Improving Internet Mobility: The TCP-to-SCTP Translation Layer

CSE 222A – Spring 2007

June 13, 2007

Varun Almaula, Taurin Tan-atichat, Thomas Weng

Introduction

- Current implementations of TCP and IP are limited in terms of providing for mobility.
- TCP only allows for an endpoint to endpoint connection, where an endpoint is a specific IP address / port on a computer. No way of keeping track of multiple IP addresses in a connection.
- Solutions
  - MobileIP (requires special routers, triangle routing, focuses on client mobility)
  - Vertical Handoff (only one active interface at a time, requires special/complex monitoring processes, application layer sessions may not be preserved)
  - Using DNS to hide location (focused on multi-homing - not mobile clients, many DNS updates)

Introduction

- Stream Control Transmission Protocol (SCTP), an official transport layer protocol (natively supported by Linux) was standardized in 2000 that provides support for multi-homing, in addition to other features.

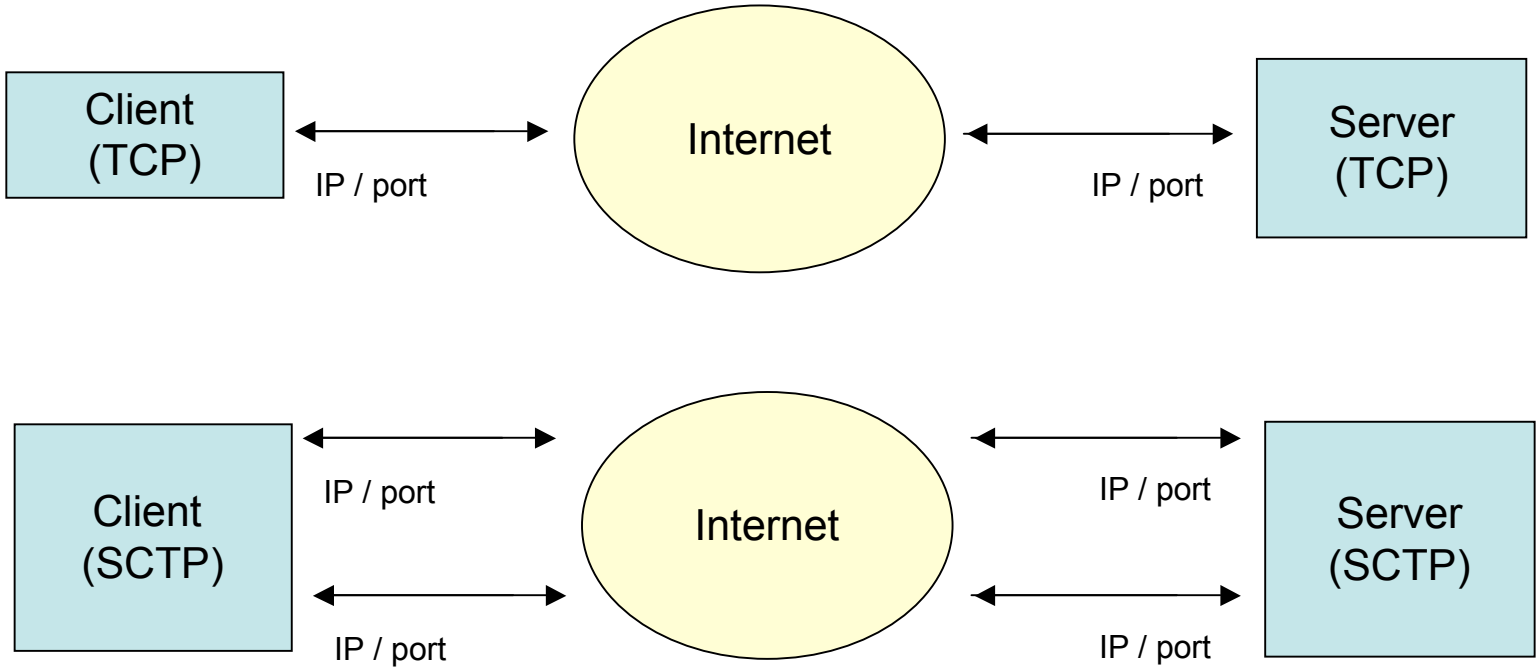
- Our research project involved designing and implementing a translation layer to convert TCP into SCTP calls, and to intelligently adapt and configure optimal settings on the fly to take advantage of SCTP.

Table of Contents

Description of SCTP

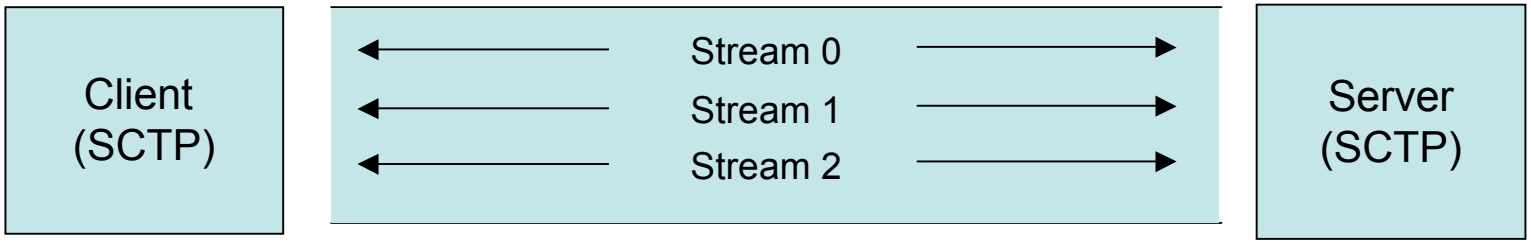
- SCTP is a transport layer protocol, like TCP and UDP, that provides end-host multi-homing, multi-streaming, and message framing capabilities.

