z/TPF Communication Enhancements

Jamie Farmer

IBM **z/TPF**
April 3rd, 2017

Agenda

- Improve throughput when outbound packets are dropped in the network.
- Reduce MIPs consumed and increase overall throughput when sending “large” TCP/IP messages in a many way tightly coupled environment.
- Efficient and easy-to-use mechanism for TPF applications sending messages to remote servers.
- Reduce z/TPF application complexity and improve performance of reading large TCP messages.

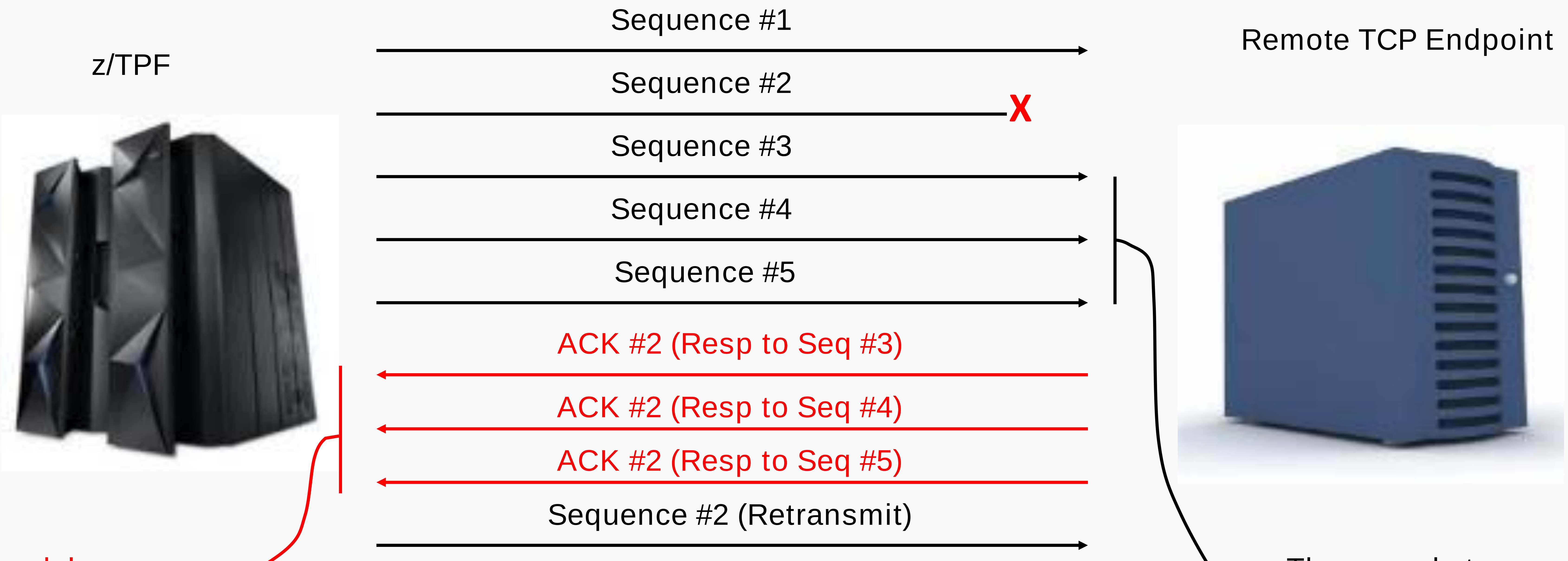
z/TPF Sub-Second Retransmission

The z/TPF system can recover from outbound packets dropped in the network in milliseconds as opposed to seconds improving overall throughput on the system.

As-Is Scenario

z/TPF Fast Retransmission

A pipe of packets sent from TPF to remote TCP endpoint
ie. MQ sender channel



z/TPF

Remote TCP Endpoint

Sequence #1

Sequence #2 X

Sequence #3

Sequence #4

Sequence #5

ACK #2 (Resp to Seq #3)

ACK #2 (Resp to Seq #4)

ACK #2 (Resp to Seq #5)

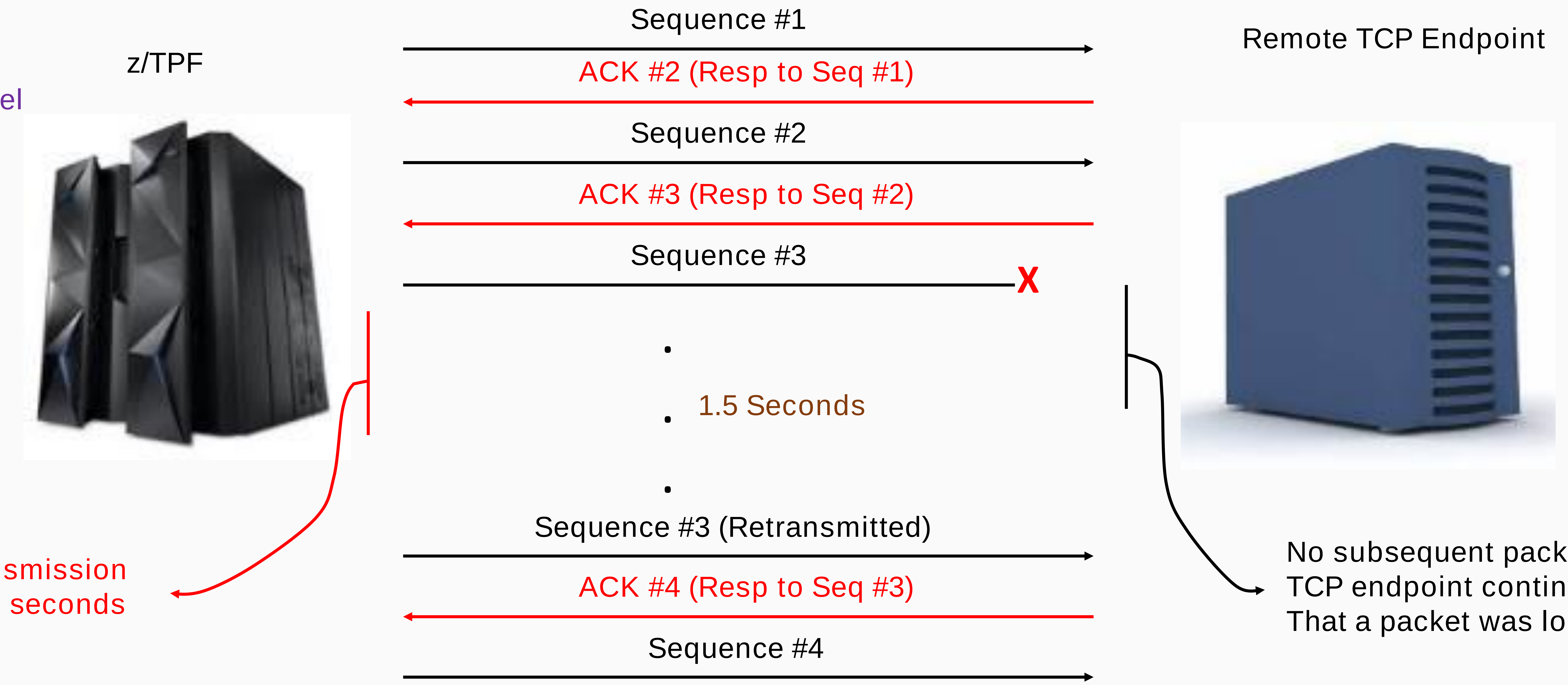
Sequence #2 (Retransmit)

These are duplicate standalone acknowledgments that trigger fast retransmission on z/TPF

These packets are considered out of order since Sequence #2 was dropped – this will generate Standalone ACKs from the remote TCP endpoint Indicating its waiting for Sequence #2

