z/TPF Communications and Security Enhancements

Communications Subcommittee

Jamie Farmer

2024 TPF Users Group Conference
May 05-08, New Orleans, LA

IBM Z

IBM

Agenda

- Upgraded OpenSSL Library - OpenSSL 3.0.7
- Support for Latest TLS Protocol - TLS 1.3
- Enhanced HTTP Client Proxy
- z/TPF Secure File Transfer Support

- HTTP 2.0 (Future)

z/TPF | 2024 TPF Users Group | May 05-08, New Orleans, LA | ©2024 IBM Corporation

OpenSSL 3.0.7

PJ46990 & PJ47068

(July 2023)

IBM Z

IBM

Problem Statement

- In September 2023, the OpenSSL community dropped support for OpenSSL 1.1.1.
  - New vulnerability fixes will not be applied to this release.

- The latest long-term release is OpenSSL 3.0

Introducing OpenSSL 3.0.7

- The z/TPF system now has an upgraded version of OpenSSL, 3.0.7
  - Required to stay on supported version and remain current with security fixes.
  - Same functionality supported in OpenSSL 1.1.1 is supported in OpenSSL 3.0.7

- Delivered July 2023 with APARs PJ46990 & PJ47068 (libCurl)

OpenSSL Error Codes

- Error code format has changed between versions.
- If you want to display the string of a 3.0.7 error code (openssl errstr xxxx) on your Linux system, the OpenSSL code on Linux must be version 3.0.7 or higher.
  - OpenSSL 1.1.1

```
openssl errstr 1408f10b

error:1408F10B:SSL routines:SSL3_GET_RECORD:wrong version number
```

  - OpenSSL 3.0.7

```
openssl errstr 0A00010B

error: 0A00010B:SSL routines:SSL3_GET_RECORD:wrong version number
```

Deprecated Functions

- Some of the external OpenSSL APIs are deprecated - No guarantee that they will exist in future releases.
  - *Replace the SSL_CTX_use_RSAPrivateKey_file* and *SSL_use_RSAPrivateKey_file APIs* with the *SSL_CTX_use_PrivateKey_file* and *SSL_use_PrivateKey_file APIs*, respectively
  - *Replace SSL_get_peer_certificate* function with the *SSL_get1_peer_certificate function*
    - *Remapping of function exists but requires a recompile*
  - Replace the *ERR_get_error_** and *ERR_peek_error_** functions with other corresponding functions
  - Some functionality like MD5 hashes have been deprecated

- See OpenSSL documentation for full list of deprecated functions.

Installation Considerations

- Fixing APAR PJ47117 must be applied to address build issues
  - Delivered August 2023
  - Pertaining to obsoletion of segments

- If you are using LDAP, shared object CLLB must be recompiled

TLS 1.3

PJ47183 (April 2024)

What Is TLS 1.3?

- z/TPF supports TLS versions 1.0, 1.1, 1.2, and now 1.3
- TLS 1.3 is the latest Transport Layer Security (TLS) cryptographic protocol version
    - Streamlined session startup
    - Enforcement of perfect forward secrecy ciphers
        - The ECDHE key exchange only
    - Only supports strong ciphers and message digest algorithms.
    - More secure because the TLS handshake flows are now encrypted.

TLS 1.3 Industry Supported Ciphers

- *TLS_AES_128_GCM_SHA256*
- *TLS_AES_256_GCM_SHA384*
- *TLS_CHACHA20_POLY1305_SHA256*
- *TLS_AES_128_CCM_SHA256*
- *TLS_AES_128_CCM_8_SHA256*

TLS_AES_128_GCM_SHA256

Cipher to encrypt user data
Message Digest for Integrity Checking

Naming convention no longer identifies key exchange or signature algorithms

TLS 1.3 simplifies the TLS configuration because only 5 ciphers are supported, compared to the dozens available with TLS 1.2.

