Designing a Collector Overlay Architecture for Fault Diagnosis in Access Networks

Mukundan Venkataraman, Shamik Sengupta, Mainak Chatterjee and Raja Neogi

Abstract—To prevent subscriber churn, network service providers of VoD, SDV and IPTV have a pressing need to pro-actively detect, isolate and fix outages within an access network. Network induced degradations prove to be detrimental for streaming applications. This typically leads to a poor quality of experience (QoE) for subscribers. By monitoring key functional points of the access network for traces of degradation, service providers can devise mechanisms to mitigate the problem.

In this work we propose a hierarchy of exporters, collectors and ANCON (ANalysis and CONtrol) nodes that can semi-autonomously monitor, detect and isolate impairments within an access network. Exporters on the data plane gather and disseminate statistics for individual subnets, which are streamed onto “collector” nodes on an orthogonal plane. Collector nodes aggregate traffic from various exporters, and stream them onto the root of the tree (ANCON). With an even placement of exporters, root cause analysis can now take the granularity of loss rates or delay rates in individual segments or subnets of an access network. As an extension to our architecture, we show that the overlay can support instrumentations of quality evaluation for streaming video. As an example, we use a simple MOS model that is in part an extension of the ITU-T Erlang model to predict the quality of a video stream much before it reaches the end user. We show that our overlay can support a wide variety of quality evaluation metrics under a simplifying assumption that such schemes work on inputs from the data plane to come up with a score that best represents the quality video stream in transit. Extensive simulations are presented to justify the design choices made in the process.

I. INTRODUCTION

Network service providers of video on demand (VoD), switched digital video (SDV) and IPTV are actively deploying triple and quadruple play networks that deliver voice, video and data services over shared and converged infrastructure. Service providers of VoD, SDV and IPTV typically revolve around an access network. This is a series of networked nodes managed by the service provider. An access network mostly consists of a source node, which acquires or generates streaming connect, which is then passed through a series of other nodes until it makes it to the edge of an access network. From the edge, media is distributed to end users, mostly using fiber optics and passive interconnecting nodes. With the ever growing use of such services, detecting, isolating, and fixing problems in IP networks have become serious problems that are bothering network operators.

End to end models of the Internet fail when we try to analyze streaming applications, since little can be done by the source if it has to rely on feedback from the destination. Further, the usage of UDP/IP networks poses its own problems: (i) IP is a best effort service, and there is no guarantee that the network will not discard, duplicate, delay or mis-order packets; (ii) UDP, on the other hand, has no provisions for detecting or recovering from packet losses, congestion, delays or jitters. Aggressive deployment of streaming content in access networks will eventually make them more sensitive to the level of service the network can provide. Such applications would then need a way to adapt their application parameters (like frame-rate, packet-injection rate etc.) to the network status. This is something the transport stack in its present form, without explicit feedbacks, cannot guarantee. Schemes that utilize such explicit feedbacks rely upon round trip time (RTT) estimations and timeout upon a failure to receive an acknowledgment. Accurate RTT estimation is known to be non-trivial, and false timeouts and stallings for acknowledgments are not uncommon. Also, since new content is to be delivered continuously, round trip times may be too long to learn about or recover from an event.

The networking aspect apart, there has been a surge of interest in trying to infer the quality of a video stream in transit. The authors in [9] classify quality assessment methods based upon the point of evaluation on the data plane: (i) Full reference (FR) methods have access to the original frame/pixel at both source and destination, (ii) reduced reference (RR) methods have only specific parameters at the encoder, and exact pixels at the decoder, and (iii) no reference (NR) have no access to any measurements at the encoder. Most of traditionally used quality evaluation either want the original frame for direct reference, or are housed at end systems alone (i.e., they are of type FR or RR). This form of quality evaluation is insufficient, because: (i) though one can get an estimate of degradation, one does not know what caused the degradation; and, (ii) when the whole process of delivery is complete, there is no time to react-to or correct any error. Traditional quality evaluation schemes were primarily designed to test the effectiveness of transcoding schemes alone, and were never designed with a networking element between the encoding and decoding processes. The process of delivering streaming content typically involves elements such as video-grooming, encoding, network transmission, decoding and playout. Since errors can potentially occur at each of these elements, there is a need to continuously monitor quality degradation. There has been a widespread interest to come up with a form of mean opinion score (MOS) to quantitatively score the quality

M. Venkataraman, S. Sengupta and M. Chatterjee are with the School of Electrical Engineering and Computer Science at The University of Central Florida, Orlando, FL 32816. E-mail: {mukundan.shamik.mainak}@cpe.ucf.edu, R. Neogi is with C-Cor Inc., 15797 NW Anduladian Way, Portland, OR 97229. E-mail: raja_neogi@ieee.org.
of a video stream. While the exact form of such a standard is debatable, it is well accepted that such a scoring mechanism, once standardized, would work on inputs from the data plane in some form of a combination of FR, RR or NR methods. Such quality evaluation strategies shall require architectural support to perform effectively.

We propose a comprehensive architectural support that can:
(i) effectively isolate impairments and outages within an access network, and (ii) facilitate continuous video quality scoring. Such an architectural support must be economical, stable, efficient and easy to adopt. The architecture should be able to seamlessly encompass a wide variety of quality scoring mechanisms, under a simplifying assumption that the scoring mechanism works on quantitative inputs that manifest as a unique quality score.

