6-21-2019 11:00 AM

Enhanced Multimedia Exchanges over the Internet

Fuad Shamieh
The University of Western Ontario

Supervisor
Dr. Xianbin Wang
The University of Western Ontario

Graduate Program in Electrical and Computer Engineering
A thesis submitted in partial fulfillment of the requirements for the degree in Doctor of Philosophy

Abstract

Although the Internet was not originally designed for exchanging multimedia streams, consumers heavily depend on it for audiovisual data delivery. The intermittent nature of multimedia traffic, the unguaranteed underlying communication infrastructure, and dynamic user behavior collectively result in the degradation of Quality-of-Service (QoS) and Quality-of-Experience (QoE) perceived by end-users. Consequently, the volume of signalling messages is inevitably increased to compensate for the degradation of the desired service qualities. Improved multimedia services could leverage adaptive streaming as well as blockchain-based solutions to enhance media-rich experiences over the Internet at the cost of increased signalling volume. Many recent studies in the literature provide signalling reduction and blockchain-based methods for authenticated media access over the Internet while utilizing resources quasi-efficiently. To further increase the efficiency of multimedia communications, novel signalling overhead and content access latency reduction solutions are investigated in this dissertation including: (1) the first two research topics utilize steganography to reduce signalling bandwidth utilization while increasing the capacity of the multimedia network; and (2) the third research topic utilizes multimedia content access request management schemes to guarantee throughput values for servicing users, end-devices, and the network.

Signalling of multimedia streaming is generated at every layer of the communication protocol stack; At the highest layer, segment requests are generated, and at the lower layers, byte tracking messages are exchanged. Through leveraging steganography, essential signalling information is encoded within multimedia payloads to reduce the amount of resources consumed by non-payload data. The first steganographic solution hides signalling messages within multimedia payloads, thereby freeing intermediate node buffers from queuing non-payload packets. Consequently, source nodes are capable of delivering control information to receiving nodes at no additional network overhead. A utility function is designed to minimize the volume of overhead exchanged while minimizing visual artifacts. Therefore, the proposed scheme is designed to leverage the fidelity of the multimedia stream to reduce the largest amount of

control overhead with the lowest negative visual impact. The second steganographic solution enables protocol translation through embedding packet header information within payload data to alternatively utilize lightweight headers. The protocol translator leverages a proposed utility function to enable the maximum number of translations while maintaining QoS and QoE requirements in terms of packet throughput and playback bit-rate.

As the number of multimedia users and sources increases, decentralized content access and management over a blockchain-based system is inevitable. Blockchain technologies suffer from large processing latencies; consequently reducing the throughput of a multimedia network. Reducing blockchain-based access latencies is therefore essential to maintaining a decentralized scalable model with seamless functionality and efficient utilization of resources. Adapting blockchains to feeless applications will then port the utility of ledger-based networks to audiovisual applications in a faultless manner. The proposed transaction processing scheme will enable ledger maintainers in sustaining desired throughputs necessary for delivering expected QoS and QoE values for decentralized audiovisual platforms. A block slicing algorithm is designed to ensure that the ledger maintenance strategy is benefiting the operations of the blockchain-based multimedia network. Using the proposed algorithm, the throughput and latency of operations within the multimedia network are then maintained at a desired level.

Keywords: Multimedia, adaptive streaming, signalling overhead, processing latency, blockchain, steganography, embedded signalling exchange

Lay Summary

The Internet was not initially designed for exchanging multimedia streams but for sending and receiving bytes of data. The consumer Internet is heavily utilized for exchanging audio-visual data while it is not the best medium for transmitting said data. The intermittent nature of multimedia traffic, the unguaranteed underlying communication infrastructure, and dynamic user behaviour collectively result in the degradation of perceived multimedia quality by end-users. To compensate for the decreased perceived quality, the volume of signalling messages increases to attempt pushing the quality metrics to the desired levels. To improve the utility of the Internet in delivering multimedia, adaptive streaming solutions as well as blockchain-based solutions are utilized to cope with the Internet's environment. Many recent studies attempt reducing the volume of signalling overhead generated by the aforementioned streaming solution quasi-efficiently. To further increase the efficiency of multimedia communications, novel signalling overhead and content access latency reduction solutions are investigated in this dissertation including: (1) the first two research topics utilize covert channels to reduce signalling bandwidth utilization while increasing the capacity of the multimedia network; and (2) the third research topic utilizes multimedia content access request management schemes to guarantee throughput values for servicing users, end-devices, and the network.

To my father Dr. Jamal Shamieh, and my mother Naila Farraj: it is impossible to adequately thank you both for everything you have done and sacrificed. I could not have hoped for more inspiring, patient, and kind parents.

Acknowlegements

First and foremost, I would like thank my supervisor, Dr. Xianbin Wang, for his continuous support, guidance, patience, and motivation. He encouraged me to pursue a deeper understanding of my areas of interest.

I would also like to acknowledge the Faculty of Engineering, especially the Department of Electrical and Computer Engineering at Western University for the excellent facilities, libraries, and resources available to graduate students.

In addition to my supervisor, I would like to thank Dr. Aydin Behnad for his mentorship, guidance, and knowledgeable/insightful conversations. I would like to thank Dr. Tadilo Bogale for his valuable insights, the countless coffee meetings, and research guidance. I would like to thank Dr. Ahmed Refaey for guiding and supporting me throughout my academic career. I would like to thank Dr. Hao Li for his continuous guidance and support as well as the countless hangouts and conversations that made the days go by easier.

I would like to thank Dr. Auon Akhtar for the countless meetings and phone calls as well as his full-time guidance and mentorship throughout my graduate academic career. Thank you for helping even when you were extremely busy and a few hundred kms away. I will always be grateful for your time. I would like to thank Dr. Mohamed Abu Sharkh for his continuous mentorship and guidance. I will always be thankful for guiding me through the tough times and for our wonderful chats on C-137. I would like to thank Dr. Mohamad Kalil for always looking out for me and guiding me whenever I needed help. I would like to thank Dr. Abdallah Al-Moubayed, Dr. Khaled Alhazmi, Dr. Mohamed Hussein, Dr. Tamer Abd El-lateef, Dr. Samantha Gamage, Dr. Fadi Salo, Dr. He Fang, MohammadNoor Injadat, and Anas Saci for the numerous conversations, their continuous guidance and mentorship. I am deeply grateful for my friends and colleagues. Special thanks to my brother and amino acid Dr. Antoan Antoun for being a true comrade, Weldon-trench partner, grass cutter, and a motivator for getting over obstacles. Special thanks to Hung Pham for being a true friend, spending countless

hours listening to my ingenious ideas, and supporting me on my adventures. Special thanks to Steven Andrews for helping through some of the toughest times and being someone I can count on. Steven, you saved me a number of times in the past and I am eternally grateful. Alfred Kenny, Ali Al-Saady, Walid Ali, Sorin Popa, Asma Khalil, Allan McCulloch, Jennifer Reid, Scott Van Heesch, Hassan Hawilo, Manar Jammal, Elena Uchiteleva, Soroush Arghavan, Ester Turečková, Bilal Al-Bataina, Mike Zylstra, Cesar Suarez, and Sabin Bhandari; your friendship and presence positively and greatly contributed to my life and for that I am eternally grateful. Special thanks to Nadine Abdulkarim for supporting me throughout my research, motivating me to achieve my goals, and being my rock. This work would not have been possible without your love and support; you are a blessing from the skies.

Special mention to Twix, Missy, Moomba, Xena, Daria, and Lucy, my 3- and 4-footed fur babies.

Last but certainly not least, I would like to thank my parents and family for their endless love, support and encouragement.

Contents

List of Figures

List of Tables

List of Abbreviations

ACK	Acknowledgment
AIMD	Additive Increase Multiplicative Decrease
BCH	Bose-Chaudhuri-Hochquenghem
BL	Base Layer
BS	Base Station
CDN	Content Distribution Network
D2D	Device-to-Device
DCCP	Datagram Congestion Control Protocol
DCT	Discrete Cosine Transform
DRM	Digital Rights Management
DT-CWT	Dual-tree Complex Wavelet Transform
DWT	Discrete Wavelet Transform
EL	Enhancement Layer
H.264	MPEG-4 Advanced Video Coding
HVEC	High Efficiency Video Codec
HAS	HTTP Adaptive Streaming
HTTP	HyperText Transfer Protocol
ICT	Information and Communication Infrastructure
IP	Internet Protocol

IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
ISP	Internet Service Provider
LSB	Least Significant Bit
LXC	Linux Containers
MPD	Media Presentation Description
NAC	Network Address Code
NAT	Network Address Translation
NS3	Network Simulator 3
PVD	Pixel Value Difference
P2P	Peer-to-Peer
PPBP	Poisson Pareto Burst Process
PPS	Packets Per Second
PSNR	Peak Signal-to-Noise Ratio
QoS	Quality-of-Service
QoE	Quality-of-Experience
RGB	Red Green Blue
RTCP	RTP Control Protocol
RTSP	Real-time Streaming Protocol
RTP	Real-time Transport Protocol
RTT	Round-Trip Time
RTO	Retransmission Timeout

SDN	Software Defined Networking
SR	Simplified Routing
SVC	Scalable Video Coding
TCP	Transmission Control Protocol
TPVD	Tri-way Pixel Value Difference
URL	Uniform Resource Locater
UDP	User Datagram Protocol
VLC	VideoLan Media Player

Chapter 1

Introduction

High fidelity multimedia streaming over the Internet has proliferated in modern society due to the ever-increasing ties amongst people and businesses, thereby occupying the majority of all exchanged traffic. The Internet is a popular multimedia exchange medium due to its wide access availability; however, it is being used outside of its original design. In the next few years, the Information and Communication Technology (ICT) infrastructures, particularly the Internet, is expected to service traffic that is 80% from multimedia origins with at least 82% in high definition [1]. However, due to the inherent properties of the Internet, there is a mismatch between the underlying infrastructure and the diverse nature of multimedia streams resulting in unwanted Quality-of-Service (QoS) and Quality-of-Experience (QoE) degradation.

1.1 Research Motivation

Initially, Internet Protocol (IP)-based networks are designed to transfer bytes of data in a one-size-fits-all fashion over their non-deterministic paths; however, they are currently used for exchanging multimedia streams in ubiquitous manner [2]. To overcome the non-deterministic paths of the Internet while servicing a large user-base, adaptive multimedia streaming technologies [3] and blockchain-based systems [4] are employed to their full potential. The outcome of utilizing adaptive multimedia streaming is a cost-effective service deployment in terms of hardware availability, however, it is costly in terms of total generated signalling overhead from

every layer of the IP communication stack. Similarly, utilizing blockchain-based solutions to enable decentralized content access will service a large number of users, however, at the cost of induced processing latencies and reduced service throughput. The lack of initiative from Internet Service Providers (ISPs) to upgrade the inadequate infrastructure to better service multimedia traffic promoted the proliferation of signalling overhead reductions mechanisms at every layer of the IP stack and enhanced decentralized multimedia access technologies. The mismatch between the Internet and audiovisual applications is the primary driver behind inducing non-payload overhead as well as processing latencies; therefore, appropriate mitigation will provision resources to deliver expected QoS and QoE requirements.

Traditional approaches to reducing control overhead traffic and enabling large scale centralized source-client multimedia models include the use of caching nodes and distribution centres located near concentrations of users [5, 6]. Although cache nodes are the ideal choice for adaptive multimedia streaming services due to their simplicity, they are extremely costly, require continuous upgrades, and are not sufficient to overcome the rapid growth of multimedia applications. Recently, signalling overhead reduction methods designed to decrease the generated overhead costs at every layer of the IP communication stack have emerged. The layer-specific solutions are effective tools in improving the efficiency of multimedia streams while complementing traditional caching approaches as well as alleviating intermediate node buffers. To further improve control overhead reduction schemes, cross-layer solutions are needed and implemented through using steganography within multimedia streams. The inherent data hiding properties of multimedia payloads are being actively exploited by Content Distribution Networks (CDNs) to deliver enhanced services to end-users or to combat theft as well as illegal distribution of intellectual property. Due to the nature of multimedia payloads, they are ideally suited for data hiding and creation of covert communication channels with large bandwidth capacity. The unique data hiding properties of multimedia are not only appropriate for enhancing user experience but also for improving the communication efficiency over the erratic paths of the Internet.

HyperText Transfer Protocol/Transmission Control Protocol (HTTP/TCP) is a combination of mature IP protocols that enable adaptive multimedia streaming services, such as HTTP Adaptive Streaming (HAS), due to their ability to leverage existing hardware as well as wide device interoperability. However, HAS-based streaming services are susceptible to packet loss, network congestion, and non-deterministic behaviour due to the underlying infrastructure. The existing control overhead reduction methods found in the literature [7–27] operate individually at each layer of the IP communication stack without coordinating with each other. Furthermore, they are inefficient in terms of responding to change in network conditions as well as on-demand requested fidelities. Overcoming the limitations of existing signalling overhead reduction techniques is achievable through cross-layer cooperation of the IP stack. Cross-layer overhead mitigation will require exploiting the steganographic capacity of multimedia streams to embed and exchange control overhead thereby alleviating delivery networks from processing non-payload data. In terms of content access, server-client models of managing multimedia are expensive as more hardware and infrastructure is needed as the demand increases, and therefore blockchain-based solutions [28–35] have emerged. Blockchain-based technologies enable scalable multimedia access, however, the processing latency of said management systems are inherently large due to the secure nature of ledger-based networks. Without undermining security features of a blockchain-based system, efficient ledger processing disciplines are needed to reduce content access latency and enable throughput provisioning. It is therefore of great importance to enhance the efficiency of multimedia systems through improved communication streams as well as seamless access to content.

1.1.1 Gauging Signalling Overhead Volume in Multimedia-based Applications

In order to overcome the challenges of delivering multimedia over the Internet, service providers utilize adaptive multimedia streaming technologies as well as decentralized content access management as inexpensive solutions. Adaptive streaming technologies enable end-

to-end multimedia delivery through sophisticated and coordinated signalling overhead traffic flows [36]. At the application layer, the employed adaptive streaming technologies require receiving processes to request multimedia chunks through signalling messages on a segment-by-segment basis. Subsequently, every byte of data generated at the application layer is transported utilizing lower-layer protocols that commission additional signalling messages to guarantee the delivery of said bytes. Consequently, due to the Internet's inherent time-varying properties as well as a client's changing demands, generated signalling overhead volume is shifting accordingly. The time-varying properties of network paths will induce packet losses, retransmissions, multimedia segment size change, and multimedia fidelity requests, at the expense of additional inefficient signalling overhead exchanges. Furthermore, multimedia request-response transactions and traffic are intermittent in behaviour with bursty tendencies. As a result, the signalling overhead generated needs to be mitigated to alleviate source-destination pairs as well as intermediate nodes from having resources consumed to process non-payload data. Unlike centralized server-client models, decentralized content access management schemes are necessary to enable efficient distribution models that are able to scale to meet significant multimedia demands efficiently. Through using blockchains, service providers are utilizing a secure and Peer-to-Peer (P2P) capable technology that enables efficient service scalability [37]. The security features of a blockchain, however, introduce large processing overheads resulting in reduced perceived QoS values.

To properly mitigate the signalling overhead volume in adaptive multimedia streaming technologies, such as HAS, the inducing processes must be identified. HAS is HTTP/TCP based, and therefore all exchanged data is encapsulated using the TCP protocol. Using the TCP protocol allows sharing large sized segments and control messages with delivery guarantees, however, it utilizes Acknowledgement (ACK) messages to drive segment transmissions [38]. Due to the nature of the Internet paths and the IP protocol stack, segment sizes are limited, and therefore, numerous segments must be exchanged in the form of packets to achieve delivery. Consequently, the number of ACK messages exchanged during a multimedia trans-

mission is inevitably large. Furthermore, all packet losses and retransmissions will generate another round of ACK messages to close the gap of missing packets. It can be seen that due to the nature of multimedia streams, HAS generates a large amount of control overhead at the transport layer of the IP communication stack.

Reducing signalling overheard does not reduce the processing overhead introduced by using blockchains for scalable multimedia content access management, and by extension, user authentication. The inherent security features built into a blockchain-based system introduce a significant processing overhead and therefore pose as an inefficient method of driving system and network events. In blockchain-based financial systems, processing fee structures are used to increase the efficiency of a system. However, blockchains used in multimedia services are not capable of using fee-based structures, and therefore, appropriate processing schemes need to be investigated. An event driving mechanism is needed to reduce the processing overhead introduced by the security features of a blockchain system for specific network tasks.

1.1.2 Efficient Multimedia Signalling Exchange Methods for Payload Delivery Over the Internet Medium

Identifying the processes contributing to the exchanged control overhead allows appropriate solutions to be designed and implemented. The types of solutions to reduce control overhead exchanged as a result of multimedia streams is classified into two categories: application layer solutions and lower layer-based solutions. Application layer solutions target reducing the number of request-response transactions performed by a source-destination pair. Lower layers-based solutions tend to address the signalling overhead generated by the transport layer and networking layer of the IP communication stack. Application layer solutions tend to modify the features of HAS-based streaming through manipulating the types of segments provided, their frequency, and their size, among other features. Lower layer-based solutions tend to modify the features of the protocols providing end-to-end connectivity and delivery. Modifying the features of lower-layer protocols includes reducing the frequency of exchanged messages, the

size of the headers used, and using alternative protocols, to name a few [39].

The end-to-end delivery is conditional on the network paths, perceived congestion, inter- and intra-ISP agreements, and node buffers, among others. The inherent multimedia data hiding capacity can be leveraged beyond providing auxiliary services or piracy combating watermark schemes to create efficient signalling exchanges. The covert channels are used to either transport complete control messages from any layer in the IP protocol stack or to be used as a protocol translator. The protocol translator can hide important protocol information within the multimedia payload and use a lightweight alternative protocol for end-to-end delivery. In one-hop communications, inflated routing protocols and their headers are not necessary for end-to-end delivery. Due to the limitations of one-hop communication networks, a simple one-hop routing scheme needs to be investigated. Finally, a blockchain-based multimedia content access management system is capable of handling a degree of fault tolerance. The fault tolerance induces processing latencies and reduces throughputs for different components of the system. Efficient event validation is performed through appropriate event-by-event processing where a feeless tiered structure is needed to introduce a network driving mechanism.

1.1.3 Decentralized Multimedia Access Management and Copyright Protection

Multimedia transmissions over the Internet should not only be efficient in exchanging control overhead but should also be efficient in scaling to meet demand volume, essentially managing processing overhead. Blockchains play a vital role in enabling secure decentralized technologies, however, they lack the proper methods of processing events and reducing related overhead. Existing solutions providing methods for processing blockchain events are geared towards blockchain-based digital currency services [41–60] and are therefore not suitable for multimedia services. The incompatibility between the available blockchain processing methods and need of multimedia services promotes the investigation of an efficient event driving scheme. A candidate solution to the aforementioned gap of processing methods is to use a

feeless, priority-driven scheme, for processing multimedia system and network events. The priority-driven blockchain processing scheme will be designed to further enable blockchains outside of digital currency applications while reducing processing overhead as well as increasing the efficiency of multimedia services.

1.2 Dissertation Contributions

The main contributions of this dissertation are summarized as follows:

- In order to reduce the amount of signalling overhead exchanged in multimedia streams, it is necessary to identify all entities contributing to the signalling stream. Once the sources are identified, a novel cross-layer steganographic-based solution is proposed and utilized to embed the generated signalling messages in a unique manner within multimedia payloads to reduce the amount of exchanged overhead. The proposed scheme utilizes the data hiding capacity of multimedia files to embed control messages within audiovisual data where the receiving node will perform a decoding process to extract the hidden signalling data. A utility function is designed to reduce the amount of overhead exchanged while ensuring a low amount of visual artifacts perceived by a user. The proposed scheme is capable of extending existing overhead reduction methods to further decrease the amount of exchanged control messages.

- Since smaller sized segments are more adaptable to network conditions, a novel steganographic-based packet size reduction scheme, as well as a simplified routing scheme, is proposed. Unlike existing stream sensitive solutions, the proposed steganographic scheme translates protocols with a large header size to a protocol with reduced header sizes. The translator operates through embedding packet header information within multimedia payloads, and therefore are less sensitive to streaming conditions. In the specific case of one-hop communications, a simplified routing scheme is used in place of the current network layer protocol to perform end-to-end routing while using a smaller sized

header. A utility function is designed to reduce the size of packets through performing translations while guaranteeing low visual artifacts.

- A novel blockchain ledger maintenance algorithm is proposed to increase the efficiency of decentralized content access management systems used in multimedia services. Existing ledger maintenance schemes utilize digital currencies for efficient tiered processing and therefore limited to currency based applications. The proposed algorithm is feeless and designed to be used in any industrial blockchain-based application free from digital currencies. The proposed solution will increase the efficiency of multimedia services through selecting the appropriate events to process at the correct time. The proposed algorithm is a block slicing technique designed to allow ledger maintainers in provisioning throughputs and reducing latencies as needed.

- The aforementioned solutions are capable of cooperating with existing methods to further enhance their performance. More specifically, the steganographic-based schemes are designed to work solely on their own or through extending existing solutions. A test-bed is designed and used as a proof-of-concept implementation to demonstrate the proposed steganographic schemes. Finally, the blockchain-based solution is capable of enabling additional tiered processing for currency as well as non-currency-based applications.

1.3 Dissertation Organization

The following details the organization of the remaining chapters of this dissertation:

Chapter 2 provides the details of adaptive multimedia streaming technologies, multimedia encoding formats, existing solutions to reducing exchanged signalling overhead, steganography in multimedia, fundamentals of blockchain technologies, and existing blockchain-based multimedia services.

Chapter 3 focuses on the proposed cross-layer steganographic-based signalling overhead reductions scheme, where a utility function, as well as a real-time algorithm, are designed to

reduce the visual artifacts perceived by users.

Chapter 4 focuses on the proposed steganographic-based header size-reduction scheme enabled through protocol translation as well as the simplified routing solutions. Furthermore, a utility function, as well as a heuristic algorithm, are given to evaluate the cost savings while maintaining a low visual error.

Chapter 5 details a feeless blockchain processing scheme designed for industrial application of blockchain technologies, such as multimedia content management and user authentication services. A blockchain processing algorithm is given to ensure the driving of system and network events to achieve desired request throughputs. The designed processing algorithm is an efficient way of reducing processing overhead while attaining specific throughputs for events from different priority classes.

Finally, in Chapter 6, concluding remarks as well as future research directions are given.

Chapter 2

Challenges and Solutions of Multimedia Streaming Technologies over IP-based Networks

This chapter summarizes the signalling overhead generating sources found in multimedia services as well as the effects of the lacking underlying communication infrastructure used for exchanging audiovisual data. Adaptive multimedia streaming, data hiding, and blockchain-based systems enable digital multimedia services and therefore are widely adopted by content providers as secure solutions for end-to-end delivery of segments. The aforementioned technologies, the insights into control overhead reduction solutions as well as scalable multimedia rights management systems are addressed in this chapter.

2.1 Interoperable Visual Encoding and Network Signalling Formats

The Advanced Video Coding (H.264/AVC) format is the most popular video compression representations used to decrease multimedia storage and transmission bandwidth [61]. All exchanged multimedia on all devices and services is compressed using an encoder and decompressed using a decoder. Prior to performing the compression process, the source multimedia input is first processed in the prediction block where it is treated in blocks of 16x16 pixels

termed as Macroblocks (MBs). Using previous blocks and data within the current frame, the encoder will try to predict a related block and correct the prediction using MB subtraction processes. The prediction phase essentially forms MBs based on the previously coded data from the current (intra) frame or previously (inter) coded frames. The predication phase performs a motion estimation process to find a suitable inter- and intra-predication frame descriptions. Subsequently, a motion compensation process subtracts a prediction block from the current MB block to construct a residual MB description. In doing so, the encoder is only informing the decoder of residual blocks where the decoder will reconstruct the frames using the inverse method of forming residual blocks.

Residual blocks are generated subsequent to the subtraction process where a 4x4 or an 8x8 Integer Transform (IT), an approximate form of the Discrete Cosine Transform (DCT), is performed on said block of residual samples. The DCT process provides insights on parts of the image that have different importance with respect to the image's quality through a transformation from the spatial domain to the frequency domain. Each of the output coefficients of the forward transform phase are weighted values of standard basis patterns. The inverse transform uses the combination of coefficient values and basis patterns to reconstruct image blocks. The H.264 format is designed to provide network friendliness through Network Abstraction Layer (NAL) packet units. The NAL units are capable of mapping H.264 Video Coding Layer (VCL) information generated through the encoding process to organized packets. The organized NAL units are also used with non-VCL information to exchange playback control information. Non-VCL NAL units will contain data, such as Sequence Parameter Set (SPS) and Picture Parameter Set (PPS), to enable the decoder in performing a correct playback [62].

H.264 variations of media encoding outputs are suitable to meet the required video representation for storage, however, they can fail at meeting the various transmission scenarios with unpredictable network conditions. Scalable media streaming is used to create layers or sub-streams from a single media source where each additional layer received enhances the received fidelity, as illustrated in Fig. 2.1. The layers are either sent with hierarchal dependence

in Scalable Video Coding (SVC) form or they are individually decodable and are sent in Multiple Descriptions Coding (MDC) form [63]. If a video stream is layered using MDC, each description sent must carry enough information about the original video in order for the decoding process to occur. Carrying additional information about the original source video results in a reduction of compression efficiency and combining multiple MDC descriptions together requires a high computational power [64]. The reduced compression efficiency and the need for a high computational power to combine multiple descriptions of MDC promoted the use of SVC instead.

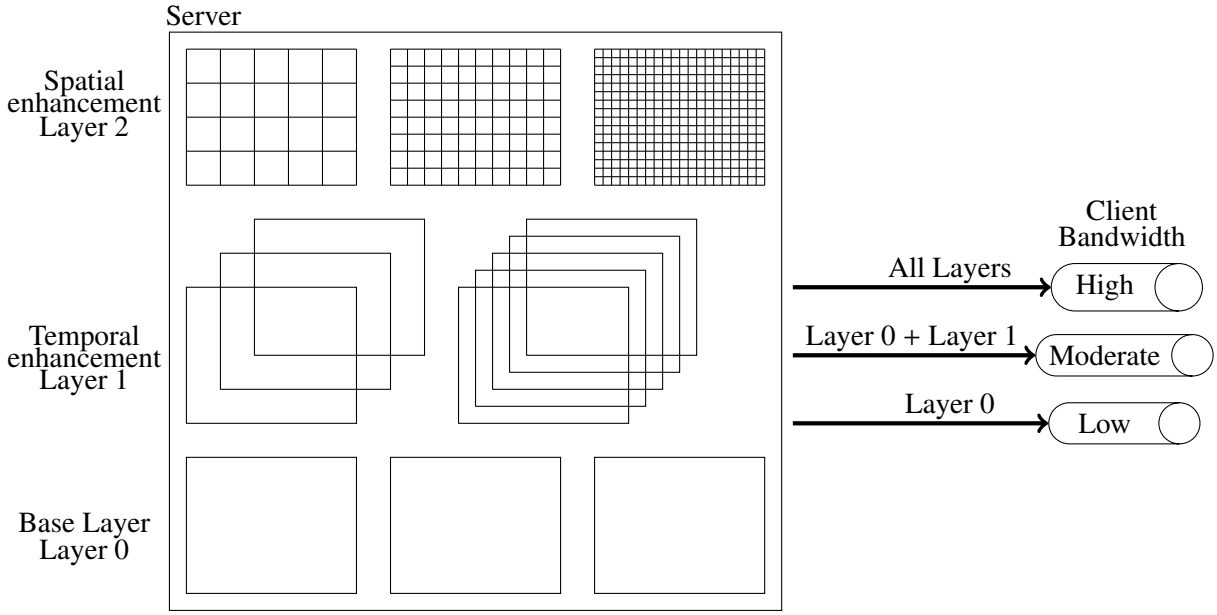


Figure 2.1: As clients request additional media representations, the media fidelity is enhanced. Layering video streams using SVC enables the receiver to progressively enhance received video quality as additional descriptions are received. The most important layer in SVC is known as the base layer (BL) and each additional layer is known as an enhancement layer (EL).

2.1.1 Adaptive Multimedia Streaming Sporadic Traffic and Signalling Sources

Streaming multimedia over the Internet is made available through application layer protocols including: Real-time Transport Protocol (RTP), RTP Control Protocol (RTCP) [67], Real-

time Streaming Protocol (RTSP) [68], and Hypertext Transfer Protocol (HTTP) [69]. The RTP protocol transfers media segments over User Datagram Protocol (UDP) where the RTCP protocol is used for congestion control as well as configurations to meet Quality-of-Service (QoS) requirements. Since multimedia is an error-tolerant function, intuitively, UDP will seem to be the optimal transport layer protocol for streaming as it is a mature protocol, with a small header size, and is inherently connectionless where data is not tracked nor retransmitted. All of the mentioned UDP-based streaming paradigms fail at traversing Network Address Translation (NAT) tables and firewalls and also require investments in network infrastructures and dedicated servers to manage separate streaming sessions. The dedicated servers are needed to compensate for UDP's stateless nature where no connection information is retained by the source-destination pairs [70]. Although initially deemed unsuitable for multimedia streaming [71], Transmission Control Protocol (TCP) is the preferred transport layer protocol for delivery due to inherently being stateful, connection-oriented, byte-oriented, and as a result reliable. The preferred application layer protocol to be used with TCP is HTTP where HTTP/TCP-based streaming is formally known as HTTP Adaptive Streaming (HAS).

The design of RTP/RTCP promotes functionality that is well suited for UDP but not for TCP-based connections and therefore are not used alongside TCP. Although RTSP utilizes TCP, it is a stateful protocol, which unlike stateless HTTP, is expensive to scale and consequently not as popular as HAS-based streaming. HAS virtually enables multimedia delivery to all devices connected to the Internet due to leveraging existing hardware and popular HTTP/TCP protocol combination, thereby minimizing deployment costs. TCP's features including: built-in feedback and congestion control mechanisms, are enabled by Acknowledgement (ACK) messages returned from receiving clients to source nodes [18]. Due to HTTP's need of TCP's features, the TCP ACK messages are overlooked since an HTTP-based streaming service is far less complex and therefore cost-effective. The benefits of HTTP are twofold; firstly, HTTP is designed to service clients at large scales effortlessly; secondly, firewalls and other network devices are easily configured to allow HTTP-based traffic [70]. HTTP is capable

of scaling due to the use of caches, widespread protocol use, and efficient hardware support as a result of HTTP's maturity. A significant contributor to HTTP's scalability is shifting the multimedia stream control to the client from the server-side through having clients send HTTP control messages, such as GET requests. A client in control of the stream will adapt the multimedia quality according to the observed network conditions to maintain a filled buffer, thereby reducing playback interruptions. A simple method for a client to detect the need for reducing playback quality is by using TCP's built-in congestion control mechanisms.

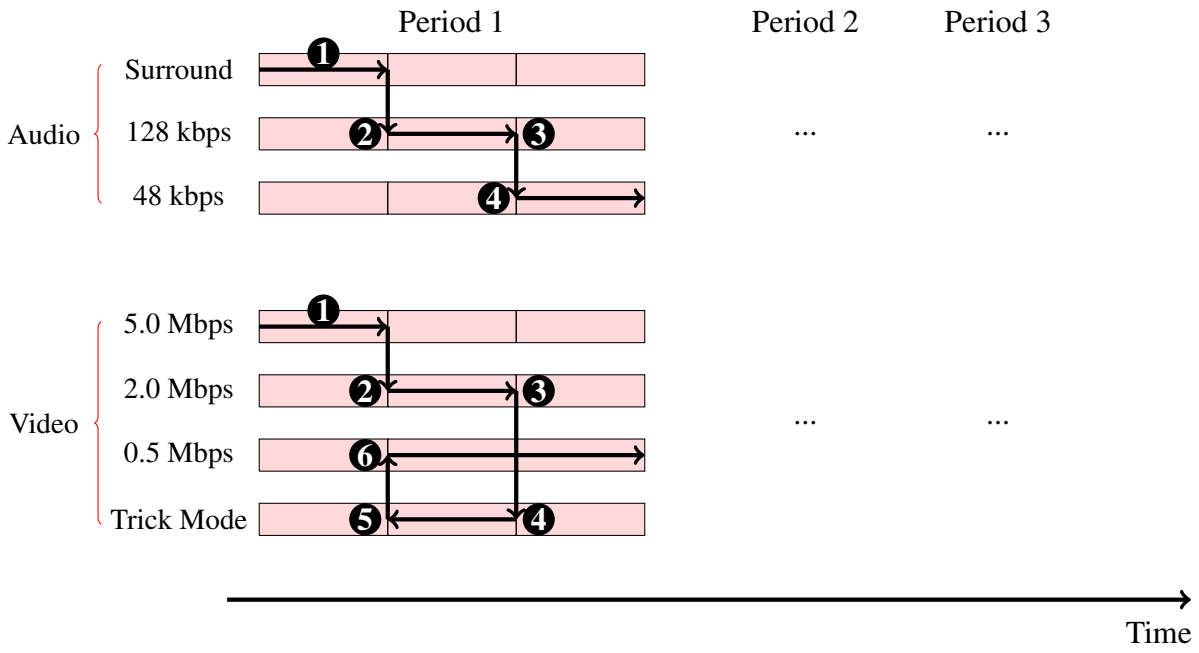


Figure 2.2: The client will utilize HTTP GET requests to choose different fidelities of multimedia descriptions suited for observed network conditions.

Clients will request multimedia descriptions of different qualities when needed, where each segment of a multimedia file is divided into multiple chunks with known starting points as shown in Fig. 2.2. A Content Distribution Network (CDN) will present clients with Media Presentation Description (MPD) [72] files to outline the available formats, bitrates, and URL addresses of available media segments as illustrated by Fig. 2.3. The MPD files contain metadata needed by clients to construct appropriate HTTP Uniform Resource Locator (URL) paths to retrieve desired segments using a chain of HTTP GET requests. The MPD files will describe

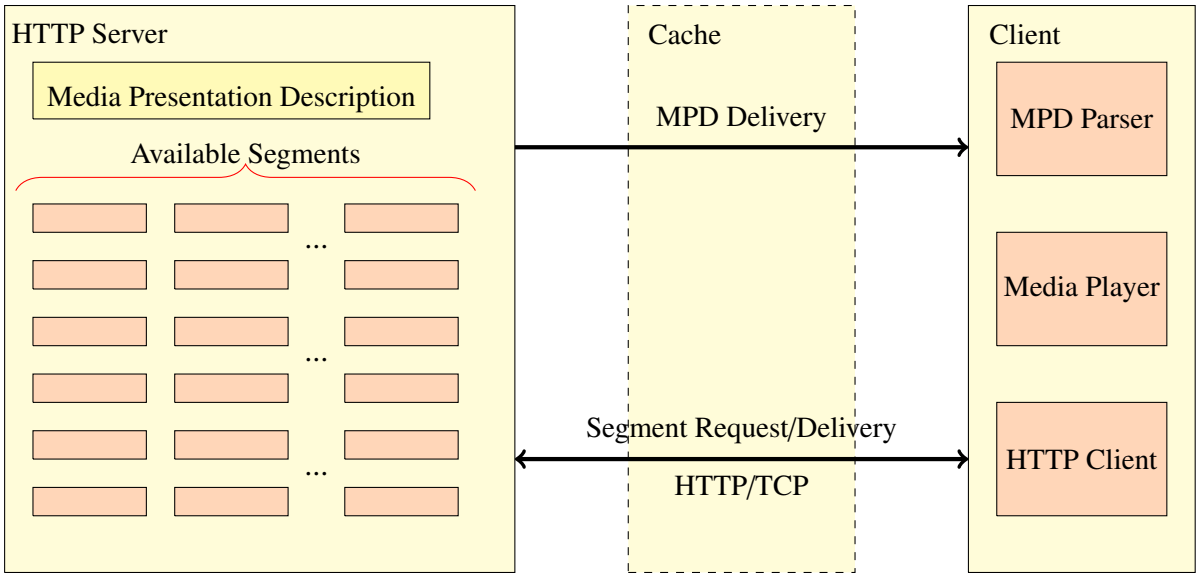


Figure 2.3: HAS utilizing HTTP as well as cache servers for efficient delivery.

the different video adaptations of the same segment, along with the different representations, available at the server during each period. For each segment identified belonging to all the representations described in the MPD file, a URL is available for the client to complete a desired segment retrieval. An MPD file is illustrated in Fig. 2.4 to show the hierarchical data structure that is used to represent deliverable versions of multimedia. The ability of seamlessly switching between segments with different representations is made possible with the media content being mapped to a global synchronization presentation timeline. For on-demand content, all segment availability and duration times are primarily identical with minor variations. For live content, segment availability occurs over time and are ideally short in length to meet end-to-end latency constraints. The simplicity of streaming multimedia using HTTP/TCP-based methods placed HAS as the underlying solution for content delivery at large multimedia enablers including Apple's HTTP Live Streaming, Microsoft's Smooth Streaming, and Adobe's HTTP Dynamic Streaming technologies.

