J. Ritchie Carroll
Grid Protection Alliance

ASP Project: STTP
Streaming Telemetry Transport Protocol

NASPI Fall Meeting
Philadelphia, PA
October 23, 2018

DOE FOA 1492
DE-OE-859

Project Partners

Advanced Synchrophasor Protocol Project

sttp

DOE FOA 1492
DE-OE0000859

ASP

Streaming Telemetry Transport Protocol

Project Collaborators	Project Financial Partner	Vendor	Utility	Demonstration Host
Bonneville Power Administration	◆		◆	
Bridge Energy Group				
Dominion Energy	◆		◆	EPG
Electric Power Group	◆	◆		
Electric Power Research Institute				
ERCOT			◆	
Grid Protection Alliance (Prime)	◆	◆		
ISO New England			◆	
MehtaTech		◆		
Oklahoma Gas & Electric	◆		◆	WSU
OSIsoft		◆		
Peak Reliability			◆	
PingThings		◆		
PJM Interconnection			◆	EPG
Southern California Edison			◆	
San Diego Gas & Electric	◆		◆	WSU
Schweitzer Engineering Laboratories	◆	◆		
Southern Company Services			◆	
Southwest Power Pool	◆		◆	WSU
Space-Time Insight		◆		
Trudnowski & Donnelly Consulting Engineers		◆		
Utilicast	◆	◆		
Tennessee Valley Authority	◆		◆	WSU
University of Southern California				
V&R Energy		◆		
Washington State University	◆	◆		
	26	11	11	12	6

Schedule and Deliverables

Deliverables

- Update PMP
- Release ASP Spec
- Develop α Toolkit
- Develop Demo Plan
- Publish Demo Results
- Publish API

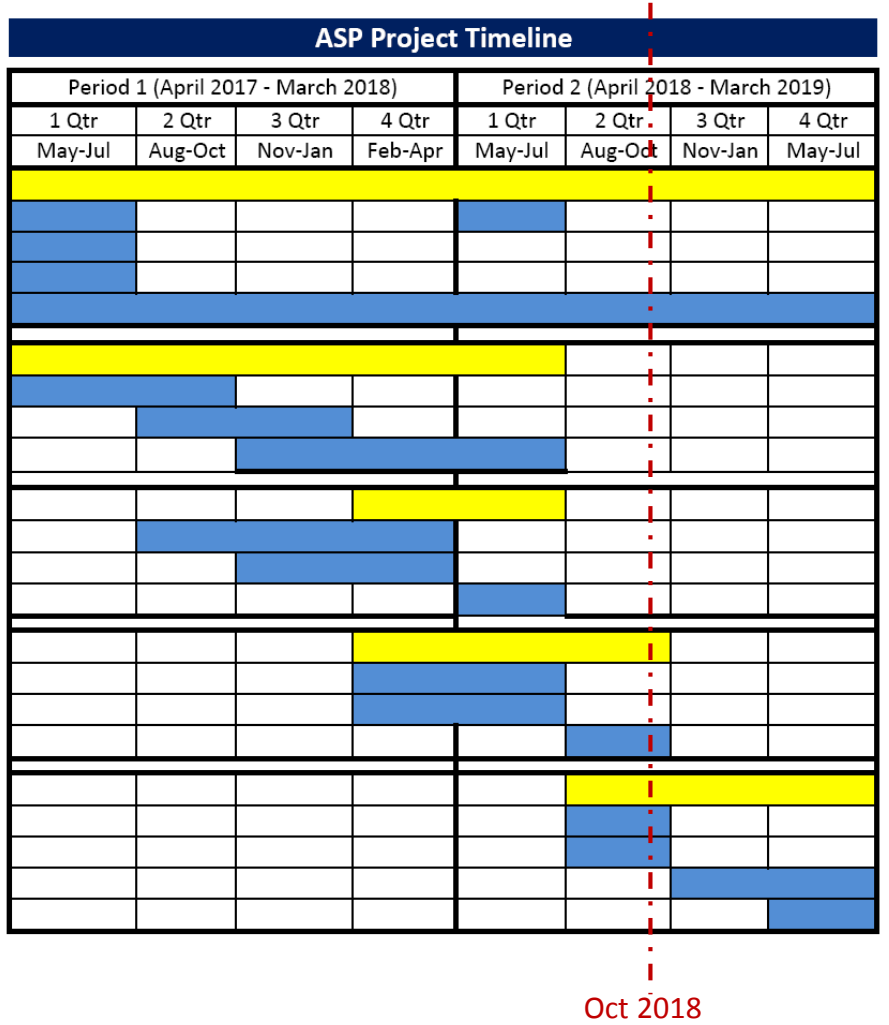
1.0 Project Governance
- 1.1 Update PMP (D1)
- 1.2 Update Data Management Plan
- 1.3 Establish Contracts
- 1.4 Manage Project and Submit Reports

2.0 Protocol Specification
- 2.1 Define Requirements
- 2.2 Create Initial Design (M1)
- 2.3 Release ASP Specification (D2 - M2)

3.0 Alpha Software Development
- 3.1 Develop Alpha APIs
- 3.2 Develop Alpha Tool Kit (D3)
- 3.3 Release Alpha Versions (M3)

4.0 Incorporate the ASP APIs into Tool Suites
- 4.1 Incorporate Alpha ASP into EPG Tools
- 4.2 Incorporate Alpha ASP in WSU Tools
- 4.3 Bench Test EPG and WSU Tools

5.0 Demonstrations and Final ASP Specification
- 5.1 Develop EPG Tool Demo Plan (D4)
- 5.2 Develop WSU Too Demo Plan
- 5.3 Conduct Demo & Publish Results (D5-M4)
- 5.4 Publish Ver 1.0 API with Documentation (D6 - M5)