Fast retransmits will occur in roughly the round trip time for the socket connection.

z/TPF Retransmission Timeouts



z/TPF

Remote TCP Endpoint

Sequence #1

ACK #2 (Resp to Seq #1)

Sequence #2

ACK #3 (Resp to Seq #2)

Sequence #3

X

1.5 Seconds

Sequence #3 (Retransmitted)

ACK #4 (Resp to Seq #3)

Sequence #4

A request / reply model application between z/TPF and the remote TCP endpoint

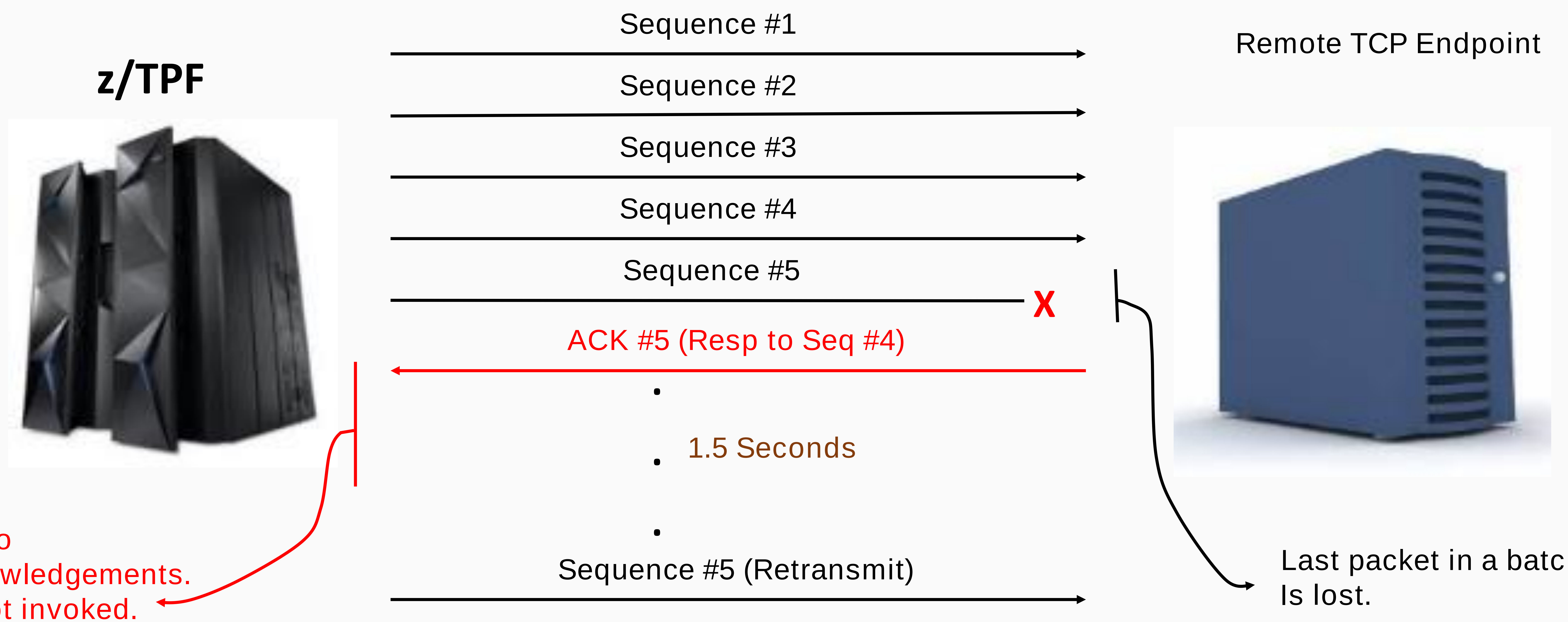
TPF waits for a retransmission timeout, between 1-2 seconds

No subsequent packets sent, remote TCP endpoint continues to wait. Unaware That a packet was lost.

z/TPF Retransmission Timeouts

Pipe Model

A pipe of messages sent from TPF to remote TCP endpoint:
 ie. MQ sender channel



z/TPF

Remote TCP Endpoint

Sequence #1

Sequence #2

Sequence #3

Sequence #4

Sequence #5

X

ACK #5 (Resp to Seq #4)

1.5 Seconds

Sequence #5 (Retransmit)

No subsequent packets to generate duplicate acknowledgements.
Fast retransmission is not invoked.
Timeout between 1-2 seconds.

Last packet in a batch of messages
Is lost.

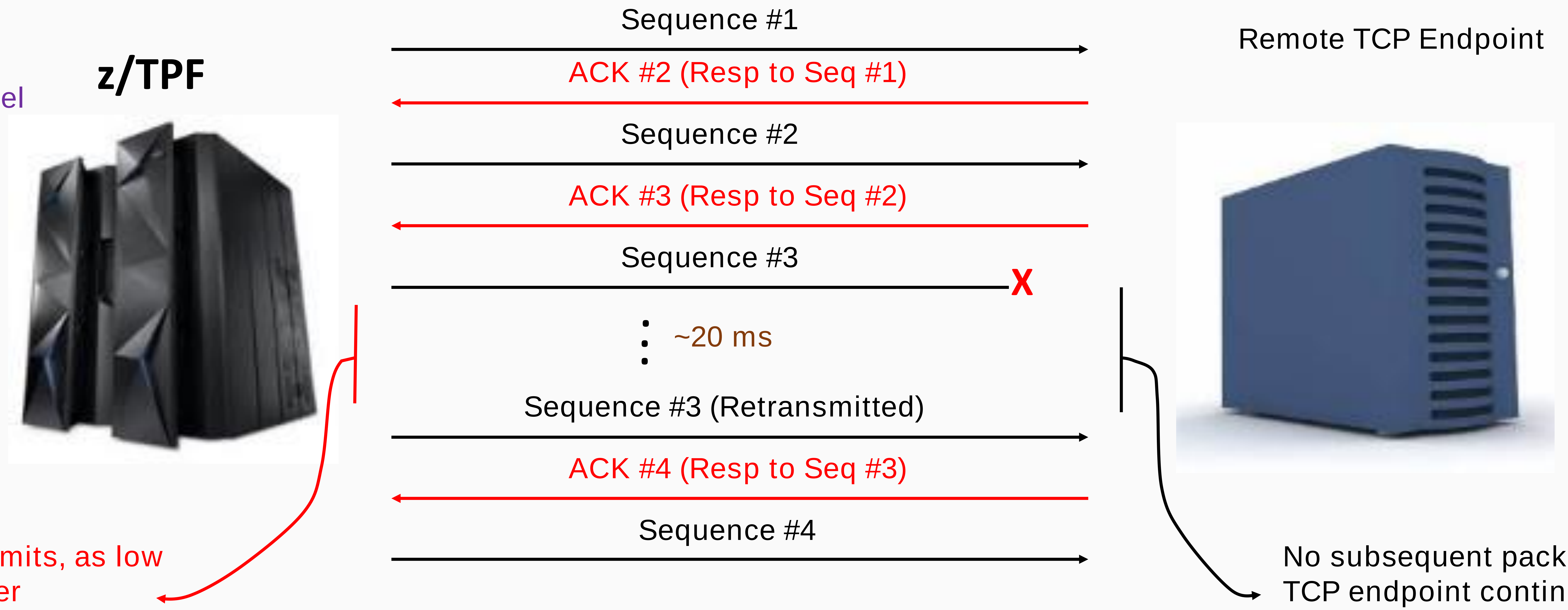
Sub-Second Retransmission Details

- Self tuning algorithm
  - Adjusts automatically based on smoothed Round Trip Time (RTT) and the variation of the Round Trip Time (RTTVAR)
- Calculated retransmission timeout (RTO)
  - The minimum RTO is 20 milliseconds
    - Lower than this you can see too many “spurious” retransmits
- Sub-Second Retransmission is automatically enabled when APAR is applied.
  - APAR PJ43958 (PUT 13)

