МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

МАТЕРИАЛЫ

**VIII Международной молодежной научной
конференции**

**«МАТЕМАТИЧЕСКОЕ
И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ
ИНФОРМАЦИОННЫХ,
ТЕХНИЧЕСКИХ
И ЭКОНОМИЧЕСКИХ СИСТЕМ»**

Томск, 26–30 мая 2021 г.

Под общей редакцией И.С. Шмырина

Томск
Издательство Томского государственного университета
2021

Заключение

Предложен подход к обфускации комбинационных схем на основе введения более простых элементов и подсхем по сравнению с традиционно используемыми элементами XOR, XNOR. Подход ориентирован как на затруднение обнаружения искажений поведения схемы, так и на последующий поиск ключа. Для поиска линий с целью введения в них дополнительных элементов предлагается использовать вычисление частичных функций вместо традиционно применяемого моделирования. Моделирование, в отличие от предлагаемого подхода, позволяет лишь приближенно оценивать подходящие линии, в особенности для схем с несколькими десятками и более входов.

ЛИТЕРАТУРА

1. Золоторевич Л.А. Аппаратная защита цифровых устройств // Вестник Томского государственного университета. Управление, вычислительная техника и информатика. – 2020. – № 50. – С. 69–78.
2. Yasin M., Rajendran J., Sinanoglu O., Karri R. On Improving the Security of Logic Locking // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. – 2016. – V. 35. – No. 9. – P. 1411–1424.
3. Karousos N., Pexaras K., Karybali I.G., Kalligeros E. Weighted Logic Locking: a New Approach for IC Piracy Protection // IEEE 23rd Int. Symposium on On-Line Testing and Robust System Design (IOLTS). – 2017. – P. 221–226.
4. Chakraborty R.S., Bhunia S. Security against Hardware Trojan through a Novel Application of Design Obfuscation // IEEE/ACM Int. Conference on Computer-Aided Design. – 2009. – P. 113–116.
5. Roy J.A., Koushanfar F., Markov I.L. EPIC: Ending Piracy of Integrated Circuits // IEEE Computer. – 2010. – V. 43. – No. 10. – P. 30–38.
6. Dupuis S., Rouzeyre B., Flottes M.-L., Natale G.D., Ba P.-S. New Testing Procedure for Finding Insertion Sites of Stealthy Hardware Trojans // DATE: Design, Automation and Test in Europe. Grenoble, 2015. – P. 776–781.
7. Xiao K., Forte D., Jin Y., Karri R., Bhunia S., Tehranipoor M. Hardware Trojans: Lessons learned after one decade of research // ACM transactions on design automation of electronic system. – 2016. – V. 22. – No. 1. – P. 1–23.
8. Shakya B., Salmani T.H., Forte D., Bhunia S., Tehranipoor M. Benchmarking of hardware Trojans and maliciously affected circuits // J. Hardw. Syst. Secur. (HaSS). – 2017. – V. 1 (1). – P. 85–102.
9. Сергейчик В.В., Иванюк А.А. Методы лексической обфускации VHDL-описаний // Information Technologies and Systems 2013 (ITS 2013) : Proc. of The Int. Conference. BSUIR. Minsk, 2013. – С. 198–199.
10. Голубева О.И. Алгоритмы получения функций тестопригодности полюса элемента комбинационной схемы // Современные проблемы физико-математических наук : материалы VI Всероссийской научно-практической конференции с международным участием (4–5 декабря 2020 г., г. Орел): научное электронное издание. Орел: ОГУ им. И.С. Тургенева, 2020. – С. 327–335.
11. Golubeva O. BDD and DNF Based Algorithms for Constructing All Testability Functions of Combinational Circuit // 2021 International Siberian Conference on Control and Communications (SIBCON), 2021, – P. 1–6. – doi: 10.1109/SIBCON50419.2021.9438879.

DOI: 10.17223/978-5-907442-42-9-2021-39

СОЗДАНИЕ ПРОГРАММЫ ДЛЯ АВТОМАТИЗАЦИИ УДАЛЕНИЯ ПУБЛИКАЦИЙ В СООБЩЕСТВЕ СОЦИАЛЬНОЙ СЕТИ «ВКОНТАКТЕ»

Савенкова М.М., Пахомова Е.Г.

Томский государственный университет
marina1412_11@mail.ru, PakhomovaEG@yandex.ru

Введение

В связи с тем, что в настоящее время социальные сети стали площадками не только для общения и развлечения, но и широко используются в качестве мест продвижения услуг и товаров, приёма коммерческих заказов и осуществления работы с огромной аудиторией, встает вопрос об автоматизации управления публикациями в сообществах. Проблема тем более представляется актуальной, что со временем количество контента в социальных сетях увеличивается, а поисковые системы ужесточают требования к качеству информации на страницах социальных сетей. Их алгоритмы ранжирования информации выдвигают на первые страницы в выдаче только свежий и наиболее востребованный контент.

