An Overview of Compressed Video Transport Protocols over IP

Ciro A. Noronha, Ph.D.

Director of Technology, Compression Division

Cobalt Digital

"The nice thing about standards is that you have so many to choose from."

Andrew Tanenbaum, *Computer Networks*, 2nd ed., p 254

- There are many protocol options in use today for transporting compressed video over an IP network
- We will present an overview of these options, and discuss where they are best applied

COBALT®

Video Delivery Parameters

- A compressed video delivery protocol may need to care about:
  - Are we delivering to one or multiple receivers?
  - Is network latency a concern?
  - Will the network reliably deliver the packets, or will there be packet loss? (some networks drop packets…)

COBALT

The Effect of Packet Loss

- Video compression works by removing redundancy from the transmission
  - Every bit of compressed video is very important
- There is a simple way to look at the effect of packet loss:
  - Every packet that is dropped by the network causes a glitch in the video

What is “acceptable” loss?

Assume a 4 Mb/s stream, with 1316-byte packets

Dropping one packet in	Produces a glitch every
1,000	2.6 seconds
10,000	26 seconds
100,000	4 minutes 23 seconds
1,000,000	44 minutes
10,000,000	7 hours 19 minutes

COBALT

Protocol Basics

End-to-end IP applications run on top of one of two protocols:

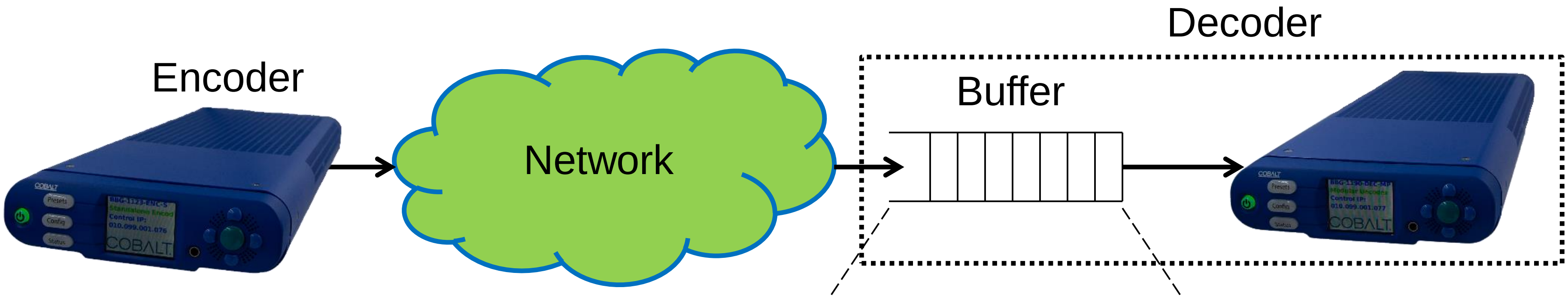
- User Datagram Protocol (UDP)
  - “Raw” network service
  - Packets are delivered as fast as possible, but may be dropped
  - Support for multicast (network replicates the packets)
- Transmission Control Protocol (TCP)
  - “Reliable” network service
  - Flow control, unbounded latency
  - Unicast only (sender has to replicate for multiple recipients)

The Tradeoff

- Fundamentally, there is a tradeoff between LATENCY and PACKET LOSS RESILIENCY:
  - Decoders cannot “wait forever” – packets have expiration dates
  - You can give yourself time to deal with packet loss by pre-buffering before the decoder – the more time you give yourself, the better job you can do to recover from lost packets

