Evaluating CRC-8 Detection Performance in Cloud Traffic

Vijitha Ananthi J.[1*], Jahnavi S.[2], Sushmitha M.[3], Vasantha M.[4], Bhanu Sasank S.[5]

Vignan's Foundation for Science, Technology & Research, Andhra Pradesh, India.

E-mail: [1*]jva_cse@vignan.ac.in

Abstract

The cloud infrastructure settings used to maintain security across distributed servers, lively and reliable data transport is necessary. Error detection techniques like Cyclic Redundancy Check (CRC) provides accuracy with low cost is important for increasing traffic volumes. This paper examines the binary format and functional role of the CRC-8 polynomial to keep data validity during server-to-server communication. The CRC identify the most common error patterns by this analysis. If it fails in some situations then sudden errors will match the generating polynomial. The performance of CRC has the detection accuracy, correction capability, bandwidth cost and relevance across various networking environments proposed in this paper compared to Forward Error Correction and Two-Dimensional Parity Checking. CRC receives 50,000 messages daily. A 1 KB payload with 16-bit CRC capability is included in every transfer. According to the test, CRC produces a 0.195% delay per message and needs 0.095MB of bandwidth. Furthermore, CRC prevents 2% of retransmissions by saving ten times more bandwidth. These result show that the polynomial provides high error detection with less resource usage is one of the most effective techniques for high-volume cloud traffic. However, FEC or hybrid algorithms are the most suitable for settings that require strong correction capabilities such as satellite connections, radio channels, or latency-sensitive systems.

Keywords: Cloud Infrastructure, Error detection, Cyclic Redundancy Check (CRC), Data Integrity, Forward Error Correction (FEC), Bandwidth Efficiency, High-Throughput Networks.

*** Corresponding Author**

DOI: https://doi.org/10.36548/jsws.2025.4.001
Received: 24.11.2025, received in revised form: 25.12.2025, accepted: 12.01.2026, published: 21.01.2026

© 2025 Inventive Research Organization. This is an open access article under the Creative Commons Attribution-NonCommercial International (CC BY-NC 4.0) License

1. Introduction

Modern computing systems depend on a fundamental infrastructure of a cloud computing network to provide a foundational work for modifying the data across multiple servers and locations. The dependence on cloud storage, computing power and managing applications has increased the importance of reliable network communication to support cloud services. Minor corruption of data during transmission can result in processing delay, inconsistencies within the system and degradation of services. Therefore, the communication protocols integrate error detection and correction capabilities to ensure data integrity during transmission.

When selecting relevant error-control techniques for cloud infrastructure communication, several issues must be considered:

1. Error characteristics of the channel (random vs burst errors)
2. Detection versus correction requirements
3. Bandwidth overhead
4. Computational complexity and latency
5. Deployment scalability in high-throughput cloud environments

Where error rates are relatively low and latency constraints are high, lightweight detection mechanisms, such as, CRC are preferred over high correction methods like FEC, which includes higher overhead and processing delay.

CRC (Cyclic Redundancy Check) used an error-control technique because of its rapid and efficient detection of error patterns. CRC's error-control function relies on polynomial division (producing a small check value or checksum). As a result, every transmitted message receives a reduced CRC check value before transmitting to the authorized receiver. The CRC can help the receiver in identifying errors by modifying one or two bits during transmission. Although, the CRC method is achieved by detecting common errors, but it has some limitations. The main limitation relates to CRC's generator polynomial can have difficult in detecting certain types of structured, or long burst errors, depending on how they are aligned.

When the cloud computing extends higher and the bandwidth increases simultaneously leads to the variations in internal server traffic.

The function and benefits of CRC compared with alternative solutions such as FEC or Two-Dimensional Parity examine the decrease of detecting accuracy, correcting ability and obtainable bandwidth. Additionally, evaluating the actual cost with CRC in real-life applications can indicate its suitability for large-scale systems. This research will explore CRC polynomial representation, assess the detection limitations and evaluate the performance efficiency in cloud message transmission and compare it to other Error-Control solutions to find the greatest value within the modern of cloud infrastructure.