The decoupling between the multimedia payload and the HTTP-based delivery service enabled the addition of new services and streaming requirements at little cost. HTTP-based services will easily stream new media formats, including three dimensional and ultra-high def-

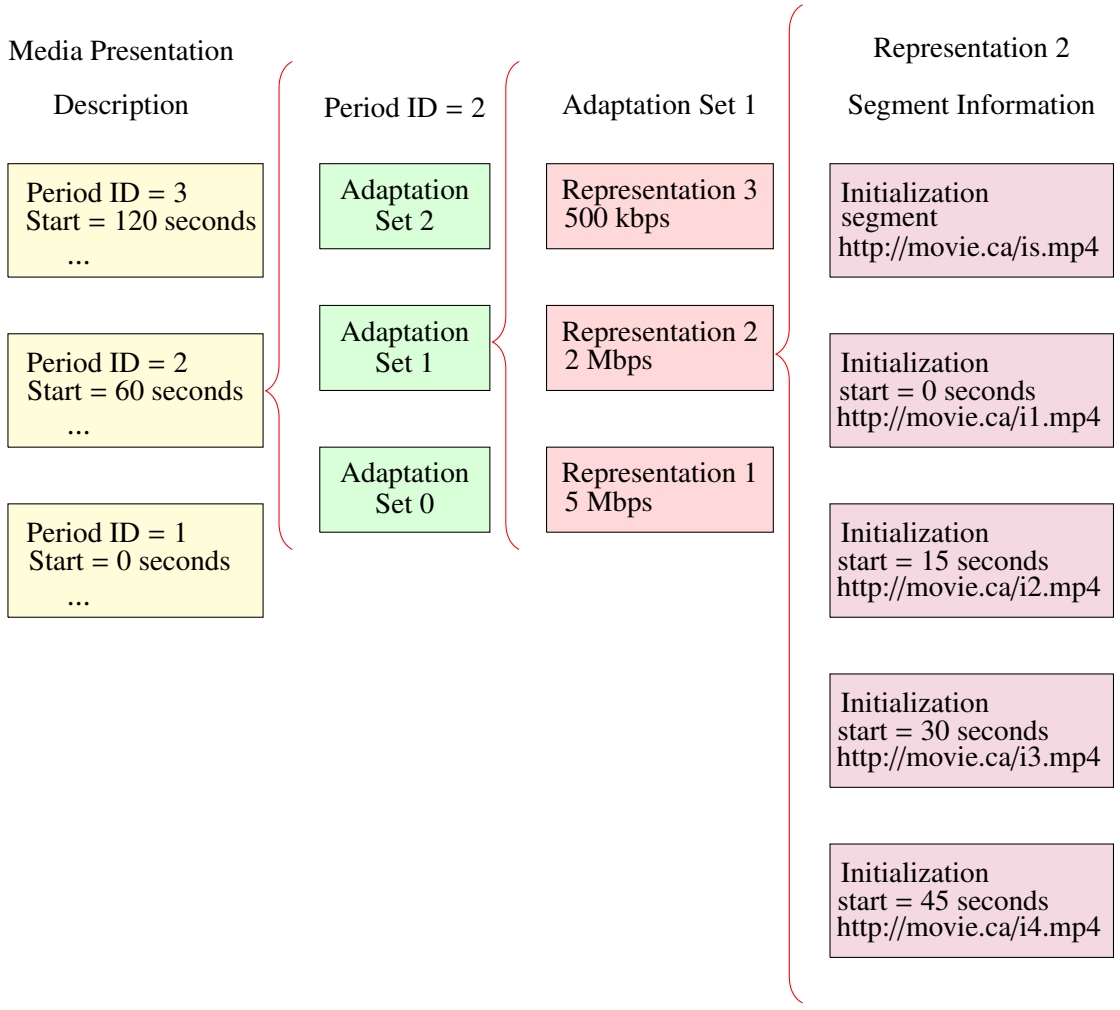


Figure 2.4: An MPD file is provided to the client for driving request-response interactions.

inition. Content providers are also capable of using new media codecs, such as SVC, on the fly. An ongoing challenge in the digital media rights protection world [73] is combating piracy where content providers are capable of doing so by extending HTTP-based streaming services as needed. The decoupling further enables a content provider in using different encapsulation formats, such as MPEG-2 Transport Stream (TS), as requested by a receiving client [3]. Another advantage of the decoupled model is the possibility to use HAS in topological systems including: multicasting [74–77], Network Coding (NC) [78–81], and Peer-to-Peer (P2P) streaming [82–86] configurations. HTTP-based delivery enables addressing privacy concerns of the consumers through leveraging end-to-end encrypted tunnels (HTTPS). The data tunnels

Layer	Service
Application	HTTP
	TLS
Transport	TCP
Network	IPv4/IPv6
Data-link	
Physical	

Figure 2.5: The TLS protocol rests above the transport layer in the application layer where it encapsulates all application generated data when enabled for a specific service. Due to being an application layer service, TLS will not encrypt any information observed at the transport layer and below.

are encrypted using an asymmetric and symmetric key cryptography with a key known only to the source-destination pair. The cryptographic Transport Layer Security (TLS) [87] protocol encrypts (decrypts) application layer information on-the-fly and is therefore easily utilized with multimedia flows. To verify that the ownership of a set of asymmetric cryptography keys, including the public keys of a service provider, Certificate Authorities (CA) are widely used. A digital certificate issued by a third-party CA is utilized to assert the public keys of a service to verify the signature of the author of the exchanged payloads. To verify a digital signature, asymmetric cryptography plays a vital role. The true author of a message initially encrypts and signs a message using a private key that is known only to the author. The receiver is capable of verifying the author using the author's known public key. Once the TLS session is established using the asymmetric keys, the TLS protocol cryptographically encapsulates all HTTP messages and payloads using the exchanged symmetric key. Therefore, TLS encryption is limited to the application layer, leaving transport layer data unencrypted as illustrated by Fig. 2.5. The TLS protocol supports multi-service configurations where end-nodes will request specific servers (services) in plain-text while establishing the TLS-session. However, in order to establish a TLS session, a TCP connection is first initiated between source-destination pairs. All of the messages generated by the TLS protocol will then be exchanged using the underlying TCP protocol.

The TLS protocol is a stateful connection that is established using plain-text exchanges, asymmetrically encrypted messages, and finally symmetrically encrypted payloads. The TLS

handshaking process will introduce a delay in establishing a communication session. Furthermore, additional signalling messages are exchanged between source-destination pairs to maintain the encrypted session, including key updates and session identification changes. Clients will initially send a Client Hello message to a server with the TLS version used, the available cipher suites, and the name of the requested server. The server will respond to the client with a Server Hello message that contains information on the supported TLS versions and session identification information. The server will then send a digital certificate that can be easily verified using information from CAs. Once the certificate is verified, the client-server pair can now communicate using asymmetric cryptography to exchange the key utilized for the symmetrically encrypting the payload. The client and server will utilize asymmetric cryptography to encrypt and exchange random strings and bytes of data that is utilized in generating the session key used in the symmetric cryptography session. Once the client and the server compute the key, the client will send the server a Finished message that is encrypted using the computed key; similarly, the server will send the client a Finished message that is encrypted using the computed key. All of the application layer payloads generated subsequent to computing the symmetric key is now exchanged in an encrypted format using the aforementioned keys. The symmetric key, however, is periodically renegotiated between source-destination pairs to enhance the security precautions. Due to the utilization of asymmetric and symmetric cryptography, the integrity of the data is verified using author signatures and verifications. Since clients utilize CAs to verify servers, service providers are now capable of signing all generated payloads using their private key. Consequently, a client will need to verify the signed payloads using the server's known and verified public key.

Whether HAS is used alone or alongside a topological delivery solution, the continuously growing overhead needed in order for a multimedia exchange to occur will occupy intermediate node buffers and increase processing time; As a result, reduced service performance is experienced through induced playback freezing. As the intermediate and source nodes become busier with processing overhead data, limited resources are being allocated away from mul-

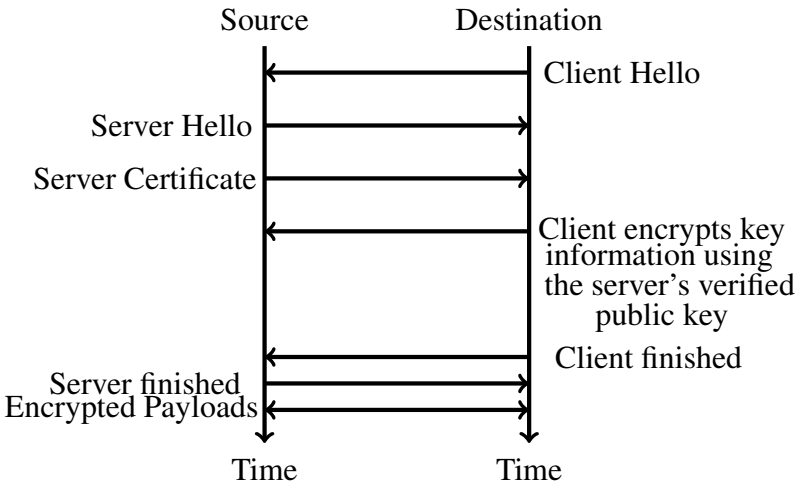


Figure 2.6: A TLS session begins with a handshake process and then moves to exchanging payloads. The TLS handshake process occurs after the TCP connection is established. In the TLS handshake process, source-destination pairs exchange sessions keys to cipher all application layer data.

timedia services to signalling plane handling. The excess overhead will lead to congestion and inevitably forcing multimedia applications to perform retransmissions. As the number of Round Trip Times (RTTs) needed to complete a transaction increases, the latency will behave accordingly. T. Flach et al. [88] demonstrate the effects of packet loss on TCP connections where flows will require longer times to complete the transfer of payloads. The retransmission timeout timer in TCP is one of the protocol's negative features as it introduces a delay that is larger than the RTT. In addition to multicasting, NC, and P2P technologies, HAS-based services will generate massive amounts of overhead at the application layer and by extension, at the transport layer, as the stream length, multimedia fidelity, stream configuration, and network conditions vary. Given that the Internet is a popular streaming platform, the inadequate underlying infrastructure furthers the overhead challenge as congestion and retransmissions will take place [89]. Unavoidable sources of latency, retransmissions, and excess overheads are due to the inter- and intra-ISP agreements and configurations for data paths, allocated bandwidth, and assigned priority class. Furthermore, the current communication infrastructure is non-deterministic and is geared towards file-based work-flows and is equipped with one-size fits-all protocols. This type of environment is not well suited for multimedia services leading

to inefficient use of network resources. The network will employ intermediate node buffering and increased end-to-end delivery time to compensate for the current lacking conditions [2].

2.1.2 Cost Effective Signalling Overhead Reduction Solutions

The wide adoption and deployment of HTTP/TCP adaptive multimedia streaming promoted the development of numerous signalling overhead reduction solutions. The proposed solutions address overhead generated at the application layer and at the lower layers in addition to using advanced multimedia compression schemes. A combination of the aforementioned solutions from each layer can be used simultaneously, however, as multimedia fidelity and service complexity increases, additional header mitigation solutions are needed.

2.1.2.1 Subsiding HTTP Redundancies and Payload Sizes of Request-Response Interactions

The popularity of HAS-based streaming echoed well within the research community as there numerous works of literature targeting overhead reduction. The existing overhead reduction solutions operate at the application layer, transport layer, and network layer [8–27]. Application layer solutions suggest changing traditional pull-based streams to push-based [8–15] to reduce the number of requests created by the client. The literature proposes using push capabilities built-in HTTP 2.0 where the server can send multiple multimedia descriptions and maintain a persistent connection with destination nodes without requiring a request for each description as illustrated by Fig. 2.7. Van der Hooft *et al.* [8, 9] promote the use of server push to deliver the base layer of SVC-HAS encoded media as well k segments to the client when receiving GET requests. Consequently, startup costs are eliminated, the number of RTT cycles decreases, and reduced playout freezes in high latency scenarios. Wei *et al.* [10] propose a scheme is designed to push k audio and video simultaneously when a request is received from a client to further reduce generated overhead. Xiao *et al.* [13] show that k-push schemes do not have a linear relationship between the value of k and the gained throughput, suffer from

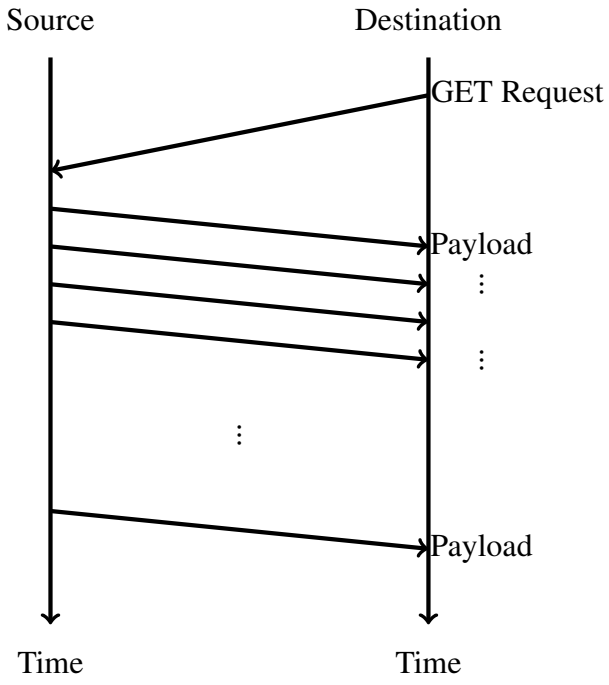


Figure 2.7: Server push technology available with HTTP 2.0 will reply to every request received from a destination node with k segments.

reduced network adaptability, and will promote over-pushing. Xiao *et al.* propose an adaptive push scheme where the value of k changes dynamically by having the system alternate between k-push and no-push states. Huysegems *et al.* [14] propose ten different server push schemes including: safety push, layered push, full push, and pull many, among others, to be used in adaptive streaming and as soon as the MPD request is received. The server responding with segments as soon as the MPD request is received eliminates startup delays. Clients will experience a small representation switching delay since the server leaped ahead by transmitting k segments of a previously requested fidelity. Therefore, when the k-push mechanism is employed, requesting a different bitrate and by extension, the k parameter method will incur additional delays. The results show that the incurred switching delays subsequent to using the k parameter are much more significant than the expected delay. Finally, the receiving nodes are capable of abusing the source by increasing the number of requests sent and as a result consume a more substantial portion of bandwidth. This is demonstrated by assuming a receiver that generates r requests in a small period of time and therefore forcing the source to respond

with $r * K$ audio/video segments.

To overcome the storage and bandwidth limitations, video encoding [62] is the first part of a bigger multi-part solution. Video encoding is a cost-saving tool that allows Content Delivery Networks (CDNs) to deliver higher-quality media to numerous users while reducing costs by virtue of using reduced amounts of bandwidth. As technological advances are made, such as cheaper and low-power chips, higher compression ratios are achievable through increasingly complex algorithms. Subsequent to capturing videos, uncompressed files are subject to video processing and signal analysis techniques prior to transmission in order to exploit the cost savings of video encoding. The uncompressed video consists of a sequence of ordered frames of real-world scenes where the aforementioned frames might have a high degree of statistical redundancies within themselves or with their neighbouring frames [65]. Video compression processes remove spatial and temporal redundancies from a sequence of frames where they are based on the perception limitations of the Human Visual System (HVS). Additional redundancies that are not perceptible by HVS are found through exploiting the high degree of inter- and intra-frame similarities [90].

With the proliferation of digital video and new emerging technologies that use the Internet and other non-deterministic communication channels, varying encoding bitrates of media are used to reduce bandwidth usage. Large CDNs including YouTube, Google, Apple, and Netflix, currently rely on employing HAS over TCP/IP [3, 5, 91], to adapt video services to current network conditions, end-device capabilities, and server load [92]. HAS operates based on client feedback, where the client measures available bandwidth and buffer status and the server provides videos in multiple representations to meet the various clients' capabilities. Overhead costs in HAS vary according to segment size, desired coding efficiency, and client feedback. Small segments will allow the HAS based systems to respond to network changes rapidly at a higher overhead cost, however, larger segments will be slower at responding to network changes while enabling higher coding efficiency at a lower overhead cost. Therefore it is vital to find the balance between segment size and desired coding efficiency to maintain a low

overhead cost [66].

2.1.2.2 TCP Alternate Configurations and Transport Layer Delivery Protocols

Reducing and manipulating ACK exchange behaviour as well as varying other transport layer configurations are promoted to decrease signalling overhead [16–19]. The aforementioned literature identifies the effects of manipulating TCP options as a means of reducing session overhead by varying the frequency of TCP ACK messages as well as employing Selective Acknowledgements (SACK). Manipulating the frequency of ACK messages will directly affect the growth of the congestion window accordingly. Delaying ACKs negatively affects the slow start phase of the congestion window since there are fewer ACKs being received in the slowest growth phase of the window. Using TCP SACK will reduce the number of ACK messages, however, while streaming real-time media, the packet being retransmitted may expire [19]. Allman in [18] propose two primary different delayed ACK schemes to reduce the number of exchanged signalling messages. The first proposed method is designed to generate delayed ACK messages once the TCP congestion window escapes the slow-start phases. Therefore, while in the slow-start phase, ACK messages are needed for growth until the congestion avoidance phase is reached. Once at the congestion avoidance phase, ACK message frequency is reduced where an ACK is exchanged every n segments. The second method utilizes Appropriate Byte Counting (ABC) to grow the congestion window. Unlike traditional congestion window growth that is based on the number of received ACK messages, byte counting is an alternative method independent of the number of ACK messages received. While using byte counting to grow the TCP congestion window, the window size will increase according to the number of acknowledged bytes by the received ACK messages, and therefore fewer ACK messages are needed to grow the congestion window. In order for the proposed methods to succeed, the source-destination pair must communicate to each other whether the transmission is in the slow-start phase or otherwise since the stream must be outside the slow-start phase to succeed. Furthermore, the proposed delay schemes will suffer in sessions where the connection needs to

reset due to the repeated occurrence of the slow-start phase.

Landstrom *et al.* [17] proposed a scheme that is designed to reduce the frequency of generated TCP ACK messages by manipulating the settings of the protocol. The aim of the proposal is to show that a stream's performance can be maintained when a reduced number of ACKs is transmitted. Furthermore, Landstrom *et al.* show that a source-destination pair does not achieve the best results when using the default TCP delayed ACK strategy, and therefore, a different ACK delay strategy is needed. Landstrom *et al.* show that in certain scenarios with favourable conditions, there is the potential to further reduce the number of ACKs transmitted from one ACK for every two-in-order segments to one ACK for every four-in-order segments. By reducing the frequency, the ACK messages are essentially delayed for delivery as illustrated in Fig. 2.8. In the default scenario, an ACK message is expected to be sent by a destination node for every segment successfully received from a source node. Furthermore, in [17], the proposed frequency manipulation solution utilizes ABC [18] to compensate for the reduced frequency of ACK messages to maintain the desired performance.

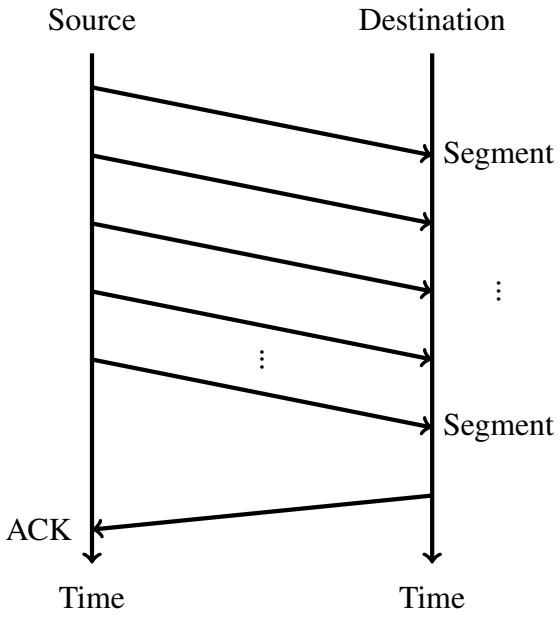


Figure 2.8: The TCP protocol is an ACK-based protocol that depends on the number of bytes successfully exchanged to transmit new segments.

Another solution involves reducing header sizes through compression [20–24] as an effec-

tive way of decreasing signalling overhead. The compression schemes include: Robust Header Compression (ROHC) [20], IP Header Compression (IPHC) [21], Compressed RTP (CRTP) [22], Enhanced Compressed RTP (ECRTP) [23], and Tunneling Multiplexed Compressed RTP (TCRTP) [24]. The proposed compression schemes are stream-based and designed to compress a variety of used protocols including UDP, TCP, and IP. Stream-based compression is sensitive to packet loss and therefore is not suitable for networks with non-deterministic behaviours, such as the Internet. Additionally, if header stripping is performed on a compressed packet at an intermediate node, a receiver will not be able to decompress the packet. The alternative to compression schemes is to use protocols that behave in a similar manner as TCP with a reduced header size [25, 26]. The Datagram Congestion Control Protocol (DCCP) is designed to replace and mimic TCP behaviour with 12 byte header. The DCCP protocol is not a mature protocol and lacks features found in TCP, such as tracking the receiver's buffer size and therefore is not as reliable [25].

2.2 Multimedia-based Steganography and its Applications

Current digital steganography is used unidirectionally and for a limited number of purposes, including copyright protection of video or audio files [40], content authentication, embedding subtitles [96], updating keys in multicast scenarios [97], and by definition, data hiding in covert channels [98]. Data hiding in covert multimedia is broadly categorized into two groups based on the application, namely watermarking and steganography. Digital watermarking is used in digital media to claim ownership of material or copyright protection while steganography is used to transfer data secretly. Digital watermarks are typically designed to withstand geometric attacks, compression, and other forms of media editing while remaining transparent. Steganography is concerned with maximizing the amount of embedded data while remaining imperceptible by insuring minimal distortion of the cover media. The importance of digital watermarking in combating piracy [73] is emphasized by the degree of easiness digital multi-

media is shared across the Internet. Digital steganography, watermarking, or data hiding in the most abstract sense can occur at any level in the TCP/IP abstraction as illustrated by Fig. 2.9 and is an inherent part of a multimedia system. The TCP/IP model inherently supports network steganography where protocol headers and behaviour are exploited for covert communications [99].

<table>
<tr><td>Application</td><td>Payload Steganography</td></tr>
<tr><td>Transport</td><td rowspan="3">Network Steganography</td></tr>
<tr><td>Network</td></tr>
<tr><td>Data-link</td></tr>
<tr><td>Physical</td><td>Physical Layer Steganography</td></tr>
</table>

Figure 2.9: TCP/IP stack separated into regions defined by the type of steganography used.

2.2.1 Data Hiding Basics in Bandwidth-Rich and Restricted Mediums

Digital steganography is implemented in multimedia using slightly less complex methods such as Least Significant Bit (LSB) embedding within pixel channels and more advanced methods such as LSB within the DCT and Dual-tree Complex Wavelet Transform (DT-CWT) coefficients. The different data embedding and data hiding techniques are measured using three factors including: capacity, transparency, and robustness as shown in Fig. 2.10. The three factors influence each other based on the desired quality, and hence, there is a tradeoff between them. For example, in a data hiding situation where robustness to geometric attacks is required with a minimum visibility of the hidden data to the user, capacity will be sacrificed as data can only be embedded in limited locations within the carrier object. Consider an image of size 512x512 pixels where each pixel in the image has three channels representing a colour from the Red Green Blue (RGB) colour gamut. The channel elements of each pixel are assumed to be represented by b bits. If the intensity of the pixel channel elements vary from 0 to 255, then 8-bits are needed to represent each element. To embed data within one of the pixel's channels,

the LSB values describing the intensities found within said channel are changed as necessary. Using the LSB method on a 512x512 image will allow approximately 33,000 bytes of data to be hidden per channel. Changing the value of an LSB in some pixels of a channel does not degrade the image quality significantly, however, the generated signal is susceptible to noise [96].

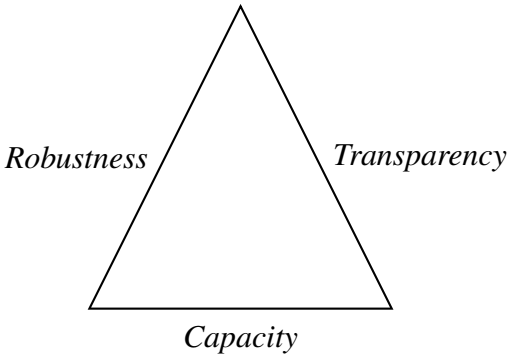


Figure 2.10: Trade-off triangle of steganographic features used in covert multimedia channels.

Advanced data embedding techniques used for embedding watermarks, such as methods that embed data within the DT-CWT transform coefficients, are more robust to noise and can withstand geometric attacks [100]. Watermarks are used in the multimedia industry for combating piracy by embedding an imperceptible payload in the content. The payload is detected in either a blind or non-blind fashion where the former does not need any reference frames and the latter will need some reference frames to extract the watermark [40]. Blind watermark extraction is a difficult process in comparison to its counterpart due to the absence of reference frames. A multitude of schemes are proposed in the literature to embed watermarks in audio and video content using different types of transformers including DCT, Discrete Wavelet Transform (DWT), and DT-CWT. Most of the latest watermarking methods use DT-CWT due to its shift-invariance, directional selectivity, and perfect reconstruction [101]. The DT-CWT is easy to implement since existing hardware can implement it using existing DWT software [102]. Given an RGB source, an encoder will transform that source into the Luma and Chroma

(YUV) channels using

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0 \\ 127 \\ 127 \end{bmatrix} + \begin{bmatrix} 0.299 & 0.587 & 0.115 \\ -0.169 & -0.331 & 0.500 \\ 0.500 & -0.418 & -0.082 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}. \tag{2.1}$$

The source will then process YUV according to the encoding process as necessary. Once the forward transform stage is reached, the watermark is embedded in the low frequency components of the DT-CWT coefficients as shown in Fig. 2.11. The extraction can be performed in a blind fashion by using the same embedding key in the decoder and following the DT-CWT procedure [101].

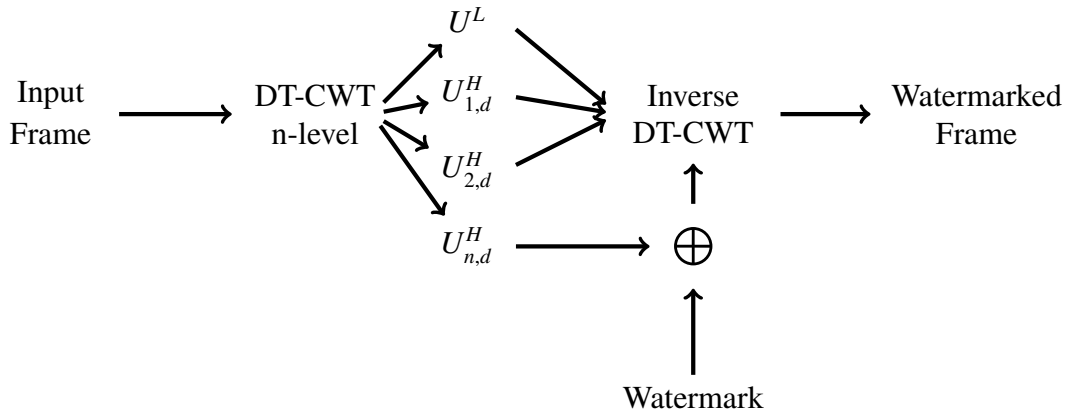


Figure 2.11: Performing a data hiding process within DT-CWT coefficients.

Using the aforementioned steganographic mechanisms, a number of algorithms and procedures [103–110] detailing how to embed data within real-time and on-demand H.264 streams are found within the literature. Furthermore, the steganographic methods are used in [111–115] to provide different services for and through multimedia transmission sessions including session key exchange and censorship bypass using covert channels. In [103], Priya and Amritha test steganoraphic operations including: Least Significant Bit (LSB), Pixel Value Difference (PVD), and Tri-way (TPVD), on H.264, H.265, and High Efficiency Video Coding (HEVC) formats. The encoded payloads are 25 to 125 bytes long where the Peak Signal-to-Noise Ratio

(PSNR) is found to evaluate the quality of the video post embedding. In [103], the embedding process occurs in the I, B, and P frames and is distributed between the Luma and Chroma channels of a 14 s long video available in various definitions and aspect ratios. The test performed on the H.265 multimedia formats showed that H.265 is the most resilient for data hiding especially when TPVD is used. In terms of H.264, LSB and PVD-based encoding operations achieved relatively the same quality changes for all embedding sizes. In [105, 105], Bose-Chaudhuri-Hochquenghem (BCH) codes are proposed to encode steganographic data prior to embedding within H.264 multimedia formats. The BCH-based code utilizes LSB to embed data within the DCT coefficients of multimedia frames. The proposed algorithm encodes data in the Luma DCT channel blocks of the I-frame. The proposed scheme is able to embed 280 bytes in 20 I-frames. Using BCH to encode data that will reside within H.264 prior to encoding to improve robustness against quantization attacks, an essential feature when watermarking multimedia against piracy.

Nguyen *et al.* [106] propose an algorithm for hiding data in H.264 multimedia format. The proposed algorithm hides data within the Quantized Discrete Transform (QDCT) coefficients of I-frames. The proposed algorithm uses DCT coefficients for data embedding through LSB and intra-frame prediction modes to determine whether to extract an encoded payload or not. Kapotas *et al.* [107] propose exploiting the inter prediction phase of the H.264 video encoder to embed data. The motion estimation process found in the inter prediction encoding phase treats frames in chunks of blocks to evaluate the motion compensation value. The frames are chunked according to 7 different block sizes where the proposed steganographic algorithm logically ties sizes to binary codes. The proposed steganographic scheme will force an H.264 encoder to choose different block sizes to represent a specific hidden message. Chen *et al.* [109] propose using LSB on the quantized DCT coefficients to embed data in a high capacity and in a reversible manner. The proposed scheme hides data in the high frequency coefficients through histogram shifting to achieve reversibility. Using affine rules, the coefficient values correspond to binary sequences representing the embedded data where the receiver will use said rules to

extract the hidden data and recover the original stream. Laura *et al.* [110] propose a real-time steganographic technique to embed data in the luminance coefficients of the multimedia I-frames. The continuous motion prediction process performed through the comparison of I-frame blocks promotes a large steganographic embedding capacity since frames are made of a large number of block chunks. The block chunks are used in the intra-predicted motion process to create residual blocks. The DCT coefficient values of the evaluated residual blocks are modified with the hidden message where an H.264 stream is then created.

A notable service that uses steganography is found in [111] where data hiding is used in managing multimedia session access. Trappe *et al.* [111] propose a novel media-dependent key management and distribution scheme for multimedia multicast scenarios. This approach is employed at the application layer and uses data embedding schemes to hide re-keying messages within multimedia content. The advantage and focus of this method is the increased difficulty for malicious users to gather information regarding the membership system and its behavior. Since the re-keying messages are embedded within the multimedia data, the amount of messages transmitted during the broadcast session are reduced. There are three events that can trigger the use of re-keying messages which are key refresh events for security enhancement and key updates subsequent to users providing their desire to leaving or joining the multimedia session. Embedding the re-keying messages will reduce bandwidth consumption in heterogeneous applications only if the re-keying messages are invoked which might not occur as frequently as desired. In online multiplayer games, conferences, meetings, and P2P video streaming applications, re-keying messages are not necessarily invoked as frequently due to to the commitment of the user to the application.

Zhao *et al.* [112] propose a method to hide data within one of the frames of a popular video compression standard known as MPEG-2. The MPEG-2 standard uses three different types of frames in a repeating sequence, such as $I_1B_2B_3P_4B_5B_6P_7B_8B_9P_{10}B_{11}B_{12}$, to construct many Group of Pictures (GOP). The proposed method exploits the B frames of GOPs by embedding data within their multimedia packets. However, the weakness of proposed approach lies within

the B frames dependency on the I and P frames for the decoding process. If the receiver does not have the I or P frames, the B frames are useless. Decoding and presentation timestamps, among other information, are required by the receiver for display synchronization and proper frame decoding in addition to the I and P frames. The preferred protocol to transport any MPEG-2 related data would be RTP over UDP/IP thereby limiting the proposed method to embedding data at the application layer making it susceptible to packet loss. In fact, an error in an I or P frame will have an effect on the remaining B and P frames in the GOP. The proposed method will negatively affect the quality of the video transmission due to how it behaves. It operates by replacing the payload of RTP packets of B frames with a steganographic payload and by transmitting the packet on time or after an intentional delay. The receiver will then consider these packets as lost packets and are forcefully dropped. Zhao *et al.* further expanded the proposed scheme by hiding data within the RTP header as well as shown in [113]. Concealing data using the proposed method will limit the amount of information that can be transported due to the decrease in received video quality as the amount of hidden data increases. Additionally, the proposed method will not reduce network congestion nor improve the overall quality of the transmission scenario. In [114] the authors introduce a new covert channel to bypass communication censorship by encoding information as player actions in a real-time strategy (RTS) video games. The discrete proposed scheme only operates at the application layer and is designed to carry a small amount of data at the average rate of 200 bits per second. To transport RTS game player actions, UDP/IP is used thereby reliability must be ensured at the application layer. However, using UDP/IP and application layer feedback creates a covert channel that susceptible to loss and is bandwidth limited. Similar to [114], Rook is proposed in [115] as a method to bypass censorship using an online game as a covert channel. The proposed scheme is a low bandwidth covert communication platform that is capable of achieving a throughput of approximately 20 bits per second.

2.3 Basic Concepts of Blockchain-based Networks

Service decentralization is an essential component to achieving scalability as the demand on an application grows, such as the demand on multimedia applications. The blockchain is a popular solution for application decentralization due to its inherent security focused properties. A blockchain is a decentralized and secure technology that enables the scalability of numerous applications through continuous record maintenance, global consensus, and tamper-proof design. Record maintenance is an essential component of a blockchain-based system with a high task completion delay due to the scrutiny of a blockchain system's security [116]. Unfortunately, existing record maintenance solutions are primarily geared towards digital currency systems or systems with a large computational capacity.

The blockchain is a special type of database consisting of grouped units of transaction entries termed as as blocks [117]. Each block and its entries are cryptographically linked to the preceding block to achieve tamper-proof status. The tamper-proof status as well as the cryptographic dependency between blocks is an iterative process starting at the very first block of a blockchain-based system. While using blockchains, transactions are permanently recorded to the ledger unless the majority of participating record maintainers agree to a modification at the cost of reprocessing the necessary list of blocks. The sheer amount of computational power needed to reprocess existing blocks effectively promotes blockchains as a decentralized system with a high Byzantine Fault Tolerance (BZT). The underlying assumption is that it is harder to control the majority of the computing power in a blockchain network through layers of computationally expensive cryptographic puzzles. A cryptographic puzzle is solved and added to the existing layers of computations at the formation of new blocks. New blocks tend to be limited in space, and therefore are capable of framing a finite number of transactions. Each block, its transactions, and the hash value of the preceding block are computationally hashed until a nonce is found to meet the blockchain network's target value [117]. Since the essence of the security is the computational complexity, the process is time-consuming. Record maintainers, formally known as miners, are capable of purchasing additional computing power

to solve the cryptographic puzzles at a higher throughput, however, blockchain networks are designed to increase the puzzle difficulty accordingly to thwart malicious users. The limited block sizes and computationally expensive cryptographic puzzles will result in large mining pools for miners to select transactions from. In blockchain-based digital currency systems, such as Bitcoin, miners are capable of enforcing fees to enable transactions in securing a spot in a soon-to-be published block, even if said transactions arrived in the mining pool at a later time. In essence, transactions are capable of purchasing their processing guarantee and priority from miners. Existing mining disciplines are fee-based and are primarily geared towards digital currencies.

