IBM z/OS 2.4 Enhancements for Apps Leveraging REST APIs

Steve Warren, swarren@us.ibm.com
Senior Technical Staff Member
z/OS Client Architect, IBM Garage for Systems

IBM z

you IBM

Agenda

- Overview of Client Web Enablement Toolkit
  - HTTPS portion of toolkit, connection requests and response.
- Newer Functions:
  - Tracing
  - GH and samplib samples
  - AT-TLS / Toolkit Interoperability
  - Secure HTTP proxy support
  - SNI (server name identification)
  - New Patch request option
  - Recent JSON parser enhancements
- Questions and Answers

z/OS serving REST APIs

- z/OS platform has for years been labeled “the server of servers” and houses much of the world’s most critical data.
- Enhancements to the z/OS Web serving space through the years have allowed this mammoth workhorse and repository of data to be more easily accessible to other systems.

Overview of Client Web Enablement Toolkit

What about z/OS as a REST client?

- Client solutions imbedded in individual products or languages
- No generic web services or even a JSON parser available in all environments

z/OS Client Web Enablement Toolkit

The z/OS client web enablement toolkit provides a set of lightweight application programming interfaces (APIs) to enable traditional, native z/OS programs to participate in modern web services applications.

- Pieces of the toolkit:
  - A z/OS HTTP/HTTPS protocol enabler to externalize HTTP and HTTPS client functions in an easy-to-use generic fashion for user's in almost any z/OS environment
  - A z/OS JSON parser which parses and modifies JSON coming from any source, both IBM-1047 and UTF-8 encoding format.
- The toolkit allows its two parts to be used independently or combined together.
  - Payload processing is separate from communication processing.
- The interfaces are intuitive for people familiar with other HTTP enabling APIs or other parsers
- Easy for newbies
- **In base of z/OS operating system. Nothing to install!**

HTTPS portion of toolkit, connection requests and response.

z/OS HTTP/HTTPS Protocol Enabler Connections / Requests

- The HTTP/HTTPS enabler portion of the toolkit encompasses two major aspects of a web services application:
  - The ***connection*** to a server
  - The ***request*** made to that server along with the response it returns

z/OS HTTP/HTTPS Protocol Enabler
HTTP Connections

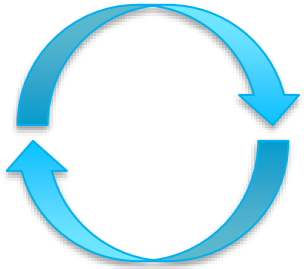
- A connection is simply a socket (pipeline) between the application and the server.
- Must be established first before a request can flow to the server.
- Many options available for connection including:
  - SSL/TLS
  - Local IP address specification
  - IP Stack
  - Timeout values

z/OS HTTP/HTTPS Protocol Enabler
Lifecycle of an HTTP Connection

- Initialize a connection (HWTHINIT)
  - Obtain workarea storage for the connection
- Set one or more connection options (HWTHSET)
  - One option at a time
- Make the actual connection (HWTHCONN)
  - Creates the socket to the specified server

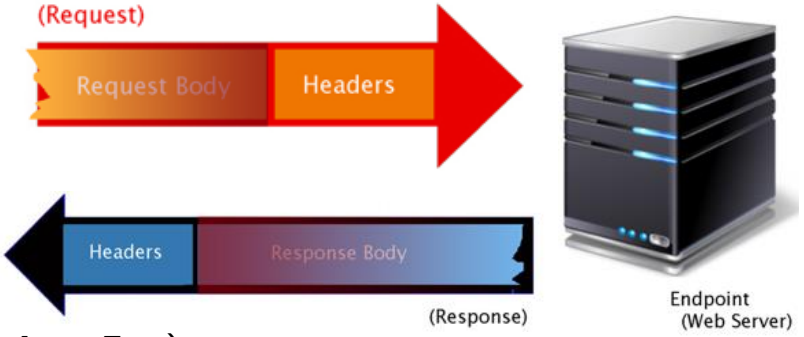
....Requests are made to the server represented by the connection.....

- Disconnect from the server (HWTHDISC)
  - Closes the socket to the specified server
- Terminate the connection (HWTHTERM)
  - Free the workarea storage



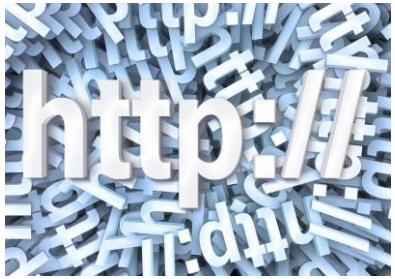
z/OS HTTP/HTTPS Protocol Enabler
HTTP Request Overview

- A client makes an HTTP request to a server (endpoint)
  - This HTTP request will typically be one of these types:
    - GET (read existing resource)
    - PUT (write/update existing resource)
    - POST (write new resource)
    - DELETE (remove existing resource)
  - May send Request Headers
  - May send Request Body (PUT and POST)



- The endpoint returns an HTTP response
  - Response consists of status (1xx, 2xx, 3xx, 4xx, 5xx)
  - Response headers
  - Response body (most requests)

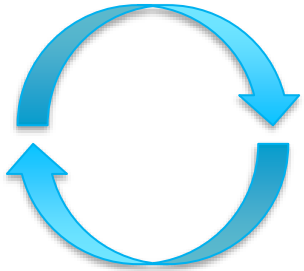
z/OS HTTP/HTTPS Protocol Enabler
HTTP Requests



- An HTTP request sent over an existing connection
  - Targets a particular resource at the domain established by the connection
  - An HTTP GET, PUT, POST or DELETE is specified as the request method
- Requests not tightly-coupled to a connection. The same request can be sent over different connections
- Response callback routines (exits) can be set prior to the request to handle returned response headers and response body.