COBALT

The Tradeoff



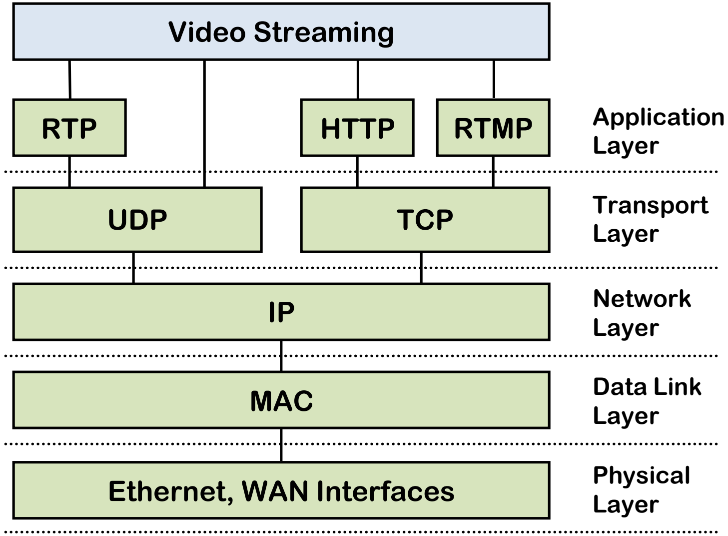
Protocol Latency

Gives you time to recover from lost packets – the more time you have, the better job you can do!

COBALT

Protocol Roadmap

- UDP-based protocols
  - Raw UDP
  - RTP
  - RTP with SMPTE 2022 FEC
  - RTSP
  - SRT
  - RIST
- TCP-based protocols
  - RTSP (tunnel mode)
  - RTMP
  - HLS



COBALT

Raw UDP

- Very simple: just transmit the video in the payload of UDP packets
- Characteristics:
  - Zero protocol latency
  - No packet loss recovery (best effort)
  - Multicast support
- Decoder support: lowest common denominator, supported by professional decoders, IP set-top boxes, software decoders, etc.

COBALT

RTP

- Thin layer on top of UDP, adding timestamps and sequence numbers
- Characteristics:
  - Zero protocol latency
  - No packet loss recovery (best effort)
  - Multicast support
  - Capable of packet re-ordering (currently not very useful)
- Decoder support: mostly professional IRDs and some software decoders

RTP Plus SMPTE 2022 FEC

- Video is sent using standard RTP
- Additional FEC packets are also sent using RTP
- If there is packet loss, the receiver MAY be able to rebuild the lost packets from the received packets and the FEC packets.
- The FEC protocol parameters allow a certain amount of tuning of overhead, latency, and recovery capabilities.

COBALT

RTP Plus SMPTE 2022 FEC

- Characteristics:
  - Non-zero, tunable protocol latency
  - Multicast support
  - Packet re-ordering support
  - May add significant overhead (typical 25%)
  - May be able to work over the Internet
    - Depends on ISP capacities, congestion, and other factors – it is a risk!
- Decoder support: limited mostly to professional IRDs

COBALT®

FEC Examples

Columns	Rows	Recovery Capability	Overhead	Latency @ 2 Mb/s	Latency @ 10 Mb/s
5	5	5 pkts every 25	20%	263 ms	53 ms
10	5	10 pkts every 50	20%	526 ms	105 ms
20	5	20 pkts every 100	20%	1052 ms	211 ms
10	10	10 pkts every 100	10%	1052 ms	211 ms

COBALT

RTSP

- Real Time Streaming Protocol (RTSP) is really a control protocol typically implemented in video servers
  - It exposes a “VCR-like” control interface to start, pause, stop playback
- The RTSP control interface is implemented over TCP
- Using the control interface, the client (decoder) negotiates the streaming parameters
- The actual streaming is done using plain RTP
  - Video and Audio are sent as elementary streams on separate ports
  - Usually no packet loss recovery at the RTP level

