Shared SSL & InetD Enhancements for Starting TLS Sessions

Communication Subcommittee

Jamie Farmer

2025 TPF Users Group Conference
May 05-07, Austin, TX

IBM Z

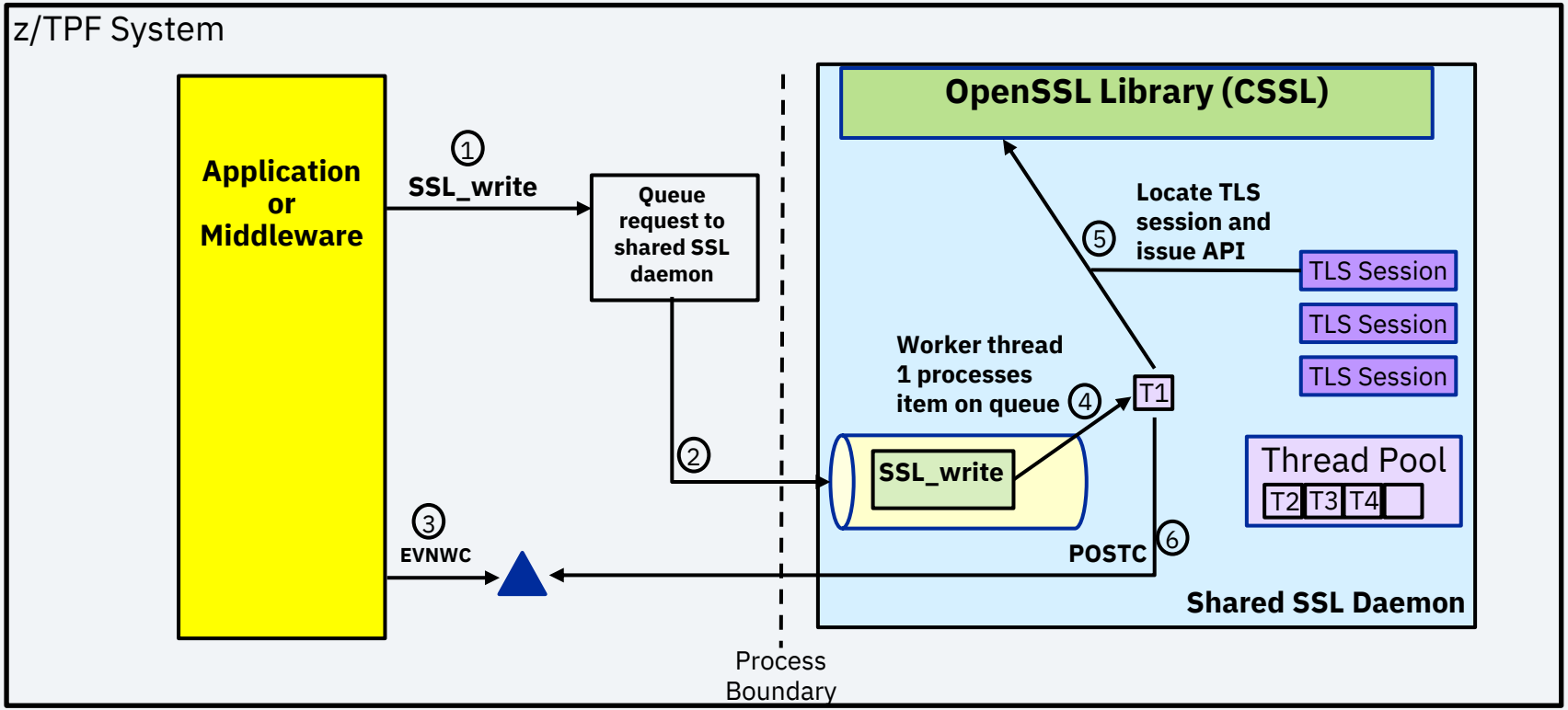
IBM

Shared SSL Enhancements for Starting TLS Sessions

What is Shared SSL?

- The shared SSL package is based on a set of long-running daemon processes each with a set of worker threads.
- The shared SSL daemon processes own the TLS sessions on behalf of applications and middleware
  - The worker threads issue TLS APIs on behalf of the applications and middleware
- Shared SSL provides …
  - Sharing TLS sessions across multiple application ECBs
  - Asynchronous I/O, like activate_on_receipt (SSL_aor)