We design an overlay architecture that can monitor, diagnose and provide feedback on the health of various segments in an access network. We propose a hierarchical network of decentralized collector nodes. A collector node is a sensor probe that captures and disseminates various statistics about a network, like number of dropped frames, network delay, jitter etc. on a given segment of a network. Note that most of present day access networks already have the necessary infrastructure to deploy this; compiling usage statistics for billing purposes requires snooping into the user traffic to gather statistics. We simply extend and leverage this basic infrastructure to form a hierarchical organization of such collectors with sensing elements at the leaf and a correlating platform at the root forming a collector network overlay, or simply, collector overlay.

A collector overlay in converged multi-service networks plays an important role in fault diagnosis, capacity planning, dynamically assessing network optimization needs, and monitoring service usage. Collector overlays are hidden (i.e., they do not have published IP addresses) and unobtrusively gather service specific information at distributed sensing points, and then propagate them to the ANCON (analysis and control) points. Based on sensitivity of observed phenomena, ANCON engines generate action events that are routed to appropriate provisioning elements of the network. The collector overlay thus becomes the distributed virtual sensor for the physical network that deploys services. Such a network implicitly solves three important problems: (i) it can locally identify impairment points; (ii) ANCON can provide real time feedback to source about the current network status; and, (iii) it enables a continuous quality scoring mechanism if such collectors can estimate video quality.

In this work, we explore a semi-autonomic collector overlay architecture for access networks, that allows in house detection and isolation. Specifically, we make the following contributions:

- We present an architecture of a hierarchical collector overlay that gathers and disseminates network statistics. The nodes are decentralized and monitor local segments, and can control distribution of this information. We justify such an architecture with extensive experiments.
- We reverse engineer causes of network induced degradation with collector networks: something which has been impossible with traditional quality evaluation metrics. For e.g., PSNR, MPQM, and VQM can give an assessment of quality degradation, but can never tell what caused it. Collector networks can isolate both the cause of degradation and identify impairment segments/subnets that caused the degradation.
- Our architecture provisions a feedback mechanism where the source/local-segments can be informed of network statistics in real time. This would allow network aware streaming applications to better adapt to changing network conditions. This form of feedback is not possible with the conventional UDP/IP stack, which is inherently lightweight and best effort in nature.
- We show that quality can be scored continuously along the entire data plane. We extend our framework by defining a quality evaluation metric that takes parameters captured at the data link layer (like delay, loss etc.) of the protocol stack to compute video MOS. Our MOS calculation works without snooping into the actual video packets, and hence, without any form of decoding.
- We create an instrumentation layer above the data plane that computes local MOS, and correlates to compute global MOS. We export to service providers an estimate of local MOS in a segment rather than a series of network statistics to monitor health of a segment. This, along with various other tunable parameters, allows for efficient root causes analysis where fault detection can be performed if the health of a segment goes down.
- Our MOS calculations at the ANCON node could predict the quality of video at destination much before playout. We compare these MOS predictions to PSNR and observe that the results corroborate with each other.
- We identify key parameters that a collector overlay can monitor to make it a viable and powerful means in achieving a variety of diagnosis, feedback and planning activities.

Though the main goal is to discover faulty behavior and predict the service quality of video flows, the architecture can be easily extended to perform other video and network performance related services. Some of these are performance discovery (e.g., measuring round trip times of TCP/UDP), topology discovery (e.g., part or full connectivity), service usage discovery (e.g., who is using what service), fraud discovery (e.g., illegal usage), and trend discovery (e.g., forecasting user demands and preferences).

The rest of the paper is organized as follows. Related work is presented in section II. In section III, we argue the need for collector overlay and discuss its relevant characteristic features. Our simulation testbed is described in section IV. In section V, we reverse engineer playout degradation to its causes in the network, and isolate impairment points. In section VI, we present a simple video-MOS model that considers delay and loss rate. We present a schema of calculating video-MOS continuously along the data path using collector nodes in section VII. Conclusions are drawn in the last section.

II. RELATED RESEARCH

Significant work has been done in the areas of overlay networks, video quality measurement and network level mea-
surements. However, to the best of our knowledge, we are not aware of an overlay network architecture within an access network that can detect, isolate, diagnose impairments, and also allow implementations of video quality measurements.

An overlay architecture is mostly applied at the application layer, and its deployment generally does not warrant changes to the networking substrate beneath it. The concept of overlays is not new to networking; in fact, the Internet itself was designed as an overlay over the telephone network. Resilient overlay network (RON) [2] improves robustness and availability of Internet paths between hosts. It can discover outages and suggest workarounds: it allows a small group of distributed applications to detect and recover from outages in seconds (compared to several minutes of recovery in the Internet). Though our work in similar in philosophy (discovering outages), RON is more applicable to general traffic over the Internet. Network aware applications can make use of the CMU Remos [12], which provides a wide range of information to the application in a network independent fashion. Like our architecture, Remos gathers and controls distribution of information about individual subnets. However, the design emphasis on Remos has to been to provide a unified information interface to applications which operate over a diverse networking substrate, like the Internet. Apart from video applications, there has been some interest in designing overlay architectures to enable packet loss recovery and rapid rerouting of VoIP packets [1] over the Internet. Once again, our area of focus and design philosophies are completely different.