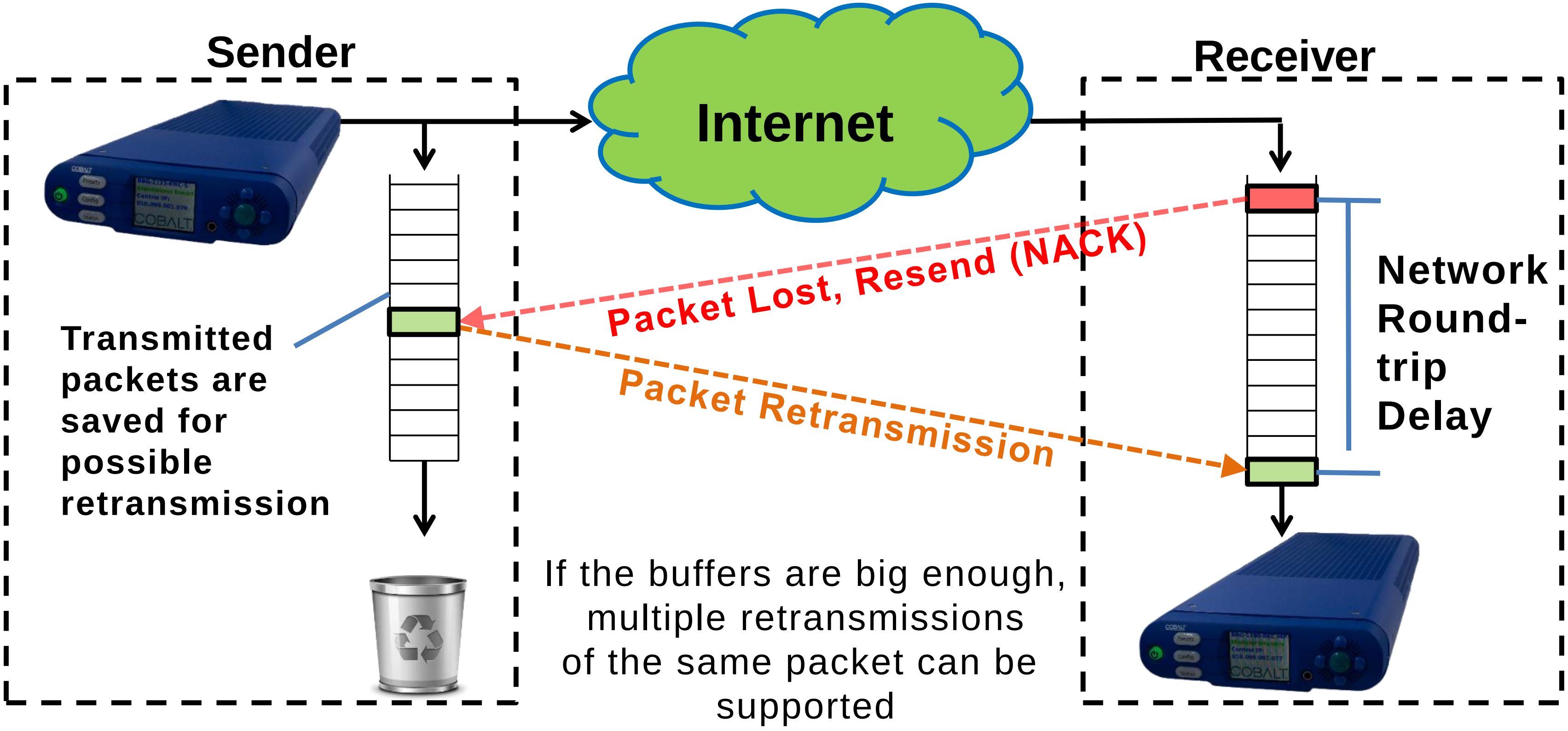
SRT

- Secure Reliable Transport (SRT) is a proprietary protocol developed by Haivision and later placed in the public domain
- Protocol is based on UDT (UDP-based Data Transfer Protocol)
  - Protocol designed for high-speed file transfer over UDP
- Operation:
  - Packets received correctly are acknowledged (similar to TCP)
  - There is an explicit NACK for dropped packets
  - Sender retransmits requested packets or un-acknowledged packets
- Device support:
  - Available in a number of professional encoders and IRDs
  - Available in the VLC software player

