POLITECNICO MILANO 1863

POLITECNICO MILANO 1863
NECST laboratory

Mario D'Onghia*, Daniele Mammone*,
Mario Polino, Michele Carminati, Stefano Zanero
(*equal contribution)

contact: mario.donghia@polimi.it

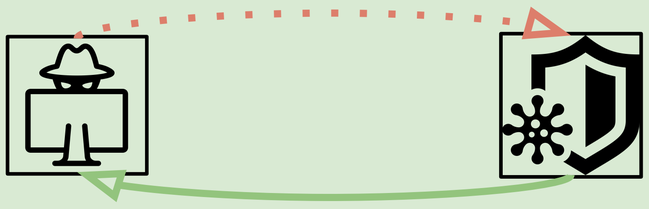
ACSAC 2023, Austin, Texas

PinPacker: Automatic Unpacking of Evasive Malware

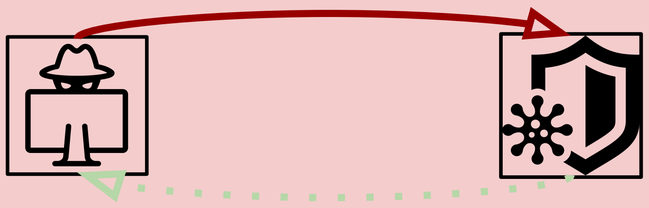
The arms-race b/w malware authors and AV developers.

➢ Packing to effectively neutralize static analysis.

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

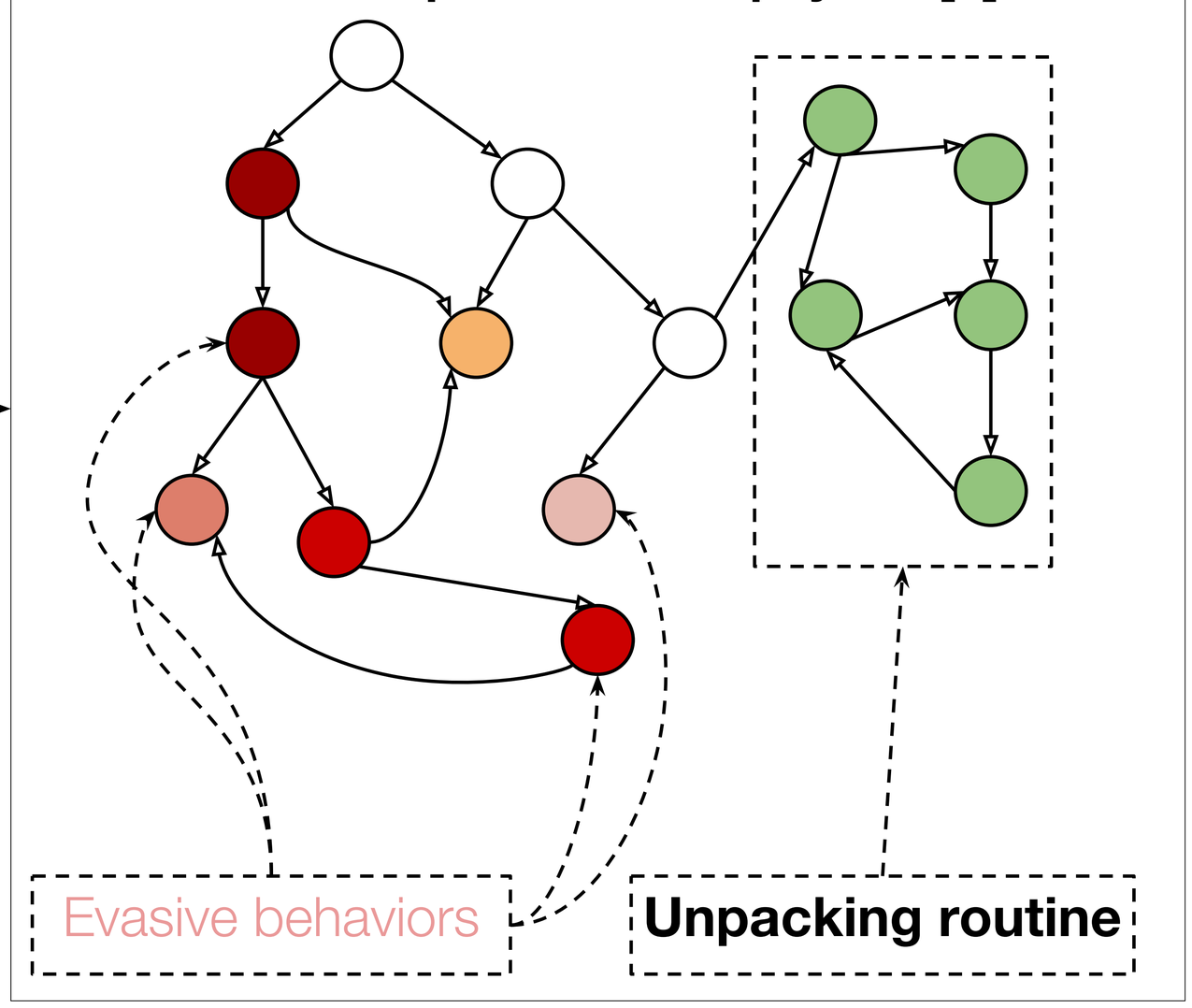


➢ Evasive behaviors to effectively counter any dynamic-analysis-based approach[1].



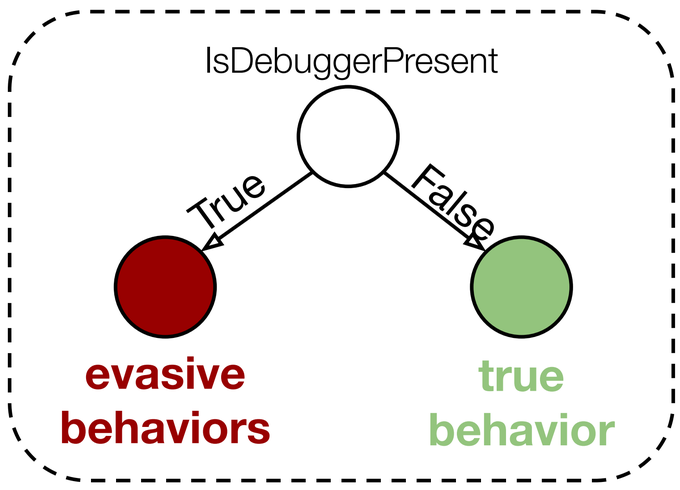
RQ: With the advantage swinging back to the attacker's side, what can we do to counter <u>both packing and evasive behaviors</u>?

Intuition 💡: influencing the program execution can prevent it from "hiding" its true behavior and force it to unpack the real payload[2].



PinPacker

Influencing the execution flow by forcing the program to take/skip certain conditional jumps.



Decisions are taken based on previous runs and on the decisions taken so far in the current execution.

Exec 1: Take;Skip;...;Take
Exec 2: Take;Take;...;Skip
...
...
Exec n-1: Skip;Take;...;Skip

A decision agent is built to maximize the amount of *yet unseen* unpacked code.

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:
1. Galloro, Nicola, et al. "A Systematical and longitudinal study of evasive behaviors in windows malware." *Computers & Security* 113
2. Peng, Fei, et al. "{X-Force}:{Force-Executing} Binary Programs for Security Applications." *23rd USENIX Security Symposium* 2014.
3. Intel Pin - https://www.intel.com/content/www/us/en/developer/articles/tool/pin-a-dynamic-binary-instrumentation-tool.html