University of Sheffield

Reconfigurable Security for IoT Application in Handling Machine-Learning/Modelling Attacks

Cheng-Wei, Tsao

Supervisor: Dr Prosanta Gope

A report submitted in fulfilment of the requirements
for the degree of BSc in Computer Science

in the

Department of Computer Science

December 4, 2021

Declaration

All sentences or passages quoted in this report from other people's work have been specifically acknowledged by clear cross-referencing to author, work and page(s). Any illustrations that are not the work of the author of this report have been used with the explicit permission of the originator and are specifically acknowledged. I understand that failure to do this amounts to plagiarism and will be considered grounds for failure in this project and the degree examination as a whole.

Name: Cheng-Wei, Tsao

Signature: Cheng-Wei, Tsao

Date: November 15, 2021

Abstract

This report investigate and provide clear introduction on PUF(physical unclonable function), aim of the project and current progress. In introudction, there are thoroughly description on PUF, reconfigurability framework and concepts corresponding to specific machine learnings for modeling attack on PUF.

The aim for the project is to propose a novel, suitable machine learning to model PUF(physical unclonable function) and then design a reconfigurability framework to fight against such attack. The PUF structure is similar to a road network so machine learning related to ETA(estimated time arrival) problem is strongly considered.

The achievements to date are having robust understanding on PUF, reconfigurability property, and attempt to implement reinforcement learning as modeling attack. SARSALambda Q learning reinforcement learning has been tested on PUF but according to false implementation, only 50% accuracy has been achieved. The Actor-Critic reinforcement learning are considered to be useful at the moment, while further research are progressing to validate the usability.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In the rapid development of the information era, many daily events are achieved by a various of electronic devices such as computer or phones. Those electronic devices highly rely on integrated circuits(ICs) to perform specific events. For example, bank transaction can be done by different devices, the process contain personal data, and including usage of sensitive information. Therefore, information security like authentication, protecting confidential data has become important in nowadays society. In order to increase security's robustness, a range of ways has been proposed. One conventional way to is by storing secret key in non-volatile memory to encrypt sensitive data with it, and use asymmetric cryptography to authenticate the device [9]. However, the implementation process of cryptography is expensive, especially on resource-constraint device, and the device is still vulnerable to invasion attack. Ideally, devices should be able to handle challenging problems corresponding to energy consumption, computational power and the ability to fight against cyber attack.

PUF(physical unclonable function) has the ability to deal with these challenges. It does not store secret in non-volatile memory, instead, the volatile secret is derived from devices' physical characteristics [9]. This is based on the inevitable random variation in ICs manufacturing process, which leads to the fact that no two IC have exact same physical characteristic. For example, each ICs has unique delay sequences in the transistors and wires. With this property, PUF does not require lots of computational power and is cost-effective because no need to implement cryptographic operation, which works particular well on resources-constraint devices such as RFID. Also, the attacker needs to perform attack when the device is on, which significantly increase the difficulty. As for invasion attack, the attacker needs to have the exact information of its unique physical characteristics to successfully derive secret. Overall, PUF provides another interesting way for reinforce security.

1.1 Aims and Objectives

The objectives split into two parts for this project. In the first stage, propose a novel machine learning to modeling different PUF(physical unclonable function) behavior, so predict the response from a given PUF when given challenge bits. For example, considered the simplest PUF which is arbiter PUF, its operation to create a response is to input a challenge bits(binary), and two signals will go thorough the multiplexers in the PUF structure depend on the value of it. Consequently response a binary bit that will indicate which signal is faster. Therefore, the machine learning for modeling will be related to ETA(estimated time arrival) problem since the structure of PUF is similar to a road network. For instance, traveling through each multiplexer is similar to traveling through each road segment, and both of them have delay to affect the time of arrival. Overall the first stage is to design a machine learning consider these concepts. In the second stage, design a reconfigurability framework to fight against such modeling. In detail, evaluate the machine learning by insert noise in PUF or experiment on OPUF(one-time-PUF) which contain reconfiguration process that can alleviate modeling attack.

