Behavioral Detection of Malwares: From a Survey Towards an Established Taxonomy

JACOB Grégoire[1/2], **DEBAR Hervé**[1], **FILIOL Eric**[2]

[1] *France Télécom R&D, Network and Service Security (MAPS/NSS).*

[2] *French Army Signal Academy, Cryptology & Virology Lab (ESAT).*

2nd International Workshop on The Theory of Computer Viruses
NANCY – May 2007

research & development

cryptology & virology lab.

summary

1

Fundamental Aspects of Behavioral Detection

1.1 Two Strategies for Detection

Appearance detection (form-based)

- Relies on syntactic markers
- Undecidable
- Problem of the signature extraction:
  cost and analysis delay / release speed and mutation mechanisms

Behavioral detection (function-based)

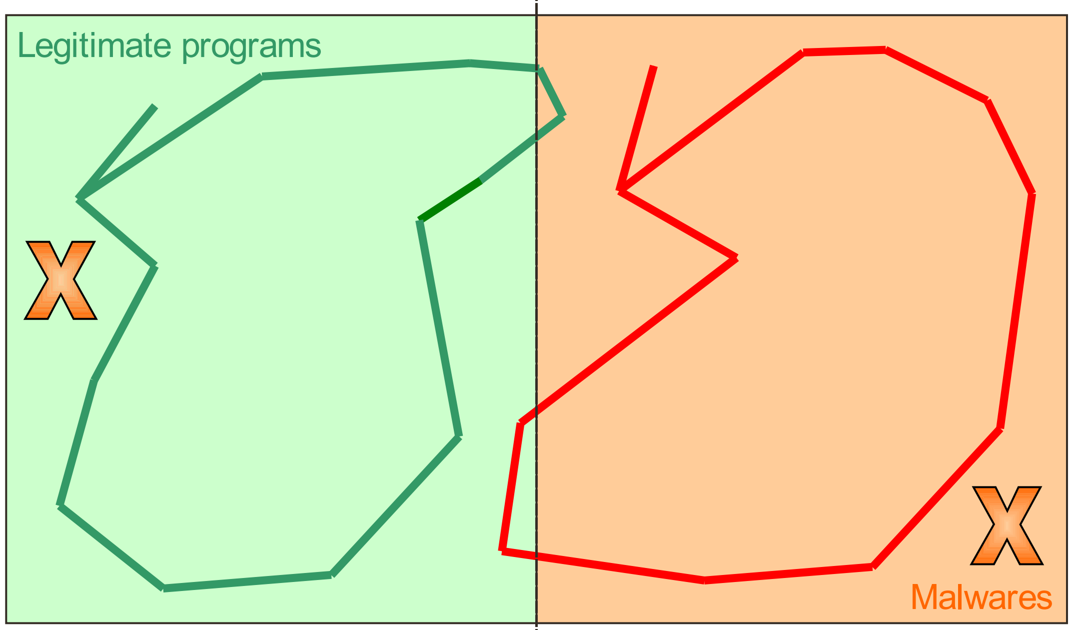
- Relies on the use of the system services and resources
- Undecidable since equivalent to syntactic analysis on request arguments
- Generic strategy more resilient to modifications, manual or automatic
- Recent interest because of the resources required once prohibiting

1.2 Two Opposite Approaches

Intrusion Detection (Vulnerabilities)

