ConDySTA: Context-Aware Dynamic Supplement to Static Taint Analysis

Xueling Zhang, Xiaoyin Wang, Rocky Savin, Jianwei Niu
The University of Texas at San Antonio

Abstract
Static taint analyses are widely-applied techniques to detect taint flows in software systems. Although they are theoretically conservative and designed to detect all possible taint flows, static taint analyses almost always exhibit false negatives due to a variety of implementation limitations. Dynamic programming language features, inaccessible code, and the usage of multiple programming languages in a software project are some of the major causes. To alleviate this problem, we developed a novel approach, DySTA, which uses dynamic taint analysis results as additional sources for static taint analysis. However, naively adding sources causes static analysis to lose context sensitivity and thus produce false positives. Thus, we developed a hybrid context matching algorithm and a corresponding tool, ConDySTA, to preserve context sensitivity in DySTA. We applied REPRODROID, a comprehensive benchmarking framework for Android analysis tools, to evaluate ConDySTA. The results show that across 28 apps (1) ConDySTA was able to detect 12 out of 28 taint flows which were not detected by any of the six state-of-the-art static taint analyses considered in REPRODROID, and (2) ConDySTA reported no false positives, whereas nine were reported by DySTA alone. We further applied ConDySTA and FLOWDROID to 100 top Android apps from Google Play, and ConDySTA was able to detect 39 additional taint flows (besides 281 taint flows found by FLOWDROID) while preserving the context sensitivity of FLOWDROID.

Taint Analysis
Taint analysis can detect taint flows in software programs and has been widely used in privacy leak detection. Static taint analyses propagate taints based on an overestimation of all possible program paths leading to the detection of all possible taint flows with no false positives but some false positives due to infeasible paths.

ConDySTA: Context-Aware Dynamic Supplement to Static Taint Analysis

Xueling Zhang, Xiaoyin Wang, Rocky Savin, Jianwei Niu
The University of Texas at San Antonio

Abstract
Static taint analyses are widely-applied techniques to detect taint flows in software systems. Although they are theoretically conservative and designed to detect all possible taint flows, static taint analyses almost always exhibit false negatives due to a variety of implementation limitations. Dynamic programming language features, inaccessible code, and the usage of multiple programming languages in a software project are some of the major causes. To alleviate this problem, we developed a novel approach, DySTA, which uses dynamic taint analysis results as additional sources for static taint analysis. However, naively adding sources causes static analysis to lose context sensitivity and thus produce false positives. Thus, we developed a hybrid context matching algorithm and a corresponding tool, ConDySTA, to preserve context sensitivity in DySTA. We applied REPRODROID, a comprehensive benchmarking framework for Android analysis tools, to evaluate ConDySTA. The results show that across 28 apps (1) ConDySTA was able to detect 12 out of 28 taint flows which were not detected by any of the six state-of-the-art static taint analyses considered in REPRODROID, and (2) ConDySTA reported no false positives, whereas nine were reported by DySTA alone. We further applied ConDySTA and FLOWDROID to 100 top Android apps from Google Play, and ConDySTA was able to detect 39 additional taint flows (besides 281 taint flows found by FLOWDROID) while preserving the context sensitivity of FLOWDROID.

Taint Analysis
Taint analysis can detect taint flows in software programs and has been widely used in privacy leak detection. Static taint analyses propagate taints based on an overestimation of all possible program paths leading to the detection of all possible taint flows with no false positives but some false positives due to infeasible paths.

ConDySTA: Context-Aware Dynamic Supplement to Static Taint Analysis

Xueling Zhang, Xiaoyin Wang, Rocky Savin, Jianwei Niu
The University of Texas at San Antonio

Abstract
Static taint analyses are widely-applied techniques to detect taint flows in software systems. Although they are theoretically conservative and designed to detect all possible taint flows, static taint analyses almost always exhibit false negatives due to a variety of implementation limitations. Dynamic programming language features, inaccessible code, and the usage of multiple programming languages in a software project are some of the major causes. To alleviate this problem, we developed a novel approach, DySTA, which uses dynamic taint analysis results as additional sources for static taint analysis. However, naively adding sources causes static analysis to lose context sensitivity and thus produce false positives. Thus, we developed a hybrid context matching algorithm and a corresponding tool, ConDySTA, to preserve context sensitivity in DySTA. We applied REPRODROID, a comprehensive benchmarking framework for Android analysis tools, to evaluate ConDySTA. The results show that across 28 apps (1) ConDySTA was able to detect 12 out of 28 taint flows which were not detected by any of the six state-of-the-art static taint analyses considered in REPRODROID, and (2) ConDySTA reported no false positives, whereas nine were reported by DySTA alone. We further applied ConDySTA and FLOWDROID to 100 top Android apps from Google Play, and ConDySTA was able to detect 39 additional taint flows (besides 281 taint flows found by FLOWDROID) while preserving the context sensitivity of FLOWDROID.

Taint Analysis
Taint analysis can detect taint flows in software programs and has been widely used in privacy leak detection. Static taint analyses propagate taints based on an overestimation of all possible program paths leading to the detection of all possible taint flows with no false positives but some false positives due to infeasible paths.