Extensive research has been performed by various groups towards objectively assessing quality of a video stream. Even at a time when there is little or no consensus on the exact nature of such a model, the basic idea has been to provide a subjective interpretation of various events on a video stream, where events could include loss, delay, jitter and so on. In particular, there is this need to accurately map an event to its effects on a human subject viewing the sequence. This would allow researchers to better understand how humans perceive degradation, and how best to avoid it. Authors in [13] train a random neural network to adjust to viewer responses, and infer the quality of a video stream based on what is observed along the data path. Typical efforts usually culminate in a mean opinion score (MOS), which on a chosen scale rates the quality of a particular sequence. MOS as a metric has been known to have its share of drawbacks [9], [24]: (i) subjects tend to avoid the extreme scores, (ii) “forgiveness effect”, where users tend to give higher rating when a playout is long and smooth, is not uncommon; and (iii) quality in itself is not a very well defined notion, and has many dimensions to it. As an alternate to asking users to provide a MOS score, certain other techniques have also been employed. Some studies, instead, ask the user to provide certain specific feedbacks. For example, users could be asked to report visibility of certain artifacts on screen [14], [15], or to suggest changes to artifacts to make the sequence more appealing to them [19], [5]. As can be seen, there is no one consensus on an acceptable quality assessment at the time of this writing. As a result, and to make our architecture flexible to many types of scoring schemes, we make a simplifying assumption: quality assessment shall operate on the basis of certain well defined inputs that can be drawn from the data plane. Note that this assumption is oversimplifying, since it assumes no reference frame in place for a direct comparison. These inputs could report certain specific events that happen on the data place, like a frame loss, the type of frame that was lost, delay, jitter and so on. As an example of instrumenting a quality evaluation based on a basic such rudimentary input, we devise a type of MOS can be calculated from such inputs. Since our goal is not to propose a new standard for MOS, we instead show that we can infer the health of a network segment (instead of the video stream) to successfully identify impairments. This also implies that other quality assessment techniques, once standardized or found suitable, can easily replace our model of MOS, which only reports the health of a network segment.

There have also been various proposals for protocols and techniques that mitigate network induced errors. Zhu et. al. [26] present an algorithm that allows for simultaneous design of source rate control and QoS-aware congestion control for streaming video. They utilize a virtual network buffer management, which allows to capture application specific QoS requirements which can be interpreted in terms of the source rate and the sending rate. Huang et. al. [8] propose JitterPath, which is a noise resilient available bandwidth estimation scheme. There has also been considerable work on packet loss resilience: Yang et. al. [25] exploit the fact that not all packets or frames are of equal importance in a video stream and design a recovery scheme based on FEC. We acknowledge the vast breadth of recovery schemes possible from prior research. We argue that, with the overlay and a feedback mechanism in place, one can use any of the available scheme or devise newer ones that can avoid or react to network related outages.

III. COLLECTOR OVERLAY

VoD and IPTV services are usually centered around an access network. An access network has a video source, and terminates at an edge-QAM (or E-QAM). Access networks are usually managed by service providers, and are typically run over IP. This section details the instrumentation of a collector overlay in such access networks.

A. Topology Overview

A video network along with the possible collector overlay is shown in Figure 1. Acquired video (either by satellites or alternative sources) is pushed by the VoD or the SDV pump. Video packets pass through a “Groomer” where variable bit-rate traffic is policed to an almost constant bit-rate traffic. Video data then passes through a series of routers (in the IP domain) before it reaches “M-join” (multicast join node). The routers (NE0 and NE1) potentially multiplex various other traffic within the access network. At Mjoin, selected flows are pulled on-demand (via control plane infrastructure not shown) through the Edge-Qam (EQAM). The access network terminates at E-QAM, and what follows next is a series of “passive” nodes. Bandwidth allocation for all video flows are done at the E-QAM and streams distributed over fiber nodes. The video signal is amplified before reaching the set-top box (STB) and the corresponding television (TV).
B. Gathering statistics on the Data Plane

We focus on the data plane within an access network (the region between VoD/SDV pump to the E-Qam). We evenly place collectors at various segments on the data plane. The collectors passively and non-intrusively monitor parameters of the video flow, and stream these statistics on an orthogonal plane.

Quality of experience has been found to be influenced by both network dependent and network independent parameters. Network dependent parameters are events like loss, delay, jitter, error etc., while the latter class includes the users mood, environment, “forgiveness effect” (when a long video sequence with errors in the beginning is rated high because a high quality was presented over a long period of time) and so on. While modeling subjectivity and its various intricacies are highly debatable topics, we continue to focus on monitoring the health of an access network, while arresting and notifying of chosen network induced impairments.

Monitoring along the data path has been classified by Reibman et. al. in [18], and in [9]. Based on the various measurement points possible on a data path, measurements could be at: (i) uncompressed video input at encoder; (ii) decompressed video at decoder (as output); and (iii) the bitstream in transit. Correspondingly, a Full-Reference (FR) requires measurements akin to (i) and (ii), Reduced-Reference (RR) requires measurements corresponding to (i) and at (ii) or (iii), while No-Reference (NR) does not require the original video.

Real time and passive monitoring would typically require the NR method, since the original bitstream is generally unavailable at various data points. Also, because the original frames are absent for a direct comparison between what is and what was, correlating this to a MOS score is a non-trivial task. We leverage information already available from the IP headers to calculate a MOS indicative health score of the network. Though this would not necessarily compute the exact subjective perception of the video at destination, what this computes is the “health” of various network segments in terms of a MOS. We acknowledge the fact that there are several other MOS score computations (for example, one that employs random neural networks [13]). Such plugins could be easily implemented in our architecture for more accurate readings.

We implement control data capture at selected data points along the access network. The implementation code is implemented in libpcap library [35] (our implementation is available for free download). Our code computes delay rate and loss rate from the data path streams for correlation at nodes one level higher than the data plane nodes. These are computed for individual segments/subnets of the network, and streamed. ANCON inspects the ID of the exporter that originated the statistic, and based on its knowledge of the geographical layout of various exporters, it infers the health of the corresponding segment.