Virology

False positive = Impossibility to create generic program profiles



Legitimate model

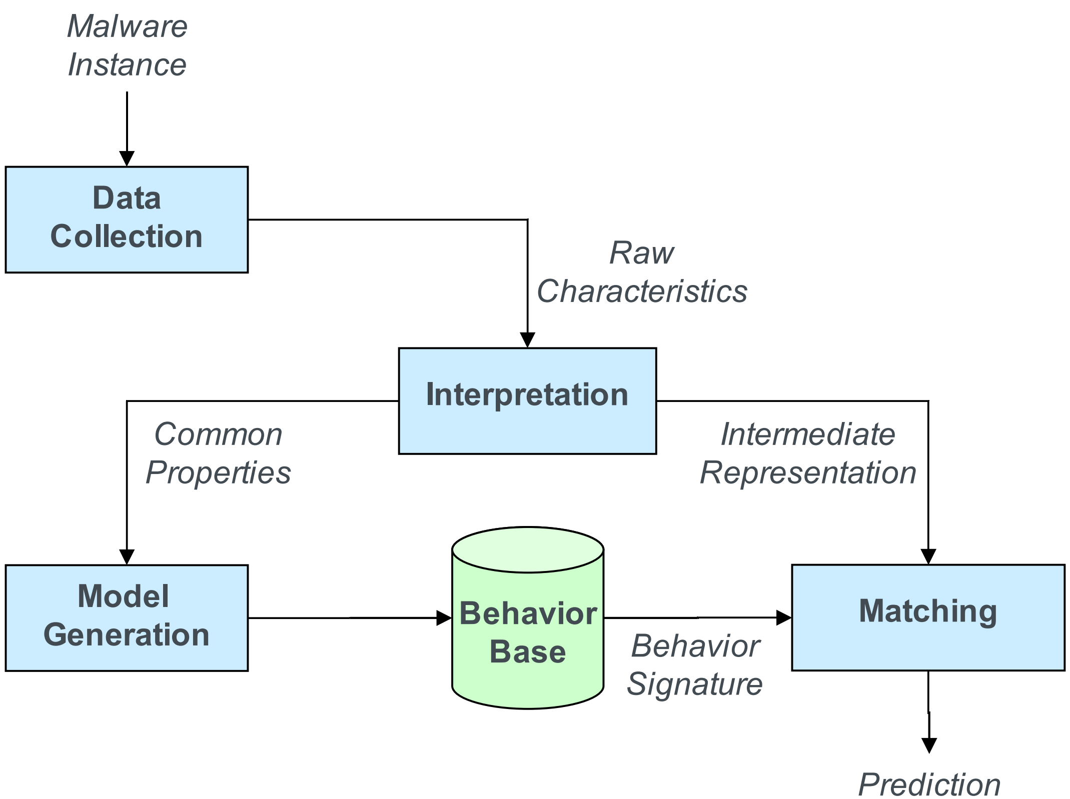
Behavior signatures

False negative = Missing signature

2

Generic Description of Behavioral Detectors

2.1 Sequential Steps of the Detection



2.2 Important Properties

Properties for assessment (AV Testing)

- Completeness / Accuracy / Adaptability:

False positives and negatives, adding new behaviors

- Performance:

Complexity, overload introduced

- Resilience to anti-analysis techniques:

Obfuscation, stealth

- Unobtrusiveness / Fault-Tolerance (dynamic):

No perturbation introduced in the malware execution by the analysis and resistance to its proactive defenses

- Timeliness (dynamic):

Detection reached before the point of no return

3

Taxonomy of the Behavioral Detectors

3.1 Data Capture Conditions

Dynamic monitoring

- Sequences of discrete events (traces): *interruptions, system calls*
- Conditions: - real time with or without action recording
  little overload but risky as effectively executed
  - sandboxes and virtual machines
  important overload, risk of detection or escape

Static Extraction

- Program structure: *control and data flow graphs*
- Conditions: - disassembly and debugging
  hindered by obfuscation, anti debug, packing

3.2 Matching Algorithms and Models

Simulation-Based Detection

- Linked to dynamic monitoring for the simulation environment
- Black box approach
- Explore the current path during execution
- Matching of sequential models :
  expert systems, heuristic engines, state machines

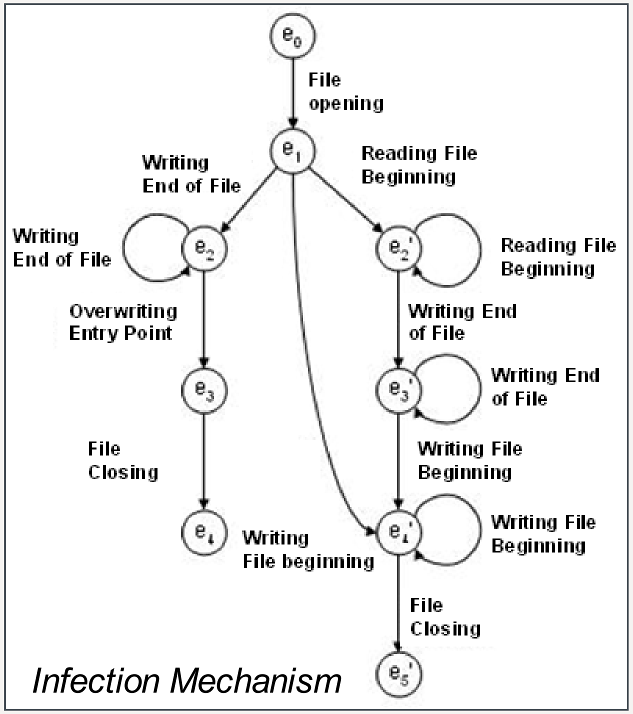
Formal verification

- Linked to static extraction for the program abstraction
- White box approach
- Explore every possible path of execution
- Bisimulation between abstractions and specifications:
  annoted graph isomorphism, model checking

3.2 Matching Algorithms and Models

Examples from both methods

- Deterministic finite automaton

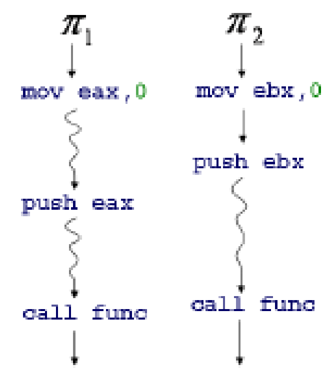


Infection Mechanism

- Model checking

$$C_1 : \exists r EF(mov(r,0) \wedge EF(push(r) \wedge EF(call(func))))$$

$$C_2 : \exists r EF(mov(r,0) \wedge AX(push(r) \wedge EF(call(func))))$$



E an exisiting path *A* any path	Quantifier operators
F an undefined future step *X* the immediate following step	Temporal operators

