DOOM

A Novel Adversarial-DRL-based Op-Code Level Metamorphic Malware Obfuscator for the enhancement of IDS

UBICOMP 2020

Mohit Sewak

Sanjay K. Sahay

Hemant Rathore

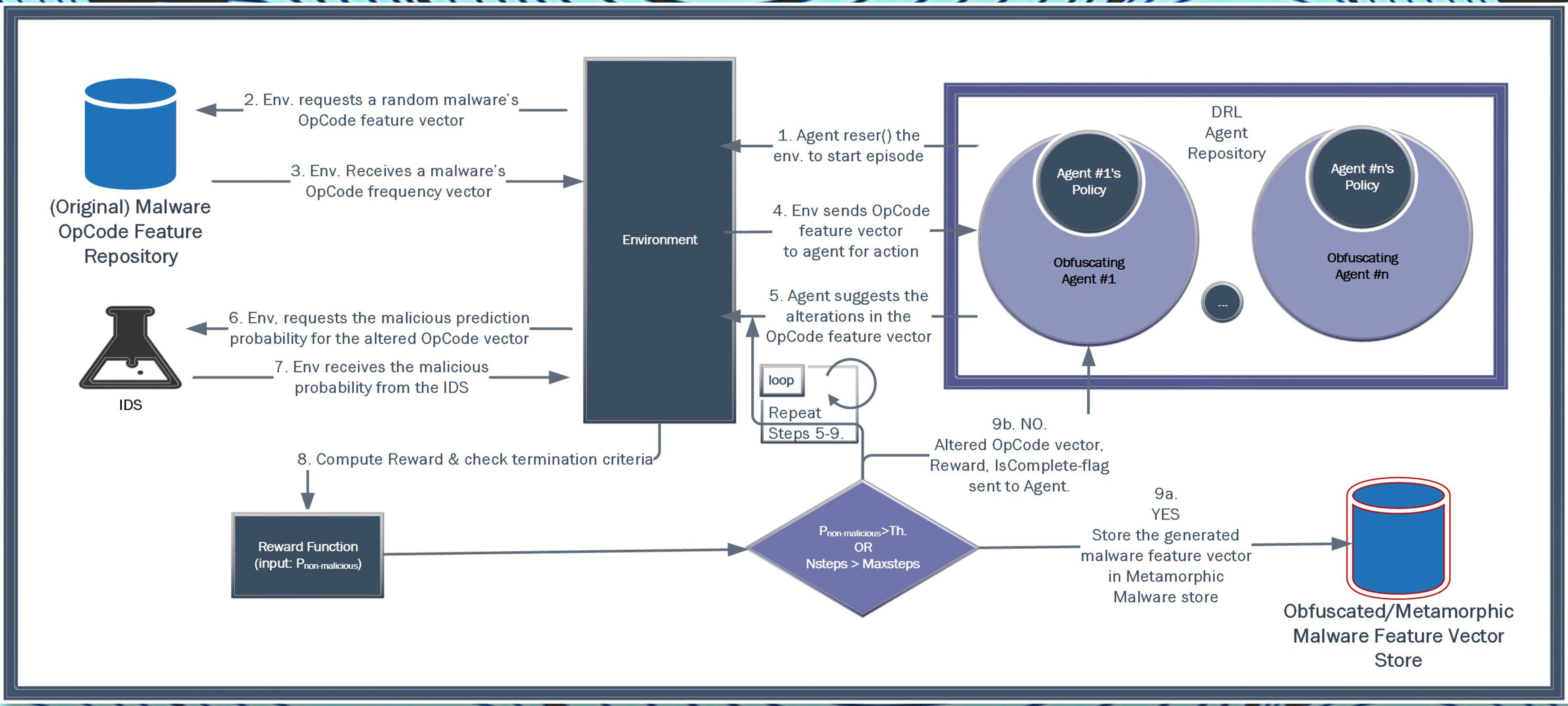
ABOUT DOOM

- DOOM stands for Adversarial-DRL based Op-code level Obfuscator to generate Metamorphic malware.
- DOOM is a novel system that uses adversarial deep reinforcement learning to obfuscate malware at the op-code level.
- The ultimate goal of DOOM is not to give a potent weapon in the hands of cyber-attackers, but to create defensive-mechanisms against advanced zero-day attacks.
- Experimental results indicate that the obfuscated malware created by DOOM could effectively mimic multiple-simultaneous zero-day attacks.
- To the best of our knowledge, DOOM is the first system that could generate obfuscated malware detailed to individual op-code level.
- DOOM is also the first-ever system to use efficient continuous action control based deep reinforcement learning in the area of malware generation and defense.
- Experimental results indicate that over67%of the meta-morphic malware generated by DOOM could easily evade detection from even the most potent IDS.
  - This achievement gains significance, as with this, even IDS augment with advanced routing sub-system can be easily evaded by the malware generated by DOOM.