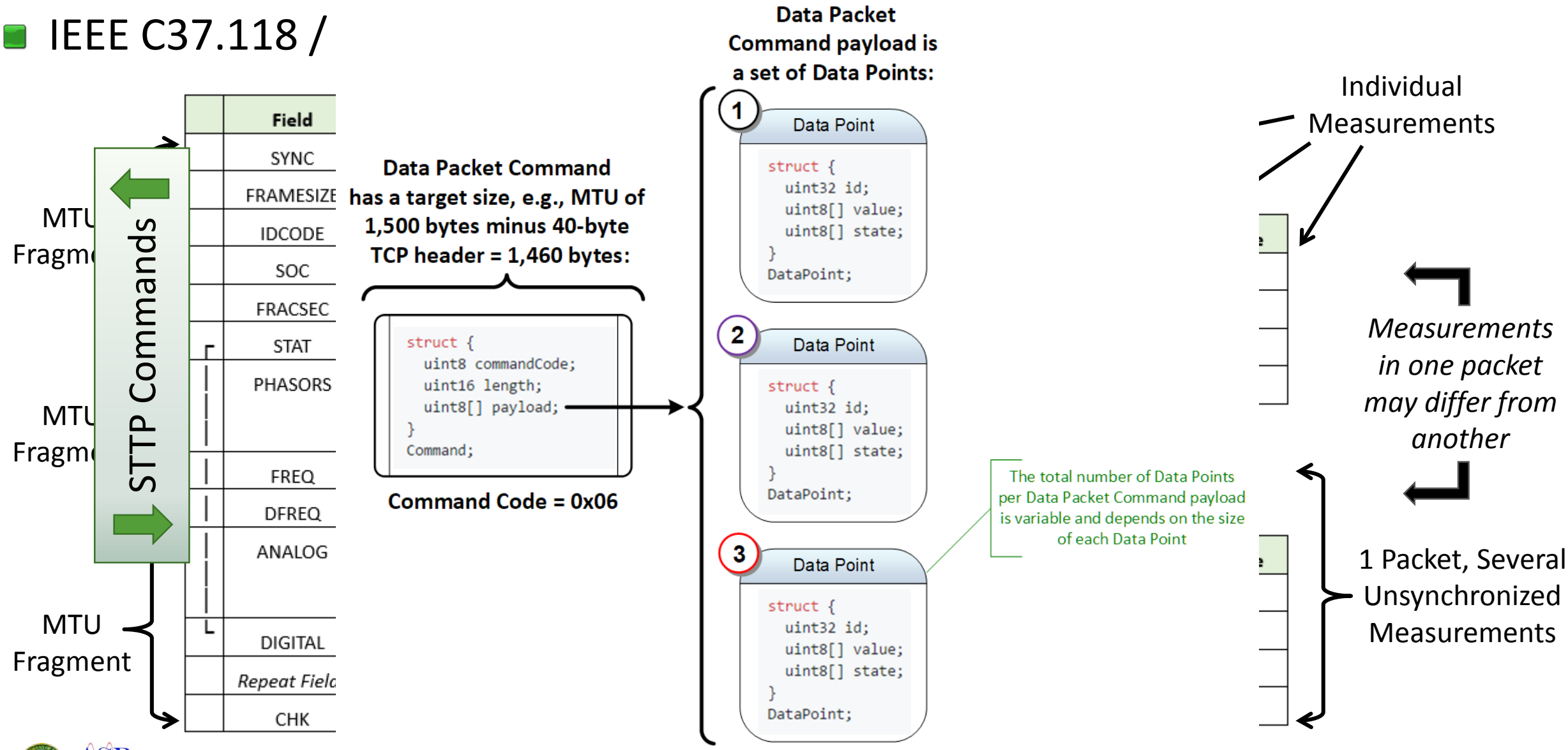


What Makes STTP Different?

- Synchronized Frames vs Atomic Packets
- Reduced Data Loss
- Lossless Compression
- Scalability (to hardware limits)
- Publish / Subscribe Model
- Publisher Data Access Control
- IP Level Security
- Configurable Connection Origin

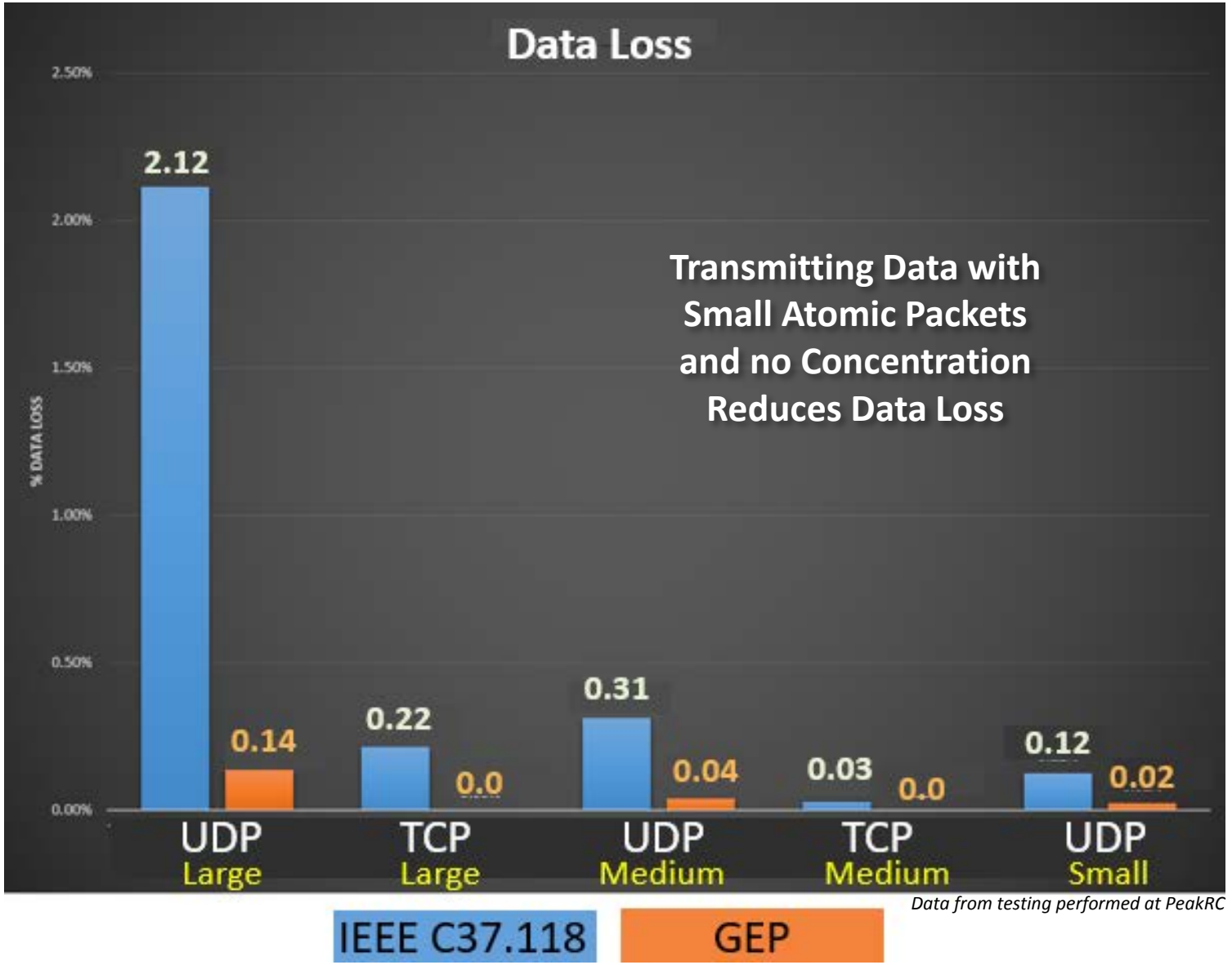
Difference: Synchronized Frames vs Atomic Packets

