але большая частка яго не мае. У любым выпадку, вызначэнне стандартнага прэфікса спрашчае задачу алгарытма.

Трэці этап – вызначэнне стандартных суфіксаў і канчаткаў. Для гэтага аналізуецца канцавы сегмент слова. Праводзіцца пошук суфіксаў, на якія звычайна падае націск (-ятк-, -еньк-), а таксама ненаціскных суфіксаў (напрыклад, паслянаціскны -нік). Акрамя таго, разглядаюцца другія часткі складаных словаў (-здольны) і стандартных частак спецыфічных прозвішчаў (прозвішчы на -швілі і -адзэ).

Апошні этап – прадказанне націску на аснове статыстыкі. У аснове механізма – гіпотэза аб тым, што ў большасці выпадкаў можна вызначыць месца націску ў слове па яго канцавой частцы. Алгарытм працуе з загадзя пабудаваным слоўнікам на аснове слоў з вядомым націскам з базы дадзеных. Гэты слоўнік утрымлівае канцавыя часткі слоў і “правілы”, якія вызначаюць націск для ўваходнага слова на аснове яго марфалагічнай прыналежнасці і колькасці складоў. Эксперыментальным шляхам было вызначана, што даўжыня канцавой часткі павінна быць не менш за 4 сімвалы, а працэнт слоў з найбольш імаверным месцам націску – не менш за 80. Так, напрыклад, для 4-складовых словаў з канцавой часткай –лава у 93% выпадкаў націск ставіцца на перадапошні склад.

Атрыманы алгарытм, які дазваляе з высокай імавернасцю вызначаць націскны склад у незнаёмых словах, быў распрацаваны ў межах стварэння лінгвістычнай інфармацыйна-пошукавай сістэмы для аўтаматызацыі падбору рыфм і напісання вершаваных радкоў. Таксама дадзены алгарытм можа быць выкарыстаны ў сістэмах аўтаматычнага сінтэзу маўлення, аналізатарах галасавых каманд і іншых сферах.

Спіс выкарыстаных крыніц:

1. Кипяткова, И.С. Разработка и оценивание модуля транскрибирования для распознавания и синтеза русской речи / И.С. Кипяткова, А.А. Карпов. - Научно-теоретический журнал "Искусственный интеллект" No.3'2009.
2. Кондратов, А. Математика и поэзия. /А. Кондратов - М.: Знание, 1962. - 42 с.
3. Слоўнік.орг [Электронны рэсурс]. – 2015 – Рэжым доступу: http://www.slounik.org / Дата доступу : 22.02.2015

ПРОГРАММНОЕ СРЕДСТВО УПРАВЛЕНИЯ РАСПРЕДЕЛЕНИЕМ ДАННЫХ МЕЖДУ ОБЛАЧНЫМИ ХРАНИЛИЩАМИ

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

Гавриленко Д.В.

Куликов С. С. – к.т.н., доцент

Современный этап развития общества подразумевает эффективное управление данными, и в данном контексте особую важность приобретают такие системы, в которых данные хранятся на многочисленных распределённых в сети серверах – так называемых облачных хранилищах данных.

В противовес модели хранения данных на собственных (выделенных) серверах, приобретаемых или арендуемых специально для нужд функционирования системы, внутренняя инфраструктура облачного хранилища может быть неизвестна потребителю услуг: данные хранятся и обрабатываются в так называемом «облаке», которое представляет собой с точки зрения клиента один большой виртуальный сервер [1].

Облачные шлюзы – технология, которая может быть использована для более удобного представления облака клиенту и более гибкой настройки многопользовательских высоконагруженных систем. К примеру, с помощью соответствующего программного обеспечения хранилище в облаке может быть представлено для клиента как локальный диск на компьютере. Таким образом, работа с данными в облаке для клиента становится абсолютно прозрачной. И при наличии хорошей, быстрой связи с облаком клиент может даже не замечать, что работает не с локальными данными у себя на компьютере, а с данными, хранящимися, возможно, за много сотен километров от него.

Одним из важных вопросов является безопасность передаваемых и хранимых данных, поэтому важно, чтобы облачный сервис использовал надёжные протоколы шифрования, что на сегодняшний день в полной мере поддерживается большинством стандартных решений в области облачного хранения данных [2].

Другим важным показателем качества облачного хранилища является надёжность хранения и доступность данных в облаке: здесь стоит отметить, что основную проблему для отечественных потребителей услуг облачного хранения данных представляют некачественные и медленные каналы передачи данных, т.к. сами по себе облачные хранилища обеспечивают в среднем более высокую надёжность хранения данных, чем локальные решения, что достигается за счёт многократного нагруженного резервирования, распределения данных между несколькими узлами хранения и иных технологических решений [3].

Кроме того, не стоит полностью доверять хранение важных документов какому-либо одному сервису, так как известны случаи удаления файлов пользователей по инициативе владельцев сервисов, например, если появились основания полагать, что файл нарушает авторские права. Все это наводит на мысли некоторого объединения нескольких облачных сервисов с использованием технологий, похожих на

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