Несмотря на относительное многообразие существующих программных решений в контексте общей задачи бывает затруднительно выбрать подходящее, т.к. требования к логике приложения для конкретного заказчика могут быть уникальными и оригинальными. Таким образом, в целях управления большими объемами сообщений на стене сообщества, поддержания информации на стене в актуальном состоянии, управления частными объявлениями пользователей сети требуется создание программного обеспечения, предназначенного для реализации конкретных требований.

В данной работе будет рассмотрено создание бота, поддерживающего в актуальном состоянии информацию на стене сообщества социальной сети «ВКонтакте», которая является одной из наиболее популярных социальных сетей в русскоязычном сегменте интернета. Поддержание актуальности будет осуществляться путем своевременного удаления утративших важность публикаций согласно настройкам пользователя. Также дается краткий обзор инструментов для реализации данной задачи, которые предоставляет эта социальная сеть.

1. Описание API «ВКонтакте»

Поскольку «ВКонтакте» представляет собой интернет-сообщество, то бота лучше разрабатывать на web-языке программирования и размещать в Интернете на хостинге. Хостинг (hosting) – это услуга, предоставляемая компаниями хостинг-провайдеринга, которая заключается в размещении информационных ресурсов пользователей на серверах компаний. Хостинги отличаются наличием тех или иных служб, возможностью поддержки различных технологий, языков программирования и скриптов.

В соответствии с рекомендациями команды разработчиков «ВКонтакте», можно использовать два подхода – использовать SDK на Java и PHP [1]. Кроме всего прочего, для реализации бота нужен web-сервер в сети Интернет с публичным IP-адресом. На данный момент самым распространенным видом хостинга является хостинг с поддержкой PHP, поэтому бот и будет реализован на PHP.

«ВКонтакте» предоставляет управление сообществом с помощью API (англ. Application Programming Interface). Для разработки бота потребовалось использовать Callback API и LongPoll API [1]. Кроме этого, понадобилось использовать авторизацию с помощью StandAlone приложения (способ OAuth) [1].

Callback API представляет из себя инструмент, который позволяет отслеживать все новые события, происходящие в сообществе. Для начала работы с Callback API необходимо подключить сервер в настройках сообщества («Управление сообществом» → «Настройки» → «Работа с API») и выбрать типы событий, данные о которых требуется получать (рис. 1а).

Рис. 1. Настройки API в сообществе

Когда в сообществе происходит заданное событие, «ВКонтакте» отправляет на сервер запрос с данными в формате JSON, заключающем в себе информацию об объекте, вызвавшем событие (например, добавленный комментарий). JSON (англ. JavaScript Object Notation) – текстовый формат обмена данными, основанный на JavaScript. Несмотря на происхождение от JavaScript, формат считается независимым от языка и может использоваться практически с любым языком программирования. Для многих языков существует готовый код для создания и обработки данных в формате JSON [2].

Long Polling – это технология, которая позволяет получать информацию о новых событиях с помощью «длинных запросов». Сервер получает запрос, но отправляет ответ на него не сразу, а лишь тогда, когда произойдет какое-либо событие (например, поступит новое входящее сообщение), либо истечет заданное время ожидания.

Другими словами, получая запрос, сервер ждет, когда произойдут события, о которых он должен вас уведомить, и, когда они произойдут, LongPoll-сервер отправляет ответ на ваш запрос, содержащий информацию о случившихся событиях. Используя этот подход, можно отображать в своем приложении важные события.

Настройки сервера производятся в специальном разделе на странице сообщества. Также в разделе под названием «Управление» необходимо получить ключи доступа (токены) для взаимодействия с сервером и использования Callback API (рис. 1б).

2. Анализ программного кода

В данной работе реализация бота представлена тремя файлами:

- *index.php* – страница для взаимодействия с Callback API и сбора статистики.
- *bot.php* – взаимодействие с LongPoll API и просмотр новых сообщений.
- *user.php* – задание параметров управления публикациями.

Также на сервере имеются некоторые вспомогательные файлы, о которых будет упомянуто позже.

2.1. Файл *index.php*

В файле *index.php* содержится скрипт, логика которого построена на основании значения суперглобального массива `$_GET`. `$_GET` – ассоциативный массив перемен-

ных, переданных скрипту через параметры URL, известные также как строка запроса [3]. Если значение не установлено, то происходит проверка массива `$_REQUEST` [3].

