Classification of Computer Viruses Using the Theory of Affordances

Matt Webster and Grant Malcolm*

Abstract

We present a new ontology for the classification of computer viruses and other forms of reproducing malware based on Gibson's Theory of Affordances. We show how an existing method for reproducer classification can be specialised for malware classification, and give a worked example of how one might classify a Unix shell script virus in three different ways, depending on the particular reproductive model being used. Finally we suggest possible applications of our classification to the area of computer virus detection.

1 Introduction

There are many interesting and useful examples of computer virus classifications [1, 2, 4, 5, 6, 8], but the area will remain open as long as we can gain new insights from new classifications. This paper describes a new approach to the classification of reproducing malware based on Gibson's Theory of Affordances [3]. This approach arose from work on the related problem of reproducer classification, in which reproducers could be classified according to whether or not their self-description and/or reproductive mechanism — two essentials for reproduction [7] — are afforded to the reproducer by an external agent or by the reproducer itself [9]. For reproducing malware, "self-description" refers to the means by which a computer virus or worm can obtain its own code for infection, and "reproductive mechanism" refers to the means by which the self-description, in the form of viral code, is used to create an offspring, e.g., in the case of a parasitic virus, the act of copying viral code into another file in the file system.

*Department of Computer Science, University of Liverpool, Liverpool, L69 3BX, UK. Email: {matt,grant}@csc.liv.ac.uk.

2 Affordance-based Classification

Affordances were introduced by Gibson as a theory of visual perception [3]; an affordance is an action possible for an animal when interacting with some object in its environment. For a small mammal, for example, a cave affords shelter and a tree affords the ability to climb for a better view of the surroundings. We take the notion of affordances and use it metaphorically, e.g., an operating system (OS) *affords* disk input/output for a computer virus.

Before we attempt to classify computer viruses, we need a formal model of reproduction and affordances.

Definition 1 *A* model of a reproductive system *consists of:*

- *a labelled transition system* $(S, A, \longmapsto)$*, where* S *is a set of states,* A *is a set of actions (labels), and* $\longmapsto$ *is a ternary relation for labelled transitions between states, s.t. if* $s \overset{a}{\longmapsto} s'$*, the action* a *occurring in the state* s *leads to the new state* s'*;*
- *a set Ent of* entities *and a relation* $\_ : \_$ *between entities and states, where for* $e \in Ent$ *and* $s \in S$*,* $e : s$ *indicates that* e *is present in the state* s*;*
- *a function Aff that assigns to two entities* e *and* e'*, a set* $Aff(e, e')$ *of possible actions, in such a way that if* $a \in Aff(e, e')$*, then for all states* s *with* $e' : s$*,* a *is possible in* s *(i.e.,* $s \overset{a}{\longmapsto} s'$ *for some state* s'*) if and only if* $e : s$*. Notionally,* $Aff(e, e')$ *is the set of affordances that* e *gives to* e'*.*

In the context of computer virus reproduction, we can say that the environment consists of the computer system in which the virus reproduces, e.g., a computer file system for a parasitic virus. The set *Ent* of entities contains the virus itself, as well as any agents in the virus's environment which might assist the virus in reproduction, e.g., OS application programmer interface (API) functions that might allow a virus to write to a file, or a network process (such as FTP) that allows data transit over the network. The set S of states corresponds naturally to the states of the machine executing the virus, but it is advantageous to allow some abstraction in the state transition relation ($\longmapsto$) between states, so that we can say that some state s' follows some other state s without explicitly modelling all intermediate states at the level of the processor, for example. Since we have made the state transition relation abstract, it follows that the actions between these states are abstract, so that we allow transitions like $s \overset{\texttt{copy}}{\longmapsto} s'$, where `copy` is an action that copies some file. Therefore it is not necessary to model the reproductive

```
st='echo st=$sq${st}$sq > .1;echo dq=$sq${dq}$sq >> .1; echo
    sq=$dq${sq}$dq >> .1;echo $st >> .1; chmod +x .1'
dq='"'
sq="'"
echo st=$sq${st}$sq > .1;
echo dq=$sq${dq}$sq >> .1;
echo sq=$dq${sq}$dq >> .1;
echo $st >> .1;
chmod +x .1
```

Figure 1: Unix shell script virus.

processes of computer viruses at the lowest level, but rather at some convenient level of abstraction that lets us observe the defining features of that virus's reproductive process.

2.1 Reproductive Types

Let *sd* and *rm* be the abstract actions corresponding to the acquisition of a self-description, and the reproductive mechanism, respectively, for some virus v. Then we can define the four reproductive types as follows:

- **Type I** computer viruses are those who afford themselves a self-description and reproductive mechanism, i.e., $sd, rm \in \mathit{Aff}(v, v)$.
- **Type II** reproducers are those who afford themselves a self-description, but their reproductive mechanism is afforded by some external agent, i.e., $sd \in \mathit{Aff}(v, v), rm \notin \mathit{Aff}(v, v)$.
- **Type III** reproducers are those who afford themselves a reproductive mechanism, but their self-description is afforded by some external agent, i.e., $sd \notin \mathit{Aff}(v, v), rm \in \mathit{Aff}(v, v)$.
- **Type IV** reproducers are those whose self-description and reproductive mechanism are afforded by some external agent, i.e., $sd, rm \notin \mathit{Aff}(v, v)$.

