

Antelope CD1.1 Tools

Kent Lindquist and Danny Harvey
Boulder Real Time Technologies

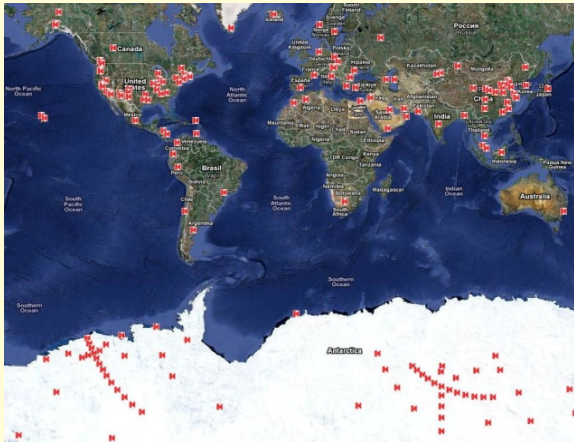
May 2017




Introduction - KMI

Kinemetrics, Inc.

- Founded in 1969
- OYO Corp owned in 1991
- ISO9001 since 1999
- \$35M FY2012 revenue
(mostly international)

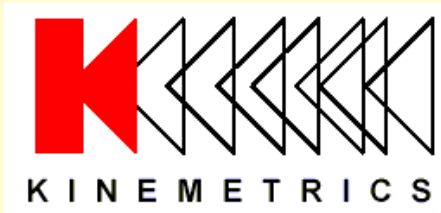


HQ's in Pasadena CA with
Sales and Project offices in
Switzerland & Abu Dhabi

 The image cannot be displayed. Your computer may not have enough memory to open the image, or the image may have been corrupted. Restart your computer, and then open the file again. If the red x still appears, you may have to delete the image and then insert it again.



Introduction – KMI Team



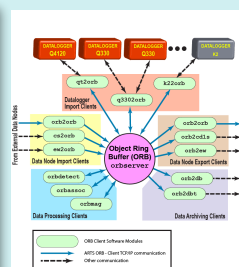
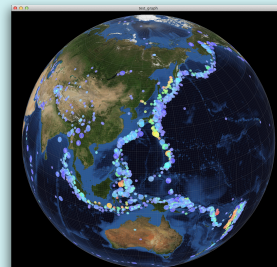
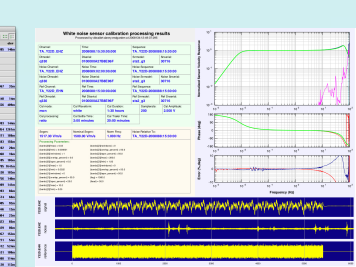
Designs and manufactures sensors and digitizers – Provides complete systems design, installation and operations



Designs High-End Digitizers



Designs High-End Sensors

A table displaying data processing results, with columns for time, sensor ID, and various data values. The table is color-coded with red and green rows.

Kinematics / BRTT

Comprehensive Hardware, Software, and Services

Kinematics Systems Solutions

- Turnkey complete systems including enterprise-class computing centers and full communications

Kinematics Hardware Manufacturer

- World class Kinematics and Quanterra dataloggers
- World class Kinematics, Metrozet and Streckeisen sensors

BRTT Software Developer

- World class acquisition software for all Kinematics hardware products
- Proven track record for large networks with difficult remote deployments (USArray)
- World class, comprehensive automated and interactive seismic processing software
- Data neutral architecture for support of non-seismic environmental monitoring networks
- Extraordinary Command & Control capabilities with SOH displaying

Kinematics Services

- Complete systems procurement, installation and training including all aspects of both hardware and software
- Network operations