z/OS HTTP/HTTPS Protocol Enabler
Lifecycle of an HTTP Request

- Initialize a request (`HWTHINIT`)
  - Obtain workarea storage for the request
- Set one or more request options (`HWTHSET`)
  - One option at a time
- Send the request over a specified connection (`HWTHRQST`)
  - Flows the HTTP REST API call over the connection (socket) and then receives the response
- Terminate the request (`HWTHTERM`)
  - Free the workarea storage associated with the request



z/OS HTTP/HTTPS Protocol Enabler
z/OS Client Toolkit HTTP Language Support

Include files and sample programs provided in:

- C
- COBOL
- PL/I
- Assembler (Include file only)
- REXX

Enhanced Tracing

Enhanced Tracing
Current Debugging Capabilities using Toolkit Tracing

- Turn on tracing option using `HWTH_OPT_VERBOSE` option
- `HWTH_OPT_VERBOSE_OUTPUT` option allows specification of a DD where HTTP trace output is to be directed
  - The DD name above must represent either:
    - a pre-allocated traditional z/OS data set which is a physical sequential (DSORG=PS) with a record format of unblocked variable (RECFM=V) or Undefined (RECFM=U) and expandable (non-zero primary and secondary extents). The DD must also specify a DISP=OLD disposition.
    - a zFS file.

```
t: An error occurred: Certificate validation error
t: Reason code: 8
t: Return code: -1
t: Service: 22
t: Service Instance: 0
```

Enhanced Tracing
Limitations of Toolkit Tracing Today

- One of the most challenging tasks with using toolkit is to get the first SSL/TLS handshake to work
  - Current tracing provides some details
  - System SSL tracing provides additional detailed information
    - System SSL tracing setup requires extra, sometimes time-consuming, steps. If using REXX, impossible.
- Limited tracing header information cut for each trace record
  - Time and process data not present
- Existing toolkit trace data may contain sensitive personal information
  - On query parms, request or response headers, or request/response body
- Toolkit tracing cannot be turned on non-programmatically
  - Not having access to source code or having to code tracing in code you didn't even write is problematic

Enhanced Tracing - New SSL/TLS tracing support

- New connection option to enable SSL/TLS tracing
  - `HWT_OPT_SSLTRACE`
    - Set to name of fully-qualified zFS file name where SSL tracing output should be directed.
      - `/u/steve/myTLSerrorTrace.trc` or
      - `/u/steve/myTLStrace.%.trc` where % is replaced by the process id of the process id issuing the REST API
    - Applies to application-specified SSL/TLS security connections only
      - Applications running under AT-TLS enabled policy must see AT-TLS publications to enable SSL/TLS tracing
    - Defaults to no SSL Tracing
    - Internals:
      - Toolkit will set the tracing level to maximum (GSK_TRACE = 255)
    - Output is raw tracing data (not human-readable)
  - Tracing data can easily be formatted using the System SSL GSKTRACE command (in the z/OS UNIX shell)
    - `gsktrace myTLSerrorTrace.trc > myTLSformattedTrace.trc`

Enhanced Tracing
New SSL/TLS tracing support example

Toolkit tracing:

```
2020-01-17T22:30:57.045858Z 0BC4980000000000 0083886193 0033554439 t: Unable to initialize SSL socket.
2020-01-17T22:30:57.046248Z 0BC4980000000000 0083886193 0033554439 t-Entry: error
2020-01-17T22:30:57.046252Z 0BC4980000000000 0083886193 0033554439 t: An error occurred: Cryptographic processing error
2020-01-17T22:30:57.046256Z 0BC4980000000000 0083886193 0033554439 t: Reason code: 9
2020-01-17T22:30:57.046259Z 0BC4980000000000 0083886193 0033554439 t: Return code: -1
2020-01-17T22:30:57.046262Z 0BC4980000000000 0083886193 0033554439 t: Service: 22
2020-01-17T22:30:57.046265Z 0BC4980000000000 0083886193 0033554439 t: Service Instance: 1
2020-01-17T22:30:57.046268Z 0BC4980000000000 0083886193 0033554439 t-Exit: error
2020-01-17T22:30:57.046272Z 0BC4980000000000 0083886193 0033554439 t-Entry: setReturnCode
```

System SSL tracing:

```
01/17/2020-22:35:34 Thd-0 EXIT crypto_rsa_verify_data_signature(): <--- Exit status 0x00000000 (0)
01/17/2020-22:35:34 Thd-0 INFO crypto_verify_data_signature(): Software signature verified
01/17/2020-22:35:34 Thd-0 INFO gsk_read_v3_record(): Calling read routine for 5 bytes
01/17/2020-22:35:34 Thd-0 INFO gsk_read_v3_record(): 5 bytes received
01/17/2020-22:35:34 Thd-0 INFO gsk_read_v3_record(): Calling read routine for 4 bytes
01/17/2020-22:35:34 Thd-0 INFO gsk_read_v3_record(): 4 bytes received
01/17/2020-22:35:34 Thd-0 INFO process_v3_client_handshake(): SERVER-HELLO-DONE received
01/17/2020-22:35:34 Thd-0 ERROR crypto_dh_generate_key_pair(): Key size 4096 is not supported
01/17/2020-22:35:34 Thd-0 ERROR send_v3_client_messages(): Unable to generate DH values: Error 0x03353034
01/17/2020-22:35:34 Thd-0 ERROR send_v3_alert(): Sent SSL V3 alert 51 to 9.37.138.219[62000]
01/17/2020-22:35:34 Thd-0 INFO gsk_write_v3_record(): Calling write routine for 7 bytes
01/17/2020-22:35:34 Thd-0 INFO gsk_write_v3_record(): 7 bytes written
01/17/2020-22:35:34 Thd-0 ERROR gsk_secure_socket_init(): SSL V3 client handshake failed with 9.37.138.219[62000]
01/17/2020-22:35:34 Thd-0 INFO default_setsocketoptions(): TCP_NODELAY restored for socket 3
01/17/2020-22:35:34 Thd-0 EXIT gsk_secure_socket_init(): <--- Exit status 0x00000009 (9)
01/17/2020-22:35:34 Thd-0 ENTRY gsk_strerror(): --->
```