1.2 Overview of the Report

The remaining of the paper will organize as follow. Chapter 2 provide literature survey of the concept of PUF, including PUF's properties, detailed circuit structure and authentication process, exist modeling attack with experiment results, and reconfigurability framework. Chapter 3 describe the aim and objectives for the project, gives in-depth analyzes how the project will be evaluated, the tests and experiments that support this. Chapter 4 demonstrate the current progress along with explanation on the project. Chapter 5 provide brief summary on the main achievements with a well organized future plan for the project.

Chapter 2

Literature Survey

2.1 The PUF concept

The simplest sentence to describe PUF is "A PUF is an object's fingerprint" [5]. The fingerprint can represent a specific human in the world, such as the PUF can represent an object. The fingerprint is inherently created when people was born, and the so does PUF, which is inherently exist in an object according to unique manufacturing random variation [5]. With the representation and inherent property, the fingerprint and the PUF is said to be unclonable since it is impossible to control and predict human's fingerprint. This is an important concept for PUF.

This intrinsic property can be extract from chip which has PUF circuit existed inside [1]. The way PUF works is by entering a certain length of bits(so called challenge) into the PUF, and it will generate another specific length of bits(so called response). According to the property of PUF that was discuss above, it is impossible to find two different PUF that will produce the same response when entering same challenge(See Figure 2.1).

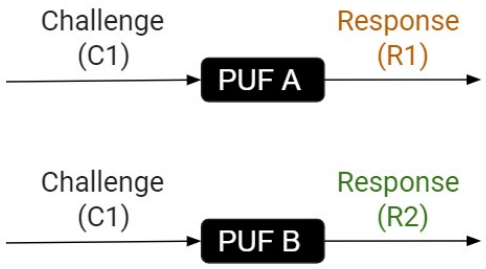

Figure 2.1: Different PUF that generate different response when input same challenge

2.2 Weak and strong PUF

PUF can be classified into two categories, weak and strong PUF according to the strength of PUF. The strength of PUF indicate the number of challenge response pairs(so called CRPs) can be generate from the PUF [6]. The higher numbers of the CRPs can a PUF generate, the better strength it has. Generally, if increasing the size of the PUF leads to a linear increase in the number of CRPs, it is consider weak PUF. On the other hand, if increasing the size of the PUF leads to a exponential increase in the number of CRPs, it is consider strong PUF.

For the weak PUF, it represent the PUF that has smaller set of CRPs. While it is impossible to create a clone of PUF, but with small set of CRPs, this will allow attacker to record all the CRPs when attacker has physical access to PUF [6]. With the knowledge of CRPs, attacker can easily provide the corresponding response to challenge as like they have a clone(See Figure 2.2). The weak PUF can be use for authentication and key storage. However, since weak PUF's CRPs can be fully access, ensure having a secure environment and whether the original PUF is being evaluating is relatively important [6].

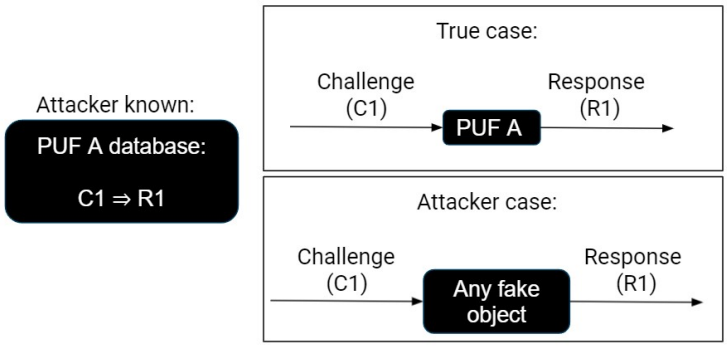

Figure 2.2: Attacker can perform same behavior as Weak PUF when have fully access to CRPs and not under secure environment

For strong PUF, means the number of CRPs is significantly large that even attacker get access, having throughout knowledge of CRPs is impossible. While the number of CRPs is so large, and the CRP are randomly selected in usage, the probability that attacker has knowledge about the CRP currently using is small. In addition, each CRPs that is used once will be discarded(See Figure 2.3) so even if attacker recorded certain CRPs, also called eavesdropped, they will not be able to put them into use. The strong PUF can also be use for authentication but do not need to protect CRPs as serious as weak PUF.