TLS 1.3 Supported Ciphers on z/TPF

- ***TLS_AES_128_GCM_SHA256***
- ***TLS_AES_256_GCM_SHA384***
- *TLS_CHACHA20_POLY1305_SHA256*
- *TLS_AES_128_CCM_SHA256*
- *TLS_AES_128_CCM_8_SHA256*

Supported z/TPF TLS 1.3 Ciphers!

ChaCha not supported on z/TPF

CCM Hardware Acceleration does not exist

- **The z/TPF system supports the two most commonly used ciphers in the industry.**
- **The GCM ciphers are also superior to CCM ciphers in terms of security**

TLS 1.3 Cipher Details

- New ciphers are equivalent to TLS 1.2 cipher – just a different name

TLS 1.3	TLS 1.2
TLS_AES_256_GCM_SHA384	ECDHE-RSA-AES256-GCM-SHA384
TLS_AES_128_GCM_SHA256	ECDHE-RSA-AES128-GCM-SHA256

- z/TPF TLS 1.3 default cipher list
  TLS_AES_256_GCM_SHA384 : TLS_AES_128_GCM_SHA256

Using TLS 1.3 for z/TPF Middleware Packages (except IBM MQ)

- Upon installation, if you are using z/TPF middleware you will automatically begin using TLS 1.3 when the following is true…
  - The remote system you are talking to is configured for TLS 1.3
  - The remote system has one of the supported TLS 1.3 ciphers in its cipher list
  - The MAXVERSION parameter in your TLS configuration file is not set to a version earlier than TLS 1.3.

- If the previous conditions are true, there is **nothing you need to do to start using TLS 1.3**!
  - When TLS 1.3 ciphers are not in the z/TPF middleware configuration file the default TLS 1.3 ciphers will be applied.
  - When MAXVERSION is not defined, the maximum version applied is the highest version supported on the platform, which is now TLS 1.3!

Using TLS 1.3 for z/TPF IBM MQ

- Because there are no cipher names in common for TLS 1.2 and TLS 1.3, we are making it easier to migrate to TLS 1.3 for receiver channels.
  - The z/TPF system will automatically allow existing receiver channels to be established with TLS 1.3
  - This way, you can use the same receiver channel specific file as remote systems migrate their channels from TLS 1.2 to TLS 1.3.

- The behavior of TLS for IBM MQ sender channels is unchanged.
  - You can migrate individual sender channels to TLS 1.3 by updating the sender channel specific TLS configuration file to use a TLS 1.3 cipher.

Application Interface Changes

- When using OpenSSL directly in your applications (not z/TPF middleware), the following API changes will require application updates to use TLS 1.3.

TLS 1.2 (or lower)	TLS 1.3	Notes
SSL_CTX_set_cipher_list SSL_set_cipher_list	SSL_CTX_set_ciphersuites SSL_set_ciphersuites	• TLS 1.3 only accepts colon separated lists • tpf_SSL_getConfig() updated to return colon separated • Can pass same cipher string to both sets of APIs and it ignores ciphers not applicable to that version of TLS.
SSL_renegotiate	SSL_key_update	• The z/TPF middleware packages issuing SSL_renegotiate have been updated

Using Compliance Tooling to Understand TLS 1.3 Usage

- When the APAR is applied and applications converted to use TLS 1.3, you can use the existing set of compliance tooling to know which applications are still using TLS 1.2….

```
ZDCOM USAGE VERSION-TLS1.2

DCOM0007I 10.35.04 NETWORK COMPLIANCE USAGE DISPLAY FOR VERSION TLS1.2
PORT   MODEL  VERSION USED
----- ------- --------------
443    SERVER       N
990    SERVER       N
5000   CLIENT       N
7700   SERVER       Y
```

```
ZDCOM USAGE VERSION-TLS1.2 PORT-7700 SERVER

DCOM0008I 12.06.31 REMOTE IP ADDRESSES THAT USE TLS1.2 FOR THE SERVER ON PORT 7700
REMOTE IP            LAST CONNECTION
---------------  --------------------
123.45.6.78        2024-05-05 10:46:47
12.34.56.78        2024-05-05 09:49:40
```

