Networking: General Discussion

Presented by Jonathan Hayase & Kaiming Cheng

DARPA design > outline

1. Introduction & fundamental goal
2. 7 second level goals
    a. Fault tolerance
    b. Multiple services
    c. Multiple networks
3. Architecture and Implementation
4. Datagrams
5. Transmission control protocol
6. Discussion

DARPA design > introduction

- DARPA = Defense Advanced Research Projects Agency
  - Government funded military research
  - Satellites, GPS, computers, *COVID-19 vaccines*
- **Fundamental goal:** “develop an effective technique for multiplexed utilization of existing interconnected networks”
  - Packet switching for multiplexing
  - Connecting existing networks for practical reasons

DARPA design > the 7 secondary goals

1. **Internet communication must continue despite loss of networks or gateways.**
2. **The Internet must support multiple types of communications service.**
3. **The Internet architecture must accommodate a variety of networks.**
4. The Internet architecture must permit distributed management of its resources.
5. The Internet architecture must be cost effective.
6. The Internet architecture must permit attachment with a low level of effort.
7. The resources used in the internet architecture must be accountable.

} we'll come back to these at the end

DARPA design > goal > fault tolerance

Goal: **Internet communication must continue despite loss of networks or gateways.**

- Where to store communication state
  - In the network
    - Intermediate packet switches store state
    - Need to implement replication of this state to survive failures
    - Complexity in network
  - **On the communicating hosts**
    - “fate-sharing”
    - Intermediate packet switches are stateless
    - Hosts are responsible for handling failures
    - Complexity at hosts

DARPA design > goal > multiple services

Goal: **The Internet must support multiple types of communications service.**

“The initial concept of TCP was that it could be general enough to support any needed type of service. However, as the full range of needed services became clear, it seemed too difficult to build support for all of them into one protocol.”

- XNET needed to operate in potentially broken environments where TCP may not work at all.
- Digitized speech applications can handle packet loss more effectively than TCP can.
- Originally TCP and IP were a single protocol but were split up, allowing other protocols to be implemented on top of IP.
    - UDP gives access to basic datagrams
    - Lots of custom protocols implemented on top of UDP

DARPA design > goal > multiple networks

Goal: **The Internet architecture must accommodate a variety of networks.**

- Long haul: ARPANET and X.25 (packet-switched) networks
- Local area: **Ethernet**, ringnet, etc.
- Broadcast satellite:
  - DARPA Atlantic Satellite [64 kbps]
  - DARPA Experimental Wideband Satellite Net [3 Mbps]
- Packet radio:
  - DARPA packet radio network
  - British packet radio network
  - Amateur packet radio network
- Serial links: between 1200 bps and T1 [1.544 Mbps]
- And more!

DARPA design > goal > multiple networks

Goal: **The Internet architecture must accommodate a variety of networks.**

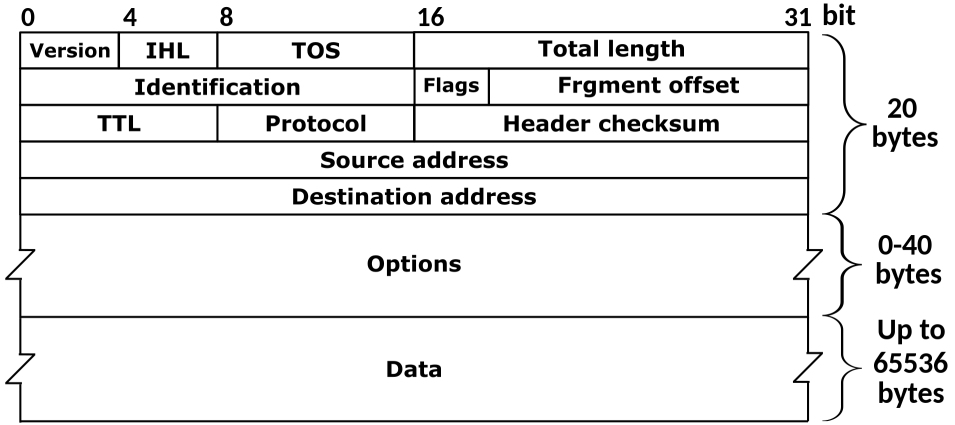
- Solution: make the minimum set of assumptions necessary about the underlying network.
  - Network can transport a packet of reasonable size.
  - With reasonable speed and accuracy.
- A given service may or may not work, depending on the underlying network network properties.
- In exchange, services are naturally network agnostic!

