ML-Based DGA Detection Evasion

1. Introduction

* Overview:

DGA\_Detection by miaWallace0681 hosted on [GitHub](https://github.com/alexdevassy/Machine_Learning_CTF_Challenges.git) was selected for the local implementation case study. It uses ensemble method and Long Short-Term Memory(LSTM).

* Purpose:

To bypass the ML-based DGA Model and map the attack to the Mitre ATLAS Matrix.

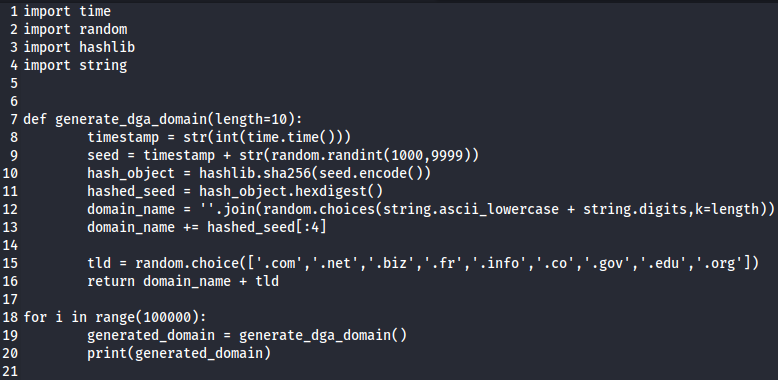
1. Testing Methodology

* Test Plan: Bypass the DGA\_Detection model by exploiting the API Inference and Input manipulation.
* Tools Used: Kali Linux Tools, Postman, ChatGPT
* Testing Techniques/Kill Chain:

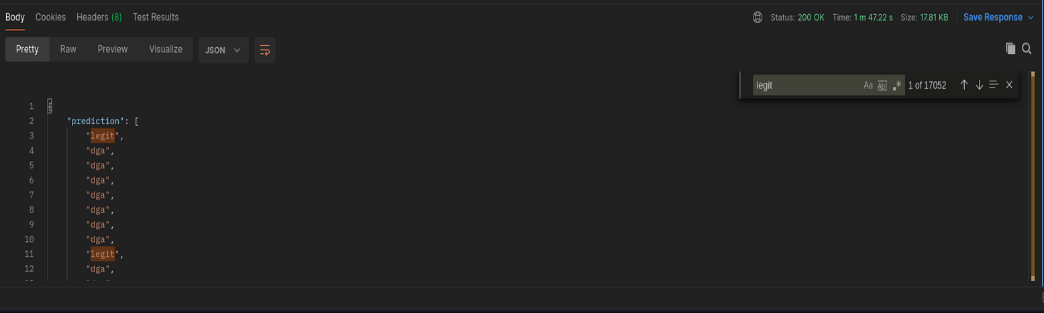
1. Test Cases

Test Case 1:

* **Technique:** AML.T0043.001 - Craft Adversarial Data: Black-Box Optimization
* **Description:** Design a DGA that can generate nearly undetectable domain names using current time, randomness, and a hash.
* **Steps:**
* Create a DGA program.



* Store the output in a .txt file and input it to the model through Postman API calls.
* Receive the output, analyse the results and note down accuracy of the model.



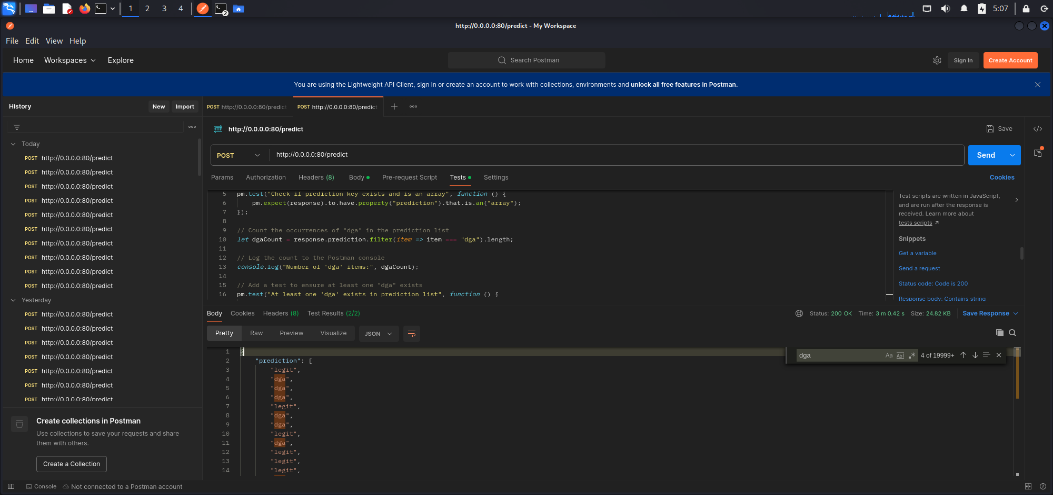
* Improve the DGA program.
* **Expected Outcome:** DGA\_Detection model shows reduced accuracy.
* **Actual Outcome:** This modification reduced the algorithm’s accuracy to 82.948% for a dataset containing 1,00,000 DGA generated domain names. (17052 out of 1 lakh DGA generated domains were classified as Legit)
* **Vulnerability Risk Level:** Moderate