To-Be Scenario

z/TPF Retransmission Timeouts

A request / reply model application between z/TPF and the remote TCP endpoint



z/TPF

Remote TCP Endpoint

Sequence #1

ACK #2 (Resp to Seq #1)

Sequence #2

ACK #3 (Resp to Seq #2)

Sequence #3

X

~20 ms

Sequence #3 (Retransmitted)

ACK #4 (Resp to Seq #3)

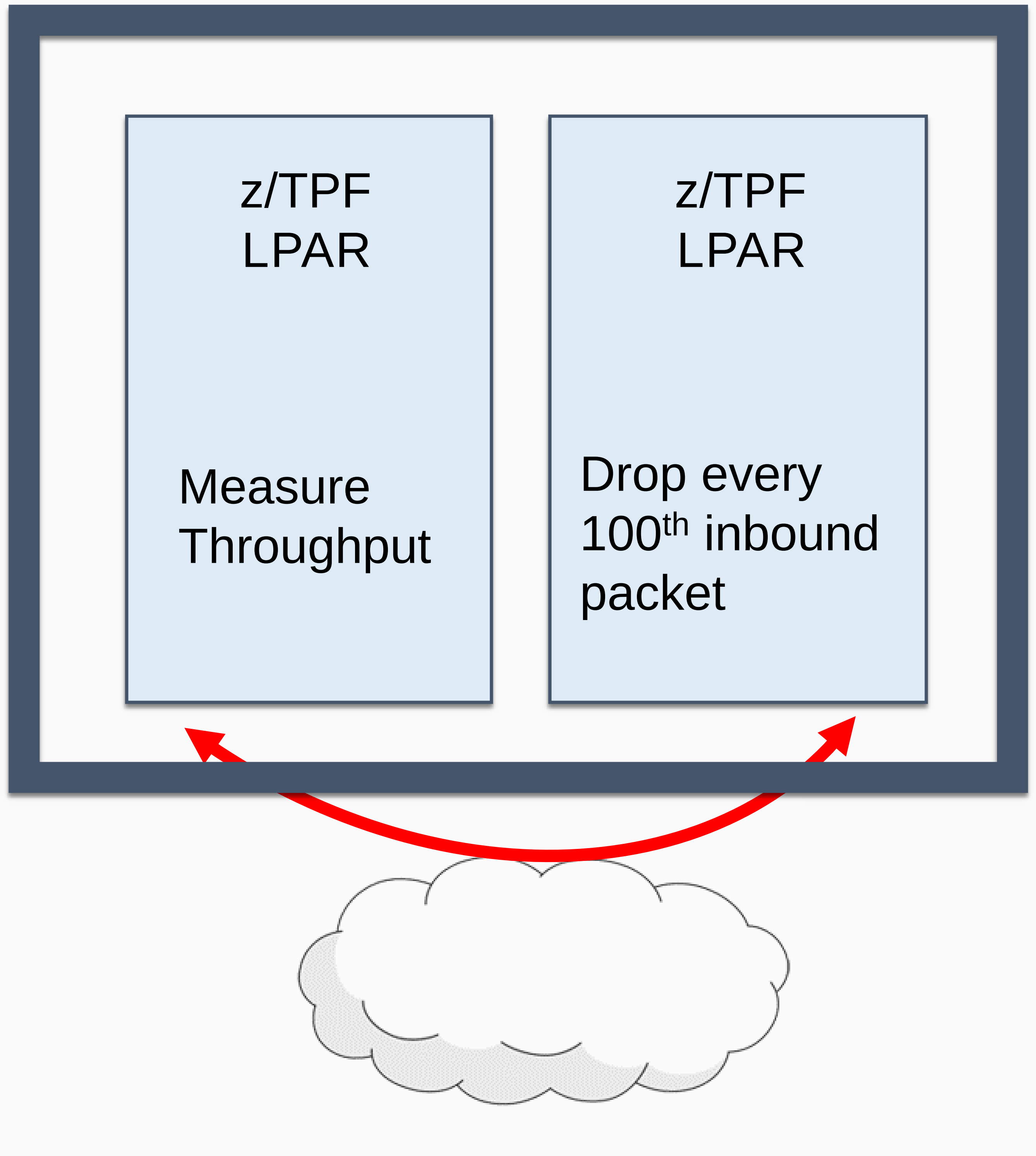
Sequence #4

TPF retransmits, as low as 20ms later

No subsequent packets sent, remote TCP endpoint continues to wait. Unaware that a packet was lost.

Retransmit timeouts will be calculated from **socket's round** trip time (RTT) and variance of it (RTTVAR).

Sub-Second Retransmission Performance Details



Ping-Pong 500 bytes messages back and forth

Num Sockets	Throughput WITHOUT Sub-Second Retransmit (msgs / sec)	Throughput WITH Sub-Second Retransmit (msgs / sec)	Increase In Throughput
1	49	1602	32x
10	489	12435	25x

Your Results May Vary

z/TPF Socket Lock Contention Enhancement

Significant reduction in z/TPF socket lock contention when sending “large” outbound TCP/IP messages - resulting in higher throughput and less MIPs consumed in a many-way tightly coupled environment with high utilization.