- IEEE C37.118 /



Difference: Reduced Data Loss

The Wire Protocol

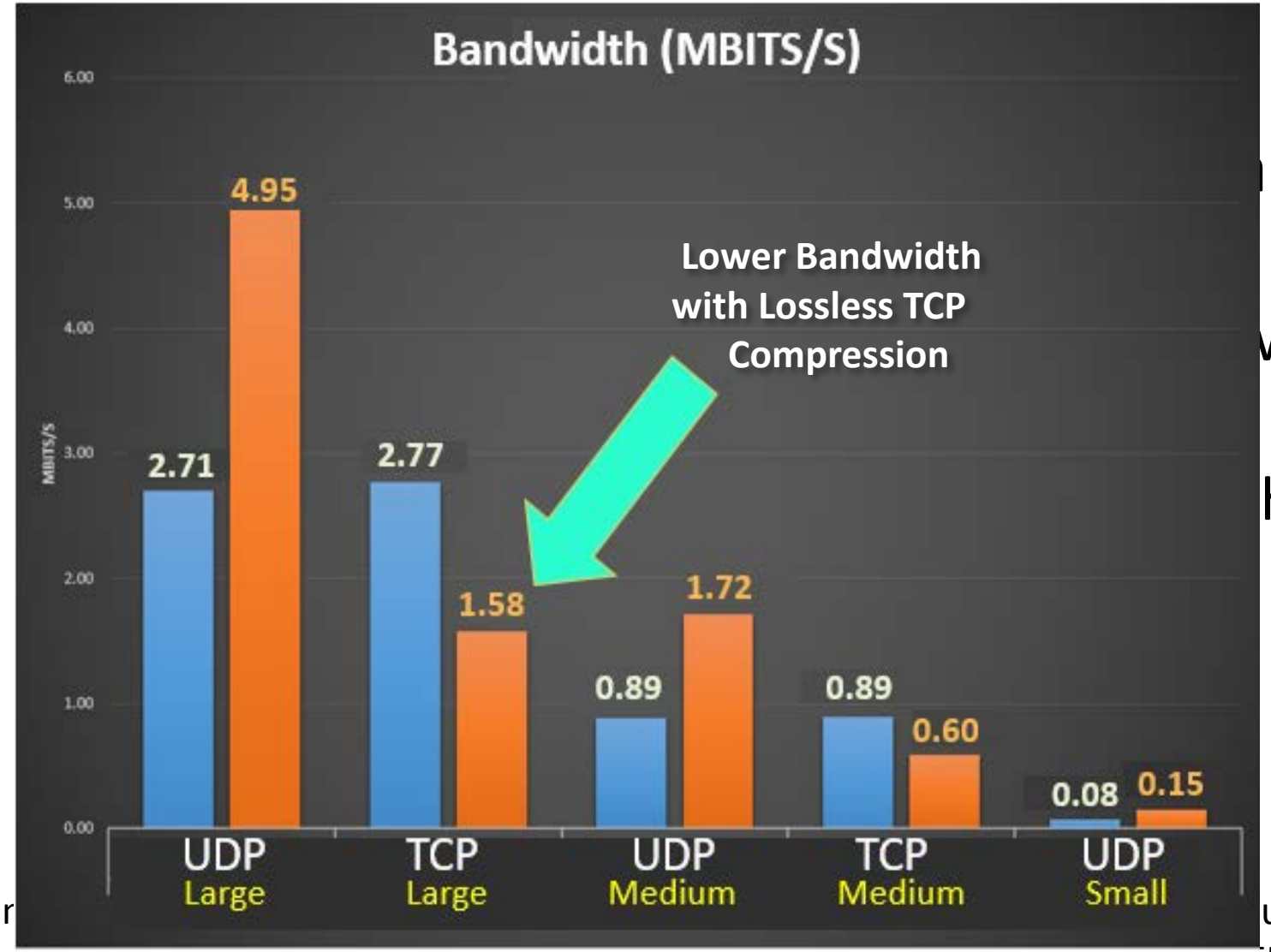


Transmitting Data with Small Atomic Packets and no Concentration Reduces Data Loss

