Malwares as Interactive Machines: A New Framework for Behavior Modelling

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

[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 A Theoretical Approach to Introduce Interactions

- Models in abstract virology
- Why interactions are so important
- Towards an interaction formalism
- Impacts of interactions on detection

2 A Semantic Expression of Malicious Behaviors

- An object-oriented semantic
- Example of malicious behavior

3 Conclusions and perspectives

1

A Theoretical Approach to Introduce Interactions

1.1 Models in Abstract Virology

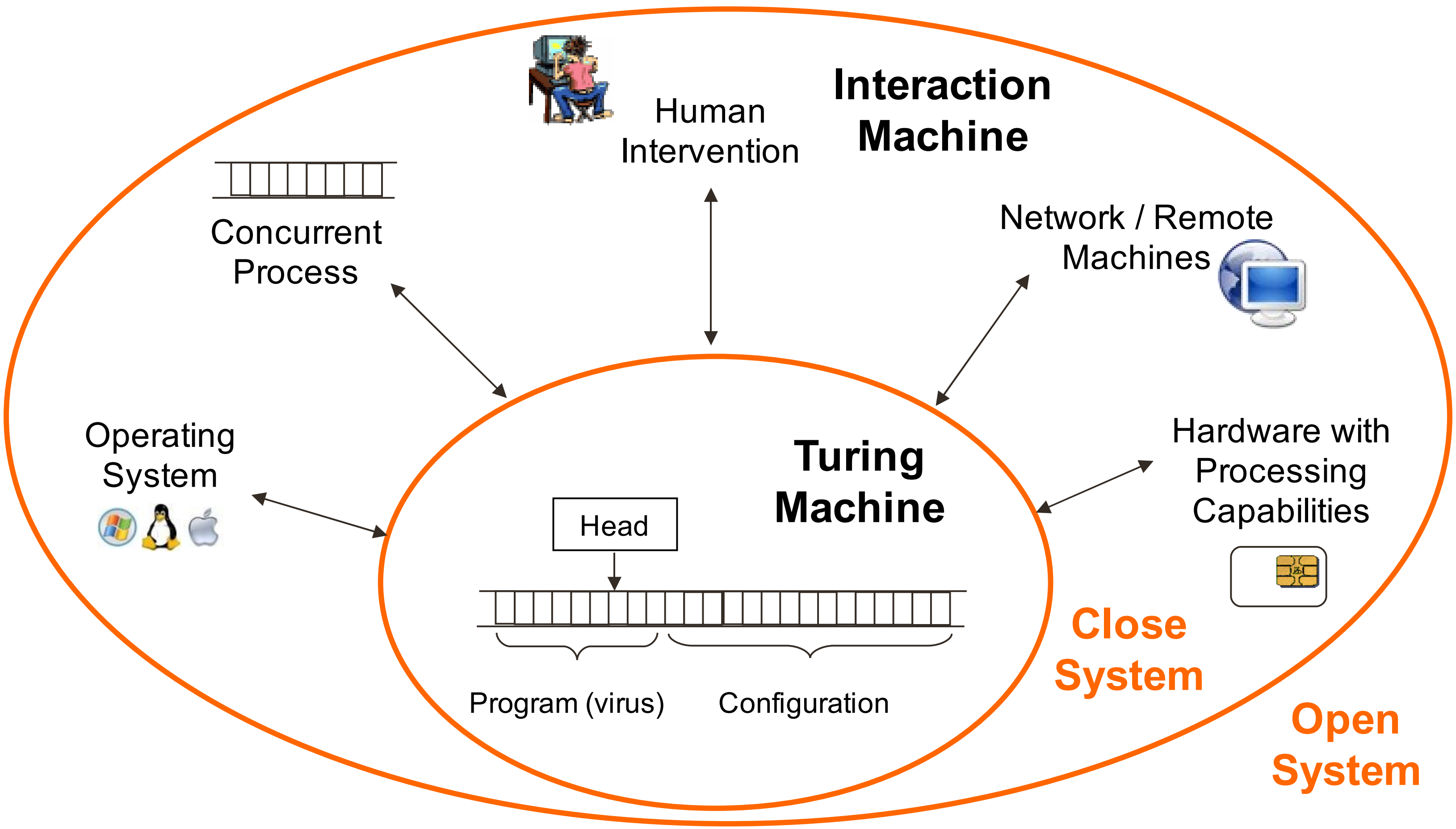
Mainly Turing equivalent formalisms

- Turing Machine (F. Cohen)
- Recursive functions (L. Adleman, Zuo & Zhou)
- Recursion theory (G. Bonfante, M. Kaczmarek & J-Y. Marion)
- Still very few publications in the domain

Major results brought

- Undecidability of the detection
- Formalization of the important viral techniques: *replication, mutation, stealth, residency*
- Yet some known limitations…