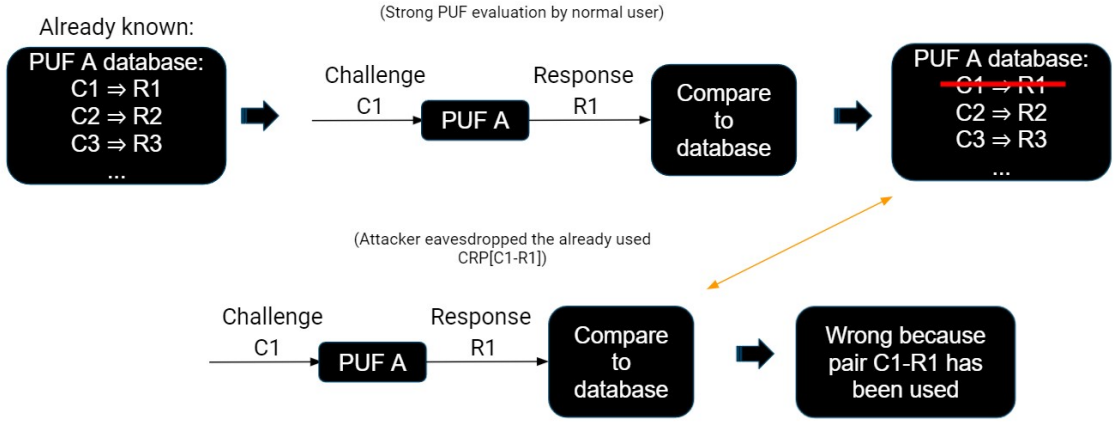

Figure 2.3: The attacker eavesdropped CRP that has been used can not successfully validate in next evaluation for strong PUF

2.3 Authentication

One of the application of PUF is authentication. As discuss in Chapter 1's introduction, PUF does not require huge computational power and are cost effective, so it is suitable for many devices, especially the resources-constraint devices. The PUF's authentication included two stages, enrollment and authentication stage. In the enrollment stage, the company possess the PUF, so company can connect server to PUF and sent lots of challenges along with recording the CRPs into the database [1] (See Figure 2.4).

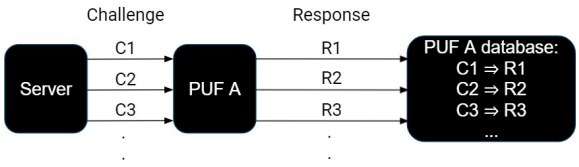

Figure 2.4: Enrollment stage in PUF authentication

After recording all the CRPs, the company can now implement PUF on electronic devices. In the authentication stage, the server sent arbitrary challenge to the devices that contain PUF while the device will return response. Afterward, the server compare the response from the device with the database, if the challenge and response pair exist in the database, the device is valid [1] (See Figure 2.5). A life example will be banking card.

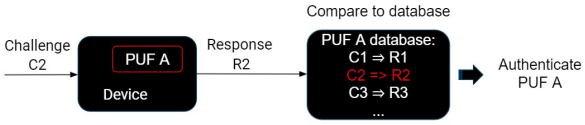

Figure 2.5: Authentication stage in PUF authentication

2.4 Arbiter PUF and XOR arbiter PUF

There are many different types of PUF such as arbiter PUF, ring oscillator PUF, lightweight PUF, etc. In this paper, arbiter PUF and its mutation will be introduced in detail. The general idea of the arbiter PUF is comparing the transition speed for two electrical signal in the PUF's structure(See Figure 2.6). The arbiter PUF's structure contains a numbers of multiplexers and a arbiter(mostly D flipflop), and two multiplexers will combined into a switching box [9]. Look at Figure 2.6, when enter a challenge bits, apply each challenge bit to a switching box, bit 1 indicate the upper and lower signal will switch while bit 0 indicate the two signals remain unchanged in each switching box. This will eventually form paths for the signals. Then the signals start transferring, the time arrived at the arbiter for two signals is different since each multiplexer and wire has unique delay. The arbiter will determine which path is faster and based on that response a binary bit, if upper path is faster, the response is 1, otherwise the response is 0.