DARPA design > architecture & implementation

- *Realization: a particular set of networks, gateways and hosts which have been connected together in the context of the Internet architecture*
- Because the Internet makes few assumptions about its realization, realization designers are granted substantial freedom.
- Realization designers face substantial engineering challenges:
    - Ensuring correctness, performance, reliability, and cost
- Difficulty specifying performance characteristics for military standards

DARPA design > datagrams

- Motivations:
  - Stateless
  - Simplest possible unit of communication
- Authors note that most service requires something that basic datagrams do not provide.
- More transport features (e.g. reliability, delay smoothing) can be implemented on top of datagrams.



DARPA design > transmission control protocol (TCP)

- Very briefly: TCP is a protocol to ensure reliable data transmission
  - Sequence numbers
  - Acknowledgement
- TCP regulates bytes not packets
  - So control information could be acknowledged as well as data (dropped)
  - So packets could be split into smaller packets (dropped)
  - Multiple dropped packets could be coalesced into a single packet for retransmission
- End-Of-Letter flag
  - Intended to separate records
  - Had complex emergent semantics and was ultimately dropped from the standard
- Retrospectives

Pre-lecture

Q1: Describe a network with a different ordering of priorities (or new priorities) and how it would manifest in the real world.

- Accountability as a priority
  - Using blockchain!
- New priorities
  - Mobility
  - Decentralization
  - Security
  - NSF Future Internet Design list
  - Zero Trust
  - Availability

DISCUSSION: https://tinyurl.com/cse550au21-network

Goals:

1. Internet communication must continue despite loss of networks or gateways.
2. The Internet must support multiple types of communications service.
3. The Internet architecture must accommodate a variety of networks.
4. The Internet architecture must permit distributed management of its resources.
5. The Internet architecture must be cost effective.
6. The Internet architecture must permit attachment with a low level of effort.
7. The resources used in the internet architecture must be accountable.

Questions:

1. What goals conflict with each other?
2. How well does the modern internet satisfy these goals?
3. What additional goals do we desire now?
4. Ask your own questions!

End To End Argument In System Design -- Outline

1. Introduction
2. Case Study: Careful File Transfer
    a. End-to-End Caretaking
    b. Real World Example
    c. Performance Analysis
3. Other Examples of End-to-End Arguments
4. Identifying The Ends
5. History & Application of other Systems
6. Conclusion

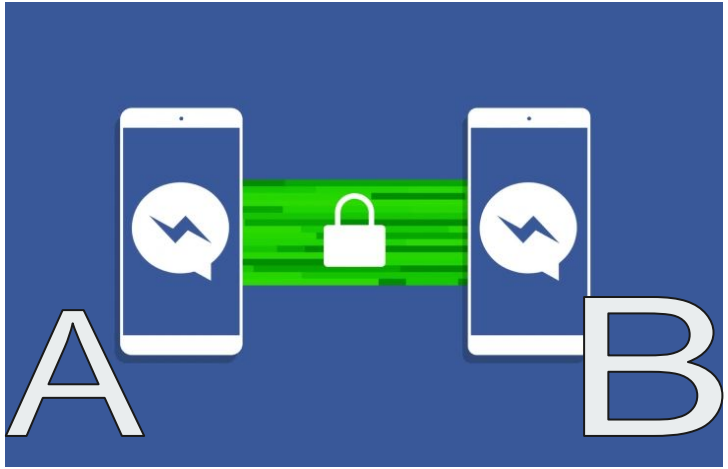
Intro

- Choosing the proper boundaries between functions is perhaps the primary activity of the designer
- This paper discusses one class of function placement argument that has been used for many years
- This paper focuses on the communication network version of this argument.