- Multi-homing – ability to use multiple network interface, and hence IP addresses, in a connection between two end-hosts. Actually called an association, it allows multiple IP addresses instead of TCP's one.

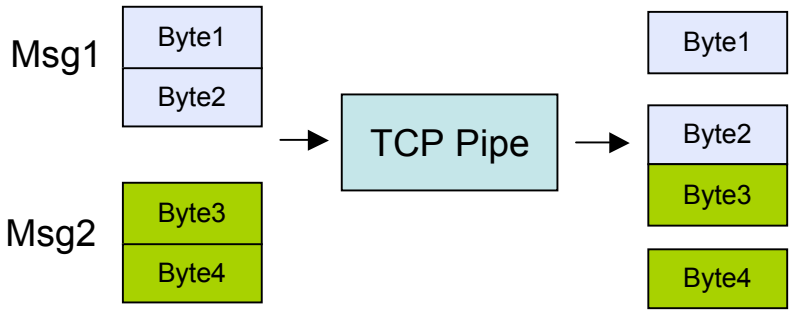


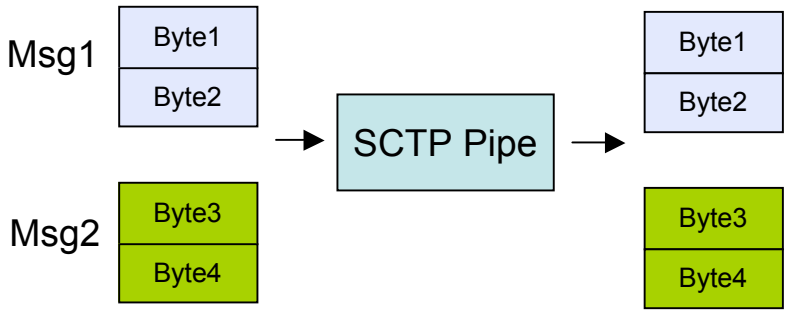
Description of SCTP

- Multi-Streaming – Each association can support multiple streams. Each stream is similar to a single TCP connection.



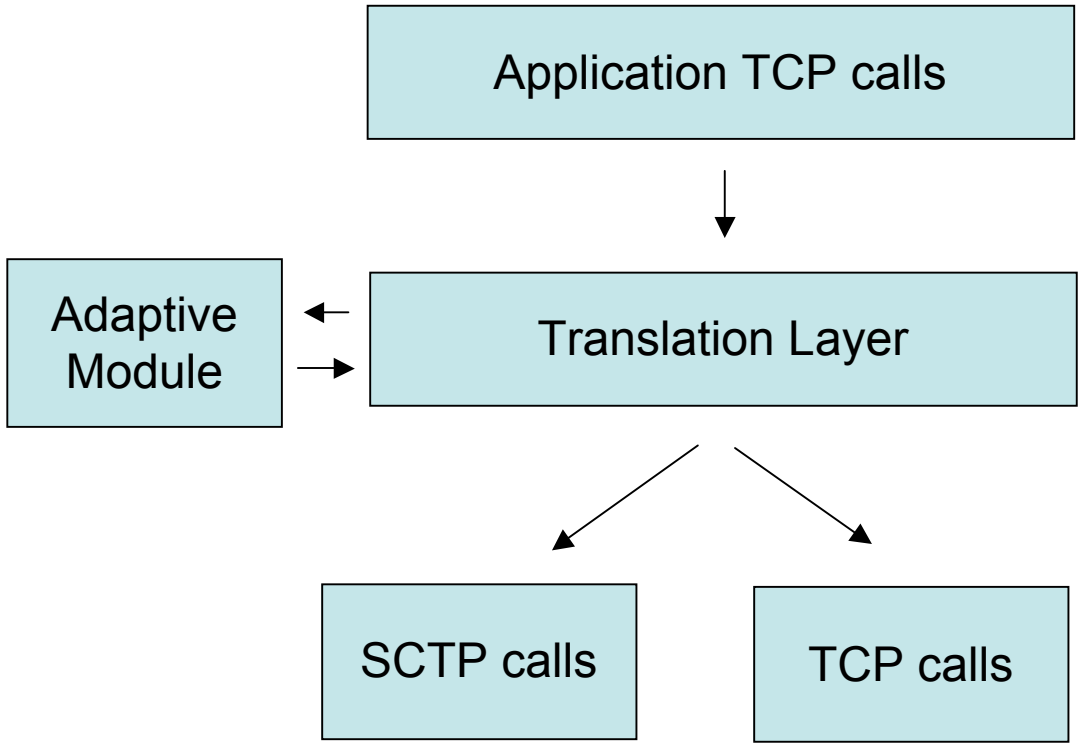
- Message Framing – Instead of sending bytes through, like TCP, SCTP will send whole messages through. This makes SCTP more of a message pipe, as opposed to a byte pipe.





