SZTPD User's Guide

Watsen Networks

February 15, 2025

Abstract

This documentation is for the SZTPD product by Watsen Networks.

This documentation is still a work-in-progress. Some sections clearly indicate when material is pending, but there are also some missing sections, and other sections may not have enough detail.

SZTPD is currently in its "alpha" stage and details captured in this document may change.

Contents

1 Introduction **3**

2 API Introduction **4**
2.1 Northbound Interface Introduction . . . 4
2.2 Outbound Interface Introduction . . . 4
2.3 Southbound Interface Introduction . . . 4

3 Getting Started **5**
3.1 Standard Startup Interactions . . . 5
3.1.1 Fetching Host-meta . . . 5
3.1.2 Fetching the RESTCONF Root Resource . . . 5
3.1.3 Fetching the YANG Library . . . 5
3.1.4 Get the Default Configuration . . . 5
3.1.5 Configure a User . . . 5
3.2 Prep for Bootstrapping Examples . . . 6
3.2.1 Set Baseline Configuration . . . 6
3.2.2 Give Server time to restart . . . 8
3.2.3 Simulate Device Trying to Get Bootstrapping Data . . . 9
3.3 Testing a Redirect Response . . . 9
3.3.1 Defining a Bootstrap Server . . . 9
3.3.2 Defining a Redirect Response . . . 10
3.3.3 Configuring Device to Return the Redirect Response . . . 11
3.3.4 Simulate Device Getting Redirect Information . . . 11
3.3.5 Delete Redirect Information . . . 12
3.4 Testing an Onboarding Response . . . 13
3.4.1 Defining a Boot Image . . . 13
3.4.2 Defining Pre- and Post- Configuration Scripts . . . 14
3.4.3 Defining a Configuration for Bootstrapping Devices . . . 14
3.4.4 Defining an Onboarding Response . . . 15
3.4.5 Configuring Device to Return the Onboarding Response . . . 16
3.4.6 Simulate Device Getting Onboarding Information . . . 16
3.4.7 Simulate Device Posting a Progress Report . . . 17
3.4.8 Delete Onboarding Information . . . 18
3.5 View Logs . . . 19
3.5.1 View Device's Bootstrapping Log . . . 19
3.5.2 View YANGcore's Audit Log . . . 21

4 Special Topics **24**
4.1 HTTP Headers . . . 24
4.2 Ordered Lists . . . 24
4.2.1 Example: POST-ing a Matched-Response after Another . . . 24
4.2.2 Example: PUT-ing an entry before another entry . . . 24
4.3 Special Characters . . . 25
4.3.1 Example: POST-ing a Download-URI after Another . . . 25
4.3.2 Example: PUT-ing a Download-URI after Another . . . 25
4.4 List Pagination . . . 26
4.5 Event Triggers . . . 26
4.6 Plugins . . . 26

5 Northbound API Details **26**
5.1 NBI YANG Library . . . 26
5.2 NBI YANG Modules . . . 28
5.3 NBI Complete Tree Diagram . . . 29
5.4 NBI Top-level Nodes . . . 33
5.4.1 /ietf-yang-library:yang-library . . . 33
5.4.2 /sztpd:conveyed-information . . . 33
5.4.3 /sztpd:device-types . . . 35

5.4.4 /sztpd:devices . . . 35
5.4.5 /sztpd:responses . . . 37

6 Outbound API Details 38
6.1 Dynamic Callouts . . . 38
6.1.1 The "get-conveyed-information" Dynamic Callout . . . 38
6.1.2 The "relay-progress-report" Dynamic Callout . . . 39
6.1.3 The "verify-device-ownership" Dynamic Callout . . . 41
6.2 Notifications . . . 42

7 Southbound API Details 43
7.1 SBI YANG Library . . . 43
7.2 SBI YANG Modules . . . 44
7.3 SBI Complete Tree Diagram . . . 45
7.4 SBI Top-level Nodes . . . 46
7.4.1 /ietf-sztp-bootstrap-server:get-bootstrapping-data . . . 46
7.4.2 /ietf-sztp-bootstrap-server:report-progress . . . 46
7.4.3 /ietf-yang-library:yang-library . . . 46
7.5 Determining the SBI's Host-meta . . . 46
7.6 Determining the SBI's RESTCONF Root Resource . . . 46
7.7 Determining the SBI's Supported Encodings . . . 46
7.8 Determining the SBI's YANG-library Version . . . 46

8 Simulator 48
8.1 Overview . . . 48
8.2 Dependencies . . . 49
8.3 Downloading . . . 49
8.4 Unarchiving . . . 49
8.5 Customizing Variables . . . 50
8.6 Initializing the PKI . . . 50
8.7 Running . . . 50
8.8 Cleaning Up . . . 51

1 Introduction

SZTPD implements the "bootstrap server" defined in RFC 8572[1]. SZTPD also implements the "SZTP server" defined in RFC 9646".

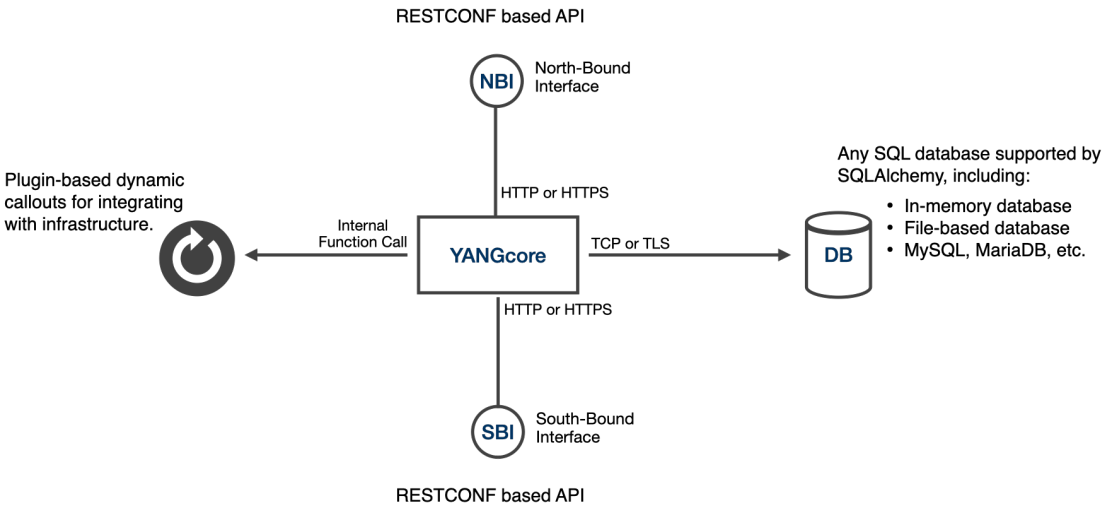
SZTPD is provided as Python software, an asynchronous event-driven executable. SZTPD is an application (not a library like YANGcore) that includes a northbound API for configuration, a southbound API for device bootstrapping requests, and outbound hooks for integration with other business systems.

This documentation is in preparation for a "1.0.0" release, using the common "major.minor.patch" semantic versioning convention.

[1]The implementation is incomplete, e.g., the implementation does not yet support Ownership Vouchers.

2 API Introduction

The following picture illustrates the API interfaces implemented by SZTPD.



The various aspects of this picture are discussed in the following sections.

2.1 Northbound Interface Introduction

SZTPD's Northbound interface (NBI) is the same as YANGcore's, with the only difference being that it adds configuration and operational state nodes to the data tree. Please see the "Northbound Interface Introduction" section in the YANGcore User Guide for details about the YANGcore NBI.

See the Northbound API Details section in this guide for a description of the new nodes that SZTPD added to the NBI.

2.2 Outbound Interface Introduction

SZTPD's Outbound interface is the same as YANGcore's, with the only difference being that it adds additional dynamic callouts. Please see the "Outbound Interface Introduction" in the YANGcore User Guide for details about the YANGcore outbound interface..

See the Outbound API Details section in this guide for a description of the additions made by SZTPD.

2.3 Southbound Interface Introduction

SZTPD's Southbound interface (SBI) is added, it does not come with YANGcore. The Southbound interface implements:

- The "Bootstrap Server" defined in Section 7 of RFC 8572.
- The "SZTP Server" The

The SBI is a RESTCONF RFC 8040 based API that implements the "ietf-sztp-bootstrap-server" YANG module. See Section 7 of RFC 8572 for details.

See the Southbound API Details section in this guide for more information about the southbound interface.

3 Getting Started

This section presents a few examples illustrating some "startup" interactions with SZTPD. These examples assume SZTPD's NBI is listening on its default address and port: 127:0.0.1:8080. For more information about SZTPD's defaults, see the [Factory Defaults] section in the SZTPD Installation Guide.

To run a fresh instance of SZTPD, in one window, run the following command:

```
$ sztpd sqlite:///:memory:
```

After answering "Yes" to accept the "Non-production Use" contract, there will be no more output to the screen. Press ^C, or send any signal other than SIGHUP to the process, to cause SZTPD to gracefully exit.

These examples use the ubiquitous 'curl' command line utility program for illustration purposes only. It is expected that production code would use a programming language to achieve the same result.

The following subsections illustrate commands run from another window.

```
Warning! All of the following commands and responses were excuted/captured for this build of the
documentation using the current SZTPD server.  Copy/pasting is expected, but please be advised
that, at least in the PDF version of the documentation, the hyphen and the single— and double—
quote characters are incorrectly converted from plain ASCII to their "fancy" UTF-8 equivalents.
They must be converted back to their ASCII forms for these SHELL scripts to run.
```

3.1 Standard Startup Interactions

The following subsections describe interactions that are somewhat typical for RESTCONF clients.

3.1.1 Fetching Host-meta

SZTPD inherits how to fetch host-meta from YANGcore. Please see the "Fetching Host-meta" section in the YANGcore User Guide for details.

SZTPD's host-meta is the same as YANGcore's host-meta.

3.1.2 Fetching the RESTCONF Root Resource

SZTPD inherits how to fetch the RESTCONF root resource from YANGcore. Please see the "Fetching the RESTCONF Root Resource" section in the YANGcore User Guide for details.

SZTPD's Root Resource is the same as YANGcore's Root Resource (i.e., "/restconf").

3.1.3 Fetching the YANG Library

SZTPD inherits how to fetch the YANG Library from YANGcore. Please see the "Fetching the YANG Library" section in the YANGcore User Guide for details.

SZTPD's YANG Library is a superset of the YANGcore YANG Library. A full listing of the SZTPD's YANG Library is in NBI YANG Library.

3.1.4 Get the Default Configuration

SZTPD inherits how to get the default configuration from YANGcore.
Please see the "Get the Default Configuration" section in the YANGcore User Guide for details.

SZTPD's default configuration is the same as YANGcore's default configuration.

3.1.5 Configure a User

SZTPD inherits how to configure users from YANGcore. Please see the "Configure a User" section in the YANGcore User Guide for details.

SZTPD does not modify any aspect of YANGcore's authentication or authorization mechanisms.

3.2 Prep for Bootstrapping Examples

3.2.1 Set Baseline Configuration

As is described in the YANGcore User's Guide, using individual commands, such as described above, can be tedious. In general, RESTCONF allows any node in the data tree to be replaced, i.e., with a PUT command. This ability applies to the root '/' node as well and, in so using, enables the whole configuration to be set in a single command.

This example illustrate the whole configuration being set using a single PUT command. The goal of this configuration is to quickly set a baseline for subsequent examples. This baseline:

- configures TLS on both the NBI and SBI, using keys placed in the Keystore.
- puts a trust anchor into the Truststore to validate bootstrapping devices.
- defines a device type that defines how to validate bootstrapping device certificates.
- defines a device for the aforementioned device-type and defines an authentication code.

Be mindful that, s described in the YANGcore User Guide, because this request modifies the server's interfaces, the server will be momentarily unavailable while it restarts.

For running code, please see the Simulator code.

This section uses scripts to generate its contents. These scripts use a PKI that has been instantiated as "pki" in the current directory. This 'pki' directory is the same as that in the Simulator code. These scripts could be run out of that directory, after creating an empty directory called "output".

Request:

```
================================ NOTE: '\' line wrapping per RFC 8792 ================================

#!/bin/bash

TEMPDIR=`mktemp -d`

# NBI Port
NBI_PORT=8443
NBI_PRI_KEY_B64=`openssl enc -base64 -A -in pki/sztpd1/nbi/end-entity/private_key.der`
NBI_PUB_KEY_B64=`openssl enc -base64 -A -in pki/sztpd1/nbi/end-entity/public_key.der`
cat pki/sztpd1/nbi/end-entity/my_cert.pem pki/sztpd1/nbi/intermediate2/my_cert.pem > $TEMPDIR/cert_chain.pem
openssl crl2pkcs7 -nocrl -certfile $TEMPDIR/cert_chain.pem -outform DER -out $TEMPDIR/cert_chain.cms
NBI_EE_CERT_B64=`openssl enc -base64 -A -in $TEMPDIR/cert_chain.cms`

# SBI Port
SBI_PORT=9443
SBI_PRI_KEY_B64=`openssl enc -base64 -A -in pki/sztpd1/sbi/end-entity/private_key.der`
SBI_PUB_KEY_B64=`openssl enc -base64 -A -in pki/sztpd1/sbi/end-entity/public_key.der`
cat pki/sztpd1/sbi/end-entity/my_cert.pem pki/sztpd1/sbi/intermediate2/my_cert.pem > $TEMPDIR/cert_chain.pem
openssl crl2pkcs7 -nocrl -certfile $TEMPDIR/cert_chain.pem -outform DER -out $TEMPDIR/cert_chain.cms
SBI_EE_CERT_B64=`openssl enc -base64 -A -in $TEMPDIR/cert_chain.cms`

# client cert (DevID) trust anchor
cat pki/client/root-ca/my_cert.pem pki/client/intermediate1/my_cert.pem pki/client/intermediate2/my_cert.pe\
m > $TEMPDIR/ta_cert_chain.pem
openssl crl2pkcs7 -nocrl -certfile $TEMPDIR/ta_cert_chain.pem -outform DER -out $TEMPDIR/ta_cert_chain.cms
CLIENT_CERT_TA_B64=`openssl enc -base64 -A -in $TEMPDIR/ta_cert_chain.cms`

# initialize body for the PUT request
cat << EOM > $TEMPDIR/running.json
{
    "ietf-restconf-server:restconf-server": {
        "listen": {
            "endpoints": {
                "endpoint": [
                    {
                        "name": "native-interface",
                        "yangcore:use-for": "yangcore:native-interface",
                        "http-over-tls": {
                            "tcp-server-parameters": {
                                "local-bind": [
                                    {
                                        "local-address": "127.0.0.1",
                                        "local-port": $NBI_PORT
                                    }
                                ]
                            },
                            "tls-server-parameters": {
                                "server-identity": {
                                    "certificate" : {
                                        "central-keystore-reference": {
                                            "asymmetric-key": "nbi-server-end-entity-key",
```



```
                                        "certificate": "nbi-server-end-entity-cert"
                                    }
                                }
                            }
                        }
                    }
                },
                {
                    "name": "rfc8572-interface",
                    "yangcore:use-for": "sztpd:rfc8572-interface",
                    "http-over-tls": {
                        "tcp-server-parameters": {
                            "local-bind": [
                                {
                                    "local-address": "127.0.0.1",
                                    "local-port": $SBI_PORT
                                }
                            ]
                        },
                        "tls-server-parameters": {
                            "server-identity": {
                                "certificate" : {
                                    "central-keystore-reference": {
                                        "asymmetric-key": "sbi-server-end-entity-key",
                                        "certificate": "sbi-server-end-entity-cert"
                                    }
                                }
                            },
                            "client-authentication": {
                              "ca-certs": {
                                "central-truststore-reference": "my-device-identity-ca-certs"
                              }
                            }
                        }
                    }
                }
            ]
        }
    }
},
"yangcore:preferences": {
    "authentication": {
        "internal-authenticator": {
            "passwords-allowed": {}
        }
    }
},
"yangcore:users": {
    "user": [
        {
            "login": "my-admin",
            "email-address": "my-admin@example.com",
            "authentication": {
                "password-based": {
                    "password": "\$0\$my-secret"
                }
            },
            "authorization": {
                "unrestricted": [null]
            }
        }
    ]
},
"ietf-keystore:keystore": {
    "asymmetric-keys": {
        "asymmetric-key": [
            {
                "name": "nbi-server-end-entity-key",
                "public-key-format": "ietf-crypto-types:subject-public-key-info-format",
                "public-key": "$NBI_PUB_KEY_B64",
                "private-key-format": "ietf-crypto-types:ec-private-key-format",
                "cleartext-private-key": "$NBI_PRI_KEY_B64",
                "certificates": {
                    "certificate": [
                        {
                            "name": "nbi-server-end-entity-cert",
                            "cert-data": "$NBI_EE_CERT_B64"
                        }
                    ]
                }
            },
            {
                "name": "sbi-server-end-entity-key",
                "public-key-format": "ietf-crypto-types:subject-public-key-info-format",
                "public-key": "$SBI_PUB_KEY_B64",
                "private-key-format": "ietf-crypto-types:ec-private-key-format",
                "cleartext-private-key": "$SBI_PRI_KEY_B64",
```



```
                    "certificates": {
                        "certificate": [
                            {
                                "name": "sbi-server-end-entity-cert",
                                "cert-data": "$SBI_EE_CERT_B64"
                            }
                        ]
                    }
                }
            ]
        }
    },
    "ietf-truststore:truststore": {
        "certificate-bags": {
            "certificate-bag": [
                {
                    "name": "my-device-identity-ca-certs",
                    "description": "A set of TA certs used to authenticate device client certs.",
                    "certificate": [
                        {
                            "name": "my-device-identity-ca-cert-circa-2020",
                            "cert-data": "$CLIENT_CERT_TA_B64"
                        }
                    ]
                }
            ]
        }
    },
    "sztpd:device-types": {
        "device-type": [
            {
                "name": "my-device-type",
                "identity-certificates": {
                    "verification": {
                        "central-truststore-reference": {
                            "certificate-bag": "my-device-identity-ca-certs",
                            "certificate": "my-device-identity-ca-cert-circa-2020"
                        }
                    },
                    "serial-number-extraction": "wn-x509-c2n:serial-number"
                }
            }
        ]
    },
    "sztpd:devices": {
        "device": [
            {
                "serial-number": "my-serial-number",
                "device-type": "my-device-type",
                "activation-code": "\$0\$my-secret"
            }
        ]
    },
    "sztpd:conveyed-information": {
    }
}
EOM

# PUT running
curl -isf -X PUT --data @$TEMPDIR/running.json -H "Content-Type:application/yang-data+json" http://127.0.0.\
1:8080/restconf/ds/ietf-datastores:running > output/put_whole_config.out
if [[ $? != 0 ]]; then
  echo "curl returned an error"
  exit 1
fi

# remove temp dir
rm -rf "$TEMPDIR"
```

Response:

```
HTTP/1.1 204 No Content
Date: Sun, 16 Feb 2025 00:23:54 GMT
Server: <redacted>
```

3.2.2 Give Server time to restart

Since this change modifies the "/restconf-server" tree, the SZTPD instance sends a SIGHUP to itself, thus causing it to reload the configuration and re-open listening ports per configuration.

