54. ИНСТРУМЕНТЫ ТЕСТИРОВАНИЯ БЕЗОПАСНОСТИ ПРИЛОЖЕНИЙ

Худницкий А.А., студент гр. 272303, Липницкая Н.И., ассистент кафедры ЭИ

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Ефремов А.А. – канд. эк. наук, доцент кафедры ЭИ

Аннотация. Данная работа предлагает общий обзор способов тестирования безопасности приложений, их преимущества и недостатки, также особенности каждого из способов. Особое внимание уделяется инструментам тестирования безопасности приложений, способам их использования, особенности каждого инструмента, а также застрагивается вопрос способа выбора правильного инструмента.

Ключевые слова. Безопасность, инструменты, способ, тестирование, приложение, код.

Обеспечение должной безопасности приложений — это очень важная, актуальная и не простая задача, с которой сталкиваются все команды, разрабатывающие соответствующие продукты. Думать о безопасности следует начинать как можно раньше. Тестирование безопасности обеспечивает обратную связь команде о состоянии продукта. Согласно ISO 9126 тестирование безопасности является частью функционального тестирования. Тестирование безопасности определяют как проверку того, что программное обеспечение защищено от внешних атак: в частности, включает проверки конфиденциальности, целостности и доступности систем и их данных.

Согласно исследованию Positive Technologies 82% уязвимостей веб-приложений находятся в исходном коде. В среднем каждая вторая система содержит уязвимость с высоким уровнем опасности. Одна из основных причин этой проблемы — отсутствие проверки кода на наличие уязвимостей на этапе написания.

В ответ на угрозы и постоянно растущую кодовую базу разработчики используют инструменты тестирования безопасности приложений (AST). AST — это процесс повышения безопасности приложений путем выявления уязвимостей в исходном коде.

Использование инструментов тестирования — важная часть концепции DevSecOps. Согласно отчету Transparency Market Research, рынок AST-инструментов разделен на следующие классы продуктов, которые мы рассмотрим.

Первым способом является статическое тестирование безопасности приложений. Это тестирование на наличие ошибок и уязвимостей в исходном коде. SAST является одним из основных вариантов поиска уязвимостей в коде. Другие преимущества SAST включают в себя:

- Возможность интегрировать статический анализ в процесс разработки;
- Обнаружение критических уязвимостей, таких как переполнение буфера, SQL-инъекция, межсайтовый скриптинг (XSS) и другие;
- Указание точного местоположения подозрительного фрагмента кода. Это особенно важно для больших проектов с тысячами и миллионами строк кода.

Однако, у технологии SAST есть свои недостатки, которые включают в себя большое количество ложных срабатываний. Из-за этого проверка результатов может занять много времени.

Следующий способ — динамическое тестирование безопасности приложений. Динамическое тестирование безопасности приложений имитирует вредоносные атаки, которые используют распространенные уязвимости. Основная задача DAST — выявить ошибки до того, как их обнаружит злоумышленник. Такие инструменты ищут уязвимые области, проверяя точки доступа и имитируя взаимодействие с пользователем. Преимущества DAST:

- В отличие от SAST, он позволяет разработчикам обнаруживать проблемы во время выполнения кода. Это могут быть недостатки аутентификации и настройки сети, либо проблемы, возникающие только после входа в систему;
- DAST находит ошибки, возникающие при работе пользователя с приложением;
- DAST не привязан к языкам программирования.

Изначально DAST-инструменты использовались реже, чем SAST. Но в связи с распространением смартфонов, в которых появляется все больше приложений, связанных с конфиденциальной информацией, доля DAST-решений значительно увеличилась и продолжает расти.

Интерактивное тестирование безопасности приложений – относительно новый (по сравнению с SAST и DAST) метод, который позволяет анализировать приложение изнутри во время его работы.

Другими словами:

- SAST работает с кодом без запуска приложения.
- DAST может работать с запущенным приложением, но без доступа к коду.
- IAST работает с кодом в работающем приложении.

IAST может охватывать больший объем кода, давать более точные результаты и проверять более широкий набор правил безопасности, по сравнению с SAST и DAST. Кроме того, IAST выявляет больше уязвимостей без ложных срабатываний.

Однако, IAST имеет и недостатки. Одним из основных минусов является то, что IAST-инструменты могут замедлять работу приложения и снижать производительность кода[1].