RIST

- Reliable Internet Stream Transport (RIST) is a specification published by the Video Services Forum intended for low-latency video contribution/distribution
- Lost packets are recovered using a variant of Selective Retransmission (ARQ – Automatic Repeat reQuest)
- Highlights:
  - Media transmission is done using standard RTP/UDP
  - Packets received correctly are not acknowledged (no flow control)
  - Receiver requests retransmission of lost packets using standard RTCP messages
  - Designed to be firewall-friendly

RIST Illustration



COBALT

RIST Discussion

- In RIST, the latency-reliability tradeoff is fully configurable by the choice of the buffer and number of times a packet can be retried
  - Latency of the protocol can be fine-tuned for the network conditions
- Using RTP as the base protocol ensures compatibility with non-RIST devices
- Supported in a number of encoders, decoders, and gateways from multiple vendors
  - Multi-vendor interoperability
- Supported in the VLC public-domain software decoder

COBALT

TCP-Based Transport

- Quick review of TCP:
  - Connection-oriented: a client explicitly connects to a server; data transmission can go in either direction (or both ways)
  - No multicast support
  - Protocol uses acknowledgments and retransmission to make sure that all bytes are received, *no matter how long it takes*
  - Protocol also provides flow-control – receiving side only acknowledges the data when it is ready to receive more
    - The ability to flow-control an encoder is limited to non-existent
  - Flow control is also used for network congestion

Encoding to a TCP Connection

- The simplest use of TCP is to create a connection between the encoder and the device that is consuming the stream
- Encoder pushes the data through the connection and hopes that the end-to-end bandwidth is enough
  - Buffering required at the encoder …
- Protocols using a raw TCP connection:
  - RTSP (tunnel mode)
  - RTMP

RTSP, Tunneled

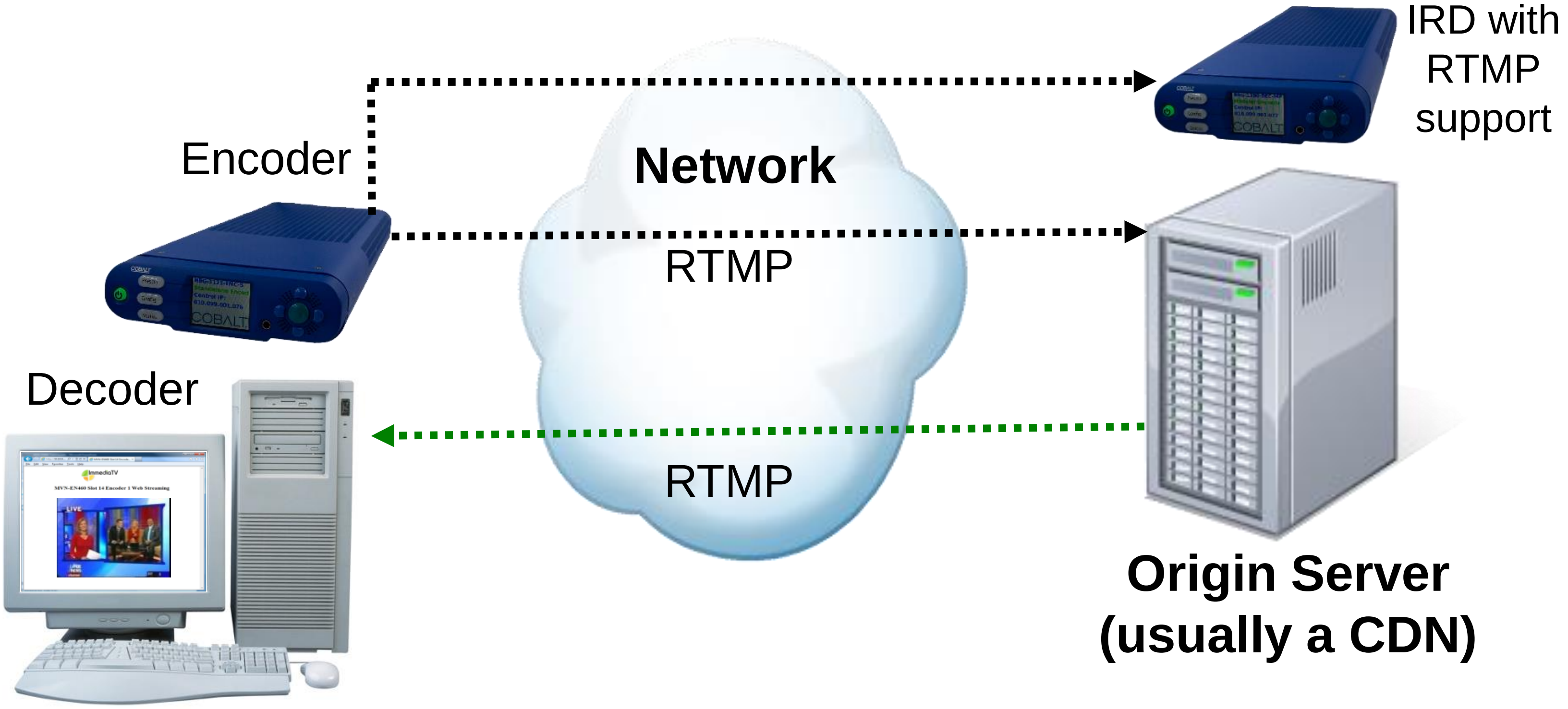
- Basic RTSP is only suitable for local managed networks
  - No packet loss recovery on RTP
  - UDP ports are dynamically negotiated – not firewall friendly
- RTSP has a mode where the RTP data is tunneled over the TCP control connection
  - Same resiliency as TCP, single connection
- Encoder support: mostly built-in encoders in surveillance cameras
- Decoder support: software decoders, some professional IRDs