SZTPD takes a few seconds to reload with a relatively empty configuration. Once sufficient time has elapsed (a few seconds), the configured ports will be available.

3.2.3 Simulate Device Trying to Get Bootstrapping Data

This example illustrates the response given to a bootstraping device before it has been configured to return a specific kind of response.

This command fails because no "responses" have been configured for the device on the server. Note that error code 404 is used to indicate that the device may try again.

One thing new/interesting about this example is that it uses XML. The SBI supports both JSON and XML.

Request:

```
================================== NOTE: '\' line wrapping per RFC 8792 ================================

#!/bin/sh

# get a temp dir
TEMPDIR=`mktemp -d`

# initialize the 'input' node for the RPC
cat << EOM > $TEMPDIR/input.xml
<input xmlns="urn:ietf:params:xml:ns:yang:ietf-sztp-bootstrap-server">
    <hw-model>model-x</hw-model>
    <os-name>vendor-os</os-name>
    <os-version>17.3R2.1</os-version>
</input>
EOM

# POST 'input' for the "get-bootstrapping-data" RPC resource
curl -isf -X POST --data @$TEMPDIR/input.xml -H "Content-Type:application/yang-data+xml" -H "Accept:applica\
tion/yang-data+xml" --cacert sbi_trust_chain.pem --key pki/client/end-entity/private_key.pem --cert pki/cli\
ent/end-entity/my_cert.pem --user my-serial-number:my-secret https://127.0.0.1:9443/restconf/operations/iet\
f-sztp-bootstrap-server:get-bootstrapping-data > output/rpc_get_bootstrapping_data_1.out

if [[ $? == 0 ]]; then
  echo "curl succeeded?!"
  exit 1
fi

# line-return needed for markdown
echo "" >> output/rpc_get_bootstrapping_data_1.out

# remove temp dir
rm -rf $TEMPDIR
```

Response:

```
HTTP/1.1 404 Not Found
Content-Type: application/yang-data+xml; charset=utf-8
Content-Length: 359
Date: Sun, 16 Feb 2025 00:23:59 GMT
Server: <redacted>
```

3.3 Testing a Redirect Response

To configure a "redirect" response, first configure the '/bootstrapping-servers' node to define a bootstrap-server that a bootstrapping device is to be redirected too. The configured 'bootstrap-server' definition is referenced in the subsequent steps. Note that the 'trust-anchor' node's value is replaced by a variable. While this example shows just one bootstrap server being configured, any number of bootstrap servers may be configured, either in this step on anytime later.

3.3.1 Defining a Bootstrap Server

Request:

```
================================== NOTE: '\' line wrapping per RFC 8792 ================================

#!/bin/sh

# get a temp dir
TEMPDIR=`mktemp -d`

# generate values for placeholders
cat pki/sztpd2/sbi/root-ca/my_cert.pem pki/sztpd2/sbi/intermediate1/my_cert.pem > cert_chain.pem
openssl crl2pkcs7 -nocrl -certfile cert_chain.pem -outform DER -out cert_chain.cms
BOOTSVR_TA_CERT_B64=`openssl enc -base64 -A -in  cert_chain.cms`
```



```
# initialize body for the POST request
cat << EOM > $TEMPDIR/bootstrap-servers.json
{
    "sztpd:bootstrap-servers": {
        "bootstrap-server": [
            {
                "name": "my-bootstrap-server",
                "address": "127.0.0.1",
                "port": 9443,
                "trust-anchor": "$BOOTSVR_TA_CERT_B64"
            }
        ]
    }
}
EOM

# POST 'bootstrap-servers' to running
curl -isf -X POST --data @$TEMPDIR/bootstrap-servers.json -H "Content-Type:application/yang-data+json" --ca\
cert nbi_trust_chain.pem --user my-admin:my-secret https://127.0.0.1:8443/restconf/ds/ietf-datastores:runni\
ng/sztpd:conveyed-information > output/post_bootstrap_servers.out
if [[ $? != 0 ]]; then
  echo "curl returned an error"
  exit 1
fi

# remove temp dir
rm -rf $TEMPDIR
```

Response:

```
HTTP/1.1 201 Created
Content-Length: 0
Date: Sun, 16 Feb 2025 00:23:59 GMT
Server: <redacted>
```

3.3.2 Defining a Redirect Response

Configure the '/responses' node. Configure a 'redirect-response' indicating that the device is to be redirected to 'my-bootstrap-server'. Whilst this example shows just one bootstrap-server listed, an ordered list of bootstrap servers may be configured.

Request:

```
================================== NOTE: '\' line wrapping per RFC 8792 ================================

#!/bin/sh

# get a temp dir
TEMPDIR=`mktemp -d`

# initialize body for the POST request
cat << EOM > $TEMPDIR/responses.json
{
    "sztpd:responses": {
        "redirect-response": [
            {
                "name": "my-redirect-information",
                "redirect-information": {
                    "bootstrap-server": [
                        "my-bootstrap-server"
                    ]
                }
            }
        ]
    }
}
EOM

# POST 'responses' to running
curl -isf -X POST --data @$TEMPDIR/responses.json -H "Content-Type:application/yang-data+json" --cacert nbi\
_trust_chain.pem --user my-admin:my-secret https://127.0.0.1:8443/restconf/ds/ietf-datastores:running > out\
put/post_responses.out
if [[ $? != 0 ]]; then
  echo "curl returned an error"
  exit 1
fi

# remove temp dir
rm -rf $TEMPDIR
```


Response:

```
HTTP/1.1 201 Created
Content-Length: 0
Date: Sun, 16 Feb 2025 00:23:59 GMT
Server: <redacted>
```

3.3.3 Configuring Device to Return the Redirect Response

Configure the 'response-manager' node inside the 'my-serial-number' device object to return the "my-redirect-information" respsonse. Note that a catch-all rule (i.e., one without any 'match-criteria' defined) tells STZPD to always return this response to this device.

Request:

```
================================== NOTE: '\' line wrapping per RFC 8792 ================================

#!/bin/sh

# get a temp dir
TEMPDIR=`mktemp -d`

# initialize body for the POST request
cat << EOM > $TEMPDIR/response-manager.json
{
    "sztpd:response-manager": {
        "matched-response": [
            {
                "name": "catch-all-response",
                "response": {
                    "conveyed-information": {
                        "via-redirect-response": {
                            "reference": "my-redirect-information"
                        }
                    }
                }
            }
        ]
    }
}
EOM

# POST 'response-manager' to *device*
curl -isf -X POST --data @$TEMPDIR/response-manager.json -H "Content-Type:application/yang-data+json" --cac\
ert nbi_trust_chain.pem --user my-admin:my-secret https://127.0.0.1:8443/restconf/ds/ietf-datastores:runnin\
g/sztpd:devices/device=my-serial-number > output/post_response_manager.out
if [[ $? != 0 ]]; then
  echo "curl returned an error"
  exit 1
fi

# remove temp dir
rm -rf $TEMPDIR
```

Response:

```
HTTP/1.1 201 Created
Content-Length: 0
Date: Sun, 16 Feb 2025 00:23:59 GMT
Server: <redacted>
```

3.3.4 Simulate Device Getting Redirect Information

Accessing the 'get-bootstrapping-data' RPC on the SBI again, now it works. Note that the SBI supports both JSON and XML.

Request:

```
================================== NOTE: '\' line wrapping per RFC 8792 ================================

#!/bin/sh

# get a temp dir
TEMPDIR=`mktemp -d`

# initialize the 'input' node for the RPC
cat << EOM > $TEMPDIR/input.xml
<input xmlns="urn:ietf:params:xml:ns:yang:ietf-sztp-bootstrap-server">
    <hw-model>model-x</hw-model>
```



```
    <os-name>vendor-os</os-name>
    <os-version>17.3R2.1</os-version>
</input>
EOM

# POST 'input' for the "get-bootstrapping-data" RPC resource
curl -isf -X POST --data @$TEMPDIR/input.xml -H "Content-Type:application/yang-data+xml" -H "Accept:applica\
tion/yang-data+xml" --cacert sbi_trust_chain.pem --key pki/client/end-entity/private_key.pem --cert pki/cli\
ent/end-entity/my_cert.pem --user my-serial-number:my-secret https://127.0.0.1:9443/restconf/operations/iet\
f-sztp-bootstrap-server:get-bootstrapping-data > output/rpc_get_bootstrapping_data_2.out

if [[ $? != 0 ]]; then
  echo "curl returned an error"
  exit 1
fi

# line-return needed for markdown
echo "" >> output/rpc_get_bootstrapping_data_2.out

# remove temp dir
rm -rf $TEMPDIR
```

Response:

```
================================== NOTE: '\' line wrapping per RFC 8792 ================================

HTTP/1.1 200 OK
Content-Type: application/yang-data+xml; charset=utf-8
Content-Length: 2845
Date: Sun, 16 Feb 2025 00:23:59 GMT
Server: <redacted>

<?xml version="1.0" ?>
<output xmlns="urn:ietf:params:xml:ns:yang:ietf-sztp-bootstrap-server">
  <conveyed-information>MIIH3wYLKoZIhvcNAQkQASqgggfOBIIHyjw/eG1sIHZlcnNpb249IjEuMCIgPz4KPHJlZGlyZWN0LWluZm9\
ybWF0aW9uIHhtbG5zPSJ1cm46aWV0ZjpwYXJhbXM6eG1sOm5zOnlhbmc6aWV0Zi1zenRwLWNvbnZleWVkLWluZm8iPgogIDxib290c3RyYX\
Atc2VydmVyPgogICAgPGFkZHJlc3M+TI3LjAuMC4xPC9hZGRyZXNzPgogICAgPHBvcnQ+OTQ0MzwvcG9ydD4KICAgIDx0cnVzdC1hbmNob\
3I+TUlJRkR3WUpLb1pJaHZjTkFRY0NvSUlGQURDQ0JQd0NBUUV4QURBTEJna3Foa2lHOXcwQkJ3R2dnZ1RrTUlJQ1dqQ0NBZitnQXdJQkFn\
SUJBVEFLQmdncWhrak9QUVFEQWpCMU1Rc3dDUVlEVlFRR0V3SllXREVkTUJzR0ExVUVDQXdVVFhrZ1UzUmhkR1VnYjNJZ1VISnZkbWx1WTJJ\
VeEdEQVdCZ05WQkFvTUQwMTVJRTI5WjJGdWFYcGhkR2x2YmpFUU1BNEdBMVVFQXd3SFRYa2dWVzVwZERFYk1Ca0dBMVVFQXd3U2MySnBMM0\
5sY25abGNpOXliMjkwTFdOaE1DQVhEVEkxTURJeE5qQXdNak0wT0ZvWUR6azVPVGt4TWpNeE1qTTFPVFU1V2pCMU1Rc3dDUVlEVlFRR0V3S\
lIXREVkTUJzR0ExVUVDQXdVVFhrZ1UzUmhkR1VnYjNJZ1VISnZkbWx1WTJVeEdEQVdCZ05WQkFvTUQwMTVJRTI5WjJGdWFYcGhkR2x2YmpF\
UU1BNEdBMVVFQ3d3SFRYa2dWVzVwZERFYk1Ca0dBMVVFQXd3U2MySnBMM05sY25abGNpOXliMjkwTFdOaE1Ga3dFd1lIS29aSXpqMENBUVl\
JS29aSXpqMERBUWNEUWdBRWlTVWFFUnF2bmcrZFh1VEtzNTNaVlNkN283VVpJajhvU2YyeFVzTWtMbzMvTlQ1OUI3bkNBWit2UnREK2w3aj\
BGVjlSMWZJTysrVVlTNXZsRFZMWno2TitNSHd3SFFZRFZSME9CQllFRkx6eWV3WTFyckZLajJ2YXFKMDM1cHFsSFN0bk1Bd0dBMVVkRXdRR\
k1BTUJBZjh3RGdZRFZSMFBBUUgvQkFRREFnRUdNRDBHQTFVZEh3UTJNRFF3TXFBd29DNkdMR2gwZEhBNkx5OWpjbXd1WlhoaGJYQnNaUzVq\
YjIwL1kyRTljMkpwT25ObGNuWmxjanB5YjI5MExXTmhNQW9HQ0NxR1NNNDlCQU1DQTBrQU1FWUNJUUMzS2ZIOXZBME5aeE9TeGExRVhTVmg\
5WWtRc0NxNXB0K1lSekh4UXMvcGl3SWhBTTJSQVAyZmxpOVF0cnVZb1JmRWtIWEUrL25wRUo5Uloxckxi VW5KK2tQOU1JSUNnakNDQWlpZ0\
F3SUJBZ0lCQWpBS0JnZ3Foa2pPUFFRREFqQjFNUXN3Q1FZRFZRUUdFd0pZV0RFZE1Cc0dBMVVFQ0F3VVRYa2dVM1JoZEdVZ2IzSWdVSEp2ZG\
G1sdVkyVXhHREFXQmdOVkJBb01EMDE1SUU5eVoyRnVhWHBoZEdsdmJqRVFNQTRHQTFVRUN3d0hUWGtnVlc1cGRERWJNQmtHQTFVRUF3d1Nj\
MkpwTDNObGNuWmxjaT15YjI5MExXTmhNQ0FYRFRJMU1ESXhOakF3TWpNME9Gb1lEems1T1RreE1qTXhNak0xT1RVNVdqQjdNUXN3Q1FZRFZ\
RUUdFd0pZV0RFZE1Cc0dBMVVFQ0F3VVRYa2dVM1JoZEdVZ2IzSWdVSEp2ZG1sdVkyVXhHREFXQmdOVkJBb01EMDE1SUU5eVoyRnVhWHBoZE\
dsdmJqRVFNQTRHQTFVRUN3d0hUWGtnVlc1cGRERWhNQjhHQTFVRUF3d1ljMkpwTDNObGNuWmxjaTlwYm5SbGNtMWxaR2xoZEdVeE1Ga3dFd\
1lIS29aSXpqMENBUVlJS29aSXpqMERBUWNEUWdBRWVlSU9FMk5BSjdZTkhveFphQ3ovaHhOMkxWWWRoa3ZDSkJOSlNPUVdKM3FsUzJjN2dD\
eWpxVm9HQ01qUlpsdlRCNG13VWQxWUZvR2czR0xKMEVseUFhT0JvRENCblRBZEJnTlZIUTRFRmdRVXltMTEraXdyYjQ3NnIrWEwwa0gvSEV\
jaU5QVXdId1lEVlIwakJCZ3dGb0FVdlBKN0JqV3VzVXFQYTkb25UZm1tcVVkSzJjd0RBWURWUjBUQkFVd0F3RUIvekFPQmdOVkhROEJBZj\
hFQkFNQ0FRWXdQUVlEVlIwZkJEWXdOREF5b0RDZ0xvWXNhSFIwY0Rvdkwy TnliQzVsZUdGdGNHeGxMbU52YlQ5allUMXpZbWs2YzJWeWRtV\
nlPbkp2YjNRdFkyRXdDZ1lJS29aSXpqMEVBd0lEU0FBd1JRSWhBTURKSU83YjRjcXhvdGMwREUrdHZYbFVWdUFSSk5NVURnSlMxMzlzaWFG\
RUFpQXp5Rlowa0ljNHRVYndnWElCcHpSS2J3a01KU0NmcGtOS0pySnFOZzVBR2pFQTwvdHJ1c3QtYW5jaG9yPgogIDwvYm9vdHN0cmFwLXN\
lcnZlcj4KPC9yZWRpcmVjdC1pbmZvcm1hdGlvbj4K</conveyed-information>
</output>
```

3.3.5 Delete Redirect Information

Now delete the configuration from the previous three steps to get back to clean baseline, in preparation for the upcoming [Configuring an Onboarding Response] section.