Problem Definition

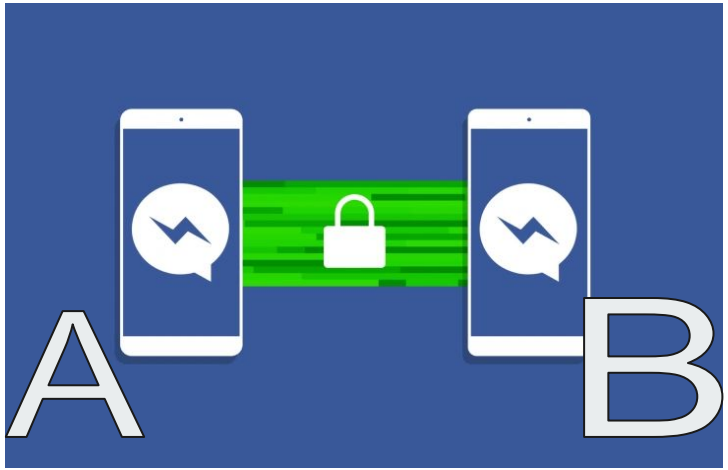
- **End-to-end principles**:
  - It should help guide placement of functions among the modules:
  - Functions placed at the low level may be **redundant** or of **little value**
  - The function in question can completely and correctly be implemented **only with the knowledge and help of the application** standing at the **endpoints** of the communication system.

End to End Caretaking -- A Careful File Transfer



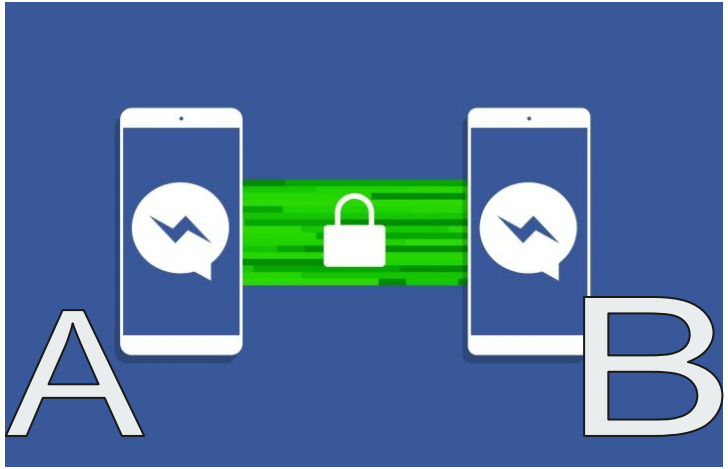
1. **A** correctly read file from the storage (disk)
2. **A's** file transfer program asks the data communication system to transmit the file (split the data into packets)
3. Data communication network moves packets from **A** to **B**
4. **B** removes the packets and hands the data to **B**'s file transfer application
5. **B**'s file transfer application asks the file system to write the data on its storage

End to End Caretaking -- A Careful File Transfer



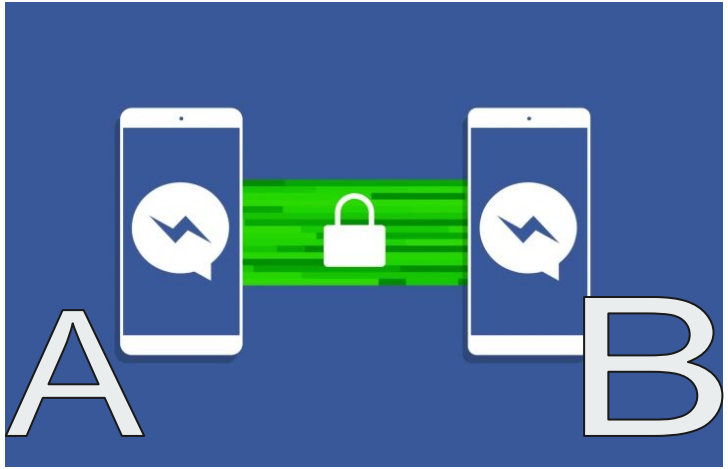
1. **A** correctly read file from the storage (disk)
2. **A's** file transfer program asks the data communication system to transmit the file (split the data into packets)
3. Data communication network moves packets from **A** to **B**
4. **B** removes the packets and hands the data to **B**'s file transfer application
5. **B**'s file transfer application asks the file system to write the data on its storage