USES OF METAMORPHIC OBFUSCATIONS GENERATED BY DOOM

The op-code level obfuscations generated by the DOOM can be used for:

- Improving the IDS's classifier against new or metamorphic variants of existing malware.
- Training/ augmenting other internal sub-systems of the IDS with the capability to de-obfuscate the incoming file's features vector before sending it to the IDS's classifier.
- Creating/ training other external sub-systems for normalizing obfuscations of different variant of existing malware. This can augment any existing IDS with metamorphic detection capabilities without warranting any changes.



RELATED WORK AND GAPS IN LITERATURE

- There has been many attempts to generate obfuscations at the code level [1], but these are not scalable.
- Later efforts were also made to use machine-learning (ML) models [2] to automate the obfuscation mechanism. However, these ML methods does not effectively replicate the advanced metamorphic attack required to train an adversarial mechanism.
- There has been attempt to use CNN based GANs [16],[6],[7],[15] as well. But the adversarial-learning produced from such mechanisms is immune to secondary gradient-attack.
- Recently DRL [12],especially Q Learning has been utilized [18] to alter the binary code of the file to evade attacks. Systems based on creating perturbations at binary-code level are not only limited to very small action-space of MDP (limited to adding some specific 4-bit code in [18]), and also can not be used to mimic an actual malware, because it require a code/op-code level obfuscation.

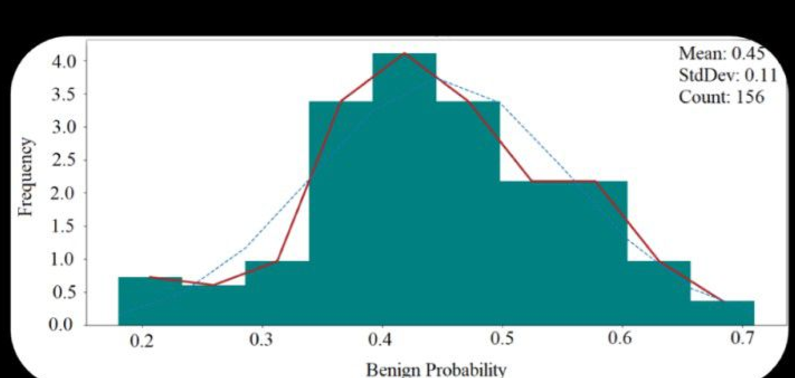
DOOM'S ARCHITECTURE

DOOM's architecture broadly consists of four subsystems, which are

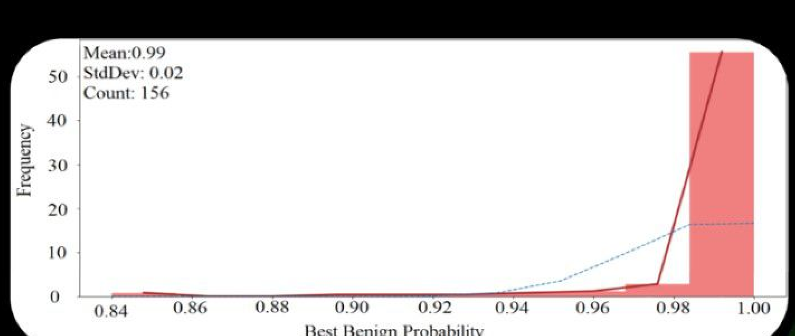
1. The op-code repository of the original-malware feature-vectors and its subsequent obfuscated instances generated by DOOM.
2. Repository of existing trained IDS to act as the adversary(Discriminator 'D' Network as in GAN).
3. A custom RL-environment (to emulate the MDP with which the agents could interact and learn against).
4. The Obfuscating DRL Agent(s).

RESULTS & DISCUSSION

- The mean miss-classification probability of all the files against all the trained agents indicate that the resulting trained agents could obfuscate most of the malware and uplift their metamorphism ($P_{non\text{-}malicious}$) to substantial degree to evade even the best IDS.
  - As shown in figure (top), the mean probability of the non-detectable malware file (PN DMF) has been uplifted by ≥ 0.45 (from almost 0.0). Thus, indicating that the IDS can not effectively detect such obfuscated instances of malware.
- Another interesting observation is related to the op-code similarity between the original malware variants and its generated obfuscated version.
  - Figure (bottom) shows a histogram of similarity of the generated op-code sequence to that of the original malware and indicates that the op-code frequency vector for the obfuscated variant is very similar to the original malware variant.