2. Related Work

Emerging advancements in high-performance networking, cloud security, traffic engineering, and intelligent intrusion detection systems highlight a shift toward more scalable, programmable and secure network infrastructure, driven by increasing data-plane complexity and the need for real-time, resource-efficient operations. Recent innovations such as CRC-8–based lookup mechanisms have demonstrated significant promise in enabling low-memory, high-speed table lookups for next-generation network applications, with both the ICNP prototype and its later Parrot Hashing extension offering scalable hash-table designs suitable for line-rate packet processing and memory-constrained environments [1][2]. In parallel, the broader domain of traffic engineering and routing continues to evolve, with Segment Routing (SRv6) performance-evaluation frameworks enabling fine-grained measurement of forwarding efficiency, overhead, and latency behaviour in modern programmable networks [3]. Hardware acceleration further enhances these trends, as SmartNIC and FPGA-based offloading architectures such as those explored through Rosebud demonstrate substantial reductions in processing delay, CPU utilization, and packet-handling bottlenecks, reinforcing FPGA integration as a cornerstone for high-performance network virtualization and data-plane acceleration [4]. Meanwhile, cloud platforms demand stronger data-security guarantees, with secure deduplication frameworks like DEDUCT providing cryptographically resilient techniques for reducing cloud storage redundancy while preserving confidentiality against insider threats and dictionary attacks [5]. Communication environments relying on distributed and mobile infrastructures, including UAV-assisted data acquisition networks, increasingly benefit from network-coding techniques that enhance throughput, reliability, and resilience under highly dynamic topologies [6]. The rapid expansion of IoT ecosystems introduces new attack vectors, prompting the need for sophisticated anomaly-detection and botnet-mitigation systems such as TJO-based feature selection combined with tree-growth–optimized LSTM classifiers, which significantly strengthen intrusion detection precision in resource-constrained IoT deployments [7]. Modern traffic-control research adds further programmability through innovations in ECMP customization, offering fine-grained, per-flow load balancing and deterministic routing controls that eliminate the limitations of traditional hash-based distribution mechanisms [8]. Industrial control systems at the edge also requires advanced security enhancements, prompting the development of multi-layered protection frameworks that incorporate anomaly monitoring, hardened communication channels, and authenticated edge-gateway architectures tailored to high-risk operational environments [9]. Finally, realistic simulation platforms for embedded defence systems contribute to safer and more reliable software deployment pipelines, enabling robust regression testing, fault identification, and scenario-driven validation that support mission-critical system reliability in defence-grade embedded environments [10].

2.1 Research Gap

Current research work focused f on the mathematical strengths of CRC but it lacks practical evaluation of how CRC performs in real cloud environments with large volumes of internal server traffic and real-time logical errors. Existing studies rarely compared CRC directly with FEC and 2D parity in terms of detection accuracy, correction capability, and bandwidth overhead within high-throughput cloud settings. Additionally, there is insufficient analysis of the actual bandwidth decreases the CRC overhead and it prevents the retransmission. An alternative CRC polynomial can be suitable for dynamic cloud traffic patterns to assess. This work addresses these gaps by examining CRC’s limitations, comparing multiple error-control techniques and quantifying the bandwidth impact in large-scale distributed systems.

This work presents a quantitative, system-level evaluation of CRC-8 performance in a realistic cloud-traffic situation by analyzing mathematical error detection behavior, bandwidth overhead, and prevents retransmission. The previous works focused on CRC’s mathematical properties. This proposed study combines theory and practical aspects by modeling undetected burst-error conditions by comparing CRC with FEC and 2D parity under same workloads and particularly quantifying the net bandwidth benefits of CRC in large-scale cloud infrastructures.

3. Methodology

The research methodology applied within this article is designed specifically for analytical and quantitative assessment of the limitations of CRC-based approaches to cloud computing infrastructure error detection. The sequential work (as illustrated in Figure 1) of the overall process includes polynomial modeling, pattern-based detection characteristics, evaluations will compare and evaluate the CRC techniques interact with other error detection methodologies and assesses the effectiveness of CRC error detection on specific kinds of resources (i.e. level of impact on available bandwidth). Figure 1 shows the proposed workflow of the system.

