

IEEE Standard for Streaming Telemetry Transport Protocol (STTP)

IEEE Power and Energy Society

Developed by the
Power System Communications and Cybersecurity Committee

IEEE Std 2664™-2024

STANDARDS

IEEE Standard for Streaming Telemetry Transport Protocol (STTP)

Developed by

Power System Communications and Cybersecurity Committee
of the
IEEE Power and Energy Society

Approved 21 March 2024

IEEE SA Standards Board

Abstract: Streaming Telemetry Transport Protocol (STTP) is defined in this standard. This protocol was initially designed for streaming power system data over Internet protocol (IP) communication systems. It is focused on streaming real-time measurements but can be used with any type of continuously flowing data. It features built-in security and lossless data compression options for efficient transport. It specifies data types and transport control methods using a publish-subscribe architecture for controlled signal-level data access. It uses standard security and transport options, so it is easy to maintain and update. It is highly configurable for time precision, compression level, and packet sizes to optimize the stream within data and transport limitations. It has been proven and refined in several demonstration projects and production environments.

Keywords: IEEE 2664™, IEEE C37.118™, PDC, phasor, phasor data concentrator, phasor measurement unit, PMU, real-time data, streaming data, streaming telemetry, STTP, synchrophasor, time-series data

The Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York, NY 10016-5997, USA

Copyright © 2024 by The Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 2 August 2024. Printed in the United States of America.

IEEE and POSIX are registered trademarks in the U.S. Patent & Trademark Office, owned by The Institute of Electrical and Electronics Engineers, Incorporated.

The Microsoft® .NET logo is a registered trademark of Microsoft Corporation in the United States and/or other countries.

UNIX® is a registered trademark of The Open Group.

PDF: ISBN 979-8-8557-0967-4 STD27119
Print: ISBN 978-8-8557-0968-1 STDPD27119

IEEE prohibits discrimination, harassment, and bullying.

For more information, visit <http://www.ieee.org/web/aboutus/whatis/policies/p9-26.html>.

No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without the prior written permission of the publisher.

Important Notices and Disclaimers Concerning IEEE Standards Documents

IEEE Standards documents are made available for use subject to important notices and legal disclaimers. These notices and disclaimers, or a reference to this page (<https://standards.ieee.org/ipr/disclaimers.html>), appear in all IEEE standards and may be found under the heading “Important Notices and Disclaimers Concerning IEEE Standards Documents.”

Notice and Disclaimer of Liability Concerning the Use of IEEE Standards Documents

IEEE Standards documents are developed within IEEE Societies and subcommittees of IEEE Standards Association (IEEE SA) Board of Governors. IEEE develops its standards through an accredited consensus development process, which brings together volunteers representing varied viewpoints and interests to achieve the final product. IEEE standards are documents developed by volunteers with scientific, academic, and industry-based expertise in technical working groups. Volunteers involved in technical working groups are not necessarily members of IEEE or IEEE SA and participate without compensation from IEEE. While IEEE administers the process and establishes rules to promote fairness in the consensus development process, IEEE does not independently evaluate, test, or verify the accuracy of any of the information or the soundness of any judgments contained in its standards.

IEEE makes no warranties or representations concerning its standards, and expressly disclaims all warranties, express or implied, concerning all standards, including but not limited to the warranties of merchantability, fitness for a particular purpose and non-infringement. IEEE Standards documents do not guarantee safety, security, health, or environmental protection, or compliance with law, or guarantee against interference with or from other devices or networks. In addition, IEEE does not warrant or represent that the use of the material contained in its standards is free from patent infringement. IEEE Standards documents are supplied “AS IS” and “WITH ALL FAULTS.”

Use of an IEEE standard is wholly voluntary. The existence of an IEEE standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard.

In publishing and making its standards available, IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity, nor is IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing any IEEE Standards document should rely upon their own independent judgment in the exercise of reasonable care in any given circumstances or, as appropriate, seek the advice of a competent professional in determining the appropriateness of a given IEEE standard.

IN NO EVENT SHALL IEEE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO: THE NEED TO PROCURE SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE PUBLICATION, USE OF, OR RELIANCE UPON ANY STANDARD, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE AND REGARDLESS OF WHETHER SUCH DAMAGE WAS FORESEEABLE.

Translations

The IEEE consensus balloting process involves the review of documents in English only. In the event that an IEEE standard is translated, only the English language version published by IEEE is the approved IEEE standard.

Use by artificial intelligence systems

In no event shall material in any IEEE Standards documents be used for the purpose of creating, training, enhancing, developing, maintaining, or contributing to any artificial intelligence systems without the express, written consent of IEEE SA in advance. “Artificial intelligence” refers to any software, application, or other system that uses artificial intelligence, machine learning, or similar technologies, to analyze, train, process, or generate content. Requests for consent can be submitted using the Contact Us form.

Official statements

A statement, written or oral, that is not processed in accordance with the IEEE SA Standards Board Operations Manual is not, and shall not be considered or inferred to be, the official position of IEEE or any of its committees and shall not be considered to be, or be relied upon as, a formal position of IEEE or IEEE SA. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that the presenter’s views should be considered the personal views of that individual rather than the formal position of IEEE, IEEE SA, the Standards Committee, or the Working Group. Statements made by volunteers may not represent the formal position of their employer(s) or affiliation(s). News releases about IEEE standards issued by entities other than IEEE SA should be considered the view of the entity issuing the release rather than the formal position of IEEE or IEEE SA.

Comments on standards

Comments for revision of IEEE Standards documents are welcome from any interested party, regardless of membership affiliation with IEEE or IEEE SA. However, **IEEE does not provide interpretations, consulting information, or advice pertaining to IEEE Standards documents.**

Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Since IEEE standards represent a consensus of concerned interests, it is important that any responses to comments and questions also receive the concurrence of a balance of interests. For this reason, IEEE and the members of its Societies and subcommittees of the IEEE SA Board of Governors are not able to provide an instant response to comments or questions, except in those cases where the matter has previously been addressed. For the same reason, IEEE does not respond to interpretation requests. Any person who would like to participate in evaluating comments or revisions to an IEEE standard is welcome to join the relevant IEEE SA working group. You can indicate interest in a working group using the Interests tab in the Manage Profile & Interests area of the [IEEE SA myProject system](#).¹ An IEEE Account is needed to access the application.

Comments on standards should be submitted using the [Contact Us](#) form.²

¹ Available at: <https://development.standards.ieee.org/myproject-web/public/view.html#landing>.

² Available at: <https://standards.ieee.org/about/contact/>.

Laws and regulations

Users of IEEE Standards documents should consult all applicable laws and regulations. Compliance with the provisions of any IEEE Standards document does not constitute compliance to any applicable regulatory requirements. Implementers of the standard are responsible for observing or referring to the applicable regulatory requirements. IEEE does not, by the publication of its standards, intend to urge action that is not in compliance with applicable laws, and these documents may not be construed as doing so.

Data privacy

Users of IEEE Standards documents should evaluate the standards for considerations of data privacy and data ownership in the context of assessing and using the standards in compliance with applicable laws and regulations.

Copyrights

IEEE draft and approved standards are copyrighted by IEEE under U.S. and international copyright laws. They are made available by IEEE and are adopted for a wide variety of both public and private uses. These include both use by reference, in laws and regulations, and use in private self-regulation, standardization, and the promotion of engineering practices and methods. By making these documents available for use and adoption by public authorities and private users, neither IEEE nor its licensors waive any rights in copyright to the documents.

Photocopies

Subject to payment of the appropriate licensing fees, IEEE will grant users a limited, non-exclusive license to photocopy portions of any individual standard for company or organizational internal use or individual, non-commercial use only. To arrange for payment of licensing fees, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400; <https://www.copyright.com/>. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Updating of IEEE Standards documents

Users of IEEE Standards documents should be aware that these documents may be superseded at any time by the issuance of new editions or may be amended from time to time through the issuance of amendments, corrigenda, or errata. An official IEEE document at any point in time consists of the current edition of the document together with any amendments, corrigenda, or errata then in effect.

Every IEEE standard is subjected to review at least every 10 years. When a document is more than 10 years old and has not undergone a revision process, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE standard.

In order to determine whether a given document is the current edition and whether it has been amended through the issuance of amendments, corrigenda, or errata, visit [IEEE Xplore](#) or [contact IEEE](#).³ For more information about the IEEE SA or IEEE's standards development process, visit the IEEE SA Website.

Errata

Errata, if any, for all IEEE standards can be accessed on the [IEEE SA Website](#).⁴ Search for standard number and year of approval to access the web page of the published standard. Errata links are located under the Additional Resources Details section. Errata are also available in [IEEE Xplore](#). Users are encouraged to periodically check for errata.

