УДК 004.056.5:004.272.2

Мостовий О.С., аспірант
Інститут кібернетики імені В.М.Глушкова НАН України

РОЗПОДІЛЕНИЙ SMT-АНАЛІЗ ВРАЗЛИВОСТЕЙ БІНАРНИХ ФАЙЛІВ АРХІТЕКТУРИ ARM64 НА HPC-КЛАСТЕРАХ

Аналіз безпеки ARM64-бінарних файлів є критичним для захисту програм IoT-пристроїв та супутникового обладнання. Символьне виконання за допомогою фреймворку Angr [1] будує граф потоку керування (ГПК), де кожен шлях до потенційно небезпечної операції породжує набір обмежень. Система автоматичного доведення теорем (SMT) Z3 [2] перевіряє здійсненність цих обмежень: SAT означає існування вхідного вектора, що експлуатує вразливість. Проте реальні бінарні файли мають тисячі шляхів, а SMT-розв'язання займає понад 90% часу виконання. Оскільки перевірка різних шляхів повністю незалежна, задача є паралельною та ідеальною для високопродуктивних обчислювальних кластерів (HPC).

Пропонується три етапна архітектура типу Map-Reduce (Рис 1)для розподіленого SMT-аналізу ARM64 бінарних файлів на HPC-кластері СКІТ [3] під керуванням планувальника Slurm.

Етап 1 : на вузлі фреймворк angr будує CFG бінарного файлу, досліджує шляхи до вузлів, для кожного формує повну формулу $\Phi_i = Pre \wedge \varphi_i \wedge Post$ та серіалізує у формат SMT-LIB v2. Результат — директорія з N незалежними файлами.

Етап 2 : масив завдань Slurm розподіляє файли по вузлах. Кожен сервіс планувальник отримує унікальний ідентифікатор, запускає Z3 із часом очікування 60 секунд та записує результат: SAT, UNSAT, або timeout.

Етап 3 : фінальний скрипт збирає результати, класифікує за типом CWE та формує звіт. SAT-результати містять конкретні моделі - вхідні дані для відтворення вразливості.

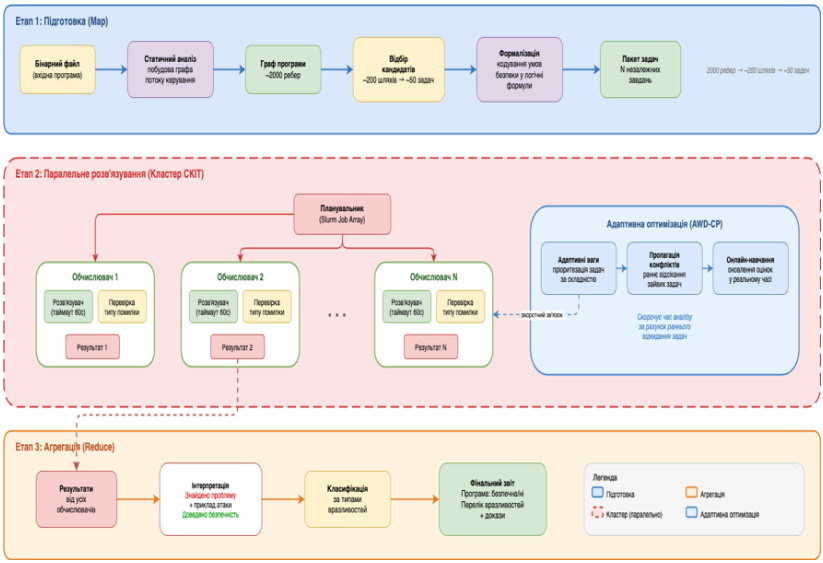

Рисунок 1 – Архітектура Розподіленого SMT-Аналізу

Запропоновано архітектуру розподіленого SMT-аналізу вразливостей ARM64-бінарних файлів, у якій кожному класу ставиться у відповідність трійка Хоара , що транслюється у формулу SMT-LIB v2 для розв'язувача Z3. Обчислювальна реалізація за парадигмою Map-Reduce на кластері СКІТ забезпечує лінійне масштабування для незалежних шляхів виконання. Подальша робота спрямована на імплементацію прототипу, інтеграцію методу та верифікацію програмного забезпечення.

Список використаних джерел

1. Shoshitaishvili Y. et al. SOK: Offensive techniques in binary analysis. IEEE S&P. 2016.

2. De Moura L., Bjørner N. Z3: An efficient SMT solver. TACAS. 2008. P. 337–340.

3. Golovynskyi A.L. et al. Development of SCIT supercomputers family. Cybernetics and Sys. Analysis. 2017. Vol. 53, No. 5.

4. Wilson A. et al. Partitioninфg strategies for distributed SMT solving. FMCAD. 2023.

5. Bucur S. et al. Parallel symbolic execution for automated real-world software testing. EuroSys. 2011.