Shared SSL Architecture



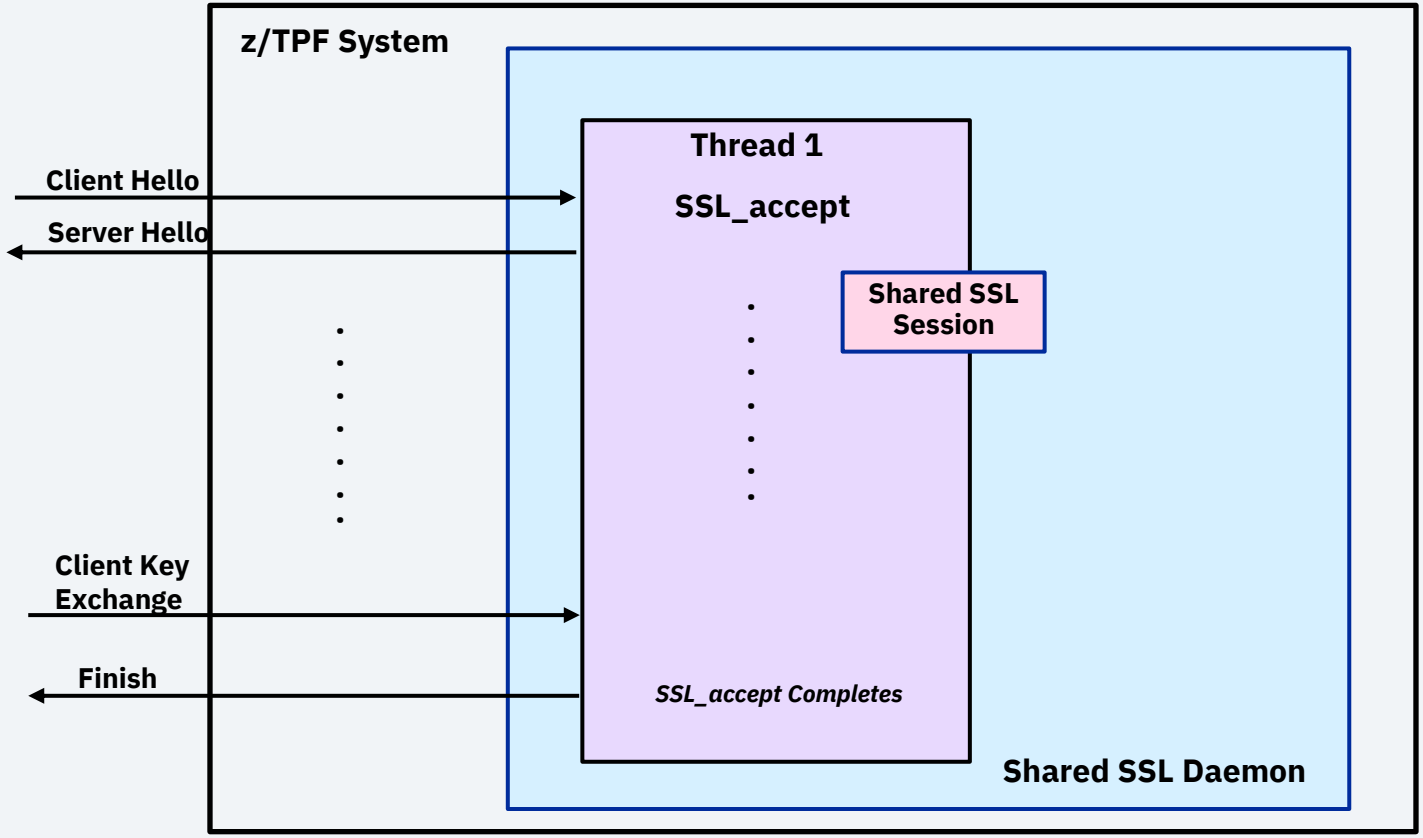
Shared SSL Architecture

1. The z/TPF application ECB issues an SSL_write API for a shared SSL session
2. The SSL_write API is queued to the shared SSL daemon
3. An EVNWC API is issued by application ECB to wait for the reply
4. A worker thread finds this item on queue and begins to process it
5. The SSL session is located and the API is issued on behalf of the application
6. A POSTC API is issued to wake up the application ECB with the response