Patents

IEEE standards are developed in compliance with the [IEEE SA Patent Policy](#).⁵

Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken by the IEEE with respect to the existence or validity of any patent rights in connection therewith. If a patent holder or patent applicant has filed a statement of assurance via an Accepted Letter of Assurance, then the statement is listed on the IEEE SA Website at <https://standards.ieee.org/about/sasb/patcom/patents.html>. Letters of Assurance may indicate whether the Submitter is willing or unwilling to grant licenses under patent rights without compensation or under reasonable rates, with reasonable terms and conditions that are demonstrably free of any unfair discrimination to applicants desiring to obtain such licenses.

Essential Patent Claims may exist for which a Letter of Assurance has not been received. The IEEE is not responsible for identifying Essential Patent Claims for which a license may be required, for conducting inquiries into the legal validity or scope of Patents Claims, or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance, if any, or in any licensing agreements are reasonable or non-discriminatory. Users of this standard are expressly advised that determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their own responsibility. Further information may be obtained from the IEEE Standards Association.

IMPORTANT NOTICE

Technologies, application of technologies, and recommended procedures in various industries evolve over time. The IEEE standards development process allows participants to review developments in industries, technologies, and practices, and to determine what, if any, updates should be made to the IEEE standard. During this evolution, the technologies and recommendations in IEEE standards may be implemented in ways not foreseen during the standard's development. IEEE standards development activities consider research and information presented to the standards development group in developing any safety recommendations. Other information about safety practices, changes in technology or technology implementation, or impact by peripheral systems also may be pertinent to safety considerations during implementation of the standard. Implementers and users of IEEE Standards documents are responsible for determining and complying with all appropriate safety, security, environmental, health, data privacy, and interference protection practices and all applicable laws and regulations.

³ Available at: <https://ieeexplore.ieee.org/browse/standards/collection/ieee>.

⁴ Available at: <https://standards.ieee.org/standard/index.html>.

⁵ Available at: <https://standards.ieee.org/about/sasb/patcom/materials.html>.

Participants

At the time this IEEE standard was completed, the P10 (STTP) Working Group had the following membership:

Kenneth Martin, *Chair*
Scott Mix, *Vice Chair*

Steven Blair
Dan Brancaccio
Ritchie Carroll
Matt Donnelly

Jean-Sebastien Gagnon *
Vasudev Gharpure
Allen Goldstein *

Dinesh Gurusinghe *
Yi Hu
Sakis Meliopoulos *
Mahendra Patel

*Corresponding Member

The following members of the individual balloting committee voted on this standard. Balloters may have voted for approval, disapproval, or abstention.

Mihaela Albu
Steven Bezner
Paul Cardinal
James Carroll
Thomas Dahlin
Matt Donnelly
Kenneth Fodero
James Formea
Carl Fredericks
Jean-Sebastien Gagnon
Gopal Gajjar

Vasudev Gharpure
Werner Hoelzl
Yi Hu
Richard Jackson
Anthony Johnson
Piotr Karocki
Chung-Yiu Lam
Greg Luri
Kenneth Martin
Scott Mix
Rajesh Murthy

Bruce Muschlitz
Mfanasibili Nkonyane
James O'Brien
Bansi Patel
Howard Penrose
Farnoosh Rahmatian
Eric Udren
Benton Vandiver
S. Venkata
John Vergis
Mina Yousef

When the IEEE-SA Standards Board approved this standard on 21 March 2024, it had the following membership:

David J. Law, *Chair*
Jon Walter Rosdahl, *Vice Chair*
Gary Hoffman, *Past Chair*
Alpesh Shah, *Secretary*

Sara R. Biyabani
Ted Burse
Stephen Dukes
Doug Edwards
J. Travis Griffith
Guido R. Hiertz
Ronald W. Hotchkiss

Hao Hu
Yousef Kimiagar
Joseph L. Koepfinger*
Howard Li
Xiaohui Liu
John Haiying Lu
Kevin W. Lu
Hiroshi Mano

Paul Nikolich
Robby Robson
Lei Wang
F. Keith Waters
Sha Wei
Philip B. Winston
Don Wright

*Member Emeritus

Introduction

This introduction is not part of IEEE Std 2664-2024, IEEE Standard for Streaming Telemetry Transport Protocol (STTP).

In 2014, the US Department of Energy (DOE) funded the Secure Information Exchange Gateway (SIEGate) Project (DE-OE0000536) to handle an expanded set of requirements for the secure exchange of the data necessary to support real-time grid operations. A new protocol called the Gateway Exchange Protocol (GEP) was developed to handle real-time data exchange including synchrophasor data, SCADA data, and file-based data. The GEP protocol allowed SIEGate to control access to data at an individual measurement, or point, level. In addition, GEP was focused on control-center-to-control-center communications and was designed for very high-volume data flows.

In 2017, this protocol was further extended and improved under the DOE Advanced Synchrophasor Protocol (ASP) Development and Demonstration Project (DE-OE0000859), with the goal of standardizing the protocol. The ASP project involved 26 vendor, utility, and university collaborators with an objective to develop, document, and demonstrate the new communications protocol. The protocol, optimized for the demands of transporting high volumes of continuous streaming data and related metadata, is not limited to synchrophasor data. The protocol allows for the transmission of any information that can be represented longitudinally, e.g., time-series data, and was named the Streaming Telemetry Transport Protocol (STTP) to emphasize its generalized applicability to the transfer of streaming data.

Comparison to existing synchrophasor protocols

Existing synchrophasor protocols, e.g., IEEE C37.118.2™-2011 [B9] and IEC 61850:2023 [B7] (also known as IEC 61850 Ed 2.1), are frame-based where measurement data is packaged together for the same timestamp into frames for transmission.⁶ While efficient and effective for smaller data volumes, when used at scales now common at many utilities involving many hundreds of phasor measurement units (PMUs), the frame-based nature of these protocols present network design and operational challenges. Large frame sizes can contribute to data loss due to fragmentation and network bottlenecks. Additionally, the commonly used User Datagram Protocol/Internet Protocol (UDP/IP) has a frame size ceiling of 65K bytes, which limits the amount of information that can be exchanged between two parties in a single message.

In contrast, STTP is a measurement-based protocol without a fixed configuration. Data is sent in individual measurement units, each with its own timestamp, identification, and status flags. Several measurement units may be packed into a network packet for efficient communication but within maximum transmission unit (MTU) limits to reduce fragmentation. With STTP, loss of a single network packet over UDP only results in the loss of the measurement samples in that packet rather than the loss of the full frame of data, as is the case with existing synchrophasor protocols. STTP streams are built from individual measurements, allowing control of information at the measurement level so that only the requested data is sent, eliminating unneeded data that may otherwise be included in frame-based measurement sets.

Comparison with IEEE Std C37.118.2-2011

STTP is a generic data streaming protocol for time-series information. It is not restricted to transferring synchrophasor data or any other particular type of time-series data. However, the capability of STTP to effectively handle and manage high-volume data flows makes it a good alternative to IEEE Std C37.118.2-2011 [B9] for large scale synchrophasor communications.

⁶ The numbers in brackets correspond to those of the bibliography in Annex A.

The key new features of STTP are as follows:

1. STTP is built as a publish-subscribe protocol rather than a fixed client-server.
 - a. A subscriber connects with a publisher and establishes a subscription of data to be sent.
 - b. All data, metadata, and exchange control are accomplished in-band through the protocol. The only out-of-band information required is the IP addresses and/or server names and data access controls.
2. Data reporting is centered on individual measurements rather than preconfigured measurement sets.
 - a. Each measurement is identified with a 128-bit universally unique identifier (UUID); the subscriber specifies which measurements to send.
 - b. Sets of measurements for subscription can be specified using **Filter Expressions** against publisher defined tabular metadata using syntax that is analogous to Structured Query Language (SQL).
 - c. Measurements are sent in a compact unit containing the measurement value, a timestamp, a quality status, and the ID.
3. STTP includes strategies to optimize data bandwidth requirements.
 - a. The size of the compact measurement structure can be reduced during runtime using offsets.
 - b. Optional lossless measurement data compression can be applied.
 - c. STTP can build packets of several measurement units while keeping the packet sizes small enough to help avoid fragmentation,
4. STTP includes security enhancements, among these are subscriber authentication, data encryption, access control, and key management.
5. STTP uses an improved timestamp format with much greater range and with high-precision options for synchronized waveform data.
6. STTP includes the ability to play back historical data or retrieve data files, in addition to handling real-time data streams.
7. STTP includes a NOTIFY scheme, which allows critical messages to be published with confirmed delivery by the subscriber.

