применяемые в RAID массивах, с дополнительными возможностями по фильтрации файлов.

Несмотря на все преимущества использования облачных хранилищ существует вероятность отказа, вызванная временным или постоянным прекращением доступа к облаку. В целях снижения вероятности наступления подобного события, предлагается программное решение, основанное на взаимодействии множества автономных клиентских модулей и решающее такие задачи как:

- Обеспечение гибкой маршрутизации передачи данных в случае сбоев и отказов в каналах передачи данных.
- Фоновая репликация данных в целях минимизации вероятности их утери при прекращении доступа к некоторому облачному сервису.
- Кэширование данных на узлах связи, обеспечивающих наименьшее время доступа для того или иного пользователя.

Предлагаемое программное средство является универсальным в плане форматов передаваемых данных и подходит как для текстовых файлов, HTML-документов, презентаций, так и для большого числа форматов, используемых для хранения пользовательской данных.

Список использованных источников:

1. Erl T. Cloud Computing: Concepts, Technology & Architecture. / Erl T., Puttini R., - Prentice Hall. – 2013.
2. Kavis M. Architecting the Cloud: Design Decisions for Cloud Computing Service Models (SaaS, PaaS, and IaaS). / Kavis M. – Wiley.. – 2014.
3. Yeluri R. Building the Infrastructure for Cloud Security: A Solutions View (Expert's Voice in Internet Security). / Yeluri R., Castro-Leon E. – Apress. – 2014.

МОДЕЛЬ ОБНАРУЖЕНИЯ УЯЗВИМОСТЕЙ В WEB-ПРИЛОЖЕНИЯХ

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

Оношко Д.Е.

Бахтизин В.В. — к.т.н., профессор

Высокая популярность web-приложений и их широкое применение в различных областях обусловили высокую значимость вопросов их качества, причём в первую очередь — надёжности и безопасности.

Наиболее распространённой угрозой для различных типов приложений (включая web-приложения) по данным OWASP являются SQL-инъекции [1]. Причиной уязвимости web-приложений к SQL-инъекциям является наличие ошибок, позволяющих полученным от пользователя данным быть подставленными в текст запроса к системе управления базами данных (СУБД) без необходимой фильтрации.

Наибольшей популярностью для решения задачи обнаружения таких уязвимостей пользуются методы, основанные на динамическом анализе поведения web-приложения в условиях эксплуатации. Между тем, являясь по сути частным случаем функционального тестирования, такие методы в значительной степени зависят от правильности подбора тестовых данных и способны выявлять только некоторое подмножество уязвимостей. Единственным способом гарантированного обнаружения всех ошибок, позволяющих провести SQL-инъекцию, является исходных кодов, который, между тем, является рутинной и трудоёмкой процедурой, ввиду чего целесообразна её автоматизация с помощью ПС контроля качества кода, ориентированных на обнаружение потенциальных уязвимостей.

Такие ПС могут быть основаны на простой модели, рассматривающей web-приложение как множество $P = \{P_1, P_2, \ldots, P_N\}$ процедур (в т.ч. операторов языка), которые вызывают друг друга с некоторыми параметрами. Формальным параметрам и возвращаемым значениям процедур назначаются оценки, в простейшем случае — бинарного характера: «опасный» (U) или «безопасный» (S). При этом параметры, рассматриваемые в рамках модели, могут быть двух видов: in-параметры (данные, передаваемые в процедуру) и out-параметры (данные, возвращаемые из процедуры).

Первоначально анализатору (ПС) известны оценки только для некоторые стандартных процедур, а также для операторов языка программирования. Пусть на i-м шаге известны оценки параметров и возвращаемых значений для процедур $P'(i) = \{P_1, P_2, \ldots, P_{C(i)}\}$, где $C(i)$ — количество таких процедур на i-м шаге, а процедурой $P_{C(i)+1}$ используются только процедуры из $P'(i)$. Тогда, анализируя операторы $P_{C(i)+1}$, можно получить оценки её параметров и возвращаемых значений: оценка фактического параметра совпадает с оценкой формального параметра. Последней анализируемой процедурой является

процедура P_N, представляющая основную программу (главный блок begin…end, функцию ёmain() и т.п.). Вычисленные оценки формальных параметров P_N (им соответствуют поступающие от пользователя данные) должны иметь значение «опасный». Параметры, для которых это не выполняется, являются потенциально уязвимыми. Анализируя путь, по которому была получена оценка, можно выявить причину возникновения уязвимости и предложить способы её устранения.

