SZTPD Alpha-9 Release Notes

Watsen Networks

February 4, 2021

Abstract

This documentation provides release notes for the Alpha-9 release of SZTPD.

Contents

1 Introduction

The sections below cover what is and is not working in this release of the SZTPD product.

Most everything listed as "not working" will be implemented before FCS.

2 Implemented / Tested

This section describes what is working in SZTPD. All of the following has been tested[1]:

- All APIs work: 'native', 'tenant', and 'rfc8572'.
- Each listening port is HTTP with or without TLS[2].
- Client auth can be TLS client-certs and/or HTTP basic auth.
- Both XML and JSON supported on SBI. Strong HTTP header checking.
- RESTCONF HEAD, GET, POST, PUT, and DELETE work over entire tree.
- Ordered-by user query parameters ('insert' and 'point') work.
- Pagination query parameters ('limit', 'offset', and 'direction') work.
- The ./well-known/host-meta, RESTCONF root (i.e., {+restconf}), and YANG-library resources.
- Tested using in-memory, file-base, MySQL (inc. AWS Aurora), and Postgres databases.
- TLS connection to backend RDBMS, with or w/o client certificate.
- Plugin-based dynamic callouts to relay notifications
- Plugin-based dynamic callouts to relay progress reports
- Plugin-based dynamic callouts for ownership verification
- Plugin-based dynamic callouts for response manager[3].
- Audit log (including per-tenant audit log)
- Bootstrapping log (including information about SZTPD's relay to remote systems)
- Both Enterpise (mode-1) and Service Provider modes work (modes '1' and 'x')
- Database-level transactions and concurrent access.
- Python 3.7, 3.8, and 3.9 (current)

3 Should Work

This section regards things that should work, but haven't been tested yet.

- IPv6: The default port binds to "127.0.0.1", but this can be changed by setting the "SZTPD_DEFAULT_ADDR" environment variable. Other than this, there is no other IPv4-only code in the product as everything is handled by Python modules.
- UTF-8: Python string types support both ASCII and unicode, like UTF-8. Presumably string handling is ambivalent, but this isn't tested yet.
- Windows: the software has been developed and tested exclusively in UNIX based systems. Python is very portable, and SZTPD has almost no interaction with the filesystem, so running on Windows might work, but this has not been tested.

[1] There is more than twice the number of lines of test code than code in SZTPD itself.
[2] An external TLS-terminator must be used when an SZTPD listening port is configured to NOT use HTTP.
[3] Plugin-based dynamic callouts have been used to test support for draft-kwatsen-netconf-sztp-csr.

4 Upcoming Features/Releases:

The following features are sorted by the expected release they might show up in. Please let us know if there is something out of place or missing.

4.1 Alpha-10

- Introduce a "SZTPD_ACCEPT_CONTRACT" env var to explicitly disable to the contract "click thru". Currently, the "SZTPD_MODE" env var implicitly disables the "click thru", but lawyers say that it should be explicit.
- Improve the Pagination API introduced in 0.0.8. The pagination standard is a work-in-progress at the IETF led by Watsen Network's founder, Kent Watsen. This effort may extend the "limit", "offset", and "direction" parameters to leaf-lists and introduce support for the "sort-by", "where", and "sublist-handling" query parameters.
- init perf / scale / soak tests

4.2 Alpha-11

- RESTCONF entity-tag (ETag) Support. The HEAD and GET operations would return the "ETag" HTTP header field. The POST, PUT, and DELETE operations would inspect request headers for the "If-Match", and "If-None-Match" fields. This enables concurrent clients to detect if changes are have been made since their using a HEAD or GET command. Support for Etag is a MUST in RFC 8040 only on the top-level datastore resource, and unspecified for inner-resources[4]

4.3 Alpha-12

- Implement the "ietf-restconf-monitoring" per Section 9 of RFC 8040. This is needed to identify which capabilities (e.g., query params) and streams (e.g., the notifications) the SZTPD server supports.
- Tenant view device-types and plugins. Currently, tenants can set 'device-types' and plugin-based 'dynamic-callouts' only if the values configured by the system administrator are provided out-of-band. For security and stability reasons, these values are only set at the "host" level, outside the tenant's purview. A tenant's clients should be able use either an RPC/action to get the supported device-types from the host, or have that information exposed as opstate (i.e., read-only "config false" values)[5].

[4] Note that the top-level resource for the tenant-view is mounted to /tenants/tenant.

[5] Current plan is to do the latter, but due to limitations in YANG, need to split the sztpd-1 schema into two: one for 'native' view and another for the 'tenant' view.