Data from testing performed at PeakRC

Difference: Lossless Compression

The Wire Protocol

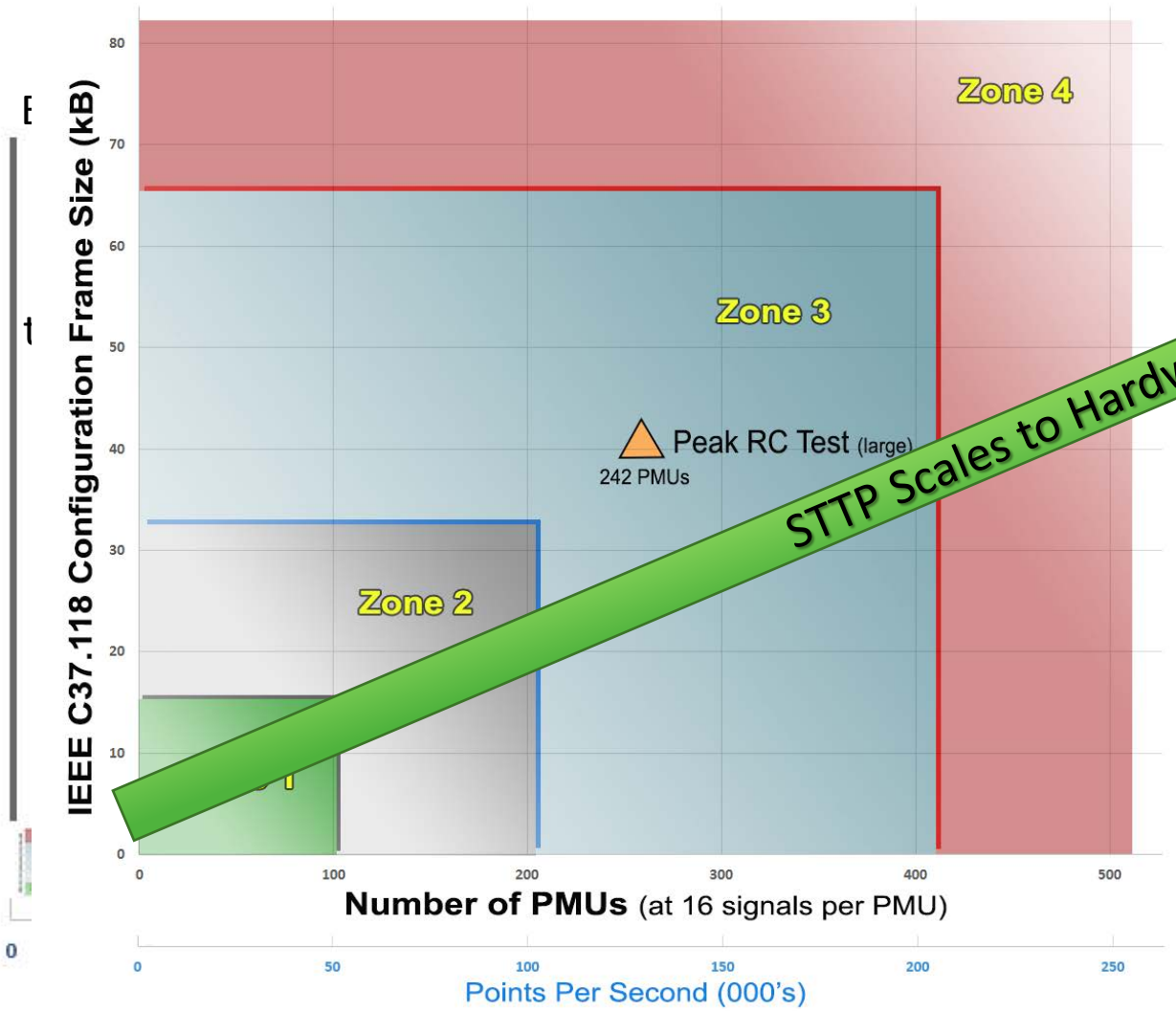


wing

h the

* Methods to impler use cases where a “no comma EE C37.118.

Difference: Scalability



4 IEEE C37.118 V1 & V… figuration frame size m… t (65K). A second strea… st be created

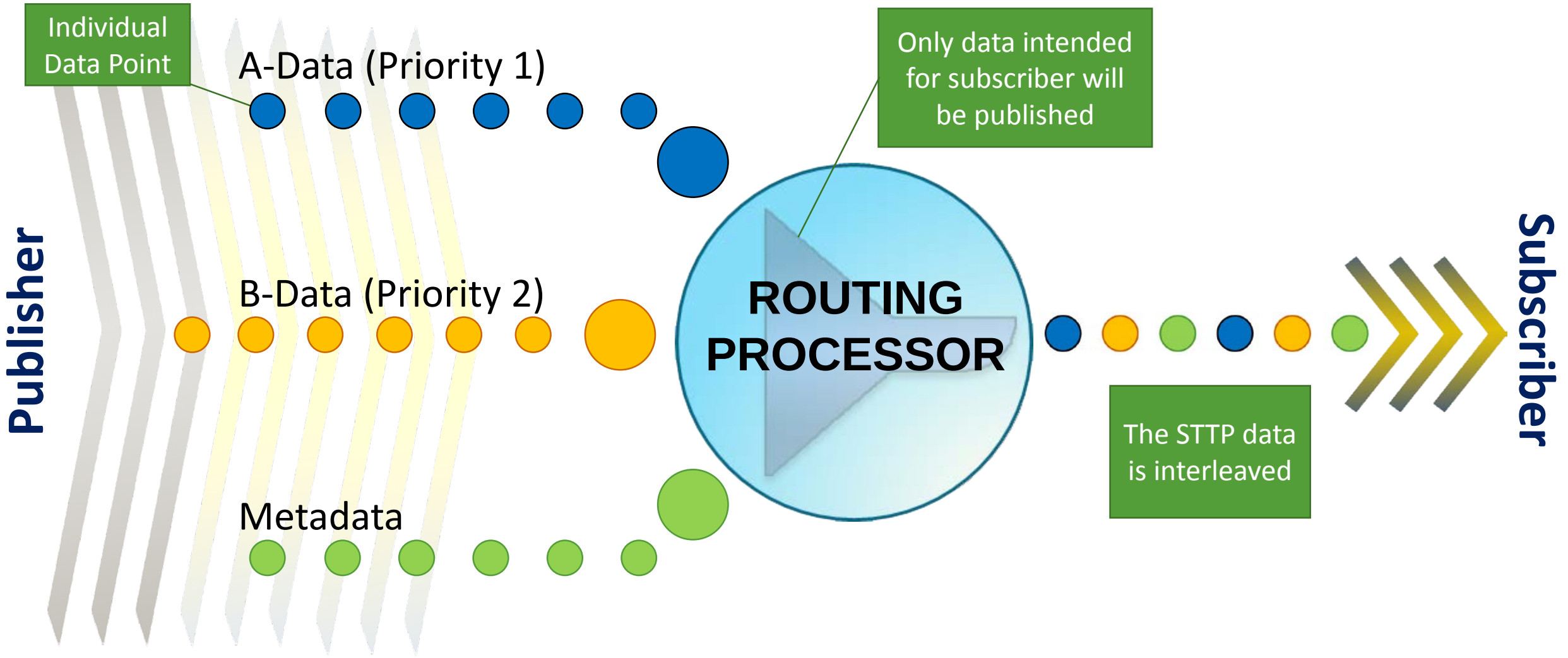
…creased data loss and latency. Purpose-built / allocated networks typically required