Contents

1. Overview	12
1.1 Scope	12
1.2 Purpose	12
1.3 Need for this standard.....	12
1.4 Conventions	12
1.5 Word usage.....	13
2. Normative references.....	14
3. Definitions, acronyms, and abbreviations	14
3.1 Definitions	14
3.2 Acronyms and abbreviations	15
4. STTP feature summary.....	17
4.1 Introduction	17
4.2 Command and response.....	19
4.3 Data units and packeting.....	21
4.4 Built-in security	22
4.5 Measurement centered	23
5. Command and response messages.....	23
5.1 Message introduction.....	23
5.2 STTP commands.....	24
5.3 STTP responses	27
5.4 STTP command messages	29
5.5 STTP response messages.....	38
6. Communication system operation	49
6.1 STTP communications.....	49
6.2 Connection establishment.....	49
7. Cybersecurity.....	52
7.1 Introduction	52
7.2 Key lifetimes.....	52
7.3 Comparison of VPN and TLS.....	53
7.4 Data integrity	54
8. Data access control.....	54
9. Metadata	55
9.1 Introduction	55
9.2 Metadata format.....	55
9.3 Types of metadata.....	56
9.4 Primary metadata.....	56
9.5 Signal selection metadata	58
10. STTP timestamps.....	60
10.1 Timestamp overview	60
10.2 Timestamp resolution	60
10.3 STTP timestamp settings	61
10.4 STTP timestamp compatibility	62

11. Data compression	63
11.1 Introduction	63
11.2 STTP TCP compression	64
11.3 STTP UDP compression	64
Annex A (informative) Bibliography	66
Annex B (normative) Time Series Special Compression (TSSC)	68
B.1 TSSC Overview	68
B.2 Example code	70
Annex C (normative) FILTER EXPRESSION format	71
C.1 Filtering syntax	71
C.2 FILTER EXPRESSION functions	75
C.3 FILTER EXPRESSION grammar for syntax parsing	77
Annex D (informative) API reference	78
D.1 Recommended subscriber API elements	78
D.2 Recommended publisher API elements	82
Annex E (normative) Default quality and state flags	85
Annex F (informative) Recommended metadata schema for synchrophasor data	88
F.1 General	88
F.2 Primary metadata schema for synchrophasor data	88
F.3 Common fields for the ActiveMeasurements table	91
F.4 Example of METADATA REFRESH response	92

IEEE Standard for Streaming Telemetry Transport Protocol (STTP)

1. Overview

1.1 Scope

This standard defines a protocol with built-in security and lossless data compression options for efficient transport of streaming power system data over Internet Protocol (IP) communication systems. It specifies data and control methods and uses a publish-subscribe architecture for controlled signal-level data access. This standard incorporates open source material hosted on the IEEE SA Open Source platform.

1.2 Purpose

The purpose for this standard is to provide a streaming data protocol meeting the specific requirements of power systems, which include reliability, security, scalability, bandwidth management, and flexible configuration. Existing protocols do not address all of these requirements. This protocol is intended to supplement or replace existing protocols used for synchrophasor data.

1.3 Need for this standard

Existing power-system protocols used for streaming data exchange are not well suited for large data sets or high-volume data flows. Shortcomings of existing protocols include message fragmentation, which can result in excessive data loss or poor bandwidth utilization; lack of support for on-line subscription for specific measurement values; lack of built-in security options; and options that do not fully support the needs of emerging applications, e.g., continuous point-on-wave data. As power systems increase real-time streaming data exchanges to support operation and analysis applications by different classes of users, a scalable protocol that addresses these communication requirements is needed.

1.4 Conventions

“0x” indicates a hexadecimal value, and all other values not specified to be hexadecimal are decimal (base-10). For example, the hexadecimal number 0x41 is equal to the decimal value 65.

Bits are referenced starting at zero. For example, within a byte numbering goes from the least-significant bit (00) to the most-significant bit (07).

Items written in Arial **SMALL CAPS** refer to the names given to STTP commands and responses.

Items written in Roman **ALL CAPS** refer to field names in an STTP command or response message.

Items written in fixed-pitch fonts represent code, XML, or expressions.

Items written in Roman UpperCamelCase represent parameter key names or metadata table or field names.

Items written in Roman **SMALL CAPS** refer to context reference strings returned with **FAILED** responses to commands.

Items written in Arial *UPPERCAMELCASESMALLCAPS* italics refer to API methods, callbacks, or classes.

Items written in Roman *lowerCamelCase* italics are API formal parameters.

Items written in Roman **UPPERCAMELCASESMALLCAPS** refer to measurement format or quality flag names,

1.5 Word usage

The word *shall* indicates mandatory requirements strictly to be followed in order to conform to the standard and from which no deviation is permitted (shall equals is required to).^{7, 8}

The word *should* indicates that among several possibilities one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required (should equals is recommended that).

The word *may* is used to indicate a course of action permissible within the limits of the standard (may equals is permitted to).

The word *can* is used for statements of possibility and capability, whether material, physical, or causal (can equals is able to).

The word “reserved” indicates:

- If used in an assignment of values to an enumeration or an attribute that the values indicated are reserved for use in future editions of this standard and shall not be used for any other purpose.
- If used in a field of a message or frame that the field is reserved for use in future editions of this standard. The field shall be present in the message or frame, with the size specified. No interpretation of reserved fields is to be made for this edition of this standard, and the fields shall not be used for any other purpose.

⁷The use of the word *must* is deprecated and cannot be used when stating mandatory requirements; *must* is used only to describe unavoidable situations.

⁸The use of *will* is deprecated and cannot be used when stating mandatory requirements; *will* is only used in statements of fact.

2. Normative references

The following referenced documents are indispensable for the application of this document (i.e., they must be understood and used, so each referenced document is cited in text and its relationship to this document is explained). For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments or corrigenda) applies.

IEEE/ASTM SI 10™-2016, American National Standard for Metric Practice.^{9, 10}

ISO/IEC 10646:2020, Information technology—Universal coded character set (UCS).¹¹

RFC 1952, GZIP file format specification, Version 4.3, Deutsch, P., May 1996.¹²

RFC 3339, Date and Time on the Internet: Timestamps, Klyne, G., and C. Newman, July 2002.

RFC 8446, The Transport Layer Security (TLS) Protocol, Version 1.3, Rescorla, E., August 2018.

RFC 9562, Universally Unique IDentifiers (UUID)s, Davis, K., Peabody, B., and Leach, P., May 2024.

3. Definitions, acronyms, and abbreviations

3.1 Definitions

For the purposes of this document, the following terms and definitions apply. The *IEEE Standards Dictionary Online* should be consulted for terms not defined in this clause.¹³

channel: The logical component of a connection that allows commands or data to be passed between a publisher and a subscriber.

command: A method invocation, directive in a header, or other statement that can be acted on by the receiving party.

command channel: Channel used to establish and manage subscriptions.

connection: The association between a publisher and a subscriber. A connection consists of one or more channels used to convey commands or data.

data point: A real-value number representing single measurement or calculated value.

payload: Information that is included in an STTP message in addition to the defined message fields.

publisher: A node of a data system that provides data on approved request to other nodes.

⁹ IEEE publications are available from The Institute of Electrical and Electronics Engineers (<http://standards.ieee.org/>).

¹⁰ The IEEE standards or products referred to in Clause 2 are trademarks owned by The Institute of Electrical and Electronics Engineers, Incorporated.

¹¹ ISO/IEC publications are available from the International Organization for Standardization (<https://www.iso.org/>) and the American National Standards Institute (<https://www.ansi.org/>).

¹² RFC documents are available at <https://www.rfc-editor.org/search/>.

¹³ *IEEE Standards Dictionary Online* is available at: <http://dictionary.ieee.org>. An IEEE Account is required for access to the dictionary, and one can be created at no charge on the dictionary sign-in page.

replay interval: The interval at which historical data is sent from a publisher to a subscriber.

session: An association context that is maintained while two nodes are communicating back and forth in a conversation of some duration.

subscriber: A node of a data system that receives data from a publisher.

subscription channel: The channel used to convey data from a publisher to a subscriber.