4.4 Alpha-X

- Authorization / access-control. It is planned to implement the admin-account "access" node, which sets an enumerated value being one of "unrestricted", "typical", and "minimal".
- Implicit change tracking. This is needed to support a few of the features listed below. It is also needed to detect when *any* part of the "transport" configuration changes, so that a SIGHUP can be issued. Currently SIGHUP is issued only when an "endpoint" is added or removed.
- Bootstrapping event counter. SZTPD needs to maintain bootstrapping event counters.
- Device record counters. It is planned to track when the device records are created, last modified, and the total number of modifications. Similarly, to track when the bootstrapping device first connected, last connected, and the total number of connections.
- Reference tracking. It is desired to track references to objects in the system. In YANG terms, these are the 'leafref' statements that reference a 'list' statement's 'key' node. Tracking includes the current number of references and the time of last reference.
- Automated purging with notifications. It is desired to send notifications when unreferenced objects have been without a reference, for some configurable amount of time. A series of notifications with escalating urgency can be to sent to configurable notification receivers. A final notification is sent when the purging occurs.
- Password expirations with notifications. This feature goes with the previously mentioned "automated purging with notifications" feature, but has its own "preferences" setting for the timeouts.
- Email-based admin activations. It is intended that SZTPD will send email based notifications to newly created admins when their accounts are first created. It is important to validate the email address because the same email address may be emailed later when the password expiration date is approaching. This is why the email address is the primary key for the admin accounts.
- Password minimum length constraints. Currently the "preference" setting for the admin account password's minimum length constraint is ignored.
- send notifications, as currently none are.

4.5 Beta

- Run performance and soak tests. Only address issues found.

4.6 FCS

- Nothing new
- Stress tests
- Soak tests

4.7 Post 1.0

- Remove support for mode '1'. This change would affect the NBI only (no impact on the device-facing 'rfc8572' SBI).
  - Reasons to remove mode '1':
    1) It may be important for non-production use environments (e.g., those used for evals and demos) to exactly mimic production environments.
    2) Eliminates an artificial constraint for Mode '1' deployments, in they can easily configure an additional "tenant" for whatever reason. It also eliminates need to ever have to migrate mode-1 deployments to a mode-x deployment, which would require a database migration..
    3) The tenant-view provided by Mode 'x' quite nicely isolates system-level configuration enabling IT organizations to do the system-level install and then pass an "application" view that excludes the system-level install to another organization within the company.
    4) Only supporting Mode-x could greatly simplify the YANG modules. This could be important to developers as they might need to look at the YANG modules in order to understand the data model exposed by the API. While the current YANG is navigable, it could be even more so if only supporting Mode-x.
    5) Further simplify the YANG modules. Note that mode '0' is already almost completely phased out, and yet the YANG modules have yet to be simplified…somewhat due to waiting to determine if also phasing out mode '1'.
  - Reasons to NOT remove mode '1'[6]:
    1) It is unclear what the market expectations are regarding metered based subscriptions versus flat-rate tiers for enterprises and service providers. Hesitant to change anything without some feedback.
- Run the "verify-device-ownership" callouts at time of bootstrapping event (in addition to when the device record was first created). Seems like something that should be opt-ed into, and hence a feature that can be implemented later.
- Webhook-based callouts dynamic callouts. Some webhooks were implemented before, but better to re-build them generically on top of plugin-based callbacks. Clients can use plugin-based callbacks for now as well.
- Client certificate based auth to NBIs. Currently SZTPD implements client cert based auth to the SBI (i.e, 'rfc8572-interface'). This is to extend that configuration into the NBIs also (i.e., 'native' and 'tenant'). This would provide 2FA to the NBI.
- RESTCONF "Last-Modified" support. The HEAD and GET operations would return the "Last-Modified" HTTP header field. The POST, PUT, and DELETE operations would inspect request headers for the "If-Modified-Since" and "If-Unmodified-Since" fields. Support for timestamps is a SHOULD in RFC 8040.
- RESTCONF Filtering (depth + fields) query parameters. These query parameters are described by Section 4.8 of RFC 8040 to reduce the amount of data returned to the client. RFC 8040 does not require these query parameters to be supported. Workaround is for the client to discard the unwanted parts of the response itself.
- RESTCONF: support the "content" query parameter. Required by RFC 8040, but very low value for SZTPD, because no "config true" values have interesting values in <operational> and most "config false" values should be queried directly…
- The RESTCONF/HTTP "PATCH" method. Though a MUST in RFC 8040, it seems mostly like a nice-to-have in SZTPD…

[6] if decide to keep mode '1', the YANG can still be simplified by removing support for mode '0'.