The arbiter at the end is always fair, which will not favor any one of the path. Even there exist bias, a simple solution of adding a delay in the structure can solve the problem. For example, if the arbiter favor the lower path, by adding a delay to upper path, it can has a head start. By looking at the Figure 2.6, it is clear that the CRPs will be exponential. Assuming there are n switching boxes, two possible cases in each switching box, so the number of CRPs is 2^n, which indicate the arbiter PUF is a strong PUF. Arbiter PUF can also return longer response by input K different challenges and get a K bits response.

The normal arbiter PUF is vulnerable to modeling attack, the propose of XOR arbiter PUF is to increase the robustness of arbiter PUF. The basic concept is to integrated multiple parallel arbiter PUF, given same challenge to each arbiter PUF and XORed each response to produce final response(See Figure 2.7) [8]. According to simulation, in book [5] provides the XOR arbiter PUF will have nonlinearity property that makes it harder to model.

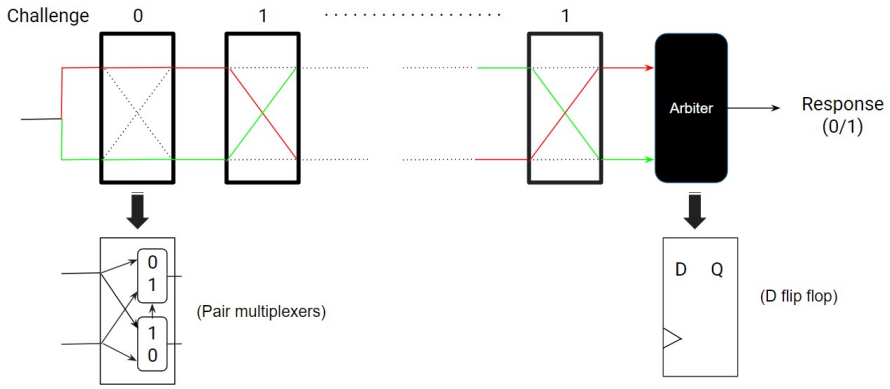

Figure 2.6: Arbiter PUF structure

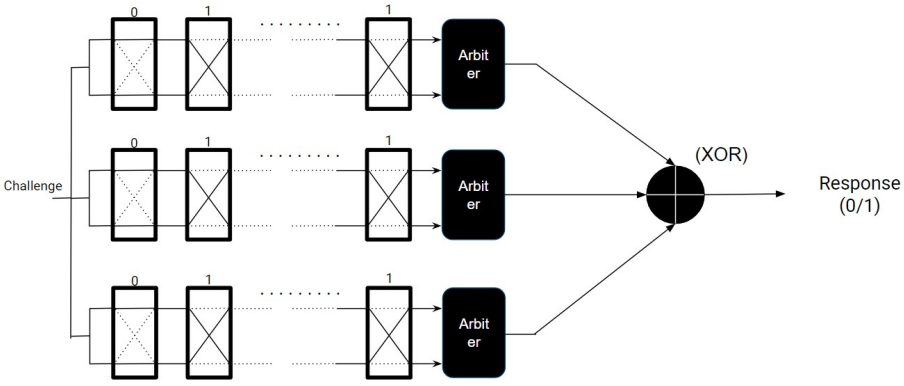

Figure 2.7: XOR arbiter PUF structure

2.5 Modeling attack on PUF

Many different threat can perform on devices, such as eavesdropped, gain access to the memory that store secret keys. For PUF, the main threat is that attacker can use technique like machine learning to simulate the behavior of CRPs(so called modeling), which means even without the devices, attacker can still response correctly when a challenge is provided. Take arbiter puf as example, assume an arbiter PUF with i switching box, and the challenge apply to each switching box is c[i]. The two signal travel through the the path determine by challenge, and arrive at the arbiter in different time because of the delay in each component. The final response depend on the sign of final delay difference:

$$r = \begin{cases} 1, & \text{if } \Delta c < 0. \\ 0, & \text{otherwise.} \end{cases} \tag{2.1}$$

which the delay difference is calculated by subtract the upper path with lower path's delay. The final delay difference Δc can represent as $w^T\Phi$, where w^T is a weight vector that represent delays for the components in PUF, and Φ is the applied i bits challenge [8]. $w^T\Phi = 0$ will provide a hyperplane that separate the space for Φ, one side of the hyperplane are predict as having response 1, the other side of the hyperplane are predict as having response -1. In conclusion, the correct hyperplane indicate the good prediction of PUF. Machine learning such as logistic regression can play the role well. The modeling result for a arbiter PUF with 64 switching boxes, by using the logistic regression can get a good performance of 99.9% in very short time with 18050 training CRPs [7].