3.2 Acronyms and abbreviations

API	application programming interface
Bool	Boolean value
byte	group of 8 bits treated as a single unit
CE	Common Era
datetime	data type representing a specific date and time that can range from 00:00:00 (midnight), January 1, 0001 CE through 11:59:59 P.M., December 31, 9999 CE in the Gregorian calendar
decimal	floating-point number with large significant integral and fractional digits with no round-off errors, e.g., decimal128 in IEEE 754™ format
double	64-bit floating point number, binary64 in IEEE 754 format (functional equivalent to float64)
float	32-bit floating point number, binary32 in IEEE 754 format (functional equivalent to float32)
float32	32-bit floating point number, binary32 in IEEE 754 format
float64	64-bit floating point number, binary64 in IEEE 754 format
GUID	globally unique identifier (functional equivalent to a UUID)
ID	Identifier
IETF	Internet Engineering Task Force
int	signed 64-bit integer using two's complement (functional equivalent to int64)
int32	signed 32-bit integer using two's complement
int64	signed 64-bit integer using two's complement
IP	Internet Protocol
ms	millisecond
μs	microsecond
MSB	most significant bit

MTU	maximum transmission unit
NaN	not-a-number as defined by IEEE Std 754
ns	nanosecond
ps	picosecond
RFC	Request for Comments, used to describe IETF documents
Single	32-bit floating point number, binary32 in IEEE 754 format (functional equivalent to float32)
SQL	Structured Query Language
string	sequence of characters
STTP	Streaming Telemetry Transport Protocol
TCP	Transmission Control Protocol
TLS	Transport Layer Security
TSSC	Time Series Special Compression
UDP	User Datagram Protocol
uint	unsigned 64-bit integer (functional equivalent to uint64)
uint16	unsigned 16-bit integer
uint32	unsigned 32-bit integer
uint64	unsigned 64-bit integer
UTC	coordinated universal time
UTF	Unicode Transformation Format: 8-bit (UTF-8)
UUID	Universally Unique Identifier as defined in RFC 9562 ¹⁴
VPN	virtual private network
X.509	standard format for ITU-T digital certificates
XML	eXtensible Markup Language

¹⁴ Information on references can be found in Clause 2.

4. STTP feature summary

4.1 Introduction

Streaming Telemetry Transport Protocol (STTP) is a measurement centric, publish/subscribe transport protocol that can be used to securely exchange time-series data and synchronize metadata among applications. The protocol supports sending real-time and historical data at full or down-sampled resolutions. When sending historical data, the replay speed can be controlled dynamically for use in visualizations to enable users to see data faster or slower than measured in real time.

Although various commercial and open source protocols are available for the exchange of streaming time-series data, few standards exist. At the time of writing, no known standards exist for general purpose, streaming longitudinal data exchange with the capability of securely multiplexing many thousands of signals at high sampling rates over a single connection. One standard, RFC 3550 [B16], exists for streaming data; however, the protocol explicitly recommends against multiplexing even two streams, e.g., audio and video, over the same connection.¹⁵ The protocol also does not support the exchange of signal-level metadata.

The wire protocol defined by STTP is targeted for packet-based transport protocols, specifically Internet Protocol (IP). STTP implements a publish/subscribe data exchange model using simple commands with a compressed binary serialization of data points. The protocol does not require a predefined or fixed configuration; that is, the identifiable data point values arriving in one DATA PACKET can be different than those arriving in another. Each packet of measurement data consists of a collection of data values where each value is defined by a compact structure containing an identifier, a timestamp or sequence index, a measurement value, and associated asset state or data quality flags.

STTP is implemented using two different data exchanges for communications between the STTP publisher and an STTP subscriber. STTP calls the first the *command channel* and the second the *subscription channel*. In IP based communication, each of these data exchanges is handled by one or more IP endpoints for sending and receiving data (called a *socket*) where the IP transport protocols for these channels can vary based on need. The two most common configurations are: (1) a single Transmission Control Protocol (TCP) (RFC 793 [B15]) transport for both the commands and subscription data, i.e., traffic for both exchanges share the same socket, and (2) a TCP-based command channel with a separate TCP or User Datagram Protocol (UDP) (RFC 791 [B14]) based subscription channel to transfer the subscription data.

The command channel is used to establish the connection between the publisher and subscriber and reliably negotiate session specific required communication, state, and protocol parameters. The command channel is also used to manage authentication and encryption keys with other STTP instances, exchange metadata on available data points, and request specific data points for subscription. The subscription channel is used to send binary encoded packets of data points or BUFFER BLOCKS.

If additional distinct subscriptions are desired, e.g., with different broadcast rates, measurement set, or timestamp resolutions, a separate connection and command channel shall be established for each additional subscription and an additional subscription channel associated to it unless the subscribed data is transferred over the command channel.

These options are shown graphically in the following figures. Figure 1 shows the legend used for the figures.

¹⁵ The numbers in brackets correspond to those of the bibliography in Annex A.

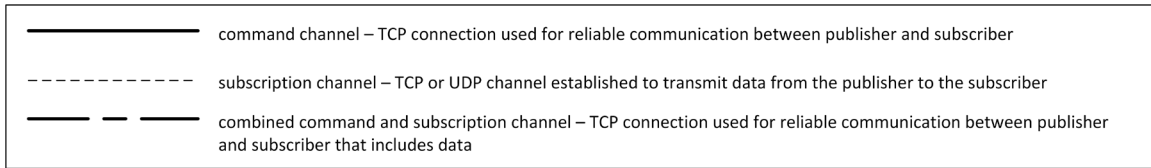


Figure 1—Legend

Figure 2 shows a single TCP command channel and its associated TCP or UDP subscription channel between the publisher and the subscriber. The command channel is used to send commands, receive responses, and manage the subscription, while all the data is transferred across the subscription channel.

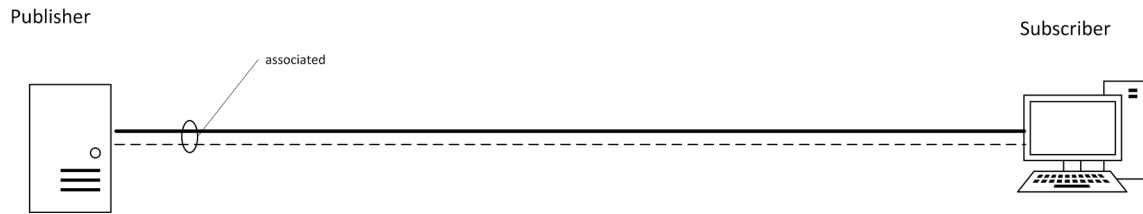


Figure 2—Single command channel and associated subscription channel

Figure 3 shows a single TCP channel between the publisher and subscriber. The single channel is used to both send commands and manage the subscription, as well as transfer the subscribed data.

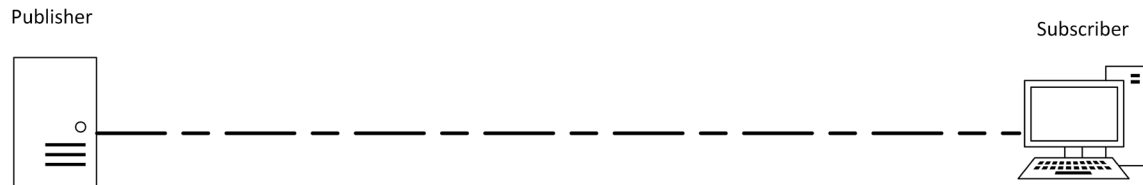


Figure 3—Single combined command and subscription channel

Figure 4 shows multiple command and subscription channels between the publisher and the subscriber. In this example, the subscription channel is separate from the command channel. Each set of command and subscription channels operates independently, and when one of the channels is no longer needed to transfer data, it is closed.

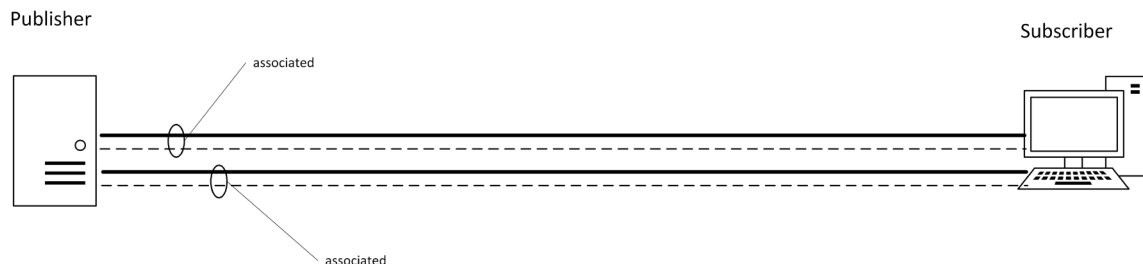


Figure 4—Multiple command and associated subscription channels—1

Figure 5 shows that both a combined command and subscription channel as well as separate command and subscription channels can be used between the publisher and the subscriber at the same time. For example, this scenario may be used when receiving streaming real-time data as well as requesting small sets of historical data.