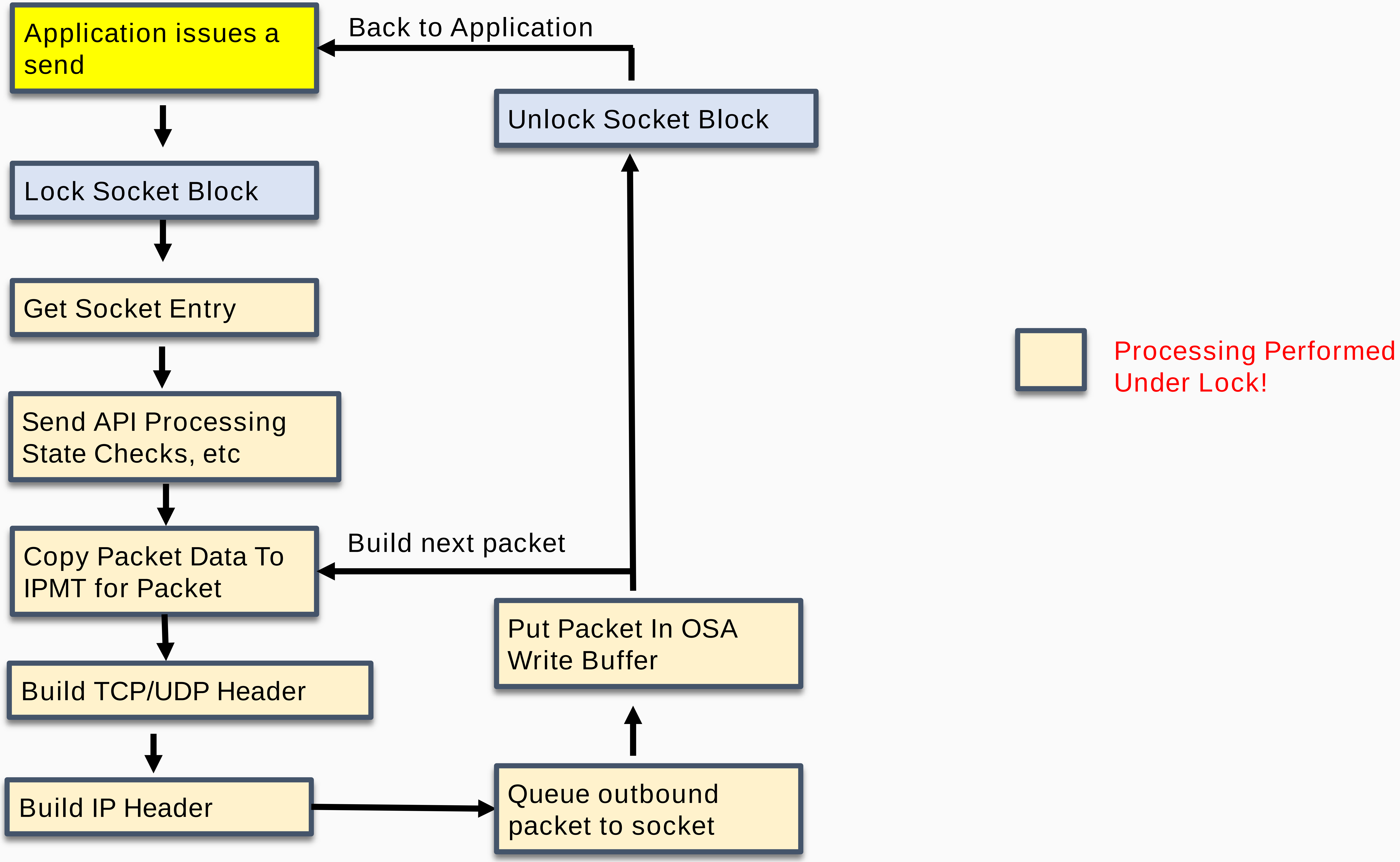
As-Is Scenario

Sending TCP/IP Application Messages

z/TPF LPAR
Application issues a send for 50K
Look up socket block
IP: 10.12.14.16
Port: 47000
Output Queue
Socket Send Buffer IPMT Blocks
Build TCP/UDP/IP headers
TCP/UDP Layer
IP Layer
IP Layer
- Build IP Header
Queue to socket send buffer
Place in OSA Write Buffer
OSA-Express Connection
OSA Write Buffers
Out to the network

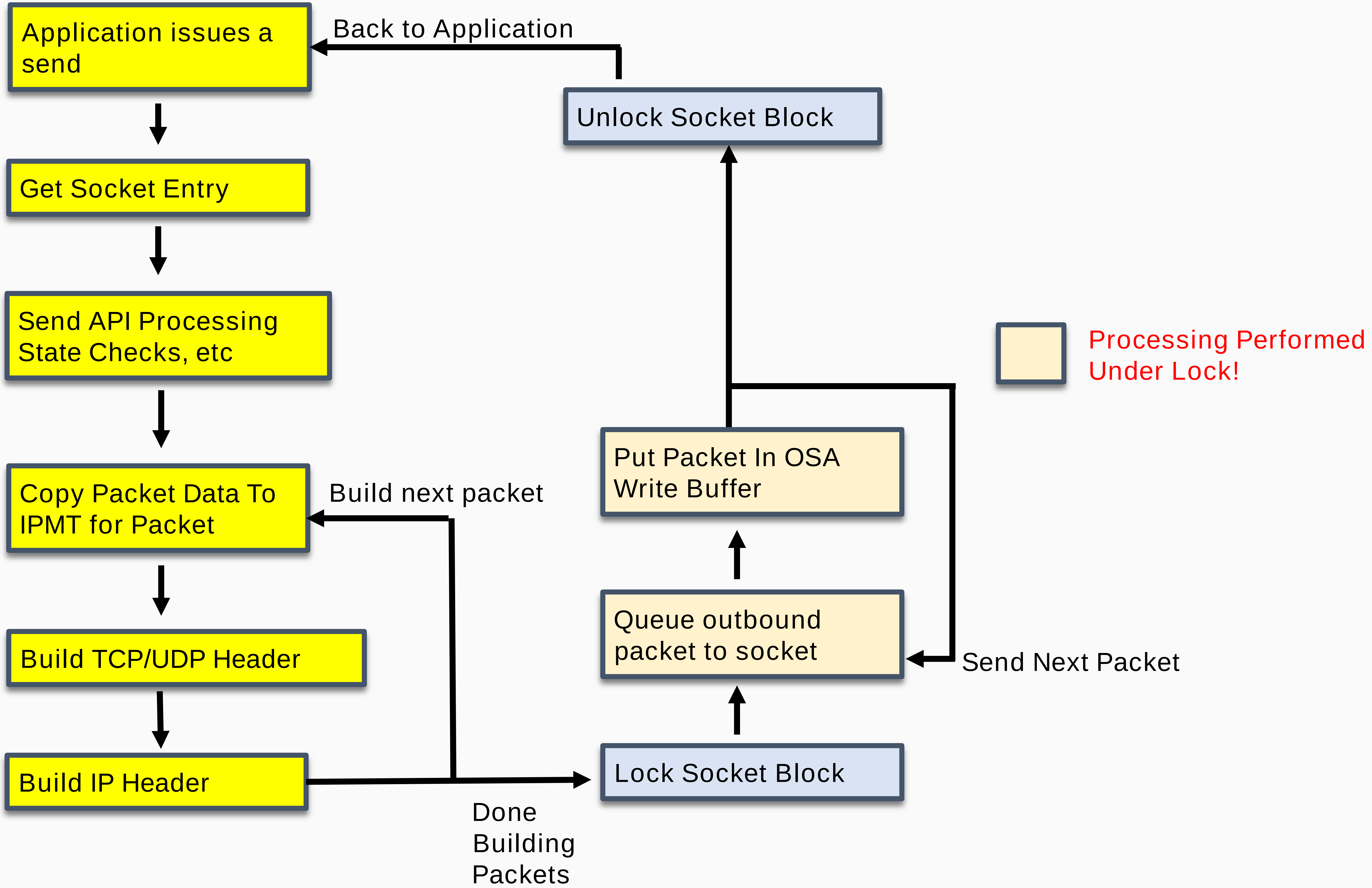
As-Is Scenario

TCP/IP Send Processing


Application issues a send
Back to Application
Unlock Socket Block
Lock Socket Block
Get Socket Entry
Send API Processing State Checks, etc
Copy Packet Data To IPMT for Packet
Build next packet
Put Packet In OSA Write Buffer
Build TCP/UDP Header
Build IP Header
Queue outbound packet to socket
Processing Performed Under Lock!

