Mitigation of Hardware Trojan in NoC using Delta-Based Compression

Hamza Amara, Cédric Killian, Daniel Chillet, Emmanuel Casseau

Mitigation of Hardware Trojan in NoC using Delta-Based Compression

Hamza Amara
Univ Rennes, Inria, CNRS, IRISA
Lannion, France
hamza.amara@inria.fr

Cédric Killian
Univ Jean Monnet Saint-Etienne, CNRS
Saint-Étienne, France
cedric.killian@univ-st-etienne.fr

Daniel Chillet
Univ Rennes, Inria, CNRS, IRISA
Lannion, France
daniel.chillet@irisa.fr

Emmanuel Casseau
Univ Rennes, Inria, CNRS, IRISA
Lannion, France
emmanuel.casseau@irisa.fr

***Abstract*—MultiProcessor System-on-Chip (MPSoC) and Network-on-Chip (NoC) are closely linked in the design of modern computer architectures. Indeed, MPSoCs are designed to integrate multiple processing units on a single chip and take advantage of NoCs to suppport communications between these multiple processors. As data exchanges via the NoC can be voluminous, it can lead to increased latency and bandwidth consumption. Thus, on-chip packet compression techniques exploiting data correlation may mitigate these issues. However, compression can make data more sensitive to faults or attacks. Therefore, it is important to consider the potential presence of hardware Trojans (HT) within some of the intellectual properties (IP) embedded in the NoC. In this article, we focus on Delta-based compression, which allows the compression of a set of data using one base and distances. After formalizing the delta-based compression cost, we analyze the impact of fault injection attacks on application quality degradation. Additionally, we present a lightweight mitigation technique focusing on bases protection by using freely available bits in the header flit. Experiments show that our mitigation technique is effective in reducing the impact of hardware Trojan attacks on compressed packet data with a mean squared error gain up to 67% and a loss in compression ratio of around 8% for CIFAR-10 database.**

***Index Terms*—Network-on-Chip, Delta-based compression, NoC Security, Hardware Trojan**

I. Introduction

The development of MultiProcessor System-on-Chip (MPSoC) has revolutionized the computer field, offering large opportunities for advanced architectures. Nowadays, such systems can satisfy all requirements of performance and flexibility for complex application. The main component that makes these systems very efficient is the Network-on-Chip (NoC) [**?**], used for interconnecting components on a single chip. Unlike bus-based architectures, NoCs [**?**] offer a flexible, scalable and efficient solution for enabling communication between multiple processor cores, memory blocks and peripherals on a single chip. The NoC is a complex system made up of Network Interfaces (NI), Links and Routers. NoC components that ensure connectivity between the various processing units usually come from third-party Intellectual Properties (IP). These IPs nodes are connected to routers by NI and communicate with each other using data packets. Since the size of the communication channel is generally smaller than the packet size, the source NI (NI^s) divides the packet into several flits, such as head, body and tail, and injects them sequentially into the local router. This division into flits ensures that data can be transferred efficiently across the NoC, even if communication channels have limited capacity. Once all the flits have been routed to the destination router, they are reassembled to reform the original packet, which is then forwarded to the destination IP component by the corresponding destination NI (NI^d). With the increasing number of IP into an MPSoC, and with intensively communicating applications, the NoCs may suffer from latency increase and bandwidth consumption. To avoid this limitation, on-chip packet compression techniques are proposed to reduce average latency. These techniques exploit data redundancy within flits and compress the flits before injecting them into the NoC. This reduces the number of flits to be sent per packet, which in turn reduces network traffic, bandwidth consumption and NoC power consumption. NI^s is in charge of the compression at the source node and NI^d is in charge of decompression at the destination node. Delta compression techniques [**?**], [**?**], [**?**], [**?**] have been chosen over other existing compression techniques thanks to their low hardware cost and high compression ratio, particularly when the data to be compressed are correlated. These compression techniques need to send one or several common bases and sets of distances that will be used to decompress the packet at NI^d. Details about the compression are discussed later. In practice, it is important to consider the potential presence of hardware Trojans (HT) within some of the IP embedded in the network-on-chip (NoC). These HT can be triggered by malicious adversary to degrade the application quality through injecting faults within data packets. Indeed, as explained later, bases are used to compress/decompress several data, which makes them more critical than distances which are used only once.