A blockchain network operates purely over Peer-to-Peer (P2P) protocols thereby achieving decentralization in the communication plane as illustrated in Fig. 2.12. In a blockchain-based system, nodes do not explicitly communicate with each other but rather communicate through the ledger by sending transactions to miners. A source node will send a transaction to a set of miners maintaining the ledger where the miners will validate the received transactions. A miner will initially select transactions from its mining pool to form a block of transactions where the block is then validated through using a cryptographic puzzle. Once the block puzzle is solved, the new block is then shared between nodes and is accessible by any node participating in the network, whether a miner or a simple node. A new node joining a blockchain network is capable of requesting an entire copy of the ledger from its peers where the ledger will contain all transactions. All participating nodes are easily capable of verifying the author of a transactions and the sequence of events recorded in the ledger. To verify a transaction, the hash value of a transaction, h_T, must be found and cryptographically evaluated with the author's public key [117].

The hash value is a mathematically computed string of bits that are mapped to the input data where ideally no two inputs will have the same string of bits as an output. The generated string of bits are identical in size regardless of the input length, a feature much needed when dealing

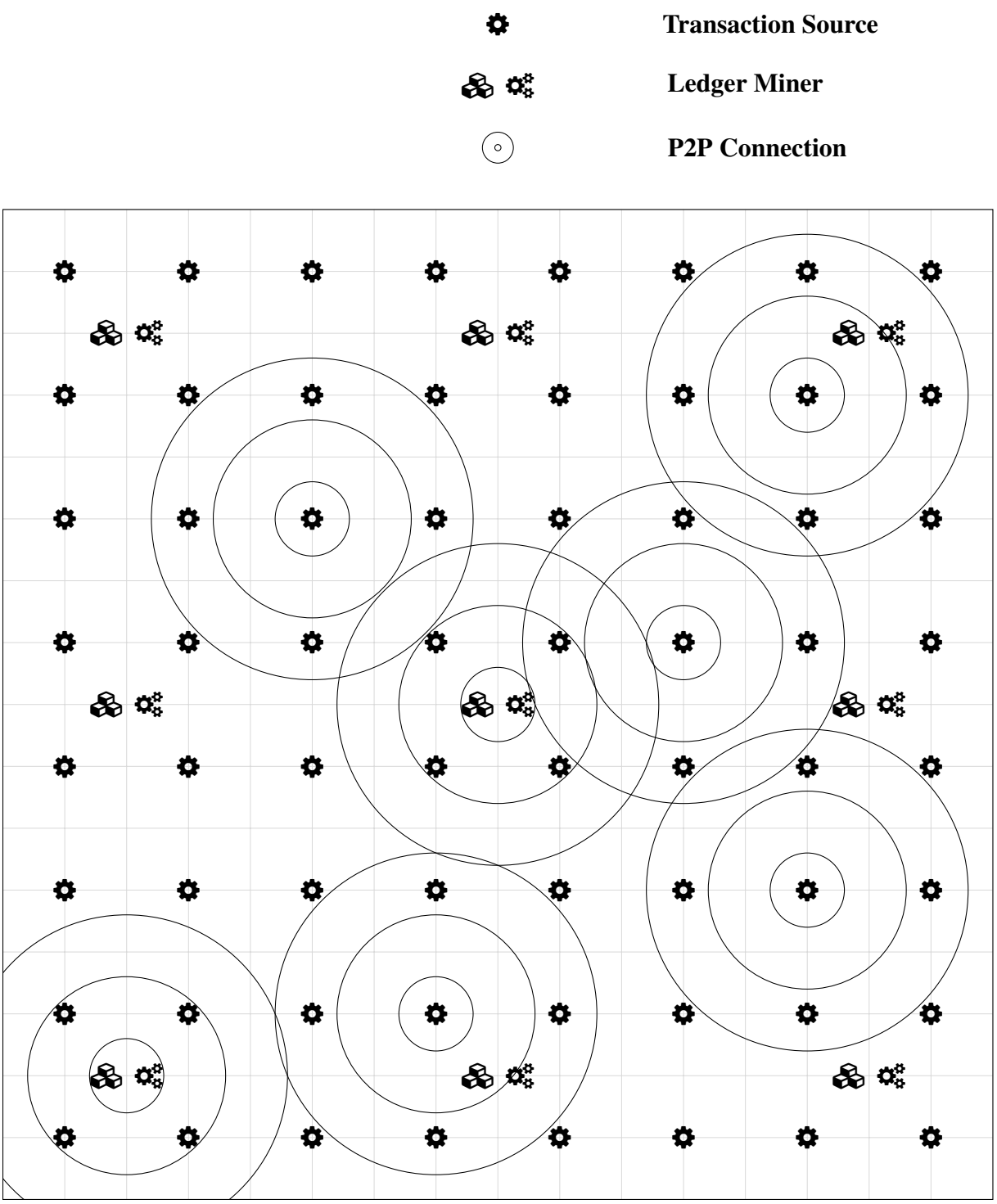


Figure 2.12: Nodes communicate with each other using P2P networks to enable blockchain-based applications and decentralized services.

with very large transactions. The hashing function plays an important role in block publishing as well. Miners will run blocks through hashing functions in addition to a nonce value until a

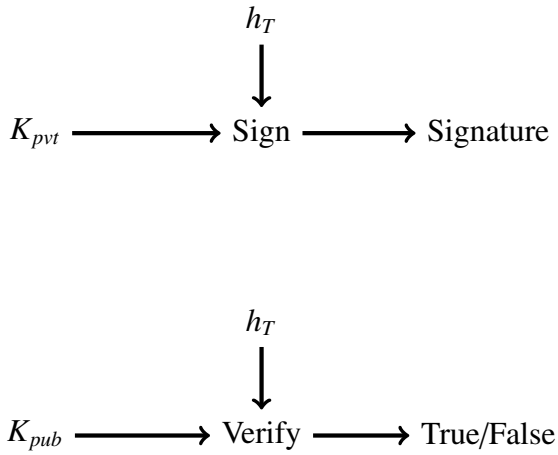


Figure 2.13: Transactions are signed using the private key and verified using the public key counterpart.

nonce is found that satisfies the current blockchain network security measures. The nonce value is a random string of numbers that is only found through iterative computationally expensive brute force mechanisms. Once a miner finds a nonce value for a block it's ready to publish, a miner will broadcast the block and the nonce to all participating nodes. All participating nodes are then capable of verifying the new block by hashing the block with the given nonce value as well as the hash value of the preceding block as illustrated in Fig. 2.14. Although the process of finding the nonce is time and resource consuming, the verification of the nonce is rather quick. Verifying the value of the nonce can be done at any block in the blockchain system and not only the most recent published blocks [117].

A blockchain system is further capable of utilizing digital smart contracts where legitimate transactions are executed without intervention from any source [117]. All mining nodes performing ledger maintenance will have a copy of all smart contracts where they are executed accordingly. The aforementioned properties prompted the adoption of blockchains by numerous applications and services that suffered from scalability issues. The inherent properties of a blockchain-system, such as a high level of fault tolerance and traceability, enabled secure large scale decentralized deployments of numerous applications and services. The applications that benefited from blockchains include: IoT-based, multimedia solutions, health care systems, energy trading, and social media platforms, among others. All of the applications and services

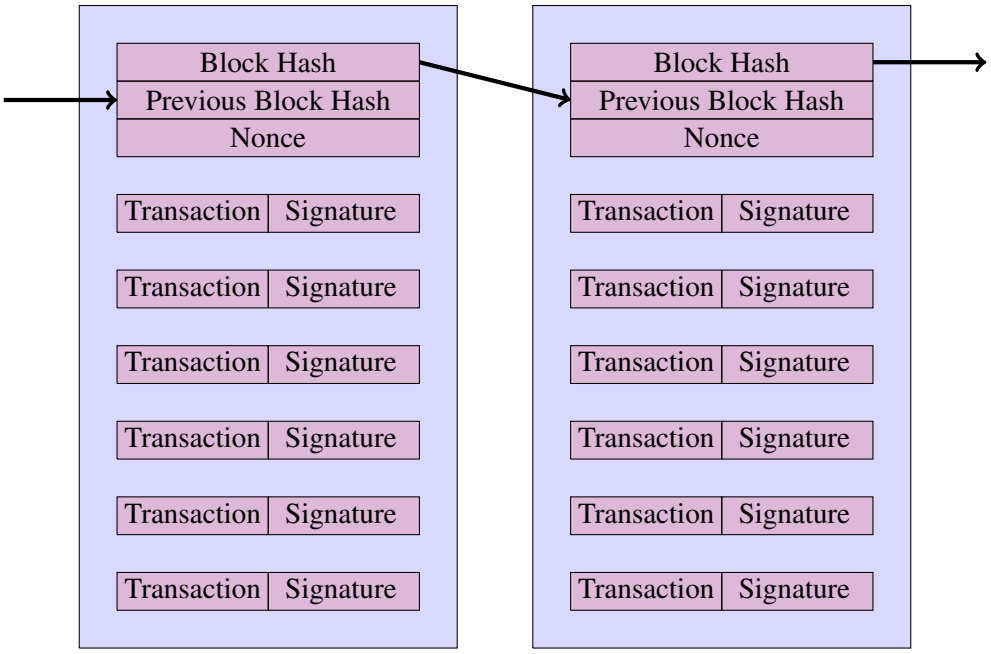


Figure 2.14: Blocks of transactions appended to a ledger by miners and shared across a blockchain-based network.

benefiting from blockchain-systems will face the challenge of building blocks since it is time consuming and resource consuming. The mining pools of maintenance nodes will definitely have an overflow where transactions must wait for a long period of time in said pool. Miners enabled transactions to minimize mining queue waiting time through a tiered processing structure where transactions pay higher fees for faster expedited processing. Existing mining disciplines are inefficient for industrial applications and therefore feeless solutions are needed for tiered mining. The alternative solution is to design miners operating in industrial settings to mine transactions in order of arrival.

2.3.1 Decentralized Multimedia Content Access and Management Solutions

Multimedia services are extremely popular, require the harmonious collaboration of numerous systems, and services large audiences simultaneously. The grand size of multimedia applications today promote the use of blockchains to securely enable different components

of the service. Accessing multimedia content and combating piracy is an ongoing challenge in the multimedia world where blockchains stepped in as an important scalable solution for protecting digital rights. Digital rights management occurs by allowing legitimate users in accessing content and by tracking down pirated content. Decentralized consensus of access, identity attestation, and identity management is a solution developed by Yasin and Liu [28]. In [28], Yasin and Liu propose using blockchain with smart contracts to secure users' identities and provide users intent, whether malicious or otherwise. The proposed blockchain solution will provide insights of users degree of maliciousness through analysis and record keeping of institutional and social media data. Zyskind *et al.* [29] propose using a decentralized personal data management system that uses distributed hash tables off the blockchain-chain to give users control of their data. Using the proposed scheme, users are capable of allowing services access to their personal data through the blockchain. The application will then use the data to verify the user through the recorded information.

Tracking and combating sources of multimedia piracy is an ongoing challenge due to the loss of revenue for the content creators and rights holders. The grand size of distribution and popularity of multimedia will require a decentralized, secure and trustworthy solution for scalable digital rights protection. Numerous digital rights protection solutions [30–35] are found in the literature promoting the use of blockchain due to their scalability and tamper resistance. Xu *et al.* [30] proposed a blockchain-based scheme for tracking down pirated content and protecting intellectual property communicated digitally. Ma *et al.* [31] propose a blockchain-based scheme where two ledgers are used separately to store information on right protected media. In [31], one of the ledgers will be used to store unencrypted data while the other will store ciphered data. The blockchain-based solution will provide traceability through maintaining an access record to all Digital Rights Management (DRM) content in addition to media watermarking for user identification. Ma *et al.* [32] propose using a blockchain-based system in a master-slave setup to provide DRM protection and user authentication. The proposed scheme sets powerful ledger maintaining nodes as masters and less powerful devices responsible for

generating transactions as slaves. The master nodes will then distribute transaction verification and block nonce evaluation to slave nodes. Zhao *et al.* [33] propose using blockchains alongside steganographic watermarking to protect multimedia digital rights. Zhao *et al.* designed a media protection scheme are users need to request data extraction mechanisms from data hiding servers in-order to enable multimedia playback. Initially, all multimedia files must be signed and baked into the ledger prior to processing by the data hiding server and viewing by a user. Bhowmik and Feng [34] propose to use blockchains to hold a complete log of transactions detailing any edits or modifications completed on all multimedia segments. All multimedia will be stegonagraphically watermarked with hash-based identification string used to retrieve the modification history from the ledger. Furthermore, the hidden watermark will contain details on the edited parts of the multimedia file in order to retrieve the original media descriptions. Bhowmik and Feng [34] successfully hid a watermark that is 8,220 bytes using a wavelet transform-based method in a multimedia file where the details stored in a blockchain are used to restore the original image and identify who performed the editing. Ma *et al.* [35] propose the Blockchain as a Service (BaaS) model to simplify utilizing blockchains in different services, such as digital rights protection in multimedia. Ma *et al.* show that using blockchain as an the infrastructure of a tamper proof DRM service, as well as a digital cryptocurrency to be used in purchasing multimedia is possible due to the customizability of blockchains.

2.4 Summary

Although the Internet is not designed for exchanging multimedia, it is the most popular multimedia communication medium within consumer markets. The Internet's infrastructure is not suitable for multimedia exchanges since its a delay and loss sensitive application operating over non-deterministic routes with best effort guarantees. To overcome the lacking infrastructure while exploiting existing hardware, adaptive multimedia streaming solutions based on HTTP/TCP protocol combination gained popularity. Albeit adaptive multimedia solutions are

used, the inherent mismatch between multimedia services and the underlying communication infrastructure as well as the best effort protocols results in large amounts of generated overhead, intermediate node buffering, network congestion, and unnecessary retransmissions. Furthermore, as multimedia fidelity and accompanying services become increasingly more prominent, additional signalling overhead will be generated in the form of request-response interactions. The overhead generated in a multimedia session occurs at every layer of the TCP/IP abstraction and as a result numerous works of literature propose overhead reductions methods to increase the efficiency of data exchanges. Additionally, as multimedia services become increasingly more complex and diverse in nature, signalling overhead mitigation mechanisms are needed to reduce service provider costs. Numerous large scale multimedia deployments will require services to protect digital content rights as well as manage access to licensed content. Blockchain technologies enable secure, immutable, and decentralized scalability of services and are thus ideal for digital rights management and other large scale industrial solutions. Unfortunately, existing blockchain maintenance solutions are geared towards digital currency services, and therefore new ledger maintenance solutions need to be investigated to address the gap.

Chapter 3

Tapering Multimedia Bandwidth Utilization through Steganographic-based Signalling Overhead Mitigation

Though they are famous for generating a sizeable amount of signalling overhead, adaptive streaming technologies are deployed on a wide scale. Adaptive streaming technologies offload session configuration and control to the receiving client to enable enhanced network response and scalability [5]. Reducing costs for streaming sessions between source-destination pairs promotes the exchange of request-response interactions at a higher efficiency, and thus minimizing redundancies. This chapter proposes a novel cross-layer steganographic-based overhead reduction mechanism for real-time and on-demand multimedia streams. In the proposed scheme, nodes will embed signalling information within multimedia payloads to reduce end-to-end session costs. The facilitated deployments of HAS-based streaming technologies enabled rapid adoption, therefore signalling overhead reduction mechanisms play an important role in extending multimedia sessions. When a client requests a segment from a server, the client will send an HTTP-based request where the server will respond with the desired segment. As the client requests additional features and services, such as increased multimedia quality, additional HTTP-based requests are generated. All of the stream data, whether requests or segments, is exchanged over TCP and is subject to retransmission when necessary. The nodes performing the embedding process will consider the aforementioned signalling data for embedding within multimedia payloads. Nodes will initially determine whether the session

conditions are appropriate for utilizing the encoding scheme through probing network conditions to reduce retransmissions and induced visual artifacts. The client node is aiming to determine the total observed reduction in resource consumption and mean queue waiting time by performing the proposed process. If the conditions of performing the embedding process surpass the desired thresholds, signalling overhead messages are then encoded within multimedia payloads. The results obtained through simulations, as well as the proof-of-concept implementation, validate the proposed scheme's ability in decreasing resource consumption by signalling overhead flows.

3.1 Introduction

Content distribution networks quickly recognized the lack of initiative from Internet service providers to update the Internet's infrastructure, and as a result HAS propelled to be the preferred mechanism for delivering multimedia [69, 70]. Adaptive multimedia streaming technologies, particularly HAS, are responsible for generating large amounts of signalling overhead at every layer of the TCP/IP stack. The generated signalling overhead is primarily due to the nature of the employed protocols that are designed to cope with the Internet's erroneous paths. The erroneous properties of the Internet's underlying infrastructure prevents service providers from guaranteeing throughput, dedicated routes, or bandwidth allocations. Although the employed protocols are designed to cope with the Internet's behavior, they are not designed for exchanging multimedia but for indiscriminately exchanging bytes of data. The employed family of protocols, particularly the IP suite, have a one-size-fits-all approach to delivering information from all services. The set of IP protocols commonly dubbed as the TCP/IP stack are designed to tolerate the Internet's unpredictable behavior through retransmissions or moving past undelivered data. Retransmissions are important for HAS-based streaming to ensure the delivery of signalling messages and MPD files, however, they unnecessary when dealing with multimedia payloads. The inherent inefficiencies in combination with the diverse nature of

multimedia traffic produces large amounts of overhead messages. The mismatch between multimedia applications and the infrastructure of the Internet resulted in network-based compensation through increased generated signalling overhead, intermediate node buffering, retransmissions, and as a result, network congestion. The mismatch is further exacerbated through the Internet's non-deterministic behavior, unpredictable routing paths, service provider agreements, and best-effort guarantees provided by the protocols employed. Essentially, the economical design of the ICT infrastructures, particularly the Internet, created a challenge and a barrier for real-time bandwidth-sharing applications [93, 94]. The Internet's economical design, protocol stack, and current hardware will result in the degradation of the QoS perceived by users of multimedia applications [2].

Reducing exchanged signalling overhead received substantial attention in the literature, more precisely in the areas of HAS-based streaming and TCP-based sessions. Signalling overhead reduction mechanisms are present at the application, transport, and network layers of the TCP/IP stack [7–27]. The existing solutions suggest modifying a server to change from traditional pull-based methods to push-based [8–15] to reduce the number of requests sent by a client. However, reducing the number of received requests will negatively affect the transmission scenario in congested networks and the stream's congestion window growth rate. By exchanging fewer segment requests, the transmission session is slower to react to network conditions and subject to abuse by a requesting node. Fewer segment requests will result in a TCP congestion window growing at a slower pace, especially during the slow-start phase. Unlike pull-based sessions where clients continuously request additional media segments, push-based schemes will prompt source nodes to utilize the HTTP 2.0 feature of responding with k-segments per request received from a client. Manipulating the configurations of transport layer protocols is another option [16–19] where TCP ACK messages are delayed to confirm a group of segments simultaneously or grow the TCP congestion window according to the number of bytes received. Manipulating the configuration will require source-destination pairs to exchange details regarding the slow-start phase continuously. Finally, alternatives to the TCP

protocol are proposed [25, 26] with a smaller header size and similar functionality, however, the protocols are not as mature as TCP.

Overcoming the inadequateness of the existing solutions is possible by employing real-time steganography for embedding signalling overhead within multimedia payloads. In this chapter, an end-to-end steganographic approach is developed to improve the performance of real-time streams by reducing bandwidth costs and queue waiting times. Unlike the aforesaid overhead reduction schemes, the proposed encoding process does not increase the burden on the client side while proactively adapting to network conditions, thereby maintaining a desired QoS. The steganographic solution is a cross-layer approach where signalling messages from all of the TCP/IP stack layers are subject to embedding within the application layer multimedia payload. The embedding process is performed on a packet-by-packet basis to avoid the effects of packet loss where receiving nodes are still capable of decoding messages independently regardless of received packets. Payload packets will benefit from the embedding process, more precisely packets at the rear of a transmission queue, will benefit by experiencing reduced mean waiting times within the said queue. The costs of transmitting a signalling message are avoided at the generating node as well as the intermediate nodes where the destination node must perform the decoding process. Finally, while using the proposed embedding scheme, the proactive and responsive behavior of HAS is maintained. The embedding process is completed using LSB [95] steganographic method and is applied adaptively in real-time through the continuous exploitation of internal TCP components as well as collected packets' statistics. By probing the RTT and Retransmission Timeout (RTO) values placed by the TCP congestion window, each stream is given a connection score to dynamically determine if a stream is fit for the embedding process.

The goal of the proposed steganographic cross-layer signalling approach is to reduce multimedia session overhead while maintaining QoS requirements. The proposed solution uses a cross-layer steganographic-based approach where signalling data from any layer of the TCP/IP stack is embedded in the application layer payload as shown in Fig. 3.1. In this manner, key

<table>
<tr><th>TCP/IP Layer</th><th>Current</th><th>Proposed</th></tr>
<tr><td>Application</td><td>Payload Steganography</td><td rowspan="4">Cross-layer Steganography</td></tr>
<tr><td>Transport</td><td rowspan="3">Lower Layer Steganography</td></tr>
<tr><td>Network</td></tr>
<tr><td>Data-link</td></tr>
<tr><td>Physical</td><td colspan="2">Physical Layer Steganography</td></tr>
</table>

Figure 3.1: The proposed scheme promotes a cross-layer steganographic solution where data from different layers is embedded in payloads from other layers.

limitations are avoided from network layer steganography such as the limited embedding capacity or consequences of header stripping. The inherent design and need of signalling in using HAS and, by extension, the TCP/IP stack produces information frequently. For example, if a service provider of a cloud conferencing application, P2P content delivery system, or STB content sharing, decides to use HAS over TCP/IP, many acknowledgement packets, GET requests, and other signalling messages are exchanged between the source-destination pairs to enable the transmission of new payload messages. The proposed mechanism is designed to improve stream performance by eliminating a significant number of the exchanged signalling messages. The encoding process is done in a logical manner where the TCP congestion window is probed to determine streaming conditions, whether they are suitable for encoding signalling data or otherwise.

The rest of the chapter is organized as follows: In Section 3.2, the system model considered in this chapter is introduced. In Section 3.3 the details of the proposed steganographic scheme for embedding signalling overhead within multimedia payloads, multi-application identification scheme, estimated throughput gains and bandwidth utilization gains, as well as handling HTTPS streams, are presented. In Section 3.4, simulation and implementation results are presented and analyzed. Finally, in Section 3.5, the chapter summary is given.

3.2 System Model Description, Assumptions, and Problem Formulation

Assume an N node scenario where the nodes are exchanging multimedia data and other services amongst each other using HAS-based streaming as well as TCP streams over the Internet as illustrated in Fig. 3.2. Each node is requesting d segments using signalling messages that vary in size and will therefore require a different amount of resources to complete the embedding. The total number of signalling messages k, in terms of ACK and GET messages, generated per d segments is $k \geq d$ due to the number of ACKs and retransmissions. The total amount of payload segments generated throughout the session is $d \in D$ where the total number of signalling messages $K \geq D$. To avoid playback freezing, the total playback time of the d requested segments is larger than t seconds of multimedia descriptions. Each multimedia packet is at least 1,460 bytes large and each non-payload packet is at least 70 bytes large. Using the existing data exchange models, session information is transferred as two separate streams, data and signalling, as shown in Fig. 3.3a. However, clients using the proposed steganographic approach to delivering signalling messages will still generate said messages where they are embedded within the multimedia payload on-the-fly. The embedding process will result in a unified flow stream as shown in Fig. 3.3b where data and signalling information are multiplexed on a packet-by-packet basis. A summary of frequently used notations is found in Table 3.1.

As signalling messages are embedded within payloads, there will be fewer signalling messages occupying resources within the network. However, due to the greedy behavior of TCP, an increased number of payload messages will be exchanged, triggering an increased number of signalling messages generated at layers. Therefore, the embedding capacity of the system must be increasing in a similar manner to the AIMD algorithm where the number of d segments increases and as a result k behaves accordingly. A cost formulation is designed to select non-payload messages destined for embedding to minimize the amount of exchanged overhead

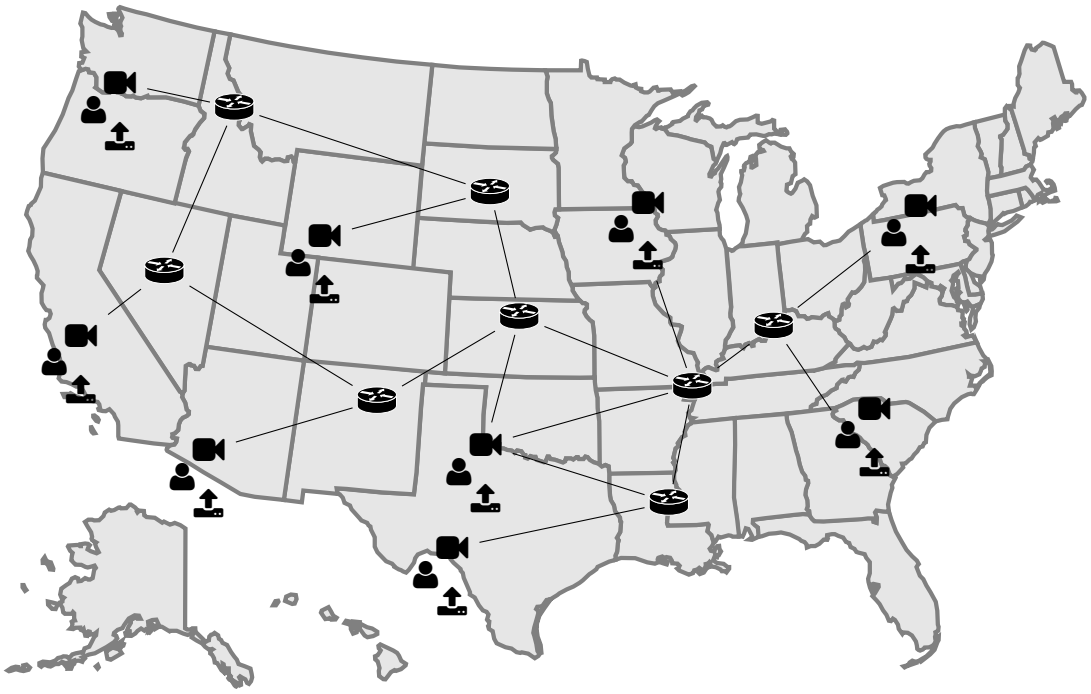


Figure 3.2: N users communicating simultaneously over the Internet to stream multimedia and complete file exchanges.

between source-destination pairs. Consider $Q \leq N$ users connected through a web-video conferencing application that uses HAS over TCP/IP. Each node $n \in N = \{1, 2, .., Q\}$ represents a different user with a total of F frames to send. Each TCP connection has a maximum window size of $W_{\max,n}$, a round-trip time of RTT_n, and a maximum achievable throughput of $\frac{W_{\max,n}}{RTT_n}$ [118]. Each participating node will minimize the amount of data sent using (3.2a).

The total amount of data a node has to send at any given instant to all receiving nodes in N is

$$T = D + I, \tag{3.1}$$

where D and I are the complete sets of payload and non-payload messages, respectively. At any given instant, the set of signalling messages being sent from a node performing the embedding process to destination n varies in size $I_k^n \in I^n$ and $k \in K = \{1, 2, ...K\}$ where K is the signalling message index and $I^n \subset I$. As the number of signalling messages increases, the number of segments exchanged the next round will increase accordingly to mimic TCP behavior. Therefore T_i is the total data exchanged at t_1 and $T_i < T_{i+1}$ due to the ever-increasing payload

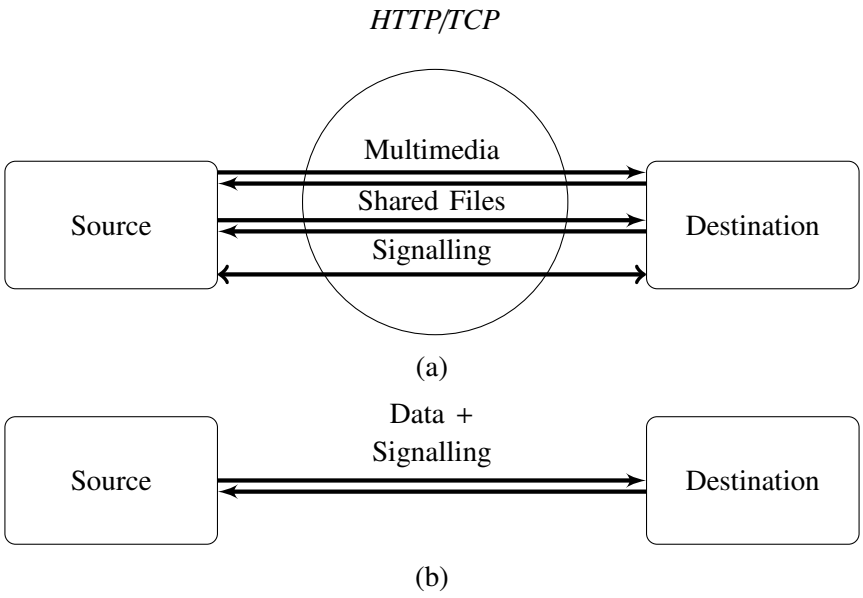


Figure 3.3: Two streaming approaches: (a) Traditional streaming involves separate signalling and data streams. (b) The proposed scheme will combine the signalling and data streams into a single stream using steganography.

and request asymptotic relationship. However to replicate congested networks, the value of T is decreased at random time intervals and at pre-determined thresholds to set an upper-bound on the maximum throughput.

The purpose of the non-linear objective function (3.2a) is to maximize the number of signalling messages $I_k^n \in I$ chosen for the embedding process consequently minimizing the value of T since there are fewer signalling messages to send. The formulation aims to ultimately decrease the amount of data sent from the source node while maintaining a low visual error rate and is formulated as

$$\min_{\{x_k^n\},\{m_n\}} (D + (I - \sum_{n\in N}\sum_{k\in K} I_k^n x_k^n m_n)) \tag{3.2a}$$

s.t.

$$\frac{\sum_{k\in K} Er_k^n x_k^n}{\sum_{k\in K} x_k^n} \leq \delta^n \quad \forall n \in N \tag{3.2b}$$

Table 3.1: Summary of the most significant notation used in this section.

Notation	Definition
Q	Total number of nodes/users
n	Node number n
K	Total number of signalling messages per node
k	Current signalling message k
D	Total data waiting to be sent to all nodes
I	Total signalling information waiting to be sent to all nodes
I_k^n	Signalling message k being sent to node n
x_k^n	Binary decision variable for I_k^n and node n
m_n	Network condition binary decision variable for node n
Er_k^n	Error generated due to embedding I_k^n
δ^n	Maximum allowed error
C	Available resources for performing the embedding process
e_k^n	Resources needed to perform embedding process for I_k^n
S_n	Connection score for node n
L	Minimum allowed connection score
P	Frame height
W	Frame width

$$\sum_{n\in N}\sum_{k\in K} e_k^n x_k^n \leq C \tag{3.2c}$$

$$\frac{\sum_{n\in N} S_n m_n}{\sum_{n\in N} m_n} \geq L \tag{3.2d}$$

$$m_n \in \{0,1\} \qquad \forall n \in N \tag{3.2e}$$

$$x_k^n \in \{0,1\} \qquad \forall n \in N,\ \forall k \in K. \tag{3.2f}$$

To compute the Er_k^n of a frame or for a portion of a frame, (3.3) is used. The height and width of the frame are of size P and W pixels, where the error is calculated and averaged using

all samples, $p \in P$ and $w \in W$, in the reference frames. In (3.3), $f(.)$ refers to the original sample pixel value prior to the embedding process while $f'(.)$ refers to the new sample pixel value subsequent to the embedding process.

$$Er_k^n = \frac{1}{PW} \sum_{p \in P} \sum_{w \in W} (f(p,w) - f'(p,w))^2. \tag{3.3}$$

Signalling data may be embedded in one or all of the channels of a frame's pixels, depending on the employed data-hiding scheme. Due to the hiding process, the original frame data is modified and therefore an error, Er_k^n, is induced. The embedding process completed for destination node n on the k^{th} signalling message, I_k^n, generates a visual artifact, Er_k^n, subsequent to the encoding process due to changing frame information. The total error generated must be maintained under a certain limit, δ^n, to prevent the user from seeing visual artifacts as shown in constraint (3.2b). To minimize wasted embedding processes, only destinations with reliable connection scores, S_n, are considered to satisfy perceived path conditions. Each score, S_n, is based on the TCP congestion window state and the observed exchanged messages statistics between a source node and destination node n. The TCP congestion window is chosen as one of the means of evaluating S_n due to said window's behavior reflecting the state of the propagation path. If there are multiple streams between a source-destination pair, then S_n is based on all streams between said pair. The connection score is continuously updated since the TCP is designed to update the RTO value as time progresses. The congestion window dynamically changes RTO values due to the non-deterministic behavior of the delivery paths. By using the values computed at the transport layer by the TCP protocol, each receiving node is assigned a connection score as

$$S = \frac{CR}{BR}, \tag{3.4}$$

where BR is the optimal observed RTO value throughout the lifetime of the transmission and CR is the most recent estimate of the RTO value.

The purpose of S_n is to combine multiple stream statistics to characterize the overall

path conditions between source-destination pairs to reduce the number of retransmissions and wasted embedding processes. If there are various sets of multiple streams heading towards a number of destinations, then the average score of the chosen stream sets must be greater than or equal to L to ensure proper delivery and minimize the effects of packet-loss as shown in constraint (3.2d). To prevent the source node from overwhelming itself, the capacity of embedding processes, C, is limited to said node's capabilities. Data embedding consumes additional system resources where each embedding process will consume e_k^n resources and therefore it must be maintained under a certain capacity C as shown in constraint (3.2c). Signalling messages with a larger size will have a larger embedding effort, e_k^n, and a result, will consume additional resources when compared to smaller signalling messages. Finally, binary decision variables responsible for declaring whether the embedding process will take place or otherwise. The decision variables m_n and x_k^n are binary variables as shown in statements (3.2e) and (3.2f), respectively, and are subject to the optimization process. The x_k^n decision variable is used to determine whether a packet is considered for embedding and the m_n decision variable is used to identify whether a certain stream meets the score requirement constraint placed by L.