Background

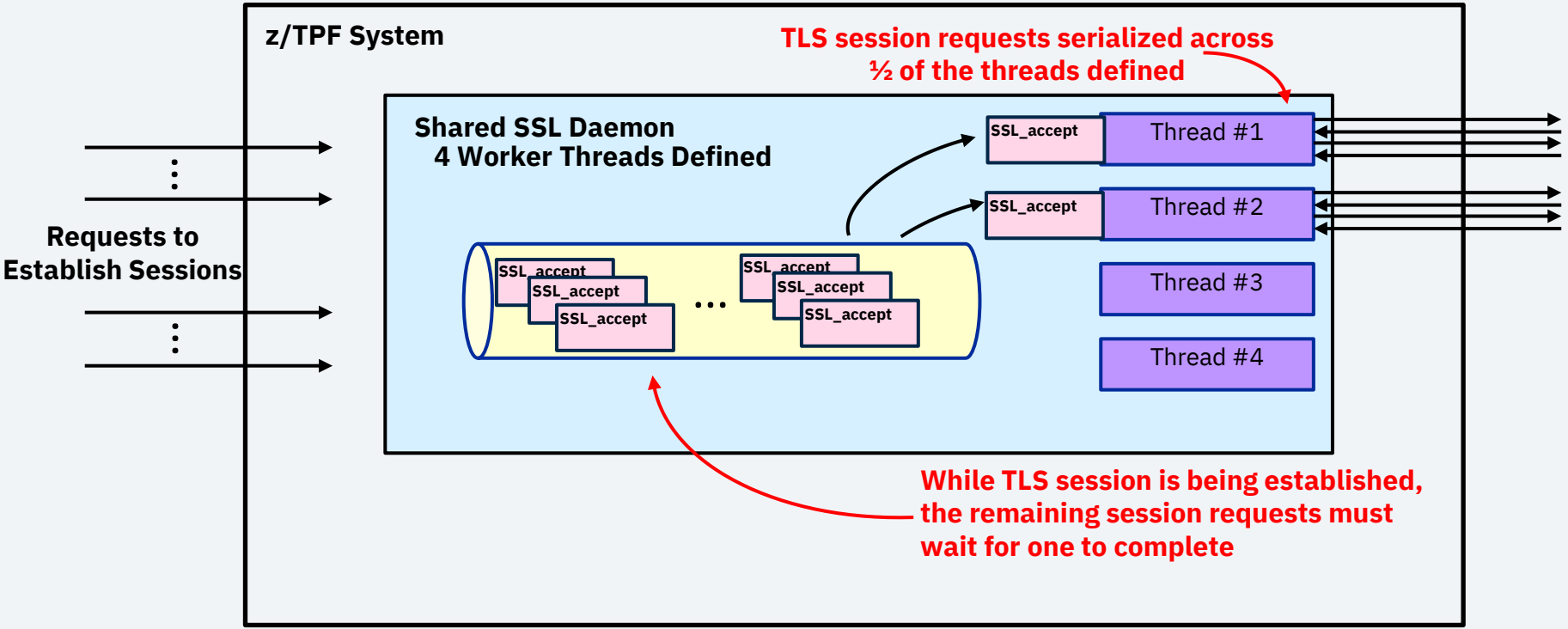
- Most TLS APIs are processed by the shared SSL daemon worker threads in a non-blocking fashion
  - SSL_connect and SSL_accept are processed as blocked APIs
    - Shared SSL worker thread cannot be used for any other processing until SSL_connect or SSL_accept completes
    - SSL_connect and SSL_accept APIs require multiple network flows
  - As many as half of the shared SSL worker threads will be used for starting TLS sessions
    - Ensures there are threads always available for transactional APIs like SSL_write and SSL_read

As-Is: Establish Shared SSL TLS Sessions

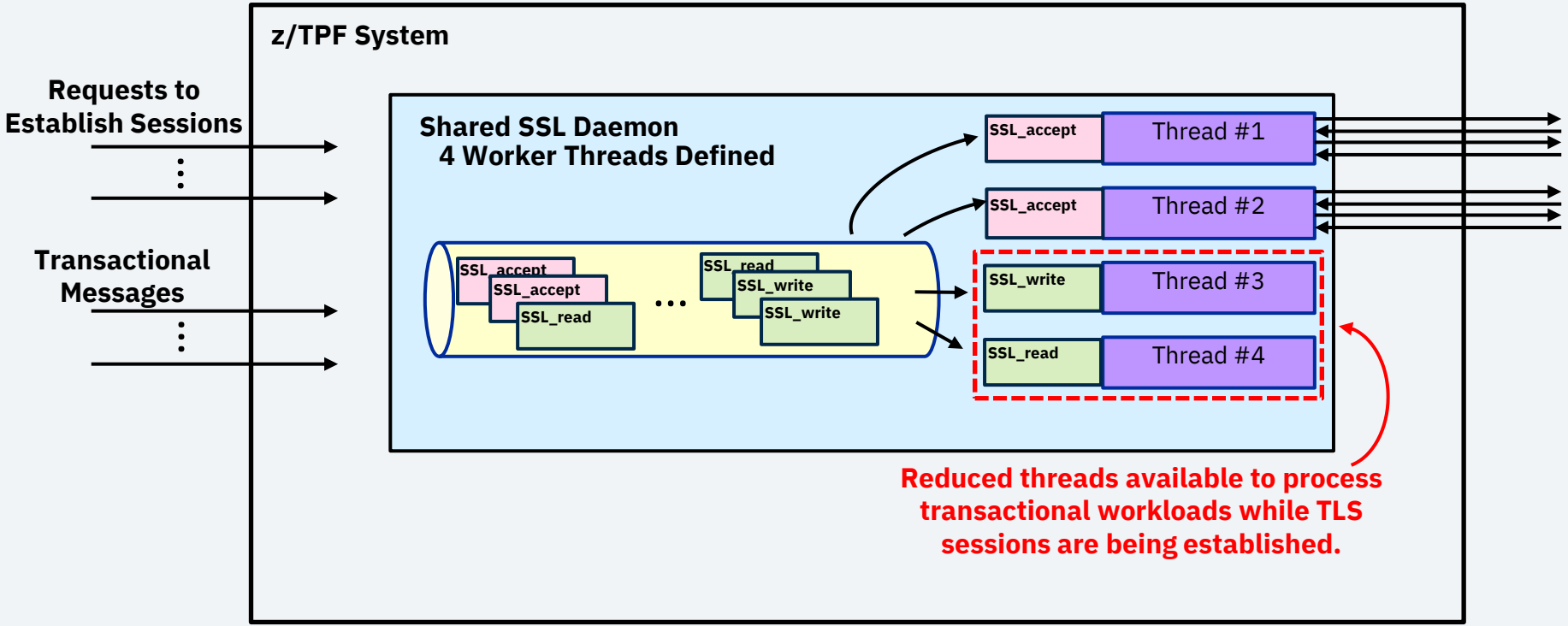


When a shared SSL thread processes a TLS session start request, the thread cannot do any other work until the session is established.