Note that the configuration is removed in the reverse order to avoid SZTPD throwing a validation error due to dangling references.

Request:

```
================================== NOTE: '\' line wrapping per RFC 8792 ================================

#!/bin/sh

# These are removed in opposite order to avoid dangling reference validation errors.

curl -isf -X DELETE --cacert nbi_trust_chain.pem --user my-admin:my-secret https://127.0.0.1:8443/restconf/\
ds/ietf-datastores:running/sztpd:devices/device=my-serial-number/response-manager > output/delete_redirect_\
info.out
if [[ $? != 0 ]]; then
```



```
  echo "curl returned an error"
  exit 1
fi

curl -isf -X DELETE --cacert nbi_trust_chain.pem --user my-admin:my-secret https://127.0.0.1:8443/restconf/\
ds/ietf-datastores:running/sztpd:responses >> output/delete_redirect_info.out
if [[ $? != 0 ]]; then
  echo "curl returned an error"
  exit 1
fi

curl -isf -X DELETE --cacert nbi_trust_chain.pem --user my-admin:my-secret https://127.0.0.1:8443/restconf/\
ds/ietf-datastores:running/sztpd:conveyed-information/bootstrap-servers >> output/delete_redirect_info.out
if [[ $? != 0 ]]; then
  echo "curl returned an error"
  exit 1
fi
```

Response:

```
HTTP/1.1 204 No Content
Date: Sun, 16 Feb 2025 00:23:59 GMT
Server: <redacted>

HTTP/1.1 204 No Content
Date: Sun, 16 Feb 2025 00:23:59 GMT
Server: <redacted>

HTTP/1.1 204 No Content
Date: Sun, 16 Feb 2025 00:23:59 GMT
Server: <redacted>
```

3.4 Testing an Onboarding Response

3.4.1 Defining a Boot Image

First configure information about a hypothetical "boot-image.img" the device should be running:

Request:

```
================================ NOTE: '\' line wrapping per RFC 8792 ================================

#!/bin/sh

# get a temp dir
TEMPDIR=`mktemp -d`

# initialize body for the POST request
dd if=/dev/urandom of=$TEMPDIR/my-boot-image.img bs=64k count=1 >> /dev/null 2>&1
BOOT_IMG_HASH_VAL=`openssl dgst -sha256 -c $TEMPDIR/my-boot-image.img | awk '{print $2}'`

cat << EOM > $TEMPDIR/boot-images.json
{
    "sztpd:boot-images": {
        "boot-image": [
            {
                "name": "my-boot-image.img",
                "os-name": "FooBarOS",
                "os-version": "1.2.3",
                "download-uri": [ "https://example.com/my-boot-image.img" ],
                "image-verification": [
                    {
                        "hash-algorithm": "ietf-sztp-conveyed-info:sha-256",
                        "hash-value": "$BOOT_IMG_HASH_VAL"
                    }
                ]
            }
        ]
    }
}
EOM

# POST 'boot-images' to running
curl -isf -X POST --data @$TEMPDIR/boot-images.json -H "Content-Type:application/yang-data+json" --cacert n\
bi_trust_chain.pem --user my-admin:my-secret https://127.0.0.1:8443/restconf/ds/ietf-datastores:running/szt\
pd:conveyed-information > output/post_boot_images.out
if [[ $? != 0 ]]; then
  echo "curl returned an error"
  exit 1
fi

# remove temp dir
rm -rf $TEMPDIR
```


Response:

```
HTTP/1.1 201 Created
Content-Length: 0
Date: Sun, 16 Feb 2025 00:23:59 GMT
Server: <redacted>
```

3.4.2 Defining Pre- and Post- Configuration Scripts

Configure information about the "pre" and "post" scripts the device should run. This example shows the scripts are 'bash' scripts, but the scripts may be any executable supported by the bootstrapping device.

Request:

```
================================== NOTE: '\' line wrapping per RFC 8792 ================================

#!/bin/sh

# get a temp dir
TEMPDIR=`mktemp -d`

# encode a pre-configuration script
cat << EOM > $TEMPDIR/my-pre-configuration-script
#/bin/bash
echo "inside the pre-configuration-script..."
EOM
PRE_SCRIPT_B64=`openssl enc -base64 -A -in $TEMPDIR/my-pre-configuration-script`

# encode a post-configuration script
cat << EOM > $TEMPDIR/my-post-configuration-script
#/bin/bash
echo "inside the post-configuration-script..."
EOM
POST_SCRIPT_B64=`openssl enc -base64 -A -in $TEMPDIR/my-post-configuration-script`

# initialize body for the POST request
cat << EOM > $TEMPDIR/scripts.json
{
    "sztpd:scripts": {
        "script": [
            {
                "name": "my-pre-configuration-script",
                "type": "PRE-CONFIG",
                "code": "$PRE_SCRIPT_B64"
            },
            {
                "name": "my-post-configuration-script",
                "type": "POST-CONFIG",
                "code": "$POST_SCRIPT_B64"
            }
        ]
    }
}
EOM

# POST 'scripts' to running
curl -isf -X POST --data @$TEMPDIR/scripts.json -H "Content-Type:application/yang-data+json" --cacert nbi_t\
rust_chain.pem --user my-admin:my-secret https://127.0.0.1:8443/restconf/ds/ietf-datastores:running/sztpd:c\
onveyed-information > output/post_scripts.out
if [[ $? != 0 ]]; then
  echo "curl returned an error"
  exit 1
fi

# remove temp dir
rm -rf $TEMPDIR
```

Response:

```
HTTP/1.1 201 Created
Content-Length: 0
Date: Sun, 16 Feb 2025 00:23:59 GMT
Server: <redacted>
```

3.4.3 Defining a Configuration for Bootstrapping Devices

Set the device-specific configuration the device should run. The example configuration is nonsensical, but note that it is encoded in XML because, presumably, this device's native configuration is in XML. The configuration may be encoded in any format knwon to the device.

Request:

```
================================== NOTE: '\' line wrapping per RFC 8792 ================================

#!/bin/sh

# get a temp dir
TEMPDIR=`mktemp -d`

# initialize body for the POST request
cat << EOM > $TEMPDIR/my-configuration
<top xmlns="https://example.com/config">
  <any-xml-content-okay/>
</top>
EOM
CONFIG_B64=`openssl enc -base64 -A -in $TEMPDIR/my-configuration`

cat << EOM > $TEMPDIR/configurations.json
{
    "sztpd:configurations": {
        "configuration": [
            {
                "name": "my-configuration",
                "handling": "merge",
                "config-data": "$CONFIG_B64"
            }
        ]
    }
}
EOM

# POST 'configurations.json' to running
curl -i -X POST --data @$TEMPDIR/configurations.json -H "Content-Type:application/yang-data+json" --cacert \
nbi_trust_chain.pem --user my-admin:my-secret https://127.0.0.1:8443/restconf/ds/ietf-datastores:running/sz\
tpd:conveyed-information > output/post_configurations.out
if [[ $? != 0 ]]; then
  echo "curl returned an error"
  exit 1
fi

# remove temp dir
rm -rf $TEMPDIR
```

Response:

```
HTTP/1.1 201 Created
Content-Length: 0
Date: Sun, 16 Feb 2025 00:23:59 GMT
Server: <redacted>
```

3.4.4 Defining an Onboarding Response

Configure an "onboarding-response" referencing all of the above:

Request:

```
================================== NOTE: '\' line wrapping per RFC 8792 ================================

#!/bin/sh

# get a temp dir
TEMPDIR=`mktemp -d`

# initialize body for the POST request
cat << EOM > $TEMPDIR/responses.json
{
    "sztpd:responses": {
        "onboarding-response": [
            {
                "name": "my-onboarding-information",
                "onboarding-information": {
                    "boot-image": "my-boot-image.img",
                    "pre-configuration-script": "my-pre-configuration-script",
                    "configuration": "my-configuration",
                    "post-configuration-script": "my-post-configuration-script"
                }
            }
        ]
    }
}
EOM

# POST 'responses' to running
```



```
curl -isf -X POST --data @$TEMPDIR/responses.json -H "Content-Type:application/yang-data+json" --cacert nbi\
_trust_chain.pem --user my-admin:my-secret https://127.0.0.1:8443/restconf/ds/ietf-datastores:running > out\
put/post_responses_2.out
if [[ $? != 0 ]]; then
  echo "curl returned an error"
  exit 1
fi

# remove temp dir
rm -rf $TEMPDIR
```

Response:

```
HTTP/1.1 201 Created
Content-Length: 0
Date: Sun, 16 Feb 2025 00:23:59 GMT
Server: <redacted>
```

3.4.5 Configuring Device to Return the Onboarding Response

Configure the 'response-manager' node inside the 'my-serial-number' device object to return the "my-onboarding-information" respsonse. Note that a catch-all rule (i.e., one without any 'match-criteria' defined) tells STZPD to always return this response to this device.

Request:

```
================================== NOTE: '\' line wrapping per RFC 8792 ================================

#!/bin/sh

# get a temp dir
TEMPDIR=`mktemp -d`

# initialize body for the POST request
cat << EOM > $TEMPDIR/response-manager.json
{
    "sztpd:response-manager": {
        "matched-response": [
            {
                "name": "catch-all-response",
                "response": {
                    "conveyed-information": {
                        "via-onboarding-response": {
                            "reference": "my-onboarding-information"
                        }
                    }
                }
            }
        ]
    }
}
EOM

# POST 'response-manager' to *device*
curl -isf -X POST --data @$TEMPDIR/response-manager.json -H "Content-Type:application/yang-data+json" --cac\
ert nbi_trust_chain.pem --user my-admin:my-secret https://127.0.0.1:8443/restconf/ds/ietf-datastores:runnin\
g/sztpd:devices/device=my-serial-number > output/post_response_manager_2.out
if [[ $? != 0 ]]; then
  echo "curl returned an error"
  exit 1
fi

# remove temp dir
rm -rf $TEMPDIR
```

Response:

```
HTTP/1.1 201 Created
Content-Length: 0
Date: Sun, 16 Feb 2025 00:24:00 GMT
Server: <redacted>
```

3.4.6 Simulate Device Getting Onboarding Information

Note that the SBI supports both JSON and XML.

Request:


```
================================== NOTE: '\' line wrapping per RFC 8792 ================================

#!/bin/sh

# get a temp dir
TEMPDIR=`mktemp -d`

# initialize the 'input' node for the RPC
cat << EOM > $TEMPDIR/input.xml
<input xmlns="urn:ietf:params:xml:ns:yang:ietf-sztp-bootstrap-server">
    <hw-model>model-x</hw-model>
    <os-name>vendor-os</os-name>
    <os-version>17.3R2.1</os-version>
</input>
EOM

# POST 'input' for the "get-bootstrapping-data" RPC resource
curl -isf -X POST --data @$TEMPDIR/input.xml -H "Content-Type:application/yang-data+xml" -H "Accept:applica\
tion/yang-data+xml" --cacert sbi_trust_chain.pem --key pki/client/end-entity/private_key.pem --cert pki/cli\
ent/end-entity/my_cert.pem --user my-serial-number:my-secret https://127.0.0.1:9443/restconf/operations/iet\
f-sztp-bootstrap-server:get-bootstrapping-data > output/rpc_get_bootstrapping_data_3.out

if [[ $? != 0 ]]; then
  echo "curl returned an error"
  exit 1
fi

# line-return needed for markdown
echo "" >> output/rpc_get_bootstrapping_data_3.out

# remove temp dir
rm -rf $TEMPDIR
```

Response:

```
================================== NOTE: '\' line wrapping per RFC 8792 ================================

HTTP/1.1 200 OK
Content-Type: application/yang-data+xml; charset=utf-8
Content-Length: 1649
Date: Sun, 16 Feb 2025 00:24:00 GMT
Server: <redacted>

<?xml version="1.0" ?>
<output xmlns="urn:ietf:params:xml:ns:yang:ietf-sztp-bootstrap-server">
  <conveyed-information>MIIEXgYLKoZIhvcNAQkQASqgggRNBIIESTw/eG1sIHZlcnNpb249IjEuMCIgPz4KPG9uYm9hcmRpbmctaW5\
mb3JtYXRpb24geG1sbnM9InVybjppZXRmOnBhcmFtczp4bWw6bnM6eWFuZzppZXRmLXN6dHAtY29udmV5ZWQtaW5mbyI\
1hZ2U+CiAgICA8b3MtbmFtZT5Gb29CYXJPUzwvb3MtbmFtZT4KICAgIDxvcy12ZXJzaW9uPjEuMi4zPC9vcy12ZXJzaW9uPgogICAgPGRvd\
25sb2FkLXVyaT5odHRwczovL2V4YW1wbGUuY29tL215LWJvb3QtaW1hZ2UuaW1nPC9kb3dubG9hZC11cmk+CiAgICA8aW1hZ2UtdmVyaWZp\
Y2F0aW9uPgogICAgICA8aGFzaC1hbGdvcml0aG0geG1sbnM6aWV0Zi1zenRwLWNvbnZleWVkLWluZm89InVybjppZXRmOnBhcmFtczp4bWw\
6bnM6eWFuZzppZXRmLXN6dHAtY29udmV5ZWQtaW5mbyI+aWV0Zi1zenRwLWNvbnZleWVkLWluZm86c2hhLTI1NjwvaGFzaC1hbGdvcml0aG\
0+CiAgICAgIDxoYXNoLXZhbHVlPjJhOjZkOjE1OjQzOjQ2OmRkOjg2OmIxOjRmOjhjOjNmOjBhOmYxOjc0OjFiOjNlOjZlOjcyOjk4OmQzO\
jdiOmMwOmNjOjdmOjgwOjk4OjcxOjE0OjhkOjk1OjgwOmQ4PC9oYXNoLXZhbHVlPgogICAgPC9pbWFnZS12ZXJpZmljYXRpb24+CiAgPC9i\
b290LWltYWdlPgogIDxwcmUtY29uZmlndXJhdGlvbi1zY3JpcHQ+SXk5aWFXNHZZbUZ6YUFwbFkyaHZJQ0pwYm5OcFpHVWdkR2hsSUhCeVp\
TMWpiMjVtYVdkMWNtRjBhVzl1TFhOamNtbHdkQzR1TGlJSzwvcHJlLWNvbmZpZ3VyYXRpb24tc2NyaXB0PgogIDxjb25maWd1cmF0aW9uLW\
hhbmRsaW5nPm1lcmdlPC9jb25maWd1cmF0aW9uLWhhbmRsaW5nPgogIDxjb25maWd1cmF0aW9uPlBIUnZjQ0I0Yld4dWN6MGlhSFIwY0hNN\
kwyVjRZVzF3YkdVdVkyOXRMMk52Ym1acFp5SStDaUFnUEdGdWVTMTRiV3d0WTI5dWRHVnVkQzF2YTJGNUx6NEtQQzkwYjNBK0NnPT08L2Nv\
bmZpZ3VyYXRpb24+CiAgPHBvc3QtY29uZmlndXJhdGlvbi1zY3JpcHQ+SXk5aWFXNHZZbUZ6YUFwbFkyaHZJQ0pwYm5OcFpHVWdkR2hsSUh\
CdmMzUXRZMjl1Wm1sbmRYSmhkR2x2Ymkxelkzbm5SbjSFF1TGk0aUNnPT08L3Bvc3QtY29uZmlndXJhdGlvbi1zY3JpcHQ+Cjwvb25ib2FyZG\
luZy1pbmZvcm1hdGlvbj4K</conveyed-information>
</output>
```

3.4.7 Simulate Device Posting a Progress Report

In addition to RFC 8572 enabling a device to obtain bootstrapping data, it also enables a device to report progress, enabling a bootstrap server to:

1) learn of any issues preventing bootstrapping from succeeding.
2) learn of any dynamically-generated SSH host keys and/or TLS certificates, such as may be needed to authenticate the device via other protocols (e.g., NETCONF, RESTCONF, etc.)

Note that the SBI supports both JSON and XML.

Request:

```
================================== NOTE: '\' line wrapping per RFC 8792 ================================

#!/bin/sh

# get a temp dir
```



```
TEMPDIR=`mktemp -d`

# initialize the 'input' node for the RPC
cat << EOM > $TEMPDIR/input.xml
<input xmlns="urn:ietf:params:xml:ns:yang:ietf-sztp-bootstrap-server">
    <progress-type>bootstrap-complete</progress-type>
    <message>example message</message>
    <ssh-host-keys>
        <ssh-host-key>
            <algorithm>ssh-rsa</algorithm>
            <key-data>BASE64VALUE=</key-data>
        </ssh-host-key>
        <ssh-host-key>
            <algorithm>rsa-sha2-256</algorithm>
            <key-data>BASE64VALUE=</key-data>
        </ssh-host-key>
    </ssh-host-keys>
    <trust-anchor-certs>
        <trust-anchor-cert>BASE64VALUE=</trust-anchor-cert>
    </trust-anchor-certs>
</input>
EOM

# POST it to the "report-progress" RPC resource
curl -isf -X POST --data @$TEMPDIR/input.xml -H "Content-Type:application/yang-data+xml" --cacert sbi_trust\
_chain.pem --key pki/client/end-entity/private_key.pem --cert pki/client/end-entity/my_cert.pem --user my-s\
erial-number:my-secret https://127.0.0.1:9443/restconf/operations/ietf-sztp-bootstrap-server:report-progres\
s > output/rpc_report_progress.out

# cleanup
rm -f $TEMPDIR/input.xml
```

Response:

```
HTTP/1.1 204 No Content
Date: Sun, 16 Feb 2025 00:24:00 GMT
Server: <redacted>
```

3.4.8 Delete Onboarding Information

Now delete all of the onboarding config to get back to baseline (not that it matters). Note that the configuration is removed in the opposite order to avoid SZTPD throwing a validation error due to dangling references.

