3 Machine Learning Attack on Arbiter PUF

Main Objective In this project, you will learn that PUFs are not perfect! In particular, you will perform a machine learning (ML) based attack on arbiter PUFs.

Background Research shows that arbiter PUFs can be modeled with certain machine learning techniques [1], [2]. If adversaries can accurately model a PUF, they can predict its challenge-response pairs and impersonate the PUF, thereby breaking the security of protocols that rely on the response is unpredictable.

In the simplest form, a machine learning-based modeling attack proceeds as follows:

1. The attacker learns some number of challenge-response pairs for the PUF. This could happen if, for example, a manufacturer remotely authenticates its chips on an unencrypted channel.
2. The attacker must know the PUF architecture.
3. A mathematical model is constructed of the PUF, and parameters of the model are derived (or, equivalently, "the model is trained") using known challenge-response pairs.
4. New challenges can now be supplied to the trained model (which now acts as the software clone of the original PUF) to predict the real PUF's responses.

More efficient modeling attacks accurately model a PUF's responses given a small number of challenge-response pairs.

Project Goal

1. For the given CRP dataset (CRPSets.zip), implement two machine learning techniques for predicting PUF responses given previously unseen inputs (SVM: [1,3], regression: [2]). In other words, train your model using the different training/testing splits as described in "Results to submit".
2. Evaluate your model. Use CRPs that were not in your training set to determine how well your model predicts the PUF output for inputs that it has not previously seen.

Software Requirements

- Any software and scripting language capable of performing numerical analysis and building machine learning-based trainers and classifiers (MATLAB, R, Python, C/C++).
- CRPSets.zip contains (12000x65) data for a 64-stage arbiter-PUF with 12000 challenges, as rows indicate. Each challenge is 64-bit long (Columns 1-64), and the corresponding response is one bit, 0/1, given in column 65.

Report to submit.

- IEEE conference format. (4-6 pages)
- The report should include (tentative marks dist.):
    1. Top Sheet with group members' names and ID
    2. Introduction, motivation, and problem statement. (10%)
    3. Prior techniques (5%)
    4. Methods: model selection and design (40%)
        - Implementation of PUF modeling attack
        - Reason of choice, brief description, pros/Cons
        - Working flow-chart/algorithm
        - Used boundary conditions, parameters, etc. as necessary.

5. Results (35%)
   - Evaluate the impact of training set size on the accuracy of your PUF model. What is the smallest number of CRPs you could observe that would allow your model to predict PUF responses with 80% accuracy?
     * 2000, 6000, and 10000 samples (total of 12000 challenges/samples available)
     * All remaining samples are used for evaluation.
   - Compare different attack models (Grad only)
   - Tables, plots, etc., summarizing the accuracy of your modeling attacks.
   - Compare your modeling results with those from [1,2].
6. Conclusion and personal comments (10%)
7. Clarity, organization, etc. (5%)

Submission guidelines

- Submit a .zip file with name: Project#Group#
- Include all of the following:
  - Report.pdf
  - Working directory for your analysis
    * readme.txt (provide necessary instruction for running your code)
    * All source files.
    * Data set

References

1. Rhrmair, Ulrich, et al. "PUF modeling attacks on simulated and silicon data." IEEE Transactions on Information Forensics and Security 8.11 (2013): 1876-1891.
2. Rhrmair, Ulrich, et al. "Modeling attacks on physical unclonable functions." Proceedings of the 17th ACM conference on Computer and communications security. ACM, 2010.
3. Lim, Daihyun. Extracting secret keys from integrated circuits. Diss. Massachusetts Institute of Technology, 2004.