Pain Point – Shared SSL Session Flood



Pain Point – Session Startups Might Affect Transactional Workloads

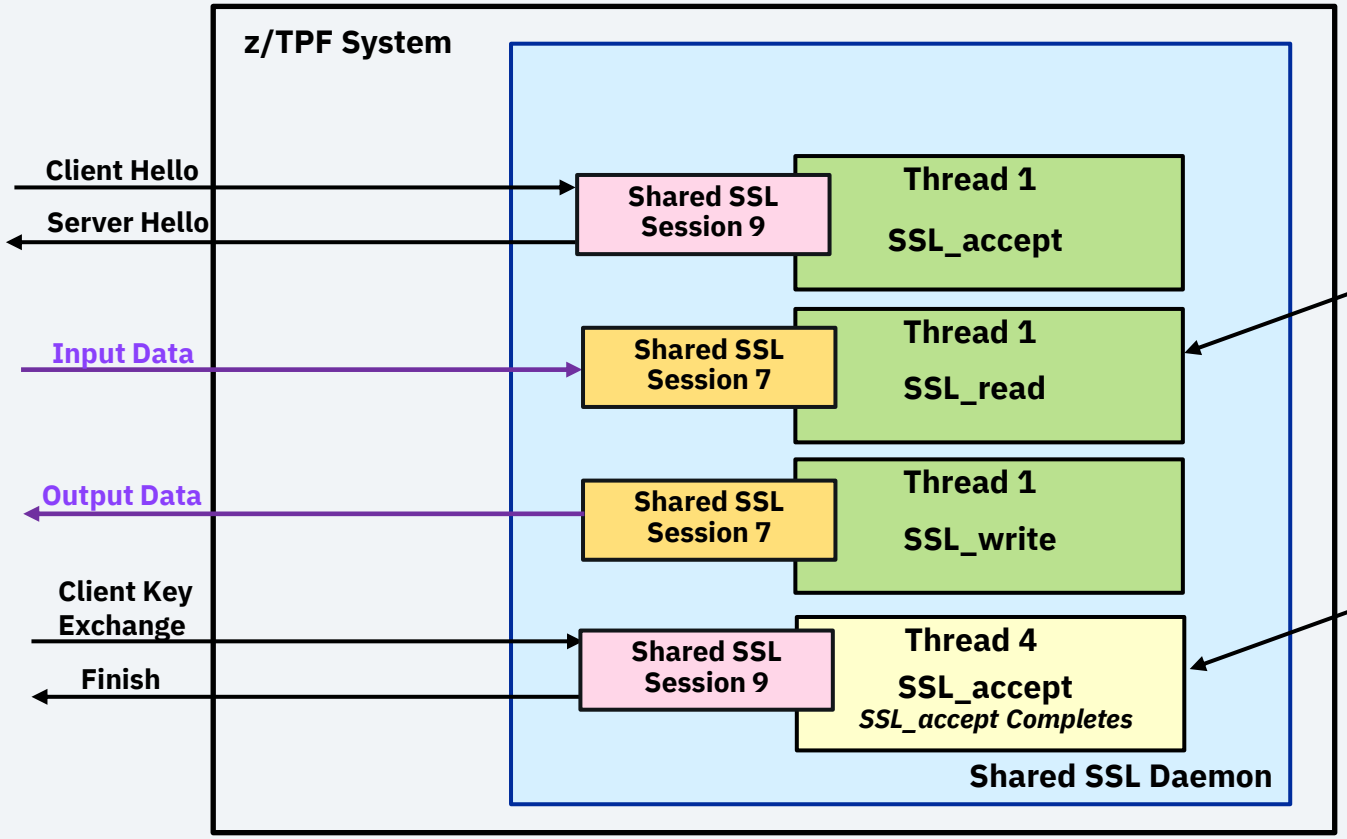


Unblocking Threads When Sessions Start

- The shared SSL daemons were updated to make worker threads available while a TLS handshake is waiting for network flows.
  - The shared SSL daemon can start more TLS sessions concurrently.
  - More threads are available for transactional workloads when TLS sessions are being established.
  - No application changes are required.