3.3 Behavioral signature generation

Manual generation

- Defined by a specialist based on its experience

  Refined and deployable on any system
- Defined by the user policy

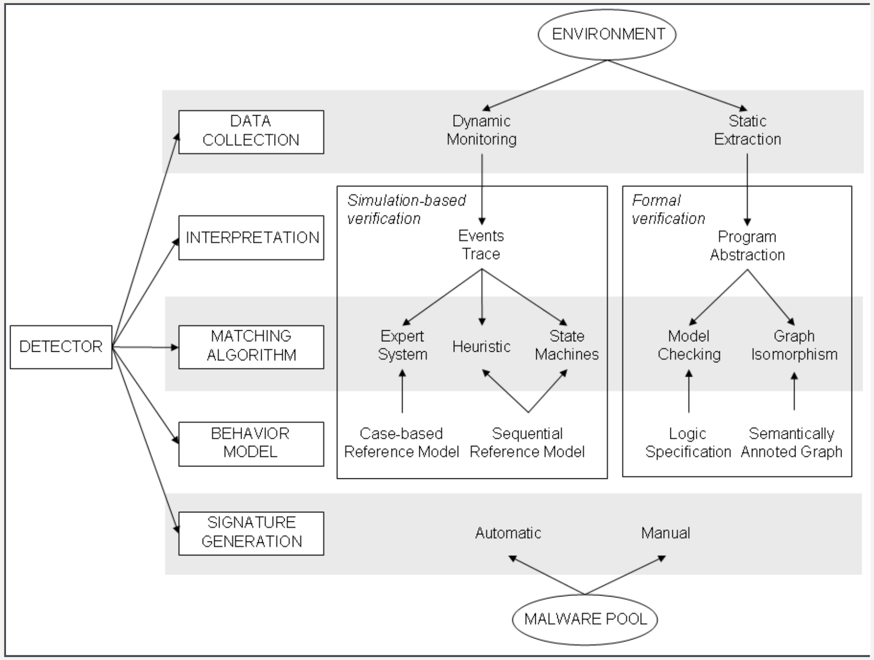
  Better adequacy but reserved to knowledgeable users
- Time consuming and hardly evolving

Automated generation

- Data mining and classifier
- Mainly three paradigms:
  Rules induction / Bayesian Statistics / Clustering
- Large, noise-free, learning and testing pools required

3.4 Synthesis on the Classification

Classification scheme



Major trends

- Expert system and heuristics with sandboxing in commercial products
- Classifiers with virtual machines / formal verification in research

4

Conclusions and Perspectives

4. Conclusion and Perspectives

Heterogeneity of the systems

- Model multiplication inducing vocabulary inconsistency

  Explain the necessity of a taxonomy

  Need of a high level reference model for behaviors

Two main axes in the classification

- Choice between formal verification or simulation

  Condition the capture, the model and associated matching algorithm

- Complementary strength and weaknesses

  Possible combination already explored

Thank you for your attention,

Any questions?

A Few References

- [01] F. Cohen, "*Computer Viruses*", PhD thesis, University of South California,1986.
- [02] E. Filiol, "*Computer viruses: from theory to applications*", Springer, IRIS Collection, 2005, ISBN:2-287-23939-1.
- [03] M. Debbabi, "*Dynamic monitoring of malicious activity in software systems*", in Proceedings of the Symposium on Requirements Engineering for Information Security (SREIS), 2001.
- [04] F. Veldman, "*Heuristic anti-virus technology*", in Proceedings of the International Virus Protection and Information Security Council, 1994.
- [05] B. L. Charlier, A. Mounji, and M. Swimmer, "*Dynamic detection and classification of computer viruses using general behaviour patterns*", in Proceedings of the Virus Bulletin Conference, 1995.

A Few References

- [06] M. Christodorescu, S. Jha, S. A. Seshia, D. Song, and R. E. Bryant, "*Semantic-aware malware detection*", in Proceedings of IEEE Symposium on Security and Privacy, pp. 32–46, 2005.
- [07] J. Kinder, S. Katzenbeisser, C. Schallhart, and H. Veith, "*Detecting malicious code by model checking*", Lecture Notes in Computer Science, vol. 3548, pp. 174–187, 2005.
- [08] M. G. Schultz, E. Eskin, and E. Zadok, "*Data mining methods for detection of new malicious executables*", in Proceedings of IEEE Symposium on Security and Privacy, pp. 38–49, 2001.
- [09] E. Kirda, C. Kruegel, G. Banks, G. Vigna, and R. Kemmerer, "*Behavior-based spyware detection*", in Proceedings of the 15th USENIX Security Symposium, 2006.