Performance

- TLS 1.3 session establishment uses slightly more CPU resources compared to TLS 1.2
  - Expected as the TLS 1.3 session flows are encrypted now.
  - Recommendation for TLS has always been to use long-running sessions

- Reduced TLS 1.3 session startup flows will reduce time to establish sessions
  - Greater benefit the further away remote systems are.

- TLS 1.3 data transfer has equivalent performance of TLS 1.2 for the equivalent cipher algorithms.
  - This is the performance critical code for TLS

Enhanced HTTP Client Proxy

PJ47044 (Aug 2023)

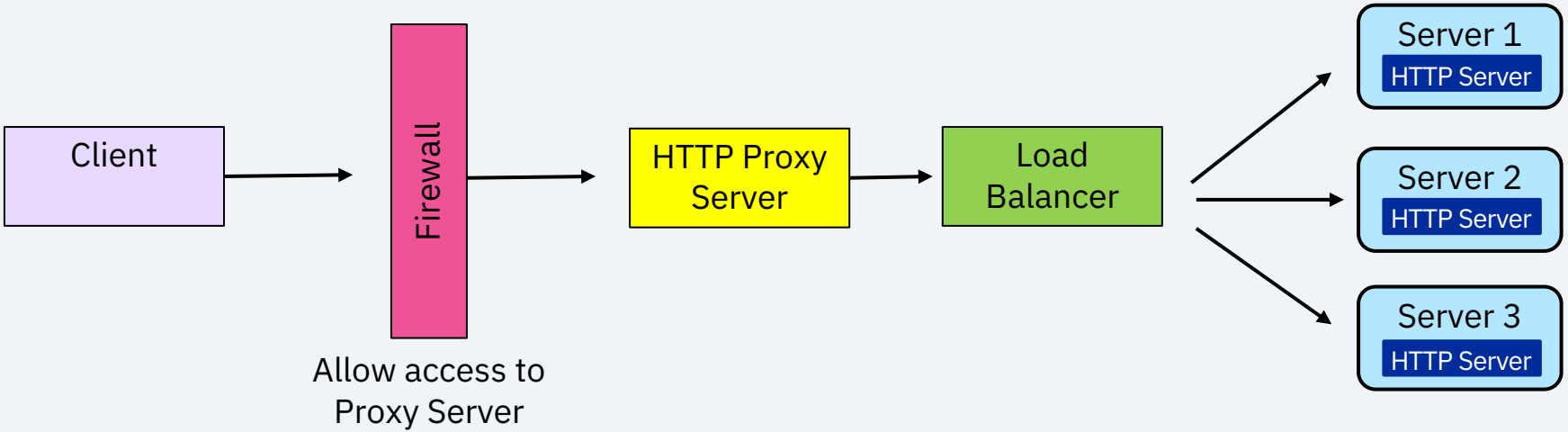
Access Through a Proxy for Enhanced HTTP Client

- Many HTTP servers are accessed through a proxy for security reasons
- Support for HTTP proxy existed in the older HTTP client based on libCurl
  - Note: The older HTTP Client based on libCurl is no longer supported
- This support is adding the ability to send enhanced HTTP client requests through a proxy server.

Advantages of an HTTP Proxy

- Main advantage is for protection of HTTP servers
  - No direct connections into the HTTP Server
  - Can examine web traffic to block malicious activity or to filter and validate requests to prevent garbage requests
- Other Advantages
  - Can act as a tunnel to provide devices access to restricted networks.
  - Can cache results data to reduce network bandwidth and load on the servers.
  - Load balancing across multiple HTTP servers.