Real Time Messaging Protocol (RTMP)

- Proprietary protocol designed by Macromedia for its Flash player (later acquired by Adobe)
- Protocol specification was placed in the public domain by Adobe
- Used primarily by Flash players to retrieve content from servers
- Protocol has an option for the client to publish a stream to the server – this is what encoders use
- Protocol is becoming obsolete as it is media-specific
  - Out of the modern encoding standards, only supports H.264 and AAC audio
  - Limited to Flash container format
  - Not very well documented

COBALT

RTMP Operation



COBALT

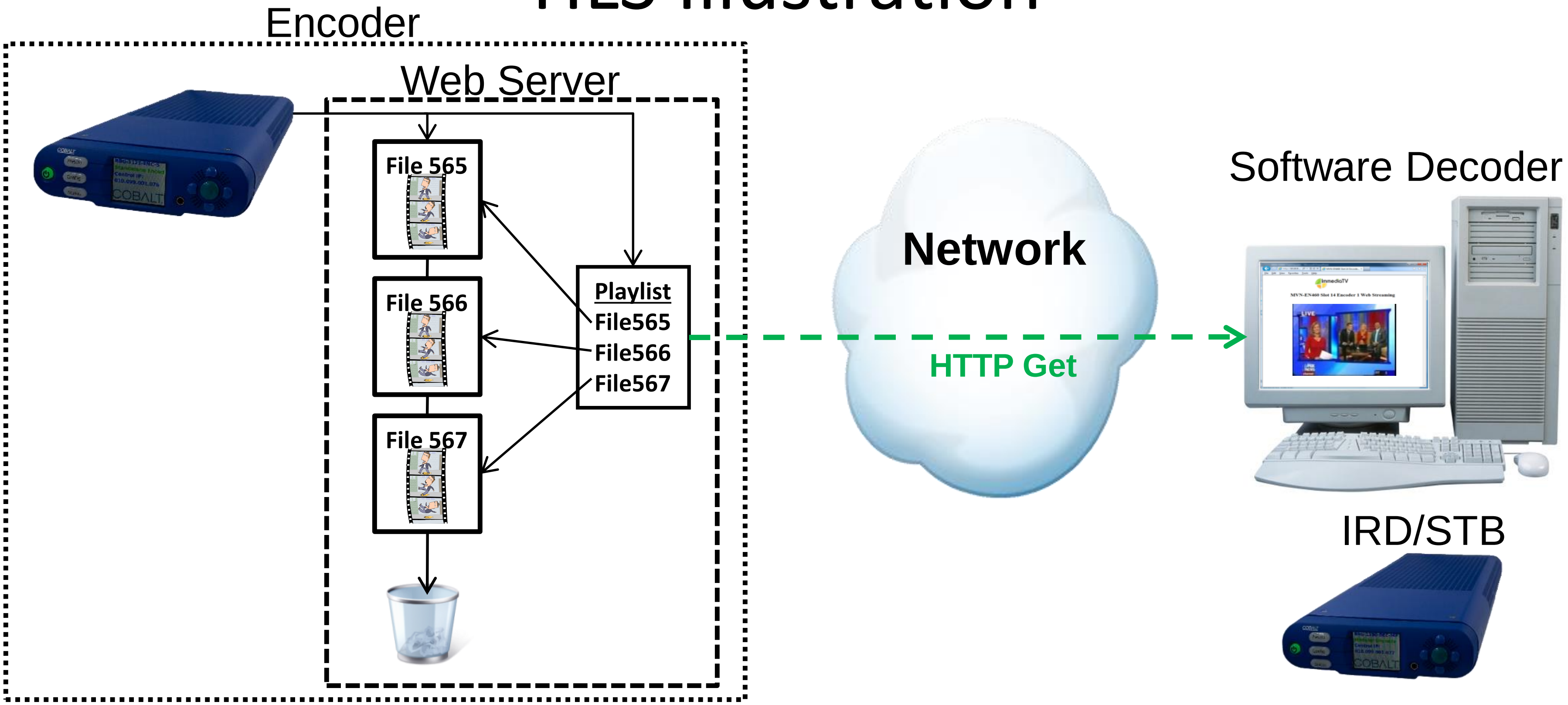
RTMP Discussion

- Characteristics:
  - Latency will depend on what processing is done in the server – typically on the order of several seconds or more
  - Resilient to packet loss (uses TCP)
  - Scalability is done at the server (commercial products by Adobe, Wowza, and open-source variants)
  - De facto standard for publishing live streams in the Internet
    - Industry is starting to move away from it as the protocol is obsolete
- Decoder support: Software decoders, some IRDs
  - Servers usually do protocol conversion for other decoders

HTTP Live Streaming (HLS)

- HLS is a protocol designed by Apple to provide streaming using a standard (unmodified) web server
- The video stream is divided into “chunks” of a few seconds each
- The decoder downloads the chunks as files from the web server with standard HTTP transactions, using a playlist
- Protocol supports adaptive streaming (multiple bit rates)
- Encoder can publish to a local (built-in) web server or to a remote server using HTTP PUT/POST
- MPEG-DASH is similar

HLS Illustration



COBALT

HLS Details

- Characteristics:
  - Very high latency: 3-4 times the chunk size (which varies from 2 to 30 seconds)
  - Extremely robust
  - Scalability can be done using external web servers
  - No TCP flow-control issue on the encoder side when publishing locally
- Decoder support: native support on all Apple and Android devices; supported in a number of IP set-top boxes and some professional IRDs

Protocol Comparison Matrix

	Low Latency	Medium Latency	High Latency
High Packet Loss Resiliency		RIST SRT	HLS/DASH RTMP/RTSP
Medium Packet Loss Resiliency	RTP+FEC		
No Packet Loss Resiliency	RTP/RTSP UDP		

TCP-based

UDP-based

COBALT

Q&A

- Questions?
- Thanks!

Contact:

ciro.noronha@cobaltdigital.com

COBALT