In this work, we study the impact of a Hardware Trojan (HT) attack on compressed data packet in a NoC-based

MPSoC. We assume that the HT is within NoC routers and is able to modify the header fields [**?**] and the payload [**?**] of packets. To tackle this problem, we first study on the sensitivity of compressed packets regarding attacks and we then propose a mitigation technique. Thus the contributions presented in this paper are the following:

- Formalize the concept of packet compression using the delta compression method.
- Analyze the impact of attacks on compressed packet data (bases and payloads).
- Propose a lightweight mitigation technique to reduce the attack impact on the data transferred over the NoC.

II. Background and Related Work

In this section, we propose some definitions to formalize the delta-based compression techniques when applied to on-chip communications. The different notations used are listed in **Tab. ??**. Related works on protection against Hardware Trojans within NoCs are also presented.

TABLE I: Notation summary

Symbol	Definition
Δ_i	Difference between B and D_i
B	Base for a set of data $\{D_i\}$
C	Code associated to base B
D_i	i^{th} data in the set of data $\{D_i\}$
F_j	j^{th} flit of a payload packet
N_C	Number of cores of the NoC architecture
N_D	Number of data per payload flit
N_F	Number of payload flits per packet
S_Δ	Size of Δ_i (bits)
S_B	Size of base B (bits)
S_C	Size of code C (bits)
S_D	Size of data (bits)
S_F	Size of flit (bits)
S_{HF}	Size of the HF (bits)
S_O	Size of others field in the HF (bits)

A. NoC-related notations

As already briefly explained in the introduction, a NoC consists of routers, network interfaces, and several IP cores as presented in Fig. **??**. Without loss of generality, we consider a 2D mesh NoC of N_C cores. Messages are divided into packets with one header flit (HF) and a fixed number of payload flits per packet (N_F). The flits carried by the NoC have a size of S_F bits. Messages consist of data whose size is S_D bits (for example, in the case of an image, data are pixels), which is usually smaller or equal to the flit size ($S_D \leq S_F$). Therefore, each flit contains an integer number of data, $N_D = \lfloor \frac{S_F}{S_D} \rfloor$.

The number of bits of the header flit must be large enough to encode at least the source and destination IP addresses of the packet (i.e. X and Y coordinates, hence $2 \times log_2(N_C)$). The HF may contain other useful information such as packet ID, encoding message type, channel, etc. Let us assume S_O bits are required to represent this useful information. In this

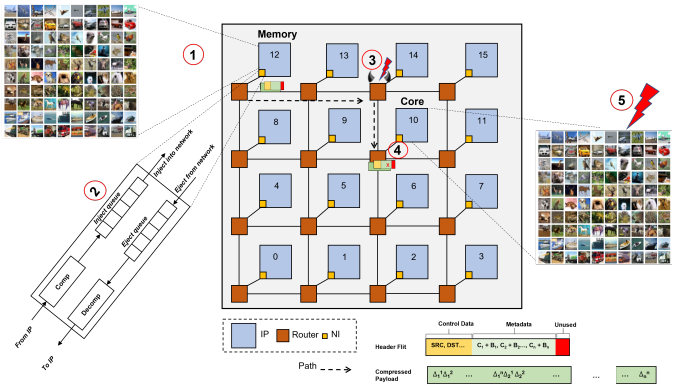


Fig. 1: 4 x 4 mesh NoC with HT router.

case, the minimum number of bits to represent the header flit ($S_{HF_{min}}$) is given by equation (**??**):

$$S_{HF_{min}} = 2 \times log_2(N_C) + S_O \quad (1)$$

B. Delta-based on-chip data compression

Delta-based compression is a technique that consists of defining a common base B for a set of data $\{D_i\}$ and then defining a set of differences $\{\Delta_i\}$ from base B (equation (**??**)).

$$\Delta_i = B - D_i \quad (2)$$

with i the index of data D_i in the set of data $\{D_i\}$.