HTTP Proxy in Action – Access Through Firewall



Specifying HTTP Proxy – Nonpersistent Sessions

- For nonpersistent sessions
  - Specified on the tpf_httpSendRequest or tpf_httpSendAsyncRequest
  - Pointer to a proxy IP/Host in the ConnectParms structure

```
char *proxy = "proxy.example.com:82";
   …
connectParms.httpProxyServer = proxy;
  …
rc = tpf_httpSendRequest(host, &requestParms, &response, &connectParms, 0);
```

  - In this case, session is established to proxy.example.com on port 82 and that proxy server forwards requests to the actual host specified on API

Specifying HTTP Proxy – Persistent Sessions

- Persistent sessions are established using z/TPF high-speed connector
- A new element has been added to the high-speed connector endpoint descriptor file to specify a proxy server
  - Specified at a group level – one proxy for the group of endpoints (hosts) the group is connecting to.

```
<tns:httpProxyServer>proxyserver.example.com:8080</tns:httpProxyServer>
```

  - In this case all sessions in the group are established to proxyserver.example.com on port 8080 and that proxy server forwards requests to the actual host specified on the group's endpoint definition

Specifying HTTP Proxy – REST Consumer

- You can update your REST consumer service descriptor to add a new httpProxyServer element.
  - This is not needed if the REST consumer is targeting a persistent high-speed connector group.
- The httpProxyServer element in the REST consumer service descriptor will be passed on the corresponding tpf_httpSendRequest or tpf_httpSendAsyncRequest on the connectParms structure

Updates to ZSRVC Command

- The ZSRVC command has been updated to include the proxy server and whether it will be used or not.
  - For example, if the proxy server is specified in the service descriptor, but the service descriptor is using a high-speed connector group (persistent sessions), the specified proxy server is ignored

```
System: SRVC0006I 13.14.01 REST SERVICE PROPERTIES AND ARTIFACTS DISPLAY
              SRVCNAME-tpf1SrvcBDDef VERSION-2.0.0
                    POST /tpf1Services/v2/serviceBDDef
                    HOST-http://host.example.com
                    PROXY-exampleProxy.com:8000
                    TIMEOUT-10000 PROVIDERTYPE-Program PROVIDER-QHH9
                    UNORDERED-TRUE DFDLFORMAT-NONE EXCLUDE-NONE
                    MAXREQUESTS-0 MAXREQUESTSERROR-503 MAXREQUESTSWARNINGINTERVAL-0
```

Support Statement Reminder – Curl based HTTP Client

- As of June 2023: The ‘older’ HTTP client that is based on the libCurl library is no longer supported
  - The *tpf_perform* and *tpf_perform1* APIs should be transitioned to use the enhanced HTTP client
    - *tpf_httpSendRequest* and *tpf_httpSendAsyncRequest*

Secure File Transfer

PJ46830 (July 2023)

Secure File Transfer Background

- A secure FTP client exists on z/TPF to transfer files securely to and from the z/TPF system when z/TPF is acting as the client.
- Customers need the ability to securely transfer files with z/TPF acting as a server.
    - For example, the ability to securely transfer loadsets into the z/TPF system by using loadtpf.

Java Based Secure File Transfer Solution

- z/TPF secure file transfer is implemented as a stand-alone SSH server
  - Uses the SSHD subproject of Apache Mina (https://mina.apache.org/sshd-project)
- SSH server is configured as a JAM
  - JAM support provides message logging, JVM monitoring and more.
- Connect to the SSH server on z/TPF by using any client that supports SFTP
  - This APAR adds SFTP server support only to z/TPF (not SFTP client support)
- Provides authentication of users by exchanging keys and authorization through z/TPF administration.
- Functionality is managed with new ZSSHD commands
  - Including the management if users and their keys

