**INTRODUCTION**

From the designer, to the foundry, the consumer, microchips go through a lengthy process which has lots of risk spread throughout. The most vulnerable moment for a microchip begins straight out of the designer's workshop, when the logic gates inside the chip can be tampered or altered.

Several methods revolving around the concept of logic locking have been introduced in order to combat this. Logic locking is a general method which integrates keys in the form of extra logic gates. If an illegal user feeds a wrong key value, the chip will be "locked", which means it will function incorrectly.

During our research, we applied random logic locking (RLL), which randomly inserts XOR/XNOR gates across the circuit. To evaluate the attack resilience of this technique, we used a tool called NEOS. Furthermore, we explored the possibility of training a graph neural network (GNN) model to predict the attack resilience of any locked circuit.

**METHOD**

1. Use RLL to encrypt original circuits with different key lengths
2. Use NEOS to break locked circuits and collect the attack runtime
3. Generate a dataset including node features and graph labels
4. Train a GNN model using the dataset
5. Evaluate the performance using root mean squared error (RMSE)

\[ \text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (\text{pred}_i - \text{actual}_i)^2} \]

**LEARNING PROCESS & RESEARCH**

In order to understand what was going on behind the scenes, I started by learning how logic gates work.

After studying circuit composition, I delved into intensive research on various machine learning models and algorithms, each serving different purposes. These models consist of layers that utilize complex mathematics to produce the correct function. One prevalent model we focused on is the GNN.

![Figure 1: Security threats in outsourced fabrication](image)

![Figure 2: (left) non-encrypted; (right) encrypted with XOR/XOR gates](image)

![Figure 3: The Boolean truth tables for logic gates](image)

![Figure 4: A basic representation of a GNN](image)

**RESULTS & CONCLUSION**

We generated a dataset of 124 encrypted circuits, and use the gate type, size of fan-in cone, and size of fan-out cone as the features for training. We attack them with a timeout of 4 hours (14400 seconds) and use the attack runtime as the graph label. In the plots above, the x-axis represents the number of epochs, and the y-axis is the value of RMSE. After training, the model can also realize a similar accuracy in the test dataset, which exhibit the generalization of our model.

**Next Steps**

1. In order to gain more consistent and applicable results, introducing more original data to encrypt would be beneficial to the model's long-term success. Additionally, changing the way we encrypt the circuits could benefit the way our model learns over time.
2. There is a big fluctuation in test curve. We are going to apply different solutions, including increasing the proportion of test dataset, tuning the hyperparameters of our model, and adding more features.

**ACKNOWLEDGEMENTS**

I wish to express my utmost gratitude to Professor Pierluigi Nuzzo, whose unwavering kindness and infectious enthusiasm have revealed the sheer excitement and captivating nature of pursuing one's passion.

I am forever indebted to Kaixin Yang, my esteemed mentor, whose invaluable guidance and unwavering support have fundamentally transformed my life and bestowed upon me a fresh perspective.

And Thank you to Matthew Ai and Marcus Gutierrez, who have always demonstrated their support and availability.