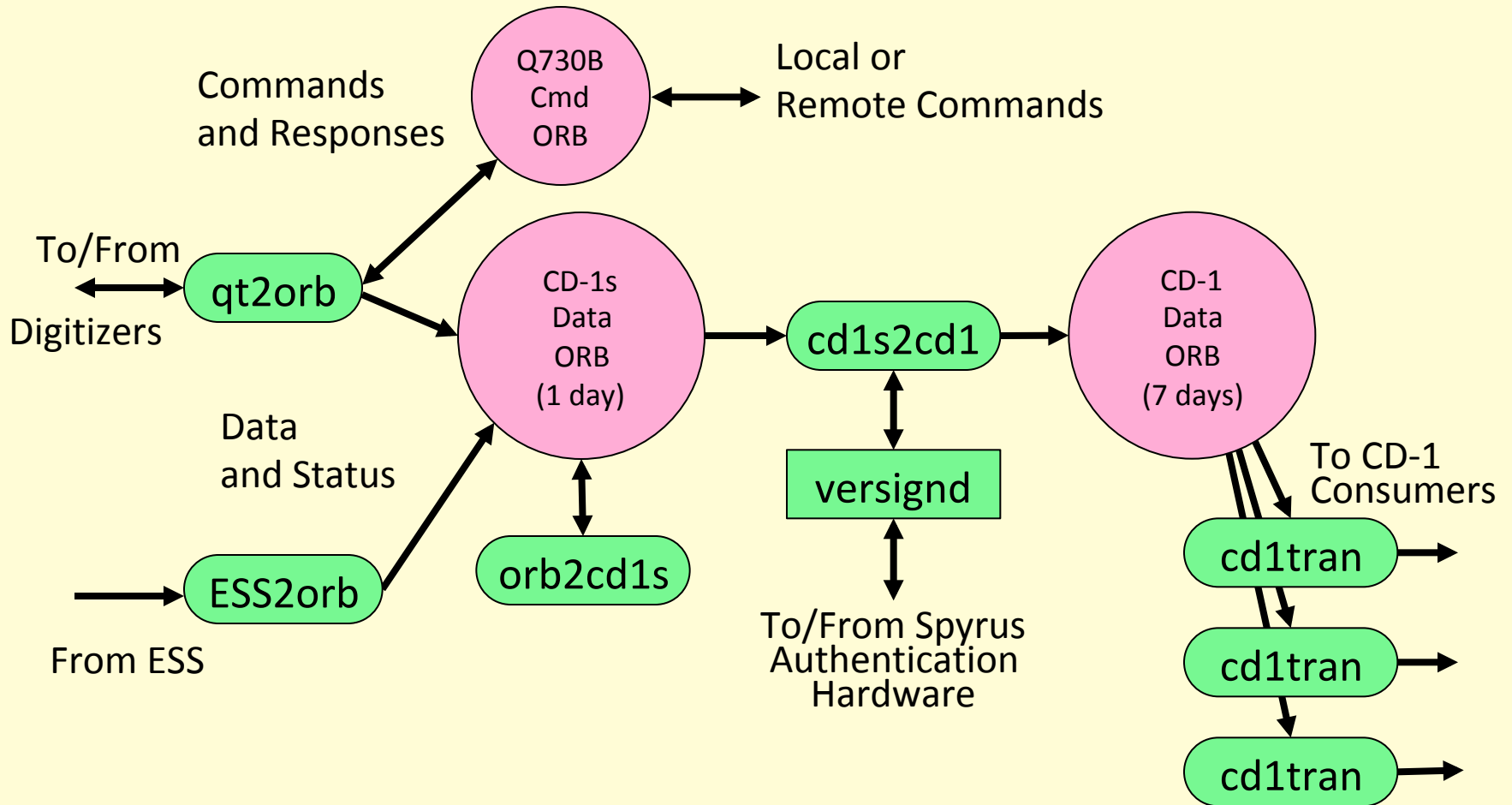
What is CD1.1

- Data transmission format and protocol for Continuous seismic waveform Data (hence “CD”) used by the International Nuclear Test-ban Monitoring Community
- TCP-based mechanism to transmit data in “Frames”
- Last-In, First-Out (“LIFO”) transmission prioritizes most recent data
- Acknowledgment mechanism provides for re-request of missed data packets
- Successor to “CD1.0” Protocol