SCTP Translation Layer

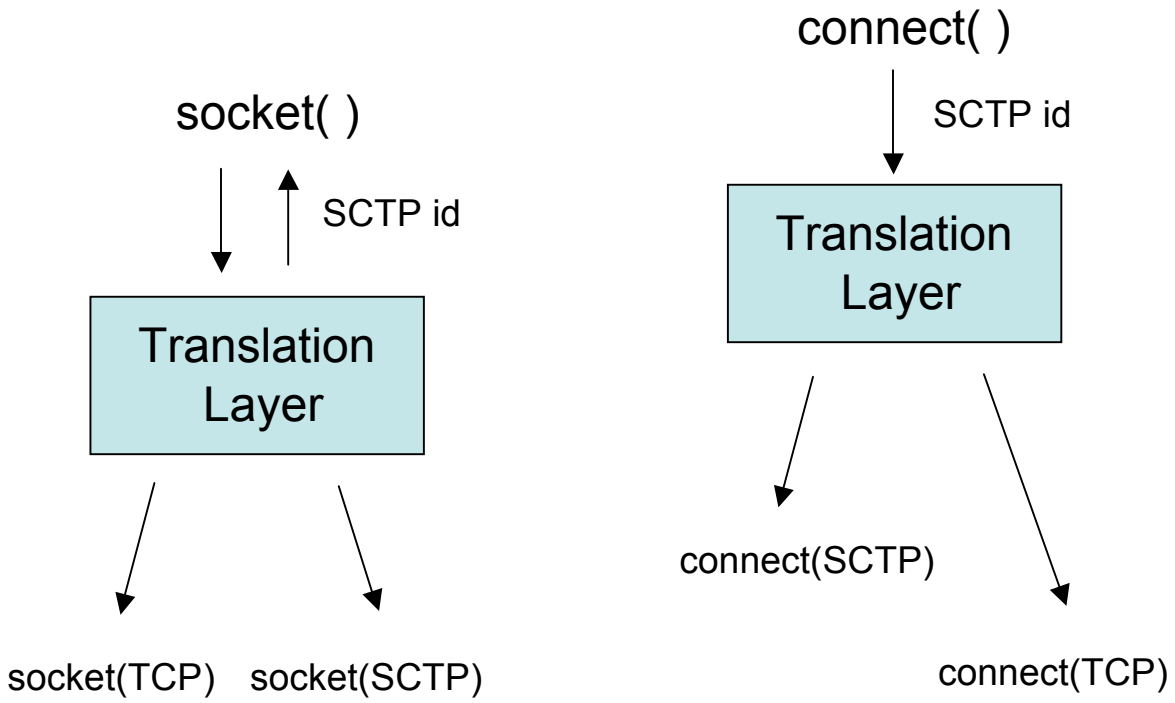
- The TCP-to-SCTP translation layer intercepts TCP commands from a TCP application.
- Intelligently converts TCP commands to SCTP commands, if possible, while at the same time ensuring that TCP still works too.



SCTP Translation Layer – Client calls

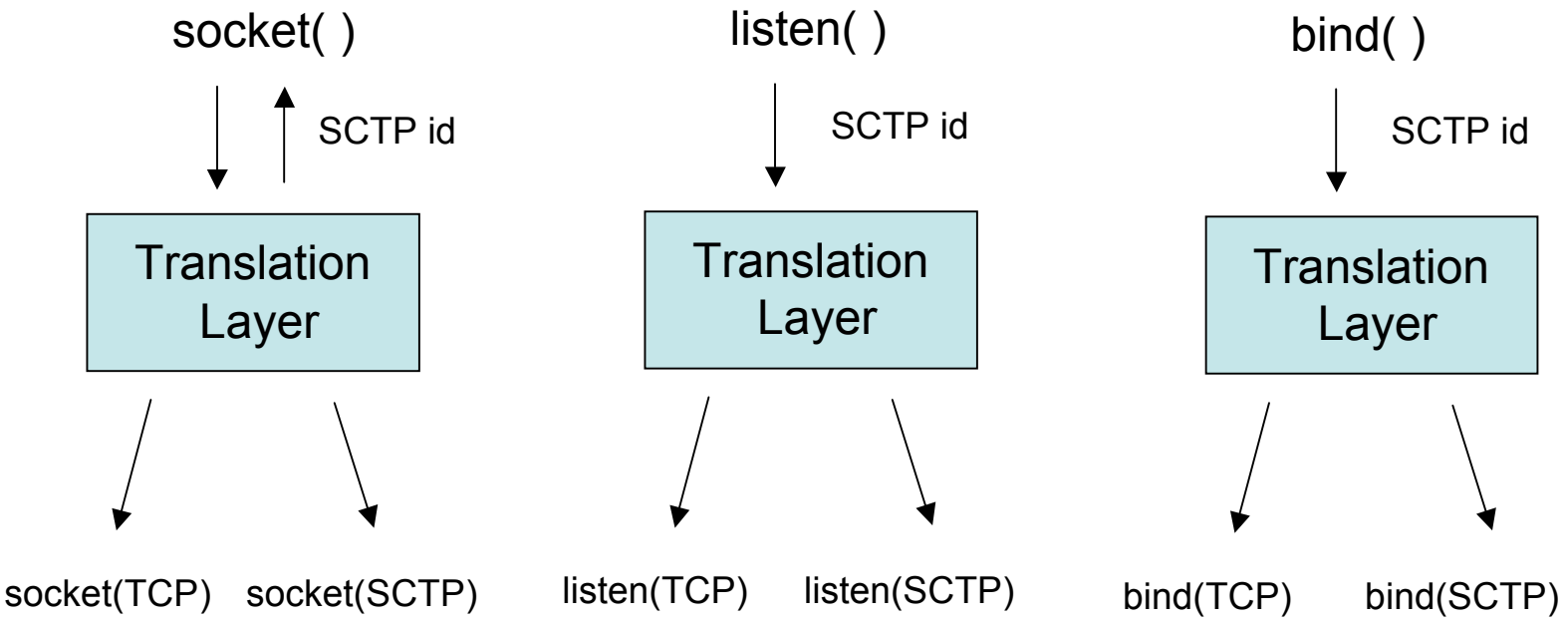
- For the TCP client, both TCP and SCTP sockets are created. The SCTP socket ID is returned. This value is used by the translation layer to keep track of the TCP/SCTP socket pair for that socket ID.