For the XOR arbiter PUF, it is also possible to use logistic regression to predict the behavior but will be harder.

2.6 Reconfigurability of PUF

In order to alleviate the problem that PUF is vulnerable to modeling attack, reconfiguration property embedded on PUF has been proposed. For example, the one-time-PUF(so called OPUF), the general idea is that the its configuration alter after every authentication session, which means CRPs behavior of the PUF is changed and invalidate the modeling attack. This is based on the forward unpredictability and backward unpredictability properties that OPUF possess [3]. In Figure 2.8 shows the concept of forward unpredictability and backward unpredictability. The orange line represent the forward unpredictability, which will ensure the responses collected before the reconfiguration process is invalid afterward. As for the blue line, which represent backward unpredictability, it will ensure the pattern observed by attacker using modeling attack on PUF' is invalid for predicting the PUF before reconfiguration

process. In this case, assume an attacker are performing a modeling attack on the PUF, and required certain amount of CRPs to gain proper prediction. However, the OPUF's structure keep changing every execution, which means the CRPs behavior is changing, so the modeling attack will can't be up to date or do mot have time to collect enough CRPs.

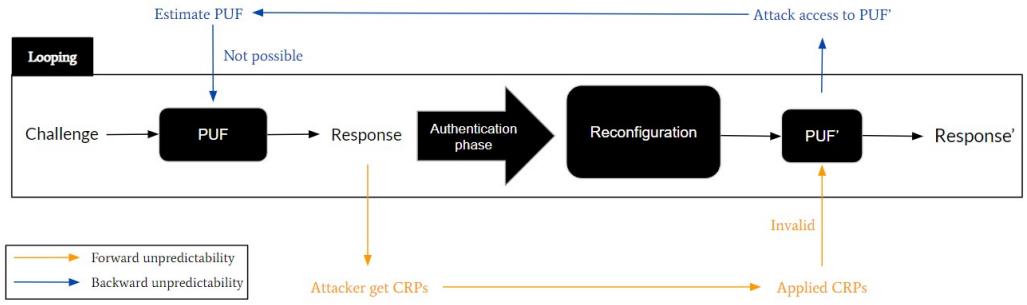

Figure 2.8: Concept of forward unpredictability and backward unpredictability

DPUF can be consider as an example for creating the OPUF, it is build up with bit cells which contain capacitors and transistors, each cell store information of value 0 and 1. However, these component will leak electrical signals every period of time, which means the behavior might be eavesdropped [3]. Therefore performing reconfiguration process every period of time is effective. The reconfigurability can be shown in Figure 2.9, by varying parameters such as refresh-pause interval and the memory block, where the former can cause random bit flip in the cells while the latter store final response in an random memory block which will alter every period of time, PUF can present unpredictable behavior that prevent from attacker perform modeling attack.

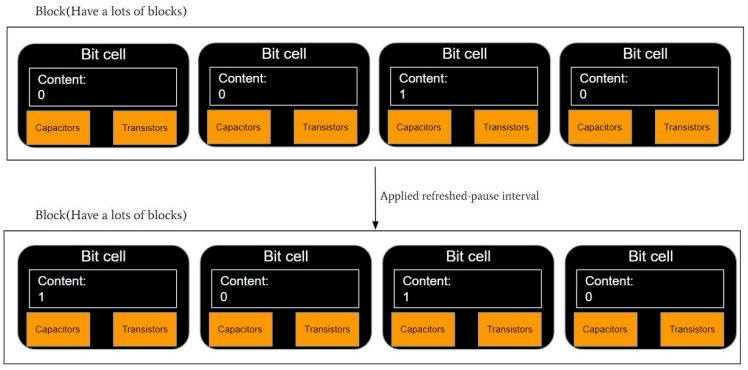

Figure 2.9: Reconfigurability framework of DPUF

2.7 Summary

In summary, chapter 2 describe PUF as a novel approach to increase the security robustness of electrical devices by making use of its unique physical characteristics and how it can be used for devices authentication. Also, the working process and concepts for arbiter PUF and XOR arbiter PUF is introduced in detail. Last, largest potential threat on PUF called the modeling attack and its countermeasure are analyzed in details.