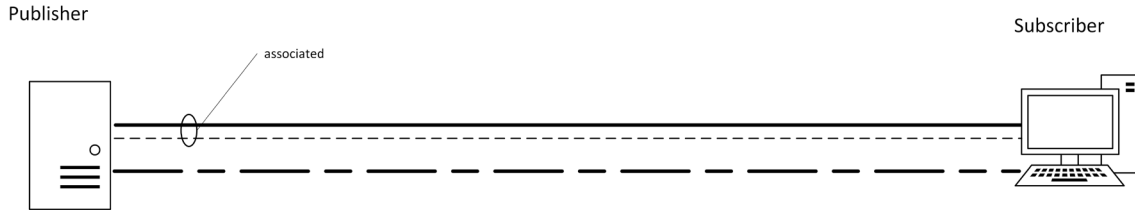


Figure 5—Multiple command and associated subscription channels—2

4.2 Command and response

All STTP protocol traffic is composed of simple commands and responses that can each optionally carry a payload. Typically, a subscriber shall issue commands and a publisher shall answer with responses. Commands from the subscriber include a command type, a payload length, and optional payload bytes. Responses from the publisher include a response type, an in-response-to command type, payload length with payload bytes. Often the response type simply indicates success or failure with an associated message; otherwise, the response type indicates that a specific payload format is in use, e.g., a DATA PACKET. A common command and response sequence is shown in Figure 6. Command-response messages are described in Clause 5.

The command and response sequence shown in Figure 6 describes a typical sequence of steps that a subscriber and publisher follow such that the end state for the subscriber is continuous reception of desired streaming data from the publisher, see Figure 6 (Rn). Before allowing a subscriber connection, the publisher shall be configured with the required access control, data parameters, and communication parameters to allow meeting security, data selection, and communication requirements. Operation is described in Clause 6 and security in Clause 7.

After the subscriber has initiated a TCP/IP connection to the publisher, and the publisher has accepted the connection, the subscriber shall send the DEFINE OPERATIONAL MODES command (5.4.6) to the publisher, Figure 6 (C1). This shall be the very first command that is sent to the publisher; any other commands sent before the DEFINE OPERATIONAL MODES command results in the publisher dropping the connection. If the publisher accepts the command flags (Table 4) and key/value parameters (Table 6) defined in the DEFINE OPERATIONAL MODES command parameter expression payload, the publisher shall send a SUCCEEDED response (5.5.1) to the subscriber, Figure 6 (R1).

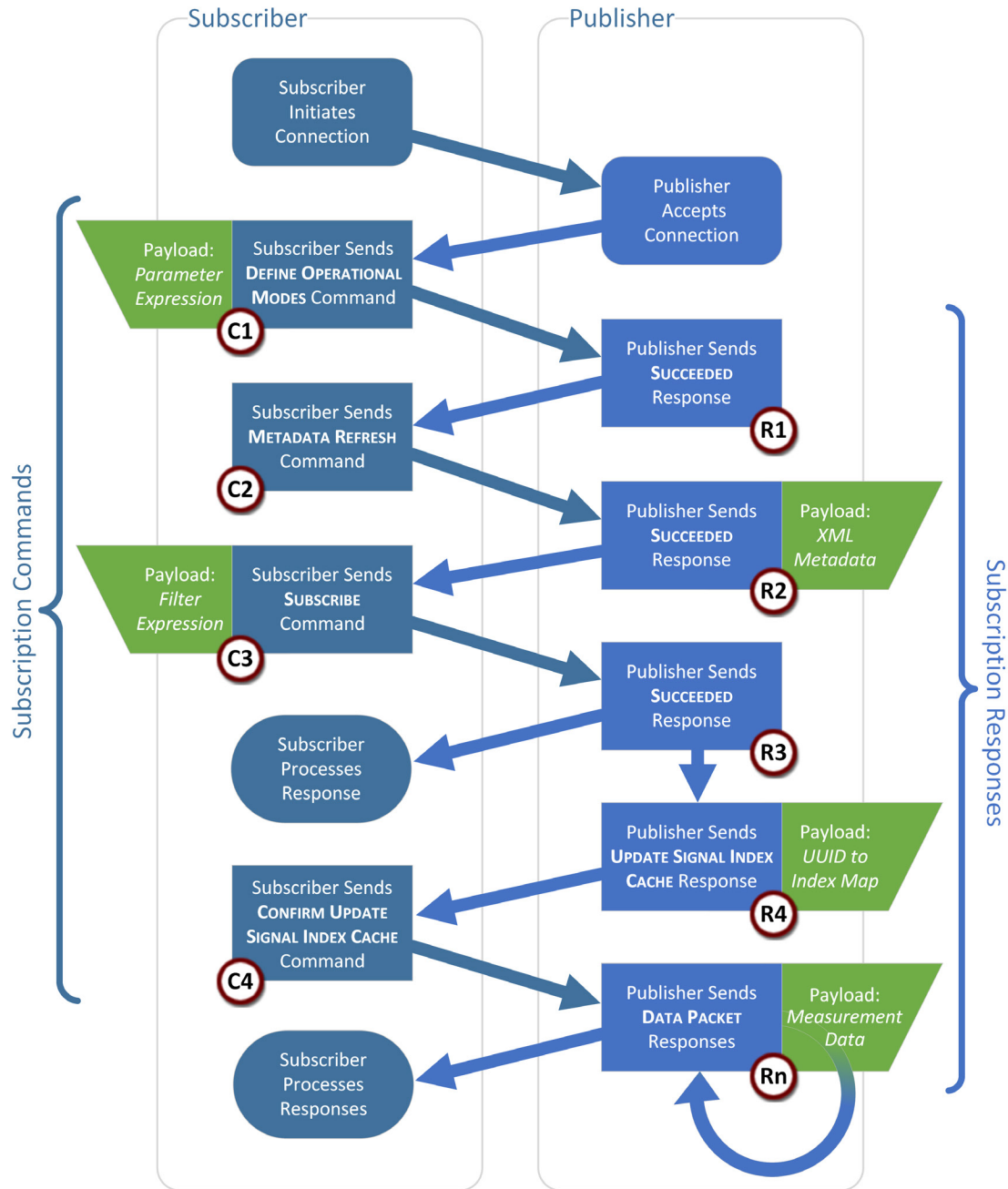


Figure 6—Common command and response sequence

Once the subscriber has received a SUCCEEDED response from the publisher for the DEFINE OPERATIONAL MODES command, the session is established, and typical data exchange operations can proceed. A common next step is for the subscriber to request updated metadata from the publisher. The subscriber shall request updated metadata by sending a METADATA REFRESH command (5.4.1) to the publisher, Figure 6 (C2). Refreshing metadata is important because the publisher's available data sources may have changed since the subscriber's last subscription, or the publisher's configuration may have been altered changing the accessibility of information for which the subscriber is authorized to receive. If the publisher accepts the METADATA REFRESH command, the publisher shall send a SUCCEEDED response (5.5.2) to the subscriber, Figure 6 (R2). The payload associated with the SUCCEEDED response for a METADATA REFRESH command consists of tabular eXtensible Markup Language (XML) metadata (Clause 9) that describes data that the

subscriber is authorized to receive. The metadata includes Universally Unique Identifier (UUID)-based identifiers that uniquely identify each signal the publisher can provide (see RFC 9562). The UUID values are permanent identifiers for measurement metadata that the publisher can provide that shall be persistent and consistent between subscriber connections.

When the subscriber has received the XML metadata advertising what data the publisher can provide, the subscriber can then choose which data is desired for streaming. Choosing data for a subscription commonly involves a user configuration process to determine exactly which data is desired; selected data is based on target use cases, and/or specific application needs. Data selection for a subscription may use **Filter Expressions** that either specify exactly which measured data is desired using the identifiers defined in the metadata or syntax that is analogous to Structured Query Language (SQL). Syntax for **Filter Expressions** is described in Annex C. Once a **Filter Expression** for the desired data has been defined, the subscriber shall establish a subscription by sending a **SUBSCRIBE** command (5.4.2) to the publisher, Figure 6 (C3). If the publisher accepts the **Filter Expression** defined in **SUBSCRIBE** command payload, the publisher shall send a **SUCCEEDED** response (5.5.1) to the subscriber, Figure 6 (R3). The **SUCCEEDED** response shall be immediately followed by an **UPDATE SIGNAL INDEX CACHE** response (5.5.5), which includes a payload that maps each subscribed UUID-based identifier (as defined in the metadata) to a unique session specific integer-based runtime index required for proper deserialization of received data, Figure 6 (R4). Before the publisher can begin streaming selected data to the subscriber, the subscriber shall confirm reception of the **UPDATE SIGNAL INDEX CACHE** response by sending a **CONFIRM UPDATE SIGNAL INDEX CACHE** command (5.4.10) to the publisher, Figure 6 (C4). Note that the publisher may send a new **UPDATE SIGNAL INDEX CACHE** response any time its source configuration has changed. The subscriber shall confirm receipt by sending a **CONFIRM UPDATE SIGNAL INDEX CACHE** command to enable uninterrupted data flow.