2 Data loss and latency issues begin to appear. Network tuning may be required

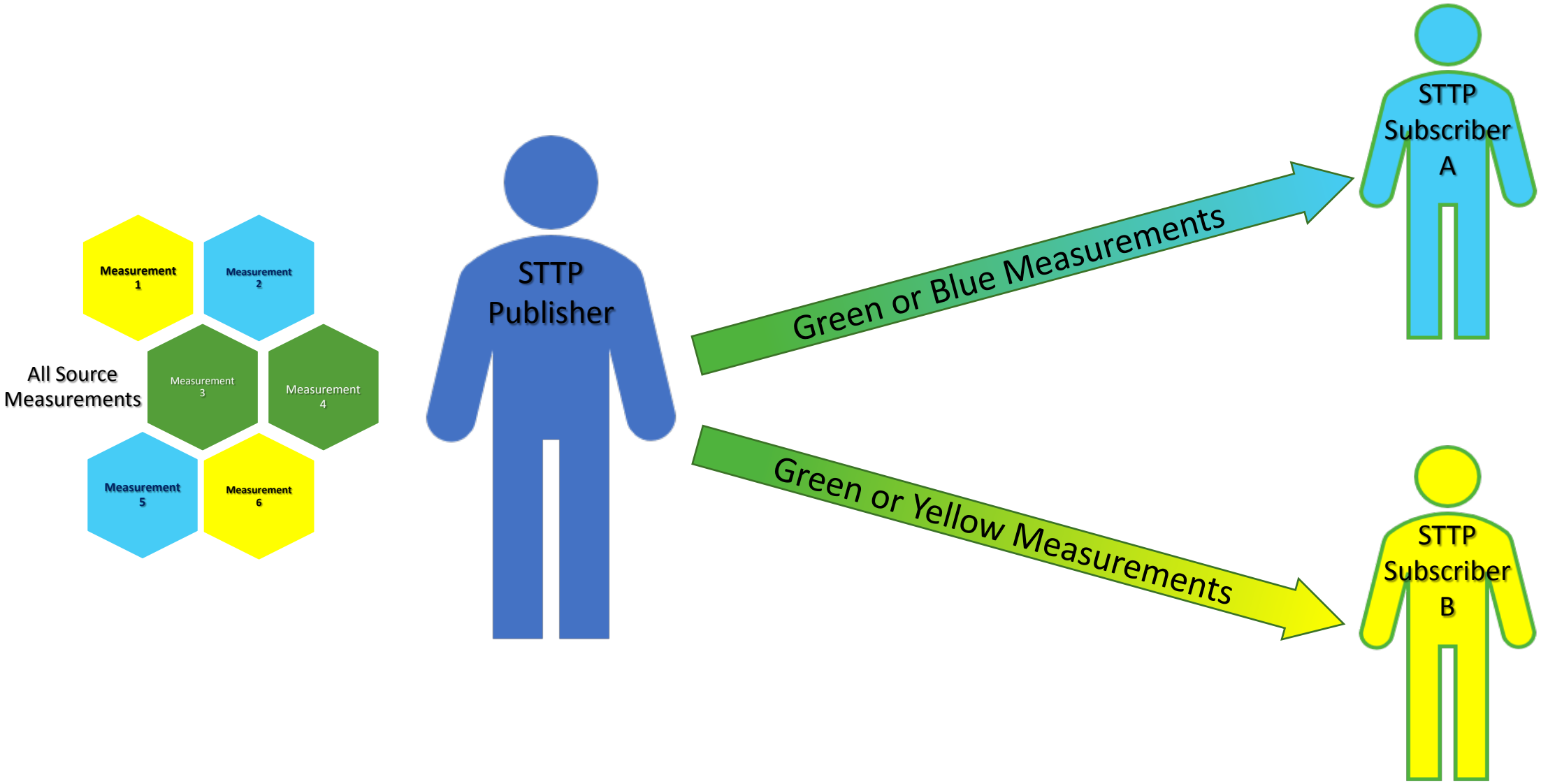
1 Issues, if any, are easy to resolve

STTP handles from 3 to 5 million points per second per connection on common hardware.

Difference: Publish / Subscribe Model



Difference: Publisher Data Access Control



Difference: IP Level Security

- Security at IP Layer
  - ***TCP***: Primary security is added at the socket using industry standard Transport Layer Security (TLS or SSL). X.509 certificates are used to authenticate connections and provide encryption through public key infrastructure.
  - ***UDP*** (optional): When existing command channel is secured with TLS, UDP uses AES symmetric encryption with keys exchanged over the TLS secure channel.

Difference: Configurable Connection Origin

- Two Types of Connections Supported

  Forward

  - *Subscriber connects to Publisher* – typical operation where a listening server-based publisher with connecting client-based subscribers

  Reverse

  - *Publisher connects to Subscriber* – operation where client-based publisher connects to listening server-based subscriber; used to cross security zones in desired direction

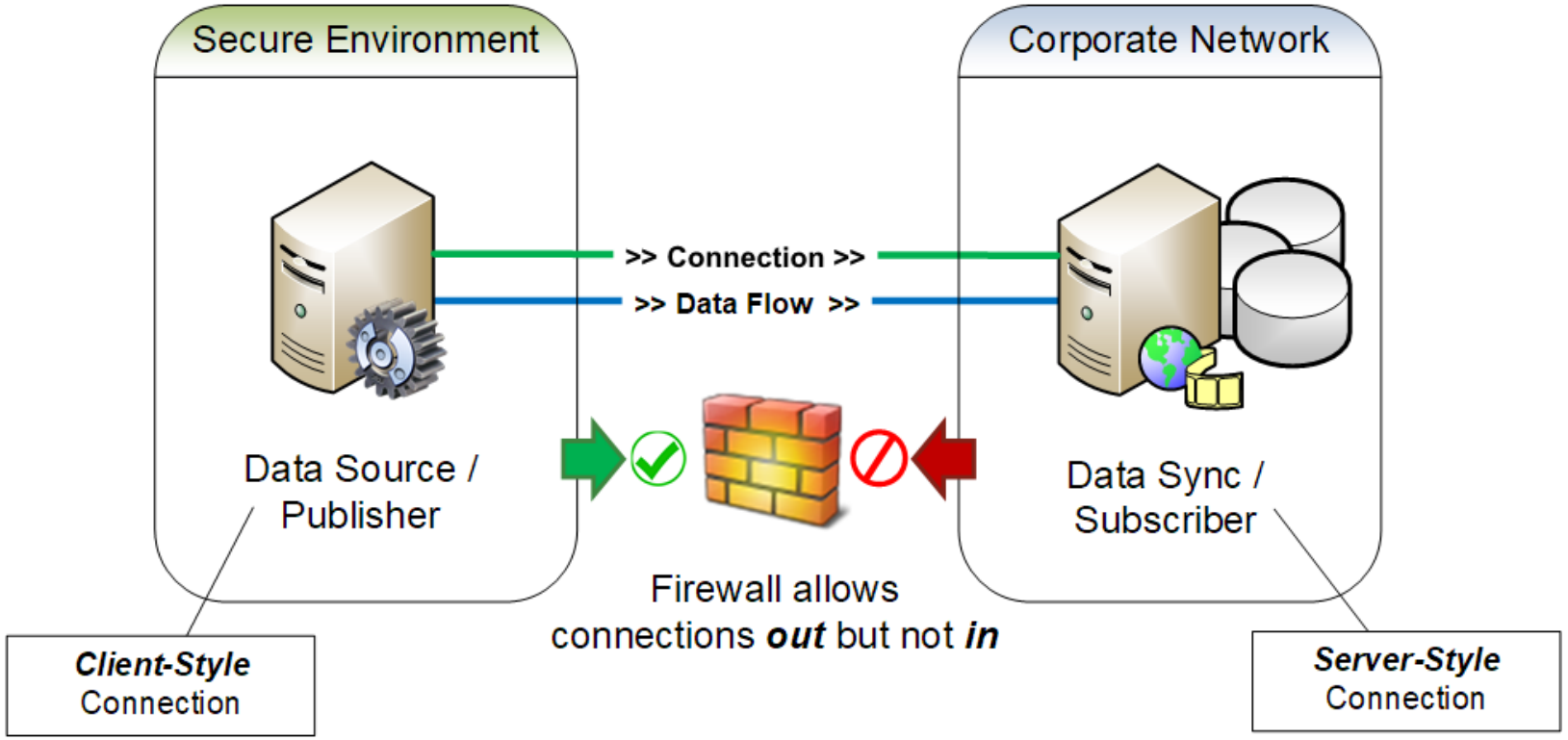
- Bidirectional Communications Allowed
  - Once connection is established, publisher/subscriber functions can operate in either direction over the single connection