В общем случае можно сформулировать 5 правила, описывающие процесс назначения оценок и проверки web-приложения на наличие уязвимостей к SQL-инъекциям.

1. В качестве in-параметра с оценкой S должны передаваться только данные, имеющие оценку S.
2. В качестве in-параметра с оценкой U могут передаваться любые данные.
3. Переменным, переданным в процедуру в качестве out-параметров, назначается оценка, совпадающая с оценкой соответствующего out-параметра процедуры.
4. При наличии у переменной или out-параметра нескольких различных оценок выбирается «наихудшая», т.е. в случае бинарной оценки предпочтение отдаётся оценке U.
5. При наличии у in-параметра нескольких различных оценок выбирается «наилучшая», т.е. в случае бинарной оценки предпочтение отдаётся оценке S.

Предлагаемая модель позволяет обнаруживать не только эксплуатируемые, т.е. действительно позволяющие злоумышленнику провести атаку, уязвимости, но и потенциальные — не позволяющие провести атаку в данной версии приложения, но способные стать эксплуатируемыми после внесения изменений в исходный код web-приложения, причём необязательно в проблемный участок кода.

Между тем, поскольку в некоторых случаях процедуры web-приложения могут, выполняя определённые преобразования, произвести фильтрацию данных, которая не будет распознана в рамках модели, целесообразно предусмотреть дополнительную оценку UDS (User-Defined Safe), которая позволит программисту явно обозначить те или иные параметры процедур, как безопасные, независимо от результатов анализа. Использование такой оценки позволяет сократить число ложных срабатываний.

Предложенная модель может использоваться как в качестве самостоятельного инструмента, для анализа web-приложений на предмет уязвимости к SQL-инъекциям, так и в качестве вспомогательного инструменты при обеспечении качестве web-приложений, поставляющего исходные данные (сведения о количестве и расположении проблемных участков кода) для модели качества.

Список использованных источников:

1. OWASP Top 10-2013. The Ten Most Critical Web Application Security Risks [Электронный ресурс]. — Режим доступа: http://owasptop10.googlecode.com/files/OWASP%20Top%2010%20-%202013.pdf. — Дата доступа: 31.10.2013.

АЛГОРИТМ ПОВЫШЕНИЯ ТОЧНОСТИ ЗАШУМЛЕННЫХ ГЕОДАННЫХ

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

Базаревский В.Э.

Бранцевич П.Ю.,к.т.н, доцент

Развитие социальных сетей, а так же мобильных технологий (в том числе повсеместное внедрение точных акселерометров, компасов и gps-локаторов) позволило с относительно малой трудоемкостью создавать удобные геоприложения различной бизнес-направленности.

В качестве одного из таких приложений может быть рассмотрено приложение поиска субъекта в незнакомой местности (например,это может быть новый незнакомый район или город во время заграничнойпоездки). К сожалению, не смотря на относительную точность датчиков современных мобильных устройств (погрешность 50 м), этого недостаточно для точного определения “визави” в людных местах, когда на площади 50*50 метров может находиться несколько сотен людей. Более того, точность такого геопозиционирования зачастую оказывается еще меньше в помещениях, что объясняется искажением магнитных полей от железобетонных конструкций, а так же искажением распространения радиоволн в разных материалах.

Зачастую, основным решением, предлагаемым в качестве решения проблемы точности геопозиционирования является использованием так называемых beacon-ов, датчиков, работающих по bluetooth протоколу на небольшом расстоянии (при этом мобильное устройство так же может выступать в качестве beacona). Такой подход хорошо работает при необходимости обнаружить, в какой конкретно геоточке находится пользователь в данный момент с большой точностью (такой подход используется, например, при показе таргетированной рекламы в торговых центрах), однако плохо работает при поиске необходимой конкретной точки. Это объясняется тем, что beacon-ы работая по bluetooth протоколу могут сообщать информацию только о том, насколько силен сигнал до beacona, с возможностью последующей аппроксимации этих данных в расстояние до устройства. Такимобразом возможно получение только метрики приращения расстояния, без возможности получения информации о изменении относительных координат