- Upon a connect, the SCTP socket is tried first. If no connection is made, then the TCP connect is done instead.



SCTP Translation Layer – Server calls

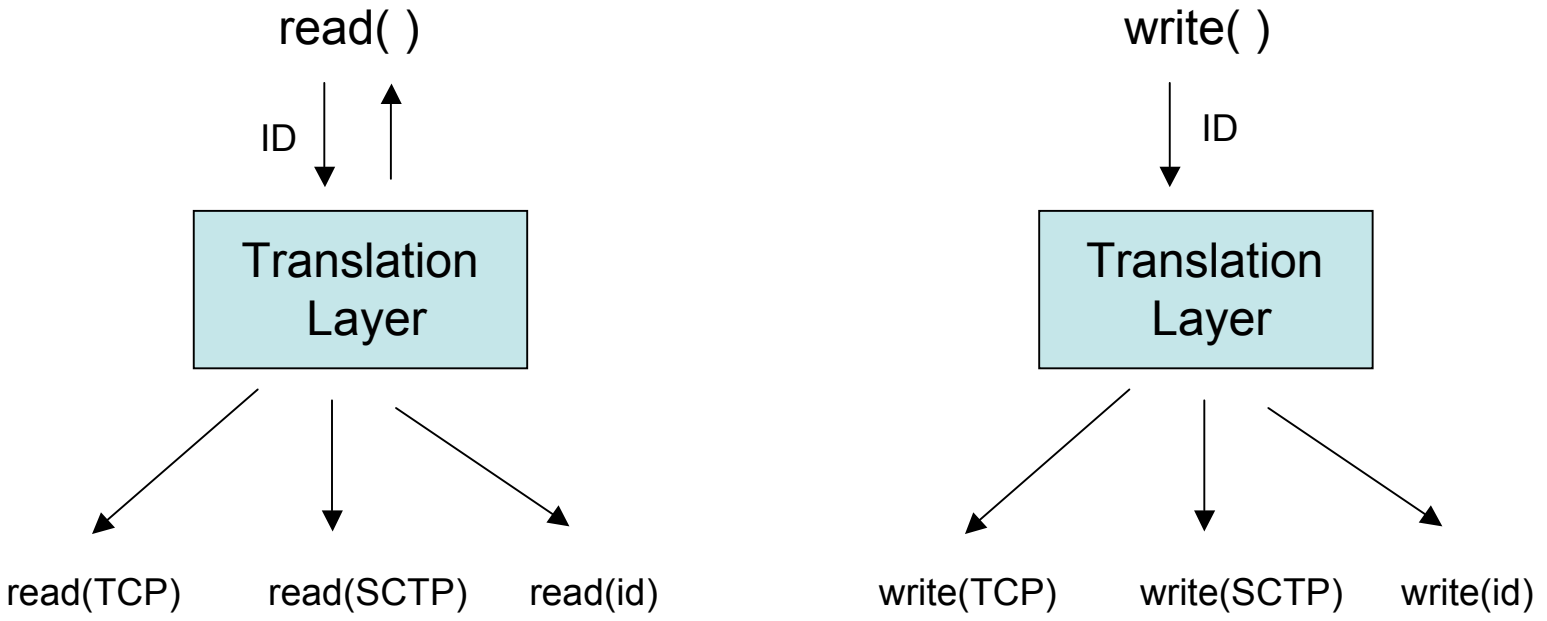
- For the TCP server, both TCP and SCTP sockets are created. The SCTP socket ID is returned. This value is used by the translation layer to keep track of the TCP/SCTP socket pair for that socket ID.
- Both sockets are maintained by the translation layer. Thus, both SCTP and TCP requests can connect to the server.