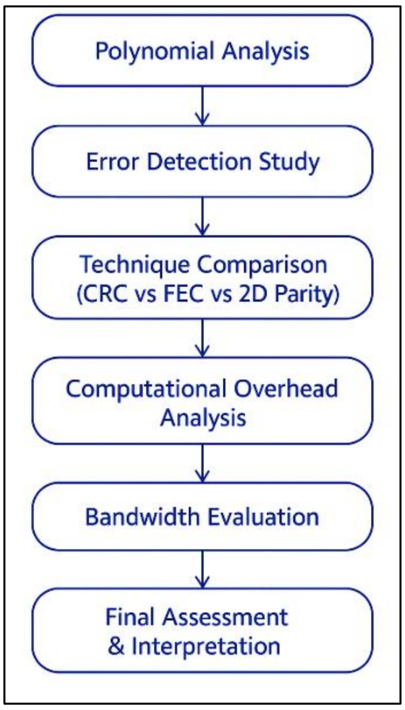
Polynomial Analysis
Error Detection Study
Technique Comparison (CRC vs FEC vs 2D Parity)
Computational Overhead Analysis
Bandwidth Evaluation
Final Assessment & Interpretation

Figure 1. Workflow of the Proposed Work

3.1 Polynomial Analysis

The first methodological step involves constructing the binary generator sequence for CRC-8 polynomial specified in the scenario.

$$G(x) = x^8 + x^2 + x + 1$$

According to the coefficient-mapping process, each term from x^8 to x^0 is assigned a binary coefficient of 1 or 0 depending on its presence or absence. Therefore, this CRC-8 polynomial represents a balance between its capability to identify errors and the amount of time/resources to be allocated in order to utilize it for generating CRC values on a large scale. This yields the 9-bit generator:

$$G=100000111$$

The binary generator follows the cloud provider's CRC encoding procedure by using the CRC generator polynomial to identify error patterns that can be identified and/or overlooked. The algebraic relationship between polynomials and the maximum burst length that will be detected depending on the polynomial type. It also establishes vulnerability to particular long-burst patterns. A selected polynomial minimizes the likelihood of undetected errors while optimizing detection coverage. The CRC-8 polynomial has been selected to use a minimum hamming distance of 4 for the most widely used message lengths to detect all single, double bit and multi-bit random errors. A zero-padded 8192-bit message block (1 KB) is divided by G(x) in a modulo-2 polynomial division model. The CRC check sequence is attached to the message for division output. This stage serves as the foundation for subsequent error-detection analysis and permits accurate reconstruction of the provider's actual CRC generation mechanism.

3.2 Error Detection

The Step 2 explain the types of misidentified burst patterns presented in burst errors that makes the CRC unable to detect. Theoretical CRC properties state that, if the burst polynomial is divisible by the generator polynomial, a CRC of degree r may fail for bursts longer than r, but it will detect all burst errors shorter than or equal to r. When CRC-8 is used as an error detection method, errors in large groups will be unnoticed. CRCs identify errors by dividing the error polynomial by the CRC's generator polynomial. The modulo-2 division will yield zero when a burst error polynomial is an exact multiple of the CRC generator polynomial. As a result, even if a frame has one or more errors, it can still be processed as valid. This results in large multi-bit burst errors is unnoticed.

To scientifically model this behavior, the methodology performs the following:

1. Construct error polynomials $E(x)$ simulating bursts of varying lengths (9–50 bits).
2. Compute divisibility of $E(x)$ by the generator polynomial $G(x)$.
3. Identify failure regions' sets of burst patterns for which

$$E(x)\ \mathrm{mod} G(x) = 0$$

4. Map observed errors to these theoretical failure regions, validating the provider's field observations of error escapes.

This analysis produces a mathematical explanation for the undetected corruptions observed in the distributed cloud servers.

3.3 Comparative Framework for CRC, FEC, and 2D Parity

The third methodological component involves a structured comparative analysis of CRC, Forward Error Correction (FEC), and 2D Parity based on:

- Detection accuracy
- Correction capability
- Processing complexity
- Latency
- Bandwidth overhead

The comparison uses the performance criteria presented in the tabulated analysis (e.g., CRC detecting all single-bit errors but it is lacking in correction. FEC correcting up to t errors per block and 2D parity locating single-bit errors via row/column intersections)

This phase adopts a multi-criteria evaluation model where each technique is valued across:

- Computational load (XOR count, decoding cycles)
- Channel conditions (error probability, burst characteristics)
- Deployment context (LAN, satellite, wireless, storage arrays)

This enables precise mapping of technique suitability to network environments.