3.3 Proposed Real-time Cost Savings Evaluation

The proposed objective function (3.2a) is non-linear and consumes a large amount of time to find a solution. Therefore a linear objective function is proposed in this section subsequent to analyzing the costs of data delivery in the time domain. The costs of delivering data between a source-destination pair can be measured in terms of end-to-end response time and bandwidth. The one-way delay of an application is determined by the total delays a packet experiences on the path towards the destination. The one-way delay is

$$D = d_t + d_{pg} + d_{pc} + d_q, \tag{3.5}$$

where d_t is the transmission delay or the time needed to send a packet of size L at rate R, d_{pg} is the propagation delay that is determined by the signal propagation speed and by the length of the links used along the path, d_{pc} is the processing time needed for the packet to determine the output link and other information, and d_q is the time a packet waits for transmission in the output queue of a link. The delay value D is therefore reduced to a much smaller value by using LSB to embed signalling information within multimedia frames. Subsequent to using LSB, the new delay value for the encoded data at the sourced-destination pair is,

$$D_e = d^e_{pc} + d^e_q, \tag{3.6}$$

where d^e_{pc} is the new processing delay generated by finding and embedding the signalling data in a suitable media frame. The d^e_q term is the new queuing delay experienced by the signalling data. The frame used for embedding the data will experience a new delay value at the source and receiving nodes as well. The new frame delay value at the source node is,

$$D_f = d_t + d_{pg} + d^f_{pc} + d^f_q, \tag{3.7}$$

where d^f_{pc} and d^f_q are the new processing and queuing delays experienced by the frame data, respectively. As a result, the value of D_f is $D_e < D \leq D_f$. However, at intermediate nodes, the delay of messages carrying payload information is expressed by D. Due to hiding signalling messages within other payloads, intermediate node queues will have increased buffer spaces and a number of delays of length D are avoided. Assume a queue of outgoing packets originating from a source application that is generating and sending data to N destinations. The sum of the total number of payload packets D_n and signalling packets I_n heading towards destination n is:

$$P_n = D_n + I_n. \tag{3.8}$$

As a result the total number of expected packets is

$$P_T = \sum_{n=1}^{N} P_n. \tag{3.9}$$

Each node n receiving encoded packets is assigned a sliding window of size G_n at the source node. G_n defines the maximum number of packets allowed for embedding at any given time instant, t. Therefore the total number of embedding slots available at the source node is $\sum_{n=1}^{N} G_n = G$. The source (destination) node performing the embedding (decoding) process has a limited capacity C available to perform the encoding (decoding) process and as a result $G \leq C$. Using C, a source node will be able to embed the maximum number of messages without overwhelming itself as well as the receiving nodes. A Linear Programming (LP) approach is found by relaxing the constraints of the non-linear objective function (3.2a) and using Little's theorem to transform it to a minimization of waiting time rather than signalling data. The purpose of the new objective function (3.10a) is to minimize the total waiting time at the source node or essentially maximize the amount of waiting time reduced by performing the embedding process. The objective function uses a binary decision variable, x_k^n, and aims to optimize it to determine whether each node will receive embedded data or otherwise. Furthermore, for the objective function to determine a list of nodes that will receive embedded data, each packet k in the queue of the source node heading towards destination n will receive an expected waiting time value of w_k^n. The expected waiting time, w_k^n, of a considered packet within a transmission queue is determined by summing the total waiting time of every single packet ahead of the considered packet where the waiting time is directly proportional to the processing speed of a node.

$$\max_{\{x_k^n\}} x_k^n w_k^n \tag{3.10a}$$

s.t.

$$\sum_{n \in N} x_k^n \leq C \quad \forall k \in K \tag{3.10b}$$

$$x_n \in \{0, 1\} \quad \forall n \in N. \tag{3.10c}$$

The linear function in (3.10a) will maximize the waiting time reduced while meeting constraint (3.10b). The constraint described in (3.10b) ensures that the total number of packets embedded does not exceed the allowed capacity C, assuming the same amount of resources are needed by all receiving entities. The second constraint (3.10c) describes the decision variable as a binary decision variable for all nodes and is set to high if the node n is chosen to receive embedded data. Subsequent to solving the objective function, decision variable matrix x_k^n is obtained, and is used to identify which nodes will receive embedded data as well as the total waiting time reduced from matrix w_k^n.

It is known that VBR traffic resembles an $M/G/\infty$ input process that is closely related to a Poisson Pareto Burst Process (PPBP) [155–158]. Therefore a PPBP is setup and is used to simulate a VBR application as means of estimating the expected queue length. Therefore the total number of packets generated is given by,

$$P_n = \frac{\lambda_n t r_n \delta \gamma}{\gamma - 1}, \tag{3.11}$$

where λ_n is used to describe the arrival of bursts as a Poisson process with a mean of $\frac{1}{\lambda}$, t is the period of time being considered, r_n is the amount of work generated by each burst, δ is the minimum allowed burst length and the start of the Pareto tail, and γ is the decay of the Pareto tail. Knowing that the said process has a property of self-similarity with decay, the Hurst parameter is therefore used to define γ. To maintain self-similar behavior, a finite mean, and infinite variance, γ must be in the range of $1 < \gamma < 2$. Therefore the Hurst parameter is given as,

$$H = \frac{3 - \gamma}{2}. \tag{3.12}$$

To approximate the number of packets in the queue subsequent to future embedding processes while taking into account the arrival rate of packets as well as their processing rate, P_{new} must be evaluated. Using Little's theorem, $N_q = \lambda_l W_{avg}$, where the total number of packets in

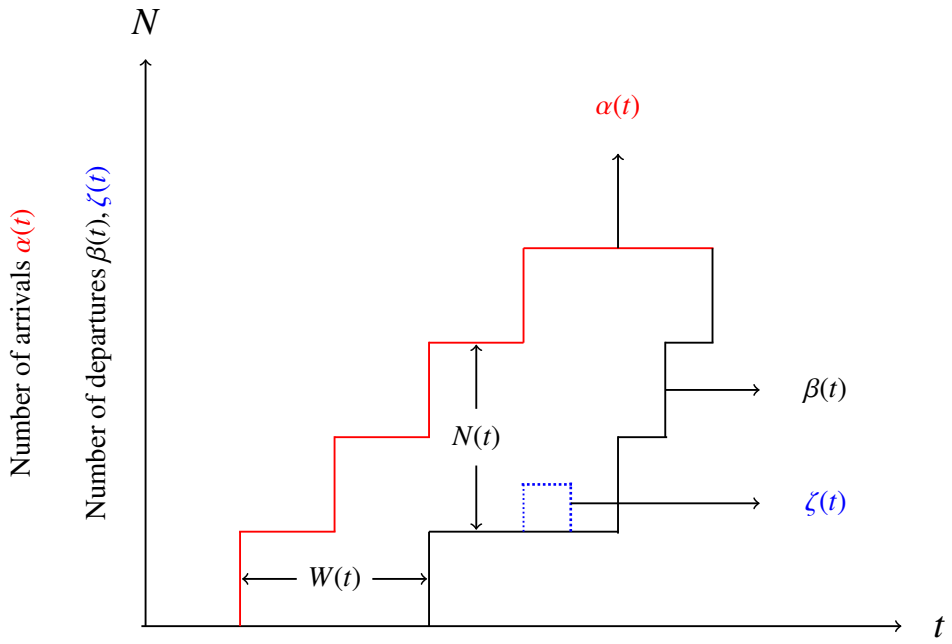


Figure 3.4: The packet arrival and departure rates are expressed by $\alpha(t)$ and $\beta(t)$, respectively, where each packet spends a delay of $W(t)$ in a given queue.

a queue, N_q, is determined using the arrival rate, λ_l, and the total time each packet spends in the queue, W_{avg}. The average arrival rate, $\lambda_l = \frac{\alpha(t)}{t}$, is the total number of packets that arrived, $\alpha(t)$, with respect to the time interval, t. Knowing the number of packets departed is $\beta(t)$, then the number of packets in the queue at time t, is $N(t) = \alpha(t) - \beta(t)$ as shown in Fig. 3.4. To determine the amount of time each packet spends in a queue, the waiting times of the packets ahead of the considered packet must be summed. The average time each packet spends in the queue within time interval t can therefore be expressed as:

$$W_{avg} = \frac{\sum_{i \in I} W_i}{\alpha(t)}, \tag{3.13}$$

where W_i is the waiting time of packet i in the queue.

If LSB embedding is used, the departure rate $\beta(t)$ will increase by $\zeta(t)$ as shown in Fig. 3.4, where a new departure rate is found as

$$\beta_{new}(t) = \beta(t) + \zeta(t). \tag{3.14}$$

The new departure rate is used to find the new waiting time. More specifically the new waiting time is

$$\sum_{i \in I_{new}} W_{new,i} = \int_{\tau_0}^{\tau_1} \alpha(\tau) d\tau - \int_{\tau_0}^{\tau_1} \beta_{new}(\tau) d\tau = \sum_{i \in I} W_i - \int_{\tau_0}^{\tau_1} \zeta(\tau) d\tau. \tag{3.15}$$

It is possible to estimate the increase in departure rate by using objective function (3.10a). The waiting time W_g is the time a packet would have waited within a queue if it is not embedded. By performing the embedding process, the waiting time W_g is skipped at the node performing the embedding process. The proposed LP objective function (3.10a) is used to determine the value of W_g, and therefore the new waiting time is

$$\sum_{i \in I} W_i - \sum_{g \in G} W_g = \sum_{i \in I_{new}} W_{new,i}. \tag{3.16}$$

If the new waiting time is divided by the old waiting time, factor j is found as

$$\frac{\sum_{i \in I_{new}} W_{new,i}}{\sum_{i \in I} W_i} = j, \tag{3.17}$$

where it is used to estimate the new number of packets in the queue with respect to the original number of packets in the queue, P_T. Therefore the new number of packets in the queue is

$$P_{new} = j \times P_T = j \times \sum_{n \in N} \frac{\lambda_n t r_n \delta \gamma}{\gamma - 1}. \tag{3.18}$$

Using the new estimated number of packets in the queue, the number of packets considered for the embedding process at time instant t is then given by

$$N_t = P_T - P_{new}. \tag{3.19}$$

The new values are then used to estimate the total saved costs in terms of packets S_p. The total amount of saved costs at the source node at a given instant from all of the streaming sessions is

$$S_p = \sum_{t \in T} (1 - P_e)(1 - P_c)N_t, \tag{3.20}$$

where P_e is the probability of an error occurring within the transmission channel and P_c is the embedding channel's error probability.

3.3.1 Systematic Multi-service Identification Mechanism

The embedding process is designed to be stream agnostic and destination-dependent to allow non-payload data of different transmissions to be embedded within different streams. The proposed scheme employs an LSB-based method for embedding data as it provides high capacity with minimal signal degradation and is usable with popular video compression formats such as H.264 [103]. When an image or sequence of images are transferred over an IP network using the TCP/IP stack, the images are segmented and acknowledged according to an agreement between the client and the server. When a network lacks the resources to process a large amount of data, the employed scheme will embed signalling data within the payload using software based methods without the need of additional hardware. The embedded data is hidden on a packet-by-packet basis to minimize the effects of packet loss. Subsequent to successful encoding processes, a signalling packet at the rear of the queue will advance to front of the queue if there is a suitable payload available for embedding. Likewise, payload messages waiting in the transmission queue will have fewer signalling messages ahead of them. Furthermore, the embedded packet will not experience any delays within intermediate nodes and as a result the destination will receive signalling messages at an earlier time-slot.

To inform a receiver of the presence of embedded data, a special code is embedded indicating whether data is present or otherwise as shown in Fig. 3.5. To facilitate the decoding process for the receiving nodes and allow multiplexing of different streams, unique codes are embedded ahead of the signalling message information. A unique code is chosen to represent TCP Acknowledgement (ACK) messages in the different phases of transmission including the

Synchronization (SYN) phase, Established Transmission (EST) phase, and ending the transmission session phase (FIN). Furthermore, clients will use HTTP requests known as GET to request additional media representations from the source node and therefore GET messages are assigned a unique identification code as well. Essentially, the codes are unique to each type of signalling message embedded and to the given data stream. The unique codes are embedded ahead of the signalling information and will power the multiplexing process of embedding various signalling messages of a number of streams simultaneously. Using this method, a receiver will be able to correctly identify the type of embedded signalling message and the stream it belongs to.

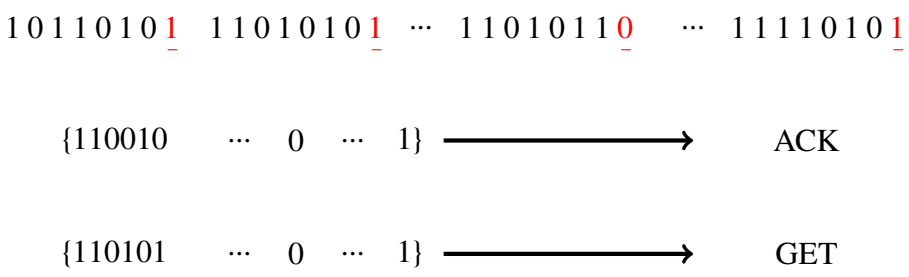


Figure 3.5: Using the embedded codes, the receiver will be able to identify the type of message embedded, the phase of transmission the message belongs to, and the stream it belongs to.

A source-destination pair communicating with each other can have a number of streams exchanging payload and signalling messages over different protocols originating from various applications and services. Furthermore, each communication protocol utilizes a predefined set of signalling messages aiming to complete tasks necessary for the applications communicating between said pairs. In the presence of multiple streams, the embedding process must be capable of handling all streams seamlessly. To facilitate the multiplexing process of embedding signalling information originating from multiple streams, a unique preamble known to all participating nodes is encoded with the subject control information. The preamble is determined by the source-destination pairs and will be used by the receiving nodes to identify the service and stream each embedded signalling message belongs to. The preamble is determined by collectively considering multiple characteristics including: the number of services

active between a source-destination pair, the number of protocols employed, and the number of signalling messages subject to the embedding process. For each source-destination pair, the aforementioned values are combined together to find their sum expressed as

$$Z = A + J + V, \tag{3.21}$$

where A is the total number of services active between a source-destination pair, J is the number of protocols used, and V is the number of signalling messages that may be embedded. The total number of bits needed to express Z is found as

$$b = \lceil \log_2 Z \rceil, \tag{3.22}$$

where b is then expressed as the nearest multiple of 8-bit words as shown in Fig. 3.6. The source node performing the embedding process will then assign a unique b-bits value code that will be used by the receiver to infer the embedded signalling message, the stream it belongs to, and the service generating it. If the total number of services, protocols, and messages subject to the embedding process is < 255, then the total number of bits needed for the embedding preamble is 8 bits. As the total value of Z increases, additional bits are needed to represent the value. The 8-bit words are embedded using the same process of encoding used to embed the signalling messages where they are extracted for decoding by the receiver as shown in Fig. 3.5. Using the decoded values, the receiver will be able to uniquely identify each embedded message. As shown in Fig. 3.7, a packet with a payload containing an embedded signalling message will have its unique identification code encoded prior to said message. The unique code is initially embedded prior to the signalling message to facilitate the decoding process. In Fig. 3.7, the bits with a single line beneath them are indicating the unique code used to identify the signalling message. The bits with two lines beneath them are indicating the signalling message information. The source-destination pairs need to exchange the unique codes in advance to facilitate the decoding process. It is assumed that the node performing the embedding

process is capable of grouping services and related protocols based on relevant destinations to enable the encoding process.

Z value	b bits
< 255	8
< 65535	16
< 4294967295	32

Figure 3.6: The total number of bits needed is found by using (3.22) subsequently formatting it to nearest multiple of 8-bit words.

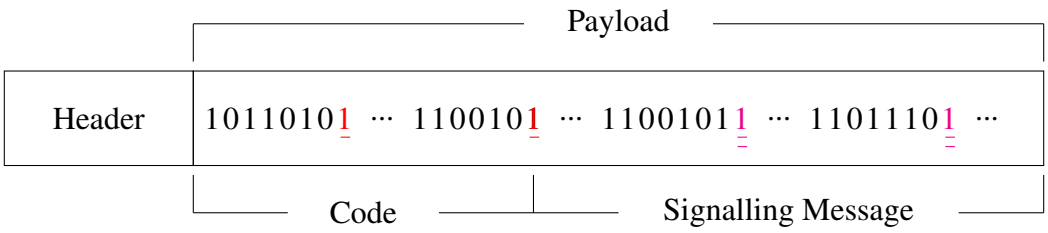


Figure 3.7: A sample packet with a payload containing an embedded signalling message also includes a unique identification code related to said message.

3.3.2 Enabling Data Embedding in HTTPS-based Flows

Incorporating secure HTTP (HTTPS) will force the utilization of TLS or Secure Socket Layer (SSL) for privacy and end-to-end encryption. The application of TLS will enforce symmetric and asymmetric key cryptography for encryption, identity authentication, and handshaking. In order to operate the proposed steganography-based scheme with TLS, an Identity Management (IM) bus is needed to run locally. The steganographic scheme requires a bi-directional traffic flow to enable data embedding. Therefore, when using TLS with bi-directional multimedia flows, the IM bus will facilitate the communication channels between said flows. The IM will enable bidirectional communication between multiple encrypted TLS-based flows through mimicking a secure message exchange router. The IM bus will essentially enable a real-time messaging system between all TLS applications through the underlying kernel as illustrated in

Fig. 3.8. The kernel abstraction that also defines the TCP/IP stack is now modified where the scheme will utilize the IM bus for inter-flow communication. The IM bus will require application layer services to provide their symmetric encryption keys to enable cross-service payload decryption and data embedding.

i\j	**a**	**b**	**c**	**...**
a				
b			X	
c	X			
...				

Figure 3.8: The IM bus will enable local communication between TLS-based flows. The IM bus is essentially exploiting the locality of the exchanged symmetric keys to access the payload information. Application *i* will utilize the keys of application *j* to decrypt application j's data and perform an embedding process.

An alternative to the IM bus is to create a unified identity that is used by the bi-directional TLS flows simultaneously. Using the unified identity shared between all HTTPS flows, the information exchanged during the TLS/SSL handshake process will be identical across multiple transmission flows. Therefore, clients contacting multiple service providers on the same destination node will need to utilize the same information across all TLS flows to create identical encryption (decryption) key. Similarly, the service providers on the destination node must mimic the client behaviour and enable the sharing of information across servers. In doing so, both, the client and source nodes, will force a TLS handshaking process that generates the same keys used for ciphering the payloads. Since the session keys are identical, an application can easily decode the payload of another application and perform the aforementioned steganographic embedding process. The final solution to overcome the unique TLS flows is to utilize a blockchain for a cross-flow identification ledger. The cross-flow identity ledger is consulted to enable privacy and end-to-end encryption through leveraging the blockchains cryptographic services. An application that desires to perform an embedding process will consult the blockchain on the necessary information needed to decrypt the flow of another application. Once the application receives the needed keys and information from the blockchain,

the application will perform the embedding process. The advantages of the IM bus is two-fold: firstly, the entire bus is contained locally within the host systems, thereby, improving performance; and secondly, it can be implemented purely in software as a kernel function similar to iptables or route. The blockchain-based solution will enable a secure, scalable, and decentralized solution, however, will induce additional communication as well as deployment costs. As a result, the IM bus is the idle candidate for enabling cross-application TLS-based flows.

3.4 Simulation and Implementation Evaluation and Analysis

The proposed embedding scheme was simulated in a parking-lot configuration using Network Simulator 3 (NS3) [153] as a platform. The topology was tested under varying number of nodes transmitting data to each other using TCP/IP protocol stack, as shown in Fig. 3.9. Each topology configuration is repeated with an induced path error of of 0%, 5%, and 10%. Only node 0 is capable of encoding information within payload data on the forward channel. Each node is generating packets at an average rate of 100 packets per second (pps), where each packet is 1460 bytes large. The packets are generated according to a Pareto distribution due to VBR's traffic shape similarity of the aforementioned distribution [155–158]. Two scenarios are used to generate the simulation results, where one allows nodes to use the proposed LSB-based technique and the other is free from the proposed scheme to be used as a benchmark. Each simulation is repeated with 10, 16, 20, 26, and 30 nodes. Each source node is generating a video with a frame rate of 16 Mbps as payload data. The optimization functions are then simulated in MATLAB and subsequently compared to the results obtained in NS3. Subsequent to the aforementioned simulation scenarios, a $proof-of-concept$ implementation was completed using LXC virtualization technology connected over a multi-hop contained network within NS3. The intermediate nodes formed a Wide Area Network (WAN) where each node was an individual LXC machine configured for routing packets across multiple networks. Both

machines exchanged multimedia data using VideoLan Media Player (VLC) as the server and receiver for the application layer process.

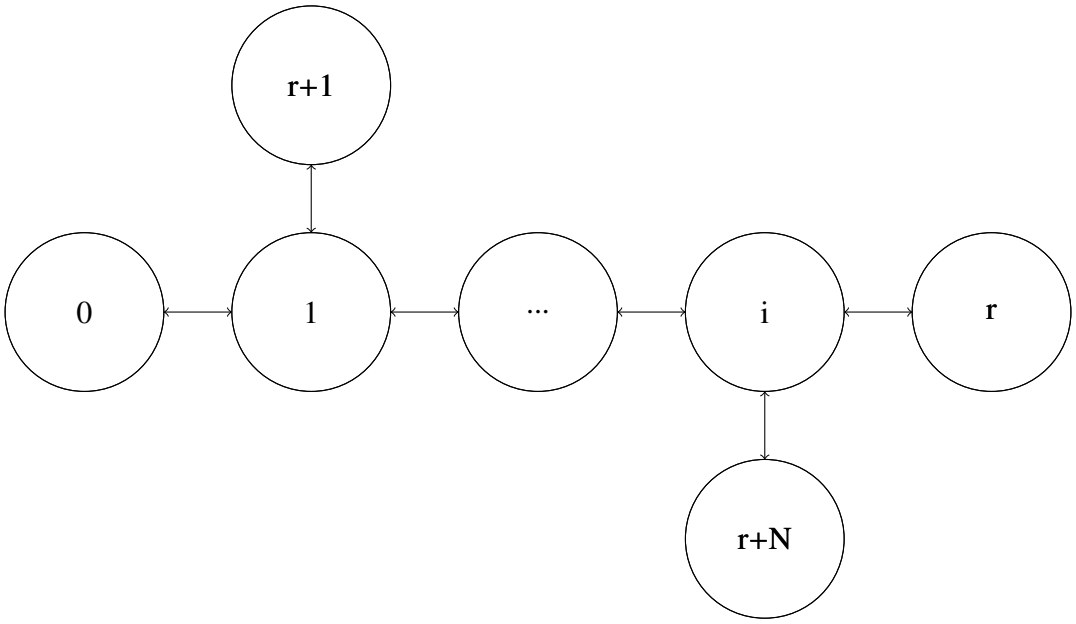


Figure 3.9: Simulated parking-lot topology with N source and destination nodes.

3.4.1 Numerical and Discrete Event Simulation Results and Analysis

The proposed solution was verified using simulations of parking-lot topology networks with a varying number of nodes were simulated using NS3. Each node was generating 16 Mbps of multimedia information per stream. The intermediate nodes were given a large transmission capacity capable of processing a volume of packets relative to the network size. Several session metrics were observed at each node including: the total number of packets exchanged, the total amount of data exchanged, throughput, the average queue waiting time, average packet size, the payload message ratio, the signalling message ratio, and the number of encoded messages. The perceived QoS and QoE metrics are directly affected by the bandwidth consumed in sending and receiving messages. As the number of payload messages increases, the QoS and QoE increase due to the availability of additional multimedia descriptions. The decrease in the average packet size indicates an increased number of embedding processes as well as an

increased number of exchanged payload packets, further improving the QoS and QoE. The average packet size, payload message ratios and signalling message ratios were measured on the up-link and down-link streams. In addition to varying the size of the topology, the proposed scheme was tested under 0.00%, 5.00%, and 10.00% induced channel errors to observe the proposed scheme's behaviour in problematic environments. The aforementioned metrics were observed to quantify the effects of using LSB as means of exchanging signalling information. The effects of the proposed scheme were not limited to the source-destination pairs, but they propagated throughout intermediate nodes as well.

Table 3.2: Sent and received packet sizes were observed at the encoding node. The encoding node was consistently sending larger number of payloads and therefore achieved larger packet sizes on the up-link.

Sent Packet Size	**Encoded**			**Benchmark**		
Nodes / % Error	**0.00**	**5.00**	**10.00**	**0.00**	**5.00**	**10.00**
10	0.53	0.98	1.00	0.53	0.95	0.96
16	0.54	0.98	1.00	0.53	0.95	0.95
20	0.54	0.98	1.00	0.53	0.96	0.97
26	0.60	0.98	1.00	0.53	0.96	0.98
30	0.78	0.97	0.99	0.52	0.95	0.98
Received Packet Size	**Encoded**			**Benchmark**		
Nodes / % Error	**0.00**	**5.00**	**10.00**	**0.00**	**5.00**	**10.00**
10	0.98	0.28	0.18	0.97	0.21	0.20
16	0.98	0.28	0.18	0.97	0.21	0.22
20	0.98	0.27	0.17	0.98	0.20	0.17
26	0.95	0.27	0.17	0.98	0.19	0.15
30	0.89	0.28	0.18	1.00	0.22	0.15

The number of packets exchanged is observed at the source node performing the encoding process and encompasses the total number of sent, received, and dropped packets. Likewise, the total amount of data (MB) exchanged is based on the total amount of data sent, received, and dropped. During the simulation with 0.00% induced channel error, the total number of sent of packets increased by 5.33% however the amount of sent data, excluding the encoded data

stream, increased by approximately 23.13% and as a result, the throughput increased by 5.77%. Considering the encoded data stream, an additional 4.71% increase in number of packets sent is observed. The encoded data stream contributes an additional 0.48% in the amount of data sent. The total number of received packets increased by 18.89% which corresponds to a 14.17% increase in the amount of received data. The number of packets dropped subsequent to using the proposed scheme remained the same. The queue waiting times are observed at the source node performing the encoding process where the average queue time decreases by 29.61%. Even though the average queue time decreased only by 29.61%, the amount of additional sent data did not exceed 23.13% as a result of TCP's greedy behaviour. TCP's greedy behaviour is observed at the source-destination pair, where the source is the node performing the encoding process and the destination is the node receiving encoded data. The destination nodes were receiving signalling messages at an earlier time-slot prompting the transmission of additional payload messages to the source, hence the increase in the number of received packets and data observed at the source node.

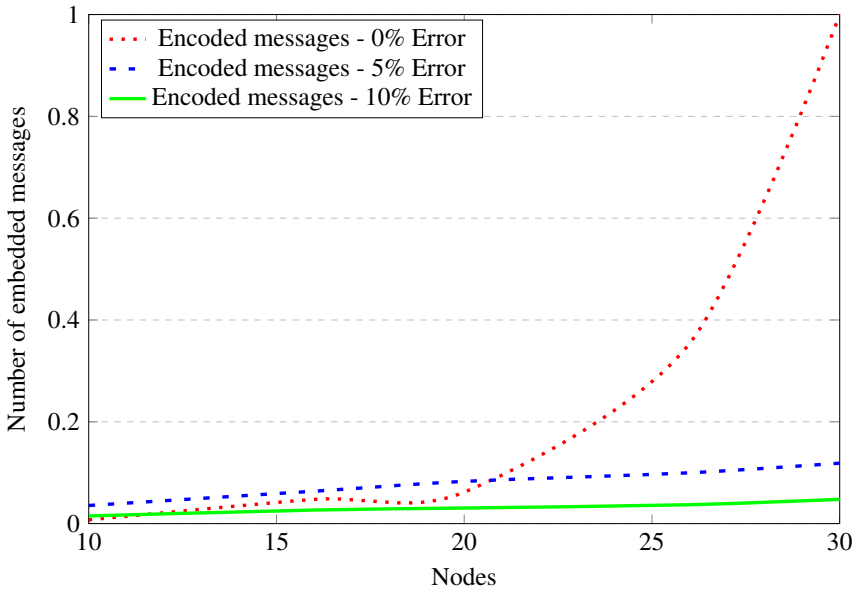


Figure 3.10: The number of embedded messages tends to increase as the size of the topology increases. During the 0.00% induced channel error, the most substantial increase is observed, specifically at the largest topology size. During the 5.00% and 10.00% induced error, the number of messages embedded is significantly less than that of the 0.00% error, however steadily increases as the topology size increases.

The average size of packets sent and received are shown in Table 3.2 where both payload and signalling messages are considered proportionally to the total number of packets sent and received. In Table 3.2, the normalized packet sizes are shown where as the number of payload (signalling) messages exchanged increases (decreases), the average size of packets increases (decreases) as well thereby indicating that the amount of overhead is reduced (increased). On average, the sent packet size increased by 13.94% and the received packet size decreased by 2.70%. The reason for the decreased received packet size is due to the increase in payload messages which prompted an increase in signalling messages received. Subsequent to the embedding process, the ratio of payload messages sent from the total number of packets increased by 13.43% while the ratio of signalling messages sent decreased by 15.28% when compared to the benchmark scenario. The ratio of payload messages received decreased by 2.52% while the ratio of signalling messages received increased by 3.15%, hence the aforementioned decrease of 2.70% of received packet size is inevitable. From Fig. 3.10, the largest number of packets embedded is when the network size was 30 at 0.00% error. Consequently, the largest packet size sent during the 0.00% induced error simulation is also for the 30 node topology as shown in Table 3.2. From Table 3.2, the largest received packet size is seen during the 30 node topology simulation without the proposed scheme. The reason behind the decrease in the size of received messages during the 30 node simulation is due to the increase in the size of sent messages. The increase in the size of sent payload messages prompted an increase in received signalling messages, thereby decreasing the received message size by 11.00% when compared to the benchmark scenario. The decrease in message size did not affect the total amount of received data where, in fact, the total amount of received data did increase by 14.17%.

The simulations conducted under the 5.00% induced channel error achieved a positive balance between the amount of data sent and received. From Table 3.2, under the 5.00% simulation, the average size of packets sent was approximately 2.40% larger while using the proposed scheme when compared to the benchmark scenario. Additionally, the average size of received packets was approximately 7.00% larger than the benchmark counterpart. Taking into consider-

ation the fact that 2.20% fewer packets were sent, the sent payload ratio did increase by 2.18% forcing the average sent packet size to grow. The largest average sent packet size was seen during the simulations with 10.00% induced error as shown by Table 3.2. On average, the sent packet size was 3.00% larger while using the proposed scheme as opposed to the benchmark counterpart. The reason for the increase in the size of sent packets is due to the fact that 2.50% fewer packets were sent while the sent payload packet ratio increased by 2.54%. Subsequent to experiencing a decrease in the number of packets sent, the amount of data sent, excluding the encoded data, remained the same. The encoded data contributed an additional 2.66% to the number of packets sent and that was equivalent to a 0.08% increase in the total amount of data sent. Since the amount of data sent remained relatively the same, the throughput value increased by 0.08%. The average received packet size and the number of received packets remained relatively the same however the amount of received data did increase by 2.03% as a result of a 1.15% increase in the ratio of received payloads.

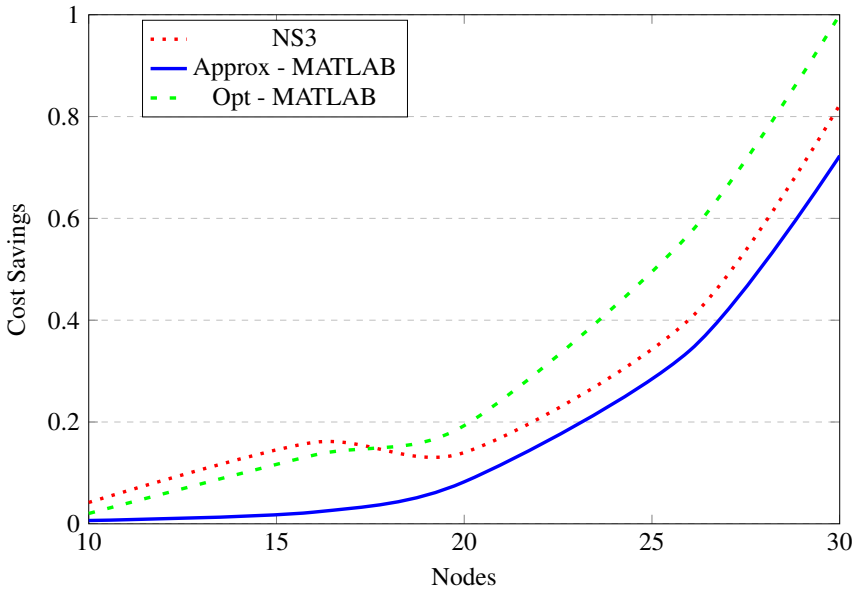


Figure 3.11: The optimal and approximate formulations were evaluated in MATLAB using the OPTI toolbox [154] and compared to the results achieved in NS3. Due to TCP's unpredictable greedy behavior, the NS3 results attempt to remain situated between the approximated and optimal results achieved in MATLAB. While the network was simulated in smaller topologies, TCP's competition for network access is far more aggressive than when the network is simulated in larger topologies.

The results obtained in NS3 were compared to the results obtained using MATLAB as

shown in Fig. 3.11. The results obtained in the NS3 simulation are extremely close to the results obtained using MATLAB and essentially lie between the approximate and optimal solution. The optimal solution is infeasible since the amount of time needed to evaluate the algorithm is significantly large and therefore, cannot be used in real-time. Additionally, the amount of time needed to solve the optimal solution is exponentially larger than that of the approximate solution. The approximate solution is solved almost instantaneously and provides a real-time alternative to the optimal solution. The reason why the NS3 solutions are slightly different for the smaller sized simulations is due to TCP's greedy behaviour, the consequential behaviour that resulted from the embedding process, and the response of the intermediate nodes. The difference is primarily visible for the small-sized simulations since a small topology size allows the effects of TCP's greedy behaviour to be exacerbated. Furthermore, the embedding process forced the TCP congestion window to grow at a faster pace prompting the source node to generate an increased number of signalling messages and consequently, an additional number of embedding processes took place. The intermediate nodes played a vital role in shaping the effects of the embedding process. During the 0.00% induced error simulation, the amount of data sent and received processed by the intermediate nodes increased by 11.14% and 11.15%, respectively. The increase in the size of processed data forced the average queue waiting time to grow by 2.80%. However, during the 5.00% and 10.00% induced error simulations, the increase in the amount of sent and received data was not as significant as that of the 0.00% error simulation where the average queue waiting time decreased by 2.98% and 2.16%, respectively.

In Fig. 3.12, a number of simulations were completed where the overhead size was observed and compared to a benchmark scenario without any overhead reduction schemes. The proposed scheme was simulated and compared to the k-push scheme proposed in [10] and the frequency manipulation scheme proposed in [17]. In each conducted simulation, all participating nodes were performing the aforementioned solutions in different topology sizes. Furthermore, the k-push scheme was extended to use the embedding scheme proposed in this chapter

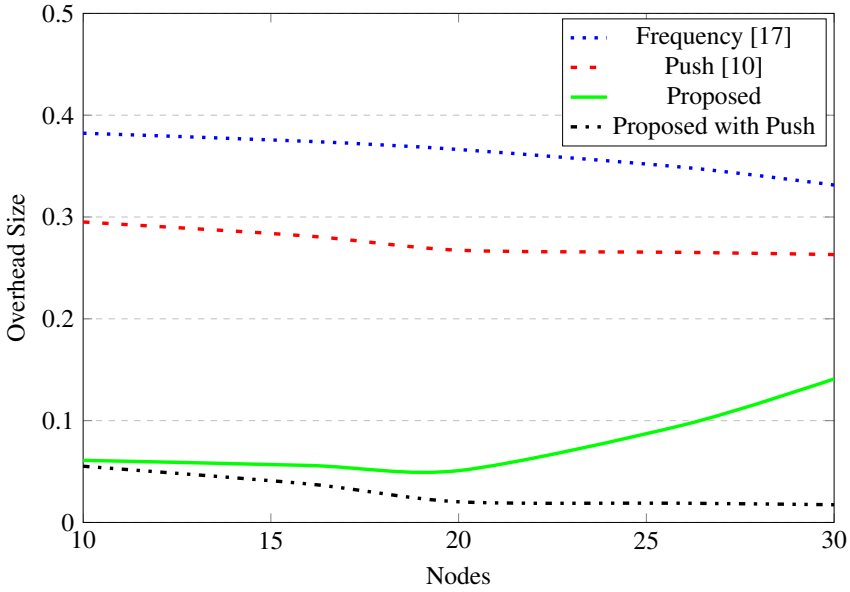


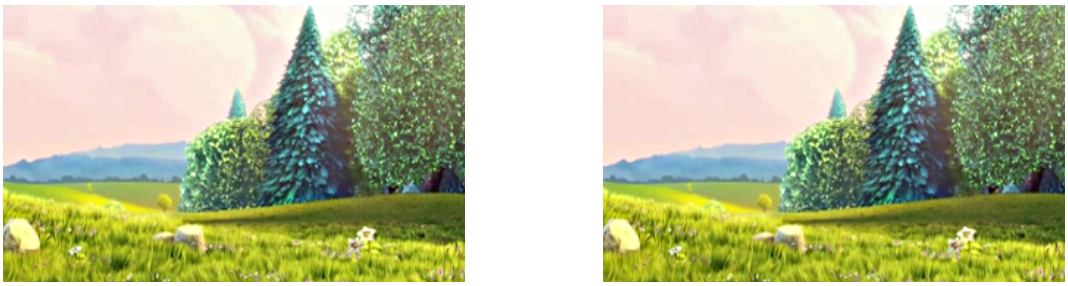
Figure 3.12: The proposed embedding scheme was compared to and extended a k-push and signalling message frequency manipulation schemes. The proposed solution achieved the most considerable reductions in the signalling overhead volume. The embedding scheme was also used to further increase the Push scheme in reducing the amount of signalling information generated during the session.

to further reduce the size of the observed overhead. From Fig. 3.12, the proposed scheme achieved a reduced amount of overhead when compared to the k-push and frequency manipulation solutions regardless of topology size. Additionally, the amount of data exchanged while using the proposed embedding scheme was approximately 5.61% larger than the amount of data exchanged in the k-push and frequency manipulation solutions. The largest amount of overhead reduction was observed when the proposed embedding scheme was combined with the k-push solution. The reason behind the massive success of extending the existing k-push work was the increased number of payload messages when compared to the number of signalling messages generated. From the results illustrated in Fig. 3.12, the proposed embedding scheme was capable of drastically reducing the amount of overhead observed on its own and in cooperation with existing solutions.