z/TPF Secure File Transfer Summary

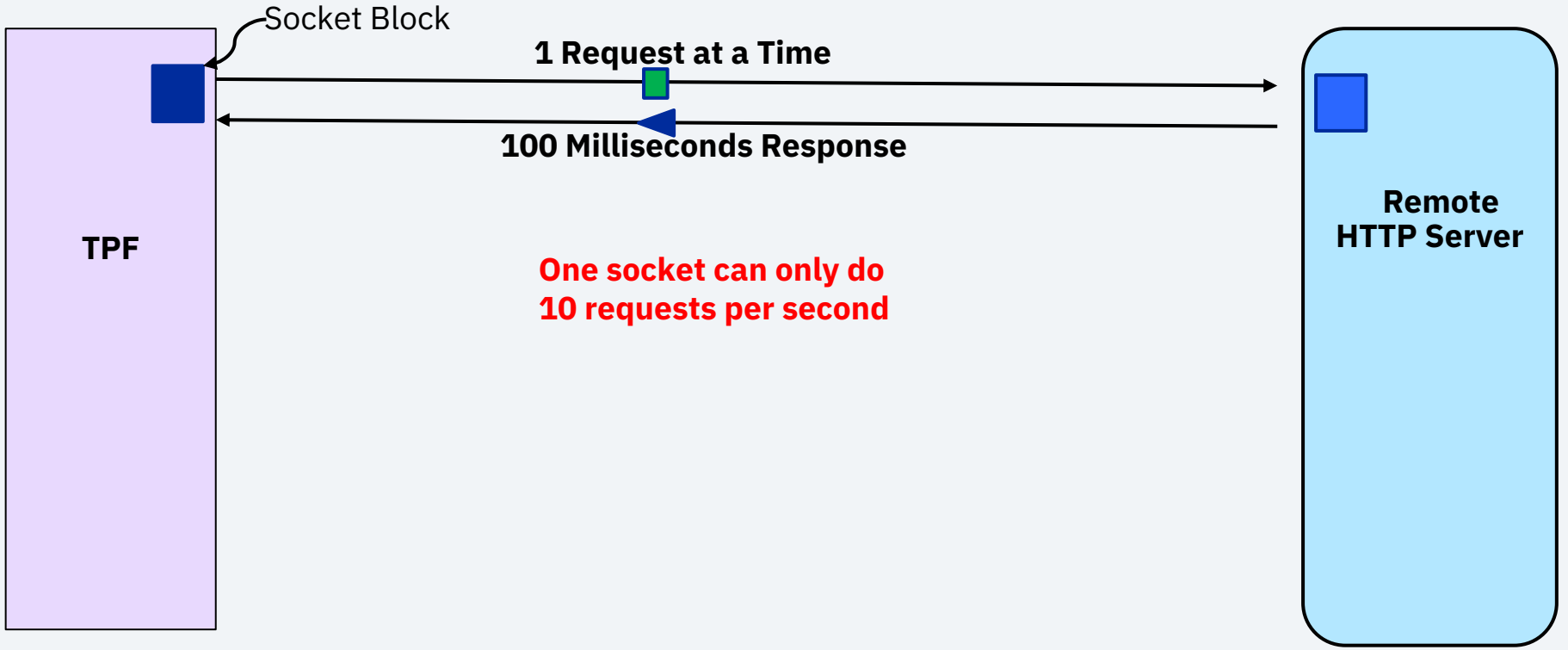
- PJ46830 (July 2023) provides a secure SSH server to securely transfer files by using SFTP.
  - System must be configured for Java to use SFTP
- Easy to manage the secure server and its users by using the new ZSSHD command.
- Use the TPF Toolkit or loadtpf to securely transfer loadsets to the z/TPF system

