Применение компиляторных преобразований для повышения стойкости программного обеспечения к эксплуатации уязвимостей

Нурмухаметов Алексей Раисович[1]
Саргсян Севак Сеникович[2]

1: *Аспирант, ИСП РАН, Москва, Россия*
2: *Аспирант, МФТИ (ГУ), Москва, Россия*
E-mail: oleshka@ispras.ru, sevaksargsyan@ispras.ru

Программное обеспечение, написанное на широко распространенных языках C/C++, потенциально содержит в себе значительное количество уязвимостей (порядка одной ошибки на каждую тысячу строк кода [2]), используя которые злоумышленник может с помощью специально подготовленных эксплойтов захватить контроль над системой. Для повышения стойкости к эксплуатации уязвимостей существует несколько различных подходов: обнаружение уязвимостей путем проверки исходного или бинарного кода человеком и автоматическими анализаторами с последующим исправлением; модификация среды выполнения программы (неисполняемый стек, рандомизация адресного пространства процесса); компиляторные преобразования [1], использующие различные приемы для контроля за порядком исполнения программы и усложняющие атакующему планирование атаки.

Было проведено исследование возможности применения компиляторных преобразований для повышения стойкости программного обеспечения к эксплуатации уязвимостей переполнения буфера на стеке [3].

Большое количество приложений в нынешнее время распространяется через магазины приложений. Предлагается подход, при котором в такой схеме для каждого клиента будет компилироваться своя версия бинарного кода приложения, отличающаяся от других версий этого же приложения у других клиентов.

Для осуществления этого подхода были реализованы следующие трансформации: случайная перестановка местами функций, добавление случайного числа локальных переменных в функции и их перемешивание с используемыми в функции на стеке.

Для проверки применимости рассматриваемых компиляторных преобразований были проведены испытания. Они проводились над программой, содержащей в себе уязвимость переполнения буфера,

компилировались две ее версии. Первая версия компилировалась с применением разработанных преобразований, вторая - без применения. После этого на обе версии проводились атаки, заранее приготовленным эксплойтом. В случае с первой версией атакующему не удалось заполучить контроль над системой, приложение аварийно завершило свою работу с ошибкой сегментации. Во втором случае атакующий успешно перехватил управление. Проведенное испытание показывает принципиальную возможность успешного применения реализованных компиляторных преобразований для защиты от эксплуатации уязвимостей.

Замеры производительности показали падение производительности на 30% на приложении SQLite [4].

Проведенное исследование показывает, что использование предлагаемых методов повышает уровень стойкости приложений к атакам на уязвимые места, более того обнаружение уязвимости и создание рабочей эксплуатации уязвимостей для одного экземпляра защищенного приложения не угрожает безопасности других экземпляров этого приложения. Исследуемый метод успешно сочетается с другими методами защиты и может служить полезным звеном в обширном наборе инструментов для защиты программного обеспечения.

Литература

1. Ахо А., Лам М., Сети Р., Ульман Д. Компиляторы. Принципы, технологии, инструментарий, 2-изд.: Пер с англ.- М.: ООО «И.Д. Вильямс», 2008. – 1184с.

2. Dazhi Z., Detecting Program Vulnerabilities Using Trace-Based Security Testing, Ph. D. Dissertation, University of Texas at Arlington, Arlington, TX, USA, Advisor(s) Donggang L, AAI3474008, 2011.

3. Foster J. C., Osipov V., Bhalla N., Buffer Overflow Attacks, Syngress Publishing, 2005

4. SQLite. `http://www.sqlite.org/`