- Delivered with APAR PJ48040, January 2025

To-Be: Establish Shared SSL TLS Sessions

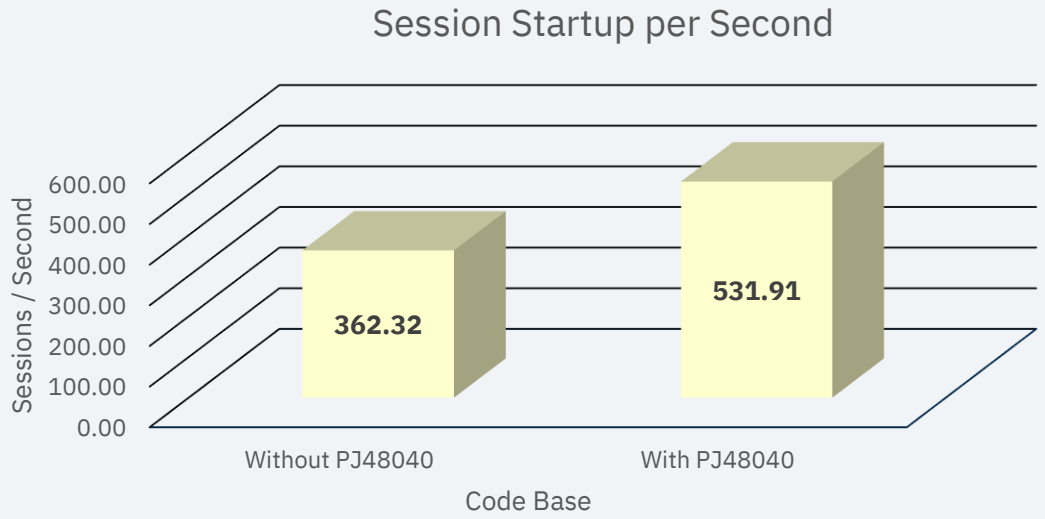


Thread 1 is free to perform other work while session 9 is being established.

Any thread can resume and finish establishing the TLS session for session 9

Improved Throughput When TLS Sessions Start

- Flooded a z/TPF system with tens of thousands of TLS session requests



46% Increase In Throughput

Environment
- **Single shared SSL daemon with 4 worker threads**
- **4-dedicated I-streams**
- **700 series IBM z16**