To-Be Scenario

TCP/IP Send Processing

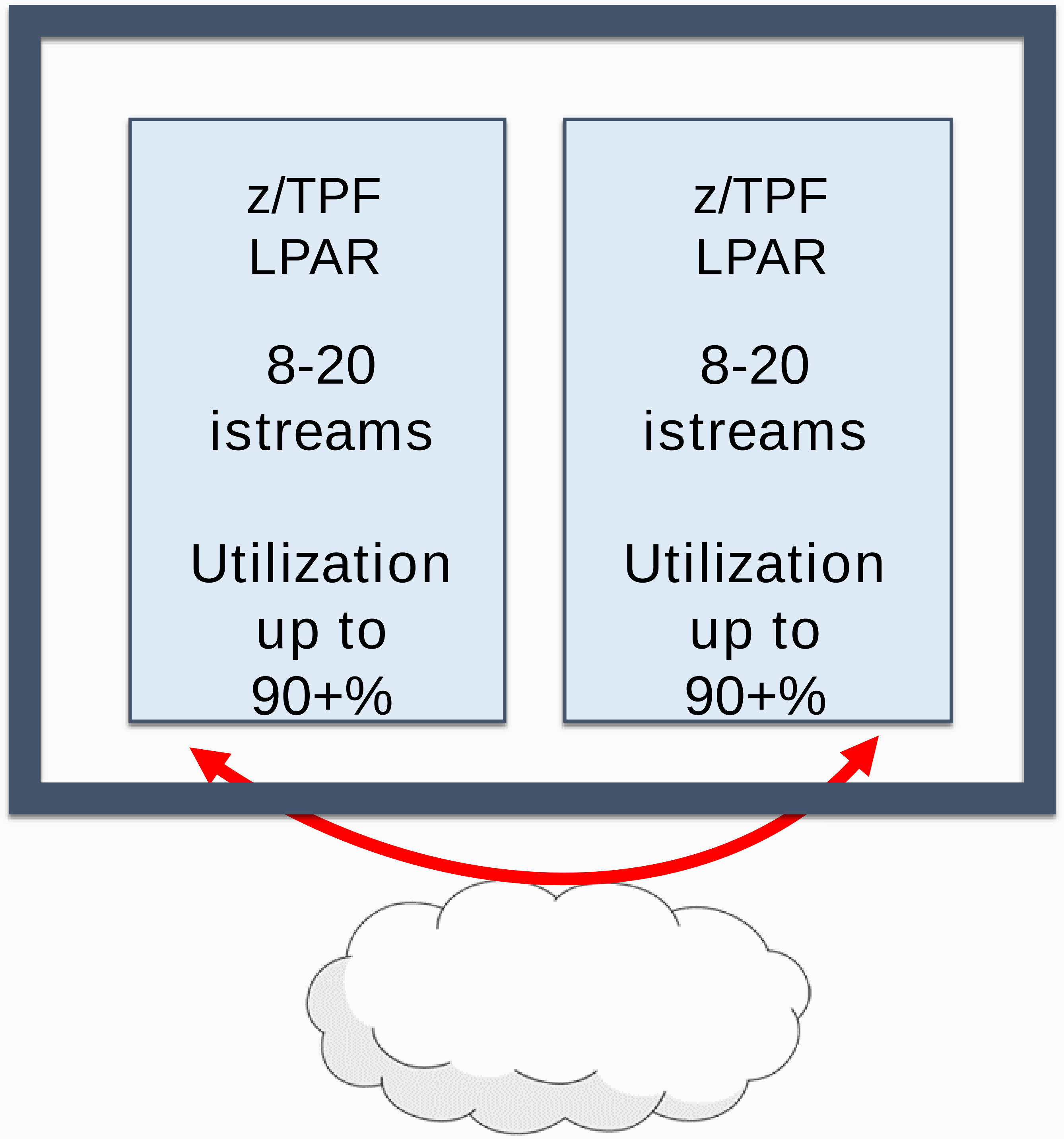


Application issues a send
Back to Application
Unlock Socket Block
Get Socket Entry
Send API Processing State Checks, etc
Processing Performed Under Lock!
Put Packet In OSA Write Buffer
Copy Packet Data To IPMT for Packet
Build next packet
Queue outbound packet to socket
Build TCP/UDP Header
Send Next Packet
Lock Socket Block
Build IP Header
Done Building Packets

Socket Lock Contention Enhancement Details

- Greatly reduced send processing time under the socket block lock
- Enhanced other TCP/IP APIs like read, AOR to reduce processing time under the socket block lock
    - Reduced the number of SVC calls
- Enhanced socket lock contention is automatically enabled when APAR is applied.
    - APARs PJ43697 (PUT 13), PJ44521 (PUT 14)

Socket Lock Contention Enhancement Performance Results



Ran mix of traffic
1000 – 800,000 bytes
Various buffer sizes
Various Read Sizes

- Up to 15% increase in normalized throughput
- Up to 40% decrease in time spent spinning on socket block lock

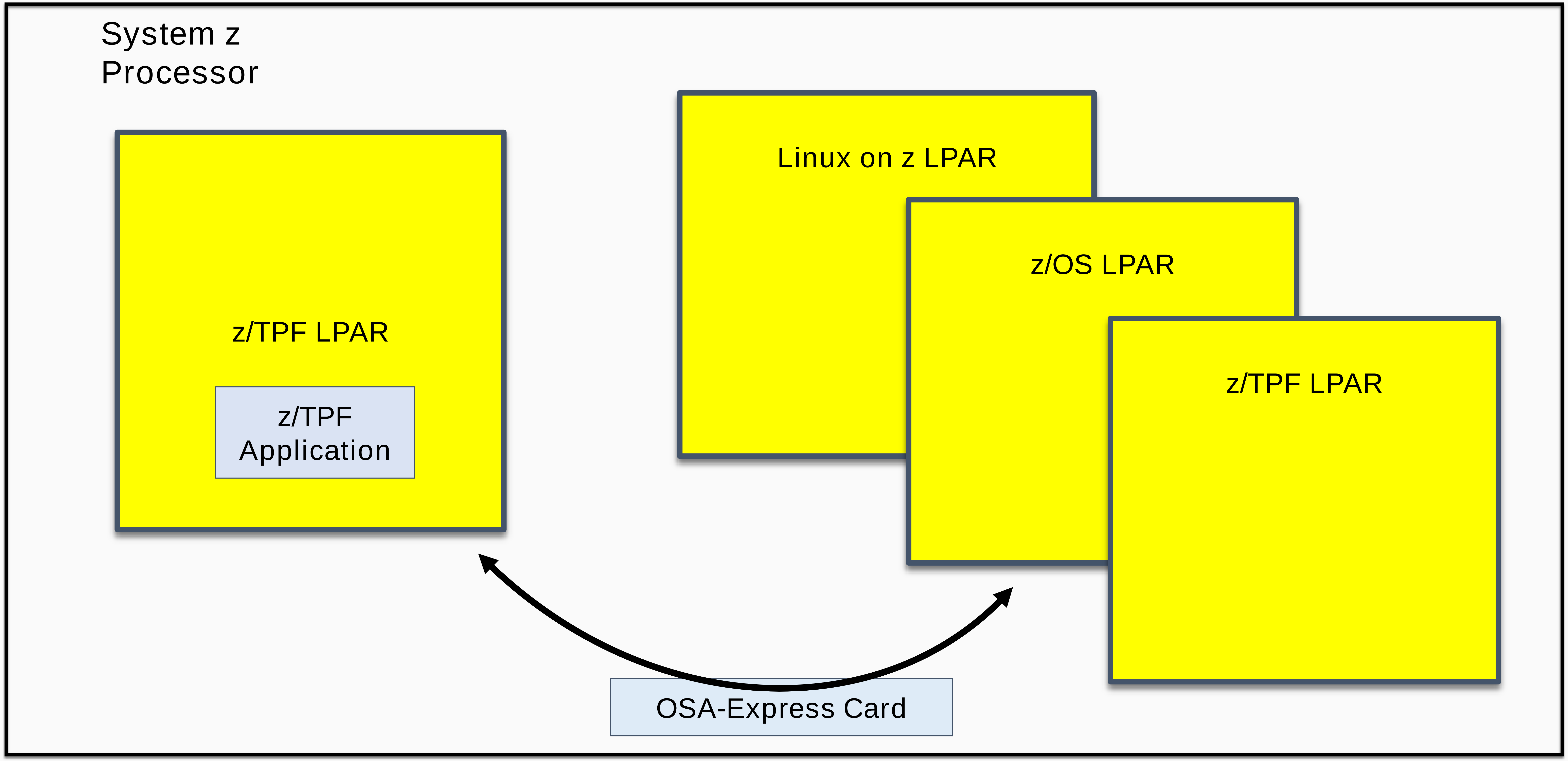
Your Results May Vary

z/TPF High Speed Connector

z/TPF applications can send messages to remote servers efficiently and without knowledge of the connections between z/TPF and the remote servers.