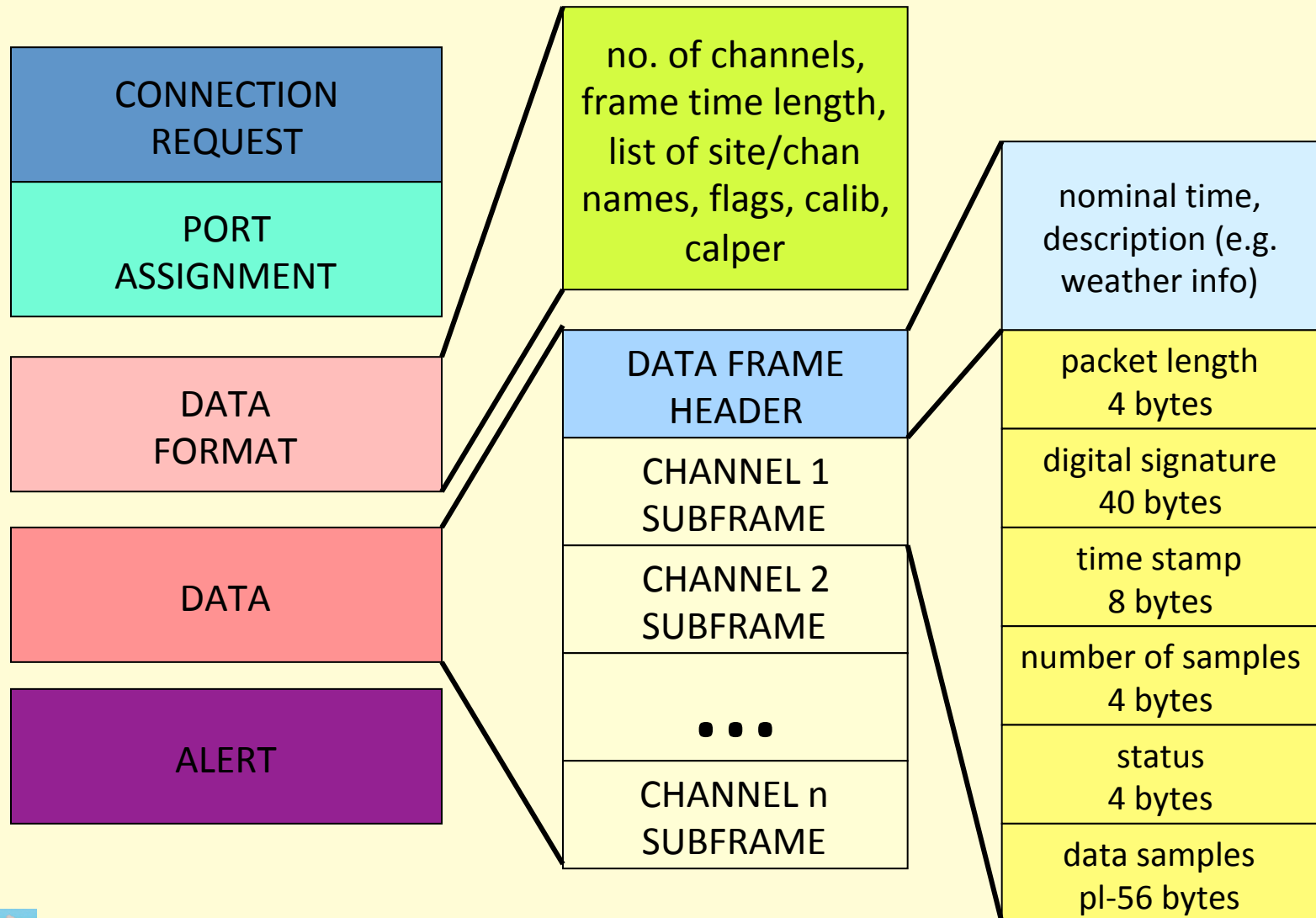
(Extremely) Short History

- BRTT Implemented CD1.0 Support years ago
- CD1.1 is now established for CTBTO Operations
- We have made our first foray into CD1.1 Support due to ongoing interest, basing our CD1.1 work on the IDC3.4.3 Protocol document and our previous CD1.0 system architecture

Architecture—CD1.0 Historic (SDAS)