After a successful subscription and confirmed reception of an **UPDATE SIGNAL INDEX CACHE** response, the publisher shall continuously stream **DATA PACKET** responses (5.5.4) to the subscriber, Figure 6 (Rn). Note that the publisher sends small groups of data to the subscriber as it is received from the data source. There is no further time alignment, or post-processing of data before publication.

4.3 Data units and packeting

As introduced previously, the protocol uses a command channel and separate subscription channel for transferring data. The publisher supplies the data, including metadata that describes the measured data. The subscriber makes the connection with the publisher establishing the command channel, and then using this channel makes a request for a data set that is desired and sets the communication parameters. If the request is approved, the publisher supplies the metadata and then sends the data as specified, using the subscription channel. This data may be a continuous stream or fixed set of historical records. Command-response activity other than continuous data flow of **DATA PACKET** (5.5.4) or **BUFFER BLOCK** (5.5.10) response messages is intermittent and limited, so there are no special constraints applied to these packet sizes not related to data flow. If multiple sets of data are to be sent, separate command and subscription channels are required.

The primary data type targeted in STTP is time tagged data, such as time-series values. This data could be measurements, counts, codes, or anything else that could be represented as a computer encoded value, e.g., real numbers.¹⁶ As an example, in the power system domain, these numbers could be individual measurements such as frequency, voltage magnitude, voltage phase angle, and real power. Data is sent in individual data units consisting of a value, a relevant time tag, a data identifier, and a quality flag. The timestamp indicates when the value was received or derived, such as the time a measurement was made.¹⁷

¹⁶ Value type defaults to 32-bit floating pointer number, however, the field size is 4 bytes, which can be used for any other native type that can fit into the same field size, e.g., an unsigned 16-bit integer. Actual data type is indicated by metadata (see Clause 9). Non-numeric and custom data types larger than 32 bits are supported through **BUFFER BLOCK** response messages (see Figure 33).

¹⁷ In some use cases, the timestamp could simply represent a continually incrementing index or be excluded altogether.

The ID links the value to a description in the metadata and the flag indicates data quality.¹⁸ This is the basic unit for transferring data with STTP.

The basic data value is 4 bytes, along with its associated identifier at 16 bytes, a timestamp of 8 bytes, and 32 bits of flags at 4 bytes. Thus, the full measurement unit is 32 bytes. As this is a large amount of data to send for each value, STTP employs two methods to reduce the amount of data to be sent on the wire. The first is a compact measurement reduction that maps the 32-byte unit to a 9- to 17-byte structure. The second is the use of custom compression algorithms. When using the TSSC lossless compression algorithm, the 32-byte unit can compress to an average size of 2.7 bytes. Figure 7 shows the full, compact, and compressed structures together for comparison. Details for converting the full measurement set to the reduced sets are given in 5.5.4, Clause 11, and Annex E.

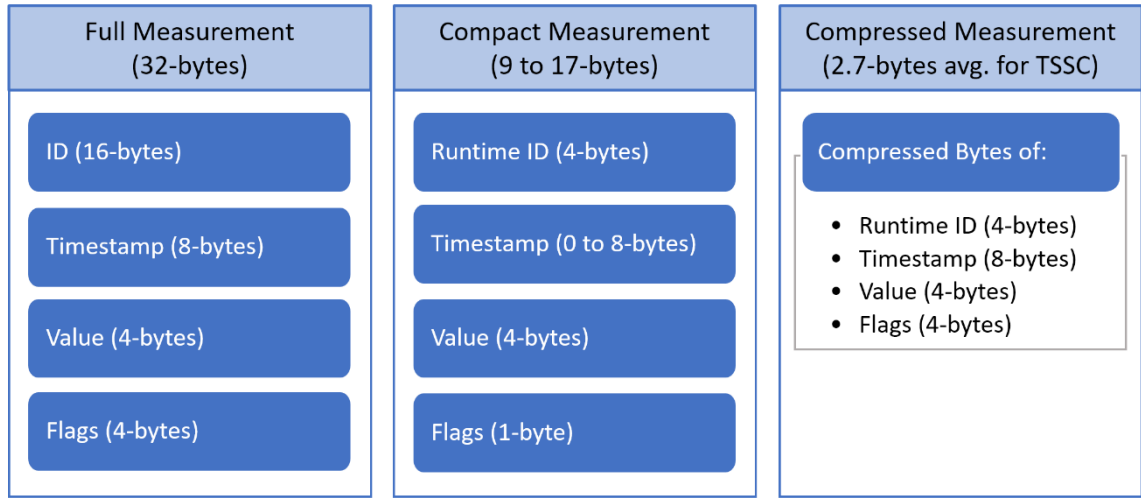


Figure 7—Full, compact, and compressed data units

The Full Measurement unit is only used in memory—there are no serialization options for sending it. The Compact and Compressed units are more compact and used for wire serialization. As these are considerably smaller than a typical DATA PACKET, STTP can be set to map several data units into one DATA PACKET for transmission to reduce data handling and bandwidth use. The publisher fills a packet with individual measurement units until it reaches the maximum packet size and then sends the packet. The maximum packet size for a DATA PACKET can be chosen to be less than the MTU for the network to avoid fragmentation and reduce data loss.

For types of data that do not conform to a time-series values, a free-form data structure called a BUFFER BLOCK can be used (see 5.5.10). Like the DATA PACKET, a BUFFER BLOCK supports the use of compression algorithms. Although the BUFFER BLOCK could contain measured values, it does not conform to this standard as a time-series value. However, a BUFFER BLOCK is still considered a data item that can be subscribed.

4.4 Built-in security

The command channel is used to initiate and manage all connections, including application of standard IP security mechanisms, i.e., transport layer security (TLS¹⁹) (RFC 9150 [B19]). Publishers authenticate

¹⁸ For the compact measurement structure, as seen in Figure 7, quality is reduced to a single byte for bandwidth minimization and is intended for simple quality indications, see Figure 27 for details. More commonly, the DATA PACKET payload is compressed (see Clause 11), and a full 32 bits of quality resolution travel with each measurement, see Annex E.

¹⁹ Information on TLS V1.2 (RFC 8446) and TLS V1.3 (RFC 8446) can be found in Clause 2.

subscribers using TLS X.509 (ISO/IEC 9594-8:2020 [B10]) certificates and use the certificate's public-key to secure communications. The publisher is in control of which subscribers shall be allowed to make connections as well as the level of security of each. As security and security level is optional, the publisher may opt to not use security. Details of the security available and how it is built-in and applied are covered in Clause 7.

4.5 Measurement centered

Data reporting is centered around individual measurement values. In the electric power industry values could be, for example, real power, voltage magnitude, and frequency. Each measurement value has a unique ID (a UUID), a timestamp, and a quality indicator (as shown in Figure 7). Individual measurements are mapped from specific metadata descriptions, so they can be chosen individually. This approach facilitates selection of specific items from a data set to reduce the amount of data that needs to be sent from the publisher to each subscriber.

NOTE—Compared to frame-based protocols that include a single timestamp and quality flags per frame of simultaneously measured data, the native STTP binary format for an equivalent group of measurements encoded using the compact measurement format is larger, as each measurement travels with its own ID, timestamp, and quality flags. However, STTP deployments allow data to be sent in compressed form in addition to compact measurement format. Compression techniques result in messages no larger than the uncompressed data sent in most other comparable protocols. Refer to the compression details in Clause 11.²⁰

5. Command and response messages

5.1 Message introduction

STTP protocol traffic shall be composed exclusively of enumerated commands and responses, see Table 1 and Table 2, respectively, for STTP command and response types. Some commands or responses carry a payload. In a typical exchange, a subscriber shall issue commands and a publisher shall answer with responses (see Figure 6). Commands from the subscriber include a command type, an optional payload length, and optional payload bytes. Responses from the publisher include a response type, an in-response-to command type, optional payload length, and optional payload bytes. Often the response type simply indicates success or failure with an associated message, otherwise the response type indicates that a specific payload format is in use, e.g., a DATA PACKET.

5.1.1 Payloads with strings

Some of the payloads of command and response messages may be a string or may contain strings. In STTP, strings shall be encoded in natural order from left to right, using the UTF-8 (ISO/IEC 10646:2017) character encoding method.