3.4 Bit-Level Overhead and Bandwidth Modeling

The methodology computes CRC-induced bandwidth overhead using the explicit values given in the situation:

- Payload: 8192 bits
- CRC: 16 bits
- Total: 8208 bits/message
- Overhead: 0.19493% (≈0.195%)

In this study, the generator polynomial used a CRC-8 polynomial. This work uses a 16-bit CRC field at the frame level to confirm the fixed message-format constraints. This design maintains the algebraic properties of the CRC-8 polynomial allows better compatibility with system-level framing and also providing a simple calculation on bandwidth in large cloud-based deployments.

Daily CRC traffic is then calculated:

$$50{,}000 \times 16 = 800{,}000 \text{ bits/day}$$

This is converted to bytes and MiB for practical cloud-scale interpretation. The total message-plus-CRC bandwidth (410.4 Mb/day) is also computed. This quantitative analysis allows precise evaluation of CRC's effect on communication resources in a real cloud deployment.

Bandwidth overhead is a critical metrics in error-control evaluation because of the additional redundancy directly consumes network capacity and scales in high-volume cloud environments. Even the small per-message overheads can accumulate into the substantial daily traffic costs. Therefore, an effective error-control mechanism must balance the detection reliability with a minimal bandwidth consumption to maintain a scalability and the cost efficiency.

3.5 Retransmission Avoidance Modeling and Savings Quantification

The fifth step models the amount of bandwidth saved by preventing retransmissions. With a stated 2% reduction in message retransmissions due to CRC:

$$0.02\times 50{,}000=1{,}000 \text{ avoided retransmissions/day}$$

Bandwidth saved:

$$1000\times 8208 \text{ bits}=8.21 \text{ Mb/day}$$

Subtracting CRC overhead (0.8 Mb/day) yields net savings of:

$$7.4 \text{ Mb/day}\approx 926 \text{ kB/day}$$

This step forms the basis for evaluating the cost-benefit tradeoff between CRC overhead and prevented retransmission traffic a key factor in high-volume cloud networks.

4. System Architecture

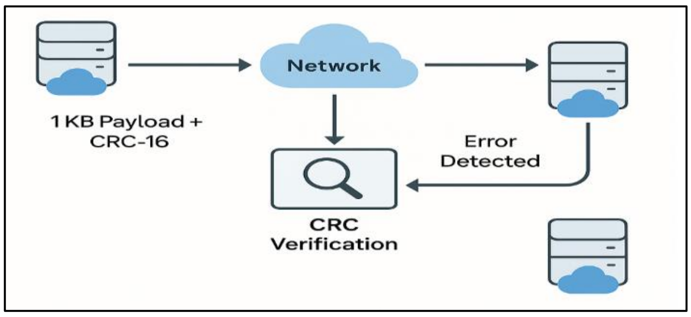

Figure 2. System Architecture of the Proposed Work

Figure 2 represents the system architecture of the proposed work. The CRC error detection scheme has been developed using a three-level system with most basic level comprising the method will allow the simulation workload generated in the form of a 1KB message payload has been encoded with the CRC using a defined polynomial. In addition to this simulation feature on this particular level, a message scheduler also used to create the number of active normal-level cloud services in a given cloud system with 50,000 messages daily. A conventional modulo-two division method used for the processing of all generated frames to add the remainder bits corresponding to the CRC method to be studied into the basic message.

The Channel/Sockets Device is employed for simulating network environments with the help of a network emulator to insert error bursts at random times. The Network Emulator follows the models with transmission of packets in a real-world environment of networking. For example, the latency in the whole transmission of packets in a real-world network environment in addition to the packet transmission towards the destination. The Error Injector injects errors in a controlled manner by means of error statistical distributions. The device is able to inject arbitrary error bursts of bits in addition to the alignment of the bursts for testing the system in relation to reach any position or situation that affects the CRC error failure to test the record of the initial time of error to interpret the results of the undetected errors without compromise in accuracy.

The messages are processed through channel to the receiver, detector, and CRC checker within the analysis layer. The polynomial remainder may be calculated as if there is no Corruption, no exists. All frames that fail the integrity test required to recorded by the analysis layer. When all the frames are identified within the analysis layer, the CRC will be detected and the frames for evaluation will be compared to all other frames. The analysis layer requires a CRC checker and relies on the capability of either Forward Error Correction or Two-Dimensional Parity within their evaluation of overall detection performance from the methods for their respective error types/cycle conditions. Failure analysis within the analysis layer relies on its method for the determination of the polynomials within its failure region analysis.