2.2 Example: Unix Shell Script Virus

We shall now demonstrate classification of a Unix shell script virus (see Fig. 1), adapted from a similar virus[1] by Bruce Ediger.

The first three lines of the virus define three variables that contain the program code and aliases for single and double quotation marks. The first three statements of the program code output these data into a new file called `.1`. The fourth statement of the program appends the program code to `.1`,

[1] Available from `http://www.nyx.net/~gthompso/self_sh.txt`.

and the final statement of the program changes the file permissions of `.1` so that it is executable. At this point the reproductive process is complete.

We sketch[2] a model of a reproductive system M as follows. Let the set of entities $Ent = \{v_B, \texttt{bash}\}$, which contains the virus v_B and the `bash` shell program. Let the set of actions $A = \{osd, out, sp\}$ where *osd* is the means by which the virus obtains its self-description for the purposes of output to a file (the variable assignments at the start of the file); *out* is the output process itself (the `echo` statements), and *sp* is the "set permissions" action (the `chmod` statement). Then, the states of the reproductive process $S = \{s_1, s_2, s_3, s_4\}$ and we define the relation $\longmapsto$ as follows: $s_1 \overset{osd}{\longmapsto} s_2 \overset{out}{\longmapsto} s_3 \overset{sp}{\longmapsto} s_4$. We assume that for all $s \in S$ and $e \in Ent$, $e : s$.

Classification of the shell script virus is as follows. The abstract action *sd* corresponds to the action *osd*, which is an abstraction of the data (variables) stored within the virus. These data provide a self-description, and therefore $sd \in \mathit{Aff}(v_B, v_B)$. This means that the virus is either Type I or II. The abstract action *rm* corresponds to actions *out* and *sp*. Since *out* and *sp* use `echo` and `chmod` respectively, which are commands provided by `bash`, we know that $rm \notin \mathit{Aff}(v_B, v_B)$ and therefore v_B is a Type II reproducer.

If we do not consider `bash` to be external to v_B, then all of the actions that it affords the virus are effectively afforded by the virus to itself. We therefore form a new reproductive model M' which includes a new entity, v'_B, as the conglomeration of v_B and `bash`. We can see that $sd, rm \subseteq \mathit{Aff}(v'_B, v'_B)$ and therefore in this new model v'_B is a Type I reproducer.

If we consider that all actions of the reproductive process are afforded by `bash`, perhaps because the virus could not execute without its presence, then we form another model M'' in which $osd, out, sp \in \mathit{Aff}(\texttt{bash}, v_B)$, so $sd, rm \notin \mathit{Aff}(v_B, v_B)$ and therefore in M'' the virus is classified as Type IV.

3 Conclusion

We have shown that computer viruses, as a subclass of the class of reproducers, can be classified using our ontology. Additionally, we have shown that the classification of a particular virus depends on the granularity of the reproductive model used. Therefore there are no implicit assumptions about the attribution of actions to external agents, but once a schema has been decided on, the classification gives insight into the reliance of a virus on those agents. We demonstrated this by classifying a Unix shell script virus using three different reproductive models.

[2]We will define these models fully in an extended version of this paper.

We anticipate that our ontology will be beneficial in the classification of computer viruses according to their degree of reliance on external agents. For example, some viruses use a compiler in their reproductive processes. The compiler can be recognised as an external agent, and thus by disabling the agent (e.g., by deleting the compiler) we can also disable the virus. However, this would not be possible if the virus was not reliant on the compiler. Therefore we might consider a virus that does not rely on any external agents to be more dangerous. Within our classification, viruses like this can be classified as Type I, and thus our ontology could allow rational prioritisation for antivirus scanners on systems where resources are limited.

References

[1] L. M. Adleman. An abstract theory of computer viruses. In *Advances in Cryptology — CRYPTO '88*, volume 403 of *Lecture Notes in Computer Science*, pages 354–374, 1990.

[2] E. Filiol. *Computer Viruses: from Theory to Applications*. Springer, 2005. ISBN 2287239391.

[3] J. J. Gibson. *The Ecological Approach to Visual Perception*. Houghton Mifflin, Boston, 1979. ISBN 0395270499.

[4] L. A. Goldberg, P. W. Goldberg, C. A. Phillips, and G. B. Sorkin. Constructing computer virus phylogenies. *Journal of Algorithms*, 26(1):188–208, 1998.

[5] E. H. Spafford. Computer viruses as artificial life. *Journal of Artificial Life*, 1(3):249–265, 1994.

[6] P. Ször. *The Art of Computer Virus Research and Defense*. Addison-Wesley, 2005. ISBN 0321304543.

[7] J. von Neumann. *Theory of Self-Reproducing Automata*. University of Illinois Press, 1966.

[8] N. Weaver, V. Paxson, S. Staniford, and R. Cunningham. A taxonomy of computer worms. In *WORM '03: Proceedings of the 2003 ACM Workshop on Rapid Malcode*, pages 11–18. ACM Press, 2003.

[9] M. Webster and G. Malcolm. Reproducer classification using the theory of affordances. In *Proceedings of the First IEEE Symposium on Artificial Life (IEEE-ALife'07)*. To appear.