End to End Caretaking -- A Careful File Transfer



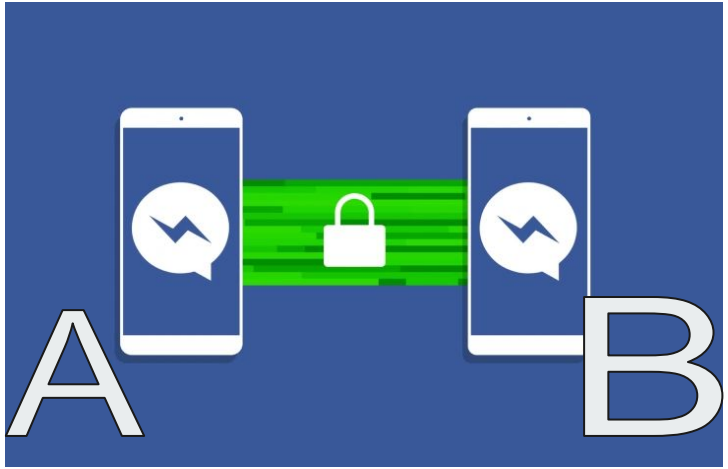
1. **A** correctly read file from the storage (disk)
2. **A**'s file transfer program asks the data communication system to transmit the file (split the data into packets)
3. Data communication network moves packets from **A** to **B**
4. **B** removes the packets and hands the data to **B**'s file transfer application
5. **B**'s file transfer application asks the file system to write the data on its storage

End to End Caretaking -- A Careful File Transfer



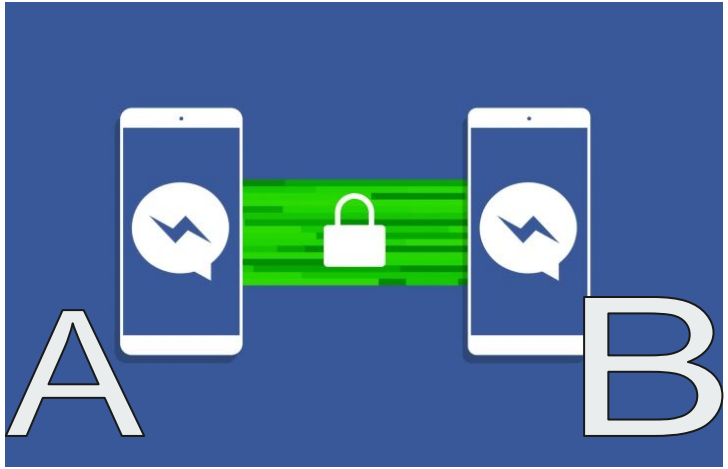
1. **A** correctly read file from the storage (disk)
2. **A's** file transfer program asks the data communication system to transmit the file (split the data into packets)
3. Data communication network (hardware processor or its local memory) moves packets from **A** to **B**
4. **B** removes the packets and hands the data to **B**'s file transfer application
5. **B**'s file transfer application asks the file system to write the data on its storage

End to End Caretaking -- A Careful File Transfer



1. **A** correctly read file from the storage (disk)
2. **A's** file transfer program asks the data communication system to transmit the file (split the data into packets)
3. Data communication network moves packets from **A** to **B**
4. **B** removes the packets and hands the data to **B**'s file transfer application
5. **B**'s file transfer application asks the file system to write the data on its storage

End to End Caretaking -- A Careful File Transfer



1. **A** correctly read file from the storage (disk)
2. **A's** file transfer program asks the data communication system to transmit the file (split the data into packets)
3. Data communication network moves packets from **A** to **B**
4. **B** removes the packets and hands the data to **B**'s file transfer application
5. **B**'s file transfer application asks the file system to write the data on its storage

A too-real Example

- One network system involving several local networks connected by gateways used a **packet checksum** on each hop from one gateway to the next, on the assumption that the primary threat to correct communication was corruption of bits during transmission.
- Application programmers assumed that the network was providing reliable transmission
- One gateway computer developed a transient error **while copying data**
- Over a period of time many of the source files of an operating system were repeatedly transferred through the defective gateway. Some of these source files were corrupted by byte exchanges.