Disclaimer

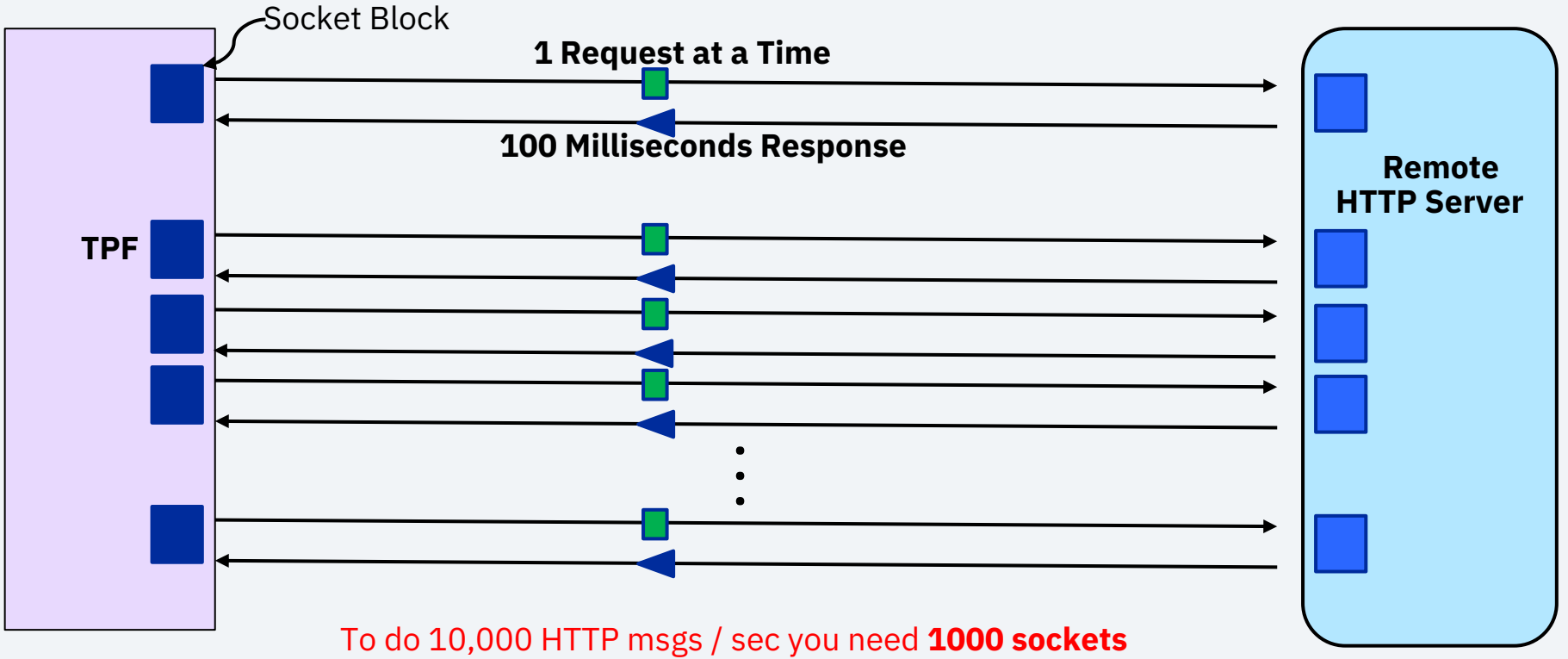
Any reference to future plans are for planning purposes only. IBM reserves the right to change those plans at its discretion. Any reliance on such a disclosure is solely at your own risk. IBM makes no commitment to provide additional information in the future.