1.2 Why Interactions Are so Important



1.2 Why Interactions Are so Important

Shortcomings of Turing equivalent formalism

- Interactions
- Concurrency
- Non-termination

Techniques though used in current malwares

- Interactions with the OS and network: *-replication, propagation*
- Distributivity over several modules: *-k-ary virus (E. Filiol)*
- Residency in rootkits: *-proactive defense and stealth*

1.3 Towards an Interaction Formalism

Extension of abstract machines

- First attempt to capture interactions in virus (F. Leitold) : *Random Access Stored Program with Attached Background Storage*
- Our contribution: *Interaction Machines (P.Wegner)*
- Expressive power equivalent to Turing Machines with Oracles

Modelling interactions through oracles

- The interaction is function of the history and temporality:

$$I(data\ transmitted, time, interaction\ history) = data\ received$$

- An oracle associated to an adversary embeds the previous notions:

$$\Theta(data\ transmitted) = data\ received$$

- Size of data unspecified: null in case of unilateral interaction, infinite
- Adaptable to previous models

1.3 Towards an Interaction Formalism

Definition of an interactive virus

- Based on the definition introduced by Bonfante et al.
- Execute different functions according to the adversaries

$$\varphi_v(p,x) = \begin{cases} V_{1,1}(v,p,x,\Theta^1(v)) & \text{if} \quad (p,x,\Theta^1(v)) \in C_1 \\ \dots & \\ V_{n,k}(v,p,x,\Theta^n(v)) & \text{if} \quad (p,x,\Theta^n(v)) \in C_k. \end{cases}$$

Definition of a distributed virus

- The viral function *f* requires the collaboration of two components
- Can be generalized over *n* components using interaction graphs

$$\varphi_{v,w}(p,x,y) = f(\varphi_v(p,x,\Theta^w(v)), \varphi_w(p,y,\Theta^v(w))).$$

1.4 Impact of Interactions on Detection

Type of interaction	Complexity	Examples
Interaction with inert objects	Linear	Files, Registry entries
Interaction with active objects through interfaces	NP-Complete	Network, Synchronization objects
Unconstrained interaction with adversaries	Undecidable	Rewriting process memory

Detection complexity

- The detection of interactive and distributed viruses is no longer $_2$but 2 3 , or *undecidable* according to the class of interaction

2

A Semantic Expression of Malicious Behaviors

2.1 An Object-Oriented Semantic

Assets of a semantic approach

- Semantic can convey the final purpose of a malicious behavior
- Language theory offers a solid theoretical background
- Easy application in operational contexts

Richer language facilities required

- Turing-Complete languages insufficient
- Polarity of the grammar units: listen and transmit operators
- Non-deterministic operator

2.1 An Object-Oriented Semantic

Application of the object perception

- Internal mechanisms and attributes

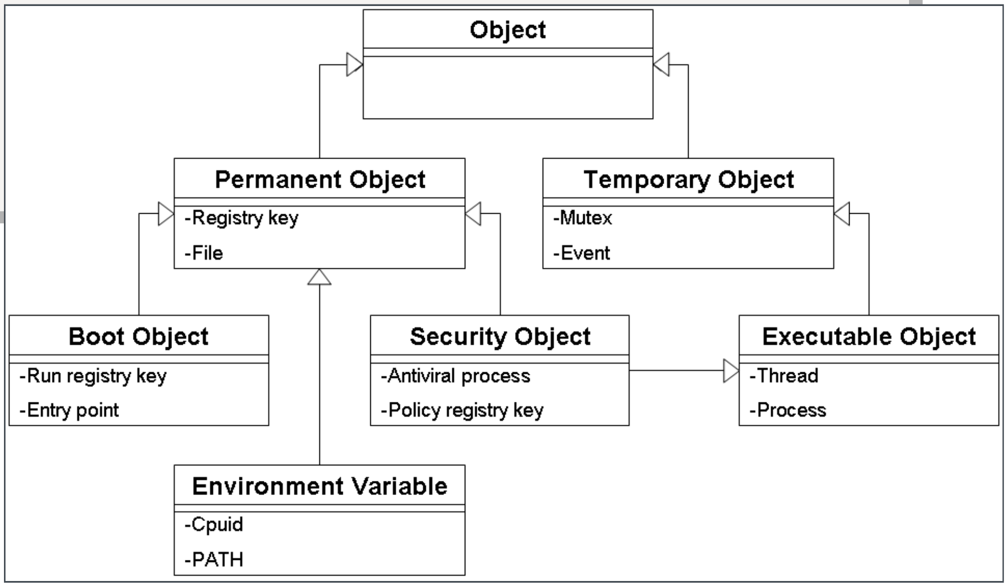
  Turing-Complete grammar defining basic computing operations

- External interfaces

  Addition of control and communication operations

- Adversary typing (inheritance)