C. Correlation Platform

While the video passes through the data plane (shown with solid arrows), probing can be done at various stages to monitor the quality of the flow. The collector overlay shown in Fig. 1 has five collector nodes. These collector nodes constantly monitor and measure various network and flow parameters. Information gathered by leaf collectors are highly localized. In other words, a given collector is completely oblivious to impairments happening in a preceding and succeeding segment. The statistics thus gathered are streamed to the ANCON node. Such a mode of operation achieves two things: (i) local collector data can be inferred as an absolute health of a particular segment, and (ii) global correlated data at ANCON shows relative fluctuations among various segments.

D. Streaming Protocol

Given the collector overlay set-up, there is a need for a streaming protocol that exports network statistics to correlation platforms. Streaming usage statistics to correlation nodes for usage, accounting and billing purposes have been popular, but they are inadequate in supporting evolving requirements from service providers. For example, RADIUS [20] is a widely

1 Download site: http://www.cs.ucf.edu/~mukundan/netdoctor. Alternatively, for test-bed implementations equipments from IneoQuest [33] could also be used.
deployed protocol that may be used for exporting usage information. However, it can only handle a few outstanding requests and has extensibility challenges due to its limited command and attribute address space, complexity, and vendor specific attributes [32]. DIAMETER [4] is a new Authentication, Authorization and Accounting (AAA) protocol that retains most of the RADIUS structure but overcomes many of its shortcomings. However, it is highly optimized for streaming accounting information and its design does not stress on efficiency and performance for a large number of flows. SNMP based approaches are also inadequate, since they require a large amount of processing and bandwidth [7]. Further, the request-reply dialog of SNMP is not too well suited for such purposes.

Of particular interest is the recent Internet Protocol Detail Record, or IPDR [31] which is especially geared towards high volumes of reporting traffic. IPDR specifies the mode of exchange of such streaming statistics, and is generally classified in two parts: (i) exchanging information in XDR format, and (ii) an IPDR streaming protocol (IPDR/SP [32]). The streaming protocol, though not freely available for download, identifies key requirements that such a protocol must possess: (i) reliable, in-order delivery of data, (ii) connection oriented, (iii) session level authentication, (iii) message based delivery and (iv) fast connection failure detection. A recommended transport layer for such requirements is the SCTP [21] (though TCP [16] possesses features (i) and (ii), it does not support either of (iii), (iv) and (v) as requirements). We use a network stack (with SCTP) for the collector overlay that unconditionally streams statistics. The streaming can be controlled by a “rate” visible to the network provider to better adapt to variations in the overlay. Note that topology maintenance, load balancing and failure detection in the overlay itself are valid and exciting problems. We, however, focus on using a simplistic collector overlay tree (of depth 1) to identify and isolate impairment segments only, and do not delve any deeper in these related problems.

E. Monitorables

The collector overlay can be thought of as sensors in the form of embedded SNMP or IPDR [31] agents in the data and/or control path. These sensors gather monitorables (monitored parameters) that are propagated to measurement nodes of the collector network. Measurement entails applying operators to the monitorables to generate processed data that can be stored or consumed by various correlation applications. For example, delay, loss, jitter and bit-errors (monitorables) are captured by data-path sensors. Video MOS operators are applied to generate video quality measurements, which can be consumed by a network operator application.

Usage of the collector overlay calls for some judiciousness in monitoring. If too many parameters are monitored per flow, the traffic volume of the collectors will begin to show exponential growth. This problem is attributed to two clauses: (i) the number of parameters per flow, which is to identify the most pertinent information desired; and (ii) the rate (i.e. frequency) of information streaming, which in our architecture is a tunable parameter exported to service providers.

F. Collector Nodes: What needs to be observed?

Transcoding and transrating were always considered bad choices, but infrastructure economics have made their use unavoidable (MPEG-2 capture and MPEG-4 delivery). Similarly delivery in converged networks and playout with finite size buffers introduces undesirable visual hazards. There is a clear need for in-network fault detection and possible correction. Collector nodes provide a possibility of arresting events as they occur. Based on a survey of various other MOS models already in place, we advocate the following metrics to be brought to focus: (i) knowledge of what a stream carries; (ii) the relative importance of a packet; (iii) the statistic of what a packet has already been through; and (iv) what needs (in terms of resource provisioning) might have to be met to do best with it.

Extensive simulations have been conducted (see, for e.g. [23]) to test the sensitivity of subjective video quality perceptions in the face of various networking events. Of the metrics tested, namely delay, loss, jitter and error, it is well accepted that loss is the most damaging, with jitter and error being the other factors in that order. Loss correlates to missing information. Each lost packet is either a part of a given frame, or worse, part of a frame that is used as a reference frame to reconstruct other frames. Also, packet errors tend to propagate to other frames, since successful decoding depends on both a given frame and its reference frames to be intact. Since the I-Frame is used as a reference for constructing B and P frames, it is intuitively the most important element to be present in a GOP. I-Frames with errors turn out to be more degrading than other frame types, and quietly rightly so. Jitter has implications on the receiver queue length, and leads to overflow or underflow. Unless the information is interactive or time critical in nature, delay as a component has little effect.

Evaluating quality at end systems alone make little sense since errors potentially creep in during encoding, transcoding, transmission and decoding. If a stream with poor transcoding is passed onto a network, scoring this just at endpoint collector node would indicate severe network induced impairments: clearly a case where the network infrastructure can do little about. Similarly, poor decoding can mask optimal encoding and transmission. We argue that video quality evaluation is a continuous process, and isolating impairment requires a local perspective and a global perspective. This implies the presence of many collector nodes, each tracking a segment, and global correlation to assess collective behavior.