The delta-based compression technique has been adapted in the NoC domain to compress either packets [**?**] or flits [**?**], [**?**]. In the case of payload packet compression, the base is common to the set of data $\{D_i\}$ inside a payload packet. On the other hand, in the case of payload flit compression, the base is common to the set of data $\{D_i\}$ inside a flit. Usually, the size of the base (S_B) equals to the size of the data ($S_B = S_D$) and the size of the differences Δ_i, denoted S_Δ, must be smaller than the base size S_B ($S_\Delta < S_B$) when there is data compression. Otherwise, there is no data compression and the data is transmitted as it is. A specific code C is thus assigned to each base B to specify the number of bits S_Δ used to encode the set of related differences $\{\Delta_i\}$. NoΔ compression technique [**?**] compresses packets, making it an intra-packet compression technique. FlitZip [**?**] is an intra-flit compression technique where each payload flit, denoted F_j with $j \in (1...N_F)$, is compressed individually, that often enables higher compression performance than NoΔ. Each payload flit contains N_D data D_i. The base B is calculated as the average of the smallest and largest D_i within each payload flit.

In case of compression, a delta-based compressor is used to generate the bases B and the codes C. Both B and C constitute metadata that are usually stored in the header flit. The related differences $\{\Delta_i\}$ are placed in the payload flit. For example, in the case of FlitZip intra-flit compression technique, the size of the HF, denoted S_{HF}, is given by equation (**??**):

$$S_{HF} = S_{HF_{min}} + (S_C + S_B) \times N_F \quad (3)$$

with S_C the size of code C.

Pullaiah *et al.* proposed $B\Delta$-NIS [**?**] as an improvement of the FlitZip intra-flit compression technique. This method uses two compression modes: $B\Delta$ for flits already compressed using FlitZip compression method, and Neighbourhood Indexing Sequence (NIS) to compress flits that can not be compressed using FlitZip. The NIS involves finding an optimal code word for uncompressed D_i in flit F_j using zero's and one's based traversals. The number of extra control bits to be placed in the metadata header for flits compressed by NIS is $c = 2 \times N_D$.

In [**?**], the authors have also proposed an enhancement to the FlitZip compression technique [**?**], focusing on reducing the complexity of base B and Δ_i calculations. The common base B for a set of data $\{D_i\}$ in flit F_j is constructed from the similar MSB bits among these data D_i. The remaining non-similar bits represent the set of $\{\Delta_i\}$ in flit F_j. This approach leads to consumes less area and power. However, it results in a loss of compression ratio compared with [**?**] and [**?**].

To simplify the presentation of our technique, we use FlitZip as baseline compression method, but our technique could be extended and implemented for the other compression techniques.

C. Related work

Several works have been carried out on protection against Hardware Trojans (HTs) within the NoC [**?**]. Various threat models linked to HTs have been studied, leading to the risk of Denial of Service (DoS). Among these threats, packet corruption can be caused by a Hardware Trojan router R^{HT} on the NoC, resulting in the drop of the packet [**?**]. Kulkarni et al. [**?**] proposed an HT model capable of modifying the destination address field of specific packets. A packet attacked by the HT, called Victim Packet in this paper, is therefore sent to the wrong destination, resulting in the packet being dropped. In [**?**], the authors proposed the use of bit shuffling as an encoding mechanism within NoC switches. This strategy aims to counter HT attacks targeting critical packet fields such as head bit, tail bit, destination address and quantity, while preserving the interactive performance of the MPSoC. The aim is to mitigate the impact of a Denial of Service (DoS) attack. Wang et al. [**?**] proposed an HT detection method DetectANN based on an artificial neural network (ANN). This model is trained off-line and uses various features such as link and buffer utilization of each input port, local temperature and transient error rate. During the detection phase, the trained model identifies routers as HT-infected or HT-free. This detection is followed by the HT mitigation stage, in which predicted labels are passed to an intelligent routing module, SmartRoute, based on Deep Reinforcement Learning (DRL) to choose from three routing algorithms: O1TURN, West-First and Negative-First. Unlike previous works which focus on the study of an HT within a classical NoC, our work explore the impact of attacks by an HT targeting compressed packet data in an NoC using delta-based compression.