Enhanced Tracing
Tracing able to redact sensitive personal information

- New toolkit connection option value for `HWTH_OPT_VERBOSE`
  - `HWTH_VERBOSE_OFF`
    - No tracing desired
  - `HWTH_VERBOSE_ON`
    - Tracing is enabled
    - All sensitive data is redacted
      - Query parms
      - Non-”allow-listed” header values (as specified by RFC7231)
      - Cookie values (not the cookie meta-data)
      - Any request or response body data
  - `HWTH_VERBOSE_UNREDACTED`
    - New value to see all data unredacted except “block-listed” headers (as specified by RFC7231)
      - Authorization and Proxy_Authorization headers
    - Almost identical to the old HWTH_VERBOSE_ON option value

Enhanced Tracing
Tracing redacting example

- Example showing the new tracing data using HWTH_VERBOSE_ON option:

```
2020-01-21T23:13:47.334427Z 0E36B00000000001 0083886187 0000000001 t: HWTH_OPT_VERBOSE has been set to HWTH_VERBOSE_ON
2020-01-21T23:13:47.367759Z 0E36B00000000001 0083886187 0000000001 t: GET /?[redacted] HTTP/1.1
2020-01-21T23:13:47.393080Z 0E36B00000000001 0083886187 0000000001 t-Entry: headerCallback
2020-01-21T23:13:47.393371Z 0E36B00000000001 0083886187 0000000001 t-Exit: headerCallback
2020-01-21T23:13:47.393668Z 0E36B00000000001 0083886187 0000000001 t: Header: Vary = Accept-Encoding
2020-01-21T23:13:47.393993Z 0E36B00000000001 0083886187 0000000001 t-Entry: headerCallback
2020-01-21T23:13:47.394293Z 0E36B00000000001 0083886187 0000000001 t-Exit: headerCallback
2020-01-21T23:13:47.394597Z 0E36B00000000001 0083886187 0000000001 t: Header: X-Cache = [redacted]
2020-01-21T23:13:47.400722Z 0E36B00000000001 0083886187 0000000001 t: Invoking the user specified body exit
2020-01-21T23:13:47.401035Z 0E36B00000000001 0083886187 0000000001 t: Client received 1636 byte response [redacted]
2020-01-21T23:13:47.401331Z 0E36B00000000001 0083886187 0000000001 t-Entry: finalizeResponse
```

- Example showing the tracing data using HWTH_VERBOSE_UNREDACTED option:

```
2020-01-21T23:16:16.060339Z 0E36B00000000001 0000000108 0000000001 t: HWTH_OPT_VERBOSE has been set to HWTH_VERBOSE_UNREDACTED
2020-01-21T23:16:16.063515Z 0E36B00000000001 0000000108 0000000001 t: GET /?secretstuff HTTP/1.1
2020-01-21T23:16:16.084825Z 0E36B00000000001 0000000108 0000000001 t-Entry: headerCallback
2020-01-21T23:16:16.085127Z 0E36B00000000001 0000000108 0000000001 t-Exit: headerCallback
2020-01-21T23:16:16.085444Z 0E36B00000000001 0000000108 0000000001 t: Header: X-Cache = HIT
2020-01-21T23:16:16.090112Z 0E36B00000000001 0000000108 0000000001 t: Invoking the user specified body exit
2020-01-21T23:16:16.090428Z 0E36B00000000001 0000000108 0000000001 t: Client received 1636 byte response:
2020-01-21T23:16:16.091051Z 0E36B00000000001 0000000108 0000000001 t:
Response:  First 40 (of 1256) bytes: <!doctype html>.<html>.<head>.    <title (Hex:
4c5a849683a3a897854088a394936e154c88a394936e154c888581846e15404040404ca389a39385)
2020-01-21T23:16:16.091914Z 0E36B00000000001 0000000108 0000000001 t: Last 40 (of 1256) bytes: ation...</a></p>.</div>.</body>.</html>. (Hex:
81a38996954b4b4b4c61816e4c61976e154c618489a56e154c61829684a86e154c6188a394936e15)
2020-01-21T23:16:16.092209Z 0E36B00000000001 0000000108 0000000001 t-Entry: finalizeResponse
```

Enhanced Tracing
Improved tracing header information

- Both time and process id appear in the header
  - Easier to create a timeline of what took place and when
  - Easier to trace multi-threaded applications

```
Example prefix output, ISO 8601 format (GMT):
          1    1    2    2    3    3    4    4    5    5    6    6
....5....0....5....0....5....0....5....0....5....0....5....0....5
2019-07-10T01:57:24.105881Z 0BD2580000000000 0050331656 0016777222
```