Теперь стоит уделить внимание четырем конкретным примерам инструментов, которые чаще всего используются.

Первым является w3af. w3af — это бесплатный сканер безопасности веб-приложений с открытым исходным кодом, позволяющий проводить тестирование как в ручном так и автоматизированном режимах, с использованием графического интерфейса и интерфейса командной строки. Сканер способен идентифицировать более 200 уязвимостей, в том числе межсайтовый скриптинг, SQL-инъекции. Среди особенностей хотелось бы отметить:

- Широкий спектр плагинов.
- Информативный отчет с результатами тестирования, с возможностью экспорта данных в различных форматах.
- Неустойчивая работа под Windows.

Следующим примером является SQLMap. SQLMap — бесплатный сканер с открытым исходным кодом, главная задача которого — автоматизированный поиск и эксплуатация SQL-инъекций. Основной особенностью сканера является эксплуатация найденной уязвимости, кроме того это:

- поддержка 5 основных классов SQL-инъекций: boolean-based blind, time-based blind, error-based, UNION query, stacked queries.
- поддержка различных баз данных: MySQL, Oracle, PostgreSQL, Microsoft SQL Server.

Burp Suite — это платная (бесплатная содержит ограниченный функционал) мощная платформа для тестирования безопасности веб-приложений как в ручном так и автоматизированном режимах с интуитивно понятным интерфейсом, включает инструменты по созданию карты веб-приложения, изменения запросов, фаззинга и т.д. Является одним из самых популярных инструментов тестирования безопасности, как говорится «его рекомендуют друзьям».

Среди особенностей стоит отметить:

- возможность обнаружения большого числа разнообразных уязвимостей.
- поддержку ряда веб-технологии: REST, JSON, AJAX и SOAP,
- возможность тестирования безопасности мобильный приложений платформы iOS (Burp Suite Mobile Assistant).

OWASP ZAP— один из самых известных инструментов для тестирования безопасности веб-приложений. Является бесплатным кросс-платформенным, простым в использовании, мультиязычным (переведен на множество языков) инструментом с открытым исходным кодом. Включает в себя: перехватывающий прокси, активный и пассивный сканеры, поддержку веб-сокетов, поддержку аутентификаций и сессий, поддержку большого кол-ва скриптовых языков и т.д[2].

Остается самый главный вопрос: как выбрать правильные инструменты. Недостатка в инструментах нет, и можно запутаться в выборе вариантов. В целом, существуют Open Source-инструменты, лучшие в своем классе инструменты. Инструменты с открытым исходным кодом, как правило, очень тактические по своей природе, сфокусированные на чем-то одном. В качестве примера можно привести бесплатный сканер безопасности веб-приложений w3af, бесплатное средство проверки качества кода и уязвимостей Snyk, SQLmap. Если бюджет является проблемой, Уортингтон рекомендует начать с бесплатной версии инструмента тестирования, которые сейчас предлагают многие поставщики. Например, компания Snyk, известная своим инструментом анализа состава ПО, имеет бесплатную Open Source-версию. После того, как инструмент доказал свою ценность, можно решить, стоит ли платить за полнофункциональную версию[3].

Подводя итоги, можно сказать, что тестирование безопасности приложений является очень важной темой в современном мире. Каждый человек, который хоть немного связан с разработкой приложений должен быть погружен в тему безопасности, следить за тенденциями в этой сфере, знакомиться с новыми продуктами, поступающими на рынок инструментов, а также активно пользоваться ими. Ознакомившись с материалами, связанными с данной темой, лично я понял, что без этого невозможно представить современное программирование, поэтому буду продолжать разбираться в ней и делать безопасным продукт, который буду выпускать.

Список использованных источников:

[1] Способы тестирования безопасности приложений [Электронный ресурс]. – Режим доступа: https://www.securitylab.ru/analytics/533602.php– Дата доступа: 09.04.2023

[2] Как выбрать инструмент для тестирования безопасности ПО [Электронный ресурс]. – Режим доступа: https://www.itweek.ru/security/article/detail.php?ID=224323 – Дата доступа: 09.04.2023

[3] Инструменты для тестирования безопасности [Электронный ресурс]. – Режим доступа: https://testerchronicles.ru/security-testing-tools/– Дата доступа: 09.04.2023