3.4.2 NS3 Emulated Proof-of-concept Implementation

To validate the proposed scheme, the results achieved in MATLAB, and the results achieved in NS3, a real multimedia transmission session between 2 virtual machines was completed. Using LXC [161] technology, 2 virtual machines were set up as transceivers exchanging multimedia information. Each machine was using VLC [162] to host a HTTP/TCP server that was providing a video stream to the other receiving node. Additionally, each virtual machine was using VLC media player to request more data and render the received output. The multimedia file used in the transmissions was an open-source video known as Big Buck Bunny [163] and was provided in the popular H.264 format. Fig. 3.13 shows a side-by-side comparison of a frame (right) from the original video without any data embedded and a frame (left) from the video with 70 bytes of embedded information. The two virtual machines were connected over a multi-hop network with five intermediate nodes contained in NS3. The intermediate nodes were set up using LXC containers with routing capabilities and were responsible for connecting the machines with the VLC-based transceivers. The nodes containing the VLC transceivers were performing the encoding (decoding) process.

Subsequent to the validation process, the original video was compared to the video stream with embedded data to identify the perceived Peak Signal-to-Noise (PSNR) ratio. Due to the embedding process modifying only I-frames, the PSNR was $\approx$ 37 dB, which was within the acceptable range of quality change [165]. To reduce the effects of data embedding and achieve improved PSNR values, signalling data must be embedded in B and P-frames rather than I-frames. The achieved cost reductions were $\approx$ 0.19 % in terms of bandwidth, $\approx$ 90.23 % in terms of queue waiting time, and 3.45% in the number of sent packets, while the average sent packet size and throughput increased by 8.03% and 2.81%, respectively. The increase in the average packet size was due to the increase in the number of signalling messages encoded on the forward channel.



(a) (b)

Figure 3.13: The open-source Big Buck Bunny video in H.264 format was used to test the proposed scheme. (a) The image on the left and on the (b) right is a side-by-side comparison of a frame containing embedded data and a frame from the original video, respectively.

3.5 Summary

Adaptive multimedia streaming services over the Internet are notorious for generating large amounts of signalling overhead. A novel cross-layer steganographic scheme is given to reduce overhead resource consumption in multimedia applications. The proposed scheme aims to reduce costs at source nodes and intermediate network nodes. Furthermore, the scheme is designed to multiplex a various number of streams using a unique code identification mapping scheme. The proposed scheme utilizes LSB encoding to intelligently hide signalling messages within payload packets to create a unified data stream. A utility function is given to minimize server side transmission costs while reducing visual artifacts induced by the embedding process observed by users. The proposed scheme dynamically selects nodes to receive embedded signalling information based on continuously updated end-to-end path conditions. Elaborate simulations were conducted to prove the performance of the proposed embedding scheme. The proposed scheme reduces bandwidth consumption and time delays typically introduced by the exchange of signalling messages. Additionally, the stream's performance tends to improve as the number of exchanged messages with content data increases subsequent to a growth in the number of encoded messages. An implementation of the proposed scheme was completed using LXC virtualization technology where two servers and receivers are connected using a multi-hop network exchanged a multimedia segment in the popular Advanced Video Coding

(H.264) format. The LXC setup showed that the video quality post processing is well within the range of acceptable PSNR values while reducing bandwidth consumption, queue waiting times, and number of exchanged messages.

Chapter 4

Reducing Protocol Header Sizes for Enhanced Network Adaption of Request-Response Interactions

After recognizing that steganographic-based embedding of signalling overhead flows is a playable method of reducing the volume of complete non-payload packets, an alternate embedding scheme is proposed to promote further cost savings. HAS-based servers initially provide clients with MPDs to inform them of all available segments where clients will proceed in performing requests of said segments [5]. The available encoding, size, format, length, and other segment configurations deemed necessary by the client will directly influence the amount of signalling overhead generated. Smaller sized segments respond to network changes promptly, are more likely to be delivered during network congestion, and reduce the amount of resources used on a segment-by-segment basis. This chapter proposes a novel protocol translator and routing schemes to reduce HAS overhead while combing the properties of the UDP and TCP protocols within the same stream. The proposed protocol translator exploits steganographic encoding to effectively reduce the size of packets prior to transmission. By using the low complexity LSB-based encoding process, the translator will reduce the impact of using TCP as a reliable end-to-end protocol in multimedia streams through using an alternate protocol. The translator powers source-destination pairs to be capable of converting a TCP stream to a UDP stream prior to-flight and vice versa. In this manner, the benefits of using TCP are observed at the source-destination pairs while reducing resource consumption of packets in-flight due to

using UDP and, by extension, a smaller header size. In the specific case of Device-to-Device (D2D) communications, a Simplified Routing (SR) scheme is designed to be used in place of Internet Protocol Version 4/6 (IPv4/IPv6). A utility function is developed to find the optimal overhead savings where simulations are conducted to verify the designs. The proposed translator is then implemented using VLC [162] transceivers residing in Linux Containers virtual machines where a multimedia file is exchanged in H.264 format. The extensive simulations and implementation platform show that the proposed methodologies will systematically decrease the amount of overhead needed while increasing the overall network capacity. Furthermore, the simulation results demonstrate that the proposed methods are capable of successfully extending existing overhead reduction methods.

4.1 Introduction

In addition to embedding complete signalling messages, the bandwidth of a multimedia-based covert channel can be further exploited to promote additional cost savings through inter-packet embedding. Inter-packet embedding is concerned with hiding current packet information within the payload it is carrying. In this chapter, a novel protocol translator and routing schemes are proposed to extend the existing signalling overhead reduction methods. The proposed protocol translator exploits steganographic encoding to effectively reduce the size of packets prior to transmission. By using the low complexity LSB-based encoding process, the translator will reduce the impact of using TCP as a reliable end-to-end protocol in multimedia streams. The translator is used by source-destination pairs to switch TCP-based streams to UDP-based prior to-flight and vice versa. In this manner, the benefits of using TCP are observed at the source-destination pairs while reducing resource consumption of packets in-flight due to using UDP and, by extension, a smaller header size. The proposed translator operates by having source nodes encode important TCP header information within multimedia payloads using LSB. Once the encoding procedure is completed, a UDP header is used for

packets in-flight, and as a result decreasing the average packet size of a stream. Subsequent to a destination node receiving a payload with encoded information, the receiver will decode the packet and use the extracted information for restoring TCP related information. In addition to the protocol translator, a Simplified Routing (SR) scheme is given to be used in place of Internet Protocol Version 4/6 (IPv4/IPv6). Resource-constrained devices are limited to the number of connections that they can maintain, and therefore, IPv4/IPv6 addresses are wasteful. In the specific case of D2D communications using SR will provide nodes with short logical addresses instead of IPv4/IPv6 addressing, thereby further decreasing header sizes and resource consumption.

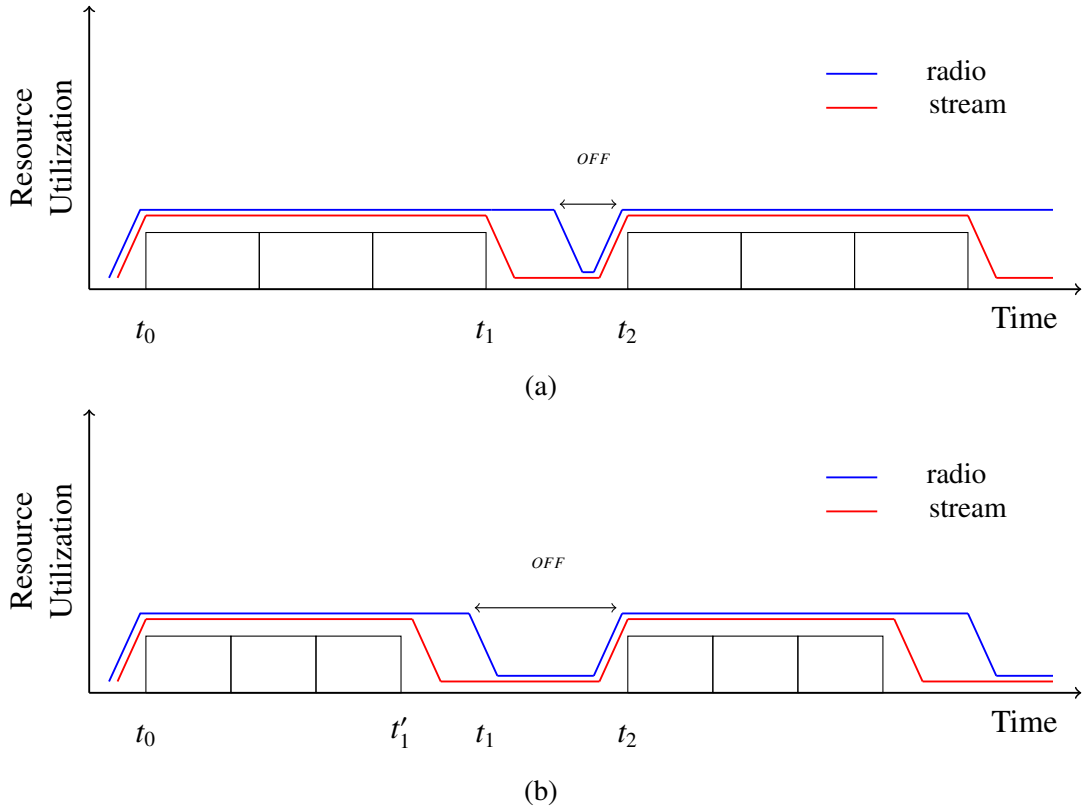


Figure 4.1: The activity patterns of multimedia currently (a) *ON-OFF* behavior due to multimedia streams where ideally (b) a longer *OFF* slot is desired.

Multimedia streams tend to have an intermittent behaviour of *ON-OFF* patterns [119] as illustrated in Fig. 4.1a where resources in devices are behaving accordingly. The benefits of exploiting the proposed LSB and SR schemes are two-fold: firstly, an *ON* period that will

end at t_1' rather than t_1, allowing the radio to enter an extended *OFF* period; secondly, nodes are able to exchange packets at an earlier point such as t_1 rather than t_2. Finally, due to the reduced packet sizes, transmission queues are capable of tightly grouping an increased number of packets within said queue. Reducing packet sizes will ultimately lead to decreased resource consumption in terms of energy, for portable devices, and bandwidth as shown in Fig. 4.1b. The decrease in energy consumption is observed during the transmission of bits of packets as there are a reduced number of bits representing said packets. Additionally, network interfaces in portable devices are able to enter an *OFF* period for an extended period of time. In terms of the transmission queue, packets will occupy a smaller space and will be processed in a shorter period of time due to the aforementioned size reductions. Furthermore, the packets processed with the proposed schemes will utilize a smaller amount of bandwidth where as multimedia fidelity increases, hence the number of packets exchanged increases, the effects of the proposed schemes are more significant. Finally, intermediate nodes participating in the delivery of multimedia streams will have packets occupying smaller spaces within their buffers.

To achieve the aforementioned overhead reductions, the effective size of packets is decreased by employing the proposed protocol translator. Networks lacking in resources will benefit from the LSB technique at very low costs since the implementation is possible through software methods without installing additional hardware. The aforementioned procedures harmoniously cooperate to reduce the average packet size, ultimately decreasing network radio usage within resource-constrained devices and consequently leading to resource savings [120–122]. An optimization formulation is given to minimize the visual artifacts induced by the encoding process while maximizing the amount of overhead saved. LSB is chosen to cipher the TCP related information due the simplicity of performing the encoding (decoding) process as well as the insignificant power consumption [123, 124]. The proposed schemes essentially reduces the overhead on a packet-by-packet basis by replacing TCP headers with UDP headers, as well as the networking layer protocol to use SR instead of IPv4/IPv6, thereby decreasing packet sizes.

The remainder of this chapter is organized as follows: In Section 4.2 the system model description, energy-draining and device cooperation model, as well as the proposed cost-saving utility function is given. In Section 4.3, the proposed translation scheme as well as network offloading evaluation are detailed. In Section 4.4 the simulation and implementation platforms and results and analysed. Finally, a summary is given in Section 4.5.

4.2 System Model Description and Preliminaries

A wireless network consisting of N mobile devices located in a cluster overseen by a Base Station (BS) as illustrated in Fig. 4.2 where $\{n_1, n_2, ...n_j\} \subset N$ devices are requesting a popular multimedia file. In the wireless network mobile devices are capable of establishing D2D transmissions operate by establishing single-hop links between source-destination pairs while reusing cellular channels occupied by nearby radios [133–135]. The nodes will request sets of K segments from each neighbour where each segment is at least 1,460 bytes large. As the fidelity of a multimedia stream increases, the number of segments K_j increases accordingly as well for each destination node j. The total sum of number of bits sent, N_t, and number of bits received, N_r, is larger than total number of bits needed $N_t + N_t \geq \sum_{j \in N} K_j$ due to retransmissions and other propagation errors. Due to utilizing TCP, a connection $S_{i,j}$ is assigned through evaluating the current congestion window size over the maximum achievable window size; consequently, larger window sizes will allow higher throughputs and as result increase the number of encoding operations. The connection between two devices is initially evaluated using the minimum required RTO where a smaller RTO is an indicator of appropriate transmission conditions. However, as the RTO increases, the size of the congestion window decreases and is therefore $\propto \frac{1}{RTO}$. Therefore, $S_{i,j} = \frac{CW}{MW}$, where CW is the current window or current RTO value and MW is the maximum congestion window size or minimum RTO observed. The connection score needs to be greater than pre-determined threshold, L, to reduce the number of lost embedding operations due to retransmissions.

It is assumed that prior to establishing any D2D network connections, each node will receive a unique address that is smaller in size than the 32-bit IPv4 address. Furthermore, each node is capable of performing an affine transformation between the simple address and the true identity of the participating receiver.

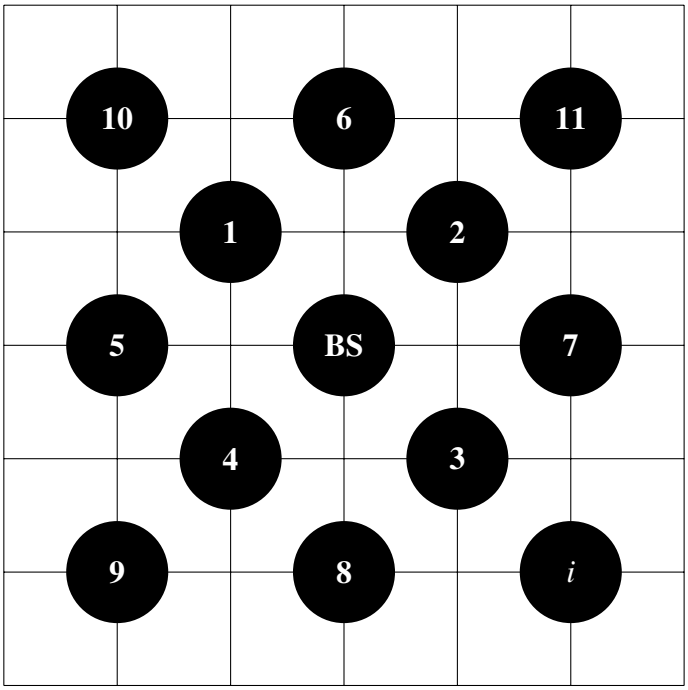


Figure 4.2: Mobile nodes requesting a multimedia file from nearby devices through one-hop connections.

D2D communications are ultimately limited by the available resources of participating nodes where each node will be able to support only a few number of established links. The source-destination pairs will directly exchange live content while leveraging cached multimedia [136–150] files to alleviate Base Stations (BS) from completing redundant requests [125]. Mobile devices utilizing D2D links are subject to significant power consumption and battery life reduction as a result of neighboring users digesting multimedia [126]. Assuming one of the aforementioned D2D caching schemes are utilized, requesting node n_j will be serviced by a nearby device n_{j+k} using D2D communications. For simplicity, devices n_{j+k} and n_j are referred to i and j from hereon in, respectively. Each node will have K payloads available for transmission upon reception of requests. However, due to the limited resources available

Table 4.1: Frequently used notations in this section.

Notation	Definition
i	Source node i
j	Destination node j
N_t	Total number of bits transmitted
N_r	Total number of bits received
T_i	Device i's cooperation time
N	Total number of nodes
K	Total number of messages per node
k	Current message k
I_k^j	Message k being sent to node j
$I_{k,new}^j$	Message k being sent to node j post processing
x_k^j	Binary decision variable for I_k^j in node i
m_j	Network condition binary decision variable for node i
Er_k^j	Error generated due to embedding I_k^j
δ_j	Maximum allowed error
$e_{k,j}$	Effort needed for embedding I_k^j
D_i	Available resources for performing the embedding process
$S_{i,j}$	Connection score between node i and j
L_j	Minimum connection score threshold

within D2D devices, the number of embedding processes must consume a total effort that is $\sum_{k\in K}\sum_{j\in J} e_{k,j} \leq D_i$ to maintain cooperation with nodes. The embedding effort of each TCP header related data is determined by the number of bits subject to embedding for completing the encoding operation. The reason for said limitation is to prevent nodes from consuming the entire energy budget, B_i, of node i. Each node's cooperation is limited with a finite energy budget, B_i, that is split between bits received, b_r, and bits sent, b_t, consumption of energy. Key notations used in this chapter are shown in Table 4.1.

4.2.1 Simplified Routing Address Allocation and Zone Identification

D2D transmissions operate by establishing single-hop links between source-destination pairs while reusing cellular channels occupied by nearby radios. The source-destination pairs

will directly exchange live content while leveraging cached multimedia files to alleviate BSs from completing redundant requests [125]. Delivering multimedia over D2D links is a challenge exacerbated by the limitations of participating nodes, non-deterministic channel conditions, and best-effort guarantees provided by underlying protocols [135]. Participating nodes consume a significant amount of their limited resources, namely energy and bandwidth, to deliver multimedia descriptions over D2D links. For the designed optimization function to operate, nodes using D2D links will agree on their SR identification prior to exchanging multimedia segments to be used in-place of the existing network layer header. Through using the SR scheme, packet sizes are effectively smaller without contributing visual degradation to multimedia streams consequently extending the energy budget contribution. To determine the addresses used in SR, the number of bits representing the participating nodes is given as

$$b = \lceil \log_2 n \rceil, \tag{4.1}$$

where n is the number of participating nodes overseen by the BS. The number of bits b, must be greater than b_{min}, if there are too few nodes sharing the addressing scheme. Each node is then assigned a unique Network Access Code (NAC) by the BS to be used as an alternative to IPv4/IPv6 addresses. If a BS tends to place devices in clusters, the clusters will be given addresses in according to (4.1) as well to generate a complete address as shown in Fig. 4.3.

Logical Address

$0 \cdots 0 - 0 \cdots 0$

$Zone - NAC$

Figure 4.3: The first half of the address represents the cluster while the second half represents a node within the given cluster.

The minimum size of the SR header is at least 1 byte long. With a 1 byte header, the pro-

posed scheme can represent up to 256 devices uniquely in a one-hop communication scenario. If 1 byte SR headers are used in-place of IPv4 protocol headers, the new header size is only 95% smaller than the original headers.

4.2.2 Utilization of Limited Energy Budgets by One-hop Connections

In a D2D network, each device within the cluster is limited by an energy budget of B_i dedicated for D2D communication. The energy budget is defined as

$$\sum_{j \in N_t} b_t(i,j) \sum_{c \in C} r^c_{i,j} + \sum_{j \in N_r} b_r(j,i) \sum_{c \in C} r^c_{j,i} \leq B_i, \tag{4.2}$$

where $b_t(i, j)$ is the energy consumed by device i to perform a transmission of one bit of data to device j, $b_r(j, i)$ is the energy consumed by device i to receive one bit of data from device j, $r^c_{i,j}$ is the throughput of the up-link for file c, and $r^c_{j,i}$ is the throughput of the down-link [127]. It is evident from equation (4.2) that a device in D2D consumes energy while transmitting and receiving information, as well as while idle [128–130], thereby further reinforcing the need of reducing packet sizes using the proposed translator and SR scheme. Using (4.2), the cooperative duration of device i is

$$T_i = \frac{B_i}{\sum_{j \in N_t} b_t(i,j) \sum_{c \in C} r^c_{i,j} + \sum_{j \in N_r} b_r(j,i) \sum_{c \in C} r^c_{j,i}}, \tag{4.3}$$

where T_i is a ratio of the energy budget allocated by i and the total amount of consumed energy in exchanging bits. Maximizing T_i will be an optimal solution to preventing D2D outages from occurring and as a result further increasing the off-loading process from the BS. In (4.2), the energy used in transmitting bits is strictly defined by the underlying circuitry and is difficult to manipulate unless hardware with greater energy efficiency is used. The throughput however plays a significant role in dictating the length of T_i since it is directly proportional to the size, M, of the transmitted segments at rate R. Through leveraging the protocol translator, the size of

packet headers will decrease by 12 bytes at least, due to UDP's 8 byte header size. Therefore, the combination of the protocol translator as well as the SR header will decrease a packet header size by at least 75% when compared to the original header. As the number of bits used in each packet decreases, the total cooperation time of each node will increase accordingly as illustrated in (4.3).

4.2.3 Problem Formulation, Constraints, and Network Assumptions

To minimize the size of the exchanged segments, variables x_k^j and m_j subject to optimization by each device i using the following non-linear optimization objective function given as

$$\max_{\{x_k^j\},\{m_j\}} \sum_{j=1}^{N}\sum_{k=1}^{K}(I_k^j - I_{k,new}^j)x_k^j m_j \tag{4.4a}$$

s.t.

$$\frac{\sum_k^K Er_k^j x_k^j}{\sum_k^K x_k^j} \leq \delta_j \qquad \forall j \in N \tag{4.4b}$$

$$\sum_j^N \sum_k^K x_k^j e_{k,j} \leq D_i \tag{4.4c}$$

$$\frac{\sum_j^N S_{i,j} m_j}{\sum_j^N m_j} \geq L_j \tag{4.4d}$$

$$m_j \in \{0,1\} \qquad \forall j \in N \tag{4.4e}$$

$$x_k^j \in \{0,1\} \qquad \forall j \in N,\ \forall k \in K \tag{4.4f}$$

In (4.4a), I_k^j is the packet with identification k chosen to use the proposed solutions before heading to destination j and $I_{k,new}^j$ is the new size of the packet post processing. To determine

whether a packet will be processed by the proposed schemes, binary decision variables x_k^j and m_j must be set to high. The x variable is determined by the first and second constraints of the proposed optimization. Constraint (4.4b) ensures that the embedding process does not create visual artefacts by maintaining an error level below δ_j. Constraint (4.4c) ensures that the embedding effort required $e_{k,j}$ by all packets does not exceed the available processing limit D_i to avoid overwhelming the source-destination pair from the encoding and decoding process. Each TCP connection established by source-destination pairs will be given a score $S_{i,j}$ based on the size of the congestion window and the received control messages where higher performing streams will receive higher scores. Using m_j in constraint (4.4d) will ensure that only stream with a connection score greater than L_j will receive encoded data and as a result reduce the effects of packet loss on the system. Reducing the effects of packet loss will avail by decreasing wasted time and energy in resource scarce devices. Due to the translation process and SR, packet sizes are subject to change, and therefore the observed throughput values of streams between i and j are subject to change as well. Since the device cooperation time T_i is affected by throughput values, it is important to maintain it above T_{min} to maximize the cooperation time. As illustrated in Fig. 4.4, the proposed translator and SR schemes will end packet transmissions at t_1' rather than t_1. Decreasing the size of each packet will result in shorter bursts ending at t_1' thereby enabling longer off periods and increased cost savings.

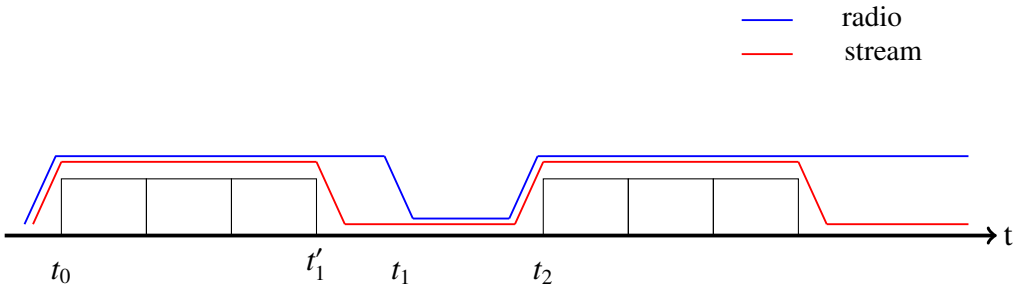


Figure 4.4: The shorter bursts will allow radios of resource constrained devices to enter the $OFF-state$ for a longer duration or exchange packets at an earlier point in time.

Furthermore, nodes establishing D2D links need to choose peers with Signal-to-noise ratio (SNR) [131] that must be larger than a γ_{min} for optimal performance. The SNR is evaluated as

$$\gamma_{i,j} = \frac{p_{t_i} h_{i,j} d_{i,j}}{\sigma^2}, \tag{4.5}$$

where p_{t_i} is the transmission power of node i, $h_{i,j}$ is the channel gain from node i to j, $d_{i,j}$ is the distance between the source-destination pair, and σ^2 is additive Gaussian noise. However noise is not only present within the channel, it is also present in the multimedia payloads subsequent to the embedding process. The visual error due to the embedding process is

$$Er = \frac{1}{PW} \sum_{p=1}^{P} \sum_{w=1}^{W} (f(p,w) - f'(p,w))^2, \tag{4.6}$$

where P is the height and W is the width of a considered frame. In (4.6), the initial sample value, $f(.)$ located at $p \in P$ and $w \in W$ of the reference frame, is compared to the new sample value, $f'(.)$ located at $p \in P$ and $w \in W$ in the new frame, subsequent to the embedding process.

4.3 Steganographic-based Protocol Translation Scheme

Unlike traditional traffic flows similar to what is shown in Fig. 4.5a, steganography is used to enable protocol translation by embedding information in the payload as shown in Fig. 4.5b. Using steganography a source node is capable of translating an HTTP/TCP stream to be HTTP/UDP in-flight and a destination node translates the received stream back to HTTP/TCP using the encoded information. The receiver is aware of the presence of embedded data by fingerprinting special codes similar to the ones proposed in chapter 3. Using the unique codes, the correct interpretation by the receiver is ensured. The protocol translation process requires cooperation between the application layer and transport layer as shown in Fig. 4.6. The node performing the translation process is designed to embed header data within multimedia streams on a packet-by-packet basis. Prior to the embedding process, a source node must decide whether to embed complete TCP headers, to embed unique TCP header data only to minimize visual artifacts, or use a modified UDP header for transmission to further reduce packet sizes.

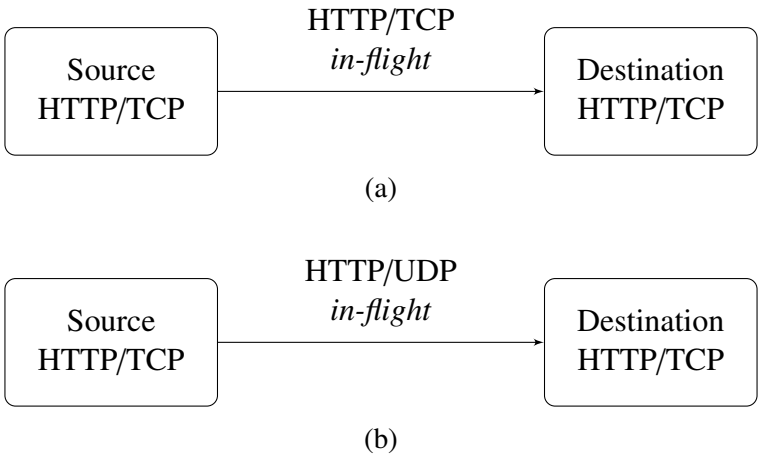


Figure 4.5: Multimedia streaming will utilize (a) HTTP/TCP from end-to-end or (b) HTTP/UDP during flight as enabled through the protocol translator.

Stack	**Proposed**
Application	Translation
Transport	
Network	SR
Data-link	Data-link
Physical	Physical

Figure 4.6: Due to the cross-layer commitment, the application layer and transport layer are summed into a single layer declared as the translation layer.

Embedding complete TCP headers is the simplest form, however they contain redundant data since TCP and UDP headers share the same source and destination port numbers. To reduce the visual effects of the embedding process and minimize unnecessary redundancies, source nodes will not embed data that is shared between TCP and UDP headers. By the virtue of the encoding process, essentially modifying the multimedia payload, the UDP and TCP checksum fields are now redundant as well. The UDP checksum field is therefore utilized to further reduce the visual artifacts by embedding data within said field while the TCP checksum field will be completed by the receiving node prior to processing. To further reduce the observed visual artifacts, the maximum packet length will be agreed upon ahead of time between source-destination pairs. By determining the maximum packet length, the maximum number

of bits needed by the length field in the UDP header will be known. Knowing the maximum number of bits needed from the UDP length field will free the remaining bits to be used in a similar manner as the UDP checksum field. The aforementioned decisions made by the source node will decrease the effects of the embedding process. If a source node decides to focus on reducing packet sizes, a modified UDP header will be used in place of the default UDP header.

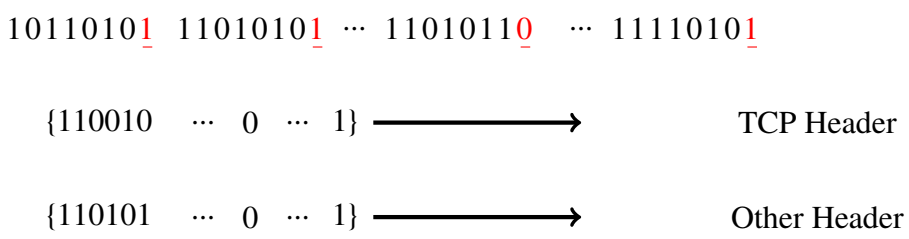


Figure 4.7: Using the embedded codes, the receiver will be able to identify the type of header that is embedded.

The modified UDP header will not contain the source and destination port number fields since embedding a complete TCP header will include said fields. By not including the source and destination port numbers, the UDP header will be 50% smaller in size. The source node will proceed by treating the length and checksum fields in the previously mentioned manner to reduce visual artifacts while using the modified UDP header. To use the modified UDP header, it must be employed in conjunction with a networking layer protocol that contains a protocol identification field, such as IPv4. The IPv4 header contains a protocol identification field that is used to inform the receiver of the presence of a transport layer protocol, such as TCP or UDP, where TCP is identified as protocol number 6 and UDP is identified as protocol number 17. The protocol identification field in the IPv4 header can uniquely identify 256 protocols since the field is 8 bits long [132]. Since not all 256 numbers of the protocol identification field are assigned, a protocol number for modified UDP is chosen at the networking layer to be used by source-destination pairs. The receiving node will have to perform packet reconstruction operations to restore the original HTTP/TCP packet from the HTTP/UDP packet received. By receiving an HTTP/UDP packet, a node will be aware that the packet is modified and contains embedded data. To properly reconstruct a packet, a receiving node will use the unique codes

found in Fig. 4.7. Using the unique codes, a receiving node will need to either decode a complete TCP header, or reconstruct a TCP header using the UDP source and destination port numbers, or reconstruct a modified UDP header using the encoded TCP information.

4.3.1 Evaluating Base Station Offloading within One-hop Networks

The non-linear objective function presented in the previous section requires a large amount of time to find a solution, and therefore a more convenient process is needed. In this section it is assumed that the BS will behave in a greedy manner with the desire to off-load the maximum amount of data possible to surrounding nodes. It is in the best interest of the BS to maximize the collaboration time T_i of participating nodes and as a result achieve a high data offload rate. Maximizing the duration T_i is determined by how well the available energy budget B_i of each D2D device is utilized. For the optimal balance between the needs of the BS and the length of T_i, the BS will communicate to the D2D devices available the desired hit ratio hr to manipulate T_i of the said devices according to the needs of the BS. In order for the BS to successfully determine hr, each D2D device must inform the BS of the available B_i. Algorithm 1 is designed to take into consideration the needs of the BS subsequent to D2D devices declaring their limitations. The BS evaluates the desired hr_{min} based on

$$hr_{min} = \frac{\sum_{i \in N} B_i}{N \times hr_{max}}, \tag{4.7}$$

where hr_{max} is the total number of cache hits expected by each node i, N is the total number of devices participating, and B_i is the energy budget of each participating node. The given algorithm will continuously be evaluated and updated after each iteration. Using the proposed algorithm, each node will participate a minimum number of times prior to the BS needing to re-evaluate its strategy. Once the BS receives the B_i of participating D2D devices, D2D participant devices will receive an expected hr_{min}. Essentially, hr_{min} is a measure of expected amount of work given to each D2D device to allow enhanced B_i expenditure.

Algorithm 1: Balancing packet processing with BS needs.

```
if D2D available then
    while hr_i < hr_min do
        if Packets determined successfully then
            if Encoding successful then
                Update T_i and hr_i ;
            else
                select other packet;
            end
        else
            Wait for new time-slot;
        end
    end
    if hr_i ≥ hr_min then
        if B_i ≥ B_min then
            BS updates hr_min value ;
            goto 1;
        else
            Node i is not considered for D2D
        end
    else
        End D2D communication;
    end
else
    BS completes request ;
end
```

Subsequent to exchanging B_i and hr_{min} values, each device i will determine the packets subject to the translation process by using the linear optimization function in (4.8a). The proposed linear formulation aims to maximize the number of translations while making sure that connection established between the D2D nodes is robust, thereby reducing the number of needed retransmissions and as a result extending the collaboration time T_i. The linear formulation optimizes variable $x_{j,k}$ for maximizing the reduction of bits is

$$\max_{\{x_{j,k}\}} \sum_{j=1}^{N} \sum_{k=1}^{K} (I_k^j - I_{k,new}^j) x_{j,k} \tag{4.8a}$$

s.t.

$$\sum_{j}^{N}\sum_{k}^{K} x_{j,k}e_{k,j} \leq D_i \tag{4.8b}$$

$$x_{j,k} \in \{0,1\} \qquad \forall j \in N, \forall k \in K \tag{4.8c}$$

Constraint (4.8b) ensures that the sum of embedding efforts $e_{k,j}$ is less than the total embedding capacity D_i to prevent overwhelming the source-destination pairs. Using the number of bytes available for sending subsequent to the embedding process, it is possible to estimate the remaining cooperation time of node i by updating the throughput values and calculating T_i from (4.3). The new value of T_i will be sent to the BS to help the BS determine a new value of hr_{min} if necessary. Once the D2D device reaches the predetermined hr_{min}, the device will update the BS with the new values of T_i and by extension the remaining B_i. Using the updated values, the BS will determine whether a given device is fit for servicing future requests, and if so, the number of allotted requests.

4.4 Simulation and Implementation Configurations and Results

The proposed translator and SR schemes are simulated in a wireless cluster of nodes using NS3 [153] and MATLAB as two separate validation platforms. The D2D network is set up to use the energy saving methods in a number of simulations subsequent to performing a series of counterpart benchmark tests. The general topology configuration used is shown in Fig. 4.8 and is subject to grow in number of rows and columns as additional nodes are added. In addition to varying the simulation size, the number of established links varied as well. All nodes are capable of encoding (decoding) information within payload data on the up-link (down-link) channel. Each node is generating data at no more than 2 Mbps where each packet is at least 1,460 bytes large. The packets are generated according to a Pareto Poisson

Burst Process (PPBP) to mimic a realistic traffic shape [155–158, 160]. Furthermore, all nodes establishing D2D links are using the proposed SR scheme for routing rather than the traditional IPv4/IPv6 networking layer. Subsequent to the aforementioned simulation scenarios, a $proof - of - concept$ implementation was completed using machine virtualization residing within a portable battery powered device. The virtual machines established a D2D network over a wireless channel contained within NS3 where one virtual machine acted as a server while the other requested multimedia content. The device's energy drain rate and transmission statistics were logged during a benchmark scenario and while using the proposed translation scheme to determine the effects of said scheme.