  According to their use
  in the malware's lifecycle



2.2 Example of Malicious Behavior

Behavior description: propagation

- Overview of the generative behavior grammar put forward
- Preliminary survey over several malware strains
- Each grammar unit can be associated to different instantiations: *Connexions between instruction meta-structures and grammar units*
- Illustration adapted from the MyDoom source code

(i) < Propagation > ::= < Opening >< Reading >< Mutation >< Transmitting >
| < Reading >< Opening >< Mutation >< Transmitting >

2.2 Example of Malicious Behavior

$O_{channel}$ in { obj_com }

(ii) < Opening > ::= open $O_{channel}$;

```
/* Open socket */
struct hostent *h = gethostbyname(hostname);
struct sockaddr_in addr = *(h->h_addr_list[0]);
sock = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP);
connect(sock, addr, sizeof(struct sockaddr_in));
```

2.2 Example of Malicious Behavior

V_{code} in { var }

(iii) < Reading> ::= open this ;
receive V_{code} this ;

```
/* Open currently executing file */
GetModuleFileName(NULL, selfpath, MAX_PATH);
HANDLE hFile = CreateFile(selfpath, GENERIC_READ,
  FILE_SHARE_READ|FILE_SHARE_WRITE, NULL, OPEN_EXISTING,
  FILE_ATTRIBUTE_NORMAL, NULL);
/* Reading file content in buffer */
DWORD dwSize = GetFileSize(hFile, &dwUp);
ReadFile(hFile, pBufferCode, dwSize, &dwRead, NULL);
```

2.2 Example of Malicious Behavior

$| \; V_{formatted}$ in { var }

$$\text{(iv)} < \text{Transmitting} > ::= \text{send } V_{code} \;\; O_{channel}\,;$$
$$| \;\; < \text{Formatting} >$$
$$\text{send } V_{formatted} \;\; O_{channel}\,;$$

```
/* Optional formatting */
…
/* Sending information */
send(sock, pBufferCode, lstrlen(pBufferCode), 0);
```

2.2 Example of Malicious Behavior

$V_{position}$ in { var }
C_{header}, C_{hsize} in { const }

(v) < Formatting > ::= $V_{position}$:= (&($V_{formatted}$)) ;
[$V_{position}$] := (C_{header}) ;
$V_{position}$:= (+($V_{position}$,C_{hsize})) ;
< Encoding >
[$V_{position}$] := (V_{code}) ;

```
/* Concatenate header */
char header[] = "From: myadresse@domaine.ext\r\nTo: target
  adresse@domaine.ext\r\nSubject mail subject\r\nDate\r\nM
  IME-Version\r\nContent-Type: multipart/mixed\r\n";
lstrcat(pFormatted, header);
/* Optional encoding */
…
/* Concatenate code */
lstrcat(pFormatted, pBufferCode);
```

2.2 Example of Malicious Behavior

| $C_{parameter}$ in { const }

(vi) $<$ Encoding $>$ ::= V_{code} := ($<$ Op2 $>$ (V_{code}, $C_{parameter}$)) ;
$<$ Encoding $>$
| ϵ

```
/* Base 64 table */
BYTE alpha[] =
  "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz01234
  56789+/";
/* Base 64 encoding */
 q[0] = alpha[t[0] >> 2];
 q[1] = alpha[((t[0] & 03) << 4) | (t[1] >> 4)];
 q[2] = alpha[((t[1] & 017) << 2) | (t[2] >> 6)];
 q[3] = alpha[t[2] & 077];
```

3

Conclusions and Perspectives

3. Conclusion and perspectives

Interaction formalization in a viral context

- A first approach to introduce interactions in virus models
- A preliminary evaluation of the impact on the detection complexity

Explore the dedicated formalisms for new definitions: π-calculus?

Interpretation of malicious interactions

- Semantic description at a high level of abstraction
- Brings into light equivalent functionalities
- Several behaviors described: *duplication, infection, propagation, polymorphism, metamorphism, stealth, overinfection and activity tests*

A more complete survey for a richer description

Integrate this semantic in existing detectors

Build classification mechanisms for adversaries

Thank you for your attention,

Any questions?

A Few References

- [01] G. Bonfante, M. Kaczmarek, and J.-Y. Marion, "*On abstract computer virology from a recursion theoretic perspective*", Journal in Computer Virology, vol. 1, no. 3-4, pp. 45–54, 2006.
- [02] P. Wegner, "*Why interaction is more powerful than algorithms*", Communications of the ACM, vol. 40, no. 5, pp. 80–91, 1997.
- [03] E. Filiol, "*Formalisation and implementation aspects of k-ary (malicious) codes*", Journal in Computer Virology, vol. 3, no. 3, EICAR 2007 Special Issue, V. Broucek Ed., 2007.
- [04] F. Leitold, "*Mathematical model of computer viruses*", in Best Paper Proceedings of EICAR, pp. 194–217, 2000.
- [05] G. Perrier, "*Interaction grammars*", in Proceedings of the 18th conference on Computational linguistics - Volume 2, pp. 600–606, 2000.