III. Detail of the attack

A. Threat model

As previously explained, we consider an MPSOC architecture based on a mesh NoC supporting a compressed data traffic. In our threat model, we consider an HT capable of modifying data of the packet following the destination path through the malicious router. The proposed attack consists of flipping part of the base bits in the header flit and/or bits of the payload packets. We provide an illustrative example that explains how HT works in a NoC using a delta-based compression technique, as shown in **Fig. ??**. On this figure, the numbers correspond to:

1. Input images to be sent from IP_{12} to IP_{10};
2. Packetization and compression at source network interface NI_{12};
3. Fault injection attack on the packets;
4. Decompression of the packets at destination network interface NI_{10};
5. Resulted degraded images.

In this example, we suppose that the data to be sent are CIFAR-10 images. The message is packetized within the source NI_{12}, then these packets are compressed using the delta compression technique before being injected into the NoC. IP_{12} sends compressed packets to IP_{10} using the NoC managed by an XY routing algorithm for example. Passing through router R_{14}, the HT tampers the victim packets. Thus, after the packets have been decompressed by destination NI_{10}, data are received by IP_{10} with a degraded quality.

B. HT attack impact analysis

To assess the impact of HT proposed in section **??**, we opted for FlitZip delta-based compression technique as an example for this experiment. In the context of the FlitZip technique described in [**?**], the values of the parameters are:

$$N_C = 64 \quad S_F = 128 \quad S_B = 8 \quad S_D = 8$$
$$N_D = 16 \quad N_F = 6 \quad S_C = 3 \quad R_p = 50\%$$

with R_p the packet attack rate, i.e. the rate at which packets passing through the malicious router fall victim of fault injection attacks. When R_p equals 50%, half of the packets fall victim of fault injection attacks.

Based on these values and on equation (**??**), the minimum number of bits required to store all the control and metadata bits in the HF (S_{HF}) is equal to 119.

For this context, we studied the consequences of fault injection on the bases and/or payload of compressed packets using the CIFAR-10 database. The **Fig. ??** summarizes these results. The number of faults injected per victim packet is given by X axis, while Y axis shows average Mean Squared Error (MSE), where a lower MSE is better. As a reference, we show the impact of fault injection attacks on payload of uncompressed packets (blue curve on the figure). The impacts of fault injection attacks on payload of compressed packets as well as on both payload and bases of a compressed packets are respectively shown with orange and red curves.

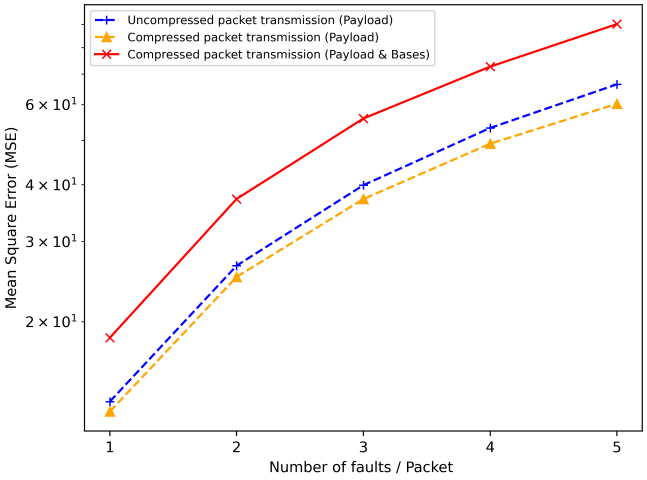


Fig. 2: Quantifying the impact of the fault injection attack on CIFAR-10 database using the MSE metric, N_F = 6.