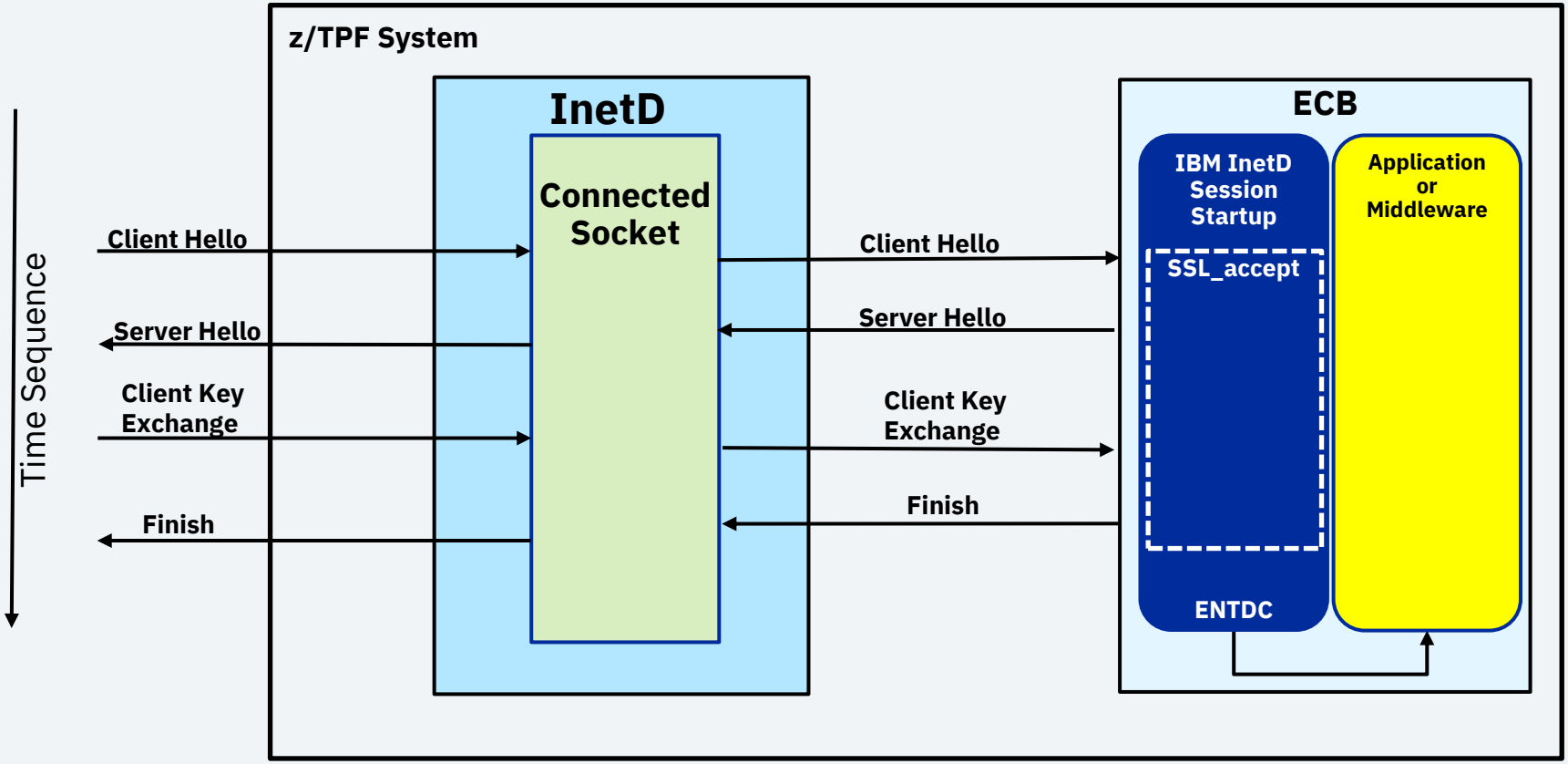
Your results might vary

Internet Daemon Enhancements for Starting TLS Sessions

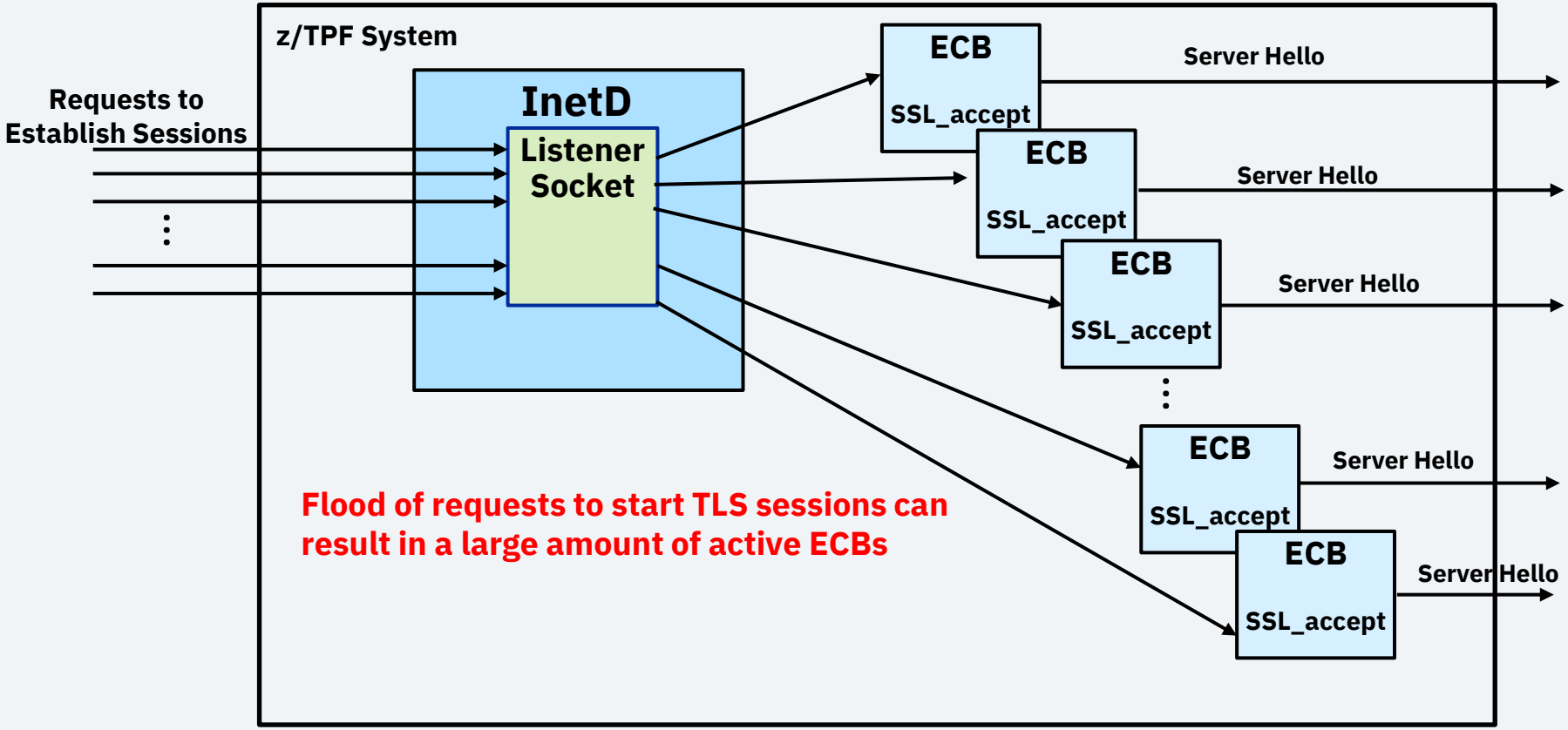
Background

- When a new TCP connection is received for a server that uses the SSL model of the z/TPF Internet Daemon, a new ECB is created through the activate_on_accept() API.
  1. Issues an activate_on_accept API to read the next connection request
  2. The created ECB subsequently issues an SSL_accept API.
     a. The created ECB remains active until the SSL_accept API is complete.
  3. Enters the InetD SSL model server application with the established TLS session
- A flood of TLS session requests can cause a spike of active ECBs that are all waiting for TLS session start requests (SSL_accept API) to complete