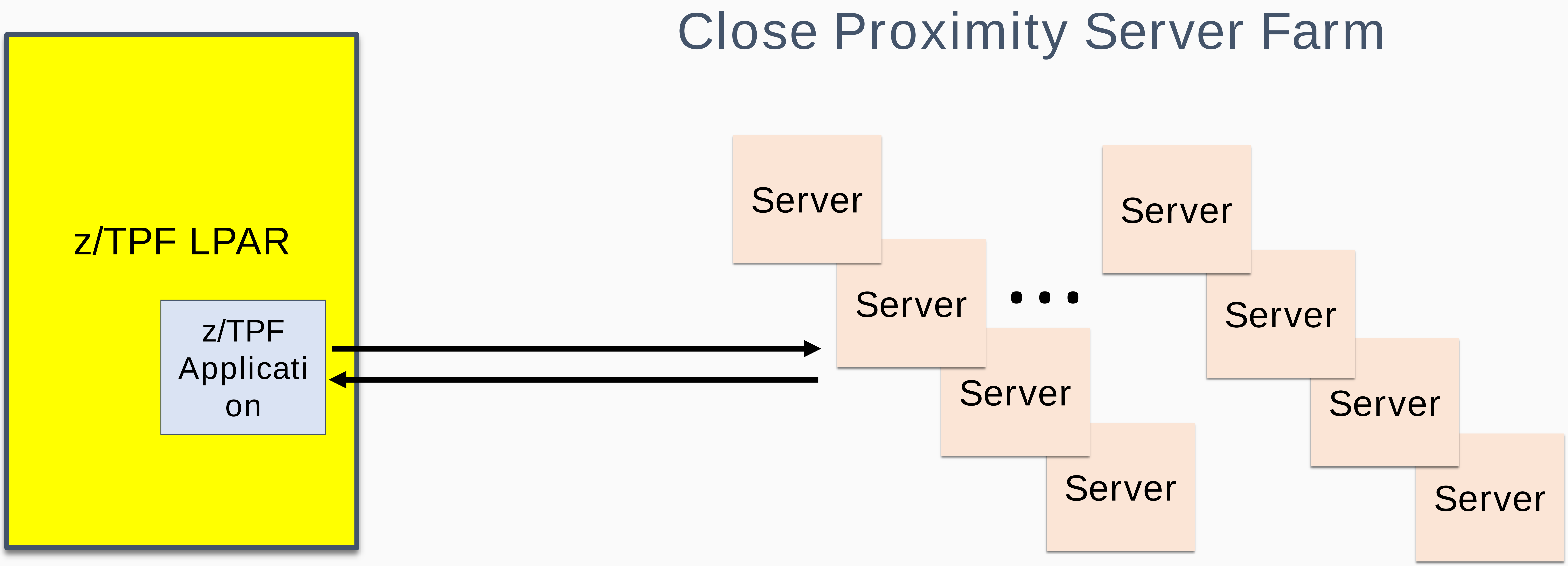
High Speed Connector

z/TPF Communicating with Other LPARs in CEC



High Speed Connector

z/TPF Communicating with a Server Farm



Communicating with Remote Systems

- Could use standard middleware, but many times it is too heavy compared to the request processing.
- z/TPF application would need to handle the following
  - Load balancing of servers
  - Primary and Fallback scenarios
  - Error Handling
  - Managing pools of persistent connections
  - Queueing when no servers are available
- Changing the topology or number of servers may require application updates.

High Speed Connector

- Through configuration, an administrator can define groups of servers
- From an application
  - Send a request to a “group”
- The High Speed Connector processing handles
  - Load balancing requests across the group of servers
  - Ability to define servers as primary and backup (only used when primary is not available)
  - Error Handling and automatic session establishment
  - Ability to display statistics and provide management of endpoints
    - Handle maintenance on any one server non-disruptively.
  - Queueing requests when no servers are available
  - Dynamic increase of connections to endpoint or number of endpoints is immediate and non-disruptive.

Remote Server Code

- You can write your own server logic on remote systems
  - Sample server code is available on z/TPF download page

OR

- You can use standard Open Source packages with minor modifications
  - For example, LogStash

Defining An Endpoint Group

```
<tns:EndpointGroup ... >
  <tns:Endpoint>
        <tns:endpointName>PRCRYP1</tns:endpointName>
        <tns:role>PRIMARY</tns:role>
    <tns:destination>remHost.ibm.com:15000</tns:destination>
        <tns:startSocket>25</tns:startSocket>
        <tns:maxSocket>100</tns:maxSocket>
  </tns:Endpoint>
  <tns:Endpoint>
     <tns:endpointName>BKCRYP1</tns:endpointName>
     <tns:role>BACKUP</tns:role>
     <tns:destination>9.57.13.155:15000</tns:destination>
     <tns:startSocket>25</tns:startSocket>
     <tns:maxSocket>100</tns:maxSocket>
  </tns:Endpoint>
  <tns:groupName> CRYPSVR1 </tns:groupName>
  <tns:qMaxDepth>400</tns:qMaxDepth>
  <tns:qThreshold>45</tns:qThreshold>
  <tns:syncTimeout>500</tns:syncTimeout>
  <tns:heartbeatInterval>300</tns:heartbeatInterval>
</tns:EndpointGroup>
```

TPF Endpoint Definitions

- Name
- Primary/Backup
- Host/Port
- Starting/Max Sockets

TPF Group Definitions

- Name
- Queue Depth
- Queue Threshold
- Sync Timeout
- Heartbeat Interval

*Endpoint descriptor is loaded using standard loaders

Invoking The Send Message API

```
#include <tpf/tpfapi.h>

hsc conn_parms;                                   ← Structure that contains the parameters for tpf_send_message
char endpoint_in[9], endpoint_out[9];
char* endpoint_group_name = “CRYPSVR1”;                   ← The group name defined in Endpoint Descriptor
connector_parms.version             = HSC_VERSION_1;      ← The version of the API to use.
connector_parms.endpointGroup = endpoint_group_name;      ← The name of the endpoint group to send a message to.
connector_parms.request             = request_buffer;     ← Buffer where the request message is stored.
connector_parms.timeout             = 2000;               ← Time in ms until the API times out
connector_parms.resp_len           = 256;                 ← Length of the response.  0 if no response expected
connector_parms.response          = malloc(256);          ← Allocating storage for the response buffer.
connector_parms.endpoint_in      = endpoint_in;           ← Specific endpoint to send a message to
connector_parms.endpoint_out    = endpoint_out;           ← Endpoint that a message was sent to.
int rc;

if ((rc = tpf_send_message(&conn_parms))!=TPF_SEND_MESSAGE_OK)
    printf(“error”);
else
    printf(“success”);
```

Supported Commands

- ZCONN START GROUP-ept_grp [ENDPOINT-ept]
- ZCONN STOP GROUP-ept_grp [ENDPOINT-ept]
- ZCONN QUIESCE GROUP-ept_grp ENDPOINT-ept
- ZCONN DISPLAY (ALL|GROUP-ept_grp)
- ZCONN STATS GROUP-ept_grp
- ZCONN MAXSTATS GROUP-ept_grp
- ZCONN CLEARSTATS (ALL|GROUP-ept_grp)