When the HT targets only the payload of compressed packets, sensitivity is low. In this case the bases are not faulty, thus MSE is lower than uncompressed packet transmission. On the contrary, as expected, when the HT targets the bases in addition to the payload of compressed packets, attack sensitivity increases significantly. Indeed, the bases are generally related to sensitive bits of the data (MSB bits), while the distances are related to less sensitive bits (LSB bits). Thus, a fault on the MSB of these bases may have a high impact on reconstructed data. As bases are used to reconstruct several data, they play a crucial role in the transfer quality, thus it is important to ensure their protection during transmission.

IV. Countermeasure technique

To protect the bases, we propose to integrate a lightweight protection technique capable of adapting to the number of free bits in the Header Flit. Before implementing any protection, it is necessary to determine the number of unused bits S_U in the HF. This number of unused bits for encoding the protection is then given by equation (**??**):

$$S_U = S_F - S_{HF} \tag{4}$$

Knowing this number of unused bits in the HF, several light protection techniques can be envisaged, such as duplication, integration of an error detection code (EDC) or an error correction code (ECC).

In the case of a header flit containing only the source/destination fields and metadata for compression, and utilizing the parameter values defined in Section **??**, $S_U = 50$, i.e. 50 bits are available for integrating a protection technique such as base duplication. In practice, since the header flit usually includes other fields as mentioned in section **??**, equation (**??**), we have fewer bits available to integrate an effective protection technique. Using FlitZip technique as presented in [**?**], only 9 bits are available. With this number of bits, we can integrate only low level protection techniques, like a parity bit for each base.

Our contribution focuses on applications needing more efficient protection. To address this problem, we propose to increase the number of unused bits in the header flit by reducing the number of payload flits compressed in a packet. Thus, we propose to limit the number of payload flits to $N_F = 5$, rather than 6 in the FlitZip technique. Removing one flit in each packet leads to remove one couple (base+code) in the HF, which allows to increase the number of unused bits from 9 to 20. Now, with these 20 free bits, 4 bits are available for each of these 5 bases. In this case, it is now possible to integrate more complex protection technique, like a CRC-4 or a single error correction and double error detection (SECDED) for each base. For our experiments, as the base size is equal to 8 bits, we use an SECDED Hamming (12,8) code due to its ability to detect two errors, and to correct one. This implies to add 4 parity bits (S_{PB}) for each base, which exactly corresponds to the number of free bits in the HF. The total number of parity bits to be added in the FlitZip header, denoted S_{PB_C}, is given by equation (**??**):

$$\begin{aligned} S_{PB_C} &= S_{PB} \times N_F \\ &= 20\,bits \end{aligned} \tag{5}$$

To show the effectiveness of our proposed technique in mitigating the impact of HT attacking the bases and the payload of compressed packets, we used raw data from the Cameraman image. We keep the same experimental parameter values as those previously defined, with the exception of N_F, which is now set to $N_F = 5$.

Fig. **??**.a shows the reference image without faults. Fig. **??**.b shows the image received after attacks targeting the bases and payload of FlitZip-based compressed packets without any protection technique while Fig. **??**.c shows the resulting image when our protection technique is integrated. These figures highlight the worst case scenario with a number of faults injected per victim packet equals to 5. The artifacts that can be shown in **Fig. ??** are caused by the faults affecting the bases, which finally similarly affect several reconstructed data. The **Fig. ??** shows an important artifacts reduction thanks to the integration of our protection technique. Results clearly show the effectiveness of the proposed mitigation technique.