As-Is: Establish InetD TLS Sessions



Pain Point – TLS Session Flood

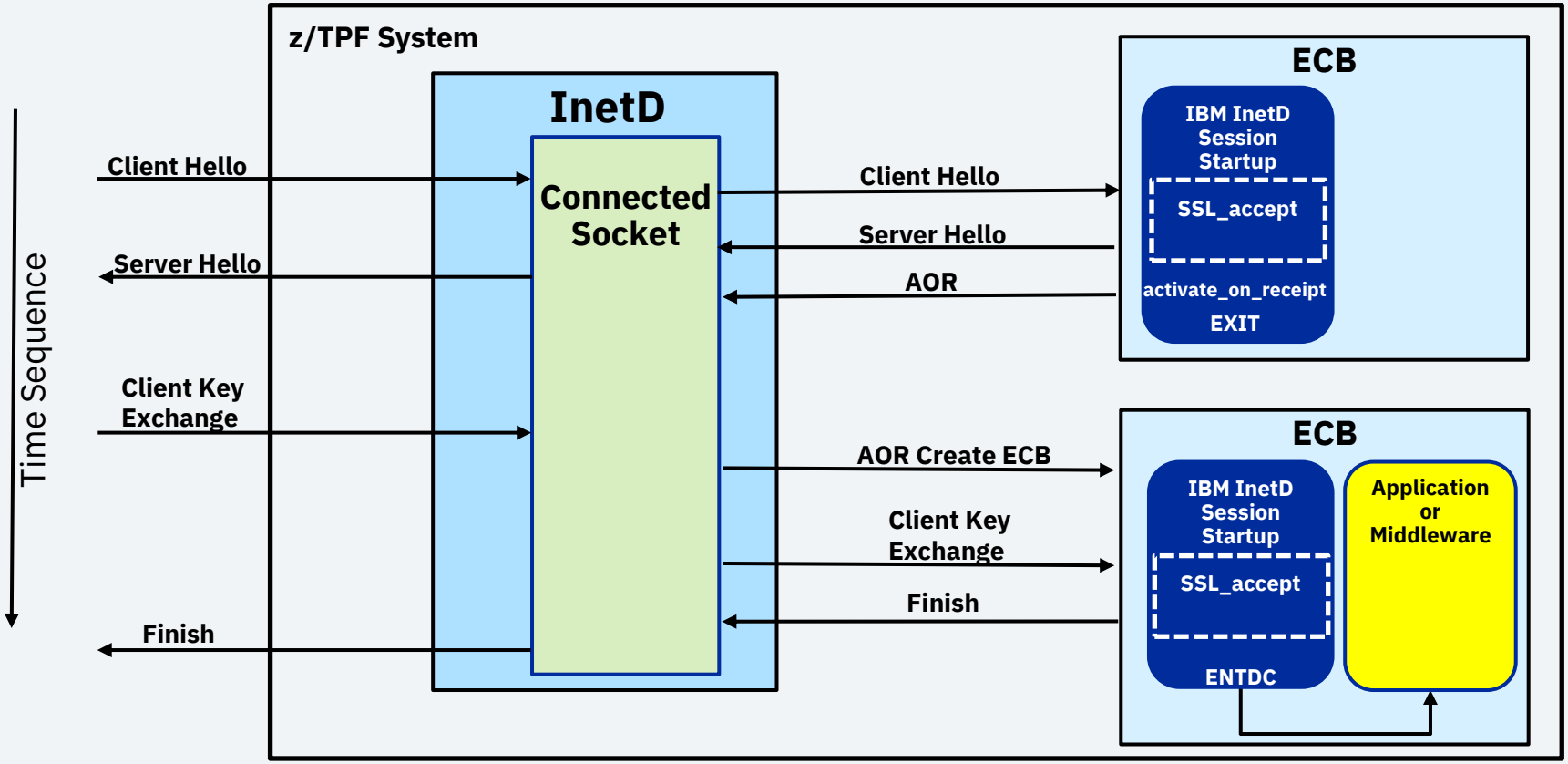


Establish Asynchronous TLS Sessions for SSL model InetD

- ECBs created from the Internet Daemon SSL model will now operate in a non-blocking manner when establishing a TLS session.
- The InetD ECB will exit when it is waiting for data from the remote client.
- Results in a much lower usage of ECBs and for a much shorter time period.
- No application changes are required.

