Разработка блокчейн-хранилища данных для приложения Learning Switch сетевого контроллера RUNOS

Шибаев Павел Павлович

Студент

Факультет ВМК МГУ имени М. В. Ломоносова, Москва, Россия

E-mail: pshibaev@lvk.cs.msu.ru

Научный руководитель *— Писковский Виктор Олегович*

RUNOS [1] — это распределенный контроллер SDN/OpenFlow с открытым исходным кодом для программно–определяемых сетей (Sotware Defined Networks, SDN), или сокращенно SDN–контроллер. Возможности RUNOS можно расширять с помощью написания приложений. В официальной документации [2] RUNOS представлен пример приложения Layer 2 Learning Switch (обучающийся коммутатор второго уровня), который позволяет наладить автоматический обмен кадрами между устройствами в сети, идентифицируя интерфейсы по их MAC–адресам, запоминая, где находится каждый из них. Данные хранятся в виде набора кортежей вида $(dp\_id, ethaddr, in\_port)$, где dp_id — уникальный идентификатор свитча OpenFlow, $ethaddr$ — MAC–адрес, in_port — входящий порт (т.е. порт, на который получен пакет). В примере из документации [2] эти данные хранятся напрямую в оперативной памяти в виде отображения $(dp\_id, ethaddr) \to in\_port$. Однако такой способ хранения не надежен, т.к. при перезапуске приложения произойдет потеря данных. Эффективным решением этой проблемы является NoSQL база данных типа «ключ–значение», например, Redis. Однако подобные базы данных не защищены от подмены данных, достаточно сложно проводить аудит их безопасности. Решением этой проблемы может быть распределенное приватное хранилище данных на основе технологии блокчейн.

Для того, чтобы показать жизнеспособность предполагаемого решения, разработано блокчейн–хранилище на основе Cosmos SDK [3]. Cosmos SDK — набор инструментов и библиотека на языке Go, которая позволяет разработчикам быстро создавать блокчейны под конкретные практические задачи. Cosmos представляет собой экосистему совместимых блокчейнов, что позволяет осуществлять передачу данных между блокчейнами (IBC, inter–blockchain communication). Фундаментально, все блокчейн–сети в экосистеме Cosmos оперируют на базе алгоритма консенсуса Tendermint Core [4]. Модульная

природа Cosmos SDK позволяет разработчикам фокусироваться на разработке логики блокчейн–приложения, а не его базовых компонентах по типу алгоритма консенсуса.

Архитектура экспериментального стенда представлена на Рис. 1. Блокчейн–приложение развернуто в 4 экземплярах на отдельных машинах. Также присутствует экземпляр сетевого контроллера RUNOS, на котором запущено приложение L2 Learning Switch, написанное на языке C++. Для того, чтобы обеспечить совместимость приложений, на языке Go написан REST API сервер, к которому обращается L2 Learning Switch. В свою очередь, сервер направляет запросы к gRPC–узлу блокчейна. В блокчейне хранятся кортежи вида $(dp\_id, ethaddr, in\_port)$. Произведены тестовые запуски, которые показали совместимость и корректную работу компонентов стенда.

Код разработанных приложений доступен по ссылке [5]. В качестве дальнейших шагов предусмотрены нагрузочные испытания с целью протестировать производительность предлагаемого решения. Также особенный интерес представляет разработка математической модели, которая позволила бы определять оптимальные параметры конфигурации блокчейн–приложения (размер блока, максимальное количество транзакций в очереди узла, размер кеша блокчейн–узла).

Иллюстрации

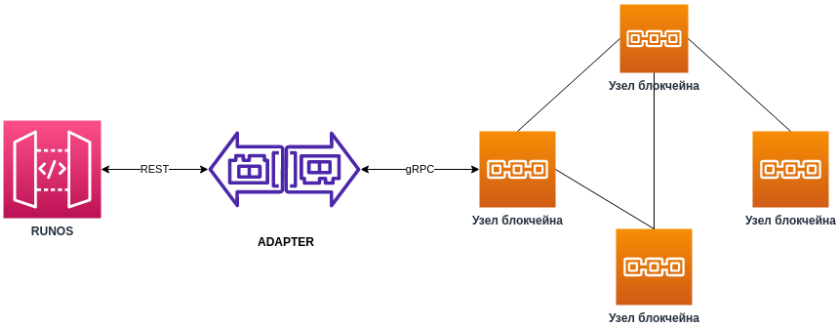

Рис.1 Архитектура экспериментального стенда

Литература

1. Официальный сайт проекта RUNOS: `https://runsdn.com/index.php/products/runos`

2. Пример из документации с L2 Learning Switch: https://arccn.github.io/runos/docs-2.0/eng/31_tutorial_01.html
3. Официальный сайт проекта Cosmos SDK: https://docs.cosmos.network/
4. Kwon J. Tendermint: Consensus without Mining https://www.semanticscholar.org/paper/Tendermint-\%3A-Consensus-without-Mining-Kwon/df62a45f50aac8890453b6991ea115e996c1646e
5. Репозиторий с исходными кодами приложений: https://github.com/shibaeff/runos_chain