5.1.2 Endian byte order

Unless otherwise explicitly specified, all numeric byte values (integer and floating point) that are encoded as binary components in STTP commands and responses shall be in big endian order (also known as *network byte order*), regardless of the endianness of the host operating system. *Endianness* refers to the

²⁰ Notes in text, tables, and figures of a standard are given for information only and do not contain requirements needed to implement this standard.

method by which the most significant byte (MSB) of a numeric value is stored. For example, big endian stores the MSB at the lowest byte address, whereas, little endian is the opposite: it stores the MSB at the highest byte address.

5.1.2.1 UUID encoding

UUID values that are binary encoded for wire serialization in STTP shall also be encoded in big endian order, per RFC 9562. More specifically, big endian byte order is be used for the (a) time_low / uint32, (b) time_mid / uint16 and (c) time_hi_and_version / uint16 fields of the UUID value, regardless of the endianness of the host operating system. All other remaining byte field values are encoded left to right.

5.1.3 Connections and subscriptions

A *connection* is established when a publisher issues a SUCCEEDED response to a subscriber's DEFINE OPERATIONAL MODES command using the command channel. A connection is terminated when the TCP connection associated with the command channel is terminated. When the TCP command channel connection terminates, the publisher shall terminate any associated subscription channels if any exist. While a connection is active, the publisher shall issue a response to every properly formatted command sent by the subscriber except for confirmation commands, i.e., CONFIRM NOTIFICATION, CONFIRM BUFFER BLOCK, CONFIRM UPDATE BASE TIMES, CONFIRM SIGNAL INDEX CACHE, and CONFIRM UPDATE CIPHER KEYS, none of which require a response. The FAILED response may be a valid response to some commands.

A *subscription* is established when a publisher issues a SUCCEEDED response to a subscriber's SUBSCRIBE command. A subscription shall be terminated when the subscriber issues an UNSUBSCRIBE command or when the publisher issues a PROCESSING COMPLETE response in the case of a historical subscription. A subscription is also terminated when the command channel terminates. While a subscription is active, the subscriber shall be prepared to receive DATA PACKET responses and all other response messages from the publisher relevant to the subscription. During data transmission, the publisher continues to send response messages, such as UPDATE SIGNAL INDEX CACHE, UPDATE BASE TIMES, etc., that are required to keep the subscriber harmonized with the publisher. There may be multiple subscriptions active for any given publisher-subscriber pair, each using a separate command and subscription channel for transferring data. Each of these is a separate connection between the publisher and subscriber representing a unique session.

Either the publisher or subscriber may initiate an action to terminate a connection or a subscription.

5.2 STTP commands

The subscriber command message shall consist of a command type, an optional payload length, and any payload bytes necessary to fully parameterize the command. All STTP commands shall be transmitted using the command channel. The generic STTP command message is shown in Figure 8.

FIELD	LENGTH (BYTES)	DESCRIPTION
COMMAND TYPE	1	Command type (see Table 1.)
PAYLOAD LENGTH	0 or 4	Number of bytes of payload, if specified command type includes payload (int32)
PAYLOAD	Variable	Actual bytes of payload, if any

Figure 8—STTP command message

NOTE 1—A command can have a length of one, five or greater than five bytes. If the length is one byte, there is no payload. If the length is five bytes, then a PAYLOAD LENGTH is included, but contains the value zero indicating no payload. If the PAYLOAD LENGTH is greater than five, then the command contains a payload, and value contained in the PAYLOAD LENGTH field shall be five less than the total message length. For example, some commands have no payload (see Table 1); in these cases, command length shall be exactly one byte. Otherwise, all commands that have payload shall be at least five bytes. When specified, a command can have a PAYLOAD LENGTH value of zero, meaning empty payload; in this case, PAYLOAD LENGTH with a value of 0x0000 shall be included, and command length is exactly five bytes.

When describing command and response messages, the payload and any repeated fields are indented to indicate the preceding length or count field specifies the length or repeat count of the indented fields. The length or count fields indicate how the length or count is to be specified. For example, for PAYLOAD LENGTH, “(remainder of message)” is specified to indicate the length is the number of bytes in the remainder of the message. For repeating fields, the COUNT field indicates the names of the fields that are repeated, for example in Figure 29, the REFERENCE COUNT field specifies to “(Repeat fields SIGNAL INDEX to SIGNAL ID for REFERENCE COUNT times)”.

An STTP publisher shall support the STTP command message types shown in Table 1.. Many command types require a payload specific to the requested command. Where required, these are described in 5.4.

NOTE 2—After sending a command message from a subscriber to a publisher, the publisher normally sends a SUCCEEDED or FAILED response message containing an associated message in the payload, i.e., a string of text, detailing the success or failure of the command operation. The payload type for other response successes is based on the response type. For example, the publisher response for a successful METADATA REFRESH command is a serialized dataset of the available publisher metadata specifically allowed for the subscriber. The payload content for FAILED responses is a string of UTF-8 text representing the error message.

Table 1—STTP command types

Command	OPCODE	Description	Payload
METADATA REFRESH	0x01	Requests that the publisher send an updated set of metadata so that the subscriber can update its dataset. The successful response message type is a dataset containing the server device and measurement metadata. Devices and measurements contain UUIDs that are used to uniquely identify metadata in local repository allowing datasets to be merged as received from multiple subscriptions.	Yes, but PAYLOAD LENGTH of zero is allowed
SUBSCRIBE	0x02	Requests a subscription of streaming data from the publisher based on a provided parameter string expression. The string expression contains parameters that specify the desired measurements and control if the subscription is for real-time or historical data, when supported. A new subscription may be requested during transmission of an existing subscription. The existing subscription shall be terminated before the new one is started. The successful response is a message containing information about the subscription. Upon successful subscription the subscriber shall also receive an UPDATE SIGNAL INDEX CACHE response and may also receive an UPDATE BASE TIMES response that allows parsing of newly subscribed measurement data. For a successful subscription, data shall begin to flow following publisher reception of a CONFIRM UPDATE SIGNAL INDEX CACHE command.	Yes
UNSUBSCRIBE	0x03	Requests that the publisher stop sending streaming data to the subscriber. This command cancels the current subscription and closes any associated subscription channel.	No

Table 1—STTP command types (*continued*)

Command	OPCODE	Description	Payload
ROTATE CIPHER KEYS	0x04	Requests that the publisher send a new set of cipher keys for encryption of information transmitted in the subscription channel when using UDP. The response includes two keys (an old one and a new one) to accommodate transitioning from one key to another. A bit in the payload indicates which key to use. To protect symmetric key exchange, this command should only be used in conjunction with an encrypted (e.g., TLS-based) command channel.	No
UPDATE REPLAY INTERVAL	0x05	Requests that the publisher update its historical data replay interval with the specified value in milliseconds (ms)—this only applies for a subscription setup to replay historical data, not to a real-time stream. Except for the values of -1 and 0, this value specifies the desired replay interval for data, i.e., basically a delay, or timer interval, over which to process data. A value of -1 means to use the default replay interval while a value of 0 means to process data as fast as possible.	Yes
DEFINE OPERATIONAL MODES	0x06	Defines the protocol version and operational modes for a subscriber connection. As soon as a connection with a publisher is established, this command defines the server operational modes used for subscriber and publisher communication (e.g., compression options or text encoding style). This command shall be used only once per connection and shall be the first command to be sent by the subscriber to the publisher.	Yes
CONFIRM NOTIFICATION	0x07	This command is sent to a publisher to confirm that a NOTIFY response was received. This is used to verify delivery of critical messages from publisher to subscriber, e.g., a control operation.	Yes
CONFIRM BUFFER BLOCK	0x08	This command is sent to a publisher to confirm that a BUFFER BLOCK response was received. This is used to verify delivery of data blocks that may require continuity and sequencing, e.g., a file data block.	Yes
CONFIRM UPDATE BASE TIMES	0x09	This command is sent to a publisher to confirm the UPDATE BASE TIMES response was received. This is used to verify delivery of the base time updates indicating to the publisher that it is permissible to transition to the new offsets.	Yes
CONFIRM UPDATE SIGNAL INDEX CACHE	0x0A	This command is sent to a publisher to confirm that UPDATE SIGNAL INDEX CACHE response was received. This is used to verify delivery of the signal index cache indicating to the publisher that it is permissible to transition to the new cache.	No
CONFIRM UPDATE CIPHER KEYS	0x0B	This command is sent to a publisher to confirm that the UPDATE CIPHER KEYS response was received. This is used to verify delivery of the cipher keys indicating to the publisher that it is permissible to transition to the new keys.	No
GET PRIMARY METADATA SCHEMA	0x0C	Requests that the publisher send the schema for the primary metadata (see Clause 9). The successful response message type is a dataset containing the schema of the primary metadata.	Yes, but PAYLOAD LENGTH of zero is allowed
GET SIGNAL SELECTION METADATA SCHEMA	0x0D	Requests that the publisher send the schema for the signal selection metadata (see Clause 9). The successful response message type is a dataset containing the schema of the signal selection metadata.	Yes, but PAYLOAD LENGTH of zero is allowed
EXTENDED COMMANDS 0-15	0xD0-0xDF	Defines extended commands for custom functionality. Note that extended commands shall not be used to re-implement a standard function command.	Custom

5.3 STTP responses

The publisher response message shall consist of a response type, an in-response-to command type, payload length, and actual payload bytes if any. All STTP responses, unless otherwise specified, shall be transmitted using the command channel. The in-response-to command type is required, even if the response is unsolicited. The generic response message is shown in Figure 9.