Date & time in ISO8601 format, GMT — pthread — pid — ppid

```
2020-01-17T22:30:56.623223Z 0BC4980000000000 0083886193 0033554439 t: Enabling the TLSV1.2 protocol
2020-01-17T22:30:56.623232Z 0BC4980000000000 0083886193 0033554439 t: Disabling the TLSV1.3 protocol
2020-01-17T22:30:56.623241Z 0BC4980000000000 0083886193 0033554439 t: Setting SSL key database to: *AUTH*/*
2020-01-17T22:30:56.663425Z 0BC4980000000000 0083886193 0033554439 t-Entry: ignoreSignal
2020-01-17T22:30:56.663438Z 0BC4980000000000 0083886193 0033554439 t: now ignoring signal: SIGPIPE
2020-01-17T22:30:56.663441Z 0BC4980000000000 0083886193 0033554439 t-Exit: ignoreSignal
2020-01-17T22:30:56.663445Z 0BC4980000000000 0083886193 0033554439 t: Invoke gsk_secure_socket_open()
2020-01-17T22:30:56.663455Z 0BC4980000000000 0083886193 0033554439 t-Entry: restoreSignal
2020-01-17T22:30:56.663458Z 0BC4980000000000 0083886193 0033554439 t: restoring signal: SIGPIPE
2020-01-17T22:30:56.663462Z 0BC4980000000000 0083886193 0033554439 t-Exit: restoreSignal
2020-01-17T22:30:56.663471Z 0BC4980000000000 0083886193 0033554439 t: No applicable peerid.
2020-01-17T22:30:56.663475Z 0BC4980000000000 0083886193 0033554439 t: hostIsName() is TRUE 'barney.rtp.raleigh.ibm.com'
2020-01-17T22:30:56.663505Z 0BC4980000000000 0083886193 0033554439 t: Applied SNI extension for 'barney.rtp.raleigh.ibm.com'
```

Enhanced Tracing
New non-programmatic way to turn on tracing

- Great for when you don’t have access to the source code or when modifying the program to enable tracing could be challenging
- User can specify runtime environment variables
  - `HWTH_OPT_VERBOSE`
  - `HWTH_OPT_VERBOSE_OUTPUT`
  - `HWTH_OPT_SSLTRACE`
- Values specified will override any tracing options specified in the toolkit application
  - Even if tracing is turned explicitly off in application, user can enable tracing

Enhanced Tracing
How do I set these runtime environment variables to set the tracing options?

- Application running in LE environment
  - Use the z/OS UNIX export command
- Application running in non-LE environment
  - Set the variables by using the CEEOPTS DD statement
- TSO example:
  - Data set `JOEUSER.TRACING.OPTIONS` contains the options:

```
ENVAR("HWTH_OPT_VERBOSE=HWTH_VERBOSE_UNREDACTED",
      "HWTH_OPT_VERBOSE_OUTPUT=MYDD",
      "HWTH_OPT_SSLTRACE=/u/joeuser/gskssl.trc")
```

  - TSO user allocates the 2 required DDs: for CEEOPTS ('JOEUSER.TRACING.OPTIONS') and MYDD ('u/joeuser/joe.trc'))
  - You will have unredacted tracing stored in `/u/joeuser/joeuser.trc` and SSL/TLS tracing stored in `/u/joeuser/gskssl.trc`.

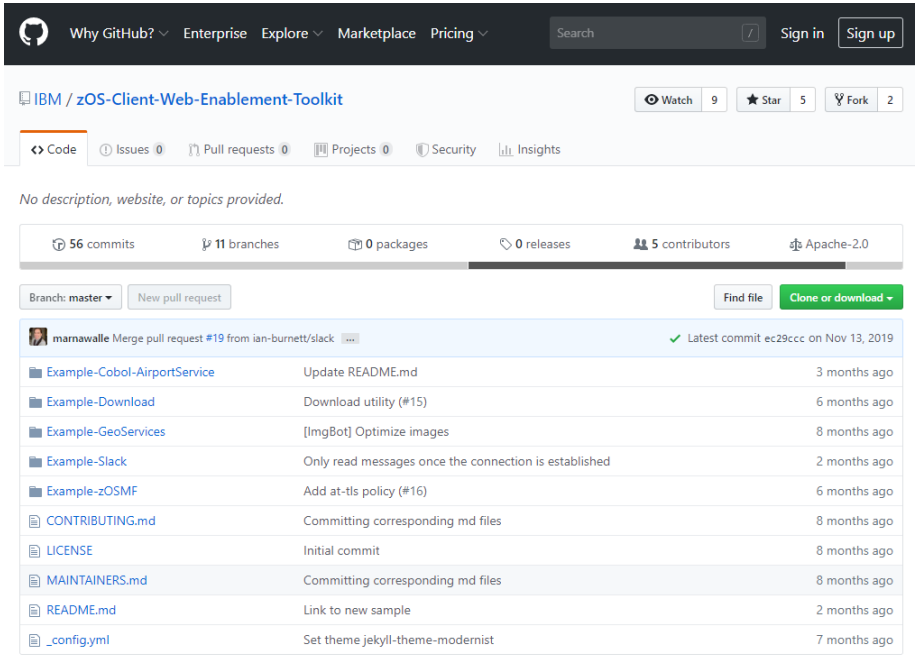
Availability of tracing enhancements

- **All tracing enhancements available in APAR OA58707**
  - **V2R3 and higher (by end of 1Q20)**

GH samples and samplib

z/OS Web Enablement Toolkit Samples on Github!

https://github.com/IBM/zOS-Client-Web-Enablement-Toolkit



- GeoServices
  - Demonstrates how to use the toolkit to obtain the distance between two cities using the Geo Services REST API.
- Download
  - Demonstrates how a native z/OS application can use toolkit to download content from a REST API endpoint.
- z/OSMF
  - Shows how to use a sampling of different z/OSMF REST APIs
- Slack
  - Shows how to use the toolkit to post a message to a Slack channel.
- New airport service
  - Shows how to use the toolkit to get descriptive information about an airport using a 3 character IATA as input.