To quantify the effectiveness of our mitigation technique, we calculate the gain in MSE, where MSE is defined by equation (**??**):

$$MSE_{Y-Y'} = \frac{1}{n_p} \times \sum_{i=1}^{n_p} (Y_i - {Y'}_i)^2 \tag{6}$$

with Y and $Y^{'}$ the baseline image (image without fault) and the reconstructed image respectively, Y_i and $Y_i^{'}$ pixels of the baseline and reconstructed images respectively, and n_p the number of pixels per image.

Let $MSE_{Y-Y'}^{n_f}$ be the MSE for n_f faults injected per victim packet. Average MSE is computed by equation (**??**):

(a) Baseline image (Error free)

(b) HT attack targeting the payload and the bases of compressed packets without protection

(c) HT attack targeting the payload and the bases of compressed packets using our mitigation technique

Fig. 3: Effect of attacks on Cameraman image quality: Comparison before and after integration of the protection technique.

$$\overline{MSE} = \frac{1}{N_Y \times n_f} \sum_{n=1}^{n_f} \sum_{Y'=1}^{N_Y} (MSE^n_{Y-Y'}) \tag{7}$$

with N_Y the total number of images.

Let $\overline{MSE_U}$ be the final average MSE for unprotected attacked image, and $\overline{MSE_P}$ be the final average MSE for protected attacked image. We can then calculate the gain in MSE using equation (**??**):

$$G_{MSE} = \left(1 - \overline{MSE_P}/\overline{MSE_U}\right) \times 100 \tag{8}$$

To get a first idea of the trends in this section, we assume that the source IP sends 100 times the cameraman image to the destination IP ($N_Y = 100$) and every image is attacked [1]. In this case, MSE gain is $G_{MSE} = 73.5\%$, showing the interest of the approach. More precise results on a large set of images will be given in experiment section.

However, as we propose to reduce the number of payload flits in packets from 6 to 5, the compression ratio is affected.Thus, to be fair, it is thus important to take into account the loss in compression ratio when using our technique. Let CR_{Y_U} be the compression ratio using $N_F = 6$ and without protection technique, and CR_{Y_P} be the compression ratio using $N_F = 5$ and using our proposed technique for mitigation of the attacks.

Loss in compression ratio C_{LR} can then be computed by equation (**??**):

$$C_{LR} = \left(1 - \frac{1}{N_Y} \sum_{Y=1}^{N_Y} \frac{CR_{Y_P}}{CR_{Y_U}}\right) \times 100 \tag{9}$$

Experiments carried out on the Cameraman image show a 9.2% reduction in compression ratio when switching from 6 flits to 5 flits, i.e. MSE gain comes with a loss in compression ratio, as it could be expected. To have an idea of the trade-off between MSE gain and loss in compression ratio, size of flit S_F can be taken into account. In this goal, we reproduced the same experiments for S_F equal to 64 bits and 32 bits respectively. For these flits sizes, the MSE gain is respectively 74% and 73%, while the loss in compression ratio equals to 2.4% and 1.2% respectively.

V. Experiments

In this section, we want take a closer look at the impact of attacks on the payload of uncompressed packets, as well as on the bases and/or payload of compressed packets with and without protection. The CIFAR-10 database is thus used so that we can generalise the preliminary results presented in previous section [2].

A. Experimental Setup

For our experiments, we used the same threat model and the same parameter values as presented in section **??** with the exception of N_F and S_F: N_F is set to 5 and S_F = 32 or 64 or 128 bits. The packet compression technique is still FlitZip.