Test Case 2:

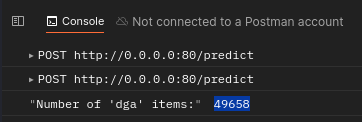
* **Technique:** AML.T0043.001 - Craft Adversarial Data: Black-Box Optimization
* **Description:** Design a DGA that can generate nearly undetectable domain names using current time, randomness, and a hash appended with ‘-safe’ or ‘-secure’ keyword.
* **Steps:**
* Modify the DGA program to append ‘-safe’ or ‘-secure’ at the end before TLD.

Image of additional code to be added in Source code


* Store the output in a .txt file and input it to the model through Postman API calls.



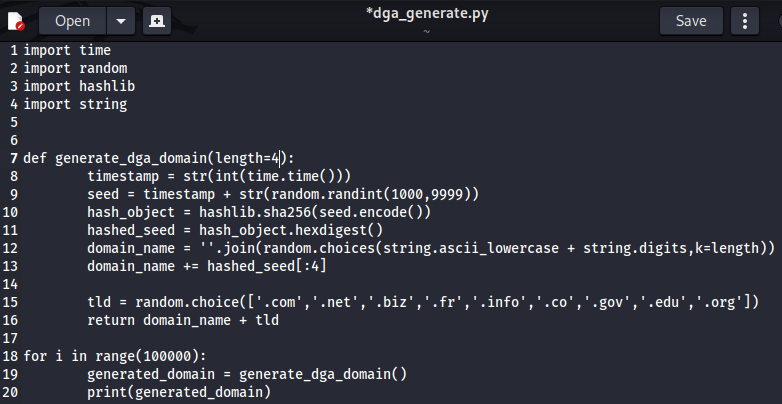
* Receive the output, analyse the results and note down accuracy of the model.



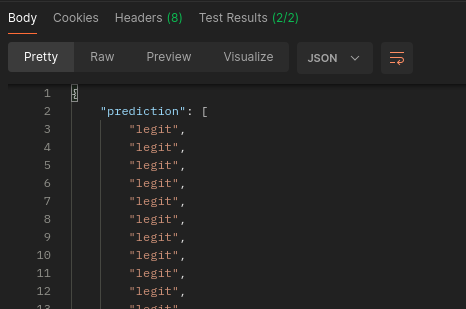
* Improve the DGA program.
* **Expected Outcome:** DGA\_Detection model shows reduced accuracy.
* **Actual Outcome:** This modification increased the algorithm’s accuracy to 49.658% for a dataset containing 1,00,000 DGA generated domain names. (50342 out of 1 lakh DGA generated domains were classified as Legit)
* **Vulnerability Risk Rating:** High

Test Case 3:

* **Technique:** AML.T0043.001 - Craft Adversarial Data: Black-Box Optimization
* **Description:** Design a DGA that can generate nearly undetectable domain names using current time, randomness, and a hash with length constraint set to 8 or less.
* **Steps:**
* Modify the DGA program to domain names of length 8 excluding TLD.



* Store the output in a .txt file and input it to the model through Postman API calls.



* Receive the output, analyse the results and note down accuracy of the model.



* Improve the DGA program.
* **Expected Outcome:** DGA\_Detection model shows extreme reduction in accuracy.
* **Actual Outcome:** This modification significantly decreased the algorithm’s accuracy to 0.061% for a dataset containing 1,00,000 DGA generated domain names. (93,900 out of 1 lakh DGA generated domains were classified as Legit)
* **Vulnerability Risk Level:** Critical

1. Conclusion

* The locally implemented case studies on ML-based DGA detection evasion demonstrate how adversarial techniques can bypass machine learning security models. Through a structured attack methodology, multiple test cases were executed, showing a progressive decrease in model accuracy from 82.94% to an alarming 0.061%—revealing a critical vulnerability in the detection mechanism.
* These findings highlight the importance of continuous model evaluation, adversarial training, and robust defensive mechanisms to prevent AI-based cybersecurity solutions from being exploited. The study reinforces the need for adaptive security measures that can dynamically respond to evolving adversarial threats.
* Moving forward, integrating stronger anomaly detection, real-time monitoring, and more resilient model architectures will be essential in mitigating these risks and ensuring AI security solutions remain reliable against adversarial attacks.