Revamped Web Toolkit Samplib samples

- Replacing FAA sample with simpler sample
  - Target **http://example.org** website
  - Shipping COBOL, REXX, C, and PL/I samples
- Minor fixes to existing samples
- **Available in APAR OA57475 (V2R3 and higher)**

AT-TLS / Toolkit Interoperability

HTTP Services – AT-TLS / Toolkit Interoperability

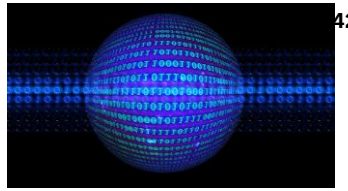
- Application Transparent – TLS is basically stack-based TLS
  - TLS process performed in TCP layer (via System SSL) without requiring any application change (transparent)
  - AT-TLS policy specifies which TCP traffic is to be TLS protected based on a variety of criteria
    - Local address, port
    - Remote address, port
    - z/OS userid, jobname
    - Time, day, week, month
  - Gives network administrators greater control over the security requirements of network applications rather than individual applications

AT-TLS

HTTP Services – AT-TLS / Toolkit Interoperability

- Toolkit is now AT-TLS aware
  - Application *does not specify* SSL/TLS options directly within toolkit application?
    - **AT-TLS policy upgrades connection to SSL/TLS?**
      - Toolkit will treat the requests over this connection as HTTPS requests
      - All cookies and redirect processing will be now operate as an HTTPS request.
    - **AT-TLS policy does not upgrade connection or no policy in effect?**
      - Business as usual. Request will operate as HTTP
  - Application *specifies* SSL/TLS directly within toolkit application?
    - **AT-TLS policy upgrades connection to SSL/TLS?**
      - Toolkit rejects the request. Network configuration and application are in conflict.
    - **AT-TLS policy does not upgrade connection or no policy in effect?**
      - Business as usual. SSL/TLS credentials will be specified by the application. If successful handshake, then request will operation as HTTPS.
  - **Available in APAR OA50957 (V2R2 and higher)**

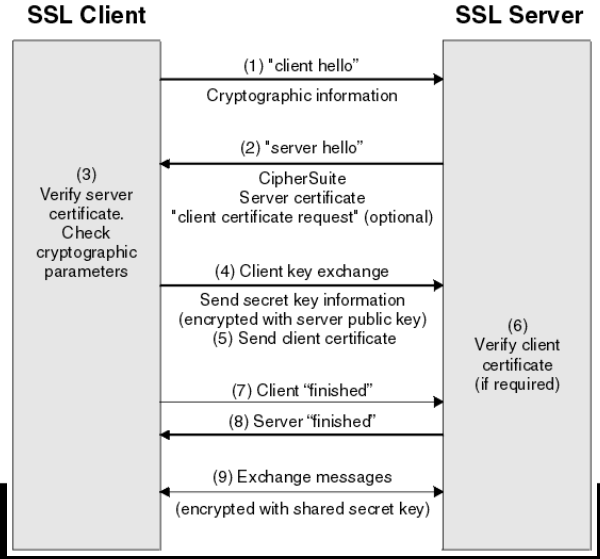
New SSL Cipher Specs support



- When a secure connection is established, the client and server negotiate the cipher to use for the connection (RFC5246). These ciphers help determine how the data will be encrypted and decrypted.
- The web server has an ordered list of ciphers, and the first cipher in the list that is supported by the client is selected.
- New HWTH_OPT_SSLCIPHERSPECS option allows the application to specify a list of 4-character cipher definitions
  - Should be ordered by preference of use
  - Requires HWTH_OPT_USE_SSL option to be set to HWTH_SSL_USE (application-initiated SSL connection)
- Allows the client application to replace the default list of acceptable cipher specifications with its own list
- **Available in APAR OA53546 (V2R2), in base V2R3**

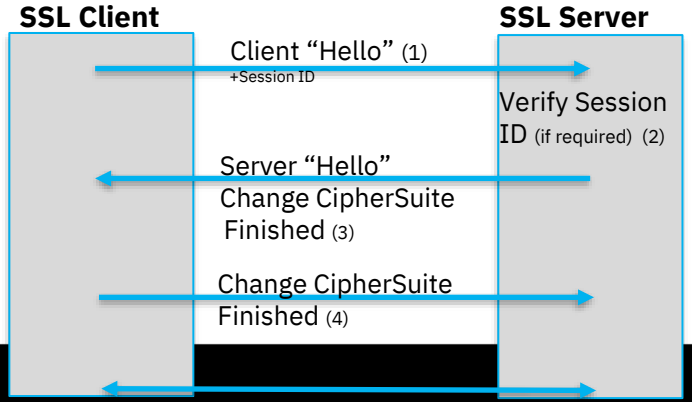
New SSL TLS 1.2 optimization support

- Full TLS/SSL handshake (prior to TLS 1.2)
  - High latency
  - Two round-trips required
  - Expensive computation to exchange keys or sign and verify certificates



New SSL TLS 1.2 optimization support

- Abbreviated (short) handshake (TLS 1.2)
  - The full handshake is required at least once.
  - The full handshake results in server sending a **session ID** back to the client.
  - This ID is cached on the server and by the toolkit (client-side)
  - If new request is made to the same server and the connection is no longer there, the toolkit attempts to send this cached session ID as part of the new handshake.
  - If the server accepts this session ID, the server quickly completes the handshake, bypassing most of the full handshake steps.
  - If the server does not accept the session ID, a full handshake will result.