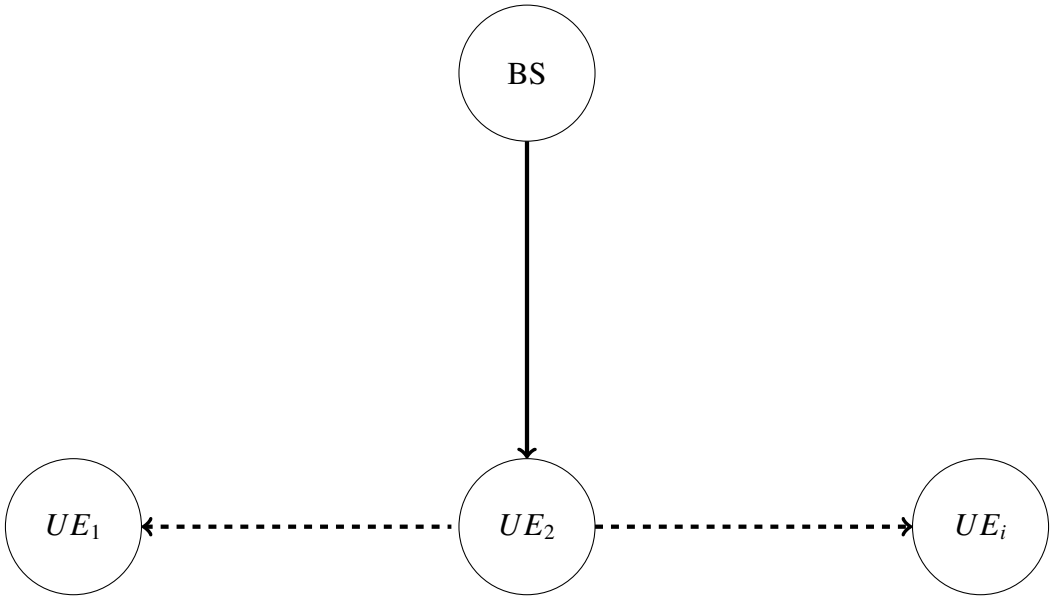


Figure 4.8: Simulated wireless topology where the dashed lines represent D2D connections and the solid line represents BS connections.

4.4.1 MATLAB and NS3 Simulation Results and Analysis

The proposed schemes were verified using simulations in NS3 and MATLAB with varying topology sizes and number of established D2D connections. In addition to varying the topology size, the nodes were stationary in the initial set of simulations while mobile in the latter. The node positions were randomly selected, however, the configuration was maintained across

repeated simulations to ensure a proper comparison scenario. The mobility of the nodes was random and described using a Gauss-Markov model in a three-dimensional space. The source applications were generating a maximum of 2 Mbps of data for each individual stream according to a PPBP process. Finally, the simulations were conducted while having multi-sources transmitting data and then repeated with a single source only. The results obtained in NS3 were then compared to the results observed in the MATLAB simulations. Three primary metrics were observed throughout the sessions including: node cooperation time, amount of data exchanged, and the amount of overhead reduced. It is assumed that all nodes have similar circuitry and energy consumption rates and therefore the NS3 generic Lithium Ion battery model was used for all simulations.

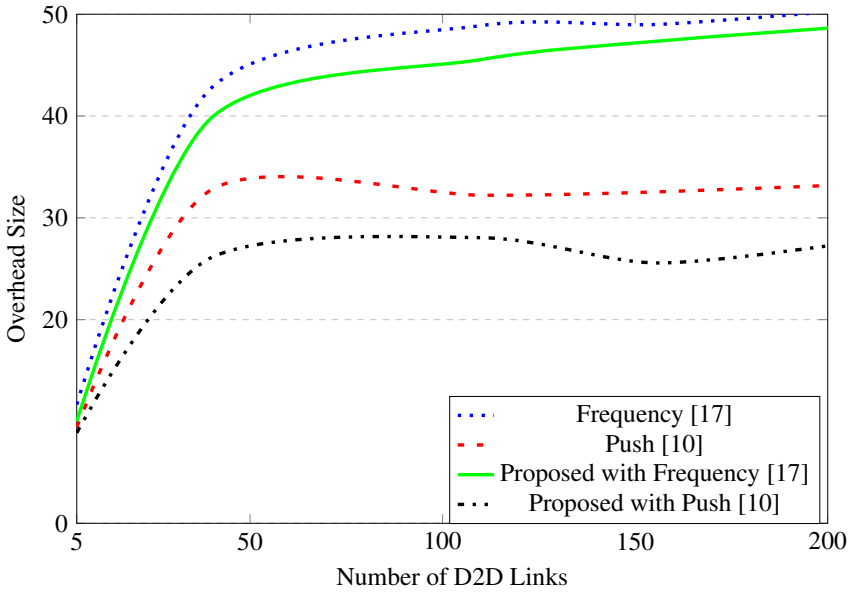


Figure 4.9: The proposed schemes successfully extended signalling reductions solutions [10, 17] to further decrease the bandwidth consumed by non-payload data. Using the proposed protocol translator and SR, an additional 6.5% in signalling overhead reductions was achieved in addition to the signalling reduction achieved by the solutions proposed in [10, 17].

The proposed scheme was used to extend existing overhead reduction methods as illustrated by Fig. 4.9. The schemes chosen to extend using the proposed protocol translator and SR layer are found in [17] and [10]. The TCP protocol is a byte-oriented stream where an ACK message is expected to be sent by a destination node for every segment successfully re-

ceived from a source node. The ACK messages are important for the TCP protocol since the congestion window grows according to the number of ACK messages received. In [17], the authors proposed reducing the number of Transmission Control Protocol (TCP) Acknowledgement (ACK) frequencies by manipulating the settings of the protocol, essentially delaying the delivery of ACK messages. Furthermore, in [17], the authors propose using Appropriate Byte Counting (ABC) to compensate for the reduced frequency of ACK messages to maintain the desired performance. When using byte counting, the source node TCP congestion window will grow according to the number of acknowledged bytes by the received ACK messages rather than the number of ACK messages received. In [10], the authors propose using server push capabilities built-in HTTP 2.0. The server push capabilities enable content providers to send multiple multimedia descriptions and maintain a persistent connection with destination nodes without requiring a request for each description. The overhead reductions methods proposed in [17] and [10] achieved approximately 60% and 70% overhead reductions, respectively. Using the proposed scheme to extend the aforementioned works, overhead size decreased by approximately an additional 6.5%. Therefore, employing the proposed scheme in conjunction with the existing signalling reduction solutions will decrease a session's bandwidth consumption by non-payload data. The decrease in bandwidth promotes a decrease in energy consumed by non-payload and similar type of data as well as enable tightly packed transmissions. Devices are accessing a wireless channel at successively smaller intervals, thereby spending a reduced amount of time in idle and an increased amount of time transmitting packets.

The proposed protocol translator and SR methods reduce the size of packets and by extension directly affect the cooperation time as described by (4.3). As the availability period of nodes increase, the length of time a device can share a media feed behaves accordingly. Bandwidth consumed through sending and receiving data affects a node's cooperation time on a bit-by-bit basis. As the number of bits exchanged between source-destination pairs decreases, a node's availability time will behave accordingly, providing an extended coverage period for the multimedia feeds available within the said device. The proposed optimal formulations of

the protocol translator and SR schemes are simulated in MATLAB to find the signalling reductions and subsequently used to determine the cooperation time of nodes using (4.3).

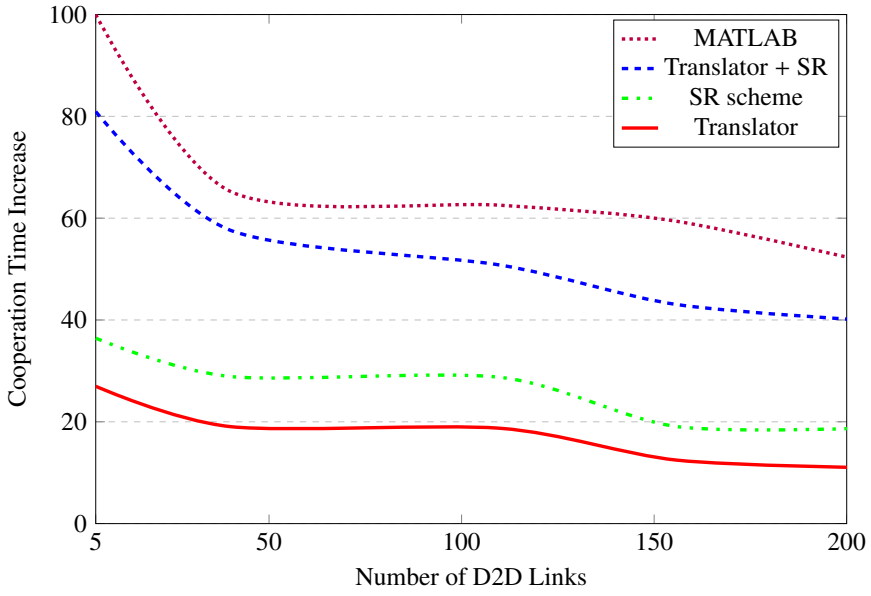


Figure 4.10: Nodes are capable of participating within the network for an extended period of time due to utilizing the proposed solutions. However, the cooperation time of nodes decreases as the number of established D2D links increases. The effects of the number of established links per nodes on the node availability period is demonstrated. The node availability time increased significantly while using both the translator and SR solutions simultaneously due to the largest decrease in packet size.

The MATLAB simulation results describing a node's availability period were then compared to the node's cooperation time measured in NS3 simulations illustrated in Fig. 4.10. The MATLAB simulations are completed under the set-up of combining both the translator and SR schemes simultaneously; however, the NS3 simulations are completed under three different variations including: translator only, SR only, and translator with SR simultaneously. According to the illustrated results, both MATLAB and NS3 confirm that the number of established D2D links affected the observed node cooperation time. The cooperation time of each node tends to decrease as the number of established links increases, whether the translator is used with or without the SR scheme, or vice versa. The largest gain in terms of node cooperation time is observed on the MATLAB platform and confirmed by NS3 simulations where the network had the fewest number of established D2D links. Therefore, in large topologies with the potential of establishing a sizeable number of D2D links, it is ideal to form clusters of small

D2D networks and limit the number of D2D links established within each cluster.

The results illustrated in this section demonstrate the capability of extending existing schemes to further reduce the signalling overhead size, increase node cooperation time, and increase the amount of data exchanged subsequent to utilizing the proposed schemes. Ideally, the proposed methods should be used in situations where nodes are clustered together in small groups to maximize the amount of signalling overhead saved in terms of cooperation time and bandwidth.

4.4.2 Virtualization-based Implementation on a Portable Resource Limited Device

To validate the protocol translation process as well as the aforementioned simulation results, a real multimedia transmission session between 2 virtual machines was completed. Using LXC [161] technology, 2 virtual machines were set up to exchange multimedia information as shown in Fig. 4.11. One virtual machine was using VLC [162] to host an HTTP/TCP server while the other was using VLC for playback. The multimedia file used in the transmissions was an open-source video known as Big Buck Bunny [163] and was provided in the popular H.264 format. Fig. 4.12 shows a side-by-side comparison of a frame, 4.12a, from the original video without any data embedded and a frame, 4.12b, from the video with 24 bytes of embedded information. The two virtual machines were connected to form a D2D network over a wireless channel contained within NS3. The implementation was set to run on a battery powered device to measure the change in the device's power consumption induced by the encoding and decoding processes. The multimedia stream exchanged 5.99% more descriptions while using the proposed translation scheme when compared to a traditional streaming scenario. The energy consumption of the device increased by 0.88% which is due to the increases in exchanged data as well as the encoding (decoding) processes completed. The average waiting time of packets within the transmission queue decreased by $\approx 10\%$. Subsequent to the validation process, the original video was compared to the video stream with embedded data to identify the perceived

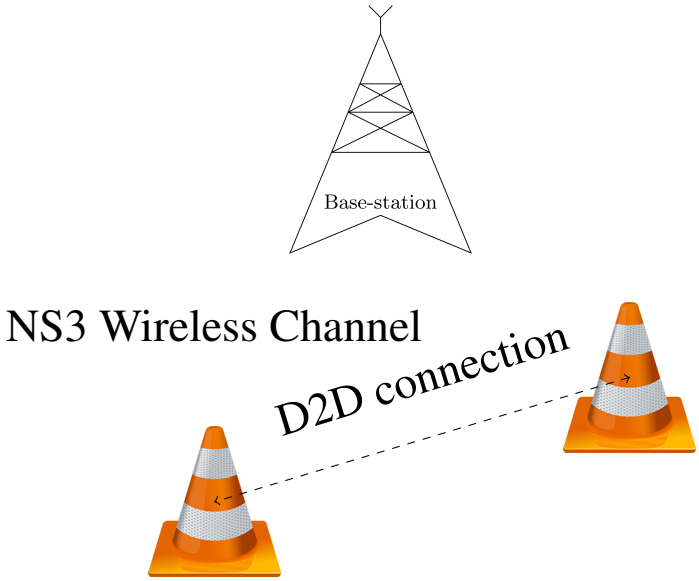


Figure 4.11: Using a configuration of a VLC media player within LXC containers on a Linux host machine, the proposed protocol translator was implemented. The D2D-capable devices where behaving as media servers and sinks connected to each other using a wireless channel contained in NS3. The devices establish a direct D2D connection to exchange live and cache multimedia segments.

peak signal-to-noise (PSNR) ratio. The PSNR was 36.7 dB, which was within the acceptable range of quality change [164]. To reduce the energy impact induced by the encoding (decoding) process, the software code must leverage energy efficient routines.

(a) (b)

(c)

Figure 4.12: The (a) original frame is compared to (b) a frame containing embedded data where a superimposed (c) figure of the two frames is used to highlight their differences.

4.5 Summary

In this chapter, a novel cross-layer steganographic translator is given to enable resource constrained devices in decreasing the average size of packets. The translator powers source nodes with the ability to use lightweight headers within packet flight while maintaining original protocol characteristics. The proposed scheme translates multimedia streams from HTTP/TCP to HTTP/UDP in-flight. A source node will use LSB to encode TCP data in multimedia descriptions where destination nodes will decode embedded data to restore TCP related information. In addition to the translator, a SR scheme is given to enable mobile nodes in using a lightweight alternative to IPv4/IPv6, further reducing the average size of packets exchanged. The proposed schemes are designed to work simultaneously and in harmony with existing overhead reduction mechanisms to induce additional resource savings across a wider range of networks. Extensive simulations are presented on MATLAB and NS3 to validate the proposed schemes. The simulation results show that the proposed methods are capable of cooperating with existing overhead reduction schemes, increase node cooperation, and increase available bandwidth while exceeding QoS constraints. An implementation of the proposed scheme was completed using LXC virtualization technology where one server and one receiver were connected using a D2D network and set to exchange a multimedia segment in H.264 format. The perceived PSNR values observed post implementation are well within an acceptable range while reducing transmission queue times for outgoing packets.

Chapter 5

Increasing Decentralized Network Throughput of Multimedia Content Access and Rights Management Systems

Reducing packet sizes and the number of exchanged signalling messages are effective methods of decreasing service costs and increasing the efficiency of multimedia payload delivery. However, as the size of multimedia services increases, blockchain-based networks are utilized for their secure decentralized features, thereby inducing service access latencies and decreased content playback throughput. Blockchain-based networks operate through generating a large number of transactions destined to be securely recorded on a ledger shared primarily by network maintainers, formally known as miners, and any interested nodes. Consequently, miners responsible for maintaining a common ledger will have large mining pools filled with transactions waiting to be processed. Therefore, in order to achieve and maintain desired QoS and QoE requirements, multimedia applications utilizing blockchains will need miners that follow a tiered transaction processing structure. The tiered transaction processing structure will enable transactions in reducing their mining pool residence time. In this chapter, a novel tiered mining structure designed to reduce transaction mining pool residence time while preventing transaction processing starvation, decreasing content access latency, and increasing network throughput is proposed. The proposed processing scheme is tiered according to the priority class a transaction belongs to where higher priority transactions are more likely to be guaranteed space in future blocks than low priority transactions. To prevent abuse of the high priority

transaction status flag, the proposed scheme is designed to circumvent low priority transaction starvation through an adaptive block slicing mechanism. The proposed block slicing scheme allocates chunks of a soon-to-be published block to transactions of varying priority classes found in the mining pool. Using the proposed slicing algorithm, a blockchain-based network is capable of guaranteeing a processing throughput of transactions of different priority classes. A transaction ageing process is proposed where transactions will increase in weight as their mining pool residence time increases. Subsequent to a miner evaluating the slice size of each priority class, the miner will select the transactions with the highest weights to fill the slices from its mining pool. The scheme is validated through elaborate simulations as well as a proof-of-concept implementation where nodes are built and used to exchange transactions of different priority classes.

5.1 Introduction

Popular applications operating over the Internet, such as multimedia streaming services, attract a large audience and as a result must be scalable to serve the growing demand. Extending applications and services with blockchain-based solutions resolves a number of deployment challenges including: scalability, decentralized trust, traceability, and immutability [169]. A blockchain-based multimedia service will utilize miners located locally [170] or in edge nodes [171, 172] to maintain a system's ledger. Each miner must try to publish a block to be appended to the ledger by grouping select transactions from said miner's mining pool to form transaction blocks. Maintaining a ledger in a blockchain-based system is one of the most important blockchain-related activities that must be performed by miners continuously regardless of the success of previous mining operations. The importance of a ledger's upkeep resulted in a low throughput and computationally expensive maintenance process where transactions wait patiently in mining pools for inclusion in soon-to-be published blocks [59]. The effects of a blockchain's low processing throughput are observed by end-users due to an increase in

multimedia access latency and verification of content licensing agreements. Finite transaction space found in blocks is a blockchain-based system's defining element that is necessary to govern the throughput of an underlying multimedia network. Furthermore, block sizes are non negotiable and must be agreed upon by all miners and ledger maintainers prior to commencing operations. Another throughput limiting factor is the mining complexity set by a blockchain network to maintain a desired security level during operations [173]. The mining complexity varies with time and is a key element to blockchain security and decentralization. While a miner is waiting until it can publish a transaction block, a miner's mining pool will continue to receive new transactions of all priority classes. The finite transaction block space limits the number of transactions included, thereby forcing a number of transactions to reside for a longer period of time within a miner's mining pool. As transactions reside in mining pools, the multimedia network will experience reduced throughputs due to the increase in transaction processing latency within miner nodes [29]. Miners adopt different policies for selecting transactions for an upcoming transaction block, and therefore not all transactions have the same processing throughput.

Existing mining policies are primarily geared for blockchain systems used in digital currency applications where miners utilize the notion of fees to guarantee priority of transaction processing. A cryptocurrency miner is capable of developing a fee structure scheme outlining charges for providing different levels of guarantees. In the presence of a fee structure scheme, transactions are capable of minimizing their mining pool residence time by paying higher fees than other transactions, thereby ensuring their inclusion in a soon-to-be published block. In the absence of fees and cryptocurrencies, such as blockchain-based multimedia access services, miners will have to resort to publishing transaction blocks by selecting transactions through alternative methods. The existing literature [41–60] provide transaction selection schemes by either depending on paid fees structures to ensure mining or by treating transactions on a First In First Out (FIFO) basis. The aforementioned solutions do not address scenarios without cryptocurrencies, such as blockchains-based multimedia networks, that require transaction pri-

ority processing. Additionally, the existing schemes depend on fees to complete transactions, thereby enabling richer sources to secure larger amounts of mining resources over less fortunate sources. Furthermore, as the number of transactions waiting to be processed in the network increases, a miner is capable of demanding an increased fee due to the larger competition for transaction spaces within soon-to-be published blocks. Finally, miners will focus on financial interest and behave in a greedy manner by only mining transactions that pay the highest fees. Although financially driven mining disciplines are simple and successfully introduced tiered mining schemes, they fail to address systems independent of fees and cryptocurrencies.

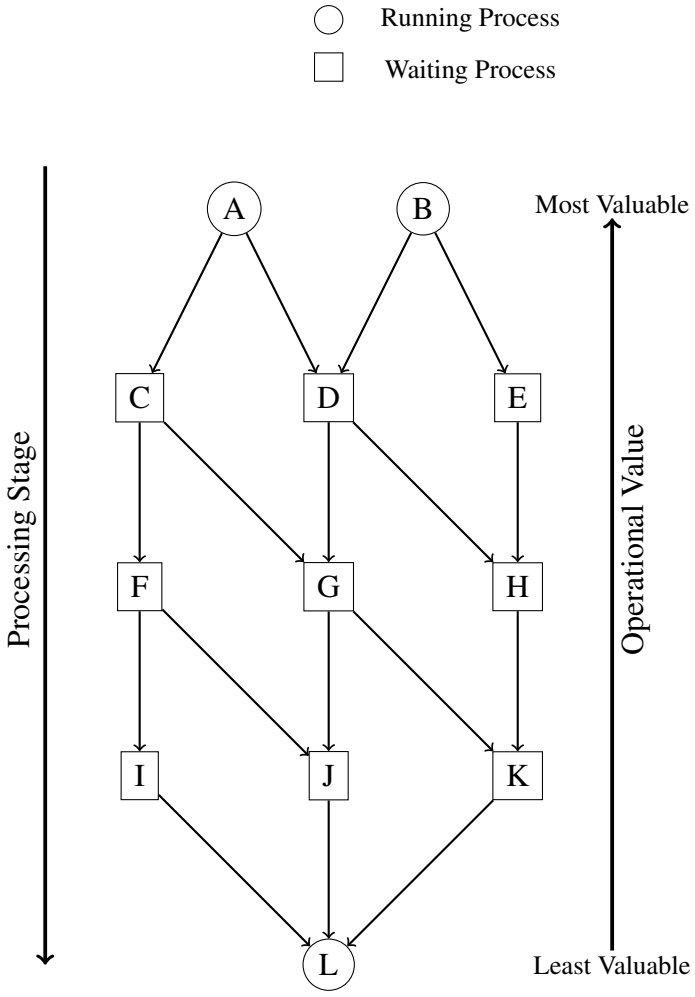


Figure 5.1: Processes at earlier stages are the most valuable due to enabling a large amount of future processes to take place.

The large size of multimedia services will demand scalable solutions to address digital

content access, rights, and management. Blockchains are a secure way of enabling the scalability of multimedia services [28–35] however said multimedia applications will suffer from decreased QoS and QoE values. The decrease in QoS and QoE descriptions is a direct consequence of incompatible miner transaction processing disciplines attempting to operate outside of digital currency applications. In this chapter, a feeless tiered mining discipline is proposed for blockchain-based multimedia applications and other non-digital currency services. Consequently, blockchain-based multimedia content access and rights management will overcome the limitations of financially driven mining schemes. By overcoming financially driven mining schemes, multimedia service providers are capable of increasing content playback throughput as well as decreasing content access latency. Similar to the fee structure of financially driven mining, the proposed scheme is a feeless tiered processing structure that is introduced to prioritize transaction mining. The tiered processing structure depends on the priority of the process generating the transactions, the priority label found within a transaction, or the priority of the process generating a transaction as illustrated by the operational value in Fig. 5.1. While employing the proposed scheme within miners, a miner will include transactions in upcoming blocks based on priority where higher priority transactions are guaranteed block space over lower priority transactions. To ensure the processing of low priority transactions, the scheme is equipped with a utility function designed to alleviate low priority packet starvation. In any tiered mining discipline, whether fee based or otherwise, low priority transaction processing starvation is possible if the number of high priority transactions found within a mining pool far exceeds lower priority transactions. The proposed process prevents low priority transaction starvation by employing a utility function designed to guarantee low priority transaction inclusion in soon-to-be published blocks.

The remainder of this chapter is organized as follows: In Section 5.2, the system model description as well as the proposed transaction selection utility function are given. In Section 5.3, the proposed block slicing algorithm is detailed. In Section 5.4 the simulation and implementation platforms and results and analysed. Finally, a summary is given in Section 5.5.

5.2 System Model Description and Problem Formulation

A blockchain-based multimedia network similar to the illustration in Fig. 5.1 will benefit from a feeless mining structure to enable transaction processing priority while preventing low priority transaction starvation. Each process at each stage is generating transactions of events and communicating them to miners within the network for ledger appending. Assume that there are multiple processes and nodes categorized into $I = \{0, 1, ..., i\}$ priority classes. Each miner is receiving K transactions from each priority class where a miner will form a block of transactions from said transactions. The total sum of the size of transactions $S = \sum_{i \in I} \sum_{k \in K} S_i^k$ must be less than the total network pre-determined block size C_B. Due to the limitation C_B, as the number of generated transactions increases, the mining pools will buffer an increased number of transactions from all classes. The key notations used in this section are shown in Table 5.1.

Table 5.1: Summary of frequently used notations in this section.

Notation	Definition
n	Current node n
N	Set of nodes
i	Priority class i
I	Set of priority classes
k	Transaction k
K	Set of transactions
x_i^k	Binary decision variable for transaction k
W_i^k	Weight of transaction k from class i
S_i^k	Size of transaction k from class i
C_B	Size of transaction block
V_i^k	Value of transaction k from class i
V_B	Expected value of current block
λ_H	High priority transaction arrival rate
λ_L	Low priority transaction arrival rate
$B(C_B)$	Denial of service probability
S_i	Class i slice size

The priority and importance of transactions are directly related to the stage the transactions belong to as indicated in Fig. 5.1. Therefore, due to the interdependence of processes, parent tasks have a higher priority when compared to tasks down the production line. As tasks are tirelessly running, transactions are continuously generated by participating devices and sent to dedicated blockchain miners for processing. Transactions generated by higher priority tasks will receive guaranteed processing and inclusion in future transaction blocks. Once a block of transactions is created, the block is then shared with the rest of the miners in the blockchain network. As transactions are being stored within the miners' mining pools, the number of high priority transactions will eventually exceed their lower priority counterparts. The slow throughput of blockchain networks in the presence of excess high priority transactions will result in starvation of lower priority transactions. A scenario where the number of transactions from high priority classes far exceeds transactions with low priority is considered to mimic low priority class starvation. The slow throughput is unavoidable due to the security features of a blockchain-based network, and as a result the throughput of content playback as well as access is therefore significantly reduced. Therefore, managing the transaction processing discipline within miners is an effective method in alleviating processing starvation.

If unmitigated, low priority class starvation will eventually result in distributed transaction blocks that do not include a significant amount of transactions of said class. To prevent low priority transaction starvation, miners will utilize (5.1a) to determine the allocation of transactions in soon-to-be published blocks. Low priority transaction starvation will lead to longer transaction residence time within a miner's mining pool. Each transaction added in a soon-to-be published block will contribute a value V_i^k to the operations of the network, where the total block contribution is $V = \sum_{i \in I} \sum_{k \in K} V_i^k$. The value of each transaction depends on the number of processes and future actions it enables to occur where as the number of succeeding transactions increases, the value V_i^k increases as well. The utility function aims to minimize the total weight of the transactions to be included in a soon-to-be published block, however, each block must contain transactions that contribute a value, $V \geq V_B$, to the operational flow.

Each transaction k from priority class i is given a weight, W_i^k, that is directly correlated with the class priority. As the priority class of a transaction increases, the weight of each transaction increases accordingly. The utility function utilizes x_i^k to minimize the total sum of weights $W = \sum_{i \in I} \sum_{k \in K} W_i^k$ and is evaluated as

$$\min_{\{x_i^k\}} \sum_{i=1}^{I} \sum_{k=1}^{K} W_i^k x_i^k \tag{5.1a}$$

s.t.

$$\sum_{i=1}^{I} \sum_{k=1}^{K} S_i^k x_i^k \leq C_B \qquad \forall i \in N \tag{5.1b}$$

$$\frac{1}{V_B} \sum_{i=1}^{I} \sum_{k=1}^{K} V_i^k x_i^k \geq 1 \tag{5.1c}$$

$$x_i^k \in \{0, 1\} \qquad \forall j \in N, \ \forall k \in K. \tag{5.1d}$$

Constraint (5.1b) ensures that the total sum of all transaction sizes S_i^k is less than or equal to the block size agreed upon by the blockchain network. If the sum of transaction sizes exceeds the network determined size, the block will be refused by other miners or the publishing miner must drop some transactions back to the mining pool. Constraint (5.1c) ensures that the transactions within the block must add an operational value of V_B or greater when being published to the network. The operational value of each process is determined by the number of child processes that depend on the considered source. If a process enables a larger number of sub-processes to occur down the production line, then the related transaction will be assigned a higher V_i^k. In (5.1a), the sum of the weights of each transaction W_i^k is aimed to be minimized to ensure the processing of low priority transactions, however, the value added by each block must be at-least V_B. To determine whether a transaction will adhere to proposed fair processing scheme, binary decision variable x_i^k must be set to high.

5.3 Proposed Block Slicing Algorithm for Decentralized Throughput Provisioning

The proposed slicing algorithm is a real-time estimation of the aforementioned utility function that divides soon-to-be published blocks into compartments dedicated to hold a number of transactions from different priority classes. While using the slicing process, low priority transactions are subject to an ageing process where the oldest transactions are included in their respective slices first. The ageing process will ensure that low priority transactions left behind from previous block publishing iterations will have priority in inclusion over recently received transactions. The objective function presented in the previous subsection requires a large amount of time to find a solution, and therefore a real-time process is provided. It is assumed that each process is generating two classes of transactions, high and low priority, according to a Poisson distribution with a rate of λ_H and λ_L, respectively. Each transaction k, whether high or low priority or in a miner's mining pool, is given a weight that is found as

$$W_i^k = \frac{\delta_i^k}{C_i}, \tag{5.2}$$

where δ_i^k is the variable used for the ageing process of transaction k and C_i is the cost of delaying a transaction of class i. The ageing process is designed to increase the priority of transactions as they age in the mining pool. By ageing transactions, their priority level is elevated to ensure future processing and inclusion in soon-to-be published transaction blocks. The ageing process is necessary since the number of high priority transactions in a miner's mining pool may completely occupy a future transaction block and absolutely prevent low priority transactions from being published. The ageing process is designed to counteract the starvation of low priority class transactions introduced by the abundance of high priority class transactions. Additionally, in scenarios where greedy processes abuse the high priority classification, low priority transactions will still be published. The ageing variable is found as

$$\delta_i = T_d - T_a, \tag{5.3}$$

where T_a is the arrival time of a transaction to the mining pool and T_d is the expected departure time of a transaction. The value of T_d is essentially T_a in addition to the waiting time T_i of a transaction within the mining pool. Subsequent to each iteration of a new transaction block formation and publishing, the value of T_d will grow by T_i. As the value of T_d grows, the weight given to a transaction will also grow and therefore, age successfully. Assuming transactions of high and low priorities are arriving according to a Poisson process with aggregate rate λ and they are processed at deterministic rate D, then the expected waiting time of a transaction from class i [151] within the mining pool is

$$T_i = \frac{R_i}{(1-\rho_1)(1-\rho_1-\rho_2)}, \tag{5.4}$$

where R_i is the residual service time of class i transactions and ρ is the utilization factor based on the arrival rate λ and miner service rate μ. The mean residual service time depends on the average transaction size L and is given by

$$R_i = \frac{1}{2}\sum_{i \in I} \lambda_i \frac{L}{D}. \tag{5.5}$$

Due to the mining process and complexity set within the blockchain network, the waiting time T_i is the time needed to complete the mining process. The mining process in large systems, such as Bitcoin, takes on average 10 minutes to see a new block published within the blockchain network. The finite size of a transaction block, C_B, and the Poisson defined behavior of transactions generated, some transactions will not be included in future blocks. By knowing the deterministic processing time of transactions found within the mining pool, the service denial probability [152] is found. The probability characterizing the possibility of a transaction not being included in a new transaction block is

$$B(C_B) = \frac{1 + (\rho - 1)Q_{C_B}}{1 + \rho Q_{C_B}}, \tag{5.6}$$

where Q_{C_B} is the number of transactions found within the current block being prepared for publishing. The denial-of-service probability (5.6) is then used to find the aggregate arrival rate λ that the miner can handle from all processes. The aggregate arrival rate is used to define the total arrival rate experienced by the miner from all processes to achieve a desired denial-of-service rate. Therefore, the miner node is aware that there are scenarios where high priority transactions will take precedence for inclusion in soon-to-be published blocks and low priority transactions must remain within the mining pool. The miner is essentially determining the ratio of stalled low priority transactions by controlling the observed λ from each participating node. The aggregate value of λ is found as

$$\lambda = \frac{\mu U}{1 - B(C_B)}, \tag{5.7}$$

where U is the utilization observed by the miner due to the incoming transactions. In order to have a 0% denial-of-service rate, a miner must request from participating processes to generate transactions at an aggregate rate that is less than μU. If there are I processes, then each device must generate at a rate of $\lambda_n = \frac{\mu U}{I}$. Likewise if a miner desires a denial-of-service rate $B(C_B)$, then the aggregate value of λ is found first using $\frac{\mu U}{1-B(C_B)}$. The value of λ is the sum of high, λ_H, and low λ_L, priority transaction arrival rates. Once a miner determines the aggregate value of λ for all processes, the expected needed occupancy for a high priority class is

$$E_H = N_H S_{H,a}, \tag{5.8}$$

where $S_{H,a}$ is the average size of transactions, N_H is the current number of high priority transactions in the mining pool as well as the expected number, $E[N_H]$, of high priority transactions arriving at λ_H. Each transaction class will then occupy space in a new transaction block according to the found expected occupancy value. Based on a transaction block's total space

available, C_B, a priority class slice size, S_i, is

$$S_H = \begin{cases} \frac{E_H}{C_B}, & E_H < C_B, \\ W_H \frac{E_H}{C_B}, & E_H \geq C_B, \end{cases} \tag{5.9}$$

where W_H is the weight of transactions evaluated as

$$W_i = \frac{W_{class}}{W_{total}}. \tag{5.10}$$

The total weight of a class, W_{class}, of transactions and W_{total} is the total weight from all classes. As low priority transactions age in the mining pool, the weight of the low priority class will increase as a whole due to the aforementioned ageing process. Through ageing, the value of W_{class} will change and therefore, the priority class slice size will fluctuate accordingly. Using the value of aggregate λ as well as α_i, the observed average number of arrivals from class i in time period t, the miner can expect high priority transactions to arrive at a rate of

$$\lambda_H = \frac{\alpha_H}{\alpha_H + \alpha_L}\lambda, \tag{5.11}$$

where $\frac{\alpha_H}{\alpha_H+\alpha_L}$ is referred to as h from hereon in. The term h is the observed fraction of arriving high priority transactions from all transactions over a period of time t. Using h, the expected arrival rate for low priority transactions is

$$\lambda_L = (1 - h)\lambda. \tag{5.12}$$

Subsequent to evaluating λ_H and λ_L, the expected number of incoming packets is found. The total number of transactions during period t is

$$N_i(t) = \int_{t_0}^{t_1} \alpha_i(\tau) d(\tau). \tag{5.13}$$

If a miner observes time period $t = t_1 - t_0$, the miner will find the expected number of transactions from class i as $N_i = \lambda_i t$. The miner's advantage of evaluating the expected number of transactions of class i, is found in estimating W_{class}, W_{total}, and as a result W_i. Furthermore, the expected number of arriving transactions will enable the miner to determine the slice size of class i by initially evaluating (5.8). Due to the miner using variable h to determine the expected number of transactions from a certain class, (5.9) is approximated as

$$S_H = hC_B, \tag{5.14}$$

where the remainder of the fraction is allocated to determine the slice size for low priority transactions as

$$S_L = (1 - h)C_B. \tag{5.15}$$

Using (5.14), a miner is able to allocate a portion of a soon-to-be published transactions block to hold at least S_H fraction of C_B as high priority transactions and S_L as low priority transactions. Subsequent to finding the amount of space reserved for each priority class, the miner will form a new block from transactions with highest W_i^k values. The finite size of a block will force a number of transactions to remain within the mining pool and therefore the weight of the transactions residing in said pool will continuously increase according to (5.2) regardless of their priority class. As a result, low weight transactions from any class will have to stay within the mining pool until inclusion in a future block or network wide adoption through a block published by another miner. In the event of transaction adoption to ledger through a block published by another miner, the W_{class}, W_{total}, and W_i variables are updated by excluding the weight of said transaction. The new slice size is

$$S_i' = G_i S_i, \tag{5.16}$$

where G_i is the smoothing factor to counteract the changes introduced by the updated W_i

value. Accounting for the decentralized and distributed cooperation found within blockchain-based networks, miner nodes can set G_i to a value that will benefit the network as a whole by increasing the total allocation for class i. The total allocation of slices for all classes, $G_1S_1 + G_2S_2 + ... + G_iS_i \leq S_T$, must be less than S_T to prevent the class allocation from exceeding C_B block size limitation as illustrated by Fig. 5.2.