Histogram of uplift in miss-classification probability for a sample Agent



Histogram of op-code Similarity for a sample Agent

ETHICAL, SAFE AND FUNCTIONALITY PRESERVATION ASPECTS

1. **Ethically-Safe Mechanism:**
   - An advanced AI based malware generation system like DOOM, in wrong hands could have serious implications.
   - Therefore to obviate such negative out-comes we have designed a process that ensures that the obfuscation component of the system would work only at the op-code level and could not be used to create a malicious executable file.
2. **Functionality-Preserving Metamorphism:**
   - DOOM ingests the malware at the instruction-level and also create obfuscations at the instruction-level.
   - The functionality of a given program is represented by the sequence of the instructions available in its assembly produced by the compiler.
   - DOOM only inserts junk instructions and does not remove the existing ones.
   - Doing so, DOOM preserves the intended functionality of the program.

REFERENCES

[1] Jean-Marie Borello and Ludovic Mé. 2008. Code obfuscation techniques for metamorphic viruses. Journal in Computer Virology 4, 3 (2008), 211–220.

[2] Priti Desai and Mark Stamp. 2010. A highly metamorphic virus generator. IJMIS 1 (2010), 402–427.

[3] David Silver et. al. 2014. Deterministic Policy Gradient Algorithms. In ICML'14 Volume 32 (ICML'14). JMLR.org, I–387–I–395.

[4] Timothy P. Lillicrap et. al. 2015. Continuous control with deep reinforcement learning. CoRR abs/1509.02971 (2015). arXiv:1509.02971

[5] Volodymyr Mnih et. al. 2015. Human-level control through deep reinforcement learning. Nature 518, 7540 (2015), 529–533.

[6] Weiwei Hu and Ying Tan. 2017. Generating Adversarial Malware Examples for Black-Box Attacks Based on GAN. CoRR abs/1702.05983 (2017). arXiv:1702.05983

[7] Zilong Lin, Yong Shi, and Zhi Xue. 2018. IDSGAN: Generative Adversarial Networks for Attack Generation against Intrusion Detection. CoRR abs/1809.02077 (2018). arXiv:1809.02077

[8] Antonio Nappa, M. Zubair Rafique, and Juan Caballero. 2015. The MALICIA Dataset: Identification and Analysis of Drive-by Download Operations. Int. J. Inf. Secur. 14, 1 (Feb. 2015), 15–33.

[9] Hemant Rathore, Sanjay K Sahay, Palash Chaturvedi, and Mohit Sewak. 2018. Android malicious application classification using clustering. ISDA. Springer, 659–667.

REFERENCES

[10] Sanjay K Sahay, Ashu Sharma, and Hemant Rathore. 2020. Evolution of Malware and Its Detection Techniques. In Information and Communication Technology for Sustainable Development. Springer, 139–150.

[11] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. CoRR abs/1707.06347 (2017).

[12] Mohit Sewak. 2019. Deep Reinforcement Learning: Frontiers of Artificial Intelligence (1st ed.). Springer Publishing Company, Incorporated.

[13] Mohit Sewak, Sanjay K. Sahay, and Hemant Rathore. 2018. Comparison of Deep Learning and the Classical Machine Learning Algorithm for the Malware Detection. In 19th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing. 293–296.

[14] Mohit Sewak, Sanjay K Sahay, and Hemant Rathore. 2020. An Overview of Deep Learning Architecture of Deep Neural Networks and Autoencoders. Journal of Computational and Theoretical Nanoscience 17, 1 (2020), 182–188.

[15] Muhammad Usama, Muhammad Asim, Siddique Latif, Junaid Qadir, and Ala I. Al-Fuqaha. 2019. Generative Adversarial Networks For Launching and Thwarting Adversarial Attacks on Network Intrusion Detection Systems. IWCMC'19 (2019), 78–83.

[16] M. Usama, M. Asim, S. Latif, J. Qadir, and Ala-Al-Fuqaha. 2019. Generative Adversarial Networks For Launching and Thwarting Adversarial Attacks on Network Intrusion Detection Systems. 78–83.

[17] Hado van Hasselt, Arthur Guez, and David Silver. 2015. Deep Reinforcement Learning with Double Q-learning. CoRR abs/1509.06461 (2015). arXiv:1509.06461

[18] D. Wu, B. Fang, J. Wang, Q. Liu, and X. Cui. 2019. Evading Machine Learning Botnet Detection Models via Deep Reinforcement Learning. In ICC'2019. 1–6.