Displaying Group Information

```
User:   ZCONN DISPLAY GROUP-CRYPSVR1

System: CONN0020I 11.13.59 ENDPOINT GROUP DISPLAY

        CURRENT QUEUE SIZE    -           0
        QUEUE HIGH WATER MARK -           0
        MAX QUEUE ALLOWED     -         400

        SERVER
        ENDPOINT  ROLE STATUS SESSIONS MAXSESS INUSE APIS/SEC API TIME TIMEOUTS ERRORS
        --------- ---- ------ -------- ------- ----- -------- -------- -------- ------
        PRCRYP1   PRIM ACTIVE       32     100    27      882    1.133        0      0
        BKCRYP1   BACK ACTIVE       25     100     0        0    0.000        0      0
        --------- ---- ------ -------- ------- ----- -------- -------- -------- ------
        TOTALS                      57     200    27      882    1.133        0      0

        END OF DISPLAY
```

Displaying Group Statistics

```
User:   ZCONN STATS GROUP-CRYPSVR1

System: CONN0022I 13.15.17 ENDPOINT GROUP STATS

        SERVER                  APIS/
        ENDPOINT        APIS      SEC API TIME TIMEOUTS ERRORS
        --------- ---------- -------- -------- -------- ------
        PRCRYP1      4567890      882    1.133        0      0
        BKCRYP1            0        0        0        0      0
        --------- ---------- -------- -------- -------- ------
        TOTALS       4567890      882    1.133        0      0

        END OF DISPLAY
```

Overloading Remote Endpoints

```
User:   ZCONN DISPLAY GROUP-CRYPSVR1

System: CONN0020I 11.13.59 ENDPOINT GROUP DISPLAY

        CURRENT QUEUE SIZE    -        20
        QUEUE HIGH WATER MARK -        29
        MAX QUEUE ALLOWED     -        400

        SERVER
        ENDPOINT  ROLE STATUS SESSIONS MAXSESS INUSE APIS/SEC API TIME TIMEOUTS ERRORS
        --------- ---- ------ -------- ------- ----- -------- -------- -------- ------
        PRCRYP1   PRIM ACTIVE      100     100   100      877    1.139        0      0
        BKCRYP1   BACK ACTIVE      100     100   100      876    1.141        0      0
        --------- ---- ------ -------- ------- ----- -------- -------- -------- ------
        TOTALS                     200     200   200     1753    1.140        0      0

        END OF DISPLAY
```

Increasing Group Capacity

- Update the group’s endpoint group descriptor
- Load the file through the version control file system
- New endpoints in the group or increasing maximum sockets will automatically take effect
- No application changes required.

Adding Endpoints to An Endpoint Group

```
 <tns:EndpointGroup ... >
  <tns:Endpoint>
          <tns:endpointName>PRCRYP1</tns:endpointName>
          <tns:role>PRIMARY</tns:role>
    <tns:destination>remHost.ibm.com:15000</tns:destination>
          <tns:startSocket>25</tns:startSocket>
          <tns:maxSocket>100</tns:maxSocket>
  </tns:Endpoint>
  <tns:Endpoint>
          <tns:endpointName>PRCRYP2</tns:endpointName>
          <tns:role>PRIMARY</tns:role>
    <tns:destination>remHost2.ibm.com:15000</tns:destination>
          <tns:startSocket>25</tns:startSocket>
          <tns:maxSocket>150</tns:maxSocket>
  </tns:Endpoint>
  <tns:Endpoint>
     <tns:endpointName>BKCRYP1</tns:endpointName>
     <tns:role>BACKUP</tns:role>
     <tns:destination>9.57.13.155:15000</tns:destination>
     <tns:startSocket>25</tns:startSocket>
     <tns:maxSocket>100</tns:maxSocket>
  </tns:Endpoint>
  <tns:groupName> CRYPSVR1 </tns:groupName>
  <tns:qMaxDepth>400</tns:qMaxDepth>
  <tns:qThreshold>45</tns:qThreshold>
  <tns:syncTimeout>500</tns:syncTimeout>
  <tns:heartbeatInterval>300</tns:heartbeatInterval>
</tns:EndpointGroup>
```

New Primary Endpoint

- Name
- Primary/Backup
- Host/Port
- Starting/Max Sockets

Statistics After Adding Endpoints

```
User:   ZCONN DISPLAY GROUP-CRYPSVR1

System: CONN0020I 11.13.59 ENDPOINT GROUP DISPLAY

        CURRENT QUEUE SIZE    -          0
        QUEUE HIGH WATER MARK -        200
        MAX QUEUE ALLOWED     -        400

        SERVER
        ENDPOINT  ROLE STATUS SESSIONS MAXSESS INUSE APIS/SEC API TIME TIMEOUTS ERRORS
        --------- ---- ------ -------- ------- ----- -------- -------- -------- ------
        PRCRYP1   PRIM ACTIVE      100     100    88      887    1.121        0      0
        PRCRYP2   PRIM ACTIVE      123     150    97      888    1.139        0      0
        BKCRYP1   BACK ACTIVE      100     100     0        0    1.141        0      0
        --------- ---- ------ -------- ------- ----- -------- -------- -------- ------
        TOTALS                     323     350   185     1775    1.134        0      0

        END OF DISPLAY
```

High Speed Connector Summary

- Complexity of z/TPF applications communicating with remote servers is greatly reduced.
- Dynamic increase of capacity as workload increases
- Allows for monitoring and management of endpoint groups and the associated connections
- APAR PJ43892 (PUT 13)
- z/TPF High Speed Connector code is TE-Eligible!
- High Speed Connector Starter Kit Available
    - Contains sample endpoint group descriptor files, remote server application code, z/TPF driver code to send high speed connector messages.
    - http://www-01.ibm.com/support/docview.wss?uid=swg24043067

Disclaimer

Any reference to future plans are for planning purposes only. IBM reserves the right to change those plans at its discretion. Any reliance on such a disclosure is solely at your own risk. IBM makes no commitment to provide additional information in the future.

z/TPF Greater Than 64K Read Support

Reduce z/TPF application and middleware complexity and improve performance of reading large TCP messages.

As-Is Scenario

Reading Large TCP Messages

- Current maximum length of a TCP read is 64K
  - Cannot set low water mark above 64K
- A given 800K message today requires at a minimum of 13 reads

Pseudo Code to Read 800K Message

```
Set low water mark to 65535
Timeout = 3
Set socket receive timeout to Timeout
While 800K not received
    If length remaining < 65335
        set low water mark to remaining.
    Get time before read
    Read data from socket
    Get time after read
    Calculate time for read
    Decrement timeout
    Set socket receive timeout
    Update length remaining and buffer pointer
```

Reading Large TCP Messages

- New maximum length of TCP read is 1M
  - Can set a low water mark of up to 1M
- A single read API can be issued to read an 800K message