IV. TESTBED SET-UP

Our simulation testbed (Fig. 2) mostly mimics the access network shown in Fig. 1, and consists of a source video, encoders, a simulated network with tunable parameters and a decoder. The set-up was designed to establish feasibility of collecting and analyzing data via placement of collector nodes. We use the MPEG-4 [11] specification to encode the source video stream and the network simulator (ns-2) [30] to simulate the network. We use packages such as Evalvid [10] and MSU Video Evaluation Toolkit [34] to better model the network as close to reality as possible. Ns-2 has well tested simulation suites to model various transport protocols, buffering strategies, links and network dynamics.
Video grooming is used to inject frames into the network at a constant rate instead of overwhelming the network with a naive UDP blasting strategy. This is achieved by creating a custom traffic in ns-2 that injects video packets at a pre-defined interval into the network. The path taken by the video stream is as follows: the source node (node “video source”), MPEG-4 “encoder”, the intermediate IP hops (nodes 1 through 7) which potentially multiplex traffic from various sources, MPEG-4 “decoder”, and a destination node (node “destination”) where decoding and playout occur. Collector nodes (marked “CN” 1 through 4) are attached to data-plane nodes (nodes 1, 3, 4 and 6). The collectors form a simple hierarchy (a tree of depth 1) and correlate data at the ANCON node. The streaming behavior mimics the IPDR streaming protocol, which works by disseminating control information periodically for a specified “rate” of streaming. We also set up a competing traffic originating at nodes with names prefixed with CS (competing source) and terminating at nodes prefixed CD (competing destination) respectively, marked 1 through 3. The rate and duration of this competing traffic, in combination with a choice of various other simulation parameters (shared buffer/queue length, link speeds etc.) are tuned to create various events in the network like delay, jitter, and loss. Other simulation parameters are shown in Table I. This simulation topology is consistently used for the rest of the paper.

### Table I

<table>
<thead>
<tr>
<th>Simulation Parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Video Parameters</strong></td>
<td></td>
</tr>
<tr>
<td>Resolution</td>
<td>352x240 pixels</td>
</tr>
<tr>
<td>Frame rate</td>
<td>90 frames/sec</td>
</tr>
<tr>
<td>Video Color Mode</td>
<td>Y, U, V (4 − 2 − 0 scheme)</td>
</tr>
<tr>
<td>GOP Length</td>
<td>30 Frames</td>
</tr>
<tr>
<td>Sequence length</td>
<td>10 secs of playout</td>
</tr>
<tr>
<td><strong>Network Parameters</strong></td>
<td></td>
</tr>
<tr>
<td>Unshared Links</td>
<td>1 Mbps</td>
</tr>
<tr>
<td>Link Delay</td>
<td>1 ms</td>
</tr>
<tr>
<td>Buffer length</td>
<td>50 packets</td>
</tr>
<tr>
<td>Queue behavior</td>
<td>Droptail</td>
</tr>
<tr>
<td>Packet Size</td>
<td>1024 bytes</td>
</tr>
<tr>
<td>Max. Fragmented Size</td>
<td>1024 bytes</td>
</tr>
</tbody>
</table>

We perform the following sequence of events. First, we take a raw video in a YUV format and perform MPEG-4 encoding to generate frames. These frames are then converted to IP packets with a fragmentation limit, and transported over the conventional UDP/IP stack. At the destination, IP packets are reassembled to form the MPEG-4 frames which are then played out.

Video grooming is used to inject frames into the network at a constant rate instead of overwhelming the network with a naive UDP blasting strategy. This is achieved by creating a custom traffic in ns-2 that injects video packets at a pre-defined interval into the network. The path taken by the video stream is as follows: the source node (node “video source”), MPEG-4 “encoder”, a video “groomer”, the intermediate IP hops (nodes 1 through 7) which potentially multiplex traffic from various sources, MPEG-4 “decoder”, and a destination node (node “destination”) where decoding and playout occur. Collector nodes (marked “CN” 1 thorough 4) are attached to data-plane nodes (nodes 1, 3, 4 and 6). The collectors form a simple hierarchy (a tree of depth 1) and correlate data at the ANCON node. The streaming behavior mimics the IPDR streaming protocol, which works by disseminating control information periodically for a specified “rate” of streaming. We also set up a competing traffic originating at nodes with names prefixed with CS (competing source) and terminating at nodes prefixed CD (competing destination) respectively, marked 1 through 3. The rate and duration of this competing traffic, in combination with a choice of various other simulation parameters (shared buffer/queue length, link speeds etc.) are tuned to create various events in the network like delay, jitter, and loss. Other simulation parameters are shown in Table I. This simulation topology is consistently used for the rest of the paper.

### V. Isolating Impairment Points

When degradation is observed during playout at destination, we seek to understand the causes of it. Though traditional quality evaluation metrics like PSNR can provide a measure of degradation, they cannot pinpoint the exact cause of it. On the networking side, though lightweight protocol stacks such as UDP/IP are highly suited for streaming applications, it is not possible to isolate impairment points in a network because of the lack of feedback between networking entities. Collector networks can, however, unobtrusively achieve these with relative ease.

In this section, we reverse engineer the causes of degradation. When a respective degradation happens during playout, we corroborate with statistics from the collector nodes to establish their ability in isolating impairments. For example, we seek data from collector nodes when the playout (i) is smooth and without distortion; (ii) has frame freezing in it; and, (iii) playout has garbled images. We conducted several experiments in which we intentionally introduce various events like extreme delay or loss in a particular segment of the topology. We observe scenarios when there are: (i) no impairments; (ii) delay impairments in only one segment and (iii) loss impairments. With collector nodes, we can identify both the cause of the impairment as well as isolate the segment that caused the impairment. Feedback received from the collectors make way for a plethora of corrective actions possible at the source host or with the choice of protocols used.