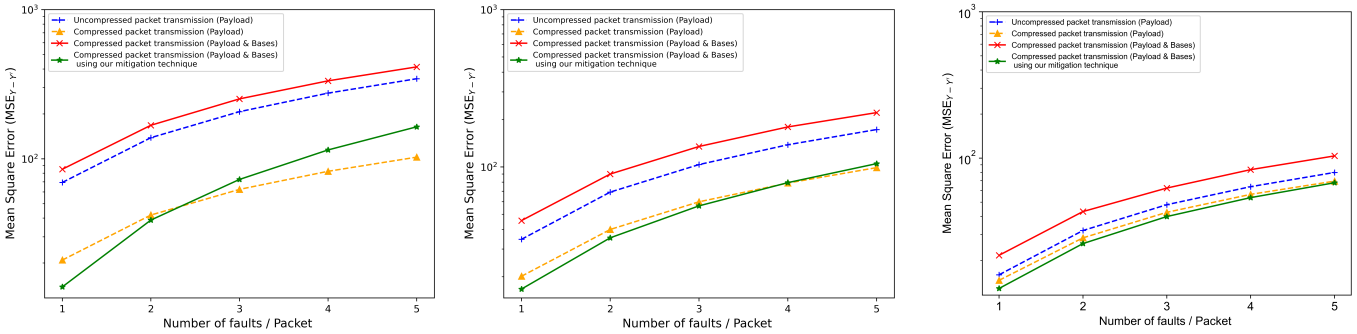
B. Experimental results

Using MSE as the metric for analysis, **Fig ??** shows the impact of the fault injection attacks. On these figures, the X axis represents the number of faults injected per victim packet, while Y axis shows average Mean Squared Error (MSE).We have implemented four different attack scenarios:

- In the first scenario, as a reference, the HT targets the payload of uncompressed packets (blue curve).
- In the second scenario, the HT targets the payload of compressed packets without protection (orange curve).
- In the third scenario, the HT targets the payload and the bases of compressed packets without protection (red curve).

[1] attack rate R_p still equals to 50%, half packets are victim of attack

[2] CIFAR-10 dataset contains 60,000 32x32 images.



(a) Flit size = 32 bits (b) Flit size = 64 bits (c) Flit size = 128 bits

Fig. 4: Quantifying the impact of the fault injection attack on CIFAR-10 database using the MSE metric, N_F = 5.

- In the fourth scenario, the HT targets the payload and the bases of compressed packets while integrating our base protection approach (green curve).

The aim of these experiments is to demonstrate the effectiveness of our mitigation technique on different flit size configurations. As expected, when the non protected bases are attacked, the quality of the reconstructed data is the lowest. As we can observe, in all three flit sizes, our mitigation technique is efficient: in most cases, MSE is less with our mitigation technique, even less than when attacking only the payload of compressed packets, except with small S_F and a large number of faults per packet. To quantify the interest of the approach, we compute the gain in MSE and the loss in compression ratio for the three flit sizes. Results are summarized in **Tab. ??**.

VI. Conclusion

In this paper, from a formalization of the delta-Based compression technique, we have analyzed the impact of fault injection attacks on compressed packet data. This analyze has confirmed that common bases must be protected as they are critical for data reconstruction. To address this problem, we have proposed a lightweight protection technique which increases the available bits in the header flits and exploits them to embed a protection of these bases. Experimental results, applied on FlitZip compression technique as an example, demonstrate the effectiveness of our proposed mitigation technique, with a gain in MSE of up to 67% and a loss in compression ratio of around 8% on CIFAR-10 database.

TABLE II: Summary of MSE gain and C$_{LR}$ performance for the CIFAR-10 database using our mitigation technique

Metric	**Flit Size**		
	32 bits	***64 bits***	***128 bits***
MSE gain	67.8%	56.4%	36.3%
Compression loss ratio	5.1%	8.3%	7.5%

Reducing the flit size leads to a reduction in the compression ratio and an increase in the MSE. In fact, as S_F decreases, the number of D_i in a flit also decreases. It results in the calculation of a base for fewer consecutive data. Since data are generally correlated, distances between data in this set of data are likely to be smaller than with higher S_F. This enables the transmission of Δ_i with reduced size in a flit, while the sensitive bits of the D_i are stored in the base. Since the bases are protected, the MSE decreases.