The settings conduct an experiment, execution schedules and procedures to extract the stored results of each execution of the experiment are all handled by the orchestration and control plane itself. The settings appear to commit the results in reproducible, affecting parameter combined with parallel execution in microservice containers is handled in the orchestration layer. Multi-level storage is supplemented by time-series databases to store metrics, object storage to store raw bitstream data used for analytical results in a relational database. Advanced results analysis in real-time error detection rates, overhead, prevention of retransmission related to performance will be extracted by utilizing visual tools.

This architecture also provides various trustable and secure features to preserve the integrity of the experimental results. These trustable and secure features include the deterministic seed, providing traceability for random traces and also traces that include noise patterns. In addition, there are verification steps to ensure traces are traceable in storage. The architecture ensures both the use and unuse integration of experimental results.

5. Simulation Setup

A validation/simulation tool is proposed to develop the results. It provides an accurate approximation of the actual dynamic processes in the cloud infrastructure. The proposed simulation tool is capable of simulating the actual dynamic processes in the message transfer processing system used by the cloud service provider, provide feature to have a controlled bit error rate, burst error rate, and message transmission rate simultaneously. In fact, the simulation model provided the simulation tool is actually very accurate model of the message being exchanged between the servers of the cloud service provider. While executing the simulation tool to process the message at a rate of 50,000 messages per day, comprising of an 8192-bit message further accompanied by a 16-bit CRC code, accurately simulated the processing of messages by the cloud service provider. This would carry the seeded message with pseudo-random binary values corresponding to the given information to test its capability for correcting a burst error. The burst errors would be bit-correlated errors compared to random bit error bursts with possible hardware malfunction or bit-correlation interference. Burst errors will always have a bit-correlated and not a random bit error burst in cloud. In this simulation, bit-correlated burst errors used instead of random bit error bursts. These burst errors are randomly placed in a message frame to test its capability to detect bit-corruption errors using CRC-8 and then ignoring or detecting them using CRC-8. The number of random tries for each type of burst error, 10,000 were used will be considered as the data referred to in the previous Table 1 in the discussion. It is considered to be 10,000 tries for each length of segment within the simulated trial for a burst error, the amount of actual data within a simulation will not lead towards sampling error and will tend towards the simulation data suggests as a better method within the actual amount of the CRC pattern instead of data patterns. The type of simulation model has been accepted for simulations of CRC are referred within the discussion. The CRC Encoder/Decoder utilized within the CRC simulation model has an operational CRC suited toward the modulo 2 polynomial division using the CRC polynomial and divisor polynomial specified within the CRC Generator $G\ (x)$ = 100000111 G(x)=100000111. Each frame is scanned for error in the polynomial remainder for error checking system, whether the error frames are undetected with detected frames. Additionally, in the undetected frames, error in secondary bit stream comparison modules is recognized in verification of difference in the received frame and sent frame in view of negative CRC values. There are no false positives exist in this work in view of two-tiered system where the error in CRC values is judged. The actual bandwidth saving is recognized in message size recorded as CRC in view of actual overheads prevented from being transmitted. The actual average is recognized in actual values of transmission (50,000), actual CRC bit overhead (16), along with actual values of prevented transmissions (2%), whereby in actual bandwidth saving at 0.095MB/day is recognized.

6. Results and Discussion

In this research work, experiments were quantified with the three main metrics (bandwidth overhead due to retransmission, burst-error-detection rates, and retransmission savings). The data provided in Table 1 demonstrates that the burst-error-detection performance of the CRC-8 is better than average but does not achieve a 100% success rate. The average burst-error-detection rate at burst-error lengths of 1 through 8 bits is 99.92% based on the 8-bit polynomial used for the CRC-8. As the length of a burst error exceeds the degree of the polynomial, there are more undetected bursts. The detection rate for burst-error lengths 9 through 15 bits is 99.84%. The detection rates for 16 through 32 bits are 99.71% and 33 through 50 bits are 99.45%. The figures presented in Figure 3 shows that the performance of the experiment produced undetectable burst-error patterns only in limited instances of long bursts of errors.

Table 1. Burst-Error Detection Rates for CRC-8

Burst Length (bits)	**Total Patterns Tested**	**Detected**	**Undetected**	**Detection Rate (%)**	**Undetected Rate (%)**
1–8	10,000	9,992	8	99.92	0.08
9–15	10,000	9,984	16	99.84	0.16
16–32	10,000	9,971	29	99.71	0.29
33–50	10,000	9,945	55	99.45	0.55

The detection rates are expressed as 95% confidence intervals based on estimates using binomial proportions. The widths of the confidence intervals in all cases of burst lengths were below ±0.3%. This justified the statistical accuracy of the obtained results.