Difference: Configurable Connection Origin

- Publisher and Subscriber operations are "*functions*" in STTP – not "*objects*"
- As such, a publisher "*sends*" data and a subscriber "*receives*" data – always

The Wire Protocol

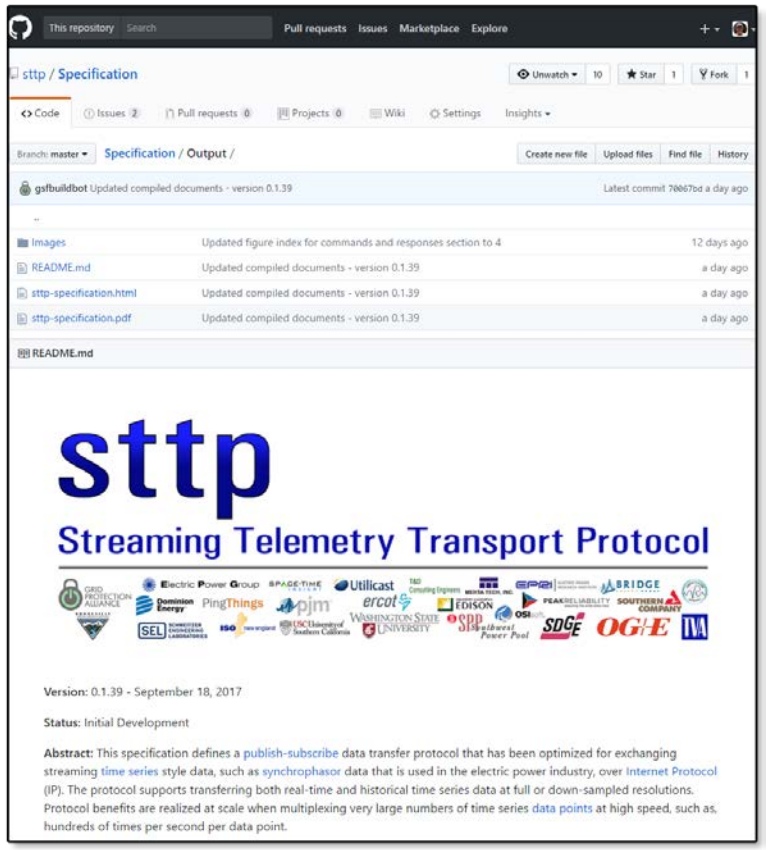
Crossing Security Zone with Reverse Connection



How is the Project Team Advancing STTP?

- Documenting the Specification
- Developing the General Use API
- Conducting Demonstrations
- Participating in IEEE Standardization

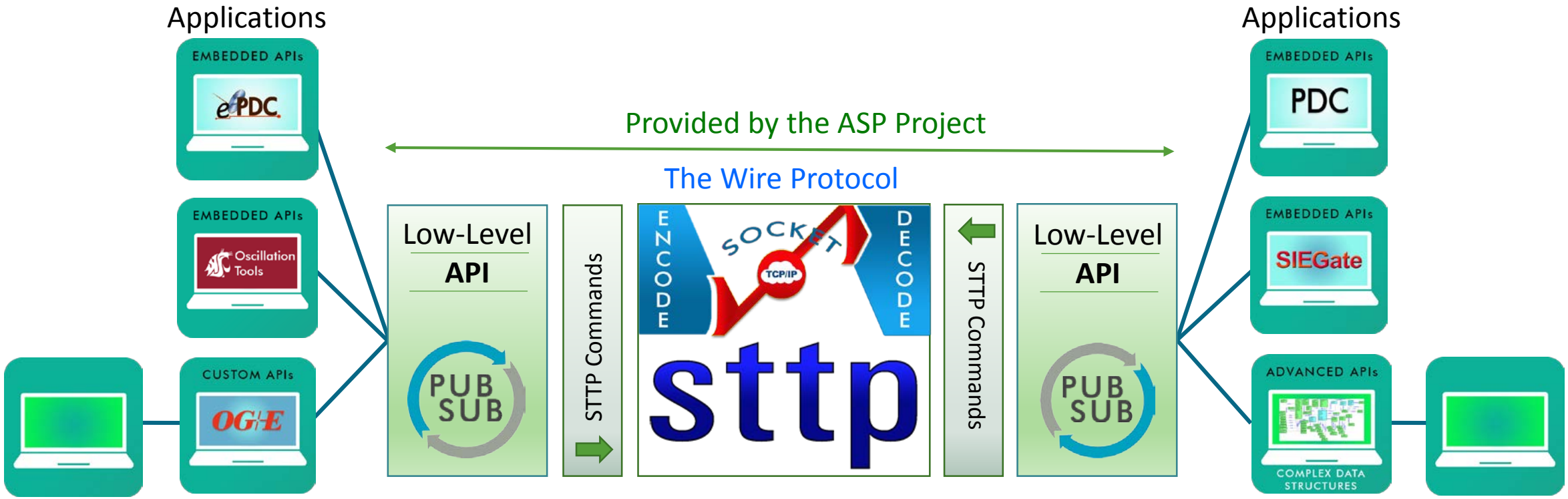
Advancement: Documenting the Specification



https://github.com/sttp

- Specification development is open on GitHub:
  - https://github.com/sttp/Specification
- Daily builds of specification are available in PDF, HTML and GitHub markdown formats
- Topics include:
  - Protocol Overview
  - Establishing Connections
  - Commands and Responses
  - Compression
  - Security
  - *among others*
- Anyone can propose an edit with a pull-request
  - See "How to Contribute" on spec site for details

