Communication Detection and Data Transfer Event Synchronization from Dual Trace

by

Huihui Nora Huang
B.Sc., Nanjing University of Aeronautics and Astronautic, 2003
M.Sc., Nanjing University of Aeronautics and Astronautic, 2006

A Dissertation Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

in the Department of Computer Science

by

Huihui Nora Huang
B.Sc., Nanjing University of Aeronautics and Astronautic, 2003
M.Sc., Nanjing University of Aeronautics and Astronautic, 2006

Supervisory Committee

Dr. German. Supervisor Main, Supervisor
(Department of Same As Candidate)

Dr. M. Member One, Departmental Member
(Department of Same As Candidate)

Dr. Member Two, Departmental Member
(Department of Same As Candidate)

Dr. Outside Member, Outside Member
(Department of Not Same As Candidate)

Supervisory Committee

Dr. German. Supervisor Main, Supervisor
(Department of Same As Candidate)

Dr. M. Member One, Departmental Member
(Department of Same As Candidate)

Dr. Member Two, Departmental Member
(Department of Same As Candidate)

Dr. Outside Member, Outside Member
(Department of Not Same As Candidate)

ABSTRACT

Contents

List of Tables

List of Figures

ACKNOWLEDGEMENTS

I would like to thank:

my cat, Star Trek, and the weather, for supporting me in the low moments.

Supervisor Main, for mentoring, support, encouragement, and patience.

Grant Organization Name, for funding me with a Scholarship.

I believe I know the only cure, which is to make one's centre of life inside of one's self, not selfishly or excludingly, but with a kind of unassailable serenity-to decorate one's inner house so richly that one is content there, glad to welcome any one who wants to come and stay, but happy all the same in the hours when one is inevitably alone.

Edith Wharton

DEDICATION

Just hoping this is useful!

Chapter 1

Modeling

I investigated some common used communication methods divide the communication methods into two categories based on their data transmission properties. Based on this investigation, I modelled the communication of two programs. I also the dual-trace of two communicating programs in the perspective of communication analysis. These two models are the foundation to decide how communications being identified from the dual-trace and how to present them to the user.

1.1 Communication Categorization and Communication Methods

The goal of this work is to identify the communications from the dual-trace. We need to understand the properties of the communications to identify them. In general, there are two types of communication: reliable and unreliable in the perspective of their reliability of data transmission. The reason to divide the communication methods into these two categories is that the data transmission properties affect the mechanism of the identification of the communications fall in different categories. In the following two subsections, I summarize the characteristics of these two communication categories. The communication methods list in Table1.1 will be discussed further to provide more concrete comprehension.

Table 1.1: Communication Methods Discussed in This Work

Reliable Communication	**Unreliable Communication**
Named Pipes	Message Queue
TCP	UDP

1.1.1 Reliable Communication

A reliable communication guarantees the data being sent by one endpoint of the channel always received lossless and in order to the other endpoint. With this property, the send data union in the send stream of one endpoint should equal to the receive data union in the receive stream of the other endpoint. The send data union is the conjunction of the data trunks in all send events in the send stream by the event time ordering. The receive data union is the conjunction of the data trunks in all receive events in the receive stream by the event time ordering. Therefore, the send and receive data verification should be in send and receive stream level by comparing the send data union of one endpoint to the receive data union of another. For some communication methods, a channel can be closed without waiting the completion of all data transmission. In this case, the receive data union can be a sub string of the send data union.

1.1.2 Unreliable Communication

An unreliable communication does not guarantee the data being send always arrive the receiver. Moreover, the data packets can arrive to the receiver in any order. However, the bright side of unreliable communication is that the packets being sent are always arrived as the origin packet, no data re-segmentation would happen. Accordingly, the send and receive data verification should be done by matching the data packets in a send event to a receive event on the other side.

1.1.3 Communication Methods

In this section, I describe the mechanism and the basic data transfer characteristics of each communication method in Table1.1 briefly. Moreover, data transfer scenarios are represented correspondingly in diagrams for each communication method.

Named Pipe

In computing, a named pipe provides FIFO communication mechanism for inter-process communication. It allows two programs send and receive message through the named pipe.

The basic data transfer characteristics of Named Pipe are:

- Bytes received in order
- Bytes sent as a whole trunk can be received in segments
- No data duplication

- Only the last trunk can be lost

Based on these characteristics, the data transfer scenarios of Named pipe can be summarized in Figure1.1.