#### A. A case with no impairment

To achieve a smooth playout at destination, we conducted experiments with little or no competing traffic in the network shown in Fig. 2. Our collector network is tuned to monitor packet losses and cumulative delay as perceived at every segment. We capture the cumulative delay for this scenario (shown in Figs. 3(a), 3(b), 3(c) and 3(d)). The plots show a consistent and marginal increase in cumulative delay as observed from one collector segment to another. It is easy to infer from the plots that the presence of a consistent uniform gap between delay values is indicative of little or no delay impairment. Also, with little competing traffic, there were no packet losses.

#### B. Delay in a segment: Frame freezing

To simulate excessive delay in a segment, we introduced a series of ephemeral VBR flows from the 3rd second of playout between CS2 and CD2 in Fig. 2. This was done in conjunction to increasing the buffer size of node-5 to ensure no packets were lost. With an increase in competing traffic and large queue build up at node-5, the segment between node 3 and node 5 is at fault.

A direct consequence of large delays is frame “freezing” during playout. Freezing is when the motion picture gets stuck at one particular frame during playout. Since successful decoding relies upon a constant arrival of frames for consumption at the destination buffer, a delayed frame is as good as a lost frame. In this case, the decoder presents the last successfully

---

$^2$Cumulative delay defined as the total delay observed by a packet since its inception in the network to its arrival at a collector.
constructed frame until another frame arrives at the destination buffer. Since our competing traffic starts at the third second of playout, the first part of the playout is smooth while the later part has frozen frames due to network delays. We present a snapshot of freezing during playout and reverse engineer its cause. While playout at source changes from Fig. 4(a) to Fig. 4(b) for frames during the 7th and 9th second, the playout at destination is stuck at one frame as seen in Fig. 4(c) and Fig. 4(d). Note, however, that no frames were lost as a congestion drop because the queue size at node 5 was chosen to be sufficiently large.

The cumulative delay plots as seen across the four collector nodes for this scenario is presented. While the first two collector nodes before the faulty segment register an even increase in delay (Fig. 5(a) and Fig. 5(b)), the collectors at and after the faulty segment clearly show increasing delay in the segment between node 3 and node 4 (Fig. 5(c) and Fig. 5(d)).

C. Loss Impairment

Competing traffic was introduced in the first segment (node 1 to node 2 region) between $CS_1$ and $CD_1$ for the first three seconds of playout. VBR flows were introduced with default queue sizes at node 1.

With IP fragmentation, each frame is broken into many packets (an I-Frame was usually broken to 16 packets with a fragmentation limit of 1024 bytes). A lost packet is hence a part of a frame. Since successful decoding depends upon the reference frames being intact, a lost packet corrupts a frame, or worse, hampers reconstruction of other frames. Loss is undoubtedly the most damaging event to video quality. Depending upon the relative importance of a packet lost in the network, the effect on playout varies from the appearance of minor “blocks” in the video (corruption of P-Frames) to a complete “whiteout” when nothing appears on screen (loss of an I-Frame, especially the first I-Frame of a temporal
VI. MEASURING QUALITY OF VIDEO

The quality of reconstructed video frames are often measured by the mean opinion score (MOS). MOS is a subjective quality score that ranges from 1 (worst) to 5 (best) and is obtained by conducting subjective surveys. To continuously monitor the quality of a video stream, we need a way of calculating MOS from the statistics provided by the collector nodes. This section presents one such model. While the MOS calculation may not be the most extensive, our goal is to: (i) establish feasibility of performing this calculation locally at collector nodes; (ii) prove that continuous monitoring is possible without actually decoding the video frames. Our model is in part a mutation of the Erlang model recommended by the ITU-T [28], [29]; and (iii) create a model that can translate link layer statistics (like delay, loss, jitter etc.) into a more meaningful subjective MOS. This would help service providers draw meaningful inferences about an access network’s health using the collector overlay.

A. Video-MOS

For the sake of simplicity, we use the video-MOS characterized by the ITU-T E-Model [28], [29] (for audio, video and audio-visual). The adaptation of video-MOS can be expressed with the help of parametric estimations that combines different aspects of quality impairment. For this purpose, we first define Q-factor, that considers the effect of delay and loss impairments on the quality of a video stream. It is given by

\[ Q_{\text{der}} = Q_{\text{original}} - I_c - I_d \]  

where \( Q_{\text{original}} \) is the quality factor of the original video and \( Q_{\text{der}} \) is the derived quality factor of the captured video at the receiver’s end. \( I_c \) is an equipment impairment factor associated with the losses due to the codecs and network and \( I_d \) represents the impairment caused by the delay.

For all parameters used in the algorithm of the E-model, the default values are listed in Table II. We use these default values for all parameters which are not varied during planning calculation to find the maximum value of quality factor, which eventually can be considered as \( Q_{\text{original}} \). If all parameters are set to the default values, the calculation results in a very high quality with a quality factor of \( Q_{\text{original}} = 93.2 \).

