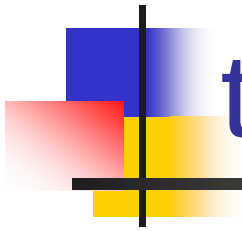
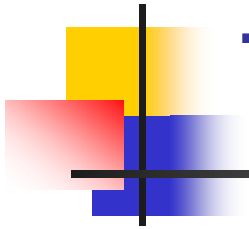


Static Analysis of Executables to Detect Malicious Patterns

Mihai Christodorescu and Somesh Jha
(University of Wisconsin, Madison)

12th USENIX Security Symposium

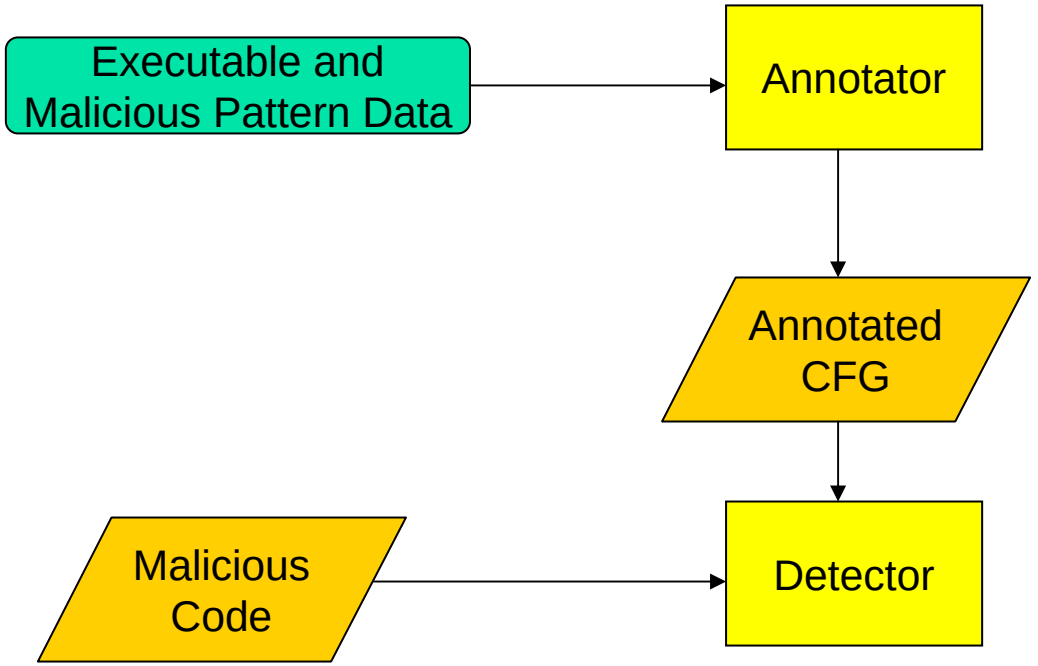
Presenter: Brett Lomas

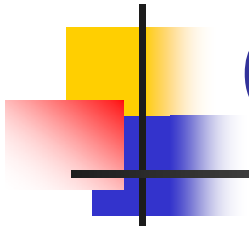


The Problem

- Given
  - a malicious sequence of instructions
- Find
  - a sequence of instructions in some obfuscated code which is semantically equivalent.

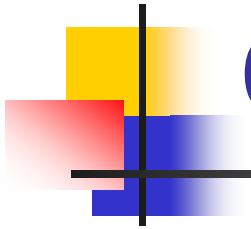
Architecture





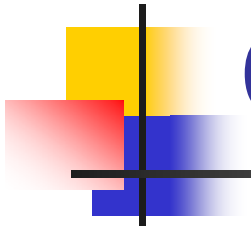
Critical Comments

- Prototype speed
  - Scanning a 1MB benign program took approximately 16 minutes
    - Annotator took over 13 minutes
    - Detector took over 2 minutes
  - People get annoyed at the speed of current virus scanners!
  - Paper authors highlighted some execution times as “unacceptably large”



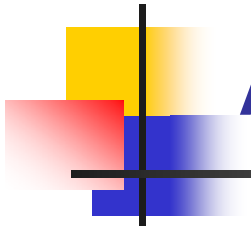
Critical Comments cont...

- They obfuscated examples of malicious code
  - Then used this for evaluating to effectiveness of their prototype
  - Reported FP and FN rate zero
  - No external sample of malicious code and its obfuscations
  - How useful are the results in the real world?



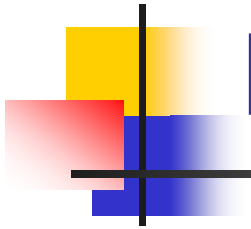
Critical Comments cont…

- Only examined ‘common obfuscation techniques’
  - E.g. Dead Code Insertion, Code Transposition, Register Reassignment and Instruction Substitution
- No external corroboration of ‘common’
  - One cannot assess effectiveness of this method objectively without.
- What about Opaque constructs to obfuscate control flow (Collberg et al.) for example?
  - Paper dismisses as not ‘common’



Appreciative Comments

- Highlighted the inability of commercial scanners to handle simple obfuscations
  - All sample scanners failed
    - Nop insertion
    - Code transposition
  - What is the risk?



Discussion

- If the speed was acceptable, is this a viable idea for the defense against viruses/Trojan horses etc?