Request:

```
================================== NOTE: '\' line wrapping per RFC 8792 ================================

#!/bin/sh

# These are removed in opposite order to avoid dangling reference validation errors.

curl -isf -X DELETE --cacert nbi_trust_chain.pem --user my-admin:my-secret https://127.0.0.1:8443/restconf/\
ds/ietf-datastores:running/sztpd:devices/device=my-serial-number/response-manager > output/delete_onboardin\
g_info.out
if [[ $? != 0 ]]; then
  echo "curl returned an error"
  exit 1
fi

curl -isf -X DELETE --cacert nbi_trust_chain.pem --user my-admin:my-secret https://127.0.0.1:8443/restconf/\
ds/ietf-datastores:running/sztpd:responses >> output/delete_onboarding_info.out
if [[ $? != 0 ]]; then
  echo "curl returned an error"
  exit 1
fi

curl -isf -X DELETE --cacert nbi_trust_chain.pem --user my-admin:my-secret https://127.0.0.1:8443/restconf/\
ds/ietf-datastores:running/sztpd:conveyed-information/boot-images >> output/delete_onboarding_info.out
if [[ $? != 0 ]]; then
  echo "curl returned an error"
  exit 1
fi

curl -isf -X DELETE --cacert nbi_trust_chain.pem --user my-admin:my-secret https://127.0.0.1:8443/restconf/\
ds/ietf-datastores:running/sztpd:conveyed-information/scripts >> output/delete_onboarding_info.out
if [[ $? != 0 ]]; then
  echo "curl returned an error"
  exit 1
fi

curl -isf -X DELETE --cacert nbi_trust_chain.pem --user my-admin:my-secret https://127.0.0.1:8443/restconf/\
```



```
ds/ietf-datastores:running/sztpd:conveyed-information/configurations >> output/delete_onboarding_info.out
if [[ $? != 0 ]]; then
  echo "curl returned an error"
  exit 1
fi
```

Response:

```
HTTP/1.1 204 No Content
Date: Sun, 16 Feb 2025 00:24:00 GMT
Server: <redacted>

HTTP/1.1 204 No Content
Date: Sun, 16 Feb 2025 00:24:00 GMT
Server: <redacted>

HTTP/1.1 204 No Content
Date: Sun, 16 Feb 2025 00:24:00 GMT
Server: <redacted>

HTTP/1.1 204 No Content
Date: Sun, 16 Feb 2025 00:24:00 GMT
Server: <redacted>

HTTP/1.1 204 No Content
Date: Sun, 16 Feb 2025 00:24:00 GMT
Server: <redacted>
```

3.5 View Logs

3.5.1 View Device's Bootstrapping Log

Notice how the "GET" on yang-library resource, and all of the 'get-bootstrapping-data' RPC requests, and the 'report-progress' RPC request, from above all appear below.

Request:

```
================================ NOTE: '\' line wrapping per RFC 8792 ================================

#!/bin/sh

curl -isf -H "Accept:application/yang-data+json" --cacert nbi_trust_chain.pem --user my-admin:my-secret htt\
ps://127.0.0.1:8443/restconf/ds/ietf-datastores:operational/sztpd:devices/device=my-serial-number/bootstrap\
ping-log > output/get_bootstrapping_log.out

if [[ $? != 0 ]]; then
  echo "curl returned an error"
  exit 1
fi

# FIXME: is YANGcore not having a closing LR?
echo "\n" >> output/get_bootstrapping_log.out
```

Response:

```
HTTP/1.1 200 OK
Content-Type: application/yang-data+json; charset=utf-8
Content-Length: 3724
Date: Sun, 16 Feb 2025 00:24:00 GMT
Server: <redacted>

{
  "sztpd:bootstrapping-log": {
    "bootstrapping-log-record": [
      {
        "timestamp": "2025-02-16T00:23:59Z",
        "source-ip": "127.0.0.1",
        "method": "POST",
        "path": "/restconf/operations/ietf-sztp-bootstrap-server:get-bootstrapping-data",
        "return-code": 404,
        "error-returned": {
          "ietf-restconf:errors": {
            "error": [
              {
                "error-type": "application",
                "error-tag": "data-missing",
                "error-message": "No responses configured."
              }
            ]
          }
        },
```



```
      "event-details": {
        "get-bootstrapping-data-event": {
          "passed-input": {
            "hw-model": "model-x",
            "os-name": "vendor-os",
            "os-version": "17.3R2.1"
          },
          "selected-response": "no-responses-configured"
        }
      }
    },
    {
      "timestamp": "2025-02-16T00:23:59Z",
      "source-ip": "127.0.0.1",
      "method": "POST",
      "path": "/restconf/operations/ietf-sztp-bootstrap-server:get-bootstrapping-data",
      "return-code": 200,
      "event-details": {
        "get-bootstrapping-data-event": {
          "passed-input": {
            "hw-model": "model-x",
            "os-name": "vendor-os",
            "os-version": "17.3R2.1"
          },
          "selected-response": "catch-all-response",
          "response-details": {
            "managed-response": {
              "conveyed-information": {
                "via-redirect-response": {
                  "referenced-definition": "my-redirect-information"
                }
              }
            }
          }
        }
      }
    },
    {
      "timestamp": "2025-02-16T00:24:00Z",
      "source-ip": "127.0.0.1",
      "method": "POST",
      "path": "/restconf/operations/ietf-sztp-bootstrap-server:get-bootstrapping-data",
      "return-code": 200,
      "event-details": {
        "get-bootstrapping-data-event": {
          "passed-input": {
            "hw-model": "model-x",
            "os-name": "vendor-os",
            "os-version": "17.3R2.1"
          },
          "selected-response": "catch-all-response",
          "response-details": {
            "managed-response": {
              "conveyed-information": {
                "via-onboarding-response": {
                  "referenced-definition": "my-onboarding-information"
                }
              }
            }
          }
        }
      }
    },
    {
      "timestamp": "2025-02-16T00:24:00Z",
      "source-ip": "127.0.0.1",
      "method": "POST",
      "path": "/restconf/operations/ietf-sztp-bootstrap-server:report-progress",
      "return-code": 204,
      "event-details": {
        "report-progress-event": {
          "passed-input": {
            "progress-type": "bootstrap-complete",
            "message": "example message",
            "ssh-host-keys": {
              "ssh-host-key": [
                {
                  "algorithm": "ssh-rsa",
                  "key-data": "BASE64VALUE="
                },
                {
                  "algorithm": "rsa-sha2-256",
                  "key-data": "BASE64VALUE="
                }
              ]
            },
            "trust-anchor-certs": {
```



```
                "trust-anchor-cert": [
                  "BASE64VALUE="
                ]
              }
            },
            "dynamic-callout": {
              "no-callout-configured": [
                null
              ]
            }
          }
        }
      }
    ]
  }
}
```

3.5.2 View YANGcore's Audit Log

Notice how all the commands run above appear below. The audit log captures all interactions on both the NBI (management) and SBI (bootstapping) interfaces. See the 'get-bootstrapping-data' and 'report-process' log entries for examples of the SBI interactions.

Request:

```
================================ NOTE: '\' line wrapping per RFC 8792 ================================

#!/bin/sh

# fetch data to file
curl -isf -H "Accept:application/yang-data+json" --cacert nbi_trust_chain.pem --user my-admin:my-secret htt\
ps://127.0.0.1:8443/restconf/ds/ietf-datastores:operational/yangcore:audit-log > output/get_audit_log.out

# ensure no error
if [[ $? != 0 ]]; then
  echo "curl returned an error"
  exit 1
fi

# FIXME: is YANGcore not having a closing LR?
echo "\n" >> output/get_audit_log.out
```

Response:

```
================================ NOTE: '\' line wrapping per RFC 8792 ================================

HTTP/1.1 200 OK
Content-Type: application/yang-data+json; charset=utf-8
Content-Length: 5824
Date: Sun, 16 Feb 2025 00:24:00 GMT
Server: <redacted>

{
  "yangcore:audit-log": {
    "audit-log-record": [
      {
        "timestamp": "2025-02-16T00:23:54Z",
        "source-ip": "127.0.0.1",
        "host": "127.0.0.1:8080",
        "method": "PUT",
        "path": "/restconf/ds/ietf-datastores:running",
        "outcome": "success",
        "comment": "No authorization required for fresh installs."
      },
      {
        "timestamp": "2025-02-16T00:23:59Z",
        "source-ip": "127.0.0.1",
        "host": "127.0.0.1:9443",
        "method": "POST",
        "path": "/restconf/operations/ietf-sztp-bootstrap-server:get-bootstrapping-data",
        "outcome": "success"
      },
      {
        "timestamp": "2025-02-16T00:23:59Z",
        "source-ip": "127.0.0.1",
        "host": "127.0.0.1:8443",
        "method": "POST",
        "path": "/restconf/ds/ietf-datastores:running/sztpd:conveyed-information",
        "outcome": "success"
      },
      {
        "timestamp": "2025-02-16T00:23:59Z",
```



```
        "source-ip": "127.0.0.1",
        "host": "127.0.0.1:8443",
        "method": "POST",
        "path": "/restconf/ds/ietf-datastores:running",
        "outcome": "success"
      },
      {
        "timestamp": "2025-02-16T00:23:59Z",
        "source-ip": "127.0.0.1",
        "host": "127.0.0.1:8443",
        "method": "POST",
        "path": "/restconf/ds/ietf-datastores:running/sztpd:devices/device=my-serial-number",
        "outcome": "success"
      },
      {
        "timestamp": "2025-02-16T00:23:59Z",
        "source-ip": "127.0.0.1",
        "host": "127.0.0.1:9443",
        "method": "POST",
        "path": "/restconf/operations/ietf-sztp-bootstrap-server:get-bootstrapping-data",
        "outcome": "success"
      },
      {
        "timestamp": "2025-02-16T00:23:59Z",
        "source-ip": "127.0.0.1",
        "host": "127.0.0.1:8443",
        "method": "DELETE",
        "path": "/restconf/ds/ietf-datastores:running/sztpd:devices/device=my-serial-number/response-manage\
r",
        "outcome": "success"
      },
      {
        "timestamp": "2025-02-16T00:23:59Z",
        "source-ip": "127.0.0.1",
        "host": "127.0.0.1:8443",
        "method": "DELETE",
        "path": "/restconf/ds/ietf-datastores:running/sztpd:responses",
        "outcome": "success"
      },
      {
        "timestamp": "2025-02-16T00:23:59Z",
        "source-ip": "127.0.0.1",
        "host": "127.0.0.1:8443",
        "method": "DELETE",
        "path": "/restconf/ds/ietf-datastores:running/sztpd:conveyed-information/bootstrap-servers",
        "outcome": "success"
      },
      {
        "timestamp": "2025-02-16T00:23:59Z",
        "source-ip": "127.0.0.1",
        "host": "127.0.0.1:8443",
        "method": "POST",
        "path": "/restconf/ds/ietf-datastores:running/sztpd:conveyed-information",
        "outcome": "success"
      },
      {
        "timestamp": "2025-02-16T00:23:59Z",
        "source-ip": "127.0.0.1",
        "host": "127.0.0.1:8443",
        "method": "POST",
        "path": "/restconf/ds/ietf-datastores:running/sztpd:conveyed-information",
        "outcome": "success"
      },
      {
        "timestamp": "2025-02-16T00:23:59Z",
        "source-ip": "127.0.0.1",
        "host": "127.0.0.1:8443",
        "method": "POST",
        "path": "/restconf/ds/ietf-datastores:running/sztpd:conveyed-information",
        "outcome": "success"
      },
      {
        "timestamp": "2025-02-16T00:23:59Z",
        "source-ip": "127.0.0.1",
        "host": "127.0.0.1:8443",
        "method": "POST",
        "path": "/restconf/ds/ietf-datastores:running",
        "outcome": "success"
      },
      {
        "timestamp": "2025-02-16T00:23:59Z",
        "source-ip": "127.0.0.1",
        "host": "127.0.0.1:8443",
        "method": "POST",
        "path": "/restconf/ds/ietf-datastores:running/sztpd:devices/device=my-serial-number",
        "outcome": "success"
      },
```



```
      {
        "timestamp": "2025-02-16T00:24:00Z",
        "source-ip": "127.0.0.1",
        "host": "127.0.0.1:9443",
        "method": "POST",
        "path": "/restconf/operations/ietf-sztp-bootstrap-server:get-bootstrapping-data",
        "outcome": "success"
      },
      {
        "timestamp": "2025-02-16T00:24:00Z",
        "source-ip": "127.0.0.1",
        "host": "127.0.0.1:8443",
        "method": "DELETE",
        "path": "/restconf/ds/ietf-datastores:running/sztpd:devices/device=my-serial-number/response-manage\
r",
        "outcome": "success"
      },
      {
        "timestamp": "2025-02-16T00:24:00Z",
        "source-ip": "127.0.0.1",
        "host": "127.0.0.1:8443",
        "method": "DELETE",
        "path": "/restconf/ds/ietf-datastores:running/sztpd:responses",
        "outcome": "success"
      },
      {
        "timestamp": "2025-02-16T00:24:00Z",
        "source-ip": "127.0.0.1",
        "host": "127.0.0.1:8443",
        "method": "DELETE",
        "path": "/restconf/ds/ietf-datastores:running/sztpd:conveyed-information/boot-images",
        "outcome": "success"
      },
      {
        "timestamp": "2025-02-16T00:24:00Z",
        "source-ip": "127.0.0.1",
        "host": "127.0.0.1:8443",
        "method": "DELETE",
        "path": "/restconf/ds/ietf-datastores:running/sztpd:conveyed-information/scripts",
        "outcome": "success"
      },
      {
        "timestamp": "2025-02-16T00:24:00Z",
        "source-ip": "127.0.0.1",
        "host": "127.0.0.1:8443",
        "method": "DELETE",
        "path": "/restconf/ds/ietf-datastores:running/sztpd:conveyed-information/configurations",
        "outcome": "success"
      },
      {
        "timestamp": "2025-02-16T00:24:00Z",
        "source-ip": "127.0.0.1",
        "host": "127.0.0.1:9443",
        "method": "POST",
        "path": "/restconf/operations/ietf-sztp-bootstrap-server:report-progress",
        "outcome": "success"
      }
    ]
  }
}
```


4 Special Topics

This section goes over standards-based behavior that may not be immediately obvious to some readers. Note that this section is very similar to the section presented in the YANGcore User Guide, but it uses new/different example that are unique to SZTPD.

4.1 HTTP Headers

SZTPD inherits its HTTP Header processing logic from YANGcore. Please see the "HTTP Headers" section in the YANGcore User Guide for details.

4.2 Ordered Lists

SZTPD inherits its support for ordered lists from YANGcore. Please see the "Ordered Lists" section in the YANGcore User Guide for details.

There are three ordered lists in SZTPD as follows:

```
1) /boot-images/boot-image/download-uri
2) /responses/redirect-response/redirect-information/bootstrap-server
3) /devices/device/response-manager/matched-response
```

4.2.1 Example: POST-ing a Matched-Response after Another

Assuming a device's "response-manager" contains the ordered list of "matched-response" entries: "resp1", "resp2", and "resp3", then the command:

```
$ cat matched-response.json
{
  "sztpd:matched-response": [
    {
      'name': 'new',
      'response': {
        'none': [None]
      }
    }
  ]
}

$ curl -i -X POST --data @matched-response.json -H "Content-Type:application/yang-data+json" \
http://127.0.0.1:8080/restconf/ds/ietf-datastores:running/sztpd:devices/device=my-serial\
-number/response-manager?insert=after&point=/sztpd:devices/device=my-serial-number/respo\
nse-manager/matched-response=resp1
```

would result in the ordered list of entries: "resp1", "new", "resp2", and "resp3".

4.2.2 Example: PUT-ing an entry before another entry

Assuming a redirect-response contains the ordered list of "bootstrap-server" entries: "bs1", "bs2", and "bs3", then the command:

```
$ cat bootstrap-server.json
{
  "sztpd:bootstrap-server": [
    "bs-3"
  ]
}

$ curl -i -X PUT --data @bootstrap-server.json -H "Content-Type:application/yang-data+json" \
http://127.0.0.1:8080/restconf/ds/ietf-datastores:running/sztpd:conveyed-information-re\
sponses/redirect-response=my-redirect-information/redirect-information/bootstrap\
-server=bs3?insert=before&point=/sztpd:responses/redirect-informat\
ion-response=my-redirect-information/redirect-information/bootstrap-server=bs2
```

would result in the ordered list of entries: "bs1", "bs3", and "bs2".

4.3 Special Characters

SZTPD inherits the need to handle special characters from YANGcore. Please see the "Special Characters" section in the YANGcore User Guide for details.

This concern experiences what might be described as its worst case scenario when inserting or moving an entry into the ordered list "boot-images/boot-image/download-uri". See Ordered Lists.

The "download-uri" list exhibits the worst case because its keys are themselves URLs, and thus a multiplicity of URLs need to be encoded into a URL. Tha e following examples illustrate using this API.