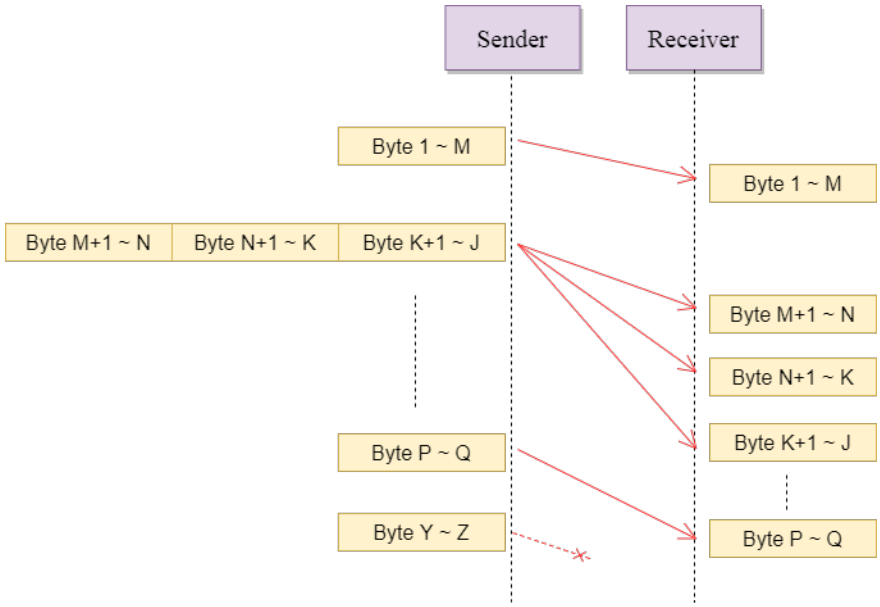

Figure 1.1: Data Transfer Scenarios for Named Pipe

Message Queue

Message Queuing (MSMQ) is a communication method to allow applications which are running at different times across heterogeneous networks and systems that may be temporarily offline can still communicate with each other. Messages are sent to and read from queues by applications. Multiple sending applications can send messages to and multiple receiving applications can read messages from one queue.[8] The applications are the endpoints of the communication. In this work, only one sending application versus one receiving application case is considered. Multiple senders to multiple receivers scenario can always be divided into multiple sender and receiver situation. Both endpoints of a communication can send to and receive from the channel.

The basic data transfer characteristics of Message Queue are:

- Bytes sent in packet and received in packet, no bytes re-segmented
- Packets can lost
- Packets received in order

- No data duplication

Based on these characteristics, the data transfer scenarios of Message Queue can be summarized in Figure1.2.

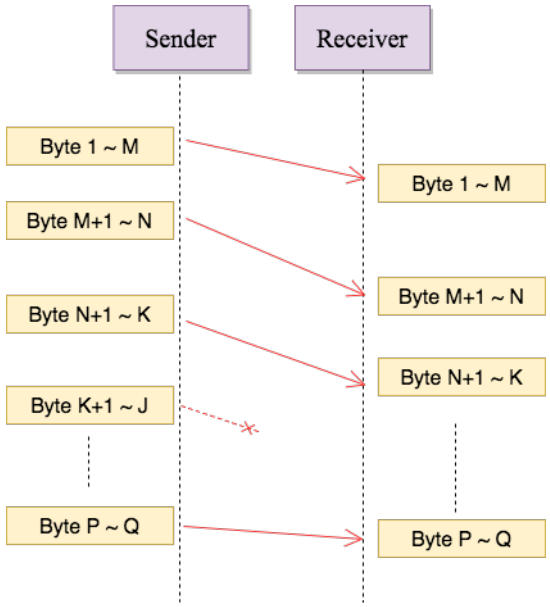

Figure 1.2: Data Transfer Scenarios for Message Queue

TCP

TCP is the most fundamental reliable transport method in computer networking. TCP provides reliable, ordered, and error-checked delivery of a stream of octets between applications running on hosts in an IP network. The TCP header contains the sequence number of the sending octets and the acknowledge sequence this endpoint is expecting from the other endpoint(if ACK is set). The retransmission mechanism is based on the ACK.

The basic data transfer characteristics of TCP are:

- Bytes received in order
- No data lost(lost data will be retransmitted)
- No data duplication
- Sender window size is different from receiver's window size, so packets can be re-segmented

Based on these characteristics, the data transfer scenarios of TCP can be summarized in Figure1.3.