FIELD	LENGTH (BYTES)	DESCRIPTION
RESPONSE TYPE	1	Response type (see Table 2)
IN RESPONSE TO COMMAND TYPE	1	The in-response-to command type (see Table 1)
PAYLOAD LENGTH	0 or 4	Number of bytes of payload, if specified response type includes payload (int32)
PAYLOAD	VARIABLE	Actual bytes of payload, if any

Figure 9—STTP response message

NOTE—A response can have a length of two, six or greater than six bytes. If the length is two bytes, there is no payload. If the length is six bytes, then a PAYLOAD LENGTH is included, but contains the value zero indicating no payload. If the PAYLOAD LENGTH is greater than six, then the response contains a payload, and value contained in the PAYLOAD LENGTH field shall be six less than the total message length. For example, some responses have no payload (see Table 2); in these cases, response length shall be exactly two bytes. Otherwise, all responses that have payload shall be at least six bytes. When specified, a response can have a PAYLOAD LENGTH value of zero, meaning empty payload; in this case, PAYLOAD LENGTH with a value of 0x0000 shall be included, and response length is exactly six bytes.

The size of the payload, if any, is specific to the response type. For example, in the case of a DATA PACKET response, the payload contains serialized measurements. Although the subscriber commands and publisher responses are on two different paths, the operation code used for response types are defined as distinct from those used for command types to make it easier to identify values from a wire analysis. Normally response messages are issued in response to subscriber commands, however they may also act like commands sent by the publisher to the subscriber that were not necessarily solicited. Regardless, they are still referred to here as responses for clarity in communicating data flow direction. Many response types require a payload specific to the specified response; these are described in 5.5.

When describing command and response messages, the payload and any repeated fields are indented to indicate the preceding length or count field specifies the length or repeat count of the indented fields. The length or count fields indicate how the length or count is to be specified. For example, for PAYLOAD LENGTH, “(remainder of message)” is specified to indicate the length is the number of bytes in the remainder of the message. For repeating fields, the COUNT field indicates the names of the fields that are repeated, for example in Figure 29, the REFERENCE COUNT field specifies to “(Repeat fields SIGNAL INDEX to SIGNAL ID for REFERENCE COUNT times)”.

If there is no payload associated with a response, the PAYLOAD LENGTH field is not contained in the message, and therefore has a length of 0; the length of such a response message is two bytes.

An STTP publisher shall provide the STTP response message types shown in Table 2. Many response types require a payload specific to the requested command. Where required, these are described in 5.5.

Table 2—STTP response types

Response	OPCODE	Description	Payload
SUCCEEDED	0x80	Informs the subscriber that its solicited server command succeeded, a success message payload follows.	Yes
FAILED	0x81	Informs the subscriber that its solicited server command failed, a failure message payload follows.	Yes
DATA PACKET	0x82	Informs the subscriber of a DATA PACKET (included in payload).	Yes
UPDATE SIGNAL INDEX CACHE	0x83	Requests that the subscriber update its runtime signal index cache with the one that follows.	Yes
UPDATE BASE TIMES	0x84	Requests that subscriber update its runtime base-timestamp offsets with those that follow.	Yes
UPDATE CIPHER KEYS	0x85	Requests the subscriber update its runtime symmetric encryption keys with those that follow and use the keys to decrypt data. This is only applicable when the subscription channel is using UDP. Keys should only be transferred used in conjunction with an encrypted (e.g., TLS-based) command channel.	Yes
DATA START TIME	0x86	Provides the start time of data being processed from the first measurement.	Yes
PROCESSING COMPLETE	0x87	Provides notification that historical replay processing has completed, established via temporal constraints parameters, i.e., StartTime and StopTime parameter string expression.	No
BUFFER BLOCK	0x88	Informs the subscriber of a BUFFER BLOCK (included in payload). This works with a specially defined metadata measurement, still requiring subscription that allows a free form transfer of data that does not conform to a time-series value. Data in the block shall be partitioned to fit within a minimal number of network packets. When requested, the subscriber shall acknowledge reception of BUFFER BLOCK with a CONFIRM BUFFER BLOCK command since blocks may exist as a sequence of packets and may require retransmission when used over an unreliable communications transport, e.g., UDP, as BUFFER BLOCKS are transmitted over the subscription channel. BUFFER BLOCKS are transmitted with a sequence number and when block reception requires confirmation, the publisher tracks expected sequence number to be confirmed. Failure to receive a confirmation for a block shall induce a retransmission of the missing block. The publisher shall verify that the subscriber acknowledges receipt of all blocks requiring confirmation and retransmit missing blocks.	Yes
NOTIFY	0x89	Informs the subscriber of a critical notification (included in payload). This works with a specially defined metadata measurement, still requiring subscription, that allows subscriber to receive messages with verified delivery. Since message is considered critical, subscriber shall respond with a CONFIRM NOTIFICATION command since message. Each notification message is transmitted with a unique 32-bit hash identifier, e.g., a hash for a UUID, and since notification reception requires confirmation, the publisher holds notifications until confirmed. All notification traffic is over the command channel.	Yes
CONFIGURATION CHANGED	0x8A	Provides a notification that the publisher's source configuration has changed, and that subscriber should make a request to refresh metadata.	No
EXTENDED RESPONSES 0-15	0xE0-0xEF	Defines extended responses for custom functionality. Note that extended responses shall not be used to re-implement a standard function response.	Custom
NO OP	0xFF	Informs the subscriber that command channel is still active. Since it is possible for the command channel to remain quiet for some time, this command allows a periodic test of continued connectivity.	No

NOTE—Solicited commands, e.g., METADATA REFRESH or SUBSCRIBE, expect a SUCCEEDED or FAILED response message. For any commands or responses transmitted on the reliable TCP command channel, dropped data is retransmitted so long as a connection is active. However, for various reasons, it is still possible for a message to never be received. STTP implementations should implement a user configurable timeout for expected messages. Sessions should be disconnected if the timeout expires without receiving an expected message.

5.4 STTP command messages

STTP commands are described in Table 1. The following subclauses describe the messages associated with each STTP command.

5.4.1 METADATA REFRESH command message

The payload for the METADATA REFRESH command is optional. If the payload length has a value of zero, no filters are applied to the publisher metadata and all records that the subscriber is authorized to view shall be returned in the dataset. When a payload is specified, i.e., the payload length is greater than zero, the payload shall conform to the FILTER EXPRESSION format defined in Annex C. Filtering expressions allow the subscriber to request a reduced metadata dataset from the publisher. For example, the subscriber may want to reduce the metadata to records for a certain company or signal type to reduce the number of records received. Multiple expressions may be defined, each separated by a semicolon. Figure 10 shows the format of the optional METADATA REFRESH command message:

FIELD		LENGTH (BYTES)	DESCRIPTION
COMMAND		1	0x01 (METADATA REFRESH)
PAYLOAD LENGTH		4	Number of bytes of payload (int32)
	EXPRESSION LENGTH	4	Number of bytes in the expression (int32)
	EXPRESSION	EXPRESSION LENGTH	String-based FILTER EXPRESSIONS separated by semicolons

Figure 10—METADATA REFRESH command message

NOTE—The METADATA REFRESH command can have a length of five or greater than five bytes. If the length is five bytes, then a PAYLOAD LENGTH is included, but contains the value zero indicating no expressions are included in the payload. If the PAYLOAD LENGTH is greater than five, then the METADATA REFRESH command contains at least one expression, and value contained in the PAYLOAD LENGTH field is five less than the total message length.

5.4.2 SUBSCRIBE command message

The payload for the SUBSCRIBE command defines the desired DATA PACKET OPTIONS and parameter string expression used to start data transport from the publisher. The response shall include information on all requested data that can be supplied. Requests for measurements that do not exist or are not allowed by the publisher for the particular subscriber shall not be included in the reply. Figure 11 shows the format of the SUBSCRIBE command message.