В случае, если значения `$_REQUEST` определены, происходит получение и декодирование уведомления. Если это уведомление кода ошибки и требуется подтверждение адреса, то необходимо отправить строку для подтверждения и закончить выполнение скрипта. Если же уведомление содержит информацию о новом сообщении, при помощи методов API «ВКонтакте» мы можем отправить ответное сообщение пользователю, проверив таким образом работу Callback API.

И во всех случаях без исключения возвращаем серверу Callback API ответ «ok», как того требуют правила работы с API «ВКонтакте» [1]. После этого делаем запись входящих сообщений в обычный текстовый файл, также хранящийся на сервере. Название файла задается в виде `YYYYMMDD.txt` (рис. 2), что в дальнейшем будет использовано в сортировке, необходимой для удобного размещения информации о событиях на странице.

Рис. 2. Файлы входящих сообщений

Если задано значение «view» массива `$_GET`, то на странице идет отображение прошедших событий.

Сначала получаем список файлов и каталогов, расположенных в папке «stats» и сортируем его в обратном порядке так, чтобы последнее произошедшее событие было первым в списке. Далее в цикле осуществляется перебор имеющихся файлов со списком событий, создание текстовых массивов на основе их содержимого и последующее инвертирование (т.к. удобнее для просмотра располагать новые события выше старых).

Также на сервере хранится файл `events.json`, представляющий из себя перечисление типов событий и их значений на русском языке. Благодаря этому типы событий на странице отображаются на русском языке.

2.2. Файл `bot.php`

Логика скрипта, содержащегося в файле `bot.php`, также разбивается на несколько различных ситуаций в зависимости от параметров строки запроса.

Перед подключением к LongPoll-серверу необходимо получить данные сессии («server», «key», «ts») методом `messages.getLongPollServer`. Чтобы начать получать ответы от сервера, необходимо получить три обязательных параметра, необходимых для работы LongPoll: «server», «key» и «ts»:

- «key» – секретный ключ сессии,
- «server» – адрес сервера,

- «ts» – номер последнего события, начиная с которого нужно получать данные.

Полученные параметры хранятся в файле *data.json* и периодически обновляются, т.к. ключ сессии и адрес сервера меняются. Также для работы с LongPoll понадобились ключ доступа управления сообществом и id группы.

Итак, рассматривается несколько случаев.

1. Получен новый секретный ключ сессии («newkey»).
 - а. Перезаписываем параметр «key» в файле *data.json*. Сообщаем, что ключ доступа получен, и отображаем на странице гиперссылку на адрес <https://vk.n-clave.ru/bot.php> с текстом «Проверка новых событий».
 - б. Если не удалось сформировать файл с ключом доступа, сообщаем об этом и предлагаем попробовать ещё раз (гиперссылка <https://vk.n-clave.ru/bot.php?newkey> с текстом «Попробовать ещё раз»), при этом проверка новых событий по-прежнему остается доступной.
2. В URL-строке задан параметр «settings».

В этом случае ответ, полученный с сервера, отображается в виде ассоциативного массива на странице, что позволяет посмотреть текущие настройки LongPoll для группы.
3. Получен номер последнего события («ts»).
 - а. Проверяем, имеется ли на нашем сервере файл с ключом доступа. Если нет, выводим сообщение, что файл не найден, и прерываем выполнение скрипта.
 - б. Перезаписываем номер последнего события в файл. В результате отображаем сообщение об успешности или неудачи установки номера.
4. В остальных случаях:
 - а. В ответ на запрос сервер может вернуть одну из ошибок, в зависимости от значения которой отображается соответствующая информация и гиперссылка:
 - «failed»:1 – история событий устарела или была частично утеряна. Бот может получать события далее, используя новое значение «ts» из ответа.
 - «failed»:2 – истекло время действия ключа. Нужно получить новый ключ доступа, поэтому на странице отображается гиперссылка <https://vk.n-clave.ru/bot.php?newkey> с текстом «Получить».
 - «failed»:3 – информация о пользователе утрачена, нужно запросить новые ключ доступа и точку отсчёта событий.
 - б. Если ошибок не было, то обновляем номер последнего события. Появляется текстовое поле для установки номера последнего события, который можно при желании изменить (рис. 3). В случае, когда номер обновить не удалось, выводим соответствующее сообщение.
 - в. Если в ответе на запрос задан параметр «updates», но их количество равно нулю, то сообщаем, что новых событий нет, иначе отображаем гиперссылку <https://vk.n-clave.ru/bot.php> с текстом «Проверка новых событий».

Рис. 3. Установка номера события

Новые события отображаются в качестве ассоциативного массива (рис. 4).

Рис. 4. Отображение новых событий

2.3. Файл *user.php*

В файле *user.php* производятся действия по удалению публикаций на стене сообщества и происходит задание пользовательских настроек, установка которых и происходит на web-странице <https://vk.n-clave.ru/user.php> (рис. 5).