Advancement: Developing the General Use API

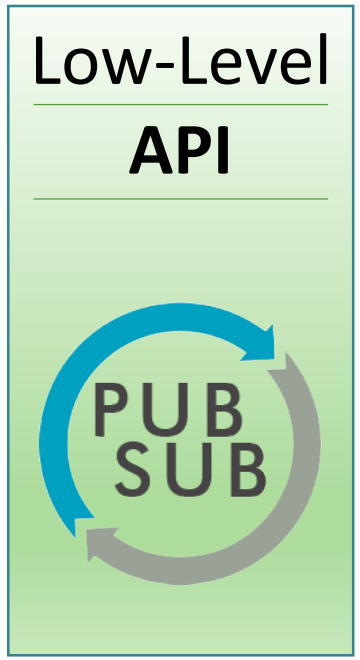


Key STTP Requirements:

- Performant Data Exchange at Scale
- Extensible Metadata
- Access Control and Security
- Bidirectional Connectivity

Advancement: Developing the General Use API

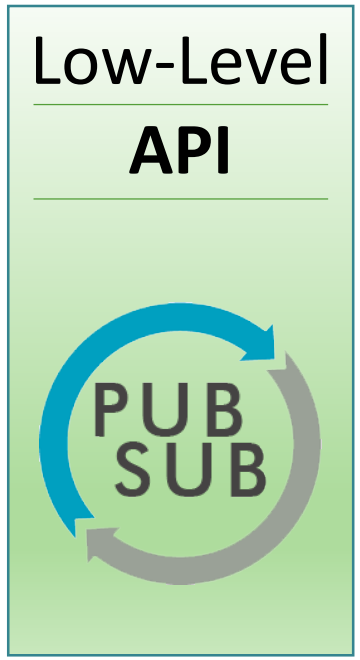
Example API Methods



- Publisher
  - *Methods*
    - Connect
    - DefineMetadata
    - Disconnect
    - DisconnectSubscriber
    - SendData
  - *Callbacks / Events*
    - SubscriberConnected
    - SubscriberSessionEstablished
    - SubscriberDisconnected
- Subscriber
  - *Methods*
    - Connect
    - Disconnect
    - RequestMetadataTables
    - RequestMetadata
    - Subscribe
    - Unsubscribe
    - SecureDataChannel
  - *Callbacks / Events*
    - ReceivedMetadataTables
    - ReceivedMetadata
    - ReceivedDataPoints