CD-1(.0) Frame Definitions



Basis for Current Work



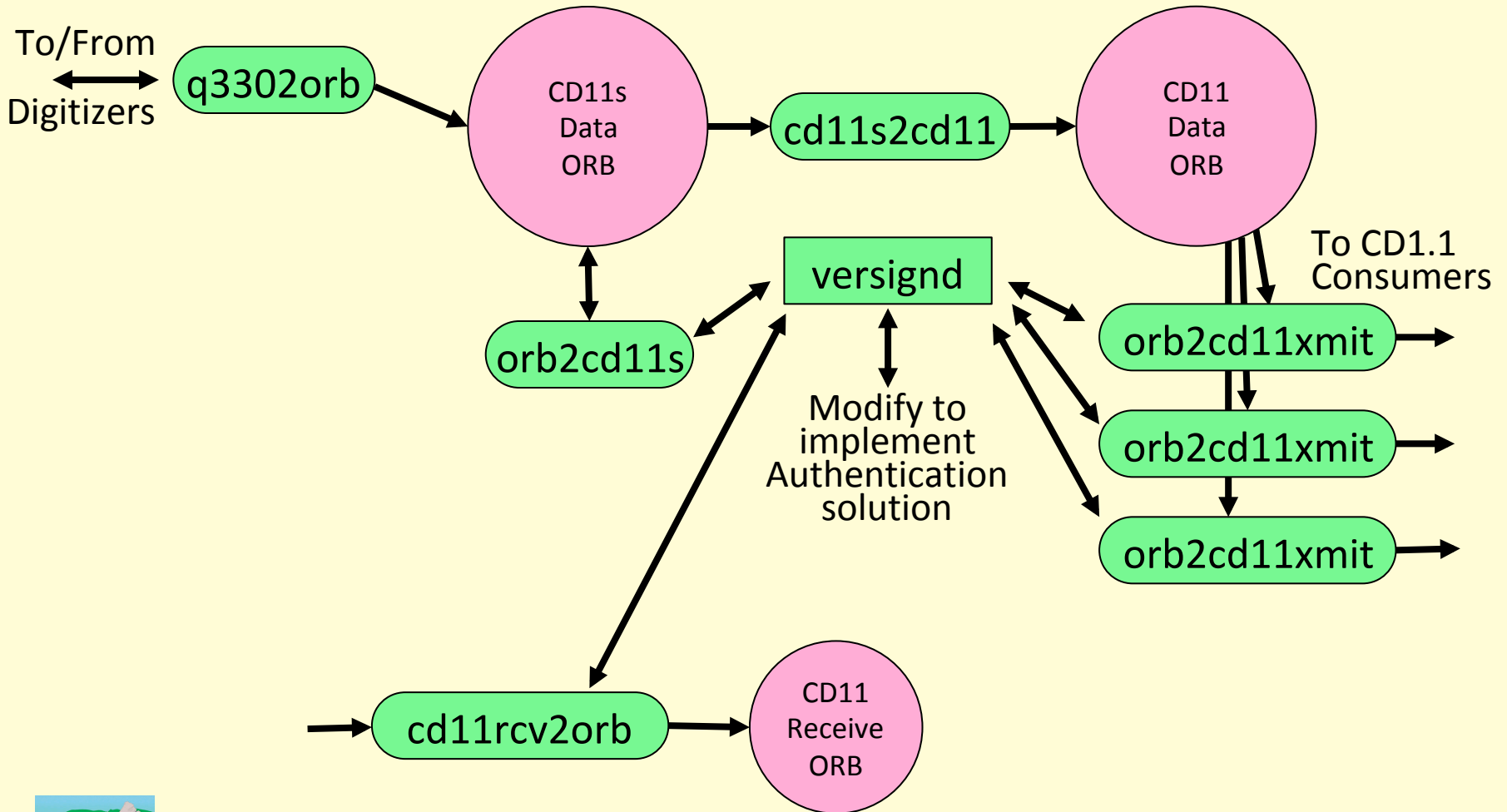
The image cannot be displayed. Your computer may not have enough memory to open the image, or the image may have been corrupted. Restart your computer, and then open the file again. If the red x still appears, you may have to delete the image and then insert it again.