Performance Aspects

The simple strategy: transmitting the file and then checking to see that the file has arrived correctly, would perform more poorly as the length of the file increased.

The key idea here is that the lower levels need not provide "perfect" reliability. ⇒ Engineering **trade-off** based on performance, rather than a requirement for correctness.

The end-to-end check of the file transfer application must still be implemented no matter how reliable the communication system becomes.

A great deal of information about system implementation is needed to make this choice intelligently.

Other Examples: Delivery Guarantee

1. Acknowledgement once it reaches the host by replying an acknowledgement message. Eg: APRANET-RFNM
    a. We don't know if the data reaches the issuing request application
2. Target host is sophisticated enough to accept delivery of data and to be responsible of delivering it to the targeted application
    a. an end-to-end acknowledgment may still be a requirement.

Other Examples: Secure transmission of data

1. If the data transmission system performs encryption and decryption, it must be trusted to manage securely the required encryption keys.
2. The data will be in the clear when it reaches the target applications
3. The authenticity of the message must still be checked by the applications

E2E ⇒ check and handle key management to protect the data

- Automatic encryption is one more firewall

Other Examples: Duplicate Message Suppression

1. A message or a part of a message may be delivered twice, typically as a result of a time-out-triggered failure detection and retry mechanisms operating within the network
2. The network can provide the function of watching for and suppressing any such deplication message
3. HOWEVER, even if the network suppress duplicates, the application itself may accidentally originate duplicate requests, in its own failure/retry procedures
4. Network can’t suppress application-level duplicates; suppression must be accomplished by the application itself with knowledge of how to detect its own duplicates.

Other Examples: Guaranteeing FIFO message delivery

1. Ensuring that message arrive at the receiver in the same order they were sent is another function usually assigned to the communication subsystem
2. For the same virtual circuit: easy! For independent virtual circuit: Not so easy
3. An independent mechanism at a higher level than the communication subsystem must control the ordering of actions

Other Examples: Transaction Management

1. Applied the end-to-end argument in the construction of the SWALLOW distributed data storage system, where it leads to significant reduction in overhead
2. The low level message communication protocol is significantly simplified
3. The acknowledgment that the originator of a write request needs is that the data was stored safely
4. No acknowledgement for read operations which boasts its performance.

Identifying the Ends

The end-to-end argument is not an absolute rule, but rather a guideline that helps in application and protocol design analysis;

One must use some care to identify the endpoints to which the argument should be applied.

History and application to other system areas

- Examples discussed are related in earlier papers
- End-to-end arguments are often applied to error control and correctness in application systems
- Banks and sensitive system may need to use the end-to-end to save them MONEY
- A version of the end-to-end argument in a non-communication application was developed in the 1950s

Conclusion

Designer may be tempted to “help” the users by taking on more function than necessary in the communication. **Awareness of end-to-end arguments can help to reduce such temptations.**

Looking for layered communication protocols, but without clearly defined criteria for assigning functions to layers. **Such layerings are desirable to enhance modularity.**

End-to-end arguments may be viewed as part of **a set of rational principles** for organizing such layered systems.

Pre-lecture

Q2: Do you think the end-to-end argument has much utility today?

YES:

- Ensure security & privacy
- More false-tolerant

In between:

- Should put more trust in the network

NO:

- Service like streaming or real-time gaming
- Multiple endpoints / blockchain
- Different endpoints may share same IP address
- Hard to do global optimization.

DISCUSSION: https://tinyurl.com/cse550au21-network

1. What are some of the best use cases for End-to-End argument?
2. The author argued that “The end-to-end check of the “file transfer application” must still be implemented no matter how reliable the communication system becomes.” Do you agree with this?
3. What are some of the risks with existing reusable modules?
4. What are some of the techniques to automatically search for boundaries?