SCTP Translation Layer – Read and writes

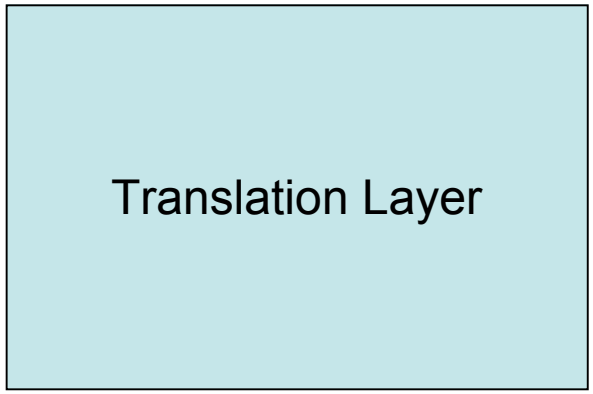
- Standard read, write calls have also been updated due to the need for supporting duo-sockets, in addition to send and recv calls.

- The translation layer records what mode is actively running for that SCTP id, either SCTP or TCP, and will do the appropriate call. If the ID is not the SCTP id, then it does a default call with the passed in ID.



SCTP Translation Layer – Translation layer

- The translation layer acts as the middle layer between the TCP application and the actual network API.
- Keeps track of the SCTP socket ID / TCP socket ID pairing. Keeps track of client and server addresses, which both can be multi-homed.
- Sets the default server IP address and the requested client IP address.



SID 3
SCTP socket ID = 3
TCP socket ID = 4

client IP addresses{
127.42.141.35:3241
192.31.15.146:3263}

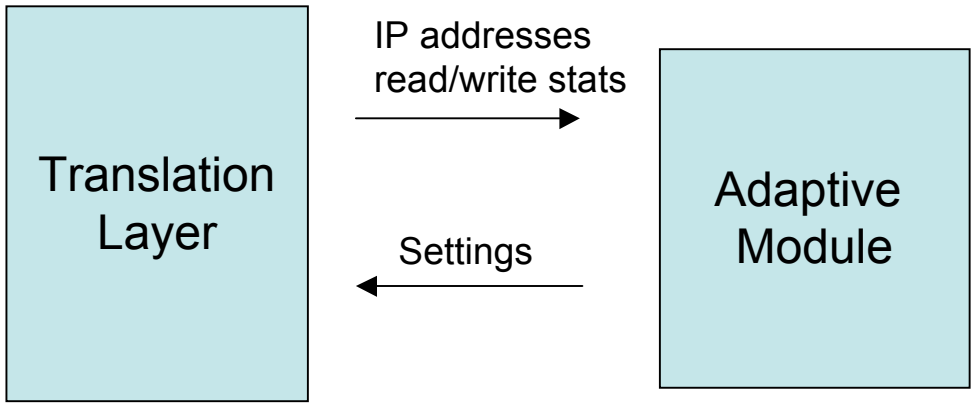
server IP addresses{
221.146.14.155:80
96.25.15.156:80}

Request server use IP address:
127.42.141.35:3241

Use server IP address:
96.25.15.156:80

SCTP Translation Layer – Adaptive module

- The adaptive module dynamically determines the optimum settings based on current usage patterns and saved information regarding this particular connection.
- Adaptive module records data for future default settings initiated by the translation layer.
- Also configures optimum settings (socket options) based on its known settings.



Peer IP Address 221.146.14.155:80

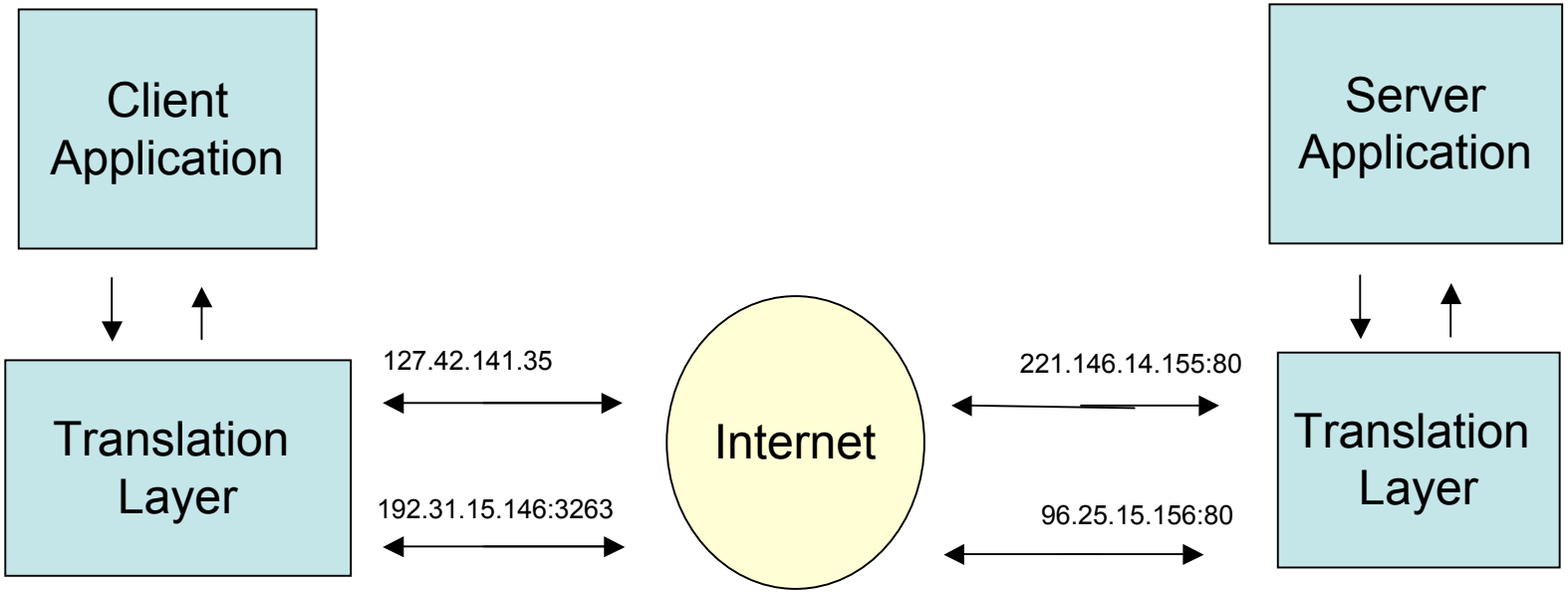
server IP addresses{
221.146.14.155:80
96.25.15.156:80}