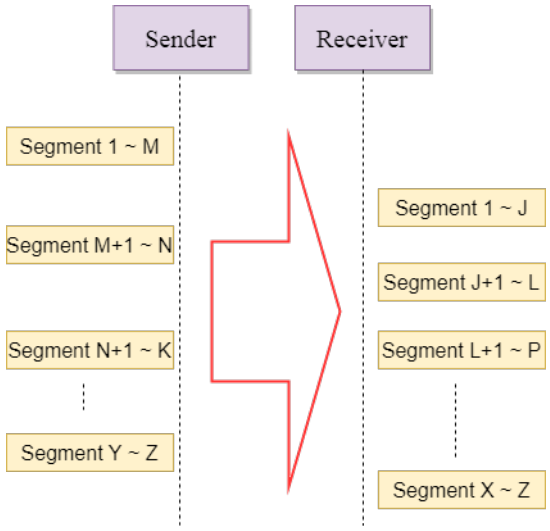

Figure 1.3: Data Transfer Scenarios for TCP

UDP

UDP is a widely used unreliable transmission method in computer networking. It is a simple protocol mechanism, which has no guarantee of delivery, ordering, or duplicate protection. This transmission method is suitable for many real time systems.

The basic data transfer characteristics of UDP are:

- Bytes sent in packet and received in packet, no re-segmentation
- Packets can lost
- Packets can be duplicated
- Packets can arrive receiver out of order

Based on these characteristics, the data transfer scenarios of UDP can be summarized in Figure1.4.

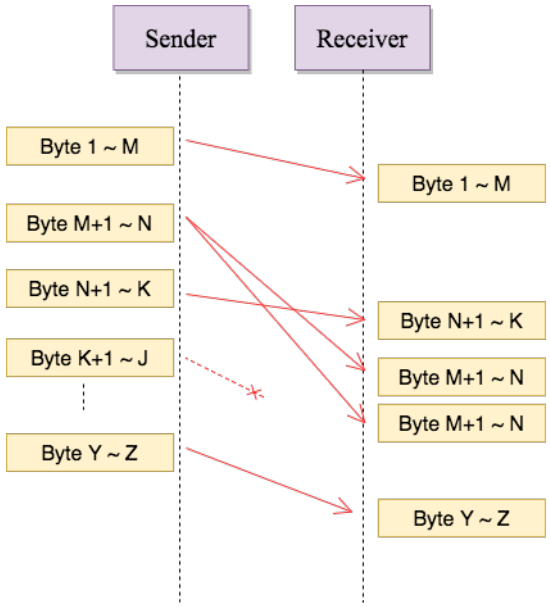

Figure 1.4: Data Transfer Scenarios for UDP

1.2 Model of Communication

This section define the communication of two programs. The communication in this work is data transfer activities between two running programs through a specific channel. Some collaborative activities between the programs such as remote procedure call is out of the scope of this research. Communication among multiple programs (more than two) is not discussed in this work. The channel can be reopened again to start new communications after being closed. However, the re-opened channel will be treated as a new communication. The way that I define the communication is leading to the communication identification in the dual-trace. So the definition is not about how the communication works but what it looks like. There are many communication methods for the data transfer communications, but all of them are compatible to this communication definition. XXX to XXX are the formal definition of a communication. The terminology of the elements can be found in AppendixA A.1

Figure1.5 shows an example of a communication according to this definition.