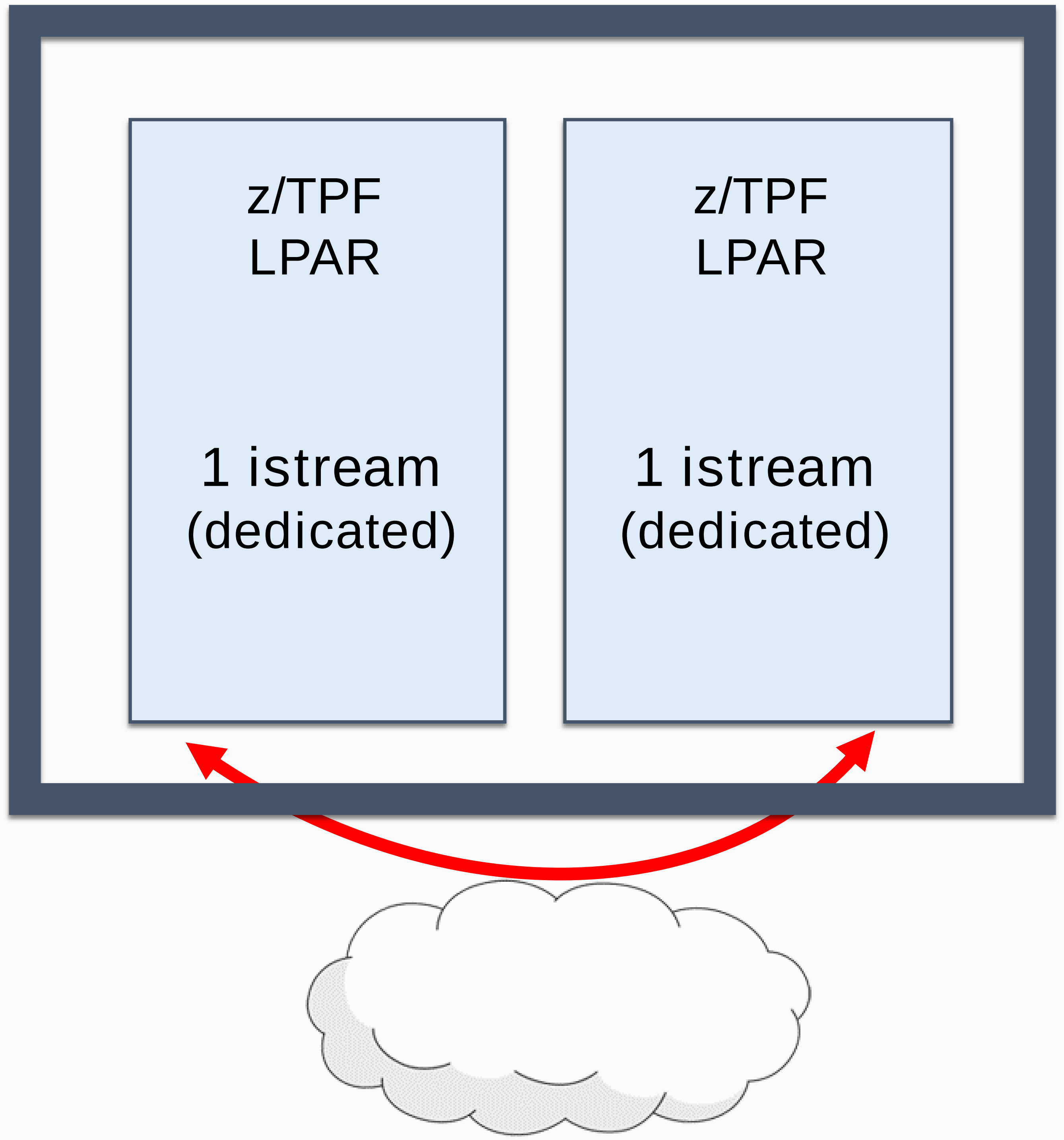
Pseudo Code to Read 800K Message

- Set low water mark to 800K
- Set socket receive timeout
- Read data from socket

Greater Than 64K TCP Read Details

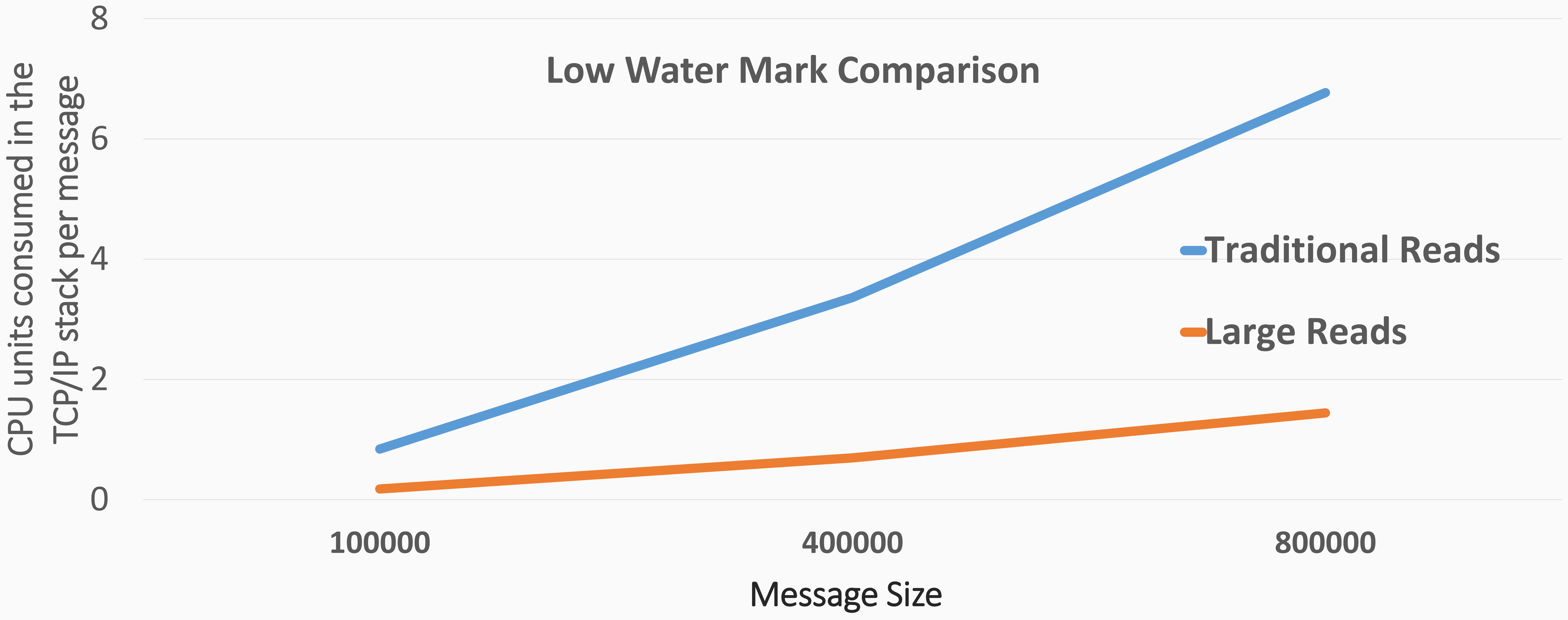
- The maximum length on ALL read-type APIs has been increased to 1M.
  - Read, recv, recvfrom, AOR,
- The z/TPF TCP message APIs have been updated to allow message sizes of up to 1M
  - tpf_read_TCP_message and activate_on_receipt_of_TCP_message
- The low water mark socket option on the setsockopt API has been expanded to 1M
- z/TPF Websphere MQ has been updated to use the new support!
- PUT 14 APAR, PJ44531, provides this support

Greater Than 64K TCP Read Performance Testing



- Application reads message and echoes message back
- 100K, 400K and 800K messages measured
- Cost per message to read and echo reply

Greater Than 64K TCP Read Performance Results



*Your Results May Vary

Summary

- PJ43958 (PUT 13)
  - The z/TPF system can recover from outbound packets dropped in the network in milliseconds as opposed to seconds improving the overall throughput on the system.
- PJ43697 (PUT 13) & PJ44521 (PUT 14)
  - Significant reduction in z/TPF socket lock contention when sending “large” outbound TCP/IP messages - resulting in higher throughput and less MIPs consumed in a many-way tightly coupled environment with high utilization.
- PJ43892 (PUT 13)
  - z/TPF applications can send messages to remote servers efficiently and without knowledge of the connections between z/TPF and the remote servers.
- PJ44531 (PUT 14)
  - Reduce z/TPF application and middleware complexity and improve performance of reading large TCP messages.

THANK YOU

Questions or comments?

Jamie Farmer

IBM **z/TPF**
April 3rd, 2017

Notes

Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.

All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.

This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

This presentation and the claims outlined in it were reviewed for compliance with US law. Adaptations of these claims for use in other geographies must be reviewed by the local country counsel for compliance with local laws.