best client IP address to use:
127.42.141.35

best server IP address to use:
221.146.14.155

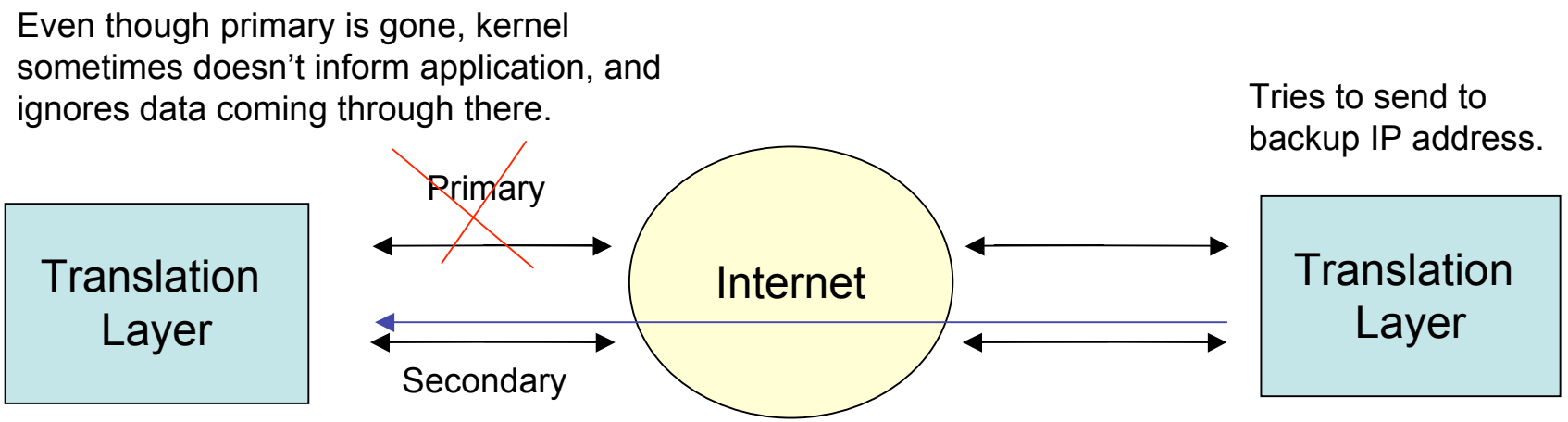
SCTP Translation Layer – Overall Functionality

- The original application attempts to make a TCP connection to a known server. Translation layer converts it into a SCTP connection instead if possible.
- Once connected via SCTP, applications can take advantage of SCTP's functionality, including multi-homing support, even though it originally used TCP.
- Able to disconnect a line from the client, and the server eventually sends on the other.



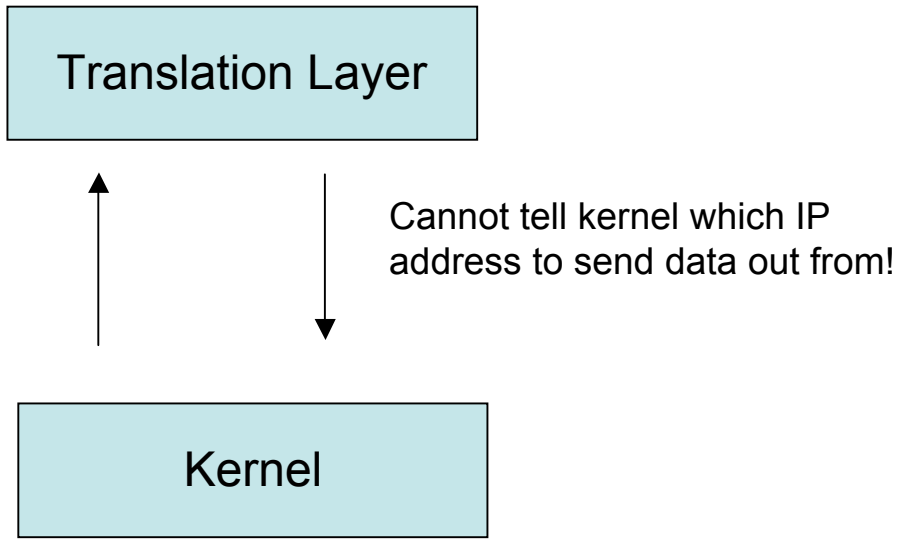
SCTP Translation Layer – Implementation Difficulties

- Linux kernel does not support multiple network interfaces with SCTP very well. Need to run Netplug.
- Multi-homing and secondary path selection sometimes did not work. Issues with the interface.
- Linux kernel needs to be updated to better take advantage of multi-homing with SCTP, despite natively supporting SCTP along with TCP and UDP (lksctp).