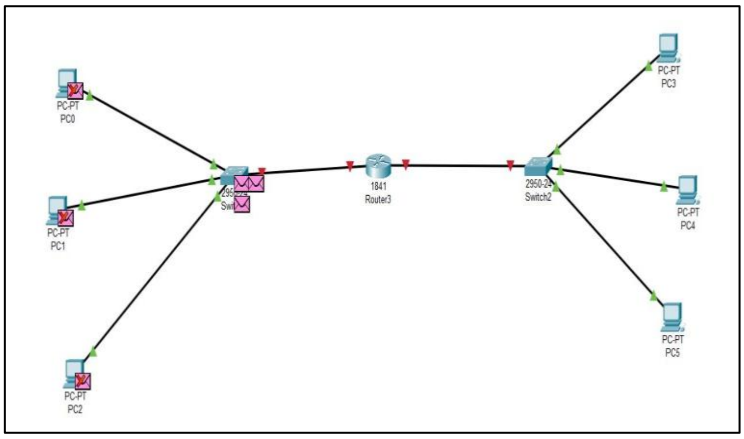

Figure 3. Detection and Undetected Error Counts for Burst Lengths Between 1 and 50 Bits

6.1 Bandwidth & Overhead Analysis

The above assessment of bandwidth & overhead analysis further supports the efficiency of CRC in the cloud setup. The addition of the 16-bit CRC trailer into the message of 8192 bits resulted around a 0.195% overhead addition, which can be made inconsequential in a large-scale implementation. The amount of 0.195% overhead of the CRC has been observed in the results demonstrates that the verification process enabled by the CRC leads to merely a trivial addition in the size of the message in comparison with the size of the data itself. This proves that CRC is achieved in its quest for high 'error detection capabilities' while sustaining 'bandwidth efficiency' fit for a high-speed communication process among the clouds. The amount of data is demonstrated per day in the graphical representation clearly illustrates the '0.8 Mb' amount of CRC data introduced into 50,000 messages. Figure 4 explains the CRC Overhead & Bandwidth saved using clear Numeric units (Mb/day).

Table 2. Bandwidth Overhead Introduced by CRC

Parameter	Value
Payload Size (bits)	8,192

CRC Size (bits)	16
Total Message Size (bits)	8,208
CRC Overhead (%)	0.195

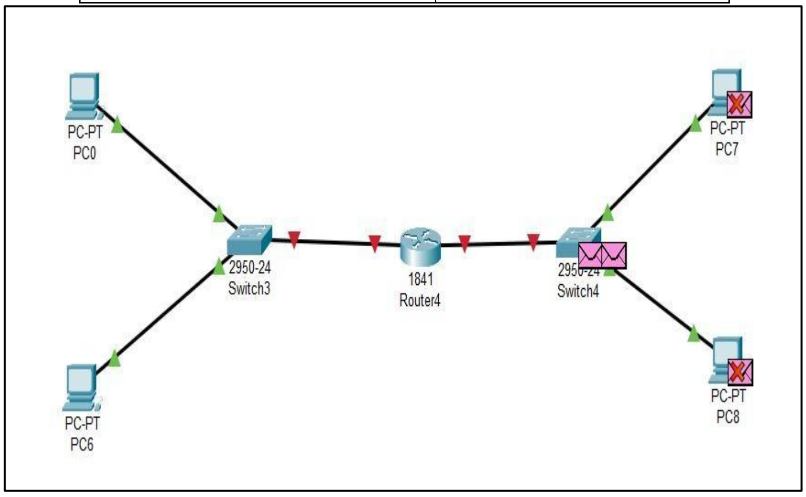

Figure 4. Comparison of CRC Overhead vs Bandwidth Saved from Prevented Retransmissions

Based on the results obtained from table 2, it can be concluded that the CRC technique is capable of functioning effectively as an economical technique in inter-cloud communication. In addition to this, it will be even more appropriate in the context of inter-cloud communication due to its lower complexity level and efficiency in recognizing most error patterns. However, the efficiency in recognizing error patterns cannot be reversed and its inefficiency in recognizing error patterns can also be taken into consideration. The error patterns are easily noticed in wireless uplink communications along with satellite communications due to higher noises.