Advancement: Developing the General Use API

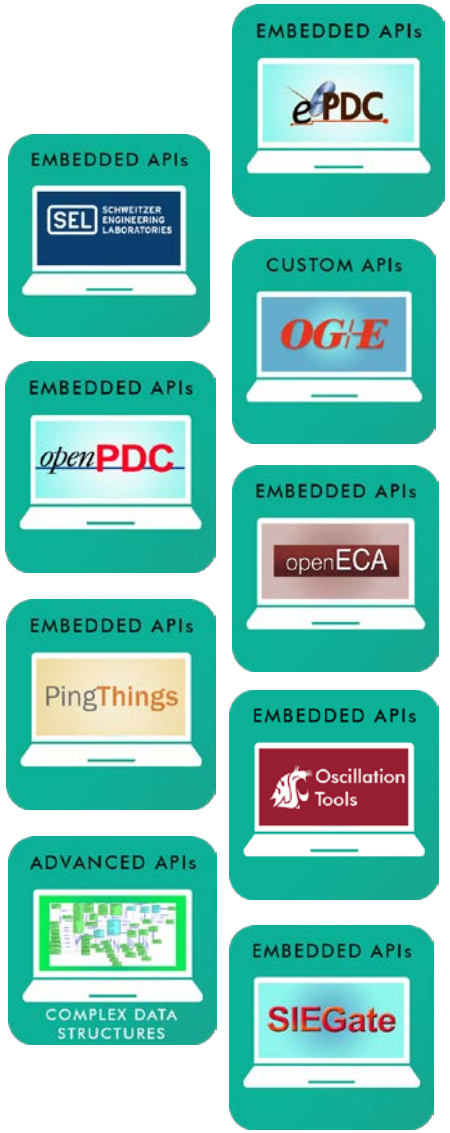
Example Metadata



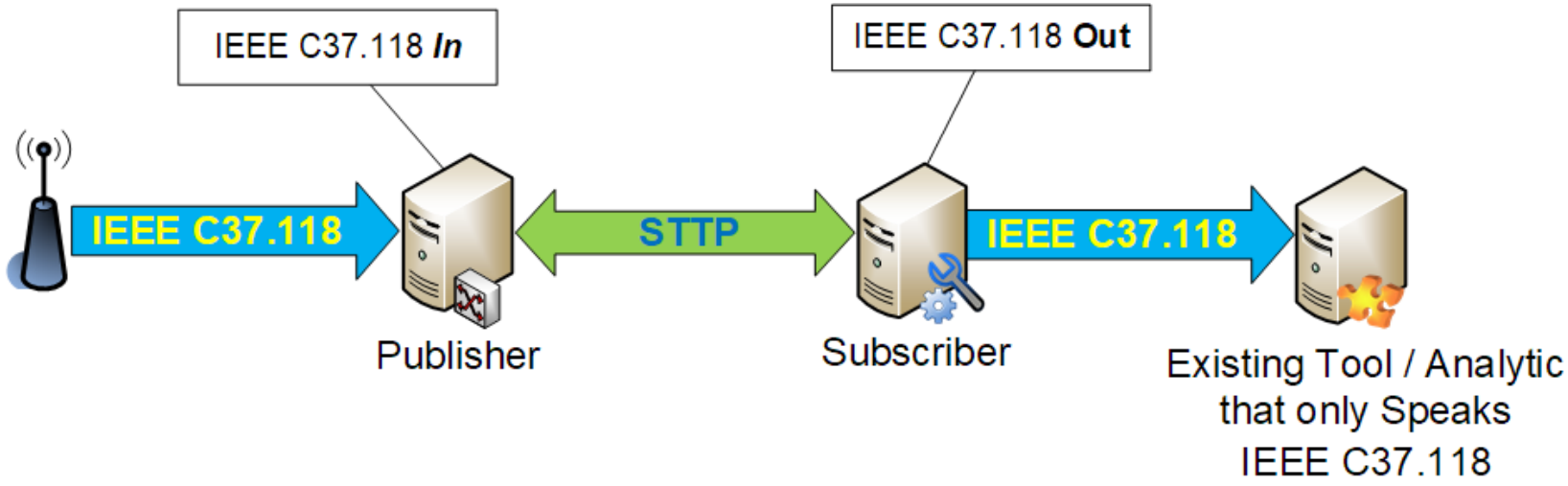
- Core DataPoint Metadata
  - Point ID (guid)
  - Device ID (guid)
  - Tag (string)
  - AlternateTag (string)
  - Description (string)
  - Enabled (bool)
  - Created (date-time)
  - Updated (date-time)
- Device Metadata
  - Device ID (guid)
  - Name (string)
  - *etc.*
- Synchrophasor Metadata
  - Point ID (guid)
  - SignalReference (string)
  - Protocol (string)
  - SignalType (string)
  - EngineeringUnits (string)
  - PhasorType (string)
  - Phase (string)
  - DataRate (float)
  - *etc.*

Advancement: Developing the General Use API

Initial Common Use Case

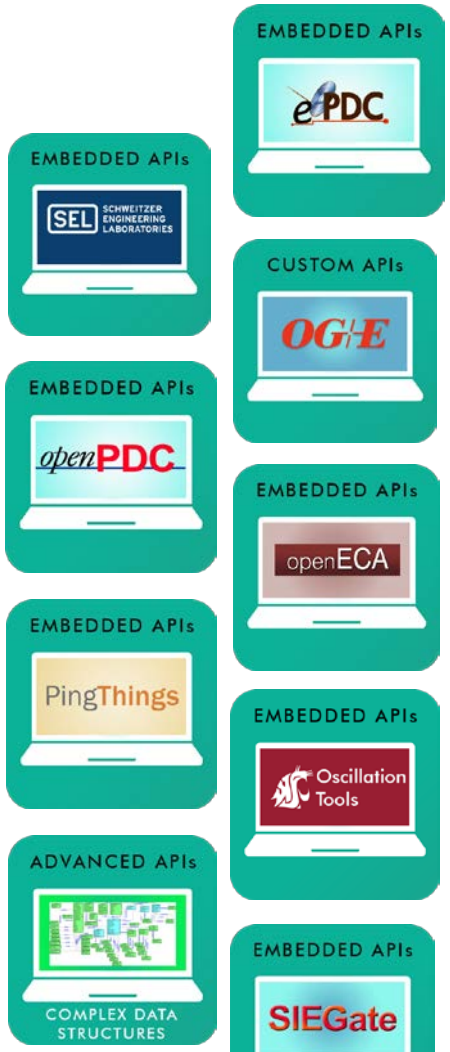


- Complex Structure Encoding (e.g., IEEE C37.118)
  - Includes, as needed, data concentration at final consumer



Advancement: Developing the General Use API

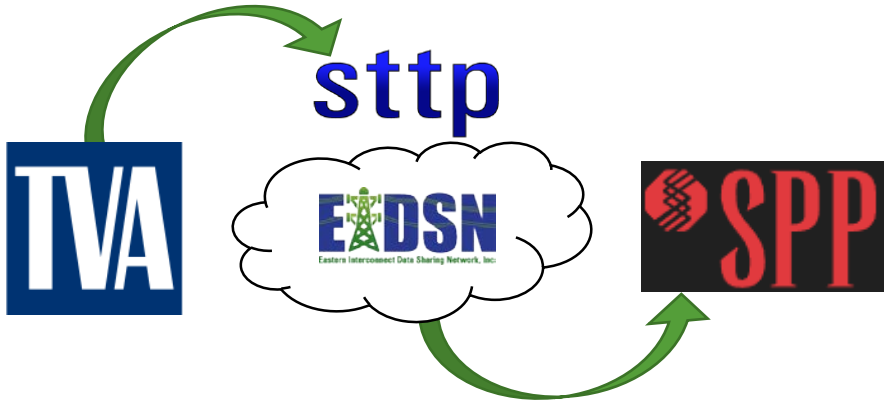
Advanced Use Cases



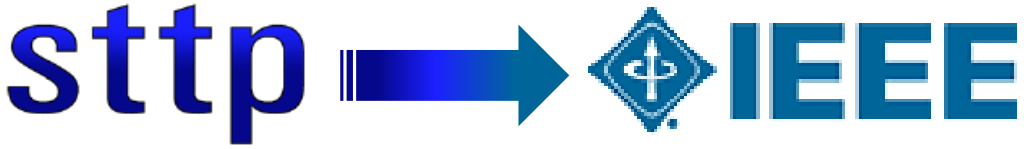
- Advanced Data Logic
  - Variable distribution of redundantly measured values
  - Blue-sky state data reduction (for apps that desire this)
- Gateway transmission of other protocol data
  - ICCP, DNP3, Modbus, OPC, OpenFMB
- Dynamic Data Volume
  - Adjust data publication volume based on system conditions, e.g., sending more information when an event has been detected for increased monitoring and detail (where desired)

Advancement: Conducting Demonstrations

- WSU Tools Demo
  - Tools testing and observations with STTP, e.g., memory utilization, data loss, CPU loading and impacts on running analytics
  - Testing at TVA, SPP, OG&E and SDG&E
- EPG Tools Demo
  - Integration and comparison testing of ePDC for observations with STTP, e.g., CPU loading, data loss, latency and memory impact when receiving STTP data
  - Testing at PJM and Dominion
- EIDSN Transfer Demo
  - Testing TVA → SPP with
    - IEEE C37.118, and
    - STTP over TCP
  - Recording and comparing results with a real-world data transfer



Advancement: Participating in IEEE Standardization



STTP on track to become:

IEEE 2664

This year the IEEE P10 STTP working group was established to develop a project authorization request (PAR).

The PAR was approved by the IEEE-SA New Standards Committee on September 27, 2018 and given a proposed IEEE standard number of ***P2664***.