4.3.1 Example: POST-ing a Download-URI after Another

Assuming the configured "download-uri" leaf-list contains values as follows:

```
{
  "sztpd:download-uri" : [
    https://cdn1.example.com/path/to/image/file,
    https://cdn2.example.com/path/to/image/file,
    https://cdn3.example.com/path/to/image/file
  ]
}
```

then the command:

```
$ cat download-uri.json
{
  "sztpd:download-uri": [
    https://new4.example.com/path/to/image/file
  ]
}

$ curl -i -X POST --data @download-uri.json -H "Content-Type:application/yang-data+json" \
http://127.0.0.1:8080/restconf/ds/ietf-datastores:running/sztpd:boot-images/boot-ima\
ge=vendoros-19.2r1b6.img?insert=after&point=/sztpd:boot-images/boot-image=vendoros-1\
9.2r1b6.img/download-uri=https%3A%2F%2Fcdn1.example.com%2Fpath%2Fto%2Fimage%2Ffile
```

would result in the list of entries:

```
{
  "sztpd:download-uri" : [
    https://cdn1.example.com/path/to/image/file,
    https://new4.example.com/path/to/image/file,
    https://cdn2.example.com/path/to/image/file,
    https://cdn3.example.com/path/to/image/file
  ]
}
```

4.3.2 Example: PUT-ing a Download-URI after Another

Assuming the configured "download-uri" leaf-list contains values as follows:

```
{
  "sztpd:download-uri" : [
    https://cdn1.example.com/path/to/image/file,
    https://new4.example.com/path/to/image/file,
    https://cdn2.example.com/path/to/image/file,
    https://cdn3.example.com/path/to/image/file
  ]
}
```

then the command:

```
$ cat download-uri.json
{
  "sztpd:download-uri": [
    https://new4.example.com/path/to/image/file
  ]
}

$ curl -i -X POST --data @download-uri.json -H "Content-Type:application/yang-data+json" \
http://127.0.0.1:8080/restconf/ds/ietf-datastores:running/sztpd:boot-images/boot-ima\
ge=vendoros-19.2r1b6.img/download-uri=https%3A%2F%2Fnew4.example.com%2Fpath%2Fto%2Fimage%\
2Ffile?insert=before&point=/sztpd:boot-images/boot-image=vendoros-19.2r1b6.img/downl\
oad-uri=https%3A%2F%2Fcdn3.example.com%2Fpath%2Fto%2Fimage%2Ffile
```


would result in the list of entries:

```
{
  "sztpd:download-uri" : [
    https://cdn1.example.com/path/to/image/file ,
    https://cdn2.example.com/path/to/image/file ,
    https://new4.example.com/path/to/image/file ,
    https://cdn3.example.com/path/to/image/file
  ]
}
```

4.4 List Pagination

SZTPD inherits its List Pagination support from YANGcore. Please see the "List Pagination" section in the YANGcore User Guide for details.

4.5 Event Triggers

SZTPD inherits event triggers from YANGcore. Please see the "Event Triggers" section in the YANGcore User Guide for details.

4.6 Plugins

SZTPD inherits plugins from YANGcore. Please see the "Plugins" section in the YANGcore User Guide for details. In addition to the dynamic callouts supported by YANGcore, SZTPD supports also defines some Dynamic Callouts, which are implemented using plugins.

5 Northbound API Details

5.1 NBI YANG Library

For any RESTCONF server, its YANG Library is the heart of the API, as it unambiguously identifies what API the server implements. SZTPD's YANG Library, presented below, is a superset of YANGCore's YANG Library. See the "YANG Library" section in the YANGcore User Guide for details.

```
{
    "ietf-yang-library:yang-library": {
        "module-set": [
            {
                "name": "shared-module-set",
                "module": [
                    {
                        "name": "ietf-datastores",
                        "revision": "2018-02-14",
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-yang-datastores"
                    },
                    {
                        "name": "ietf-yang-library",
                        "revision": "2019-01-04",
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-yang-library"
                    },
                    {
                        "name": "ietf-crypto-types",
                        "revision": "2024-10-10",
                        "feature": [
                            "cleartext-private-keys"
                        ],
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-crypto-types"
                    },
                    {
                        "name": "ietf-truststore",
                        "revision": "2024-10-10",
                        "feature": [
                            "certificates",
                            "central-truststore-supported"
                        ],
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-truststore"
                    },
                    {
                        "name": "ietf-keystore",
                        "revision": "2024-10-10",
                        "feature": [
                            "asymmetric-keys",
                            "central-keystore-supported"
```



```
            ],
            "namespace": "urn:ietf:params:xml:ns:yang:ietf-keystore"
        },
        {
            "name": "ietf-tcp-common",
            "revision": "2024-10-10",
            "namespace": "urn:ietf:params:xml:ns:yang:ietf-tcp-common"
        },
        {
            "name": "ietf-tcp-client",
            "revision": "2024-10-10",
            "namespace": "urn:ietf:params:xml:ns:yang:ietf-tcp-client"
        },
        {
            "name": "ietf-tcp-server",
            "revision": "2024-10-10",
            "namespace": "urn:ietf:params:xml:ns:yang:ietf-tcp-server"
        },
        {
            "name": "iana-tls-cipher-suite-algs",
            "revision": "2024-10-16",
            "namespace": "urn:ietf:params:xml:ns:yang:iana-tls-cipher-suite-algs"
        },
        {
            "name": "ietf-tls-common",
            "revision": "2024-10-10",
            "namespace": "urn:ietf:params:xml:ns:yang:ietf-tls-common"
        },
        {
            "name": "ietf-tls-server",
            "revision": "2024-10-10",
            "feature": [
                "server-ident-x509-cert",
                "client-auth-supported",
                "client-auth-x509-cert"
            ],
            "namespace": "urn:ietf:params:xml:ns:yang:ietf-tls-server"
        },
        {
            "name": "ietf-udp-server",
            "revision": "2024-10-15",
            "feature": [],
            "namespace": "urn:ietf:params:xml:ns:yang:ietf-udp-server"
        },
        {
            "name": "ietf-http-server",
            "revision": "2025-02-15",
            "feature": [
                "client-auth-supported"
            ],
            "namespace": "urn:ietf:params:xml:ns:yang:ietf-http-server"
        },
        {
            "name": "ietf-restconf-server",
            "revision": "2025-02-15",
            "feature": [
                "listen",
                "tcp-listen",
                "tls-listen",
                "central-restconf-server-supported"
            ],
            "namespace": "urn:ietf:params:xml:ns:yang:ietf-restconf-server"
        },
        {
            "name": "yangcore",
            "revision": "2025-02-15",
            "namespace": "https://watsen.net/yangcore"
        },
        {
            "name": "yangcore-yl",
            "revision": "2025-02-15",
            "namespace": "https://watsen.net/yangcore-yl"
        },
        {
            "name": "ietf-sztp-conveyed-info",
            "revision": "2019-04-30",
            "namespace": "urn:ietf:params:xml:ns:yang:ietf-sztp-conveyed-info"
        },
        {
            "name": "wn-x509-c2n",
            "revision": "2025-02-15",
            "namespace": "https://watsen.net/wnc2n"
        },
        {
            "name": "sztpd",
            "revision": "2025-02-15",
            "namespace": "https://watsen.net/sztpd"
```



```
                    }
                ],
                "import-only-module": [
                    {
                        "name": "ietf-yang-types",
                        "revision": "2013-07-15",
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-yang-types"
                    },
                    {
                        "name": "ietf-inet-types",
                        "revision": "2013-07-15",
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-inet-types"
                    },
                    {
                        "name": "iana-crypt-hash",
                        "revision": "2014-08-06",
                        "namespace": "urn:ietf:params:xml:ns:yang:iana-crypt-hash"
                    },
                    {
                        "name": "ietf-x509-cert-to-name",
                        "revision": "2014-12-10",
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-x509-cert-to-name"
                    },
                    {
                        "name": "ietf-restconf",
                        "revision": "2017-01-26",
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-restconf"
                    },
                    {
                        "name": "ietf-netconf-acm",
                        "revision": "2018-02-14",
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-netconf-acm"
                    },
                    {
                        "name": "yangcore-common",
                        "revision": "2025-02-15",
                        "namespace": "https://watsen.net/yangcore-rpcs"
                    }
                ]
            }
        ],
        "schema": [
            {
                "name": "shared-schema",
                "module-set": [
                    "shared-module-set"
                ]
            }
        ],
        "datastore": [
            {
                "name": "ietf-datastores:operational",
                "schema": "shared-schema"
            },
            {
                "name": "ietf-datastores:running",
                "schema": "shared-schema"
            }
        ],
        "content-id": "TBD"
    }
}
```

5.2 NBI YANG Modules

The NBI YANG Library defines the API as a collection of YANG modules. There are thirty-one YANG modules in SZTPD. The most exact description of the API is had by reviewing these YANG modules directly, in conjuction with the "features" statements in the NBI YANG Library.

Ideally this guide would include the thirty-one YANG modules used by SZTPD, but they are a combined total of over fifteeen thousand lines, which is too much to include here.

In lieu of including the YANG modules in this guide, the reader may find these YANG modules in the "yang" directory in both the 'yangcore' and 'sztpd' packages. Where these files are located varies by Python installation. The following command find the directory the YANG files are located for the "yangcore" and "sztpd" packages:

```
python -c 'import importlib.resources as resources; print(resources.files("yangcore") / "yang")'
python -c 'import importlib.resources as resources; print(resources.files("sztpd") / "yang")'
```


For example: "/Users/kent/.pyenv/versions/3.12.8/lib/python3.12/site-packages/yangcore/yang" and "/Users/kent/.pyenv/versions/3.12.8/lib/python3.12/site-packages/sztpd/yang".

5.3 NBI Complete Tree Diagram

This section presents the complete YANG tree diagram for SZTPD's Northbound API. This tree diagram is generated using the YANG library presented in NBI YANG Library. This diagram is useful as it presents the entire API one diagram.