Table 3. Daily CRC Bandwidth Consumption

Metric	**Value**
Messages per Day	50,000
Total CRC Bits per Day	800,000
Total CRC Bytes per Day	100,000
Total CRC MB per Day	0.095 MB

Total Daily Traffic (Payload + CRC)	410.4 Mb

Also, due to the high degree of computation required and the associated processing overhead. This type of system like FEC would fit for long-lived lower reliability and higher latency connections such as telecommunications than the extremely low latency data center-level workloads. In other words, data center application is the most effective technique used in CRC. Due to workload, a hybrid approach is implemented to eliminate the uncontrolled or unmonitored nature of long or irregular runs of errors associated with FEC as explained in table 3.

7. Conclusion

This research work explains the performance and efficiency of the CRC-8 polynomial in a cloud communication system. It verifies high error detection ranging from 99.45% to 99.92%, for various burst error patterns, although limited by its polynomial nature. It also verifies high bandwidth efficiency because the computation overhead per message is merely 0.195%, and it approximates about 7.41 Mb/day overhead reduction caused by the avoidance of retransmission, which enhances the communication efficiency. When compared with other alternatives, such as FEC and Two-Dimensional Parity, this method balances perfectly in error accuracy, overhead cost, and computation essential for communication processes that require low latency and high throughput. It is limited by the ability to identify structured long-burst errors correctly, it will work properly with controlled data center networks. Other alternatives can also suit applications with high error rates. Overall, this paper verified the efficiency of the proposed method in terms of data integrity at cloud communication systems.

References

[1] Liu, Yi, Shouqian Shi, Ruilin Zhou, Yuhang Gan, and Chen Qian. "Scalable, Fast, and Low-Memory Table Lookups for Network Applications with One CRC-8." In 2024 IEEE 32nd International Conference on Network Protocols (ICNP), IEEE, 2024, 1-12.

[2] Liu, Yi, Shouqian Shi, Ruilin Zhou, Yuhang Gan, and Chen Qian. "Parrot Hashing: Fast and Low-Memory Table Lookups for Network Applications with One CRC-8." IEEE Transactions on Networking (2025).

[3] Abdelsalam, Ahmed, Pier Luigi Ventre, Carmine Scarpitta, Andrea Mayer, Stefano Salsano, Pablo Camarillo, Francois Clad, and Clarence Filsfils. "SRPerf: A Performance Evaluation Framework for IPv6 Segment Routing." IEEE Transactions on Network and Service Management 18, no. 2 (2020): 2320-2333.

[4] Vargiu, Alessandro. "FPGA Acceleration in SmartNICs: Porting and Performance Evaluation of the Rosebud Framework." PhD diss., Politecnico di Torino, 2025.

[5] Ghassabi, Kiana, and Peyman Pahlevani. "DEDUCT: A Secure Deduplication of Textual Data in Cloud Environments." IEEE Access 12 (2024): 70743-70758.

[6] Vladimirov, Sergey, Ruslan Kirichek, and Vladimir Vishnevsky. "Network Coding for the Interaction of Unmanned Flying Platforms in Data Acquisition Networks." In Proceedings of the 4th International Conference on Future Networks and Distributed Systems, 2020, 1-7.

[7] Rayala, Ramya Vani, Chandrakanth Reddy Borra, Piyush Kumar Pareek, and Srinivas Cheekati. "Securing IoT Environments from Botnets: An Advanced Intrusion Detection Framework Using TJO-Based Feature Selection and Tree Growth Algorithm-Enhanced LSTM." In 2024 International Conference on Recent Advances in Science and Engineering Technology (ICRASET), IEEE, 2024, 1-8.

[8] Liu, Yadong, Yunming Xiao, Xuan Zhang, Weizhen Dang, Huihui Liu, Xiang Li, Zekun He et al. "Unlocking {ECMP} Programmability for Precise Traffic Control." In 22nd USENIX Symposium on Networked Systems Design and Implementation (NSDI 25), 2025, 87-106.

[9] Xia, Wei, Chang Deng, and He Huang. "Research on Edge Side Security Enhancement Technology for Industrial Control Scenarios." In 2024 5th International Conference on Computer, Big Data and Artificial Intelligence (ICCBD+ AI), IEEE, 2024, 379-385.

[10] Grün, Charlie. "A Realistic Simulator for Regression Testingwithin an Embedded Defence System." (2025).