Chapter 3

Analysis

Overall, total two objectives are required for the project. In the first part, propose a novel machine learning to modeling different PUF(physical unclonable function) behavior, in particular arbiter PUF and XOR arbiter PUF. In the second part, consider reconfigurability like adding noise, one-time-PUF to resilient against the modeling attack proposed in part one.

3.1 Project Aims and Evaluation

The aims for part one are that the machine learning for modeling should be related to ETA(estimated time arrival) problem or the one people haven't used for modeling PUF. Moreover, reconfigurability need to be considered when proposing the modeling attack while knowing the PUF can escape from it by changing CRPs behaviors in part two. Therefore, the machine learning should ideally adapt to the changing behavior of CRPs or the PUF. For example, unsupervised learning can be a good way since it can deal with unseen data. To evaluate the work, an overall accuracy of the modeling are expected to be around 90% and take short time like the logistic regression in paper [7].

The aims for part two are proposing a reconfigurability framework for the modeling attack in part one. Two different reconfigurability will be implemented. First, add intentional noise to PUF's response during authentication phase to disturb the modeling attack [4]. The basic idea is shown in Figure 3.1, assume database has stored PUF's CRPs and provide a challenge c to PUF in authentication phase. The PUF return a response r', and add noise e with r'. Then e and r' compose helper data h and send back to verifier, where $h = r' \oplus e$. The verifier can reproduce $r' \oplus e$ with r and h if r and $r' \oplus e$ is similar. Last, both verifier and user create

their own hashtag and compare to check if authenticate. In this case, attacker only access to c and $r' \oplus e$, which the modeling will not be accurate unless the attacker's model response is closed enough to r'. To evaluate the work of adding intentional noise, the modeling attack proposed in part one should have reasonable accuracy drop and the authentication will still work though noise is added.

