**INTRODUCTION**

Junk code injection attack is a malware anti-forensic technique against OpCode inspection. As the name suggests, junk code insertion may include addition of benign OpCode sequences, which do not run in a malware or inclusion of instructions (e.g. NOP) that do not actually make any difference in malware activities. Junk code insertion technique is generally designed to obfuscate malicious OpCode sequences and reduce the ‘proportion’ of malicious OpCodes in a malware In our proposed approach, we use an affinity based criteria to mitigate junk OpCode injection anti-forensics technique. Specifically, our feature selection method eliminates less instructive OpCodes to mitigate the effects of injecting junk OpCodes. To demonstrate the effectiveness of our proposed approach against code insertion attack, in an iterative manner, a specified proportion (f5%, 10%, 15%, 20%, 25%, 30%g) of all elements in each sample’s generated graph were selected randomly and their value incremented by one. For example, in the 4th iteration of the evaluations, 20% of the indices in each sample’s graph were chosen to increment their value by one. In addition, in our evaluations the possibility of a repetitive element selection was included to simulate injecting an OpCode more than once. Incrementing Ei;j in the sample’s generated graph is equivalent to injecting OpCodej next to the OpCodei in a sample’s instruction sequence to mislead the detection algorithm. Algorithm 2 describes an iteration of junk code insertion during experiments, and this procedure should repeat for each iteration of k-fold validation. To show the robustness of our proposed approach and benchmark it against existing proposals, two congruent algorithms described in Section 1 are applied on our generated dataset using Adaboost as the classification algorithm.