- Support a callout to retrieve an ownership voucher from an external system. This would implement the "supply-ownership-voucher" RPC defined in the "wn-sztpd-callbacks" module. The RPC is currently protected by a 'feature' statement called "supply-ownership-voucher", thus programmatically signalling that it is not supported, though visible in the YANG.
  - Support signing conveyed information sent from SZTPD using the private key associated with a configured owner certificate.
  - Support encrypting conveyed information sent from SZTPD using the device's public key from its identity certificate (e.g., IDevID).
  - Support stapling revocation responses to CMS objects returned to devices.
    - only needed for signed data? (what about the redirect-info's trust-anchor CMS?)
  - Private key encryption. It is desirable for SZTPD to have an ability to encrypt private keys via a "root" key. Note that this is above and beyond DB-level encryption; it purpose to to shield keys even from administrators having appropriate access. This root key would be protected by access control and/or an HSM.
  - Actions: None of the 'action' statements defined in the YANG modules are implemented yet. Primarily this effects the ability to create keys, and generate certificate signing requests. The workaround is to generate the keys and certs using an external PKI and then pass those values in as configuration.
- Fully remove mode '0' by 1) restructuring the YANG modules (no impact to API) and 2) in DAL, optimize code bits having mode '0' handling. Mostly to simplify code, but might improve performance.
- Factor out app-layer into its own modular package. –>

5 Known Limitations

- Due to issues with both Python (when SZTPD itself terminates TLS connections) and NGINX (when NGINX terminates TLS connections) limit client certificates to containing just the end-entity certificate (no intermediate certificates). That said, it is unusual for identity certificates (e.g., an IDevID or LDevID certificate) to include any intermediates certificates, so may not be an issue in practice. The Python issue is likely to be resolved in an upcoming Python release. It is unclear when the NGINX issue will be resolved.
- Python (currently) is unable to staple OSCP Responses to the TLS Handshake. Workaround, if needed at all, is to front SZTPD with an external TLS Terminator, which is better for perfomance anyway.
- As of 0.0.7, the wn-sztpd-1 module will NOT validate tenant-views having "device" entries. This is because the "device-type" leafref is now "require-instance true", but the tenant-view is unable to access the "device-type" entries present only at the host-level. Until resolved, clients accessing the tenant-view SHOULD modify the leafref to have "require-instance false", as described by Section 9.9.3 of RFC 7950.
- Currently supports only elliptical keys. This would be a quick fix, but a known limitation until then…

6 Change Log

6.1 0.0.9

- Enabled TLS ports to use RSA-based keys (extended deep-inspection logic)
- Enabled TLS server certs to be unordered inside the CMS structure when configured.
- Logic now removes the <content-data> wrapper from XML-based conveyed-information responses.
- Added support for the 'relay-progress-report' dyanmic-callout. Previously only the webhook was supported.

6.2 0.0.8

- Rewrote the support for "ordered-by user" lists to be more scalable.
- Added initial support for pagination query parameters ('limit', 'offset', and 'direction').
- Added XML support for the SBI (now supports both JSON and XML). Strong HTTP header checking.
- Added strong validation for known base64-encoded values (public keys, private keys, end-entity certificates, and trust anchor certificates) when being configured.

6.3 0.0.7

- Changed "/transport/listen/endpoint/use-for" to be a "mandatory true" *leaf*; it was a "mandatory false" *leaf-list*, thus removing the ability for an endpoint to present more than one API, which was unnecessarily present before.
- The "ordered-by user" query parameters (point + insert) now work, per Section 4.8 of RFC 8040[There are three "ordered-by user" lists in SZTPD: download-uris, bootstrap-servers, and matched-responses (the first two are leaf-lists). advised that the "download-uri" leaf-list uses URL as keys; the client MUST percent-encode these URL-based keys.].
- Modified the "wn-sztpd-1" YANG module to set the per-device *device-type* leafref to "require-instance true" (was false).
- Implemented the "SZTPD_DEFAULT_ADDR" environment variable, as described in the Installation Guide.

6.4 0.0.6

- Device-ownership verification callout now works using plugin-based callouts.
- The validation-layer's cache is now rolled back when database transactions fail.
- The validation-layer now tests for uniqueness in global keys.

6.5 0.0.5

- Now supports fronting SZTPD with a TLS-terminator. Requires that the SZTP-client's certificate is passed to SZTPD via the HTTP header "X-Client-Cert" as a PEM (urlencoded). Tested using NGINX.

6.6 0.0.4

- Now supports concurrent write requests.

6.7 0.0.3

- all unit tests now pass when SZTPD points to a MySQL database. With or without TLS, with or without client certificate. AWS Aurora MySQL also tested.
- RESTCONF error meesages are now returned on the SBI (RFC 8572) interface.

6.8 0.0.2

- callback-based callouts implemented to support draft-kwatsen-netconf-sztp-csr.
- fixed bugs related to bootstrapping-log and audit-log not cleaning up correctly when deleted.

6.9 0.0.1

- initial public release