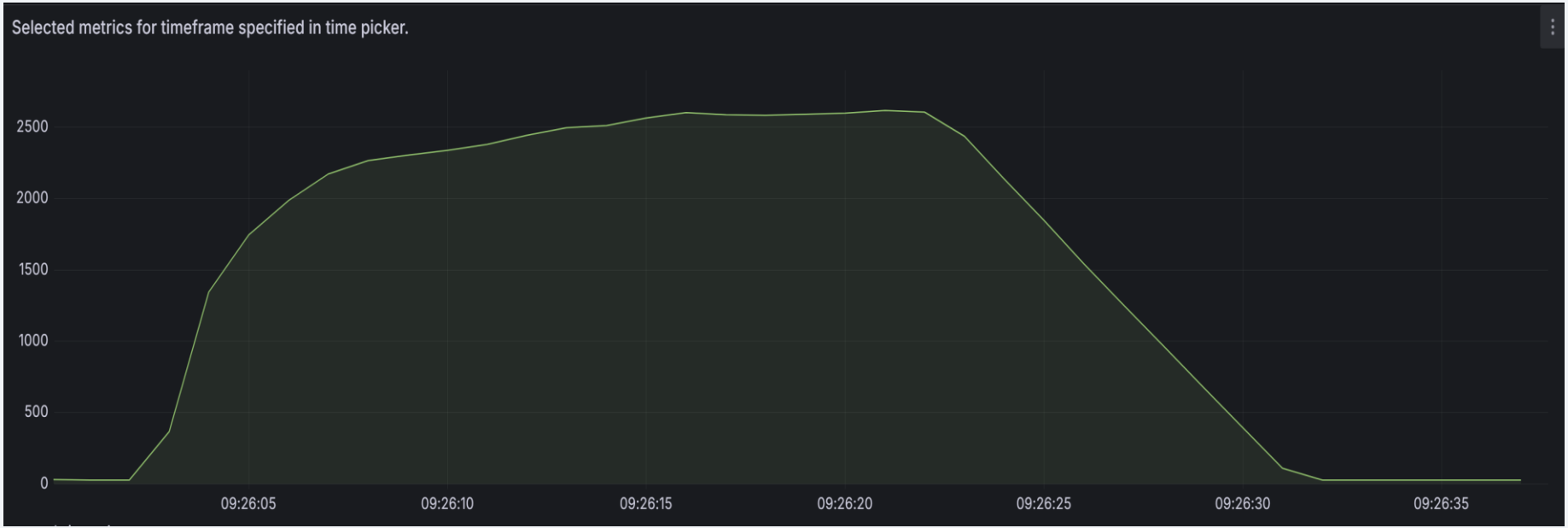
- Delivered with APAR PJ48040, January 2025

To Be: Establish Sessions Across Multiple ECBs



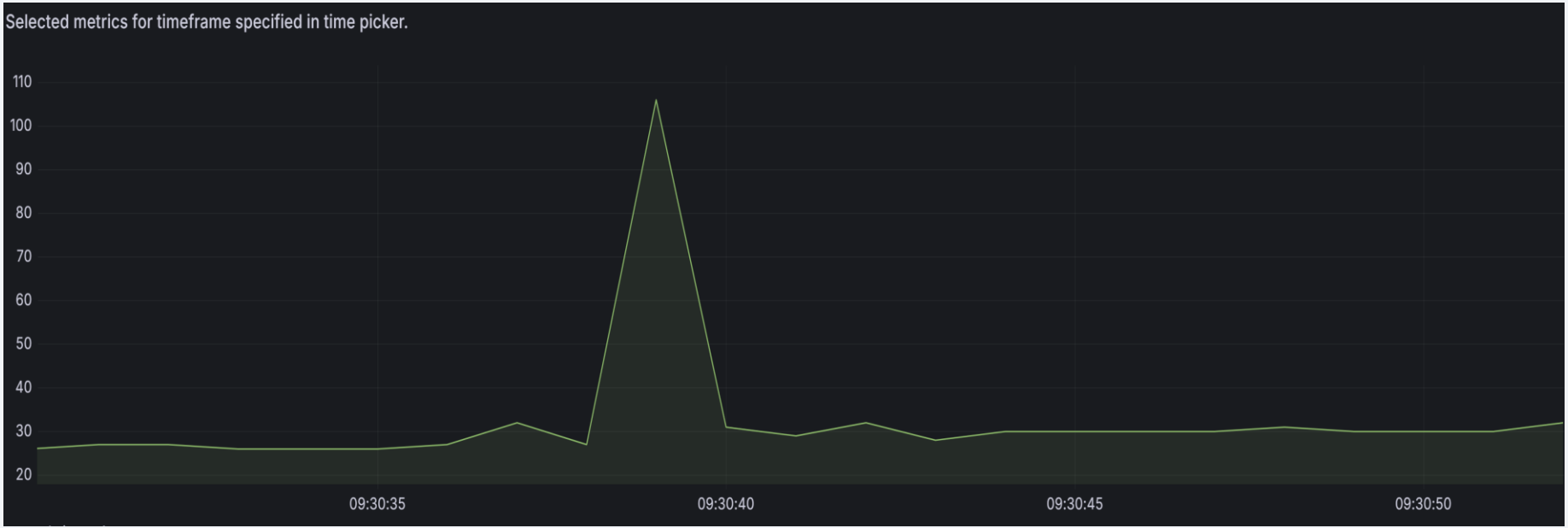
ECB Usage When Many TLS Sessions Starting – Prior to PJ48040

- Flooded the z/TPF system with 8000 TLS session requests
- RTMC showed **~2500** ECBs in use for over a 20 second period.



Reducing ECB Usage When Many TLS Sessions Starting – With PJ48040

- Flooded the z/TPF system with 8000 TLS session requests
- RTMC showed only **~100** ECBs in use and only for a 2 second period.



Summary

- PJ48040 was delivered in January 2025.
- Enhanced TLS session startup through shared SSL and the Internet Daemon SSL model.
- Improved throughput when TLS sessions are started and reduction of memory resources when system is flooded with TLS session requests.
  - Benefits achieved without any application changes.

Thank you