PinPacker: Automatic Unpacking of Evasive Malware

The arms-race b/w malware authors and AV developers.

➤ Packing to effectively neutralize static analysis.

➤ Running code in a sandbox and waiting for the packer to decrypt/inflate the actual program.

➤ Evasive behaviors to effectively counter any dynamic-analysis-based approach[1].

RQ: With the advantage swinging back to the attacker’s side, what can we do to counter both packing and evasive behaviors?

PinPacker: Implementation

Hijacking the control flow: The program is instrumented through a pintool written in Intel Pin[3]. The tool can communicate each executed instruction to the decision agent running on a separate process/computer. Conditional jumps are forced by overwriting the zero, sign, or carry flag accordingly.

Verifying if new code is unpacked: The agent dynamically reconstructs the control-flow graph (CFG) and compares it against the CFG obtained by merging the ones obtained at each previous execution.

The decision agent (ongoing work): The optimization problem can be naturally modeled as a reinforcement-learning process. A decision agent may be pre-trained to recognize the subgraphs belonging to known evasive behaviors.

References: