1. Record Nr. UNINA9910831843103321

Autore	Rupprecht Thomas
Titolo	Data Structure Identification from Executions of Pointer Programs / Thomas Rupprecht . Volume 41
Pubbl/distr/stampa	[s.l.] : , : Bamberg University Press, , 2020
ISBN	9783863097189 3863097181
Descrizione fisica	1 online resource (1 p.)
Collana	Schriften aus der Fakultät Wirtschaftsinformatik und Angewandte Informatik
Soggetti	Computers / Data Science Computers
Lingua di pubblicazione	Inglese
Formato	Materiale a stampa
Livello bibliografico	Monografia
Sommario/riassunto	The reverse engineering of binaries is a tedious and time consuming task, yet mandatory when the need arises to understand the behaviour of a program for which source code is unavailable. Instances of source code loss for old arcade games [1] and the steadily growing amount of malware [2] are prominent use cases requiring reverse engineering. One of the challenges when dealing with binaries is the loss of low level type information, i.e., primitive and compound types, which even state-of-the-art type recovery tools often cannot reconstruct with full accuracy. Further programmers most commonly use high level data structures, such as linked lists, in addition to primitive types. Therefore detection of dynamic data structure shapes is an important aspect of reverse engineering. Though the recognition of dynamic data structure shapes in the presence of tricky programming concepts such as pointer arithmetic and casts - which are both fundamental concepts to enable, e.g., the frequently used Linux kernel list [3] - also bring current shape detection tools to their limits. A recent approach called Data Structure Investigator (DSI) [4] , aims for the detection of dynamic pointer based data structures. While the approach is general in nature, a concrete realization for C programs requiring source code is envisioned as programming constructs such as type casts and pointer arithmetic will

stress test the approach. Therefore, the first research question addressed in this dissertation is whether DSI can meet its goal in the presence of the sheer multitude of existing data structure implementations. The second research question is whether DSI can be opened up to reverse engineer C/C++ binaries, even in the presence of type information loss and the variety of C/C++ programming constructs. Both questions are answered positively in this dissertation. The first is answered by realizing the DSI source code approach, which requires detailing fundamental aspects of DSI's theory to arrive at a working implementation, e.g., handling the consistency of DSI's memory abstraction and quantifying the interconnections found within a dynamic pointer based data structure, e.g., a parent child nesting scenario, to allow for its detection. DSI's utility is evaluated on an extensive benchmark including real world examples (libusb [5], bash [6]) and shape analysis examples, [7,8] . The second question is answered through the development of a DSI prototype for binaries (DSIbin). To compensate for the loss of perfect type information found in source code, DSIbin interfaces with the state-of-the-art type recovery tool Howard [9]. Notably, DSIbin improves upon type information recovered by Howard. This is accomplished through a much improved nested struct detection and type merging algorithm, both of which are fundamental aspects for the reverse engineering of binaries. The proposed approach is again evaluated by a diverse benchmark containing real world examples such as, the VNC clipping library, The Computer Language Benchmarks Game and the Olden Benchmark, as well as examples taken from the shape analysis literature. In summary, this dissertation improves upon the state-of-the-art of shape detection and reverse engineering by (i) realizing and evaluating the DSI approach, which includes contributing to DSI's theory and results in the DSI prototype; (ii) opening up DSI for C/C++ binaries so as to extend DSI to reverse engineering, resulting in the DSIbin prototype; (iii) handling data structures with DSIbin not covered by some related work such as skip lists; (iv) refining the nesting detection and performing type merging for types excavated by Howard. Further, DSIbin's ultimate future use case of malware analysis is hardened by revealing the presence of dynamic data structures in multiple real world malware samples. In summary, this dissertation advanced the dynamic analysis of data structure shapes with the aforementioned contributions to the DSI approach for source code and further by transferring this new technology to the analysis of binaries. The latter resulted in the additional insight that high level dynamic data structure information can help to infer low level type information.