  - On miss, interrupt is send to kernel
  - Kernel manages the cache
  - Cache is located in pinned memory
  - Mapping has to be consistent with physical memory

**Virtual Address Space**

![Diagram of Communication Channel](image-url)
A Survey of Messaging Software Issues and Systems for Myrinet-Based Clusters

- Standard Message Passing
- Handler carrying messages
- Multiplexing and protection
- Arrival and notification (polling, interrupt)
- Reliability - happens on higher level
- Remote memory read/write
- Pipelining

If system is asynchronous (no assumptions can be made on how long things take) and has failures, one cannot be sure of reliability.

Pipelining is faster than sending the whole chunk of data together -> better latency
Remote memory read and write
- Shared memory
- Need synchronization
- Currently being incorporated into Ethernet standard
- Memory mapping is arranged through the card
- Translation between addresses happens on both sides
- Kernel is only involved in setting up the mapping
- After read is made there are two copies of data
- Have to worry about consistency

RPC
- Invoking a remote function
- Passing thread of control
- Not a true procedure call since it can return before procedure is done (server crash, network failure, etc)

Problems
- Naming/Location
- Parameter passing
- Reliability
- Performance
- Security
- Hardware incompatibility (solved by single standard)