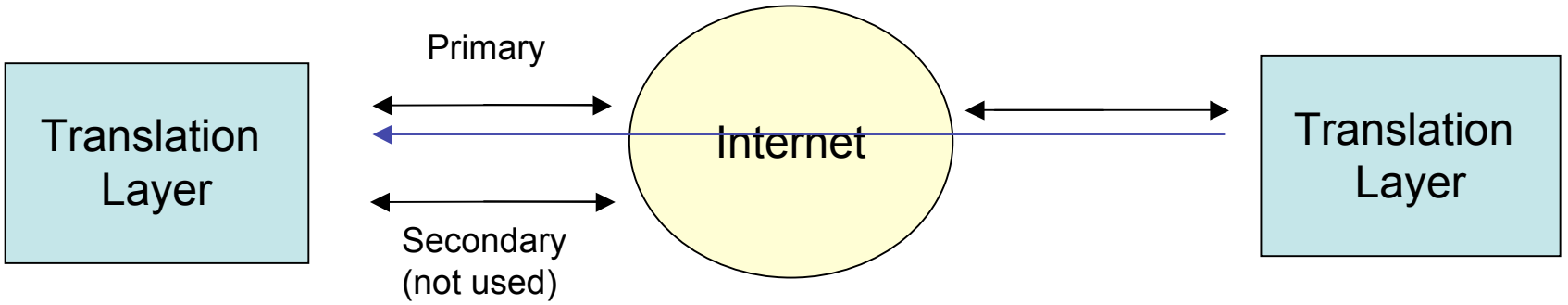
SCTP Translation Layer – Other Difficulties

- SCTP actually does not have enough functionality by itself for truly mobile computing.
- Mobile SCTP using ADDIP extension has support for adding and subtracting IP addresses dynamically. Not widely supported however, and wouldn't solve other issues.
- SCTP implementation in Linux does not have enough power over networking. Difficult to choose client IP address to use.



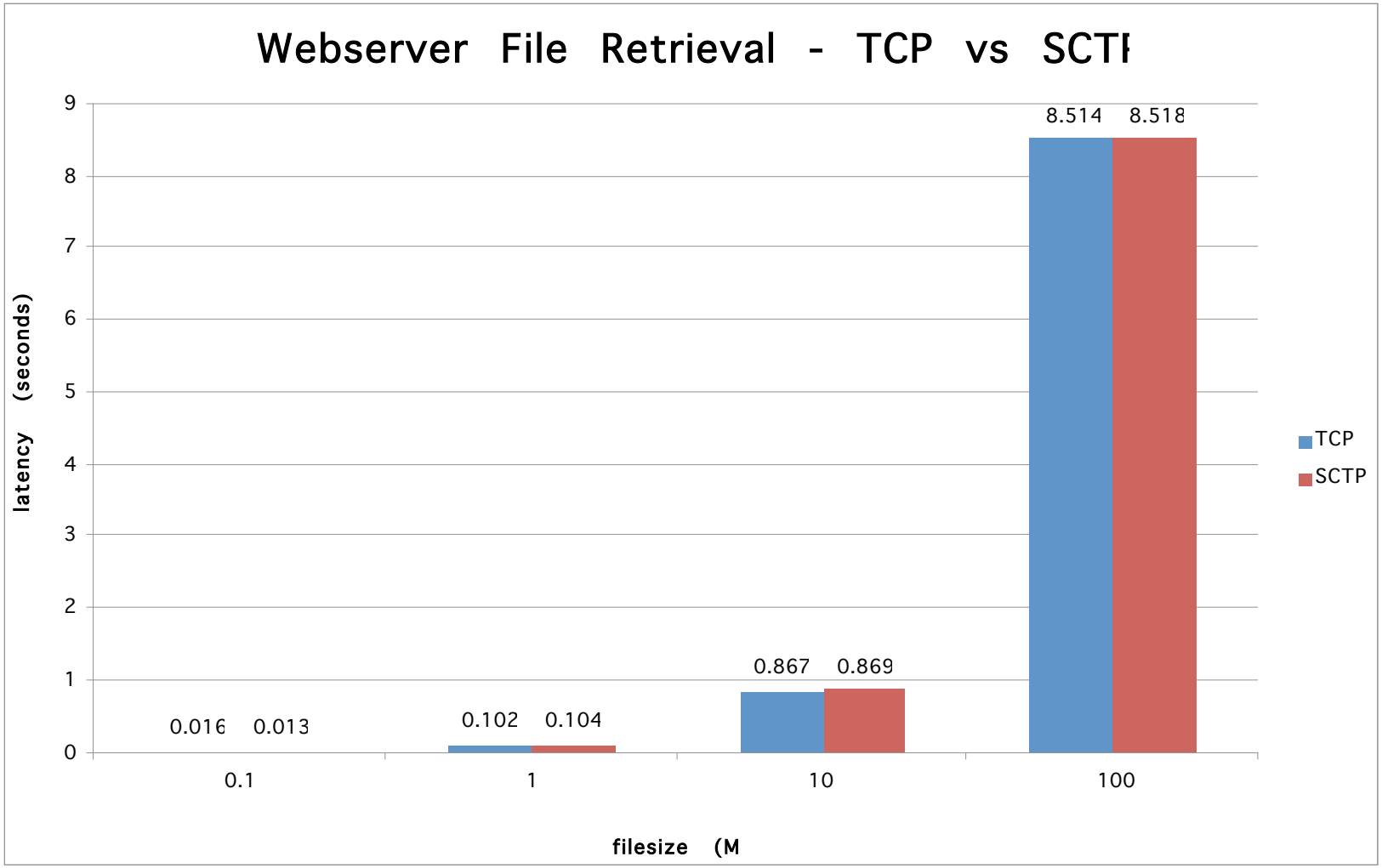
SCTP Translation Layer – Demonstration Details

