Е. А. Романович

МОДИФИКАЦИИ АЛГОРИТМА RAFT В РАСПРЕДЕЛЕННЫХ СИСТЕМАХ

Рассматривается алгоритм нахождения консенсуса в распределенных системах (Raft), а также его возможные способы улучшения для применения в реальных распределенных системах.

Введение

Алгоритм Raft – это алгоритм достижения консенсуса в распределенных системах, является результатом научной работы Диего Онгаро (Ph.D. студента Стэндфордского института), целью которой было построение легкого для понимания алгоритма: алгоритм должен не просто работать, но должно быть очевидно, как именно он работает.

В данной работе основное внимание уделяется репликации самих событий и тесно связанному механизму выбора лидера или определения кворума. С другой стороны, пренебрегают важными окружающими темами, такими как действия по обслуживанию кластера или «дела об окружающей среде», которые в реальном мире являются не менее важными компонентами при создании полного решения для высокодоступной системы.

I. Модификации алгоритма

Возможны и желательны три модификации алгоритма, которые должны быть реализованы в реальных распределенных системах:

1. Введение универсального уникального идентификатора сервера: чтобы определить, является ли слепок данных или журнал на одном сервере каким-либо состоянием конечного автомата того же конечного автомата на других серверах кластера, или является ли это снимком какого-то совершенно другого состояния машины, которая не имеет ничего общего с этим кластером.

2. Алгоритм преждевременного голосования.

3. Решение проблемы конкурирующих лидеров: основываясь на алгоритме преждевременного голосования, модифицируется Raft, чтобы отклонять сервера, которые пытаются выбрать себя в качестве лидера, если текущий лидер здоров для остальной части кластера.

В работе [1] была представлена идея алгоритма преждевременного голосования, без подробного объяснения деталей такого алгоритма.

Полезность алгоритма преждевременного голосования заключается в решении проблемы гонки выборов за лидерство при голосовании. Алгоритм Raft обладает сильным свойством, которое заставляет его участников всегда принимать самый большой term, который они получили от другого сервера. Это свойство является ключом к надежности алгоритма: из-за этого выборы становятся детерминированными, и от этого также зависит свойство сопоставления журнала. Недостаток в реалиях реального мира состоит в том, что это легко приводит к ненужной «переизбытку терма». Предполагая, что сервера будут использовать 64-битную целочисленную систему, вряд ли у них закончатся числа в течение срока службы кластера, но кластера действительно будут подвержены поведению, при котором неисправный сервер будет начинать выборы при присоединении к кластеру, даже если остальные сервера были здоровы и продолжают иметь полноценного лидера.

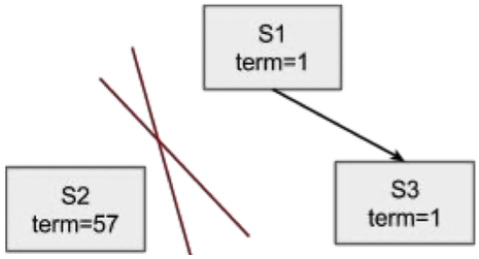

Рис. 1 – Схема Raft кластера состоящая из 3 серверов. Сервер 2 временно отключен от остальных.

1) Raft заставляет отключенный сервер начать выборы один раз в каждом тайм-ауте, что приводит к увеличению текущего term. Поскольку он не может подключиться к другим серверам, он потеряет свой выбор и производит повторную попытку.

2) Как только он сможет восстановить соединение с остальной частью кластера, его более высокий term будет распространяться на S1 и S3. Это приведет к тому, что S3 прекратит принимать новые записи журнала от S1, и заставит S1 уйти в отставку в качестве лидера.

3) Новые выборы на term = 58 в конечном счете инициированы и будут выиграны тем сервером, который успеет первым.

Решение состоит в том, чтобы ввести алгоритм предварительного голосования, который

выполняется перед переходом в статус кандидата. Сервер может переключиться на кандидата только в том случае, если превентивное задание выполнено успешно, в противном случае он должен дождаться следующего тайм-аута выборов.

Реализация алгоритма предварительного голосования проста: получатели RPC PreVote должны отвечать с тем же результатом, что и, если бы это был фактический голос. Тем не менее, важно подчеркнуть, что предварительный ответ не является обязательным для сервера. Хотя каждый сервер должен голосовать только за одного кандидата на определенный срок, это не относится к этапу предварительного голосования. Несколько кандидатов могут получить положительные показания на этапе предварительного голосования, однако, как только они приступят к реальным выборам, только один из них получит реальное голос. Такое поведение является ключом к тому, чтобы избежать гоночных условий, которые могут привести к неубедительным выборам. Например, сервер может преуспеть в получении потенциального большинства голосов на этапе предварительного голосования, но затем сам отключится, прежде чем сможет приступить к фактическим выборам. В этом случае было бы не целесообразно тратить драгоценное времени на отказ при голосовании за другого кандидата, который все еще может победить на выборах.

1. Ongaro Diego; Ousterhout John "In Search of an Understandable Consensus Algorithm". –2014
2. Ongaro, Diego. "Consensus: Bridging Theory and Practice". –2014
3. Brian M. Oki; Barbara H. Liskov "A New Primary Copy Method to Support HighlyAvailable Distributed Systems". –1998

Романович Евгений Александрович, магистрант кафедры информационных технологий автоматизированных систем БГУИР, yauheni.ramanovich@gmail.com.

Научный руководитель: Матвеенко Владимир Владимирович, доцент, кандидат физ.-мат. наук vladzimir66@bsuir.by.