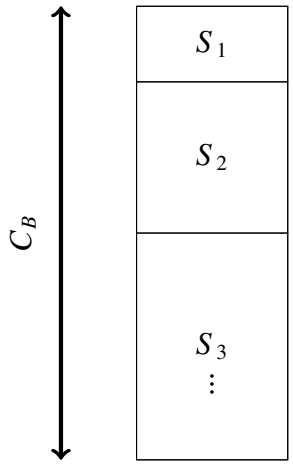


Figure 5.2: All transactions from class i must not occupy more than their class allocated slice S_i.

5.4 Simulation and Implementation Platforms and Configurations

NS3 [153] was used to simulate a network of devices sending transactions to a miner in an industrial setting as shown in Fig. 5.3. The nodes were communicating with each other as well as the miner using a wireless interface running on the IEEE 802.11 standard. By using NS3, an industrial setting was accurately mimicked through a lossy channel, random transaction broadcast times, varying volume of generated transactions from each node, random distances between nodes, random placement of nodes, random mobility of a portion of the nodes, and varying the network size. The nodes generated transactions with payload sizes ranging from 20 to 100 bytes at a maximum aggregate throughput of 16,000 transactions per second (tps). The transactions were generated according to a Pareto Poisson Burst Process (PPBP) to mimic

a realistic traffic shape [155–160]. The transaction priority class was randomly chosen by each node prior to a transmission. Although the priority class was randomly chosen, each node was designed to generate a larger number of high priority transactions to simulate low priority class starvation. To simulate node movement, a 3D Gauss-Markov mobility model [159] was used. It was assumed that a miner's mining pool holds unlimited transactions and a miner tries to publish a new block every 10 minutes. Each NS3 simulation run was used to generate a snapshot of a mining pool for further MATLAB processing. MATLAB was used for the Opti Toolbox [154] to evaluate the proposed optimization formulation as well as the evaluation of the proposed real-time estimation.

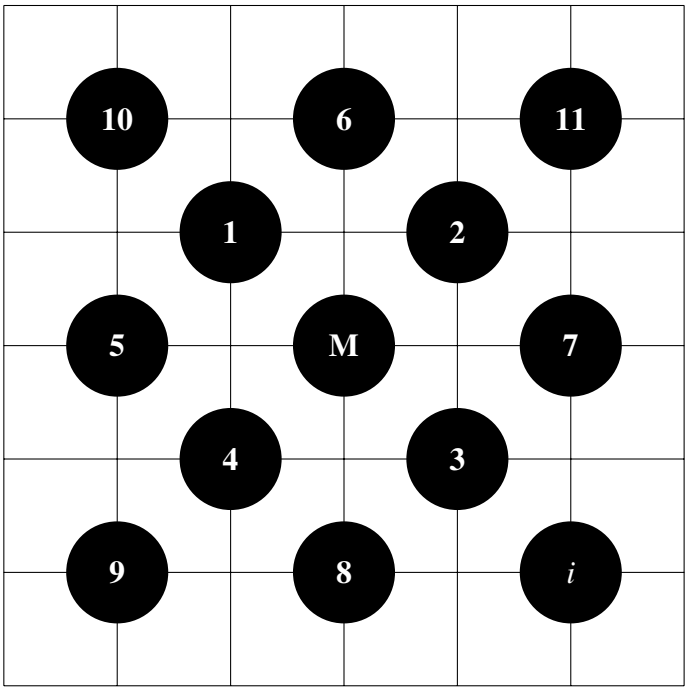


Figure 5.3: Simulated network topology where nodes are placed randomly with a miner, node labelled M, in the center.

5.4.1 Simulation Results and Analysis

The proposed scheme is verified using simulation based methods where nodes generated transactions of various sizes at random intervals of time. The network generated a maximum throughput of 16,000 tps in all simulation topologies and configurations. The nodes generated a

large number of transactions that are either high or low priority where the system is designed to generate more of the former. In doing so, the generated low priority transactions experienced starvation and had to remain within the mining pool of a miner. Subsequent to generating transactions and entering the starvation period of low priority transactions, the miner is responsible for determining the transactions that will be included in the new transactions block. The objective of the miner is to include low priority transactions in the new block, however, the published block must include transactions that are valuable to the operations of the network. Blockchain miners are the nodes responsible for publishing new transaction blocks where each block holds a finite number of transactions. The block size for the simulations is set at 1 MB while the transaction sizes ranged from 20 bytes to 100 bytes. Furthermore, the network is assumed to publish a new block every 10 minute intervals. In a blockchain-based multimedia service network, miners are capable of preparing multiple blocks for publishing in a sequential manner and therefore the block probability is illustrated as shown in Fig. 5.4. As the number of blocks increases, the the blocking probability decreases due to the presence of more entities willing to hold an additional number of transactions. The increase in number of blocks is synonymous with increasing the transaction block size from 1 MB to a larger value, or to a maximum of 7 MB in case of Fig. 5.4. Since the system is designed to simulate low priority transaction starvation, transactions sent during smaller generation rates are therefore subject to experiencing blocking as well. From Fig. 5.4, it can be seen that as the aggregate value of incoming transactions rate, λ, increases, the blocking probability will behave accordingly.

The number and size of blocks generated by blockchain miners in multimedia networks will vary according to the implementation and usage scenario. To overcome the variability of environment and implementation, the system is measured in terms of utilization, U, to achieve a desired performance as illustrated in Fig. 5.5. The utilization experienced by each miner will vary according to the aggregate value of the arrival rates of high and low priority transactions. Furthermore, the utilization is also affected by the number of blocks generated or the total size of a transactions block. As the transaction arrival rate λ increases (decreases), the observed

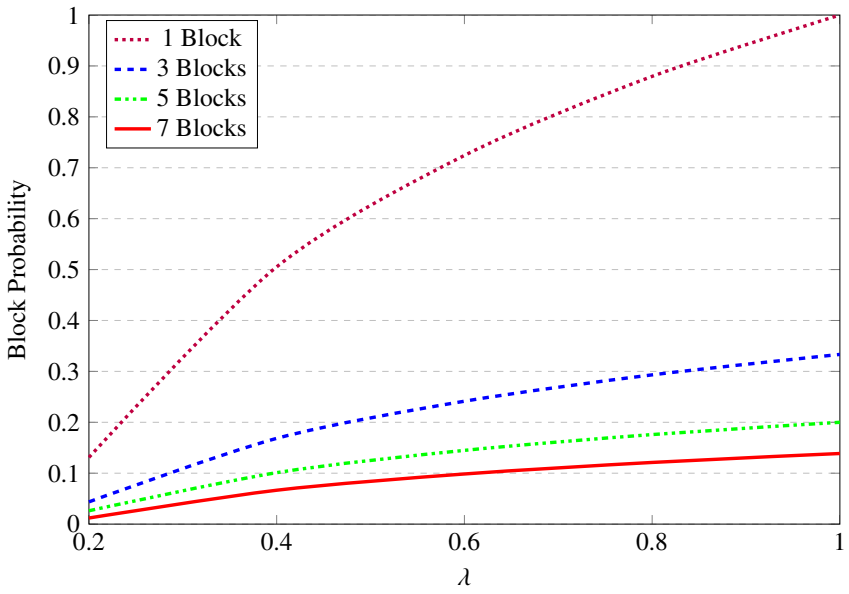


Figure 5.4: The denial of service probability experienced by low priority transaction coming as part of an aggregate rate of λ.

utilization will increase (decrease) as well. However, as the number of transaction blocks or the size of a transaction block increases (decreases), the observed miner utilization decreases (increases). The relationship between the transaction arrival rate, block size, and utilization can be used to attain a desired blocking probability. It can be seen that a miner generating only 1 block at a time will reach full utilization at a lower λ rate than when generating multiple blocks at a time. Assume a scenario where a miner is generating three blocks at a time while only observing 75% utilization. In the aforementioned scenario, the miner has the option to consolidate the operations into a single block rather than three to increase the utilization from 75% to 100%. In doing so, the miner will publish fewer blocks, thereby consuming fewer resources within the network, while achieving the same utilization. Therefore, using Fig. 5.5, a miner is capable of adapting its effort according to the load presented, essentially manipulating the denial of service probability to a desired value. If the miner chooses to increase (decrease) the number of published blocks, the miner is essentially decreasing (increasing) the block probability while maintaining a desired utilization value and by extension throughput values. A miner will ideally aim for 100% utilization to minimize wasted block space and consumed

resources. In a scenario where a miner is generating only 1 block of transactions every 10 minutes, the denial of service probability for low priority transactions will be approximately 30%.

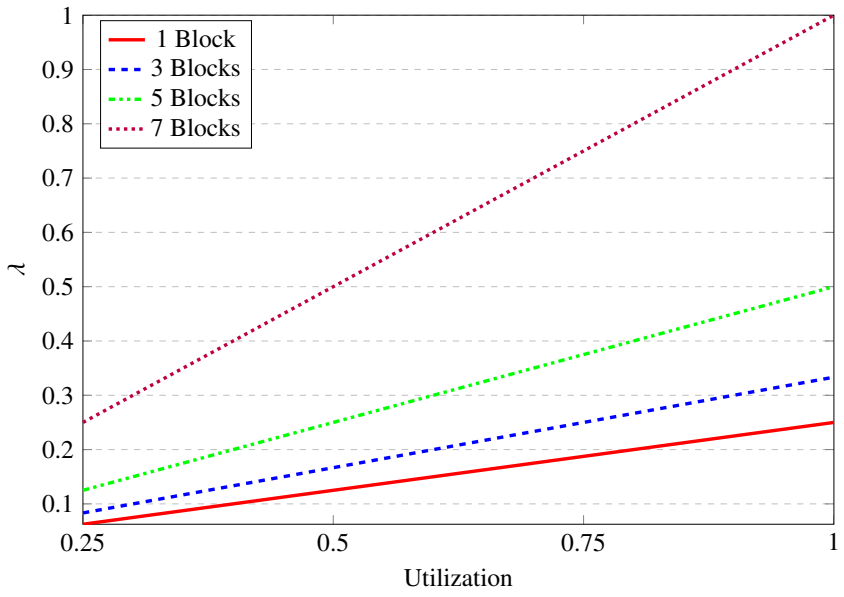


Figure 5.5: The aggregate transactions arrival rate, λ, directly affects the utilization rates of a miner.

To measure the throughput of the blockchain network, and by extension the high and low priority transaction publishing time, the number of transactions added to the ledger is observed over 10 minute intervals. The mining pool of a miner is infinitely larger than a transaction block to hold all incoming transactions and all transactions that are not published due to the overwhelming presence of high priority transactions. The throughput of high and low priority transactions as the total number of transactions increased within the mining pool is illustrated in Fig. 5.6 and in Fig. 5.7, respectively. From the simulations, it is evident that the throughput of high and low priority transactions is directly affected by the number of transactions within the mining pool due to affecting the denial of service probability. As the number of high priority transactions increased, the denial of service probability of low priority transactions increased as well, therefore the throughput experienced by low priority transactions changed. According to the benchmark scenario, as the number of transactions within the mining pool increases, the throughput will eventually be the result of high priority transactions being published only.

By publishing high priority transactions only it means that 100% denial of service probability for low priority transactions is achieved. From Fig. 5.6, it can be seen that the proposed scheme in the optimal scenario decreased the throughput of high priority transactions by approximately 10.81%. The decrease in high priority transaction throughput indicates an increase in the low priority transactions' throughput by an equivalent amount. Furthermore, the low priority transactions are no longer experiencing any denial of service regardless of the number of transactions within the mining pool. Due to the optimal formulation consuming a large amount of time to find a solution, the real-time estimation is evaluated as well. The real-time estimation decreased high priority transaction throughput by 19.15%, and therefore successfully increased low priority transaction throughput by an equivalent amount. Therefore, by using the proposed scheme, low priority transactions are still serviced even during times where they normally would have been blocked.

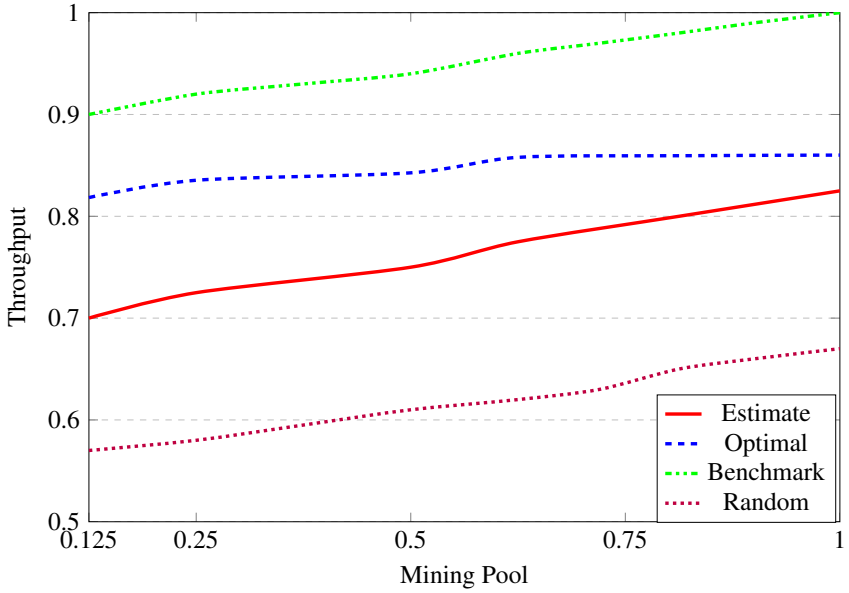


Figure 5.6: The throughput of high priority transactions as the number of transactions within the mining pool increased.

The observed throughput generated subsequent to using the proposed scheme is compared to a benchmark scenario and a random scenario. The benchmark scenario is the case where a miner behaves in a typical manner where high priority transactions are favored over low pri-

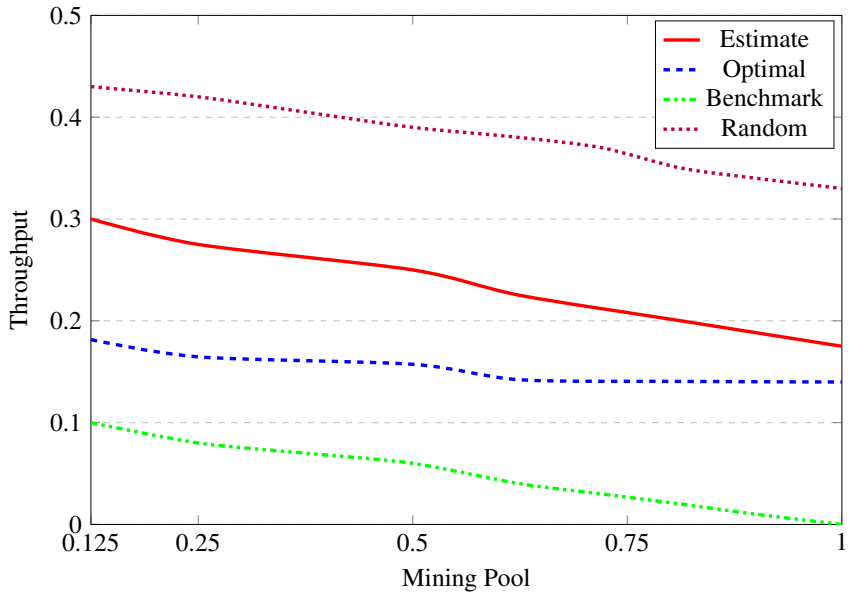


Figure 5.7: The throughput of low priority transactions as the number of transactions within the mining pool increased.

ority transactions. In the benchmark scenario, high priority transactions had a much larger throughput value when compared to the proposed schemes. This is primarily due to the miner simply processing high priority transactions first and then low priority transactions if there is room available within the current block. In the random scenario, the miner randomly picks transactions for publication within the new block. Since a larger number of high priority transactions is generated, the throughput of high priority transactions is larger during the random scenario as well. Due to the industrial setting, the value added by each transaction is important for the overall operation of the network. The value added to the network by using the aforementioned schemes is illustrated in Fig. 5.8, as the number of transactions within the transaction pool increased. The figure shows the expected value of published blocks in comparison to the optimal and estimated schemes. The expected value shown in Fig. 5.8 is the largest due to the block containing primarily high priority transactions. In contrast to the expected value, the optimal formulation produced blocks with slightly lower values due to the inclusion of low priority transactions. The low priority transactions decreased the value of the transaction blocks by approximately 7.45% subsequent to employing the optimal formulation. Likewise,

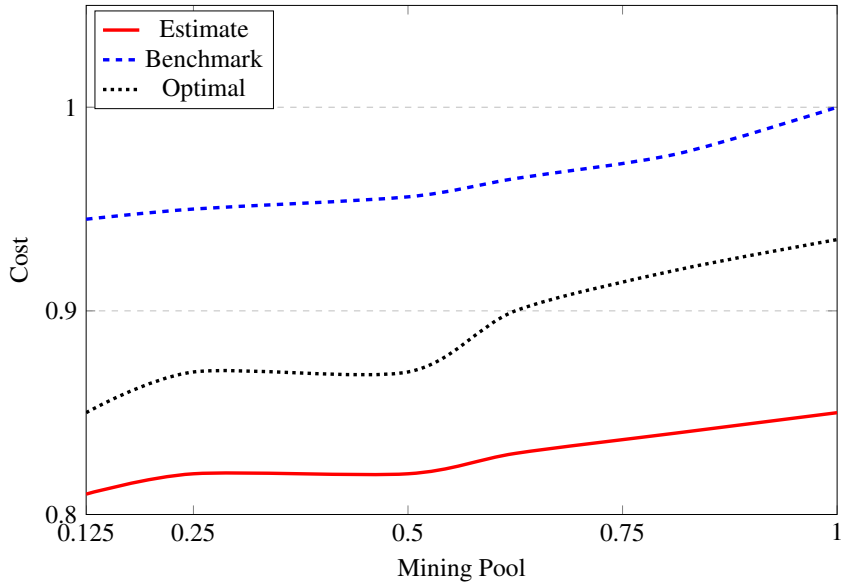


Figure 5.8: Each transaction published is necessary for the operation of the industrial setting and therefore adds a value to the network when published.

the real-time estimate decreased the block value by approximately 13.96% when compared to the expected block value. The decrease in the block value subsequent to using the real-time estimation is due to the inclusion of low priority transactions with lower value added to the network. Therefore, using the proposed scheme will decrease the value added by each block of transactions to the network, however the throughput of low priority transactions will increase. Furthermore, the proposed scheme will end the starvation of low priority transactions empowered by the abundance high priority transactions by essentially guaranteeing publication using the aforementioned block slicing scheme.

5.4.2 NetEM and WLAN-based Implementation of Feeless Mining

To validate the proposed scheme as well as the simulation results, a proof-of-concept implementation is completed with an application that does not depend on digital currencies. A number of devices are created using LXC [161] virtualization container technology as well as portable temperature sensing devices running a Linux based operating system. The portable sensors are Raspberry Pi (RPi) [167] devices with a DS18B20 [168] temperature sensor con-

nected through the RPi's General Purpose Input and Output (GPIO) pins as shown in Fig. 5.9. A blockchain miner is running on dedicated machine where the miner received transactions generated by the LXC machines as well as the portable temperature sensor devices. The LXC machines and miner are operating simultaneously within the same host device. To create an industrial environment between the miner and LXC machines, the traffic generated by the virtual machines is tunneled through a NetEM [166] emulated channel with a 1ms delay and 1% packet loss rate. The portable temperature sensing devices are placed randomly at distances that are at least 10 meters or more from the miner.

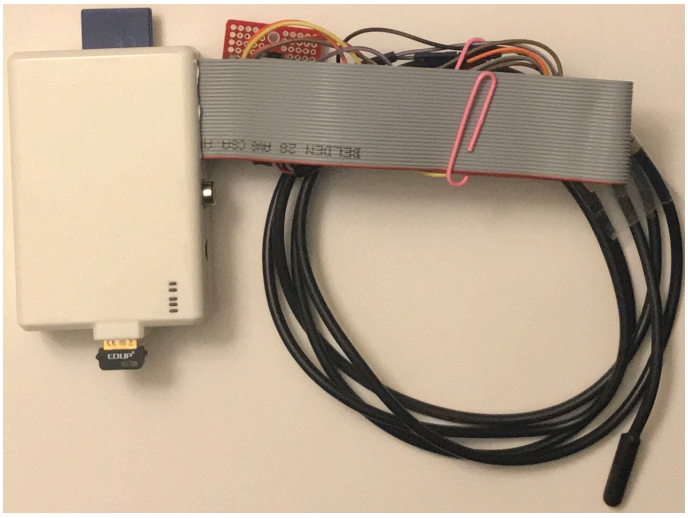


Figure 5.9: The temperature sensor is connected to the RPi through the on-chip GPIO.

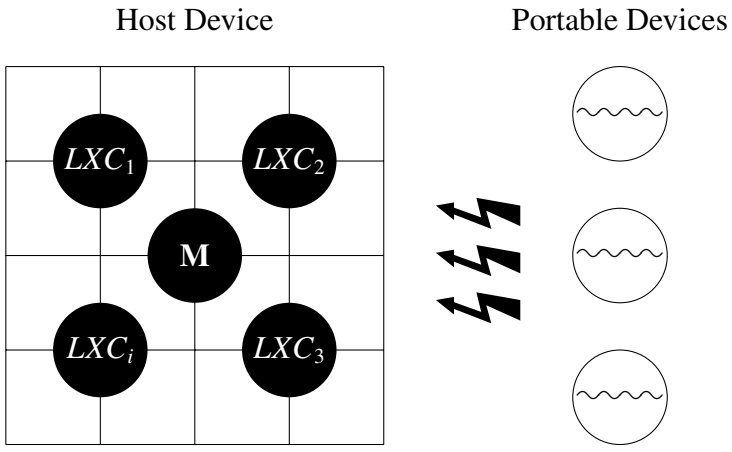


Figure 5.10: The miner and the virtual nodes shared a host device and exchanged data over a NetEM [166] simulated channel.

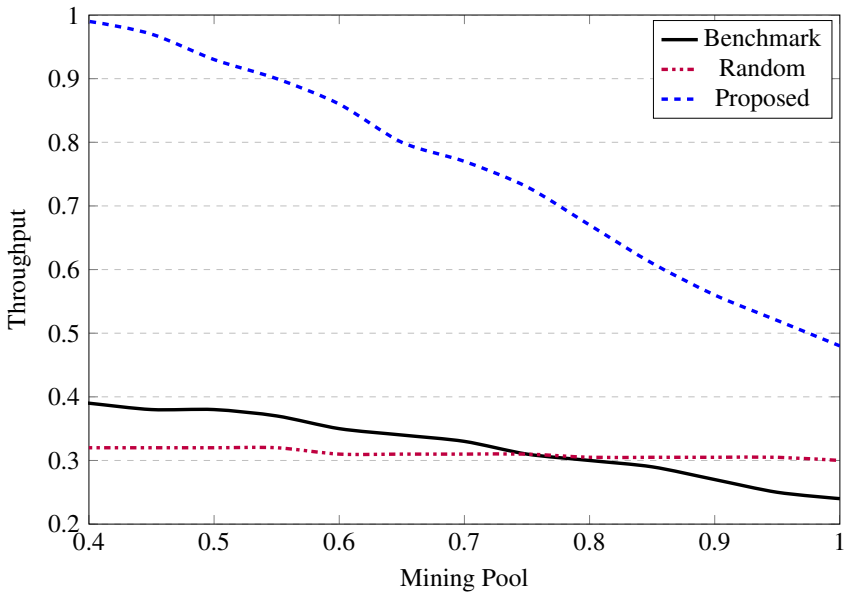


Figure 5.11: The low priority transaction throughput increased when compared to the benchmark values.

The portable sensors sent their transactions to the miner directly using an ad-hoc connection established over WiFi running the IEEE 802.11b standard as illustrated by Fig. 5.10. Not all of the portable sensors are placed in Line-of-Sight (LOS) from the miner. As the devices sensed temperature values, transactions containing the observed temperature values are created and sent to the miner. Each created transaction is randomly assigned a high or low priority status where the node is designed to generate more of the former. Nodes are set to generate a larger number of high priority transactions to mimic low priority transaction starvation, thereby successfully testing the purpose of the proposed scheme. The virtual nodes created using the LXC containers, generated random temperature values where the priority of created transactions is assigned in a similar manner as previously discussed. While low priority transactions are starved, the observed throughput of said transactions followed a similar trend of the results obtained in the previously discussed simulation results section. As illustrated by Fig. 5.11, the observed implementation results agree with the obtained simulation results, where the proposed scheme successfully increases the throughput of low priority transactions. Furthermore, as the number of transactions in the mining pool increases, the throughput gains decrease as well. From the implementation, the throughput of low priority transactions increased by ap-

proximately 23.22%, however the value of each block generated, on average, decreased by 19.05%.

5.5 Summary

Blockchains enable multimedia services to grow to a large size through their decentralized design as well as cryptographic properties. In a blockchain-based multimedia system, all transactions are recorded on a ledger by a set of miners dedicated to continuously perform a maintenance process regardless of the outcome of previous processes. Blockchain-based multimedia systems enable a secure decentralized audiovisual content access with a traceable watermarking system that is suitable for combating content piracy and unauthorized content access. Due to the reliance of large multimedia systems on blockchains, network access throughput is reduced as transaction processing latency increases due to the blockchain's security properties. Processing latencies are mitigated through mining disciplines carried out by ledger maintainers to move certain transactions at varying throughputs. Existing mining disciplines are geared towards financially-driven blockchain applications, such as cryptocurrencies, and therefore are not suitable for industrial blockchain-based services, such as multimedia based content access and rights management systems. In this chapter, a novel feeless mining discipline scheme is designed to help miners generate transaction blocks efficiently. The proposed scheme overcomes low priority transaction starvation using a given utility function designed to select transactions from a mining pool to be published in future blocks. The utility function minimizes the overall priority of transactions within the block while making sure that the operational value added by said block is sufficient for the source process. A real-time process is given to overcome the computational complexity of the aforementioned utility function. The real-time process utilizes ageing to increase the priority of transactions previously residing in mining pools. Furthermore, a transaction block is sliced into compartments to hold a number of transactions from a received priority class. Subsequent to determining the size of slices, transactions are

selected from the mining pool with selection priority given to aged transactions first. Elaborate simulations are conducted to prove the performance of the proposed mining discipline in feeless blockchain-based systems, such as multimedia networks. The simulations show that the processing throughput of a priority class can be improved using the block slicing algorithm. Furthermore, the simulations demonstrate that the proposed solution can be utilized to prevent low priority transaction starvation and subsequently used to determine the publishing throughput of transaction from different priority classes. Furthermore, a proof-of-concept implementation is completed with a non-digital currency based application to demonstrate the playability of the proposed solution. The implemented nodes are built and used to sense temperatures of office spaces that are subsequently shared with dedicated miners in the form of blockchain transactions. Each node is given a priority level and generated transactions with data regarding the sensed the temperature. To reproduce low priority transaction starvation, LXC virtual machines are used to simulate temperature sensing nodes. The LXC containers are connected to the miner over simulated wireless channels.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

Increasing the efficiency of multimedia streaming applications is a challenge due to the diverse nature of audiovisual traffic, the non-deterministic behaviour of network paths, as well as the ever-increasing demand. The challenge is further exacerbated by the best-effort guarantees provided by the Internet Protocol (IP) stack as well as the one-size-fits-all approach for delivering data. Existing multimedia signalling overhead reduction mechanisms and blockchain-based content management solutions attempt to resolve the challenges of multimedia streaming, however they are not adequate as multimedia fidelity and services increase in capacity. This dissertation investigates multimedia streaming traffic and services to: 1) reduce the amount of signalling overhead generated through a steganographic-based approach, 2) decrease the size of exchanged segments by utilizing a steganographic-based protocol translator as well as a Simplified Routing (SR) scheme, and 3) provide a block slicing algorithm for decreasing transaction mining pool residence time thereby increasing processing throughput in decentralized multimedia content rights management services.

Chapter 2 details the fundamentals of adaptive multimedia streaming technologies, including the size and role of exchanged signalling overhead, the most popular multimedia encoding format, Scalable Media Coding (SVC) formats, and data hiding in multimedia mediums. A literature survey of existing signalling overhead reduction techniques available at each layer of

the TCP/IP stack is given in addition to insights of their limitations. As multimedia applications grow to a large size and service an equally sized audience, blockchains are utilized to enable a scalable multimedia content access management system. The existing works on blockchains utilized in multimedia content access management are discussed to address combating multimedia piracy and giving access to legitimate users who paid the multimedia licensing fees.

To alleviate source-destination pairs exchanging multimedia bi-directionally, a novel steganographic-based signalling overhead reduction technique is designed. To identify messages destined for embedding, Chapter 3 presents the sources responsible for generating control overhead in terms of request-response transactions as well as HTTP requests. The proposed technique leverages steganography to create hidden signalling data traffic flows, thereby alleviating resource consumption within source and intermediate nodes. Additionally, the scheme is a cross-layer solution where nodes embed signalling messages within multimedia payloads while ensuring low visual artifacts. To minimize the observed visual artifacts, a utility function is given to evaluate the optimal cost savings while maintaining a tolerable level of visual artifacts. To enable a multi-protocol approach, where signalling overhead from any application is embeddable within a multimedia payload, a unique control message identification system is designed. The identification system assigns unique codes to identify different services and their respective signalling messages, thereby allowing receiving nodes to decode a hidden signalling message correctly. Source-destination pairs, as well as intermediate nodes, benefit from a reduced number of exchanged non-payload messages as fewer resources are allocated towards overhead exchange; as a result, the efficiency of the communication stream increases. The results obtained through simulations, as well as the proof-of-concept implementation, validate the proposed scheme's ability in decreasing the amount of resources consumed by control overhead flows while increasing the network's efficiency through increased observed throughput values.

Using the aforementioned steganographic technology, a novel covert channel-based protocol translator for multimedia streams is proposed in Chapter 4. The solution is designed to reduce packet headers on a packet-by-packet basis and as a result, increasing the efficiency of

multimedia streams. The protocol translator embeds important TCP header information within multimedia payloads while replacing the existing TCP header with a UDP header. The translator enables nodes to maintain the properties of TCP streams at the end-nodes while using UDP during flight for its reduced header size. The source-destination pairs are capable of converting packets from HTTP/TCP to HTTP/UDP and vice-versa during a multimedia exchange session. Furthermore, in the specific scenario of Device-to-Device (D2D) communications, a Simplified Routing (SR) scheme is designed to further reduce the size of packets. The SR scheme replaces inflated networking layer headers with a simplified smaller sized header due to the inherent limitation of D2D communications. A utility function is designed to evaluate the optimal cost savings while maintaining a tolerable level of visual artifacts. The extensive simulations and implementation show that the proposed methodologies will systematically decrease the amount of overhead exchanged while increasing the overall network capacity. Furthermore, the simulation results demonstrate that the proposed methods are capable of successfully extending existing overhead reduction methods.

Although the aforementioned signalling overhead reduction solutions provision bandwidth by decreasing the resource consumption of non-payload data, system throughput management is needed to enhance blockchain-based systems. As multimedia systems increase in popularity, blockchains will enable secure and decentralized deployments. Due to the security requirements of blockchain-based systems and the lack of efficient transaction processing solutions for multimedia-based systems, transaction processing throughput diminishes the system throughput prompting a decrease in QoS and QoE. In chapter 5, the use of blockchain systems in large feeless applications, such as multimedia content management and access services, is addressed where an efficient mining scheme is proposed. The proposed steganographic-based solutions can harmoniously exist with the mining schemes to ensure smooth and uninterrupted playback. Blockchains are used in large-sized systems due to their aforementioned scalability and security features; however, as a result, suffer from reducing processing throughput. The reduced throughput is an ongoing challenge that is very well addressed in blockchain-based digital cur-

rency applications where transactions are capable of paying ledger maintainers higher fees for faster processing times. The existing mining disciplines are therefore geared towards systems utilizing blockchains in digital currency applications and as a result, are inappropriate for industrial applications, such as the aforementioned multimedia services. The proposed mining scheme introduces the notion of priority in exchanged transactions where miners slice outgoing blocks according to the different priority classes of transactions found in the mining pool. As transaction age in the mining pool, an ageing process is introduced to promote transactions waiting for a longer period of time in the mining pool as the next transaction to be selected in the block formation process. The scheme is validated through elaborate simulations that are conducted to demonstrate the performance of the proposed mining discipline. The simulations focus on preventing low priority transaction starvation and subsequently used to determine the publishing throughput of transactions from different priority classes. Furthermore, a proof-of-concept implementation is completed where nodes are built and used to exchange transactions of different priority classes.

6.2 Future Work

There are several topics related to the research presented that need additional investigating and further studies. Some of the topics include:

- To reduce the effects of embedding TCP headers and control messages, recovery bits should be exchanged or a codebook should be used to reduce visual artifacts. An alternative to recovery bit exchange and special codebooks would be using header field values that are currently unassigned or undefined to carry the recovery bits. Finally, the values in header fields can be changed to odd or even as necessary to signal an additional layer of recovery bits.

- In live multimedia streams, such as broadcasting a live soccer match, requires real-time production prior to user delivery. The proposed steganographic-based solutions can be

extended to service live production networks to increase the network's production window time. Reducing signalling overhead exchanged as well as average packet sizes increases the efficiency of live production networks through decreasing the ratio of overhead to payload sent.

- Blockchains are inherently scalable and decentralized and therefore are capable of extending the proposed steganographic methods. Using blockchains, the visual artifacts induced by the proposed embedding process can be decreased through identifying the affected parts. Recovery bits are then exchanged between ledger maintainers and requesting nodes to reduce the observed artifacts. The reduction in observed artifacts will facilitate maintaining desired QoS and QoE for a large number of service consumers.

Bibliography

[1] Cisco, “The Zettabyte Era: Trends and Analysis”, San Jose, CA, USA, White Paper, June. 2017.

[2] C. Koh, “Next-Generation Techniques to Protect and Secure Realtime IP Media Transport”, *SMPTE Mot. Imag. J, vol. 122, no. 5*, pp. 32–38, 2013.

[3] T. Stockhammer and I. Sodagar, “MPEG DASH: The Enabler Standard for Video Delivery over the Internet,” *SMPTE Motion Imaging Journal*, vol. 121, no. 5, pp. 40–46, 2012.

[4] M. Liu, F. Yu, Y. Teng, V. Leung and M. Song, “Distributed Resource Allocation in Blockchain-Based Video Streaming Systems With Mobile Edge Computing”, *IEEE Transactions on Wireless Communications*, vol. 18, no. 1, pp. 695–708, 2019.

[5] A. Begen, T. Akgul and M. Baugher, “Watching Video over the Web: Part 1: Streaming Protocols,” *IEEE Internet Computing*, vol. 15, no. 2, pp. 54–63, 2011.

[6] A. Begen, T. Akgul and M. Baugher, “Watching Video over the Web: Part 2: Applications, Standardization, and Open Issues,” *IEEE Internet Computing*, vol. 15, no. 3, pp. 59–63, 2011.

[7] H. Riiser, P. Halvorsen, C. Griwodz, and D. Johansen, “Low overhead container format for adaptive streaming,” in *Proceedings of the 2010 ACM Multimedia Systems*, pp. 193–198, Feb. 2010.

[8] J. van der Hooft, S. Petrangeli, N. Bouten, T. Wauters, R. Huysegems, T. Bostoen, and F. De Turck, “An HTTP/2 push-based approach for SVC adaptive streaming,” *IEEE/IFIP Network Operations and Management Symposium*, pp. 104–111, Apr. 2016.

[9] J. van der Hooft et al., “HTTP/2-Based Adaptive Streaming of HEVC Video Over 4G/LTE Networks,” *IEEE Communications Letters*, vol. 20, no. 11, pp. 2177–2180, 2016.

[10] S. Wei and V. Swaminathan, “Cost Effective Video Streaming Using Server Push over HTTP 2.0,” *2014 IEEE 16th International Workshop on Multimedia Signal Processing (MMSP)*, pp. 1–5, 2014.