**TABLE II**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Default value</th>
<th>Permitted range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean one-way delay of echo path</td>
<td>0 ms</td>
<td>0–500 ms</td>
</tr>
<tr>
<td>Absolute delay in echo-free connection</td>
<td>0 ms</td>
<td>0–500 ms</td>
</tr>
<tr>
<td>Equipment impairment factor (I_e)</td>
<td>0</td>
<td>0–40</td>
</tr>
<tr>
<td>Random packet-loss probability</td>
<td>0%</td>
<td>0–20%</td>
</tr>
</tbody>
</table>

B. Effect of Delay Impairment \( I_d \)

For streaming video, total delay is composed of three components: codec delay (\( d_{\text{codec}} \)), playout delay (\( d_{\text{playout}} \)), and network delay (\( d_{\text{network}} \)). Codec delay represents the algorithmic and packetization delay associated with the codec and varies from codec to codec. Playout delay is the delay associated with the receiver side buffer required to smooth out the delay for the arriving packet streams. Network delay is the one-way transit delay across the IP transport network from one gateway to another. Thus the total delay is

\[ d = d_{\text{codec}} + d_{\text{playout}} + d_{\text{network}} \]  

The delay impairment \( I_d \) depends on the one way delay encountered. The effect of this delay can be modeled as [27],

\[ I_d = \alpha_1 d + \alpha_2 (d - d_{\text{threshold}})H(d - d_{\text{threshold}}) \]

where \( H(x) \) is an indicator function: \( H(x) = 0 \) if \( x < 0 \), and 1 otherwise. \( d_{\text{threshold}} \) is some delay threshold exceeding which might degrade the video quality excessively. \( \alpha_1 \) and \( \alpha_2 \) are two experimental parametric estimators to measure the effect of delay on quality factor.
C. Effect of Loss Impairment $I_e$

Similarly, we model the loss ($e$), which includes all kinds of losses. The modeling can be done as

$$e = e_{\text{network}} + (1 - e_{\text{network}})e_{\text{playout}}$$  \hspace{1cm} (4)

where, $e_{\text{network}}$ is the loss rate due to the loss in the network and $e_{\text{playout}}$ is loss rate due to the playout loss at the receiver side.

We believe that as the collector nodes are probing the video stream at the packet level, no playout of the video is necessary at this scenario thus making $e_{\text{playout}} = 0$.

Then the loss impairment factor can be given as

$$I_e = \beta_1 \ln(1 + \beta_2 e)$$  \hspace{1cm} (5)

$\beta_1$ and $\beta_2$ are two experimental parametric estimators to measure the effect of loss on quality factor.

D. Video-MOS calculation

We rewrite equation (1) using equation (3) and (5) as

$$Q = 93.2 - \beta_1 \ln(1 + \beta_2 e) - (\alpha_1 d + \alpha_2 (d - d_{\text{threshold}})) \ln(d - d_{\text{threshold}})$$  \hspace{1cm} (6)

where $Q_{\text{original}}$ is assumed as 93.2.

The quality-factor can be then related to MOS through the following non-linear mapping provided by ITU-T [27], [29].

$$MOS = 1 + 0.035Q_{\text{der}} + 7 \times 10^{-6}Q_{\text{der}}(Q_{\text{der}} - 60)(100 - Q_{\text{der}})$$  \hspace{1cm} (7)

We use this definition of MOS for calculating quality of video stream in section VII.

VII. CORRELATING MOS WITH COLLECTORS

Using the expression for MOS, we proceed to demonstrate the effectiveness of the collector overlay in locally computing MOS, and corroborating the global MOS at the ANCON node. Simulations in this section assume a full fledged network traffic with persistent competing traffic (traffic with varying sending rates at all three competing traffic segments). The collectors observe delay and loss at various segments and calculate local MOS. Using these statistics, the ANCON is able to compute global MOS of the stream and predict a subjective rating of the video much before decoding at destination happens. We consider each event as follows.

A. Layer 2 Statistics: Delay and Loss

We begin with Layer-2 statistics already available from the collector probes to calculate MOS. Statistics such as loss or delay convey little information to service providers about the behavior of a segment. While such statistics effectively characterizes the events at a segment/subnet, mapping a large combination of such network events to their effects on the video quality would be highly beneficial. We hence compute local MOS at individual collectors, and then compute a global MOS at the ANCON using statistics from various collectors. This serves two purposes: (i) provides a characterization of individual segments/subnets with respect to local or expected MOS, which makes more sense to service providers, and (ii) the ANCON node computing global MOS can predict the quality of the stream much before its actual playout occurs.

We calculate the cumulative delay of every packet carrying the video stream through the data segments with a collector node in proximity. The various plots for delay as seen (independently) by the four collectors are shown in Figs 9(a), 9(b), 9(c) and 9(d). From discussion in the preceding section, interpreting the plots is rather straightforward: CN1 shows little anomaly since it does not see queuing delays caused by the competing traffic, while the other four collectors register large variations in delay values.

Analogously, in Fig. 10, we observe how individual packets are lost at the four collector nodes. The relatively high loss at CN1 is due to the high competing traffic in segment 1 caused by the traffic between $CS_1$ and $CD_2$. There is no loss observed at collector node 3 because segment 3 has no ingress competing traffic. For the entire duration of the video, a total of 84 packets were lost with 46 packets at $CN_1$, 10 packets at $CN_2$, and 28 packets at $CN_4$. This clearly identifies the first segment (node-2 to node-3) as highly impaired by congestion, with segment-4 and segment-3 following in that order. Notice the closely spaced loss pattern at $CN_1$, which indicates that many packets were lost in quick succession in that segment.