**CTBTO (2002). Formats and
Protocols for Continuous Data
CD-1.1, Document ID IDC 3.4.3,
*Preparatory Commission for the
Comprehensive Nuclear-test-ban
Treaty Organization.***

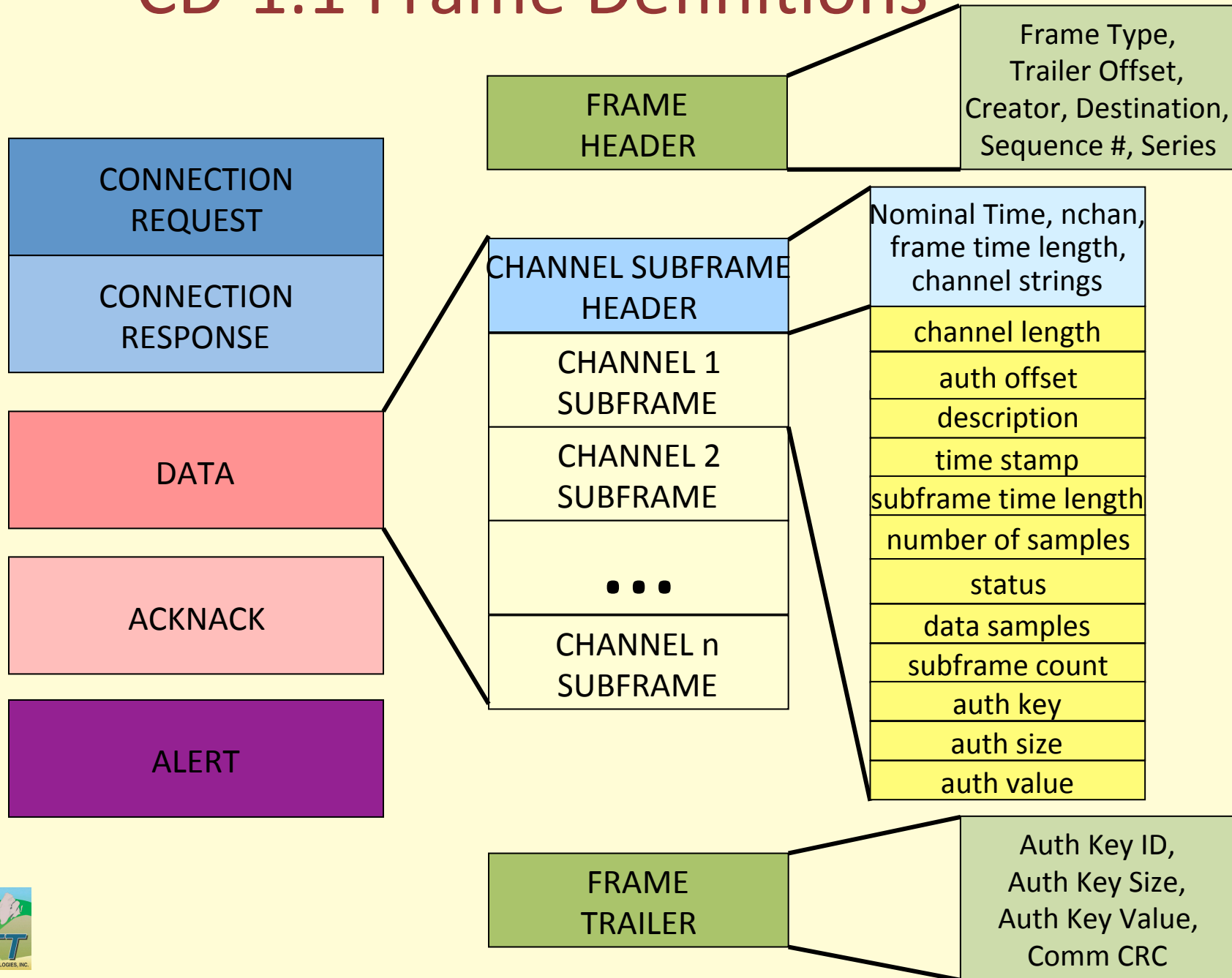
What We've Done

- Incorporated CD1.1 sub-frames (“./CD11S”) and CD1.1 Frames (“./CD11”) into the Antelope Packet library – CD1.1 frames encapsulated as orb packets
- Written two programs to generate these: **orb2cd11s(1)** to make CD1.1 sub-frames; **cd11s2cd11(1)** to combine those into CD1.1 Frames
- Written **orb2cd11xmit(1)** to transmit these as a CD1.1 Data Producer
- Written **cd11rcv2orb(1)** test-jig to receive these as a CD1.1 Data Consumer
- Written the **versignd(1)** template daemon process for end-users to implement authentication by preferred software/hardware
- Written an object-oriented **libcd11 (cd11(3))** to create and interact with CD1.1 protocol frames
- Leveraged the new object-oriented **liboorb (oorb(3))** to interact with orbserver in processing these packets