[11] S. Wei, V. Swaminathan, and M. Xiao, “Power efficient mobile video streaming using http/2 server push,” *2015 IEEE 17th International Workshop on Multimedia Signal Processing (MMSP)*, pp. 1–6, Oct. 2015.

[12] S. Wei and V. Swaminathan. "Low latency live video streaming over http 2.0," In Proceedings of *Network and Operating System Support on Digital Audio and Video Workshop*, page 37. ACM, 2014.

[13] M. Xiao, V. Swaminathan, S. Wei, and S. Chen, "Evaluating and improving push based video streaming with HTTP/2," in Proceedings of the *26th International Workshop on Network and Operating Systems Support for Digital Audio and Video*, pp. 1–6, 2016.

[14] R. Huysegems *et al*., "HTTP/2-based methods to improve the live experience of adaptive streaming," Proceedings of the *23rd ACM International Conference on Multimedia MM '15*, ACM, pp. 541–550, 2015.

[15] H. Le, T. Nguyen, N. Ngoc, A. Pham and T. Thang, "HTTP/2 Push-Based Low-Delay Live Streaming Over Mobile Networks With Stream Termination," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 9, pp. 2423-2427, 2018.

[16] A. M. Al-Jubari, M. Othman, B. M. Ali, and N. A. W. A. Hamid, "An adaptive delayed acknowledgment strategy to improve TCP performance in multi-hop wireless networks," *Wireless Personal Communications*, vol. 69, no. 1, pp. 307–333, Mar. 2013.

[17] S. Landstrm and L.-A. Larzon, "Reducing the TCP acknowledgment frequency," *SIGCOMM Computer Communication Review*, vol. 37, no. 3, pp. 5–16, 2007.

[18] M. Allman, "On the Generation and Use of TCP Acknowledgments," *ACM SIGCOMM Computer Communication Review*, pp. 4–21, Oct. 1998.

[19] S. Floyd and A. Romanow, "TCP selective acknowledgement options," *RFC 2018*, Oct. 1996.

[20] K. Sandlund, G. Pelletier Ericsson, and L-E. Jonsson, "The RObust Header Compression (ROHC) Framework," *RFC 5795*, March 2010.

[21] M. Degermark, B. Nordgren, and S. Pink, "IP Header Compression," *RFC 2507*, Feb. 1999.

[22] S. Casner, and V. Jacobson, "Compressing IP/UDP/RTP headers for low-speed serial links," *RFC 2508*, Feb. 1999.

[23] T. Koren, S. Casner, J. Geevarghese, B. Thompson, and P. Ruddy, "Enhanced Compressed RTP (CRTP) for Links with High Delay, Packet Loss and Reordering," *RFC 3545*, July 2003.

[24] B. Thompson, T. Koren, and D. Wing, "Tunneling Multiplexed Compressed RTP (TCRTP)," *RFC 4170*, November 2005.

[25] S. Floyd *et al*., "Datagram congestion control protocol (DCCP)," *Internet Eng. Task Force*, RFC 4340, Mar. 2006.

[26] M. Handley *et al*., "TCP Extensions for Multipath Operation with Multiple Addresses," *Internet Eng. Task Force*, RFC 6824, Jan. 2013.

[27] A. Tanenbaum and D. Wetherall, *Computer Networks*, Harlow: Pearson Education Limited, 2014, pp. 226.

[28] A. Yasin and L. Liu, "An online identity and smart contract management system," in *2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC)*, vol. 2, June 2016, pp. 192–198.

[29] G. Zyskind, O. Nathan, and A. Pentland. "Decentralizing privacy: Using blockchain to protect personal data," In *2015 IEEE Security and Privacy Workshops*, pp. 180–184, May 2015.

[30] R. Xu, L. Zhang, H. Zhao, Y. Peng, "Design of Network Media's Digital Rights Management Scheme Based on Blockchain Technology," *IEEE 13th International Symposium on Autonomous Decentralized Systems*, 2017, pp. 128–133.

[31] Z. Ma, M. Jiang, H. Gao and Z. Wang, "Blockchain for digital rights management," *Future Generation Computer Systems*, vol. 89, pp. 746–764, 2018.

[32] Z. Ma, W. Huang, W. Bi, H. Gao and Z. Wang, "A master-slave blockchain paradigm and application in digital rights management," *China Communications*, vol. 15, no. 8, pp. 174–188, 2018.

[33] H. Zhao, Y. LiuE, Y. Wang, X. Wang and J. Li, "A Blockchain-Based Data Hiding Method for Data Protection in Digital Video," in *SmartBlock 2018 Lecture Notes in Computer Science*, volume 11373, Tokyo, Japan, 2018, pp. 99–110.

[34] D. Bhowik, T. Feng, "The Multimedia Blockchain: A Distributed and Tamper-Proof Media Transaction Framework," in *22nd International Conference on Digital Signal Processing (DSP)*, 2017, pp. 1–5.

[35] Z. Ma, W. Huang and H. Gao, "Secure DRM Scheme Based on Blockchain with High Credibility," *Chinese Journal of Electronics*, vol. 27, no. 5, pp. 1025–1036, 2018.

[36] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hobfeld and P. Tran-Gia, "A Survey on Quality of Experience of HTTP Adaptive Streaming," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 469–492, 2015.

[37] W. Gao, W. G. Hatcher, W. Yu, "A survey of blockchain: Techniques applications and challenges," in *Proc. 27th Int. Conf. Comput. Commun. Netw. (ICCCN)*, pp. 1–11, Aug. 2018.

[38] Y. Chen, K. Wu and Q. Zhang, "From QoS to QoE: A Tutorial on Video Quality Assessment," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 2, pp. 1126–1165, 2015.

[39] Y. Sani, A. Mauthe and C. Edwards, "Adaptive Bitrate Selection: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2985–3014, 2017.

[40] K. Rudman *et al*., "Toward Real-Time Detection of Forensic Watermarks to Combat Piracy by Live Streaming," *SMPTE Motion Imaging Journal*, vol. 125, no. 1, pp. 34–41, 2016.

[41] G. C. Polyzos and N. Fotiou, "Blockchain-assisted information distribution for the Internet of Things," in *2017 IEEE International Conference on Information Reuse and Integration (IRI)*, pp. 75–78, 2017.

[42] A. Dorri, S. S. Kanhere, and R. Jurdak, "Blockchain in internet of things: Challenges and solutions," *arXiv:1608.05187*, 2016.

[43] A. Dorri et al., "Blockchain for IoT security and privacy: The case study of a smart home," *IEEE International Conference on Pervasive Computing and Communications Workshops*, pp. 618–623, 2017.

[44] A. Dorri, S.S. Kanhere, R. Jurdak, "Towards an Optimized BlockChain for IoT," in *Proc. 2nd Int'l. Conf. Internet-of-Things Design and Implementation*, pp. 173–178, 2017.

[45] L. Zhou, L. Wang, Y. Sun and P. Lv, "BeeKeeper: A Blockchain-Based IoT System With Secure Storage and Homomorphic Computation," *IEEE Access*, vol. 6, pp. 43472–43488, 2018.

[46] D. Easley, M. O'Hara and S. Basu, "From Mining to Markets: the Evolution of Bitcoin Transaction Fees," *SSRN Electronic Journal*, 2017.

[47] M. Moser, R. Bohme, "Trends Tips Tolls: A Longitudinal Study of Bitcoin Transaction Fees," *Workshop on Bitcoin Research*, 2015.

[48] J. Pan, J. Wang, A. Hester, I. Alqerm, Y. Liu and Y. Zhao, "EdgeChain: An Edge-IoT Framework and Prototype Based on Blockchain and Smart Contracts," *IEEE Internet of Things Journal*, pp. 1–15, 2018.

[49] K. Kaskaloglu, "Near zero bitcoin transaction fees cannot last forever," in *Proc. Int. Conf. Digit. Secur. Forensics*, pp. 91–99, Jun. 2014.

[50] R. Lavi, O. Sattath and A. Zohar, "Redesigning Bitcoin's fee market," *arXiv e-prints: Computer Science - Cryptography and Security*, pp. 1–34, 2017.

[51] J. A. Kroll, I. C. Davey, E. W. Felten, "The economics of Bitcoin mining or Bitcoin in the presence of adversaries," *WEIS*, 2013.

[52] G. Huberman, J. Leshno and C. Moallemi, "Monopoly Without a Monopolist: An Economic Analysis of the Bitcoin Payment System," *SSRN Electronic Journal*, 2017.

[53] J. Li, Y. Yuan, S. Wang and F. Wang, "Transaction Queuing Game in Bitcoin BlockChain," in *IEEE Intelligent Vehicles Symposium*, China, 2018, pp. 1–6.

[54] "The Next Generation of Distributed Ledger Technology — IOTA," Iota.org, 2018. [Online]. Available: https://www.iota.org/. [Accessed: 18- Dec- 2018].

[55] "Xapo - Bitcoin Wallet & Vault," Xapo, 2018. [Online]. Available: https://xapo.com/. [Accessed: 18- Dec- 2018].

[56] "Blockchain - Most Trusted Crypto Company," Blockchain.com, 2018. [Online]. Available: https://www.blockchain.com/. [Accessed: 18- Dec- 2018].

[57] S. Huh, S. Cho, and S. Kim, "Managing IoT Devices Using Blockchain Platform," in *Proc. 2017 19th Intl Conf. on Advanced Communication Technology (ICACT)*, pp. 464–467, 2017.

[58] M. Conoscenti, A. Vetr, and J. C. D. Martin, "Blockchain for the Internet of Things: A systematic literature review," in *Proc. 13th Int. Symp. Internet Things Syst. Manag. Security (IOTSMS)*, 2016, pp. 1–6.

[59] O. Novo, "Blockchain Meets IoT: An Architecture for Scalable Access Management in IoT," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1184–1195, 2018.

[60] N. Kshetri, "Can Blockchain Strengthen the Internet of Things?," *IT Professional*, vol. 19, no. 4, pp. 68–72, 2017.

[61] D. Marpe, T. Wiegand and G. Sullivan, "The H.264/MPEG4 advanced video coding standard and its applications," *IEEE Communications Magazine*, vol. 44, no. 8, pp. 134-143, 2006.

[62] I. Richardson, *The h.264 advanced video compression standard*, Hoboken, N.J.: Wiley, 2013, pp. 81–244.

[63] E. Magli, M. Wang, P. Frossard and A. Markopoulou, "Network coding meets multimedia: A review," *IEEE Trans. Multimedia*, vol. 15, no. 5, pp. 1195–1212, 2013.

[64] Jiangchuan Liu, S. Rao, Bo Li and Hui Zhang, "Opportunities and Challenges of Peer-to-Peer Internet Video Broadcast," in *Proceedings of the IEEE*, vol. 96, no. 1, pp. 11–24, 2008.

[65] Y. Huo, C. Hellge, T. Wiegand and L. Hanzo, "A Tutorial and Review on Inter-Layer FEC Coded Layered Video Streaming," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 2, pp. 1166–1207, 2015.

[66] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hobfeld and P. Tran-Gia, "A Survey on Quality of Experience of HTTP Adaptive Streaming," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 469–492, 2015.

[67] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications," *IETF*, Fremont, CA, USA, RFC 3550, Jul. 2003. [Online]. Available:https://tools.ietf.org/html/rfc3550.

[68] H. Schulzrinne, A. Rao, and R. Lanphier, M. Westerlund, and M. Stiemerling, Ed., "Real time streaming protocol version 2.0," *IETF*, Fremont, CA, USA, RFC 7826, Dec. 2016. [Online]. Available: https://tools.ietf.org/html/rfc7826.

[69] K. Bagci, K. Sahin and A. Tekalp, "Compete or Collaborate: Architectures for Collaborative DASH Video Over Future Networks," *IEEE Transactions on Multimedia*, vol. 19, no. 10, pp. 2152–2165, 2017.

[70] I. Sodagar, "The MPEG-DASH Standard for Multimedia Streaming Over the Internet," *IEEE Multimedia*, vol. 18, no. 4, pp. 62–67, 2011.

[71] S. Varma, "Flow control in video applications," in *Internet Congestion Control*, Waltham, MA, USA: Morgan Kaufmann, 2015, ch. 6,pp. 173–202.

[72] B. Li, Z. Wang, J. Liu and W. Zhu, "Two decades of internet video streaming," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 9, no. 1, pp. 1–20, 2013.

[73] K. Rudman, M. Bonenfant, M. Celik, J. Daniel, J. Haitsma and J. Panis, "Toward Real-Time Detection of Forensic Watermarks to Combat Piracy by Live Streaming," *SMPTE Motion Imaging Journal*, vol. 125, no. 1, pp. 34–41, 2016.

[74] S. Gao, M. Tao, "Joint multicast scheduling and user association for dash-based video streaming over heterogeneous cellular networks," *IEEE/CIC International Conference on Communications in China*, pp. 1–6, 2016.

[75] J. Park, J. Hwang, Q. Li, Y. Xu and W. Huang, "Optimal DASH-Multicasting Over LTE," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 5, pp. 4487–4500, 2018.

[76] Cable Television Laboratories, "IP Multicast Adaptive Bit Rate Architecture Technical," 2016.

[77] K. Kim, S. Mehrotra and N. Venkatasubramanian, "Efficient and Reliable Application Layer Multicast for Flash Dissemination," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 10, pp. 2571–2582, 2014.

[78] R. Ahlswede, N. Cai, S. Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inform. Theory*, vol. 46, pp. 1204–1216, 2000.

[79] E. Magli, *et al.*, "Network coding meets multimedia: A review," *IEEE Trans. Multimedia*, vol. 15, no. 5, pp. 1195–1212, 2013.

[80] Z. Liu, C. Wu, B. Li and S. Zhao,"Uusee: Large-scale operational on-demand streaming with random network coding," *Proc. IEEE INFOCOM*, pp. 1–9, 2010.

[81] M. Wang and B. Li, "Random push with random network coding in live peer-to-peer streaming", *IEEE J. Sel. Areas Commun.*, vol. 25, no. 9, pp. 1655–1666, 2007.

[82] L. Natali and M. Merani, "Successfully Mapping DASH Over a P2P Live Streaming Architecture," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 6, pp. 1326–1339, 2017. Available: 10.1109/tcsvt.2016.2539611.

[83] S. Lederer, C. Muller, and C. Timmerer, "Towards peer-assisted dynamic adaptive streaming over HTTP," in *Proc. IEEE 19th Int. Packet Video Workshop (PV)*, May 2012, pp. 161–166.

[84] L. Rovcanin and G.-M. Muntean, "DASH-based performance-oriented adaptive video distribution solution," in *Proc. IEEE Int. Symp. BMSB*, Jun. 2013, pp. 1–7.

[85] G. Tian, Y. Xu, Y. Liu, and K. Ross, "Mechanism design for dynamic P2P streaming," in *Proc. 13th IEEE P2P*, Sep. 2013, pp. 1–10.

[86] Jinyao Yan, W. Muhlbauer and B. Plattner, "Analytical Framework for Improving the Quality of Streaming Over TCP," *IEEE Transactions on Multimedia*, vol. 14, no. 6, pp. 1579–1590, 2012.

[87] E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.3," *RFC 8446*, Aug. 2018.

[88] T. Flach et al., "Reducing web latency: The virtue of gentle aggression," *in Proc. ACM SIGCOMM*, pp. 159–170, 2013.

[89] Q. Yu, W. Meng and S. Lin, "Packet Loss Recovery Scheme with Uniquely-Decodable Codes for Streaming Multimedia over P2P Networks," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 9, pp. 142–154, 2013.

[90] G. Reitmeier and G. Sullivan, "Video Compression and Its Role in the History of Television," *SMPTE Motion Imaging Journal*, vol. 125, no. 6, pp. 60–74, 2016.

[91] J. Wu, B. Cheng, M. Wang and J. Chen, "Priority-Aware FEC Coding for High-Definition Mobile Video Delivery Using TCP," *IEEE Transactions on Mobile Computing*, vol. 16, no. 4, pp. 1090–1106, 2017.

[92] V. Adzic, H. Kalva and B. Furht, "Optimizing video encoding for adaptive streaming over HTTP," *IEEE Transactions on Consumer Electronics*, vol. 58, no. 2, pp. 397–403, 2012.

[93] Q. Yu, W. Meng and S. Lin, "Packet Loss Recovery Scheme with Uniquely-Decodable Codes for Streaming Multimedia over P2P Networks," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 9, pp. 142–154, 2013.

[94] K. Miller, E. Quacchio, G. Gennari, and A. Wolisz, "Adaptation algo- rithm for adaptive streaming over HTTP," in *Proc. 19th Int. PV Workshop*, pp. 173–178, 2012.

[95] J. Fridrich, M. Goljan and Rui Du, "Detecting LSB steganography in color, and gray-scale images," *IEEE MultiMedia*, vol. 8, no. 4, pp. 22–28, 2001.

[96] P. Moulin and R. Koetter, "Data-Hiding Codes," *Proceedings of the IEEE*, vol. 93, no. 12, pp. 2083–2126, 2005.

[97] W. Trappe, Jie Song, R. Poovendran and K. Liu, "Key management and distribution for secure multimedia multicast," *IEEE Transactions on Multimedia*, vol. 5, no. 4, pp. 494–507, 2003.

[98] Y. Lin, C. Wang, W. Chen, F. Lin and W. Lin, "A Novel Data Hiding Algorithm for High Dynamic Range Images," *IEEE Transactions on Multimedia*, vol. 19, no. 1, pp. 196–211, 2017.

[99] T. Handel and M. Sandford,"Hiding Data in the OSI Network Model," *Proc. 1st Int. Workshop. Information Hiding*, pp. 23–38, 1996.

[100] L. Coria, M. Pickering, P. Nasiopoulos and R. Ward, "A Video Watermarking Scheme Based on the Dual-Tree Complex Wavelet Transform," *IEEE Transactions on Information Forensics and Security*, vol. 3, no. 3, pp. 466-474, 2008.

[101] M. Asikuzzaman, M. J. Alam, A. J. Lambert, and M. R. Pickering, "Imperceptible and robust blind video watermarking using chrominance embedding: A set of approaches in the DT CWT domain," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 9, pp. 1502-1517, Sep. 2014.

[102] I. W. Selesnick , R. G. Baraniuk and N. C. Kingsbury, "The dual-tree complex wavelet transform," *IEEE Signal Process Mag.*, vol. 22, no. 6, pp. 123-151, 2005.

[103] S. Priya and P. Amritha, "Information Hiding in H.264, H.265, and MJPEG," *in Proceedings of the International Conference on Soft Computing Systems*, pp. 479–487, 2016.

[104] Y. Liu, Z. Li, X. Ma and J. Liu, "A robust data hiding algorithm for H.264/AVC video streams," *Journal of Systems and Software*, vol. 86, no. 8, pp. 2174–2183, 2013.

[105] Y. Liu, Z. Li, X. Ma and J. Liu, "A robust without intra-frame distortion drift data hiding algorithm based on H.264/AVC," *Multimedia Tools and Applications*, vol. 72, no. 1, pp. 613–636, 2013.

[106] [10]D. Nguyen, T. Nguyen, C. Chang, H. Hsueh and F. Hsu, "High Embedding Capacity Data Hiding Algorithm for H.264/AVC Video Sequences without Intraframe Distortion Drift," *Security and Communication Networks*, vol. 2018, pp. 1–11, 2018.

[107] S. Kapotas, E. Varsaki and A. Skodras, "Data Hiding in H. 264 Encoded Video Sequences," in *IEEE 9th Workshop on Multimedia Signal Processing*, Crete, Greece, 2007, pp. 1–4.

[108] S. Bouchama, H. Aliane, and L. Hamami, "Reversible data hiding scheme for the H. 264/AVC codec," *IEEE International Conference on Information Science and Applications*, pp. 1–4, 2013.

[109] Y. Chen, H. Wang, H. Wu and Y. Liu, "Reversible Video Data Hiding Using Zero QDCT Coefficient-Pairs," in *Multimedia, arXiv:1804.06628*, 2019, pp. 1–11.

[110] C. Di Laura, D. Pajuelo and G. Kemper, "A Novel Steganography Technique for SDTV-H.264/AVC Encoded Video," *International Journal of Digital Multimedia Broadcasting*, vol. 2016, pp. 1–9, 2016.

[111] W. Trappe, Jie Song, R. Poovendran and K. Liu, "Key management and distribution for secure multimedia multicast," *IEEE Transactions on Multimedia*, vol. 5, no. 4, pp. 494–507, 2003.

[112] H. Zhao, Y. Q. Shi, and N. Ansari, "Hiding data in multimedia streaming over networks", *in Communication Networks and Services Research Conference (CNSR), 2010 Eighth Annual*, pp. 50–55, 2010.

[113] H. Zhao, Y. Q. Shi. and N. Ansari, "Steganography in Streaming Multimedia over Networks", *Transactions on Data Hiding and Multimedia Security VII . Lecture Notes in Computer Science Volume 7110*, pp. 96–114, 2012.

[114] B. Hahn, R. Nithyanand, P. Gill, R. Johnson, "Games without frontiers: Investigating video games as a covert channel", *IEEE European Symp. on Security and Privacy (Euro S& P)*, 2016.

[115] P. Vines and T. Kohno, "Rook: Using video games as a low-bandwidth censorship resistant communication platform," *in Proceedings of the 14th ACM Workshop on Privacy in the Electronic Society, WPES 15, (New York, NY, USA)*, pp. 75–84, 2015.

[116] R. Henry, A. Herzberg and A. Kate, "Blockchain Access Privacy: Challenges and Directions," *IEEE Security & Privacy*, vol. 16, no. 4, pp. 38–45, 2018.

[117] T. Aste, P. Tasca and T. Di Matteo, "Blockchain Technologies: The Foreseeable Impact on Society and Industry," *Computer*, vol. 50, no. 9, pp. 18–28, 2017.

[118] J. Padhye et al., "Modeling TCP throughput," *ACM SIGCOMM Computer Communication Review, vol. 28, no. 4*, pp. 303–314, 1998.

[119] T. Zhang, J. Wang, J. Huang, J. Chen, Y. Pan and G. Min, "Tuning the Aggressive TCP Behavior for Highly Concurrent HTTP Connections in Intra-Datacenter," *IEEE/ACM Transactions on Networking*, vol. 25, no. 6, pp. 3808–3822, 2017.

[120] M. Hosseini, A. Wang, and R. Etesami, "Towards energy-aware DASH for mobile video," in *Proc. 7th ACM Int. Workshop Mobile Video (MoVid)*, Portland, OR, USA, 2015, pp. 7–8.

[121] T. Ma, M. Hempel, D. Peng, and H. Sharif, "A Survey of Energy-Efficient Compression and Communication Techniques for Multimedia in Resource Constrained Systems," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 3, pp. 963–972, 2013.

[122] H. ZainEldin, M. Elhosseini, and H. Ali, "Image compression algorithms in wireless multimedia sensor networks: A survey," *Ain Shams Engineering Journal*, vol. 6, no. 2, pp. 481–490, 2015.

[123] A. AlWatyan, W. Mater, O. Almutairi, A. Al-Noori, and S. Abed, "Security Approach for LSB Steganography Based FPGA Implementation," in *2017 7th International Conference on Modeling, Simulation, and Applied Optimization (ICMSAO)*, Sharjah, United Arab Emirates, 2017, pp. 1–5.

[124] P. Moulin and R. Koetter, "Data-Hiding Codes," *Proceedings of the IEEE*, vol. 93, no. 12, pp. 2083–2126, 2005.

[125] X. Chen, B. Proulx, X. Gong and J. Zhang, "Exploiting Social Ties for Cooperative D2D Communications: A Mobile Social Networking Case," *IEEE/ACM Transactions on Networking*, vol. 23, no. 5, pp. 1471–1484, 2015.

[126] Y. Niu, Y. Liu, Y. Li, X. Chen, Z. Zhong, and Z. Han, "Device-to-Device Communications Enabled Energy Efficient Multicast Scheduling in mmWave Small Cells," *IEEE Transactions on Communications*, vol. 66, no. 3, pp. 1093–1109, 2018.

[127] B. Liu, Y. Cao, W. Wang, and T. Jiang, "Energy Budget Aware Device-to-Device Cooperation for Mobile Videos," in *IEEE Global Communications Conference (GLOBECOM)*, San Diego, CA, USA, 2015, pp. 6–10.

[128] A. Sakr and E. Hossain, "Cognitive and Energy Harvesting-Based D2D Communication in Cellular Networks: Stochastic Geometry Modeling and Analysis," *IEEE Transactions on Communications*, vol. 63, no. 5, pp. 1867–1880, 2015.

[129] J. Li, R. Bhattacharyya, S. Paul, S. Shakkottai and V. Subramanian, "Incentivizing Sharing in Realtime D2D Streaming Networks: A Mean Field Game Perspective," *IEEE/ACM Transactions on Networking*, vol. 25, no. 1, pp. 3–17, 2017.

[130] Y. Keshtkarjahromi, H. Seferoglu, R. Ansari and A. Khokhar, "Device-to-Device Networking Meets Cellular via Network Coding," *IEEE/ACM Transactions on Networking*, vol. 26, no. 1, pp. 370–383, 2018.

[131] V. Gupta, Y. Bejerano, C. Gutterman, J. Ferragut, K. Guo, T. Nandagopal and G. Zussman, "Light-Weight Feedback Mechanism for WiFi Multicast to Very Large GroupsExperimental Evaluation," *IEEE/ACM Transactions on Networking*, vol. 24, no. 6, pp. 3826–3840, 2016.

[132] "Protocol Numbers," Iana.org, 2019. [Online]. Available: http://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml.

[133] Z. Chen and M. Kountouris, "Decentralized Opportunistic Access for D2D Underlaid Cellular Networks," *IEEE Transactions on Communications*, pp. 1–12, 2018.

[134] J. Kim, G. Caire and A. Molisch, "Quality-Aware Streaming and Scheduling for Device-to-Device Video Delivery," *IEEE/ACM Transactions on Networking*, vol. 24, no. 4, pp. 2319–2331, 2016.

[135] L. Deng, C. Wang, M. Chen and S. Zhao, "Timely Wireless Flows With General Traffic Patterns: Capacity Region and Scheduling Algorithms," *IEEE/ACM Transactions on Networking*, vol. 25, no. 6, pp. 3473–3486, 2017.

[136] N. Golrezaei, P. Mansourifard, A. Molisch, and A. Dimakis, "Base-Station Assisted Device-to-Device Communications for High-Throughput Wireless Video Networks," *IEEE Transactions on Wireless Communications*, vol. 13, no. 7, pp. 3665–3676, 2014.

[137] M. Naslcheraghi, S. Ghorashi, and M. Shikh-Bahaei, "FD device-to-device communication for wireless video distribution," *IET Communications*, vol. 11, no. 7, pp. 1074–1081, 2017.

[138] D. Wu, J. Yan, H. Wang, D. Wu, and R. Wang, "Social Attribute Aware Incentive Mechanism for Device-to-Device Video Distribution," *IEEE Transactions on Multimedia*, vol. 19, no. 8, pp. 1908–1920, 2017.

[139] Y. Cao, T. Jiang, X. Chen, and J. Zhang, "Social-Aware Video Multicast Based on Device-to-Device Communications," *IEEE Transactions on Mobile Computing*, vol. 15, no. 6, pp. 1528–1539, 2016.

[140] K. Shanmugam, N. Golrezaei, A. Dimakis, A. Molisch, and G. Caire, "FemtoCaching: Wireless Content Delivery Through Distributed Caching Helpers," *IEEE Transactions on Information Theory*, vol. 59, no. 12, pp. 8402–8413, 2013.

[141] R. Wang, J. Zhang, S. Song, and K. Letaief, "Mobility-Aware Caching in D2D Networks," *IEEE Transactions on Wireless Communications*, vol. 16, no. 8, pp. 5001–5015, 2017.

[142] N. Vo, T. Duong, H. Tuan, and A. Kortun, "Optimal Video Streaming in Dense 5G Networks with D2D Communications," *IEEE Access*, pp. 209–223, 2017.

[143] Y. Shen, C. Jiang, T. Quek, and Y. Ren, "Device-to-Device-Assisted Communications in Cellular Networks: An Energy Efficient Approach in Downlink Video Sharing Scenario," *IEEE Transactions on Wireless Communications*, vol. 15, no. 2, pp. 1575–1587, 2016.

[144] M. Ji, G. Caire, and A. Molisch, "Fundamental Limits of Caching in Wireless D2D Networks," *IEEE Transactions on Information Theory*, vol. 62, no. 2, pp. 849–869, 2016.

[145] M. Chen, L. Wang, and J. Chen, "Users' media cloud assisted D2D communications for distributed caching underlaying cellular network," *China Communications*, vol. 13, no. 8, pp. 13–23, 2016.

[146] M. Ji, G. Caire, and A. Molisch, "Wireless Device-to-Device Caching Networks: Basic Principles and System Performance," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 1, pp. 176–189, 2016.

[147] L. Zhang, M. Xiao, G. Wu, and S. Li, "Efficient Scheduling and Power Allocation for D2D-Assisted Wireless Caching Networks," *IEEE Transactions on Communications*, vol. 64, no. 6, pp. 2438–2452, 2016.

[148] D. Malak, M. Shalash, and J. Andrews, "Optimizing Content Caching to Maximize the Density of Successful Receptions in Device-to-Device Networking," *IEEE Transactions on Communications*, pp. 1–16, 2016.

[149] M. Afshang, H. Dhillon, and P. Chong, "Fundamentals of Cluster-Centric Content Placement in Cache-Enabled Device-to-Device Networks," *IEEE Transactions on Communications*, vol. 64, no. 6, pp. 2511–2526, 2016.

[150] S. Vural, N. Wang, P. Navaratnam and R. Tafazolli, "Caching Transient Data in Internet Content Routers," *IEEE/ACM Transactions on Networking*, vol. 25, no. 2, pp. 1048–1061, 2017.

[151] C. Ng and S. Boon-Hee, *Queueing Modelling Fundamentals*, New York, NY: John Wiley & Sons, 2008, pp. 158–169.

[152] S. Alouf, P. Nain, and D. Towsley, "Inferring network characteristics via moment-based estimators," in IEEE INFOCOM 2001, vol. 2, 2001, pp. 1045–1054

[153] Network simulator, NS-3. [Online]. Available: http://www.nsnam.org/.

[154] J. Currie and D. I. Wilson, "OPTI: Lowering the Barrier Between Open Source Optimizers and the Industrial MATLAB User," *Foundations of Computer-Aided Process Operations*, Georgia, USA, 2012.

[155] R. Addie, I. Zukerman, and T. Neame, "Broadband traffic modeling: simple solutions to hard problems," *IEEE Communications Magazine*, vol. 36, no. 8, pp. 88–95, 1998.

[156] M. Zukerman, T. Neame, and R. Addie, "Internet traffic modeling and future technology implications," *INFOCOM*, vol. 1, pp. 587–596, March 2003.

[157] D. Ammar, T. Begin, and I. Guerin-Lassous, "A new tool for generating realistic Internet traffic in NS-3," *Proceedings of the 4th International ICST Conference on Simulation Tools and Techniques*, pp. 81–83. 2011.

[158] J. Gordon, "Long range correlation in multiplexed Pareto traffic," *Proceedings of the International IEEE/IFIP Conference on Global Infrastructure for the Information Age, Broadband Communications*, pp. 28–39, 1996.

[159] T. Camp, J. Boleng and V. Davies, "A survey of mobility models for ad hoc network research," Wireless Communications and Mobile Computing, vol. 2, no. 5, pp. 483–502, 2002.

[160] S. Lin, P. Wang, I. Akyildiz and M. Luo, "Delay-Based Maximum Power-Weight Scheduling With Heavy-Tailed Traffic," *IEEE/ACM Transactions on Networking*, vol. 25, no. 4, pp. 2540–2555, 2017.

[161] "Linux Containers," Linuxcontainers.org, 2017. [Online]. Available: https://linuxcontainers.org/. [Accessed: 06- Dec- 2017].

[162] "VLC: Official site - Free multimedia solutions for all OS! - VideoLAN," Videolan.org, 2017. [Online]. Available: https://www.videolan.org. [Accessed: Dec. 2017].

[163] "Big Buck Bunny," Peach.blender.org, 2017. [Online]. Available: https://peach.blender.org/. [Accessed: Dec. 2017].

[164] E. Kondoz, *Visual Media Coding and Transmission*, John Wiley & Sons, 2009, pp. 32–33, 346–347.

[165] E. Kondoz, *Visual Media Coding and Transmission*, John Wiley & Sons, 2009, pp. 32–33, 346–347.

[166] "networking:netem [Wiki]," Wiki.linuxfoundation.org, 2018. [Online]. Available: https://wiki.linuxfoundation.org/networking/netem. [Accessed: 13- Nov- 2018].

[167] R. Foundation, "Raspberry Pi Teach, Learn, and Make with Raspberry Pi," Raspberry Pi, 2018. [Online]. Available: https://www.raspberrypi.org/. [Accessed: 13- Nov- 2018].

[168] "DS18B20 Programmable Resolution 1-Wire Digital Thermometer - Maxim," Maximintegrated.com, 2018. [Online]. Available: https://www.maximintegrated.com/en/products/sensors/DS18B20.html. [Accessed: 13- Nov- 2018].

[169] R. Li, T. Song, B. Mei, H. Li, X. Cheng and L. Sun, "Blockchain For Large-Scale Internet of Things Data Storage and Protection," IEEE Transactions on Services Computing, pp. 1–11, 2018.

[170] M. Bedford Taylor, "The Evolution of Bitcoin Hardware," Computer, vol. 50, no. 9, pp. 58–66, 2017.

[171] Z. Xiong, Y. Zhang, D. Niyato, P. Wang and Z. Han, "When Mobile Blockchain Meets Edge Computing," IEEE Communications Magazine, vol. 56, no. 8, pp. 33–39, 2018.

[172] Z. Xiong, S. Feng, D. Niyato, P. Wang, and Z. Han, "Optimal pricing-based edge computing resource management in mobile blockchain," in 2018 IEEE International Conference on Communications (ICC), Kansas City, Kansas, May 2018.

[173] A. Reyna, C. Martn, J. Chen, E. Soler and M. Daz, "On blockchain and its integration with IoT. Challenges and opportunities," Future Generation Computer Systems, vol. 88, pp. 173–190, 2018.

Curriculum Vitae

Name:	Fuad Shamieh
Post-Secondary Education and Degrees:	2009 - 2013, Bachelor of Engineering Science The University of Western Ontario London, Ontario, Canada 2013 - 2015, Master of Engineering Science The University of Western Ontario London, Ontario, Canada 2015 - present, Ph.D. Candidate The University of Western Ontario London, Ontario, Canada
Honours and Awards:	2015, Certificate of Volunteer Appreciation - IEEE London Section. 2017, Best Presentation at the UWO ECE Symposium. 2018, Certificate of Volunteer Appreciation - IEEE Canada. 2018, Certificate of Volunteer Appreciation - IEEE WIE. 2018, Certificate of Outstanding Volunteer Appreciation - IEEE London Section.
Related Work Experience:	Teaching Assistant The University of Western Ontario 2013 - 2019 Research Assistant The University of Western Ontario 2013 - 2019 Visiting Lecturer American University of Madaba July 2015 - Sept. 2015

Publications:

1. **F. Shamieh** and X. Wang, "Transaction Throughput Provisioning Technique for Industrial IoT Blockchain-based Networks," IEEE Internet of Things Journal, 2019 (to be submitted).

2. **F. Shamieh** and X. Wang, "Steganographic-based Header Size Reduction Technique for Multimedia Streams," IEEE/ACM Transactions on Networking, 2019 (received major revision).

3. **F. Shamieh** and X. Wang, "Dynamic Cross-layer Signalling Exchange for Real-time and On-demand Multimedia Streams," IEEE Transactions on Multimedia, 2019.

4. **F. Shamieh** and X. Wang, "Novel lightweight multicasting protocol for thin-client systems," International Wireless Communications and Mobile Computing Conference (IWCMC), Paphos, pp. 553-557, 2016.

5. **F. Shamieh**, A. Akhtar and X. Wang, "Adaptive distributed compression technique utilizing intermediate network nodes," IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), Vancouver, BC, pp. 1-4, 2016.

6. **F. Shamieh**, A. Refaey, and X. Wang, "An adaptive compression technique based on real-time RTT feedback," in Proc. IEEE CCECE 2014.