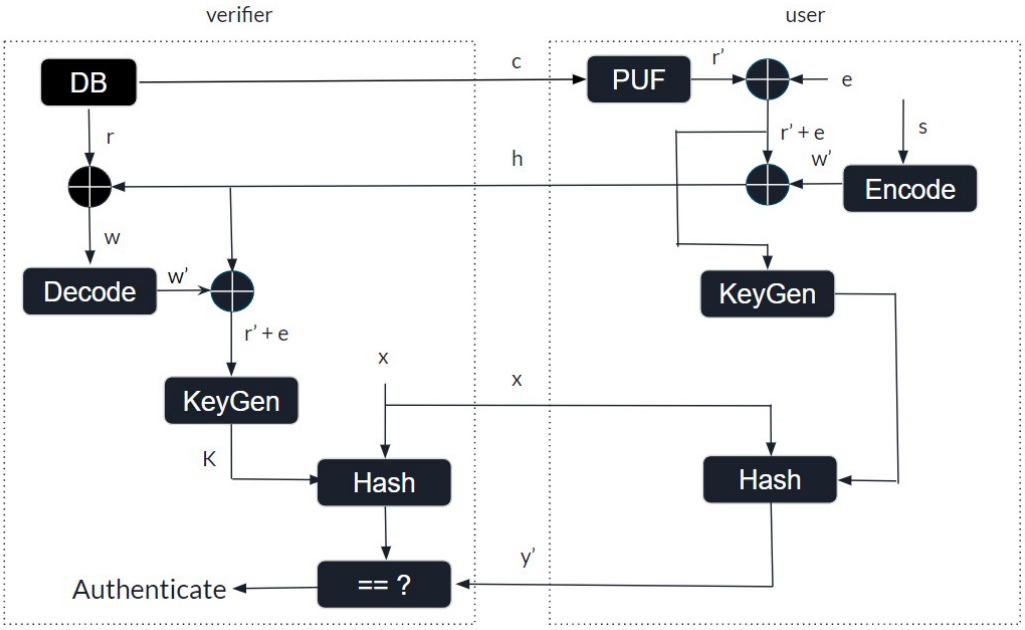

Figure 3.1: Add noise to PUF's response during authentication phase

Second, implement protection which referring to OPUF to change the behavior of CRPs and prevent modeling attack. With control of parameter like 1. Refreshed-pause timing 2. Allocating memory block, the project will examinate the result of the protection. For example, look at the accuracy and time consumed when apply the modeling attack in part one.

Chapter 4

Progress

4.1 Project progress

The project has two part, currently the progress are at the point of discovering and implementing machine learning for modeling the arbiter PUF. Methods such as reinforcement learning and graph attention neural network(so called GAT) were used. First, the type of reinforcement learning that was implemented is the SARSALambda Q learning [**?**]. The general idea is that an agent will explore the environment with a final goal by randomly choosing actions space and recording the rewards.

Assuming Figure 4.1 is the PUF environment, red and green circle is the starting point for top path and bottom path, each black rectangle represents a multiplexer with unique delay and the yellow circle is the goal. There are three actions: going up, going down and going straight. The reward of multiplexer is determined by the delay, bigger the delay, the smaller the reward, vice versa. In training phase, the agent will travel through different combination of multiplexers and construct a Q table(See Table 4.1) by collecting reward on multiplexer. The final gaol for the agent is to find the route with lowest delay. After the training, when select a CRPs, and input the challenge to the agent, ideally, the agent can calculate, compare two paths' reward and reply the correct response. For example, assume an challenge 00001 has response 1(which means bottom path is faster), the calculation operation is:

$$Challenge(00001) = \begin{cases} 0.082 + 0.008 + 0.041 + 0.002 + 0.07 = 0.203, & \text{Top path: 1,3,5,7,10.} \\ 0.022 + 0.415 + 0.222 + 0.555 + 0 = 1.214, & \text{Bottom path: 2,4,6,8,9.} \end{cases} \tag{4.1}$$

$$0.203 < 1.214, \text{ return response: } 1. \tag{4.2}$$

In general, the route with lower delay will have higher reward, on the other hand, the route with higher delay will have lower reward. If the agent can predict with high accuracy, the arbiter PUF's CRPs pattern can said to be successfully modelled. However, the accuracy for this modeling is around 60% - 69%, which is not a satisfying result. The problem can be the following: not providing enough features, the exploration does not cover every possible routes, etc.

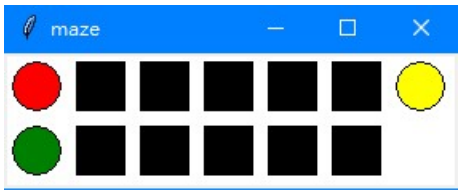

Figure 4.1: SARSALambda Q learning example environment

Multiplexer	go up	go down	go straight
1	0.000	0.028	0.082
2	0.094	0.000	0.022
3	0.000	0.357	0.008
4	0.009	0.000	0.415
5	0.000	0.181	0.041
6	0.042	0.000	0.222
7	0.000	0.641	0.002
8	0.003	0.000	0.555
9	0.000	0.070	0.000
10	0.000	0.000	0.131

Table 4.1: Q table for example environment

As for the GAT [2], the basic idea of the GAT is that each node aggregate the neighbors' features based on adjacency matrix, adjacency attention value then update each node's features. The updated features will insert into the neural network and perform classifying task. In order to reach to high accuracy, the main task for the GAT is to update adjacency attention matrix to right value.