Heads-Up! The NBI Top-level Nodes section presents all the new top-level nodes SZTPD adds, but it does not show the few additions to YANGcore top-level nodes made by the SZTPD application. However, this diagram does show them, if one searches for the "sztpd:" prefix.

```
================================ NOTE: '\' line wrapping per RFC 8792 ================================

+--ro yangcore:audit-log
|  +--ro audit-log-record*
|     +--ro comment? <string>
|     +--ro host <string>
|     +--ro input?
|     +--ro method <enumeration>
|     +--ro outcome <enumeration>
|     +--ro path <string>
|     +--ro source-ip <ip-address(union)>
|     +--ro source-proxies* <string>
|     +--ro timestamp <date-and-time(string)>
|     +--ro username? <string>
+--rw sztpd:conveyed-information
|  +--rw boot-images
|  |  +--rw boot-image* [name]
|  |     +--rw download-uri# <uri(string)>
|  |     +--rw image-verification* [hash-algorithm]
|  |     |  +--rw hash-algorithm <identityref>
|  |     |  +--rw hash-value <hex-string(string)>
|  |     +--rw name <string>
|  |     +--rw os-name <string>
|  |     +--rw os-version <string>
|  +--rw bootstrap-servers
|  |  +--rw bootstrap-server* [name]
|  |     +--rw address <host(union)>
|  |     +--rw name <string>
|  |     +--rw port? <port-number(uint16)>
|  |     +--rw trust-anchor? <trust-anchor-cert-cms(binary)>
|  +--rw configurations
|  |  +--rw configuration* [name]
|  |     +--rw config-data? <binary>
|  |     +--rw handling? <enumeration>
|  |     +--rw media-type? <enumeration>
|  |     +--rw name <string>
|  +--rw scripts
|     +--rw script* [name]
|        +--rw code <script(binary)>
|        +--rw name <string>
|        +--rw type <enumeration>
+--rw sztpd:device-types
|  +--rw device-type* [name]
|     +--rw identity-certificates!
|     |  +--rw serial-number-extraction? <identityref>
|     |  +--rw verification
|     |     +--rw central-truststore-reference
|     |        +--rw certificate? <central-certificate-ref(leafref)>
|     |        +--rw certificate-bag? <central-certificate-bag-ref(leafref)>
|     +--rw name <string>
|     +--rw ownership-authorization!
|        +--rw dynamic-callout
|           +--rw reference? <leafref>
+--rw sztpd:devices
|  +--rw device* [serial-number]
|     +--rw activation-code? <crypt-hash(string)>
|     +--ro bootstrapping-log
|     |  +--ro bootstrapping-log-record*
|     |     +--ro error-returned?
|     |     +--ro event-details
|     |     |  +--ro (event-type)
|     |     |     +--:(get-bootstrapping-data-event)
|     |     |     |  +--ro get-bootstrapping-data-event
|     |     |     |     +--ro passed-input
|     |     |     |     +--ro response-details
|     |     |     |     |  +--ro (response-type)
|     |     |     |     |     +--:(managed-response)
|     |     |     |     |        +--ro managed-response
|     |     |     |     |           +--ro conveyed-information
```



```
|     |     |     |     |              +--ro (conveyed-information-handler)
|     |     |     |     |                 +--:(dynamic-callout)
|     |     |     |     |                 |  +--ro via-dynamic-callout
|     |     |     |     |                 |     +--ro (result-type)?
|     |     |     |     |                 |        +--:(callout-configured)
|     |     |     |     |                 |        |  +--ro callout-type <enumeration>
|     |     |     |     |                 |        |  +--ro (function-or-webhook)?
|     |     |     |     |                 |        |  |  +--:(function)
|     |     |     |     |                 |        |  |     +--ro function-details
|     |     |     |     |                 |        |  |     |  +--ro function <string>
|     |     |     |     |                 |        |  |     |  +--ro plugin <string>
|     |     |     |     |                 |        |  |     +--ro function-results
|     |     |     |     |                 |        |  |        +--ro (exit-status)
|     |     |     |     |                 |        |  |           +--:(exception-thrown)
|     |     |     |     |                 |        |  |           |  +--ro exception-thrown? <string>
|     |     |     |     |                 |        |  |           +--:(exited-normally)
|     |     |     |     |                 |        |  |              +--ro exited-normally? <string>
|     |     |     |     |                 |        |  +--ro name <string>
|     |     |     |     |                 |        |  +--ro rpc-supported <identityref>
|     |     |     |     |                 |        +--:(no-callout-configured)
|     |     |     |     |                 |           +--ro no-callout-configured? <empty>
|     |     |     |     |                 +--:(onboarding-information)
|     |     |     |     |                 |  +--ro via-onboarding-response
|     |     |     |     |                 |     +--ro referenced-definition? <string>
|     |     |     |     |                 +--:(redirect-information)
|     |     |     |     |                    +--ro via-redirect-response
|     |     |     |     |                       +--ro referenced-definition? <string>
|     |     |     |     +--ro selected-response? <union>
|     |     |     +--:(report-progress-event)
|     |     |        +--ro report-progress-event
|     |     |           +--ro dynamic-callout
|     |     |           |  +--ro (result-type)?
|     |     |           |     +--:(callout-configured)
|     |     |           |     |  +--ro callout-type <enumeration>
|     |     |           |     |  +--ro (function-or-webhook)?
|     |     |           |     |  |  +--:(function)
|     |     |           |     |  |     +--ro function-details
|     |     |           |     |  |     |  +--ro function <string>
|     |     |           |     |  |     |  +--ro plugin <string>
|     |     |           |     |  |     +--ro function-results
|     |     |           |     |  |        +--ro (exit-status)
|     |     |           |     |  |           +--:(exception-thrown)
|     |     |           |     |  |           |  +--ro exception-thrown? <string>
|     |     |           |     |  |           +--:(exited-normally)
|     |     |           |     |  |              +--ro exited-normally? <string>
|     |     |           |     |  +--ro name <string>
|     |     |           |     |  +--ro rpc-supported <identityref>
|     |     |           |     +--:(no-callout-configured)
|     |     |           |        +--ro no-callout-configured? <empty>
|     |     |           +--ro passed-input
|     |     +--ro method <string>
|     |     +--ro path <string>
|     |     +--ro return-code <uint16>
|     |     +--ro source-ip <ip-address(union)>
|     |     +--ro timestamp <date-and-time(string)>
|     +--rw device-type <leafref>
|     +--ro lifecycle-statistics
|     |  +--ro nbi-access-stats
|     |  |  +--ro created <date-and-time(string)>
|     |  |  +--ro last-modified <date-and-time(string)>
|     |  |  +--ro num-times-modified <uint16>
|     |  +--ro sbi-access-stats
|     |     +--ro first-accessed <date-and-time(string)>
|     |     +--ro last-accessed <date-and-time(string)>
|     |     +--ro num-times-accessed <uint16>
|     +--rw response-manager
|     |  +--rw matched-response# [name]
|     |     +--rw match-criteria!
|     |     |  +--rw match* [key]
|     |     |     +--rw key <string>
|     |     |     +--rw not? <empty>
|     |     |     +--rw (test-type)
|     |     |        +--:(present)
|     |     |        |  +--rw present? <empty>
|     |     |        +--:(regex)
|     |     |        |  +--rw regex? <string>
|     |     |        +--:(value)
|     |     |           +--rw value? <string>
|     |     +--rw name <string>
|     |     +--rw response
|     |        +--rw reporting-level? <enumeration>
|     |        +--rw (response-handler)
|     |           +--:(managed-response)
|     |           |  +--rw conveyed-information
|     |           |     +--rw (conveyed-information-handler)
|     |           |        +--:(dynamic-callout)
|     |           |        |  +--rw via-dynamic-callout
```



```
|  |     |           |        |     +--rw reference <leafref>
|  |     |           |        +--:(onboarding-information)
|  |     |           |        |  +--rw via-onboarding-response
|  |     |           |        |     +--rw config-template-params
|  |     |           |        |     |  +--rw config-template-param* [key]
|  |     |           |        |     |     +--rw key <string>
|  |     |           |        |     |     +--rw value <string>
|  |     |           |        |     +--rw reference <leafref>
|  |     |           |        +--:(redirect-information)
|  |     |           |           +--rw via-redirect-response
|  |     |           |              +--rw reference <leafref>
|  |     |           +--:(none)
|  |     |              +--rw none? <empty>
|  |     +--rw serial-number <string>
+--rw yangcore:dynamic-callouts
|  +--rw dynamic-callout* [name]
|     +--rw (callout-type)
|     |  +--:(call-function)
|     |     +--rw call-function
|     |     |  +--rw function <leafref>
|     |     |  +--rw plugin <leafref>
|     |     +--rw opaque?
|     +--rw name <string>
|     +--rw rpc-supported <identityref>
+--rw ietf-keystore:keystore
|  +--rw asymmetric-keys
|     +--rw asymmetric-key* [name]
|        +--rw certificates
|        |  +--rw certificate* [name]
|        |     +--rw cert-data <end-entity-cert-cms(binary)>
|        |     +--rw name <string>
|        +--rw name <string>
|        +--rw private-key-format? <identityref>
|        +--rw (private-key-type)
|        |  +--:(cleartext-private-key)
|        |     +--rw cleartext-private-key? <binary>
|        +--rw public-key? <binary>
|        +--rw public-key-format? <identityref>
+--ro yangcore:notification-log
|  +--ro notification-log-record*
|     +--ro notification
|     +--ro timestamp <date-and-time(string)>
+--rw yangcore:plugins
|  +--rw plugin* [name]
|     +--rw functions
|     |  +--rw function* [name]
|     |     +--rw name <string>
|     +--rw name <string>
+--rw yangcore:preferences
|  +--rw accounting
|  +--rw authentication
|  |  +--rw internal-authenticator!
|  |     +--rw passwords-allowed!
|  |        +--rw aging-timeout!
|  |        |  +--rw amount? <uint16>
|  |        |  +--rw units? <enumeration>
|  |        +--rw is-required? <boolean>
|  +--rw authorization
|  +--rw sztpd:bootstrapping
|  |  +--rw onboarding-supported? <boolean>
|  +--rw general
|  |  +--rw email-address? <string>
|  |  +--rw hostname? <host(union)>
|  +--rw outbound-interactions
|     +--rw relay-audit-log-record-callout? <leafref>
|     +--rw relay-notification-log-record-callout? <leafref>
|     +--rw sztpd:relay-progress-report-callout? <leafref>
+--rw sztpd:responses
|  +--rw onboarding-response* [name]
|  |  +--rw name <string>
|  |  +--rw onboarding-information
|  |     +--rw boot-image? <leafref>
|  |     +--rw configuration? <leafref>
|  |     +--rw post-configuration-script? <leafref>
|  |     +--rw pre-configuration-script? <leafref>
|  +--rw redirect-response* [name]
|     +--rw name <string>
|     +--rw redirect-information
|        +--rw bootstrap-server# <leafref>
+--rw ietf-restconf-server:restconf-server
|  +--rw listen!
|     +--rw endpoints
|        +--rw endpoint* [name]
|           +--rw name <string>
|           +--rw (transport)
|           |  +--:(http-over-tcp)
|           |  |  +--rw http-over-tcp
```



```
|           |  |     +--rw external-endpoint!
|           |  |     |  +--rw address <host(union)>
|           |  |     |  +--rw port? <port-number(uint16)>
|           |  |     +--rw http-server-parameters
|           |  |     |  +--rw client-authentication!
|           |  |     |  +--rw server-name? <string>
|           |  |     +--rw restconf-server-parameters
|           |  |     |  +--rw client-identity-mappings
|           |  |     |     +--rw cert-to-name* [id]
|           |  |     |        +--rw fingerprint? <tls-fingerprint(string)>
|           |  |     |        +--rw id <uint32>
|           |  |     |        +--rw map-type <identityref>
|           |  |     |        +--rw name <string>
|           |  |     +--rw tcp-server-parameters
|           |  |        +--rw local-bind* [local-address]
|           |  |           +--rw local-address <ip-address(union)>
|           |  |           +--rw local-port? <port-number(uint16)>
|           |  +--:(http-over-tls)
|           |     +--rw http-over-tls
|           |        +--rw http-server-parameters
|           |        |  +--rw client-authentication!
|           |        |  +--rw server-name? <string>
|           |        +--rw restconf-server-parameters
|           |        |  +--rw client-identity-mappings
|           |        |     +--rw cert-to-name* [id]
|           |        |        +--rw fingerprint? <tls-fingerprint(string)>
|           |        |        +--rw id <uint32>
|           |        |        +--rw map-type <identityref>
|           |        |        +--rw name <string>
|           |        +--rw tcp-server-parameters
|           |        |  +--rw local-bind* [local-address]
|           |        |     +--rw local-address <ip-address(union)>
|           |        |     +--rw local-port? <port-number(uint16)>
|           |        +--rw tls-server-parameters
|           |           +--rw client-authentication!
|           |           |  +--rw ca-certs!
|           |           |  |  +--rw (inline-or-truststore)
|           |           |  |     +--:(central-truststore)
|           |           |  |        +--rw central-truststore-reference? <central-certificate-bag-ref(leafre\
f)>
|           |           |  +--rw ee-certs!
|           |           |     +--rw (inline-or-truststore)
|           |           |        +--:(central-truststore)
|           |           |           +--rw central-truststore-reference? <central-certificate-bag-ref(leafre\
f)>
|           |           +--rw server-identity
|           |              +--rw (auth-type)
|           |                 +--:(certificate)
|           |                    +--rw certificate
|           |                       +--rw (inline-or-keystore)
|           |                          +--:(central-keystore)
|           |                             +--rw central-keystore-reference
|           |                                +--rw asymmetric-key? <central-asymmetric-key-ref(leafref)>
|           |                                +--rw certificate? <leafref>
|           +--rw yangcore:use-for? <identityref>
+--rw ietf-truststore:truststore
|  +--rw certificate-bags
|     +--rw certificate-bag* [name]
|        +--rw certificate* [name]
|        |  +--rw cert-data <trust-anchor-cert-cms(binary)>
|        |  +--rw name <string>
|        +--rw description? <string>
|        +--rw name <string>
+--rw yangcore:users
|  +--rw user* [login]
|     +--rw authentication
|     |  +--rw password-based
|     |     +--rw password? <crypt-hash(string)>
|     |     +--ro password-last-modified? <date-and-time(string)>
|     +--rw authorization
|     |  +--rw (auth-type)
|     |     +--:(unrestricted)
|     |        +--rw unrestricted? <empty>
|     +--rw email-address <string>
|     +--rw fullname? <string>
|     +--rw login <string>
|     +--rw preferences
|     +---n user-password-aging
|     |  +--ro expiration-date <date-and-time(string)>
|     |  +--ro user <string>
|     +---n user-password-expired
|        +--ro expiration-date <date-and-time(string)>
|        +--ro user <string>
+--ro ietf-yang-library:yang-library
   +--ro content-id <string>
   +--ro datastore* [name]
   |  +--ro name <datastore-ref(identityref)>
```



```
|  +--ro schema <leafref>
+--ro module-set* [name]
|  +--ro import-only-module* [name revision]
|  |  +--ro location* <uri(string)>
|  |  +--ro name <yang-identifier(string)>
|  |  +--ro namespace <uri(string)>
|  |  +--ro revision <union>
|  |  +--ro submodule* [name]
|  |     +--ro location* <uri(string)>
|  |     +--ro name <yang-identifier(string)>
|  |     +--ro revision? <revision-identifier(string)>
|  +--ro module* [name]
|  |  +--ro deviation* <leafref>
|  |  +--ro feature* <yang-identifier(string)>
|  |  +--ro location* <uri(string)>
|  |  +--ro name <yang-identifier(string)>
|  |  +--ro namespace <uri(string)>
|  |  +--ro revision? <revision-identifier(string)>
|  |  +--ro submodule* [name]
|  |     +--ro location* <uri(string)>
|  |     +--ro name <yang-identifier(string)>
|  |     +--ro revision? <revision-identifier(string)>
|  +--ro name <string>
+--ro schema* [name]
   +--ro module-set* <leafref>
   +--ro name <string>
```

5.4 NBI Top-level Nodes

SZTPD's NBI builds on top of YANGcore's NBI. Please see "NBI Top-level Nodes" in the YANGcore User Guide for details.

This section presents the new top-level nodes SZTPD adds to the NBI, providing both detailed information on each in turn. Each node's name uses the naming convention "<module-name>:<top-level-node-name>". The top-level "conveyed-information" node is presented in parts. The following subsections are presented in sorted order.

5.4.1 /ietf-yang-library:yang-library

The top-level 'ietf-yang-library:yang-library' node, which is read-only (see "ro" in the NBI Complete Tree Diagram) defines all of the YANG modules implemented by the Northbound interface. The YANG Library is formally defined in RFC 8525.

5.4.2 /sztpd:conveyed-information

The top-level 'sztpd:conveyed-information' node groups data described as being "conveyed information" (see Section 3.1 in RFC 8572).

This information includes:

- boot-images
- bootstrapping servers
- configurations
- scripts

These are convered in the following subsections.

5.4.2.1 /sztpd:conveyed-information/boot-images

The top-level 'sztpd:conveyed-information' node contains the 'boot-images' node, which is read-write (see "rw" in the tree diagram) contains a list of "boot-image" nodes.

The 'boot-image' nodes are used to describe a boot-image to return to a bootstrapping device. Each boot-image object must specify its name and version, and may specify download URIs and hash values to verify the downloaded boot-image file.

Boot-image nodes are referenced by the 'responses' structure described in /sztpd:responses. Specifically, the 'boot-image' nodes are referenced by the 'onboarding-information' node, and thus used when returning onboarding responses.

The following tree diagram illustrates the NBI used for configuring 'boot-image' records.

```
+--rw conveyed-information
   +--rw boot-images
      +--rw boot-image* [name]
         +--rw name                    string
         +--rw os-name                 string
         +--rw os-version              string
         +--rw download-uri*           inet:uri
         +--rw image-verification* [hash-algorithm]
            +--rw hash-algorithm       identityref
            +--rw hash-value           yang:hex-string
```

5.4.2.2 /sztpd:conveyed-information/bootstrap-servers The top-level 'sztpd:conveyed-information' node contains the 'bootstrap-servers' node, which is read-write (see "rw" in the tree diagram) contains a list of "bootstrap-server" nodes.

The 'bootstrap-server' nodes describe a bootstrap server to return to a bootstrapping device. Each bootstrap server node can specify its address, port, and trust anchor certificate.

Bootstrap-server nodes are referenced by the 'responses' structure described in /sztpd:responses. Specifically, the 'bootstrap-server' nodes are referenced by the 'redirect-information' node, and thus used when returning redirect responses.

The following tree diagram illustrates the NBI used for configuring "bootstrap-server" records.

```
+--rw conveyed-information
   +--rw bootstrap-servers
      +--rw bootstrap-server* [name]
         +--rw name             string
         +--rw address          inet:host
         +--rw port?            inet:port-number
         +--rw trust-anchor?    ct:trust-anchor-cert-cms
```

5.4.2.3 /sztpd:conveyed-information/configurations

The top-level 'sztpd:conveyed-information' node contains the 'configurations' node, which is read-write (see "rw" in the tree diagram) contains a list of "configuration" nodes.

The 'configuration' nodes describe a configuration to return to a bootstrapping device. Each configuration node can specify the configuraton itself as well as how it should be handled (merged vs replaced). The configuration is any format supported by the bootstrapping device. While possible the same configuration may be used by more than one device node, it is more likely to have a distinct configuration per device[2].

Configuration node are referenced by the 'responses' structure described in /sztpd:responses. Specifically, the 'configuration' nodes are referenced by the 'onboarding-information' node, and thus used when returning onboarding responses.

The following tree diagram illustrates the NBI used for configuring 'configuration' records.

```
+--rw conveyed-information
   +--rw configurations
      +--rw configuration* [name]
         +--rw name            string
         +--rw handling?       enumeration
         +--rw config-data?    binary
         +--rw media-type?     enumeration
```

5.4.2.4 /sztpd:conveyed-information/scripts

The top-level 'sztpd:conveyed-information' node contains the 'scripts' node, which is read-write (see "rw" in the tree diagram) contains a list of "script" nodes.

[2]For this reason, it may seen unnecessary to have configurations configured outside the device node, however this was done intentionally so as to support a templating mechanism someday, whereby the configuration is 99% the same for all devices, containing only device-specific variations (e.g., the device's IP address).

The 'script' node holds both 'pre-configuration-script' objects and 'post-configuration-script' objects. Each script can be any executable supported by the device.

Script nodes are referenced by the 'responses' structure described in /sztpd:responses. Specifically, the script nodes are referenced by the 'onboarding-information' node, and thus used when returning onboarding responses.

The following tree diagram illustrates the NBI used for configuring 'script' records.

```
  +--rw conveyed-information
     +--rw scripts
        +--rw script* [name]
           +--rw name    string
           +--rw type    enumeration
           +--rw code    sztp-info:script
```

5.4.3 /sztpd:device-types

The top-level 'sztpd:device-types' node, which is read-write (see "rw" in the tree diagram) contains a list of "device-type" nodes.

The 'device-type' node describes a type of device, and is used when replying to a bootstrapping device. Each device-type node can indicate if bootstrapping devices must present an identity certificate (e.g., an IDevID) and, if so, which certificate in the "/ietf:truststore" is used to validate the device's certificate.

Each device-type node can also indicate how to test the ownership of a device, which is implemented as a dynamic callout.

Device-type nodes are referenced by device nodes, as seen in /sztpd:devices.

The following tree diagram illustrates the NBI used for configuring 'device' records.

```
============================== NOTE: '\' line wrapping per RFC 8792 ==============================

  +--rw device-types
     +--rw device-type* [name]
        +--rw name                        string
        +--rw identity-certificates!
        |  +--rw verification
        |  |  +--rw central-truststore-reference {ts:certificates}?
        |  |     +--rw certificate-bag?   ts:central-certificate-bag-ref {central-truststore-supported,cert\
ificates}?
        |  |     +--rw certificate?       ts:central-certificate-ref {central-truststore-supported,certific\
ates}?
        |  +--rw serial-number-extraction?   identityref
        +--rw ownership-authorization!
           +--rw dynamic-callout
              +--rw reference?   -> /yangcore:dynamic-callouts/dynamic-callout/name
```

5.4.4 /sztpd:devices

The top-level 'sztpd:devices' node, which is read-write (see "rw" in the tree diagram) contains a list of "device" nodes.

The 'device' node is used to configure the bootstrapping response for a device, and also holds bootstrapping-log for each device[3]. The bootstrapping log is containing in the 'device' object so that storage used for it will be released when the device object is deleted.

Each device must have a 'serial-number' and a 'device-type'. Further, the device object must either set 'activiation-code' or the 'device-type' it points to must set the 'identity-certificates' node. The device object also has a 'response-manager' containing a list of 'matched-response' rules, which are processed in order until a match is found. Note that if the matched-response contains no 'match-criteria' then it automatically matches. Otherwise, all of match-criteria must evaluate to true for a match to occur. Once a match occurs, the associated 'response' to selected.

Currently, responses are limited to just returning the "conveyed information" artifact, described in Section 3.1 or RFC 8752. SZTPD doesn't yet return to bootstrapping devices either the "owner certificate"

[3]The per-device bootstrapping-log is distinct from the [audit log][YANGcore / Audit Log]. The bootstrapping logs captures to full interaction with the bootstrapping device, whereas the audit log captures the authentication of inbound requests to SZTPD.

or "ownership voucher" artifacts. This means that bootstrapping devices must "trust" the SZTPD bootstrap server, since it does not ever return signed artifacts.