Рис. 5. Задание пользовательских настроек

Алгоритм состоит в следующем. Файлы, хранящиеся на сервере в папке «stats», содержат информацию о прошедших событиях в формате JSON, в том числе сведения о времени публикации. В цикле осуществляется перебор файлов, и время прошедшего события, для которого типом события является «wall_post_new» (новая публикация на стене сообщества), сравнивается с текущим временем. Если текущая дата больше или равна дате, когда событие должно быть откреплено (или удалено), то запись открепляется (или удаляется) со стены сообщества посредством методов API «ВКонтакте» в соответствии с выбранными настройками.

Обработка пользовательских настроек включает в себя рассмотрение нескольких ситуаций. На сервере хранятся файлы с названиями *last.json*, *settings.json* и *app.json*. Файл *last.json* хранит id прошедших событий и их статус – откреплено или удалено, *settings.json* – интервал времени в секундах, через которое должно произойти открепление или удаление публикации, *app.json* – токены и ключи для управления публикациями.

Итак, возможны следующие варианты в зависимости от параметров URL-строки:

1. Если в суперглобальном массиве `$_POST` (ассоциативный массив переменных, переданных скрипту через параметры URL) имеется значение токена, то файлы *last.json*, *app.json* удаляются, и выводится сообщение о том, что получен новый токен для управления сообществом. Формируется массив, содержащий по умолчанию текущий токен, если он был не заменен на новый, id группы и ключи безопасности. Созданный массив в виде строки помещается в файл *app.json*.
2. Если массив `$_POST` содержит параметры «unpin» и «delete», то в файле *settings.json* происходит перезапись текущих настроек.

Для того, чтобы иметь возможность управлять событиями, в данном случае, публикациями, в сообществе было создано StandAlone-приложение, которое позволяет получить специальный токен, после чего методы API «ВКонтакте» становятся доступными для использования в боте.

Существует два способа запуска бота. Первый – запуск вручную, открыв страницу в браузере, или посредством таймера, который бы запускал на выполнение скрипт. Второй осуществляется путем создания задания в планировщике сервера, которое в назначенное время будет запускать скрипт <https://vk.n-clave.ru/user.php> (рис. 6).

Рис. 6. Планировщик задач на сервере

Заключение

Таким образом, поставленная задача по созданию программы, которая поддерживает в актуальном состоянии информацию на стене сообщества социальной сети «ВКонтакте» путем удаления публикаций по таймингу, была выполнена. Благодаря предложенному инструментарию API «ВКонтакте», рассмотренному в работе, предоставляется возможность автоматизации взаимодействия со страницей сообщества, что значительно облегчает создание приложений, предназначенных для управления событиями в рамках данной социальной сети.

В свою очередь использование бота, подобного реализованному, значительно облегчает администрирование сообществ, т.к. избавляет пользователя от необходимости контроля расписания и удаления публикаций вручную.

ЛИТЕРАТУРА

1. Документация API «ВКонтакте» [Электронный ресурс] // VK.COM: социальная сеть «ВКонтакте». URL: <https://vk.com/dev/manuals>
2. Работа с данными в формате JSON [Электронный ресурс] // INTUIT.RU: Национальный Открытый Университет «ИНТУИТ». URL: <https://intuit.ru/studies/courses/19522/1309/lecture/28910>
3. Руководство по PHP [Электронный ресурс] // PHP.NET: сообщество PHP-программистов. URL: <https://www.php.net/manual/ru/index.php>

DOI: 10.17223/978-5-907442-42-9-2021-40

ПРИМЕНЕНИЕ 2-SAT ЗАДАЧИ ДЛЯ ОБНАРУЖЕНИЯ УТЕЧКИ РЕСУРСОВ В СПИСОЧНЫХ СТРУКТУРАХ

Серебренникова К.А., Андреева В.В.

Томский государственный университет
krisert@mail.ru, avv.21@mail.ru

Введение

Тестирование программного обеспечения (далее – ПО) является неотъемлемой частью жизненного цикла разработки ПО. Верификация ПО является одним из важнейших, и, в то же время, трудозатратных этапов разработки. Ошибки, допущенные на стадии верификации ПО, могут привести к существенным потерям в будущем.

Разработка программного обеспечения на языках Си/C++ дает широкие возможности для управления памятью в процессе работы программы. В частности, хранение и обработка данных в списочных структурах позволяет динамически размещать элементы в памяти по мере их необходимости. Здесь самое главное – не допустить утечку ресурсов, т.е. доступ к данным. Потеря доступа приводит к тому, что становится невозможным освобождение памяти из-под таких элементов, что, в свою очередь, приводит