УДК 004.27
А.М. Шитько, асп. (БГТУ, г. Минск)

ПРОЕКТИРОВАНИЕ МИКРОСЕРВИСНОЙ АРХИТЕКТУРЫ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Микросервисная архитектура – это архитектура, в которой сервисы: небольшие, узко сфокусированные, слабосвязанные и высокосогласованные [1]. Небольшие сервисы – сервисы, которые не могут разрабатываться больше чем одной командой. Обычно одна команда разрабатывает не более 6 сервисов. При это каждый сервис решает одну бизнес-задачу, и его способен понять один человек. Узко сфокусированные сервисы – сервисы, которые решают только одну бизнес-задачу и делают это хорошо. Такие сервис можно отделить от системы, дописав некоторую логику, и использовать как отдельный продукт. Слабосвязанные сервисы – сервисы, изменение которых не требует изменений в других. Высокосогласованные сервисы – сервисы, в которых класс или компонент содержит все нужные методы для решения поставленной задачи только в этом классе или компоненте и больше нигде.

Свойства микросервисной архитектуры.

1. Разбиение через сервисы. Построение системы путем соединения вместе различных компонент. Здесь существуют 2 понятия: библиотека и сервис. В микросервисной архитектуре библиотеки – это компоненты, которые подключаются к программе и вызываются ею в том же процессе, в то время как сервисы – компоненты, выполняемые в отдельном процессе и связывающие друг с другом с помощью REST или RPC.

2. Группировка по бизнес-задачам. Любая микросервисная архитектура должна придерживаться закона Конвея [2], который гласит, что структура вашего приложения повторяет структуру вашей команды, т.е. необходимо выделять команды по бизнес-задачам.

3. Умные сервисы и простые коммуникации. Вся логика находится в сервисах, канал передачи только передает данные, он ничего не знает о бизнес-задаче.

4. Децентрализованное управление данными. Каждый сервис имеет свою и только свою БД.

5. Автоматизация развертывания и мониторинга.

6. Проектирование через отказы. Сервисы должны работать при отказе отдельных сервисов.

Изучив свойства микросервисов и их способы построения можно выделить следующие преимущества: модульность, высокая доступность, разнообразие технологий, независимое развертывание. Но в свою очередь есть ряд недостатков: поддержка конечной согласованности (поддержка работы с отложенными данными, что требует по-

стоянной доступности приложения), сложность операционной поддержки (поддержка непрерывного развертывания, непрерывной интеграции и автоматического мониторинга).

ЛИТЕРАТУРА

1. Microservice Architecture [Электронный ресурс]. – Режим доступа: http://microservices.io/. – Дата доступа: 30.01.2017.

2. Ньюмен, С. Создание микросервисов. – СПб.: Питер, 2016. – 304 с.

УДК 003.26

Е. А. Блинова, ст. преп.; И.Г. Сухорукова, ст. преп.
(БГТУ, г. Минск)

СРАВНИТЕЛЬНАЯ ОЦЕНКА ПРИМЕНИМОСТИ СТЕГАНОГРАФИЧЕСКИХ МЕТОДОВ В ГРАФИЧЕСКИХ ФАЙЛАХ SVG

В докладе рассматриваются возможности применения стеганографических методов для файлов SVG (Scalable Vector Graphics) – векторных графических файлов, предназначенных для описания двумерной векторной и смешанной векторной и растровой графики в формате XML. Преимущества данного формата, такие как небольшой размер файлов, масштабируемость, интеграция с HTML документами, возможность встраивания растровой графики, возможность редактирования в текстовых редакторах и поддержка в большинстве современных браузеров делают SVG файлы удобным контейнером для осаждения скрытого сообщения в процессе прямого стеганографического преобразования.

Поскольку SVG файлы являются подмножеством файлов формата XML, то к ним могут быть применены классические методы текстовой стеганографии, такие как метод конечных пробелов и табуляций, а также методы, характерные для файлов разметки, такие как метод замены регистра тегов и метод перестановки атрибутов. Однако особенности формата позволяют использовать и другие методы внедрения скрытой информации. Формат тегов описания путей позволяет размещать скрытую информацию в добавлении дополнительных элементов в геометрических фигурах. При описании фигур используется цветовая модель RGB, что позволяет внедрять скрытую информацию в незначительном изменении параметров цвета. Игнорирование браузерами неверных атрибутов позволяет производить подмену атрибутов по заранее определенному алгоритму.

Комбинированное применение нескольких стеганографических