New SSL TLS 1.2 optimization support…

- For application-initiated SSL connections
  - The toolkit will use the abbreviated handshake whenever it is possible to resume a previously established secure connection.
- Connections using AT-TLS can also avail themselves of this optimization automatically when running on V2R2 or higher
- Toolkit optimization will only be available if a connection has not been disconnected. Use cases include:
  - A server times out a connection. A request is then attempted over this timed-out connection.
  - A server sends a Connection Closed response header (or fails to specify Keep-Alive (HTTP 1.0)). A request is then attempted over this closed connection.
- **Available in APAR OA53546 (V2R2) – base V2R3**

New SSL TLS 1.3 support

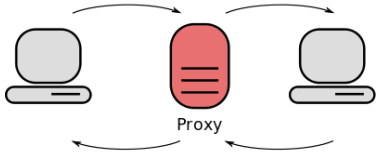
- Improved security and speed
  - State-of-the-art cryptography
  - Potentially less data flows between 2 parties
    - Handshakes are generally only a single round-trip
  - Most handshake messages are encrypted
  - Reduction in algorithms, deprecated features
  - Many other enhancements
- z/OS support is only on z/OS V2R4
- New option value for `HWTH_OPT_SSLVERSION` option
  - `HWTH_SSLVERSION_TLSv13`
    - Applies to toolkit application-specified security
- **Available in APAR OA58708**
  - **V2R4 only**

Secure HTTP proxy support

HTTP Proxy Enhancements (Authenticating Proxy Support)

- Anytime a proxy is used, the following *existing* options **must** be specified to set the proxy address and port:
  - **HWTH_OPT_PROXY**
  - **HWTH_OPT_PROXYPORT**

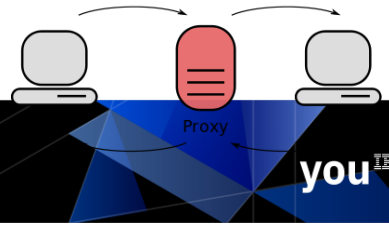
- If the proxy is an “authenticating proxy”, “Basic authentication” credentials for the proxy can be specified via new toolkit options:
  - **HWTH_OPT_PROXYAUTH**
    - **HWTH_PROXYAUTH_NONE**
      - No proxy authorization is used. Default.
    - **HWTH_PROXYAUTH_BASIC**
      - Use basic proxy authentication. **HWTH_OPT_PROXYAUTH_USERNAME** and **HWTH_OPT_PROXYAUTH_PASSWORD** will be sent to proxy in Basic auth format.
  - **HWTH_OPT_PROXYAUTH_USERNAME**
  - **HWTH_OPT_PROXYAUTH_PASSWORD**



HTTP Proxy Enhancements (AT-TLS Proxy Support)

- AT-TLS can now be used to secure HTTPS connections that go through a proxy
  - Network admin defines connection with the proxy address (defined by **HWTH_OPT_PROXY** and **HWTH_OPT_PROXYPORT**) to an AT-TLS "application-controlling" policy
  - This policy must specify a keyring that is suitable for any and all HTTPS destinations that will be reached via that proxy
  - No need to map a specific HTTPS destination to an AT-TLS
    - Unless application connects to it directly without a proxy
- **Both available with APAR OA54902 (V2R2 and V2R3)**

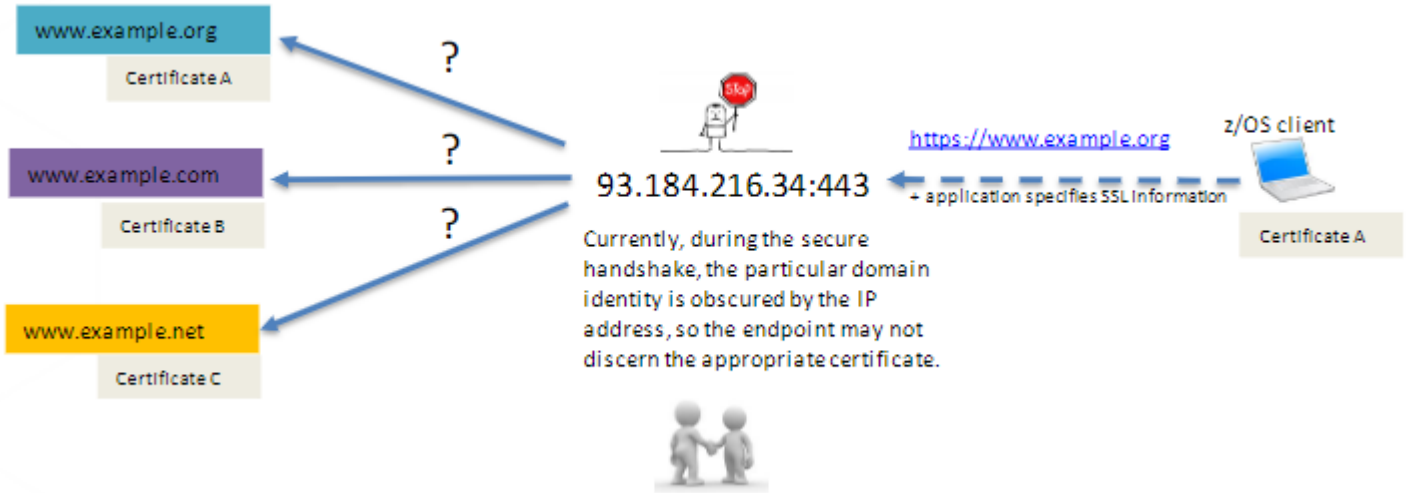
AT-TLS



SNI (Server Name Identification)

Server Name Indication (SNI)

- Allows a single IP address to support multiple domain names
  - Each domain within a single IP address can have its own unique SSL certificate



Server Name Indication (SNI)

