Поиск фрагментов исходного кода, реализующих санитизацию, методами статического анализа

Зубрилин Антон Николаевич

Студент

Факультет ВМК МГУ имени М. В. Ломоносова, Москва, Россия

E-mail: a.zubrilin2014@yandex.ru

Научный руководитель *— Самосадный Кирилл Алексеевич*

Недостатки класса инъекция являются одними из наиболее распространенных и критичных в современных веб-приложениях. Одним из способов предотвращения появления таких недостатков является санитизация недоверенных данных перед обращением во внешнюю подсистему. Современные языки программирования и фреймворки предоставляют стандартные механизмы для выполнения санитизации. Однако разработчики могут реализовать санитизацию самостоятельно как часть кода своего приложения. Наличие в исходном коде нестандартной санитизации несет в себе риски уязвимостей из-за возможных ошибок при ее реализации, а также затрудняет анализ защищенности приложения, в ходе которого необходимо обнаруживать и анализировать нестандартную санитизацию. Поэтому актуальна задача поиска фрагментов исходного кода, реализующих нестандартную санитизацию.

В данном исследовании проведен обзор существующих работ, в которых решается данная задача [1–4]. На основе обзора сделан вывод, что в существующих методах поиска фрагментов кода, реализующих санитизацию, можно выделить 3 подзадачи:

1. выделение фрагментов кода для анализа (либо отдельные вызовы функций, либо целиком определения функций),

2. описание выделенных фрагментов набором свойств, вычислимых методами статического анализа кода:

   (a) потоки данных, в которых встречается фрагмент,

   (b) комментарии, примыкающие к фрагменту, и названия содержащих его программных сущностей как тексты на естественном языке,

   (c) состав и внутренняя структура фрагмента;

3. классификация фрагментов (реализует санитизацию/не реализует санитизацию) на основе вычисленных свойств.

В результате исследования предложен метод поиска фрагментов кода, реализующих санитизацию, улучшающий существующие в следующих аспектах:

1. разработан новый алгоритм выделения фрагментов кода, группирующий вызовы функций по их близости друг к другу в графе зависимостей по данным,

2. предложен алгоритм построения признакового описания фрагментов как числовых оценок свойств, описанных в существующих методах и дополненных метрикой схожести с ранее найденными фрагментами кода, реализующими санитизацию,

3. предложен алгоритм ранжирования выделенных фрагментов и выдачи итогового результата (набора фрагментов кода, реализующих санитизацию) на его основе.

Предложенный метод реализован для языка PHP и протестирован на реальных приложениях с открытым исходным кодом на языке PHP.

Литература

1. Livshits B. et. al. Merlin: Specification inference for explicit information flow problems //ACM Sigplan Notices. 2009. Т. 44. № 6. P. 75–86.
2. Chibotaru V. et. al. Scalable taint specification inference with big code //Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation. 2019. P. 760–774.
3. Chen X. et. al. Vulchecker: achieving more effective taint analysis by identifying sanitizers automatically //2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom). IEEE, 2021. P. 774–782.
4. Su H. et. al. A sanitizer-centric analysis to detect cross-site scripting in PHP programs //2022 IEEE 33rd International Symposium on Software Reliability Engineering (ISSRE). IEEE, 2022. P. 355–365.