Untangling Reverse Engineering with Logic and Abstraction

Andy King

Portcullis Computer Security Limited, Pinner, HA5 2EX, UK

Reverse engineering is the filthy end of the security industry; it is the business of extracting information from a program when the source is unavailable. Reversing is often necessary when performing a security audit on a product that relies on third-party software such as a library. Security engineers also reverse to reason about the latest malicious programs and devise antivirus software. Security engineers (and malicious hackers) do not attempt to reverse assembler into, say C, which is the traditional aspiration in reversing, but merely to understand the code to sufficient depth to locate a vulnerability.

The most popular tool that is used for reversing is the IDA Pro dissembler [4]. This dissembler divides an executable into (more or less) its basic blocks, presenting them visually to the engineer in a flow diagram. Needless to say, the major impediment to reversing is the enormous effort required to understand an executable even when it is presented as a flow diagram. In fact, even of recovering the control-flow graph from a binary is more complicated than one would expect [2] and, IDA Pro often fails to reconstruct the complete control-flow graph.

One notable body of work that also aims to support the reversing is the thesis work of Balakrishnan [1]. Balakrishnan, under the direction of Reps, has developed a so-called value set analysis that attempts to uniformly track addresses and numeric values. The rationale for this approach is that it enables sets of addresses on some word alignment to be accurately represented. We consider this approach to be a major advance in the analysis of binaries, since it attempts to seamlessly support addresses and numeric values. Recently it has also been shown how logical techniques based on bit-blasting and SAT solving can also be applied to extract bit-level modulo relationships [3].

The tutorial will review this growing body of work, highlighting the potential for applying logical methods and abstraction techniques in reversing.

References

1. Balakrishnan, G.: WYSINWYX: What You See Is Not What You eXecute. Ph.D thesis, Computer Sciences Department, University of Wisconsin, Madison (2007)
2. Kinder, J., Veith, H., Zuleger, F.: An abstract interpretation-based framework for control flow reconstruction from binaries. In: Muller-Olm, M. (ed.) VMCAI 2009. LNCS, vol. 5403, pp. 214–228. Springer, Heidelberg (2009)
3. King, A., Søndergaard, H.: Inferring Congruence Equations using SAT. In: Gupta, A., Malik, S. (eds.) CAV 2008. LNCS, vol. 5123, pp. 281–293. Springer, Heidelberg (2008)
4. Pennell, J.: Reverse Engineering with IDA Pro. IOActive (2008)

P.M. Hill and D.S. Warren (Eds.): ICLP 2009, LNCS 5649, p. 37, 2009.