- When application specifies SSL/TLS security options (i.e. HWTH_SSL_USE is turned on)
  - Connection will automatically include an SNI extension
    - Provided the Connection URI is in Domain Name System (DNS) format
- Server will negotiate the SSL/TLS handshake using the proper certificate for the particular domain.
- **Available with APAR OA54902 (V2R2 and V2R3))**

New PATCH and OPTION HTTP Method Support

Current Toolkit HTTP Method Support

- The basic REST API (CRUD) methods
  - Create (POST)
  - Read (GET)
  - Update (PUT)
  - Delete (DELETE)
- Other HTTP method support
  - HEAD (same as a GET but without the response body)

Digging a little deeper with updating methods

- POST
  - Requests that the server accept the entity enclosed in the request as a new subordinate of the web resource identified by the URI. The newly created entity is automatically returned in the Location response header.
- PUT
  - Requests that the enclosed entity be stored under the supplied URI. If the URI refers to an already existing resource, it is modified; if the URI does not point to an existing resource, then the server can create the resource with that URI**. A complete replace**

What if I don’t want a complete replace?

- PATCH
  - Applies partial modifications to a resource
  - Uses instructions in the request body
  - May create a collection or member if it does not exist

Example contrasting **PUT** and PATCH behaviors

Resource before PUT:

```
{ “id”: 1,
  “name”: “John Smith”,
  “email”: “steve@olddomain.com”,
  “phone”: “+1 914 475 6308”
}
```


```
PUT /users/1
{“email”: “steve@newdomain.com”}
```

Resource after PUT:

```
{“email”: “steve@newdomain.com”
 }
```

Example contrasting PUT and **PATCH** behaviors

Resource before PATCH:

```
{ “id”: 1,
  “name”: “John Smith”,
  “email”: “steve@olddomain.com”,
  “phone”: “+1 914 475 6308”
}
```

```
PATCH /users/1
{“email”: “steve@newdomain.com”}
```

Resource after PATCH:

```
{ “id”: 1,
  “name”: “John Smith”,
  “email”: “steve@newdomain.com”,
  “phone”: “+1 914 475 6308”
}
```

What if I don’t know what HTTP request methods are available on a server?

- OPTIONS HTTP Method returns the HTTP methods the server supports for the specified URL
  - Can also be used to check the functionality of a web server by requesting “*” instead of a specified resource
- Example:

```
OPTIONS *                    or
OPTIONS  /user/1
```

Response Header with a 204 (No Content HTTP Status Code):

```
Allow:  OPTIONS, GET, HEAD, POST
```

Toolkit support of PATCH and OPTIONS

- New `HWT_OPT_REQUESTMETHOD` option values
  - `HWTH_HTTP_REQUEST_PATCH`
  - `HWTH_HTTP_REQUEST_OPTIONS`
- **Available in APAR OA58707 in V2R3 and higher (by end of 1Q20)**

Recent JSON Parser Enhancements

Recent JSON parser enhancements

- UTF-8 Support
  - **Available with APAR OA56139 (V2R2 and V2R3)**
- JSON Parser Delete Entry
  - **Available in APAR OA54901 (V2R2 and V2R3)**
- JSON Parser Pretty Print
  - **Available in APAR OA55438 (V2R2 and V2R3)**
- Shallow Search
  - **Available in APAR OA56227 (V2R2 and V2R3)**

Current JSON parser

{JSON}
JavaScript Object Notation

- Limited to EBCDIC (IBM 1047 codepage) JSON text
- JSON text received on z/OS in ASCII (ISO 8859-1 codepage) easily translated to EBCDIC since there is 1-to-1 mapping (parser happy ☺)
- JSON text received in UTF-8 may require extraordinary effort to convert to IBM 1047
- JSON official RFC7159 states:
  - Section 8.1 Character Encoding
    - “JSON text SHALL be encoded in UTF-8, UTF-16, or UTF-32. The default encoding is UTF-8…”

New JSON parser support for UTF-8

- JSON parser supports EBCDIC (IBM 1047) or UTF-8 encodings
- Parser will attempt to auto-detect the encoding and process appropriately
- Services also provided to manually set and retrieve the encoding of the JSON text.
- Encodings cannot be commingled
- No changes required for existing toolkit JSON parser applications already dealing with EBCDIC.

New JSON parser support for UTF-8 (Retrieving the JSON text encoding)

- New HWTJGENC service
  - Returns 3 possible encoding values
    - **HWTJ_ENCODING_UTF8**
    - **HWTJ_ENCODING_EBCDIC**
    - **HWTJ_ENCODING_UNKNOWN** (prior to parsing JSON text)

```
address hwtjson "hwtjgenc",
                    "ReturnCode",
                    "ParserHandle",
                    "Encoding",
                    "DiagArea."
```

New JSON parser support for UTF-8 (Assert the JSON text encoding)

- New HWTJSENC service
  - Issued after HWTJINIT and before HWTJPARS or HWTJCREN issued to assert to encoding of the soon to be supplied JSON text
  - 2 encoding values allowed
    - **HWTJ_ENCODING_UTF8**
    - **HWTJ_ENCODING_EBCDIC**

```
address hwtjson "hwtjsenc",
                "ReturnCode",
                "ParserHandle",
                "Encoding",
                "DiagArea."
```

New JSON parser support for UTF-8 (Assert the JSON text encoding)

- What if the asserted encoding and the actual encoding don't match?
  - **Parse existing JSON Text - HWTJPARS()**
    - Return code - HWTJ_PARSE_ERROR
    - DiagArea Reason Code - PARSE_ERR_UNEXPECTED_ENCODING
  - **Creating new JSON Text – HWTJCREN() and supplying JSON Text via HWTJ_JSONTEXTVALUETYPE**
    - Return code - HWTJ_PARSE_ERROR
    - DiagArea Reason Code - PARSE_ERR_UNEXPECTED_ENCODING