Architecture – CD1.1



CD-1.1 Frame Definitions



CD1.1 Frame Types

- **Supported** by this Antelope version:
 - *Connection Request*
 - *Connection Response*
 - *Data*
 - *AckNack*
 - *Alert*
- **Not Supported** by this Antelope version:
 - *Option Request*
 - *Option Response*
 - *Command Request*
 - *Command Response*
 - *CD1Encapsulation*

CD1.1 Communication Protocol



The image cannot be displayed. Your computer may not have enough memory to open the image, or the image may have been corrupted. Restart your computer, and then open the file again. If the red x still appears, you may have to delete the image and then insert it again.

libPkt modifications

- New “`./CD11S`” orb-packet format encapsulates verbatim CD1.1 Channel sub-frames
- New “`./CD11`” orb-packet format encapsulates verbatim CD1.1 Data Frames minus the Frame Header and Trailer
- A few extra fields are added in front to make the packet self-describing, such as `segtype`, `calib`, and `calper` for channel sub-frames, and `nchannels`, `net,sta`, `chan`, `loc`, `segtype` for each channel for data frames

libcd11

- ***cd11(3)*** library provides C++ Objects to handle CD1.1 Frames and Interactions
 - ***CD11Socket(3)*** class and subclasses implement transmission and reception of CD1.1 frames over TCP sockets
 - ***CD11Frame(3)*** class is the main implementation wrapper around IDC3.4.3 frame definitions. Each Frame has
 - ***CD11FrameHeader(3)***
 - ***CD11FramePayload(3)***
 - subclassed by type e.g. ***CD11DataPayload***, ***CD11AckNackPayload*** etc.
 - ***CD11FrameTrailer(3)***
 - ***CD11Config(3)*** class supports ***cd11(3)*** object and program configuration

Authentication -- versign

- Daemon process *constructs or verifies data signatures*
- Intended to be modified to accommodate specified data-signer methods, e.g. hardware authentication devices
- *Systems integrator must write a plugin subroutine* that gets called by **versign(1)** whenever data must be signed or verified, implementing the desired signature method
- **versign(1)** ships with one ‘test’ signer (a simple “MD5” message digest)
- **orb2cd11xmit(1)**, **cd11rcv2orb(1)** have *versign_address* (IP:port) and CD1.1 *auth_key_identifier* in their parameter-files; they call **versign(1)** through **versign(3)** routines
- **versign(1)** is independent of Antelope
- **versign(1)** source-code available

Packet CD1.1 Sub-Frame Creation

- ***orb2cd11s(1)*** Converts arbitrary Orb data packets into CD1.1 Channel Sub-Frames
- Operates in continuous mode
- Formats data into fixed-time-length frames
- Emits “.* /CD11S” orb packets
- Optionally apply standard CTBT/IMS Canadian Compression
- Data authentication signatures can be added via ***versignd(1)***
- Defines State-of-Health bits

Packet CD1.1 Frame Creation

- ***cd11s2cd11***(1) Assimilates CD1.1 channel sub-frame packets into CD1.1 Data Frame packets
- Operates in continuous mode
- Reads orb-packets output by ***orb2cd11s***(1)
- Emits “.* /CD11” orb packets
- Output Orb packets are verbatim CD1.1 data frames minus the standard CD1.1 Frame Header and Trailer