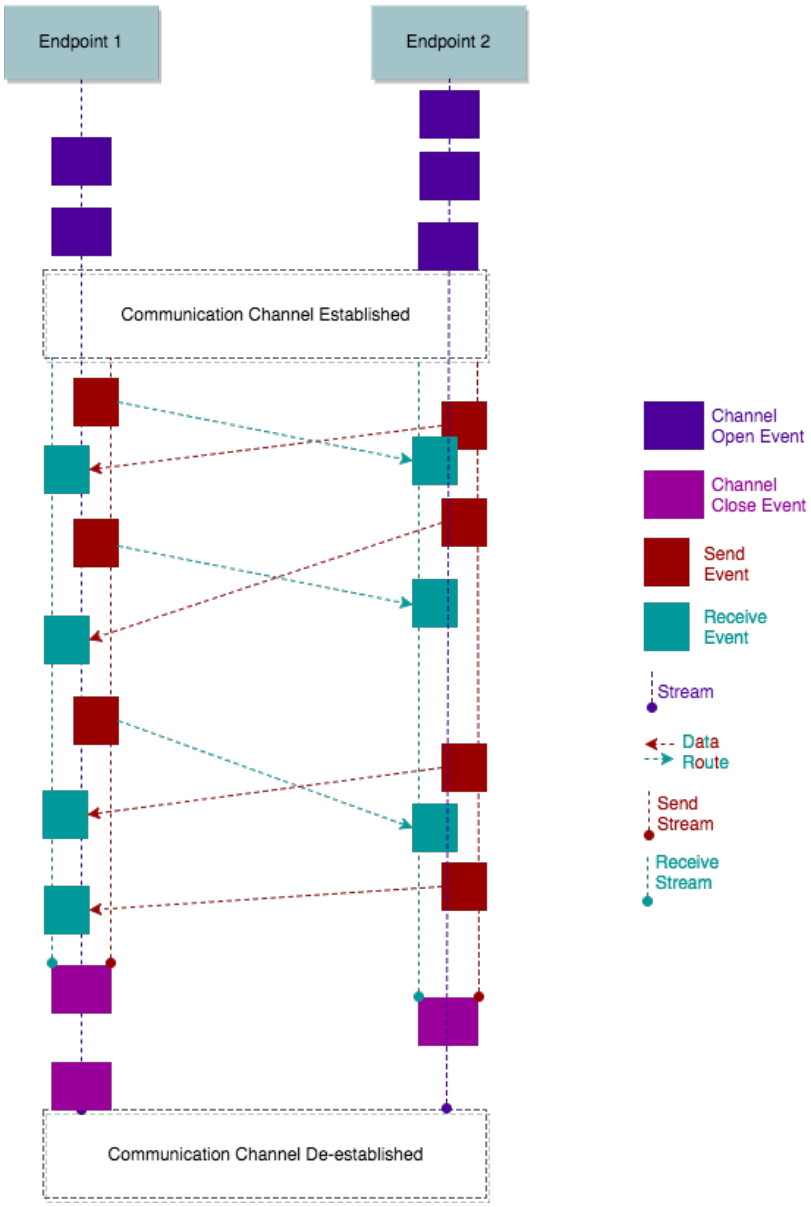

Figure 1.5: General Communication Model

1.3 Model of the Dual-Trace of Two Communicating Programs

Appendix A

Additional Information

A.1 Terminology

Endpoint:
An instance in a program at which a stream of data are sent or received (or both). It usually is identified by the handle of a specific communication method in the program. Such as a socket handle of TCP or UDP or a file handle of the named piped channel.
Channel:
A conduit connected two endpoints through which data can be sent and received
Channel open event:
Operation to create and connect an endpoint to a specific channel
Channel close event:
Operation to disconnect and delete the endpoint from the channel.
Send event:
Operation to send a trunk of data from one endpoint to the other through the channel.
Receive event:
Operation to receive a trunk of data at one endpoint from the other through the channel.
Channel open stream:
A set of all channel open events regarding to a specific endpoint.
Channel close stream:
A set of all channel close events regarding to a specific endpoint.
Send stream:
A set of all send events regarding to a specific endpoint.
Receive stream:

A set of all receive events regarding to a specific endpoint.

Stream:

A stream consist of a channel open stream, a channel close stream, a send stream and a receive stream. All of these streams regard to the same endpoint.

Bibliography

[1] B. Cleary, P. Gorman, E. Verbeek, M. A. Storey, M. Salois, and F. Painchaud. Reconstructing program memory state from multi-gigabyte instruction traces to support interactive analysis. In *2013 20th Working Conference on Reverse Engineering (WCRE)*, pages 42–51, October 2013.

[2] Mark Dowd, John McDonald, and Justin Schuh. *Art of Software Security Assessment, The: Identifying and Preventing Software Vulnerabilities*. Addison-Wesley Professional., 1st edition, November 2006.

[3] Chris Eagle. *The IDA Pro Book: The Unofficial Guide to the World's Most Popular Disassembler*. No Starch Press, San Francisco, CA, USA, 2008.

[4] Huihui Nora Huang, Eric Verbeek, Daniel German, Margaret-Anne Storey, and Martin Salois. Atlantis: Improving the analysis and visualization of large assembly execution traces. In *Software Maintenance and Evolution (ICSME), 2017 IEEE International Conference on*, pages 623–627. IEEE, 2017.

[5] Intel. Pin - A Dynamic Binary Instrumentation Tool | Intel Software.

[6] School of Computing) Advisor (Prof. B. Kang) KAIST CysecLab (Graduate School of Information Security. c0demap/codemap: Codemap.

[7] MultiMedia LLC. Named pipes (windows), 2017.

[8] Arohi Redkar, Ken Rabold, Richard Costall, Scot Boyd, and Carlos Walzer. *Pro MSMQ: Microsoft Message Queue Programming*. Apress, 2004.

[9] Chao Wang and Malay Ganai. Predicting Concurrency Failures in the Generalized Execution Traces of x86 Executables. In *Runtime Verification*, pages 4–18. Springer, Berlin, Heidelberg, September 2011. DOI: 10.1007/978-3-642-29860-8_2.