- Implemented translation layer as a user level interface. Performed bandwidth tests versus TCP implementation.
- Adaptive module monitors connection, but right now does not preemptively change settings.
- Two sample programs, both using TCP calls, were attached to this translation layer. The server program is enabled to support both TCP and SCTP calls through this layer.
- The client program is enabled to take advantage of SCTP.



SCTP Translation Layer – Bandwidth Comparisons

- Tests between native TCP and our SCTP translation layer shows very similar performances.



SCTP Translation Layer – Conclusions and Future Research

- Designed and implemented a basic version of a translation layer that will convert TCP calls into SCTP calls and support SCTP functionality through use of the translation layer and adaptive module.

- Adaptive module (when implemented) had basic rules in terms of figuring out what IP address to use, and was able to keep track of traffic.

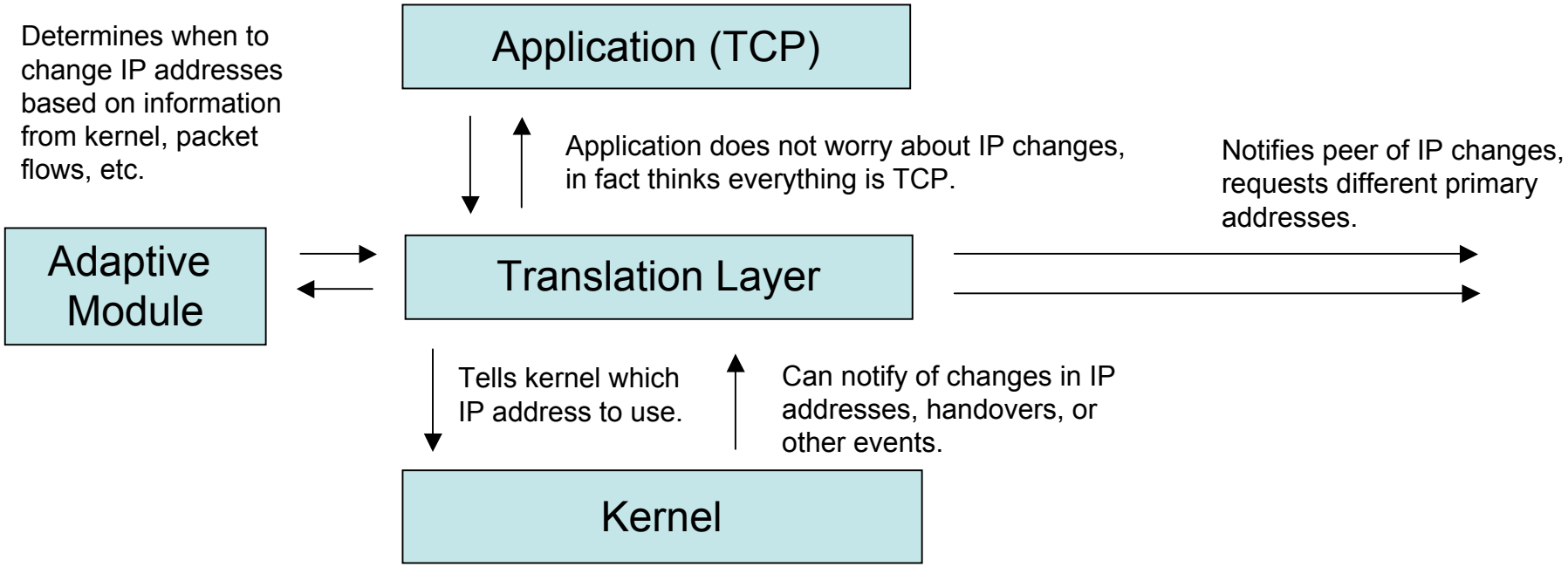
- Tested SCTP and TCP implementations of network code to compare throughput.

- Future research work:

  - Incorporate mSCTP into our translation layer.
  - Consider implementing translation layer in the kernel itself.
  - Modify Linux Kernel to better support SCTP.
  - Improved adaptive module functionality to best determine IP changes.
  - Better interface between SCTP calls and kernel support, decoupling network changes from the application.

SCTP Translation Layer – Conclusions and Future Research

- Goal is to separate application from being affected by network changes or IP changes.
- Translation layer and kernel communicate in terms of IP handoffs (vertical or horizontal), etc.
- Communicating SCTP peers notify of changes in IP addresses, including change in primary addressing.



Questions