Transmitter

- ***orb2cd11xmit*(1)** reads “.* /CD11” format orbserver packets from an orb
- Computes and applies CD1.1 Frame Headers and Trailers
- Manages CD1.1 AckNack and Alert frames
- ***orb2cd11xmit*(1)** serves as a CD1.1 “Data Provider” to send these as CD1.1 frames to a “Data Consumer”
- Signature creation for packet authentication outsourced to ***versignd*(1)** daemon
- Four levels of verbosity [-v[v[v[v]]]] with increasing levels of debug info, up to full packet-report dump
- CD1.1 packet sequence number derived from epoch time of orb packet – allows retrieval and retransmission of missed packets
- Retransmits packets missed as reported by AckNack frames
- Interleaves packet retransmission with LIFO processing

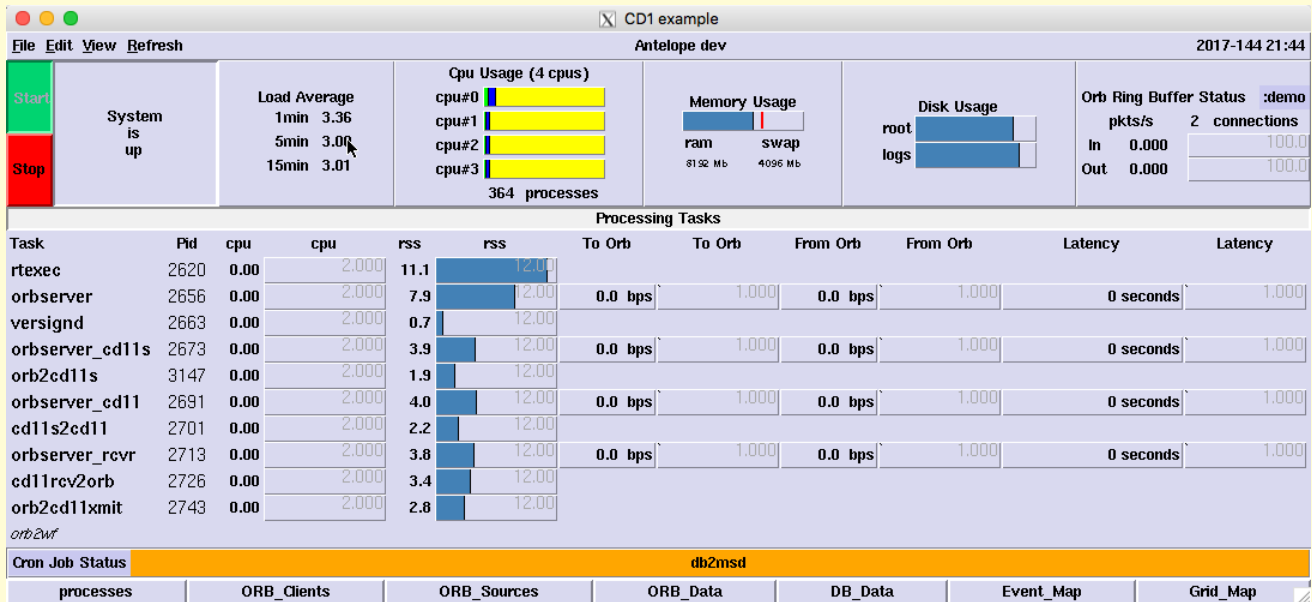
Receiver

- ***cd11rcv2orb(1)*** *Test Jig*: Serves as a CD1.1 “Data Consumer”: receives CD1.1 data frames, puts them on orbserver as “.* /CD11” packets
- Not our focus this round but it’s hard to write a transmitter without a receiver
- Signature verification outsourced to ***versignd(1)***
- Optional *Statefile* keeps non-volatile list of all gaps in the data stream for re-request
- Sends CD1.1 *AckNack* Frames periodically