For a detail example, look at Figure 4.2, which is a arbiter PUF structure. Assume each nose in Figure 4.2 represent a multiplexer, and has node feature $f1, f2, f3, f4$. If input a challenge 00, a one way relation is defined: $m3 \rightarrow m1$, $m4 \rightarrow m2$, including self-loop, and

the adjacency matrix can be constructed(See 4.3).

$$\begin{array}{c} \begin{array}{cccc} M1 & M2 & M3 & M4 \end{array} \\ \left(\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{array} \right) \begin{array}{c} M1 \\ M2 \\ M3 \\ M4 \end{array} \end{array} \tag{4.3}$$

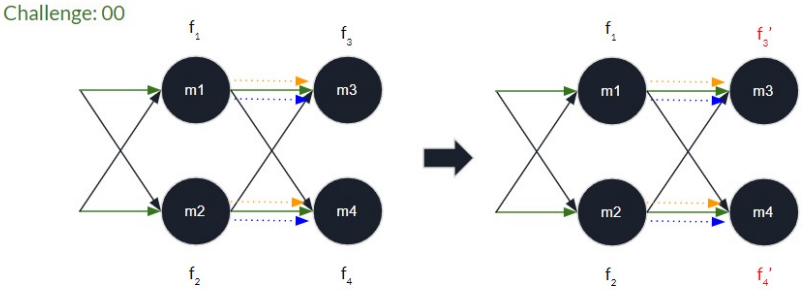

Figure 4.2: GAT aggregation

In article [2], there are four steps to aggregate the features of each node. The first step, add a weight matrix to gain enough expressive power to transform the features into higher level features:

$$n_i = \mathcal{W}_i f_i \tag{4.4}$$

The second step, calculate a un-normalized attention value between every two nodes(i and j). First, concatenates the features of the two nodes(symbol $\|$ represent concatenation), then perform a dot product with a trainable weight vector a, and applies a LeakyReLU at last:

$$c_{ij} = LeakyReLU(\overrightarrow{a}^T(n_i \| n_j)) \tag{4.5}$$

The third step, apply Softmax function to normalize:

$$\alpha_{ij} = \frac{exp(e_{ij})}{\sum_{k \in N_i} exp(e_{ik})} \tag{4.6}$$

The fourth step, aggregate the updated features from neighbors to current node, and consider the attention matrix:

$$f_i^{'} = \sigma(\sum_{j \in N_i} \alpha_{ij} n_j) \tag{4.7}$$

After updating every node's features, concatenates the node features of top path($f_1 + f_3^{'}$)and bottom path($f_2 + f_4^{'}$), and insert into a classify layer. Next, compare the output with ground fact (response of the challenge) to update the parameter to increase the prediction rate. Ideally, the adjacency matrix will keep updating according to different input challenge, and the GAT will be able to learn the pattern after looking at many sample. However, there are doubt that whether the GAT support dynamically changing adjacency matrix, so this idea is not done yet.

Chapter 5

Conclusions

5.1 Project Plan

In the future, XGBoost was a new idea to try if both the previous machine learning fail to model the arbiter PUF. However, more research on XGBoost was required.

5.2 Summary of the project

Bibliography

[1] Babaei, A., and Schiele, G. Physical unclonable functions in the internet of things: State of the art and open challenges. *Sensors*, 14 (2019).

[2] Dagar, A. Understanding graph attention networks (gat).

[3] Gope, P., Sikdar, B., and Millwood, O. A scalable protocol level approach to prevent machine learning attacks on puf-based authentication mechanisms for internet-of-medical-things. *IEEE Transactions on Industrial Informatics* (2021), 1–1.

[4] Lanet, J.-L., and Toma, C. *Innovative Security Solutions for Information Technology and Communications.* Springer, Cham, 2018.

[5] Maes, R. *Physically Unclonable Functions.* Springer, Berlin, Heidelberg, 2013.

[6] McGrath, T., Bagci, I. E., Wang, Z. M., Roedig, U., and Young, R. J. A puf taxonomy. *Applied Physics Reviews 6*, 12 (February 2019).

[7] Rührmair, U., Sehnke, F., Sölter, J., Dror, G., Devadas, S., and Schmidhuber, J. Modeling attacks on physical unclonable functions. 237–249.

[8] Santikellur, P., Bhattacharyay, A., and Chakraborty, R. S. Deep learning based model building attacks on arbiter puf compositions. *IACR Cryptol. ePrint Arch. 2019* (2019), 566.

[9] Suh, G. E., and Devadas, S. Physical unclonable functions for device authentication and secret key generation. 9–14.