New JSON parser support for UTF-8 (Other considerations)

- **Search service - HWTJSRCH()**
  - The name to be searched is expected to be in same encoding as the JSON text encoding detected
    - Not enforced – a differently-encoded search string will likely result in “not found” condition
- **Create service - HWTJCREN()**
  - If modifying JSON text which was already parsed, the data supplied is expected to be in the same encoding as the JSON text encoding detected
    - Not enforced – Comingling of data not of HWTJ_JSONTEXTVALUETYPE can occur.
      - e.g. Adding a string value “1” in EBCDIC to a UTF-8 JSON text will set the value to be “ñ” (F1 (241) in the UTF-8 codepage)

JSON Entry Delete

Support for JSON Parser Delete Entry

- Current support
  - Create JSON entry (HWTJCREN)
    - Add object, array, string, number, boolean, null
    - Add JSON text directly
- Shortfall
  - No corresponding delete function
    - Requires manual parsing and using library functions to delete an object from the JSON text (error prone and difficult)
  - RFE 82349 written to address the lack of a JSON delete entry service
  - Other internal and external requests for this function

Support for JSON Parser Delete Entry – Syntax


```
HWTJDEL(
 returnCode,
parserHandle,
objectHandle,
entryValueHandle,
diagArea
)
```

objectHandle – handle of the object containing the entry to be deleted
entryValueHandle – handle of the specific entry to be deleted

Support for JSON Parser Delete Entry – Usage

- Can be used to delete ***simple*** entry values
  - Simple values can be represented by entryValueHandle handle
    - If the entryValueHandle represents a string, number, boolean or null value, the entire entry is removed from the JSON string
    - `{"IBM products owned": [ "Db2", "IMS", "CICS", "Product A", "WebSphere"]}`
    - To delete "`Product A`", specify:
      - the handle of the array as the `objectHandle`
      - the handle of the value "Product A" as the `entryValueHandle`
    - `{"IBM products owned": [ "Db2", "IMS", "CICS", "WebSphere"]}`

Support for JSON Parser Delete Entry – Usage

- Can be used to delete ***complex*** entry values
  - Complex values can be represented by entryValueHandle
    - If the entryValueHandle represents an object or array, the entire object or array is removed from the JSON string
    - `{"IBM products owned": [ "Db2", "IMS", "CICS", "Product A", "WebSphere"]}`
    - To delete the entire "`IBM products owned`" array, specify:
      - the handle of the containing object (in this case root handle) as the `objectHandle`
      - the handle of the array as the `entryValueHandle`
    - `{}`

Support for JSON Parser Delete Entry – Availability

- New samples shipped in SYS1.SAMPLIB showing the usage of the HWTJDEL service:
  - HWTJXRX2 (REXX)
  - HWTJXC2 (C)
  - HWTJXCB2 (Cobol)
- **Available in APAR OA54901 (V2R2 and V2R3)**

JSON Parser Pretty Print

JSON parser Pretty Print

- User can create and modify JSON stream via the JSON parser APIs
- User can then serialize the JSON streaminto an output buffer
  - JSON text created, but…
  - JSON text is not formatted

- Text is very difficult to read
- There is a need to transform this data into human-readable format

JSON parser Pretty Print - Syntax

- Pretty print formatter shipped as REXX exec in both:
  - SYS1.SAMPLIB(HWTJSPRT)
  - /samples/jsonprint
- Syntax:
  - `HWTJSPRT 'JSON.file(member)'`

JSON parser Pretty Print

- Example of generated JSON text using HWTJSERI before pretty print:

```
{"errors":[{"status": "422","title":  "Invalid Attribute"}]}
```

- Example of generated JSON text after running pretty print formatter:

```
{
  "errors": [
    {
      "status"           : "422" ,
      "title"            : "Invalid Attribute"
    }
  ]
}
```

- **Available in APAR OA55438 (V2R2 and V2R3)**

Enhanced JSON Parser Search Service

Enhanced JSON Parser Search Service

- Current JSON parser search (HWTJSRCH) has two flavors:
  - ***HWTJ_SEARCHTYPE_GLOBAL***
    - Search the JSON text starting at the entry represented by the startingHandle parameter for the first “name” in a name/value pair found that matches the search string
    - **Object-ignorant** (no scoping of the search to be within the object specified by the startingHandle parameter)
  - ***HWTJ_SEARCHTYPE_OBJECT***
    - Search the JSON text starting at the entry represented by the startingHandle parameter for the first “name” in a name/value pair found that matches the search string
    - **objectHandle parameter scopes the search**, limiting the search to be within the object represented by the objectHandle parameter
- Limitation of both search types
  - ***No way to limit the depth of the search***
  - ***No way to restrict the scope of the search to only search the immediate children of a given object (aka a "shallow" search)***

Enhanced JSON Parser Search Service

- Example comparing new HWTJ_SEARCHTYPE_SHALLOW with existing HWTJ_SEARCHTYPE_OBJECT:

```
{
 "a": "A1",
 "b": {
       "c": "C1",
       "d": {
             "c": "C2",
             "e": "E1",
             "f": "F1"
            },
       "e": "E2"
      },
 "c": "C3"
}
```

- **Search for “a” or “b” from the root with either search will return the same value**
- **Search for “c” from the root:**
  - *HWTJ_SEARCHTYPE_OBJECT will return the handle for “C1”*
  - *HWTJ_SEARCHTYPE_SHALLOW will return the handle of “C3”*

Questions and Answers