SoWaF: Shuffling of Weights and Feature Maps: A Novel Hardware Intrinsc Attack (HIA) on Convolutional Neural Network (CNN)
Tolulope A. Odetola and Syed Rafay Hasan

I. INTRODUCTION

- FPGA hardware accelerators offer good performance, high energy efficiency, fast prototyping, and capability of reconfiguration.
- To achieve short time-to-market, the mapping of pre-trained CNN on hardware accelerators is often outsourced to untrusted third parties.
- Due to their untrusted nature hardware intrinsic security can be compromised via malicious hardware insertions, which are very difficult to detect, especially if the IP is provided as a bitstream file.

II. Problem Formulation

Different techniques of inserting hardware attacks into CNNs have been explored. These techniques assume:
- These attacks require a manipulation of the CNN parameters.
- The attacker has full knowledge of the CNN architecture.
- The trigger is dependent on the input image
- Their payload require extra computation
- The attack is designed for a single FPGA based inference.

![Diagram showing SoWaF Trigger Design: Offline Pre-processing](image)

- In a situation where the full CNN architecture is not accessible to any one designer as seen in Multi-FPGA CNN inference. The approaches in literature may not be applicable.
- In this work we propose a framework of attack called SoWaF (Shuffling of Weights and Feature Maps) that leads to misclassifications applicable to single and multi-FPGA CNN inference.
- This approach does not require full access to the CNN architecture.

![Diagram showing SoWaF Payload Design: Runtime Operation](image)

- The attacker collects the output feature maps to setup a trigger.
- As shown in the diagram above, during the functional verification stage, a validation dataset can be used by the attacker to access the respective VPN CNN layer’s output feature maps for all the datasets.
- By choosing an index randomly of one of the channels of the output feature map of any chosen CNN layer as shown above.
- The attacker can monitors the values (X or Y or Z) of the randomly selected index to obtain a generalized range of values (RoV)
- The selected RoV for a given CNN layer serve as the trigger for the attack.

III. SoWaF Approach

- Upon triggering, for convolution and fully connected layers, the payload shuffles the channels of the weight matrix with another one as illustrated on the right hand side of the decision block above.
- CNN layers other than convolution and fully connected layers (such as Pooling layer, etc.) do not have weight matrices and channels, so the output of the feature maps are shuffled.
- This leads to miscalculation in the layer hence leading to the layer output and consequently misclassification.

![Diagram showing SoWaF Payload Design: Runtime Operation](image)

- The attack is implemented on Lenet trained on MNIST dataset and LeNet-3D for Cifar10 datasets as shown above.
- To evaluate the SoWaF attacks, we propose 5 different scenarios, where each layer (from conv1 to conv3/5) is infected with the attack.
- From the Table above, we see that DSP and BRAM usage remains the same except for Sn3 and Sn5, where BRAM is increased (5th column in Table I).
- For LUTs and FFs in all the scenarios, other than Sn5, (i.e. Sn1-Sn4) have a very modest increment in usage (up to 2.36%).

IV. Result

![Diagram showing SoWaF Payload Design: Runtime Operation](image)

- The SoWaF attack achieves misclassification when triggered by shuffling the weight matrices of convolution layers to propagate wrong feature maps. This attack is carried out without changes in the model parameters. Our results show that CNN architectures show that in all the attack scenarios, additional latency is negligible (<0.61%), increment in DSP, LUT, FF is also less than 2.36%. Three of the five investigated scenarios show very minimal changes in BRAM.

VI. CONCLUSION

The SoWaF attack achieves misclassification when triggered by shuffling the weight matrices of convolution layers to propagate wrong feature maps. This attack is carried out without changes in the model parameters. Our results show that CNN architectures show that in all the attack scenarios, additional latency is negligible (<0.61%), increment in DSP, LUT, FF is also less than 2.36%. Three of the five investigated scenarios show very minimal changes in BRAM.

VI. ACKNOWLEDGMENT

This research is partially funding provided by Tennessee Tech University College of Engineering for achieving Carnegie classification.

VII. REFERENCES