B. Computing Loss and Delay rates

Video MOS calculations as per Equations 6 and 7 require parameters delay and loss expressed as rates. Though we capture individual statistics, there is no notion of computing rates. To obtain delay and loss rates, we considered discrete time epochs and independently calculated cumulative delay and loss per epoch. Accurate loss rate is obtained when we consider

![Fig. 7. Delay statistics at the four collectors when loss happens at the first segment: (a) CN1 varying delay patterns, since the queue at node-1 fills to maximum leading to drops; (b) CN2, (c) CN3 and (d) CN4. Observe cumulative delays after the first segment since there are no competing flows in subsequent segments.](image-url)
Fig. 9. A case for full fledged traffic in the network, with traffic in the first segment starting late, and multiple drops happening at most segments. (a) CN1 observes little fluctuations, (b) CN2 observes large fluctuations: the sharp drop to zero values at various intervals happens because that particular packet is lost in a previous segment, (c) CN3 registers slightly higher values with large fluctuations, and (d) CN4 cumulatively registers still higher fluctuations.

Fig. 10. Loss of packets at collector nodes for a full fledged competing traffic, with losses happening at all but the third segment. The drop patterns also indicate quick successive drops at CN1 and CN4.

the entire duration of the video playout. However, such long time intervals beat the purpose of real-time monitoring. Also, it does not allow the network provider to take corrective actions if any anomaly is detected. We export the epoch interval as a tunable parameter to service providers to allow them to best decide such an interval. As a default in these experiments, we use one-third of a second as an observation period.

In Figs. 12(a) through 12(d), we show the average loss rate at the four collector nodes for the same epochs. The loss rate at collector node 3 is 0 because there was no packet loss in that segment (see figure 10). Notice the sharp spike in loss rate at CN1 in Fig. 12(a). Also to be noted is the closely spaced histogram corroborating the fact that many packets were lost in quick succession.

This has direct implications in the delay rate calculations shown in Figs. 11(a) through 11(d). Since packets corresponding to the 18th epoch were lost in the very first segment, the plots show a missing value (since those packets never made it to any collector, and hence, never contributed to any delay in that epoch. In other words, the collectors noticed no activity in those epochs).

C. Computing Local MOS

With the delay and loss rates known at the four collector nodes, we can calculate the video-MOS. In Fig. 13(a) through 13(d), we show the video-MOS (as calculated from Equation 7) at the four collector nodes. Note that MOS too is calculated for individual epochs of time. The delay and loss rate used are the same ones as was shown in Fig. 11(a) through 11(d) and Fig. 12(a) through 12(d).

It is interesting to note the health of individual segments in terms of video MOS. Segment-3 intuitively provides a smooth and consistent MOS\(^3\), while segment-1 (Fig. 13(a)) and segment-4 (Fig. 13(d)) register the largest fluctuations in values. It is easy to observe from the MOS calculations that loss has a large effect on MOS: the peaks in loss rates have a direct impact in the MOS calculations.

D. MOS at Correlation Platform

Since statistics are streamed to the ANCON node, it is able to globally correlate various events at all four segments of the data path. Using this, we calculate a global MOS at ANCON. Unlike collectors which only see local events and compute local MOS, ANCON can cumulate effects of one collector to those of another. In the sense, if packets are lost at a collector segment, its effects can be easily perceived at the successive segment since quality of the video has already degraded. Also, equipped with individual MOS patterns at various collectors, the ANCON is able to identify impairments in term of local MOS.

It is interesting to observe how ANCON’s projection of MOS compares with more established metrics like PSNR. Note that ANCON calculates MOS using network events like loss and delay. PSNR, however, decodes the received frame and compares that to its corresponding frame at source. Such a direct comparison can directly identify differences in frame content, and this returns a score of quality. We plot ANCON’s projection of MOS in various epochs in Fig. 14(a), and directly compare this to PSNR values calculated with source and reference frames. For the PSNR plot, we compute the average PSNR values in various epochs (to allow a direct comparison). The plots are shown in Fig. 14(a), 14(b). Notice the striking similarity in projections. The ANCON was hence successfully able to project degradations, and this happens much before actual decoding or playout occurs at destination.

We present two screen-shots of the video at two different instances. Figure 15(a) shows the 75th frame of the video. This

\(^3\)Though the plot in Fig. 13(c) seems to be a straight line, there are very slight variations of amplitude 0.01 within the line. This is largely because of delay variations in segment-3 as seen in Fig. 11(c).
frame falls in the 9th time epoch which has a MOS of 4.25757. Similarly, Figure 15(b) shows the 261th frame. This frame falls in the 28th time epoch which has a MOS of 3.250206. As expected, there is a distinct difference in the quality of the video as predicted by the ANCON.

VIII. CONCLUSIONS

Given the stringent delivery bounds of streaming applications, there is a need to arrest network induced errors as they occur. This would naturally be the fastest way to...
isolate, react, and adapt to changing network conditions. In this paper, we successfully design and evaluate an architecture of a hierarchical overlay of “collector” nodes that can gather and disseminate network statistics. The collectors are decentralized, and can monitor the health of local segments. The statistics are routed to a root node, called ANCON, that correlates and computes various properties of the video flow.

Using such a network of collectors, we are able to: (i) isolate network induced causes of a degradation; (ii) isolate segments/subnets in the network that caused the impairment, (iii) provision local feedbacks, and inform source in real time of network statistics, (iv) create an instrumentation layer above the data plane that can correlate network statistics and compute MOS, (v) export local MOS to service providers instead of overwhelming them with network statistics, to better facilitate root cause analysis, (vi) show that quality can be scored continuously along the data path, and, (vii) predict the quality of a video much before the playout actually occurs at destination. Our evaluation of the design with extensive experiments establishes feasibility of the architecture. This study also identifies key knobs that should be exported to service providers to help them self-tune the architecture, as well as key parameters that a collector nodes should monitor.

REFERENCES