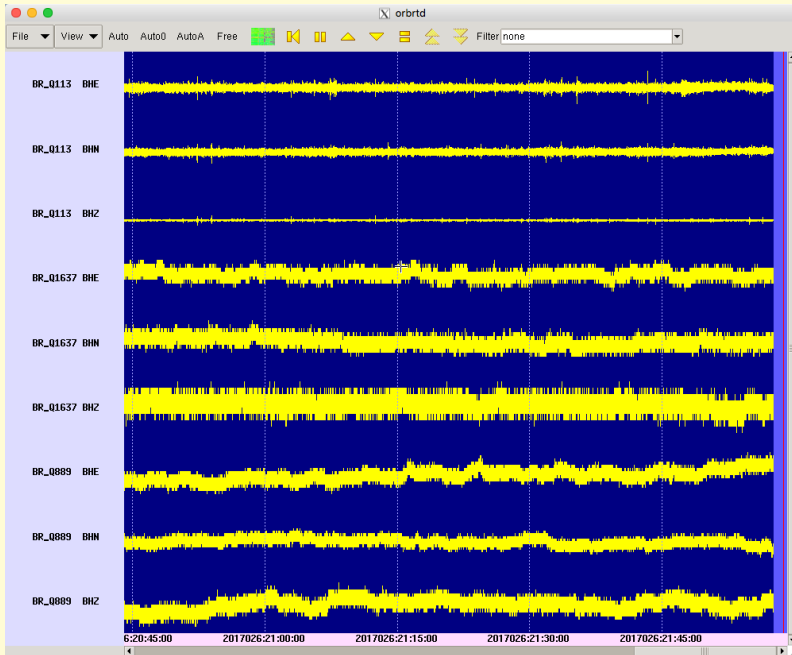
Testing

- Internal consistency:
 - Antelope transmitter works with Antelope receiver
 - Test data from Q330 in BRTT equipment room
- Reception by NDC-in-a-box:
 - Remote NDC-in-a-box receiving data from our transmitter
 - Bypassing authentication (checksums match)
 - Logs showing incoming data

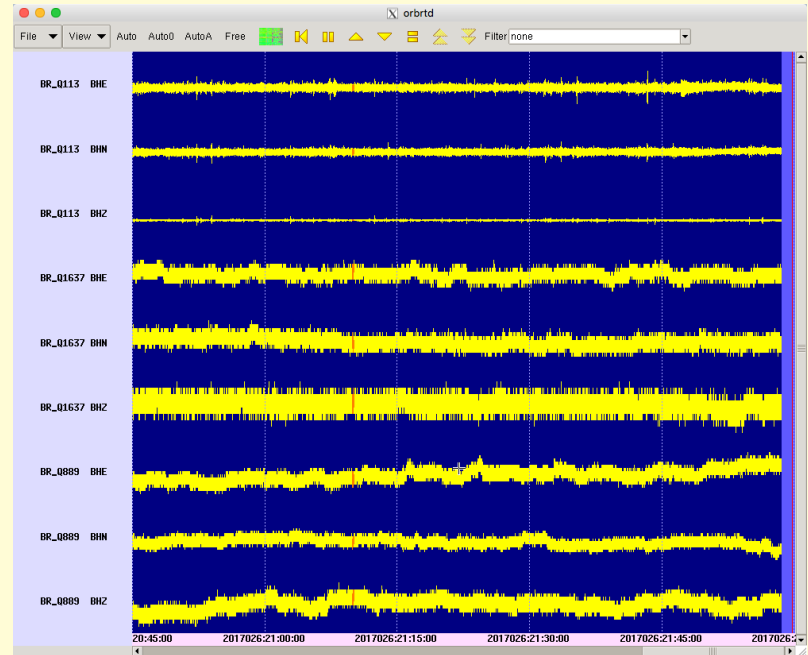
CD1.1 demo system example



Internal Consistency Testing



Test data from Q330 in office



Same data emerging from cascade:

- > orbserver
- > orb2cd11s
- > cd11s2cd11
- > orb2cd11xmit
- > cd11rcv2orb
- > orbserver

Documentation

User level:

- ***orb2cd11s(1)***
- ***cd11s2cd11(1)***
- ***orb2cd11xmit(1)***
- ***cd11rcv2orb(1)***
- ***versignd(1)***

Programmer level:

- ***cd11(3)***
- ***CD11Config(3)***
- ***CD11Frame(3)***
- ***CD11FrameHeader(3)***
- ***CD11FramePayload(3)***
- ***CD11FrameTrailer(3)***
- ***CD11Socket(3)***
- ***oorb(3)*** and subsidiary pages
- ***versign(3)***

Future options

- Transmitter/Receiver burn-in
- dlmon support
- Dynamic command-and-control facilities
- Further vetting against other packages
 - requires test collaborations



Thank You!

Questions?