The following tree diagram illustrates the NBI used for configuring 'device' records.

```
============================== NOTE: '\' line wrapping per RFC 8792 ==============================

  +--rw devices
     +--rw device* [serial-number]
        +--rw serial-number           string
        +--rw device-type             -> /device-types/device-type/name
        +--rw activation-code?        ianach:crypt-hash
        +--rw response-manager
        |  +--rw matched-response* [name]
        |     +--rw name                string
        |     +--rw match-criteria!
        |     |  +--rw match* [key]
        |     |     +--rw key               string
        |     |     +--rw not?              empty
        |     |     +--rw (test-type)
        |     |        +--:(present)
        |     |        |  +--rw present?    empty
        |     |        +--:(value)
        |     |        |  +--rw value?      string
        |     |        +--:(regex)
        |     |           +--rw regex?      string
        |     +--rw response
        |        +--rw reporting-level?              enumeration
        |        +--rw (response-handler)
        |           +--:(none)
        |           |  +--rw none?                   empty
        |           +--:(managed-response)
        |              +--rw conveyed-information
        |                 +--rw (conveyed-information-handler)
        |                    +--:(dynamic-callout)
        |                    |  +--rw via-dynamic-callout
        |                    |     +--rw reference    -> /yangcore:dynamic-callouts/dynamic-callout/name
        |                    +--:(redirect-information)
        |                    |  +--rw via-redirect-response
        |                    |     +--rw reference    -> /responses/redirect-response/name
        |                    +--:(onboarding-information)
        |                       +--rw via-onboarding-response
        |                          +--rw reference                 -> /responses/onboarding-response/name
        |                          +--rw config-template-params
        |                             +--rw config-template-param* [key]
        |                                +--rw key       string
        |                                +--rw value     string
        +--ro bootstrapping-log
        |  +--ro bootstrapping-log-record* []
        |     +--ro timestamp          yang:date-and-time
        |     +--ro source-ip          inet:ip-address
        |     +--ro method             string
        |     +--ro path               string
        |     +--ro return-code        uint16
        |     +--ro error-returned?    <anydata>
        |     +--ro event-details
        |        +--ro (event-type)
        |           +--:(get-bootstrapping-data-event)
        |           |  +--ro get-bootstrapping-data-event
        |           |     +--ro passed-input          <anydata>
        |           |     +--ro selected-response?    union
        |           |     +--ro response-details
        |           |        +--ro (response-type)
        |           |           +--:(managed-response)
        |           |              +--ro managed-response
        |           |                 +--ro conveyed-information
        |           |                    +--ro (conveyed-information-handler)
        |           |                       +--:(dynamic-callout)
        |           |                       |  +--ro via-dynamic-callout
        |           |                       |     +--ro (result-type)?
        |           |                       |        +--:(no-callout-configured)
        |           |                       |        |  +--ro no-callout-configured?    empty
        |           |                       |        +--:(callout-configured)
        |           |                       |           +--ro name                          string
        |           |                       |           +--ro rpc-supported                 identityref
        |           |                       |           +--ro callout-type                  enumeration
        |           |                       |           +--ro (function-or-webhook)?
        |           |                       |              +--:(function)
        |           |                       |                 +--ro function-details
        |           |                       |                 |  +--ro plugin      string
        |           |                       |                 |  +--ro function    string
        |           |                       |                 +--ro function-results
        |           |                       |                    +--ro (exit-status)
        |           |                       |                       +--:(exception-thrown)
        |           |                       |                       |  +--ro exception-thrown?   string
        |           |                       |                       +--:(exited-normally)
```



```
        |            |                        |              |           +--ro exited-normally?     string
        |            |                        |              +--:(webhook) {yangcore:webhook-based-callouts-i\
mplemented}?
        |            |                        |                 +--ro webhook-details
        |            |                        |                 |  +--ro num-webhooks-configured     uint8
        |            |                        |                 +--ro webhook-results
        |            |                        |                    +--ro webhook* []
        |            |                        |                       +--ro name                        string
        |            |                        |                       +--ro uri
inet:uri
        |            |                        |                       +--ro (result-type)?
        |            |                        |                          +--:(connection-error)
        |            |                        |                          |  +--ro connection-error?     string
        |            |                        |                          +--:(http-status-code)
        |            |                        |                             +--ro http-status-code      string
        |            |                        +--:(redirect-information)
        |            |                        |  +--ro via-redirect-response
        |            |                        |     +--ro referenced-definition?    string
        |            |                        +--:(onboarding-information)
        |            |                           +--ro via-onboarding-response
        |            |                              +--ro referenced-definition?    string
        |            +--:(report-progress-event)
        |               +--ro report-progress-event
        |                  +--ro passed-input       <anydata>
        |                  +--ro dynamic-callout
        |                     +--ro (result-type)?
        |                        +--:(no-callout-configured)
        |                        |  +--ro no-callout-configured?    empty
        |                        +--:(callout-configured)
        |                           +--ro name                       string
        |                           +--ro rpc-supported              identityref
        |                           +--ro callout-type               enumeration
        |                           +--ro (function-or-webhook)?
        |                              +--:(function)
        |                              |  +--ro function-details
        |                              |  |  +--ro plugin       string
        |                              |  |  +--ro function     string
        |                              |  +--ro function-results
        |                              |     +--ro (exit-status)
        |                              |        +--:(exception-thrown)
        |                              |        |  +--ro exception-thrown?     string
        |                              |        +--:(exited-normally)
        |                              |           +--ro exited-normally?      string
        |                              +--:(webhook) {yangcore:webhook-based-callouts-implemented}?
        |                                 +--ro webhook-details
        |                                 |  +--ro num-webhooks-configured      uint8
        |                                 +--ro webhook-results
        |                                    +--ro webhook* []
        |                                       +--ro name                        string
        |                                       +--ro uri                         inet:uri
        |                                       +--ro (result-type)?
        |                                          +--:(connection-error)
        |                                          |  +--ro connection-error?     string
        |                                          +--:(http-status-code)
        |                                             +--ro http-status-code      string
        +--ro lifecycle-statistics
           +--ro nbi-access-stats
           |  +--ro created                yang:date-and-time
           |  +--ro num-times-modified     uint16
           |  +--ro last-modified          yang:date-and-time
           +--ro sbi-access-stats
              +--ro num-times-accessed     uint16
              +--ro first-accessed         yang:date-and-time
              +--ro last-accessed          yang:date-and-time
```

5.4.5 /sztpd:responses

The top-level 'sztpd:responses' node, which is read-write (see "rw" in the tree diagram) contains both a list of "redirect-response" nodes and a list of onboarding-response" nodes. These responses are returned to bootstrapping devices as follows:

- The 'redirect-response' node references an ordered list of [bootstrap servers][SZTPD / Bootstrap Servers].
- The 'onboarding-response' node references a [boot-image][SZTPD / Boot Images] object, a [pre-configuration-script][SZTPD / Scripts] object, a [configuration][SZTPD / Configurations] object, and a [post-configuration-script][SZTPD / Scripts] object.

Device nodes reference the the "responses" nodes as described in /sztpd:devices. Specifically, the 'response' nodes are referenced by the device's "response-manager".

The following tree diagram illustrates the NBI used for configuring 'responses' records.

```
+--rw responses
   +--rw redirect-response* [name]
   |  +--rw name                      string
   |  +--rw redirect-information
   |     +--rw bootstrap-server*   -> /conveyed-information/bootstrap-servers/bootstrap-server/name
   +--rw onboarding-response* [name]
      +--rw name                       string
      +--rw onboarding-information
         +--rw boot-image?                  -> /conveyed-information/boot-images/boot-image/name
         +--rw pre-configuration-script?    -> /conveyed-information/scripts/script/name
         +--rw configuration?               -> /conveyed-information/configurations/configuration/name
         +--rw post-configuration-script?   -> /conveyed-information/scripts/script/name
```

6 Outbound API Details

SZTPD inherits the "Outbound" interface from YANGcore. Please see the "Outbound API Details" section in the YANGcore User Manual for details. SZTPD adds to YANGcore's Outbound inteface.

- SZTPD adds additional Dynamic Callouts
- SZTPD adds additional Notifications

6.1 Dynamic Callouts

SZTPD inherits YANGcore's plugin-based dynamic callout mechanism. Please read the "Dynamic Callouts" section in the YANGcore User Guide for more details. SZTPD adds the following dynamic callouts:

```
rpcs
  +-- rpc get-conveyed-information
  +-- rpc relay-progress-report
  +-- rpc verify-device-ownership
```

6.1.1 The "get-conveyed-information" Dynamic Callout

A dynamic callout implementing the "get-conveyed-information" RPC is used to have SZTPD dynamically generate a response for a bootstrapping device. This ability is used to, e.g., support RFC 9646.

To configure SZTPD to dynamically obtain conveyed-information from an external system, a device's "response-manager" configuration must have a "matched-response" of type "conveyed-information" with a "dynamic-callout" having the "rpc-supported" value "sztpd-rpcs:get-conveyed-information".

6.1.1.1 Tree Diagram

Following is the tree diagram for the "get-conveyed-information" dynamic callout:

```
   +---x get-conveyed-information
      +---w input
      |  +---w serial-number           string
      |  +---w source-ip-address       inet:ip-address
      |  +---w identity-certificate?   ct:cms
      |  +---w from-device?            <anydata>
      |  +---w opaque?                 <anydata>
      +--ro output
         +--ro conveyed-information    <anydata>
```


6.1.1.2 Example Usage

Here is a Python plugin that implements a function that can be used for the "get-conveyed-information" RPC.

```
import json

def handle_get_conveyed_info_request(input, opaque):

    # show input
    print(json.dumps({"input": input, "opaque": opaque}, indent=3))

    # hardcode response
    return {
     "conveyed-information": {
        'ietf-sztp-conveyed-info:onboarding-information': {
          'boot-image': {
            'os-name': 'VendorOS',
            'os-version': '19.2r1b6',
            'download-uri': ['https://example.com/path/to/image/file'],
            'image-verification': [
              {
                'hash-algorithm': 'sha-256',
                'hash-value': 'ba:ec:cf:a5:67:82:b4:10:77:c6:67:a6:22:ab:7d:50:04:a7:8b:8f:0e:db:02:8b:f4:75:55:fb:c1:13:b2:3
              }
            ]
          }
        }
      }
    }
```

When executed with an example conveyed-information request, this function has output[4]

```
{
   "input": {
      "serial-number": "my-serial-number",
      "source-ip-address": "127.0.0.1",
      "from-device": {
         "ietf-sztp-bootstrap-server:input": {
            "ietf-sztp-csr:p10-csr": "BASE64VALUE="
         }
      }
   },
   "opaque": null
}
```

6.1.2 The "relay-progress-report" Dynamic Callout

A dynamic callout implementing the "relay-progress-report" RPC is used to relay progress reports received from bootstrapping devices.

To configure SZTPD to relay progress reports, the "/yangcore:preferences/outbound-interactions/sztpd:relay-progress-report-callout" node must point to a "/yangcore:dynamic-callouts/dynamic-callout" having the "rpc-supported" value "sztpd-rpcs:relay-progress-report".

6.1.2.1 Tree Diagram

Following is the tree diagram for the "relay-progress-report" dynamic callout:

```
   +---x relay-progress-report
      +---w input
         +---w serial-number           string
         +---w source-ip-address       inet:ip-address
         +---w identity-certificate?   ct:cms
         +---w from-device             <anydata>
         +---w opaque?                 <anydata>
```

[4]The "BASE64VALUE=" values are not real; they are used in this example only for readability.

6.1.2.2 Example Usage

Here is a Python plugin that implements a function that can be used for the "relay-progress-report" RPC.

```
import json

def relay_progress_report_function(input, opaque):

    # show input
    print(json.dumps({"input": input, "opaque": opaque}, indent=3))

    # no 'output' defined
```


When executed with example input intended to simulate a device sending its "bootstrap-complete" message, this function has output[5].

```
{
   "input": {
      "serial-number": "my-serial-number",
      "source-ip-address": "127.0.0.1",
      "from-device": {
         "ietf-sztp-bootstrap-server:input": {
            "progress-type": "bootstrap-complete",
            "message": "message sent via XML",
            "ssh-host-keys": {
               "ssh-host-key": [
                  {
                     "algorithm": "ssh-rsa",
                     "key-data": "BASE64VALUE="
                  },
                  {
                     "algorithm": "rsa-sha2-256",
                     "key-data": "BASE64VALUE="
                  }
               ]
            },
            "trust-anchor-certs": {
               "trust-anchor-cert": [
                  "BASE64VALUE="
               ]
            }
         }
      }
   },
   "opaque": null
}
```

6.1.3 The "verify-device-ownership" Dynamic Callout

A dynamic callout implementing the "verify-device-ownership-record" RPC is used to verify the ownership for devices. Note that this callout primarily exists so that, in a multitenancy deployment, the host-level can ensure that the tenants only configure devices (i.e., serial-numbers) that belong to them.

To configure SZTPD to verify device ownership, the "ownership-authorization" node under the top-level node "/sztpd:device-types" must point to a "dynamic-callout" having the "rpc-supported" value "sztpd:verify-device-ownership".

6.1.3.1 Tree Diagram

Following is the tree diagram for the "verify-device-ownership" dynamic callout:

```
   +---x verify-device-ownership
      +---w input
      |  +---w tenant           string
      |  +---w serial-number*   string
      +---ro output
         +---ro verification-results
            +---ro verification-result* [serial-number]
               +---ro serial-number    string
               +---ro result           enumeration
```

[5]The "BASE64VALUE=" values are not real; they are used in this example only for readability.

6.1.3.2 Example Usage

Here is a Python plugin that implements a function that can be used for the "verify-device-ownership" RPC.

```
import json

def authenticate_device_ownership(input: dict, opaque: dict):

    # show input
    print(json.dumps({"input": input, "opaque": opaque}, indent=3))

    # hardcode response
    return {
        "verification-results": {
            "verification-result": [
                {
                    "serial-number": "sn-111",
                    "result": "failure"
                },
                {
                    "serial-number": "sn-222",
                    "result": "success"
                },
                {
                    "serial-number": "sn-333",
                    "result": "success"
                }
            ]
        }
    }
```

When executed with an example ownership-verification request, this function has the following output:

```
{
   "input": {
     "tenant": "not-applicable",
     "serial-number": [
       "serial_number": ["sn-111", "sn-222", "sn-333"],
     ]
   },
   "opaque": null
}
```

6.2 Notifications

SZTPD inherits notifications from YANGcore. Please see the "Notifications" section in the YANGcore User Guide for more details

SZTPD doesn't implement any additional notifications yet.

7 Southbound API Details

The Southbound Interface, which is added by the SZTPD application on top of YANGcore, implements the "bootstrap server" defined in RFC 8572. This interface is presented when the 'yangcore:use-for' value is the value "sztpd:rfc8572-interface".

This section primarily illustrates usage for developers implementing support for SZTP (RFC 8572) to network element devices[6]. These examples assume SZTPD's SBI is listening on port 9090.

Since the Southbound interface is a RESTCONF server, some of the API fulfills that interface contract. Thus the SBI is a superset of the API defined by RFC 8040. Use of the additional RESTCONF API endpoints should be limited to tests.

7.1 SBI YANG Library

For any RESTCONF server, its YANG Library is the heart of the API, as it unambiguously identifies what API the server implements. SZTPD's SBI YANG Library, presented below, is unique and specific to bootstrapping. The SBI's YANG Library shares nothing in common with either the NBI Yang Library or the YANG Library used by YANGCore.


```
{
    "ietf-yang-library:yang-library": {
        "module-set": [
            {
                "name": "shared-module-set",
                "module": [
                    {
                        "name": "ietf-datastores",
                        "revision": "2018-02-14",
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-yang-datastores"
                    },
                    {
                        "name": "ietf-yang-library",
                        "revision": "2019-01-04",
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-yang-library"
                    },
                    {
                        "name": "ietf-sztp-bootstrap-server",
                        "revision": "2019-04-30",
                        "feature": ["onboarding-server"],
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-sztp-bootstrap-server"
                    },
                    {
                        "name": "ietf-yang-structure-ext",
                        "revision": "2020-06-17",
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-yang-structure-ext"
                    },
                    {
                        "name": "ietf-crypto-types",
                        "revision": "2024-10-10",
                        "feature": [],
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-crypto-types"
                    },
                    {
                        "name": "ietf-ztp-types",
                        "revision": "2024-10-10",
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-ztp-types"
                    },
                    {
                        "name": "ietf-sztp-csr",
                        "revision": "2024-10-10",
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-sztp-csr"
                    },
                    {
                        "name": "yangcore-yl",
                        "revision": "2025-02-15",
                        "namespace": "https://watsen.net/yangcore-yl"
                    }
                ],
                "import-only-module": [
                    {
                        "name": "ietf-yang-types",
                        "revision": "2013-07-15",
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-yang-types"
```


[6]To developers of the client-side bootstrapping/SZTP client. Please be advised that the code, upon receiving the CMS structure from SZTPD, will need to examine the a "content-type" OID. Whilst SZTPD currently returns only OIDs "id_ct_sztpConveyedInfoJSON" and "id_ct_sztpConveyedInfoXML", it will return OIDs "id-envelopedData" and "id-signedData" in a future release. See RFC 8572 for more information about these OIDs


```
                    },
                    {
                        "name": "ietf-inet-types",
                        "revision": "2013-07-15",
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-inet-types"
                    },
                    {
                        "name": "ietf-netconf-acm",
                        "revision": "2018-02-14",
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-netconf-acm"
                    }
                ]
            }
        ],
        "schema": [
            {
                "name": "shared-schema",
                "module-set": [
                    "shared-module-set"
                ]
            }
        ],
        "datastore": [
            {
                "name": "ietf-datastores:operational",
                "schema": "shared-schema"
            },
            {
                "name": "ietf-datastores:running",
                "schema": "shared-schema"
            }
        ],
        "content-id": "TBD"
    }
}
```

7.2 SBI YANG Modules

The SBI YANG Library defines the API as a collection of YANG modules. The most exact description of the API is had by reviewing all of the referenced YANG modules directly, in conjuction with the "features" statements in the SBI YANG Library.

There are five YANG modules in the SBI. Ideally this guide would include these modules, but they are a combined total of over fifteeen thousand lines, which is too much to include here.

In lieu of including the YANG modules in this guide, the reader may find these YANG modules in the "yang" directory in both the 'yangcore' and 'sztpd' packages. Where these files are located varies by Python installation. The following command find the directory the YANG files are located for the "yangcore" and "sztpd" packages:

```
python -c 'import importlib.resources as resources; print(resources.files("yangcore") / "yang")'
python -c 'import importlib.resources as resources; print(resources.files("sztpd") / "yang")'
```

For example: "/Users/kent/.pyenv/versions/3.12.8/lib/python3.12/site-packages/yangcore/yang" and "/Users/kent/.pyenv/versions/3.12.8/lib/python3.12/site-packages/sztpd/yang".