HTTP 2.0

HTTP 1.1 Socket Pain Point – Request Reply Model

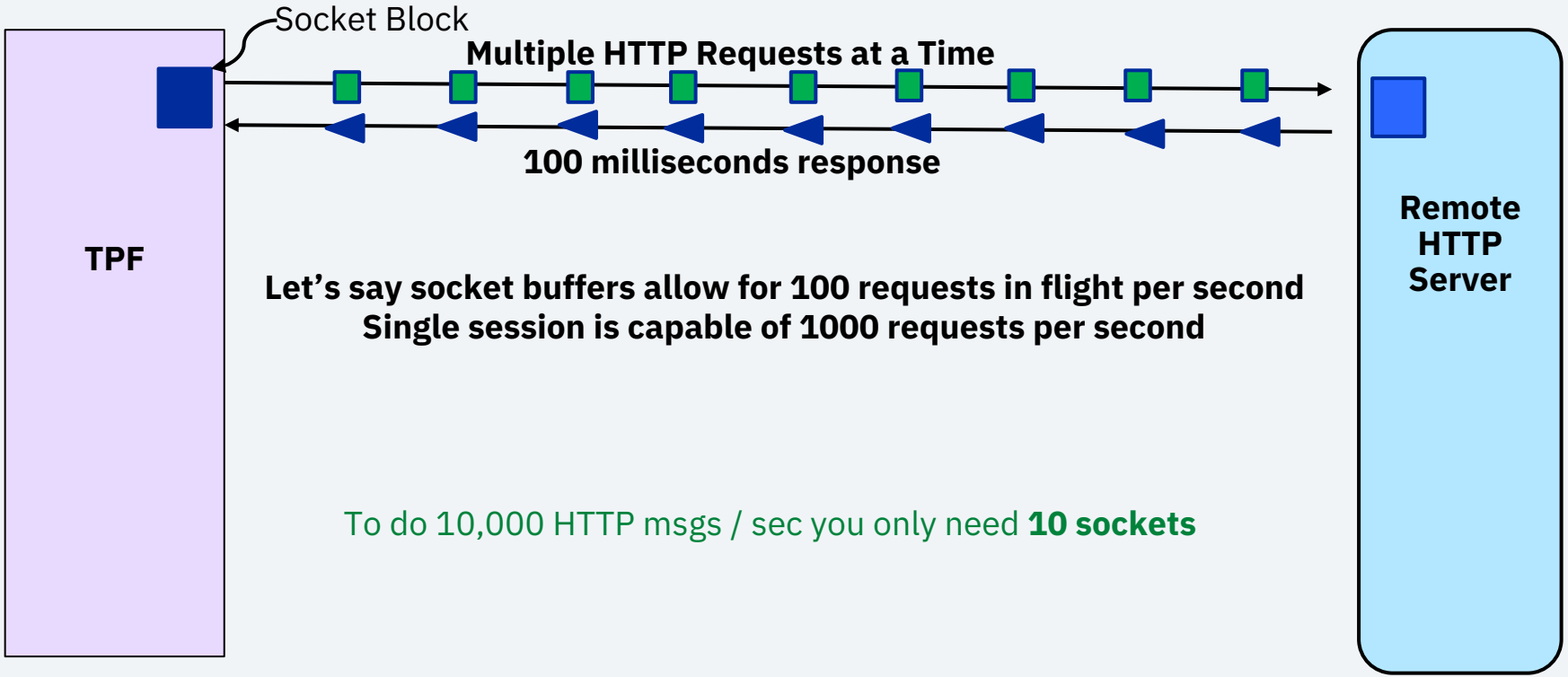


HTTP 1.1 Socket Pain Point – Hundreds of Sessions Required



To do 10,000 HTTP msgs / sec you need **1000 sockets**

Advantage of HTTP 2.0 Multiplexed Sockets



HTTP 2.0

- Why are we doing this?
  - Multiplexing and simplification of HTTP sockets and sessions on your system
  - Binary format of HTTP 2.0 should improve performance
  - Maintain currency

- Effort underway to implement a native HTTP 2.0 implementation for z/TPF.
  - Support planned for the z/TPF HTTP client and server

Be a sponsor user

Sponsor users assist in design and implementation, and your feedback drives our development cycle.

Target personas

- Application Developers
- System Administrators
- Enterprise Architects

Interested? Contact

Jamie Farmer - jvfarmer@us.ibm.com

Bradd Kadlecik – braddk@us.ibm.com

Thank you