7.3 SBI Complete Tree Diagram

This section presents the complete YANG tree diagram for SZTPD's Southbound API. This tree diagram is generated using the YANG library presented in SBI YANG Library. This diagram is useful as it presents the entire API one diagram.

```
+---x ietf-sztp-bootstrap-server:get-bootstrapping-data
|  +---ro input
|  |  +---ro hw-model? <string>
|  |  +---ro (ietf-sztp-csr:msg-type)?
|  |  |  +--:(csr)
|  |  |  |  +---ro (csr-type)
|  |  |  |     +--:(cmc-csr)
|  |  |  |     |  +---ro cmc-csr? <binary>
|  |  |  |     +--:(cmp-csr)
|  |  |  |     |  +---ro cmp-csr? <binary>
|  |  |  |     +--:(p10-csr)
|  |  |  |        +---ro p10-csr? <p10-csr(binary)>
|  |  |  +--:(csr-support)
|  |  |     +---ro csr-support
|  |  |        +---ro csr-generation
|  |  |        |  +---ro supported-formats
|  |  |        |     +---ro format-identifier* <identityref>
|  |  |        +---ro key-generation!
|  |  |           +---ro supported-algorithms
|  |  |              +---ro algorithm-identifier* <binary>
|  |  +---ro nonce? <binary>
|  |  +---ro os-name? <string>
|  |  +---ro os-version? <string>
|  |  +---ro signed-data-preferred? <empty>
|  +---ro output
|     +---ro conveyed-information <cms(binary)>
|     +---ro owner-certificate? <cms(binary)>
|     +---ro ownership-voucher? <cms(binary)>
|     +---ro reporting-level? <enumeration>
+---x ietf-sztp-bootstrap-server:report-progress
|  +---ro input
|  |  +---ro message? <string>
|  |  +---ro progress-type <enumeration>
|  |  +---ro ssh-host-keys
|  |  |  +---ro ssh-host-key*
|  |  |     +---ro algorithm <string>
|  |  |     +---ro key-data <binary>
|  |  +---ro trust-anchor-certs
|  |     +---ro trust-anchor-cert* <cms(binary)>
|  +---ro output
+---ro ietf-yang-library:yang-library
   +---ro content-id <string>
   +---ro datastore* [name]
   |  +---ro name <datastore-ref(identityref)>
   |  +---ro schema <leafref>
   +---ro module-set* [name]
   |  +---ro import-only-module* [name revision]
   |  |  +---ro location* <uri(string)>
   |  |  +---ro name <yang-identifier(string)>
   |  |  +---ro namespace <uri(string)>
   |  |  +---ro revision <union>
   |  |  +---ro submodule* [name]
   |  |     +---ro location* <uri(string)>
   |  |     +---ro name <yang-identifier(string)>
   |  |     +---ro revision? <revision-identifier(string)>
   |  +---ro module* [name]
   |  |  +---ro deviation* <leafref>
   |  |  +---ro feature* <yang-identifier(string)>
   |  |  +---ro location* <uri(string)>
   |  |  +---ro name <yang-identifier(string)>
   |  |  +---ro namespace <uri(string)>
   |  |  +---ro revision? <revision-identifier(string)>
   |  |  +---ro submodule* [name]
   |  |     +---ro location* <uri(string)>
   |  |     +---ro name <yang-identifier(string)>
   |  |     +---ro revision? <revision-identifier(string)>
   |  +---ro name <string>
   +---ro schema* [name]
      +---ro module-set* <leafref>
      +---ro name <string>
```


7.4 SBI Top-level Nodes

The SBI has three top-level nodes, which are described in this section. Two of the nodes are RPCs and one node is operational state. The SBI does not define any configuration nodes. Notable about the SBI is that all of its implemented modules are defined by the IETF (i.e., none by Watsen Networks). The following subsections are presented in sorted order.

7.4.1 /ietf-sztp-bootstrap-server:get-bootstrapping-data

The top-level "ietf-sztp-bootstrap-server:get-bootstrapping-data" node is an RPC (i.e., defined by a YANG "rpc" statement). This RPC is used by bootstrapping devices to obtain bootstrapping data. The "get-bootstrapping-data" RPC is formally defined in RFC 8572.

7.4.2 /ietf-sztp-bootstrap-server:report-progress

The top-level "ietf-sztp-bootstrap-server:report-progress" node is an RPC (i.e., defined by a YANG "rpc" statement). This RPC is used by bootstrapping devices to share bootstrapping progress with the bootstrapping server. The "report-progress" RPC is formally defined in RFC 8572.

7.4.3 /ietf-yang-library:yang-library

The top-level "ietf-yang-library:yang-library" node, which is read-only (see "ro" in the SBI Complete Tree Diagram) defines all of the YANG modules implemented by the Northbound interface. The YANG Library is formally defined in RFC 8525.

7.5 Determining the SBI's Host-meta

Since the Southbound interface is a RESTCONF server, the same command used by the Northbound Interface applies. See Fetching Host-meta for details. The NBI's host-meta is the same as SBI's host-meta.

7.6 Determining the SBI's RESTCONF Root Resource

Since the Southbound interface is a RESTCONF server, the same command used by the Northbound Interface applies. See Fetching the RESTCONF Root Resource for details. The NBI's root resource is the same as SBI's root resource.

7.7 Determining the SBI's Supported Encodings

Since the Southbound interface is a RESTCONF server, the same command used by the Northbound Interface applies. See [Fetching the Encodings Supported] for details. SZTPD's SBI supports both "application/yang-data+json" and "application/yang-data+xml".

7.8 Determining the SBI's YANG-library Version

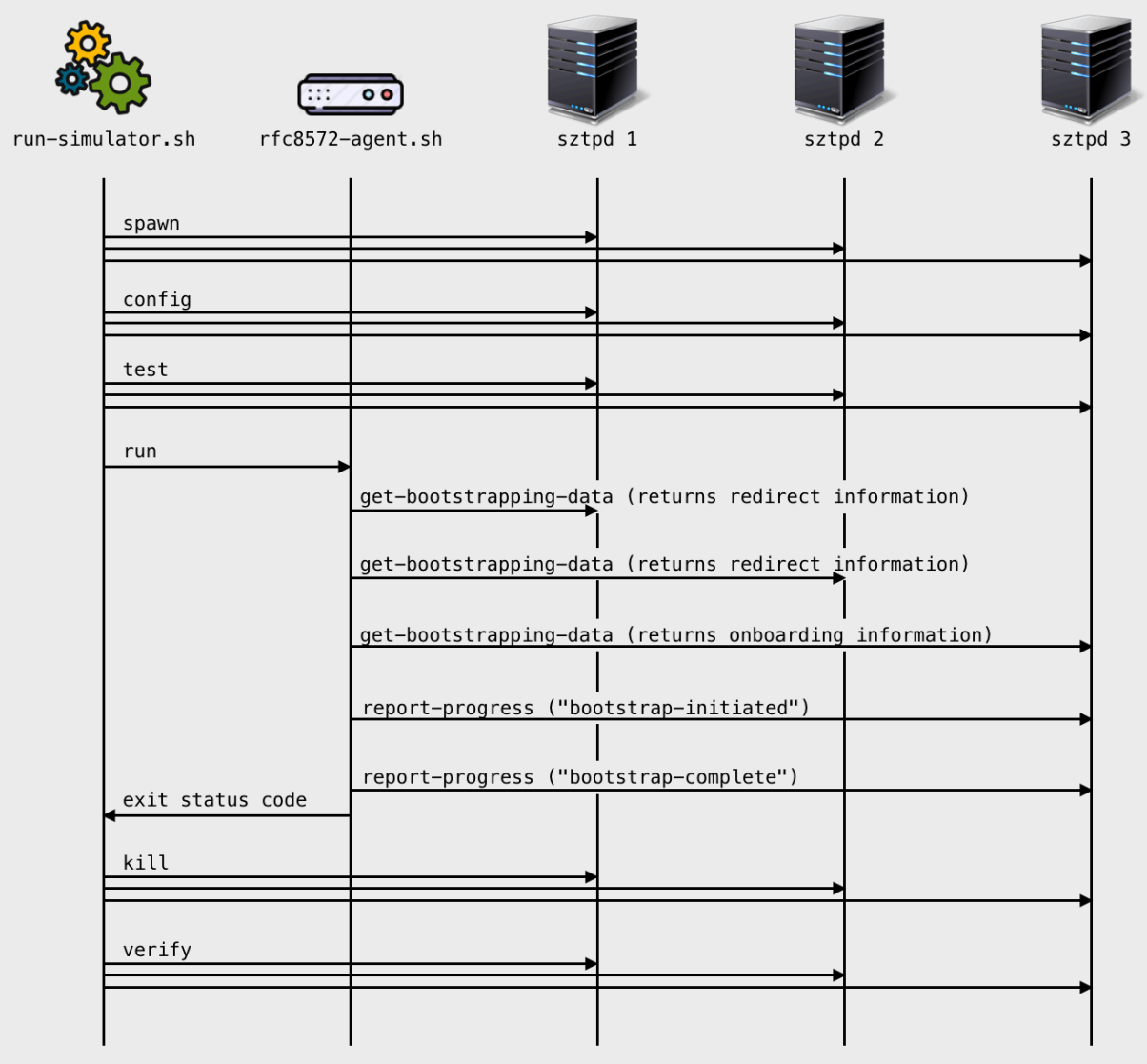
Since the Southbound interface is a RESTCONF server, the same command used by the Northbound Interface applies. See Fetching the YANG Library for details. SZTPD's SBI "implements" a different number of YANG modules. The SBI's YANG Library is below:


```
{
    "ietf-yang-library:yang-library": {
        "module-set": [
            {
                "name": "shared-module-set",
                "module": [
                    {
                        "name": "ietf-sztp-bootstrap-server",
                        "revision": "2019-04-30",
                        "feature": ["onboarding-server"],
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-sztp-bootstrap-server",
                    },
                    {
                        "name": "ietf-yang-structure-ext",
                        "revision": "2020-06-17",
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-yang-structure-ext",
                    },
                    {
                        "name": "ietf-crypto-types",
                        "revision": "2024-10-10",
                        "feature": [],
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-crypto-types",
                    },
                    {
                        "name": "ietf-ztp-types",
                        "revision": "2024-10-10",
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-ztp-types",
                    },
                    {
                        "name": "ietf-sztp-csr",
                        "revision": "2024-10-10",
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-sztp-csr",
                    }
                ],
                "import-only-module": [
                    {
                        "name": "ietf-yang-types",
                        "revision": "2013-07-15",
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-yang-types"
                    },
                    {
                        "name": "ietf-inet-types",
                        "revision": "2013-07-15",
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-inet-types"
                    },
                    {
                        "name": "ietf-netconf-acm",
                        "revision": "2018-02-14",
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-netconf-acm",
                    },
                    {
                        "name": "ietf-datastores",
                        "revision": "2018-02-14",
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-yang-datastores",
                    },
                    {
                        "name": "ietf-yang-library",
                        "revision": "2019-01-04",
                        "namespace": "urn:ietf:params:xml:ns:yang:ietf-yang-library",
                    }
                ]
            }
        ],
        "schema": [
            {
                "name": "shared-schema",
                "module-set": [
                    "shared-module-set"
                ]
            }
        ],
        "datastore": [
            {
                "name": "ietf-datastores:operational",
                "schema": "shared-schema"
            },
            {
                "name": "ietf-datastores:running",
                "schema": "shared-schema"
            }
        ],
        "content-id": "TBD"
    }
}
```


8 Simulator

A simulator has been developed, primarily to aid in the development of the RFC 8572 compliant bootstrapping agent, though inspection of its source code is helpful towards understand how to use SZTPD's [Northbound Interface] (e.g., its use of the 'curl' command line utility).

8.1 Overview



The simulator (i.e., the script 'run-sztpd-test.sh') performs the following steps:

1. Start and initialize three SZTPD servers for the following roles:

Instance	Purpose
1	A trusted bootstrap server to redirect the device to instance #2.
2	An initially untrusted bootstrap server to redirect the device to instance #3.
3	An initially untrusted bootstrap server to give the device its onboarding data.

   Each SZTP server is an instance of the 'sztpd' process.

2. Start an instance of a demo RFC 8572 agent (the 'rfc8572-simulator.sh' script).

   Parameters are passed into the agent providing necessary values (e.g., the "serial-number" value).

3. Wait for the agent to complete and then shutdown the three SZTPD instances.

   No trace on the filesystem is retained as:

   1. the 'run-sztpd-test.sh' and 'rfc8572-simulator.sh' scripts cleans up after themselves.
   2. the 'sztpd' processes are run using the "in-memory" database[7].

8.2 Dependencies

Successful execution of the requires a few packages to be installed.

- A UNIX-based system. (untested on Windows)
- Bash (the Bourne-Again SHell, any recent version)
- OpenSSL (both libraries and the command-line utility)
- Python (version 3.7 or greater)
- The following Python modules (all installed as dependencies to SZTPD)
- Curl (the 'curl' utility, many times installed by OS)
- SZTPD

8.3 Downloading

The simulator can be downloaded here: https://watsen.net/support.

8.4 Unarchiving

Unarchive the TGZ in a local directory, for example:

```
$ tar -xzvf ./sztpd-simulator-0.0.8.tgz

  // This command creates a directory called "sztpd-simulator".
```

[7] For information about the in-memory database, please See the "In-memory Database" section in the Installation Guide.

Inside the "sztpd-simulator" directory are a number of files and directories:

Resource	Purpose
run-simulator.sh	The script that starts 3 SZTPD servers and an RFC 8572 agent.
rfc8572-agent.sh	A simple 'curl' based agent that simulates an RFC 8572 agent.
templates/	A directory containing config used to init the SZTP instances.
pki/	A directory containing code to initialize the PKI for the demo.
xrd/	A directory containing files to validates the "host-meta" XRD.

8.5 Customizing Variables

There is little need to customize the simulator scripts before running.

The most likely customization is to adjust the "PYTHON" and "PIP" variables, located at the top of the "run-simulator.sh" and "rfc8572-agent.sh" files. For instance, if Python is installed as "python3.8", then the values should be set as follows:

```
PYTHON=python3.8
PIP=pip3.8
```

Another possible customization is to adjust the ports that the various 'sztpd' instances will open, for instance, to deconflict from a port already in use. To adjust the ports used, edit the top of the "run-sztpd-test.sh" and modify the various "PORT" values listed at the top of that file.

8.6 Initializing the PKI

The simulator creates a distinct PKI for each endpoint. As each SZTPD instance (in this demo) has two endpoints (an NBI and and SBI) and there are three SZTPD instances, a total of six certificate chains are created. Each certificate chain contains four certificates (a root certificate, two intermediate certificates, and an end-entity certificate), thus a total of Twenty-four certificates are created in total.

Whilst the certificates could be dynamically generated for each execution of the simulator, for multiple executions, having the PKI created already saves time.

Assuming the current directory is the "sztpd-simulator" directory, the following commands:

```
$ cd pki
$ make pki
$ cd ..
```

If ever needed, 'make clean' can be used to return the directory hierarchy to its original form.

8.7 Running

Assuming the current directory is the "sztpd-simulator" directory, the following command will run the simulator:

```
$ ./run-sztpd-test.sh
```

Each time the simulator runs, it first tests if the PKI has been initialized. If the PKI has not been initialized, the simulator exits with the message:

```
PKI needs to be initialized first.
  - e.g., 'cd pki; make pki; cd ..;'
```


Otherwise the simulator runs without any other prompts, producing output such as:

```
Creating instances...
  ^— Creating SZTPD instance 1...okay.  (SZTPD instance 1 running with PID 22089)
  ^— Creating SZTPD instance 2...okay.  (SZTPD instance 2 running with PID 220155)
  ^— Creating SZTPD instance 3...okay.  (SZTPD instance 3 running with PID 22091)

Giving servers a couple seconds to startup...

Configuring instances...
  ^— Configuring SZTPD instance 1...
  ^— Configuring SZTPD instance 2...
  ^— Configuring SZTPD instance 3...

Giving servers a couple seconds to open their ports...

Testing instances...
  ^— Testing SZTPD instance 1
  ^— Testing SZTPD instance 2
  ^— Testing SZTPD instance 3

Running simulator...
  ^— Getting bootstrapping data...
  ^— Processing bootstrapping data...
  ^— Processing redirect information...
  ^— Getting bootstrapping data from next server...
  ^— Processing bootstrapping data...
  ^— Processing redirect information...
  ^— Getting bootstrapping data from next server...
  ^— Processing bootstrapping data...
  ^— Processing onboarding information...
  ^— Bootstrap complete.

Killing 'sztpd' instances...
  ^— Sending SIGTERM to instance 1 (PID 22089)
  ^— Sending SIGTERM to instance 2 (PID 22090)
  ^— Sending SIGTERM to instance 3 (PID 22091)

Giving servers a couple seconds to shutdown...

Verifying instances are killed...
  ^— Verifying SZTPD instance 1 killed...okay.
  ^— Verifying SZTPD instance 2 killed...okay.
  ^— Verifying SZTPD instance 3 killed...okay.

All done!
```

8.8 Cleaning Up

The simulator makes effort to clean up after itself. For instance, all files are created in a temporary directory. However, currently, if the simulator experiences an unexpected error, it may exit without first shutting down the 'sztpd' instances.

If the 'sztpd' instances are not removed, they can be removed manually. For instance:

```
$ ps | grep sztpd
23816 ttys024    0:02.01 python sztpd sqlite:///:memory:
23817 ttys024    0:02.01 python sztpd sqlite:///:memory:
23818 ttys024    0:02.01 python sztpd sqlite:///:memory:

$ kill 23816
$ kill 23817
$ kill 23818
```

