Зборник радова Факултета техничких наука Нови Сад

УДК: 4.9
ДОИ: https://doi.org/10.24867/33BE20Kuzmanovic

КОМПАРАТИВНА АНАЛИЗА ОСНОВНИХ КАРАКТЕРИСТИКА И ПЕРФОРМАНСИ БРОКЕРА ПОРУКА *NATS*, *RABBITMQ* И *APACHE ROCKETMQ*

A COMPARATIVE ANALYSES OF THE MAIN CHARACTERISTICS AND PERFORMANCE OF NATS, RABBITMQ AND APACHE ROCKETMQ MESSAGE BROKERS

Огњен Кузмановић, *Факултет техничких наука, Нови Сад*

Област - ЕЛЕКТРОТЕХНИКА И РАЧУНАРСТВО

Кратак садржај – *У овом раду дато је поређење основних карактеристика и перформанси брокера порука NATS, RabbitMQ и Apache RocketMQ. Анализа перформанси урађена је кроз мерење кашњења порука и протока порука. За сваки брокер порука извршен је скуп тестова на платформи за рачунарство у облаку Azure. Конфигурација сваког теста се добила комбиновањем неколико различитих величина и броја порука. Зарад објективне компарације перформанси, осмишљен је и направљен посебан алат у програмског језику Python.*

Кључне речи: *Брокер порука, поређење перформанси, NATS, RabbitMQ, Apache RocketMQ.*

Abstract - *This paper compares the basic characteristics and performance of NATS, RabbitMQ and Apache RocketMQ message brokers. Performance is measured through message lattency and message throughput. Tests were conducted on the Azure cloud platform. The configuration of each test was obtained by combining several different message sizes and numbers. To ensure objective comparison, a custom Python-based tool was developed.*

Keywords: *Message broker, performance comparison, NATS, RabbitMQ, Apache RocketMQ.*

1. Увод

Временом, услед повећаног броја корисника интернета, расла је и количина података који су сервиси, који опслужују кориснике, морали да обраде. Стога, поједини дистрибуирани системи постојали су све комплекснији са све више повезаних компоненти које се прикључују и искључују из система са географски различитих локација, а све то ради опслуживања све више надолазећих корисника са различитих локација.

У таквим системима, уско повезани ентитети који међусобно комуницирају, у комбинацији са факторима попут непоузданости мреже и хетерегености различитих апликација могу да наштете трајности и поузданости система [1]. Из ових разлога, ексклузивно коришћење само синхроног модела комуникације „захтев/одговор“ не би пратило динамичну природу и потребе поменутих дистрибуираних система [2]. Како би се надоместиле мане синхроног модела комуникације, на значају је добио асинхрон модел комуникације који нуди већи ниво флексибилности, и то модел размене порука „објава/претплата“ са брокером порука као посредником у опслуживању порука јер не захтева да процеси који су учесници у размени порука буду континуирано активни.

Током претходног периода, настао је велики број различитих система за размену порука попут *Apache Kafka*, *ActiveMQ*, *Apache Pulsar*, *NSQ* итд. За предмет овог рада одлучено је да буду обрађене следеће имплементације: *NATS*, *RabbitMQ* и *Apache RocketMQ*. Поменути брокери порука истичу се тиме да су јавно доступни (енгл. *open source*), популарни су у инжењерско-академској заједници, нуде диверзитет различитих особина, а такође аутор није пронашао ниједан рад који је акценат ставио на баш њих.

Поред теоријске обраде поменутих брокера порука, циљ овог рада било је и мерење перформанси брокера порука на платформи за рачунарство у облаку *Azure*, као и крајње извођење закључака о томе у којим ситуацијама би требало користити који брокер порука.

2. *NATS*

Према [3], *NATS* представља инфраструктурну компоненту модерних дистрибуираних система која омогућава размену података између апликација и сервиса у виду порука. *NATS* услуге су омогућене од стране више *NATS* серверских процеса који су међусобно повезани и тако креирају *NATS* сервисну инфраструктуру. Кроз апликативни код дефинишу се клијенти који се повезују на *NATS* серверске процесе и њима објављују поруке или од њих примају поруке на које су претплаћени.

НАПОМЕНА:
Овај рад проистекао је из мастер рада чији је ментор био др Владимир Димитриески, ванредни професор.

Основни скуп функционалности које *NATS* нуди назива се *Core NATS*. На ову, базичну, *NATS* компоненту може се гледати као на једноставан систем за размену порука „објава/претплата“. Прималац порука ће поруке примати само ако су и он и генератор порука повезани на *NATS* у истом моменту. Основни скуп функционалности може се проширити кроз *JetStream* компоненту која нуди перзистентни дистрибуирани систем који омогућава олабављењију спрегу између генератора и конзумента порука, као и додатан скуп функционалности попут репликација порука, складишта кључ/вредност итд.

3. *RabbitMQ*

RabbitMQ је популаран брокер порука отвореног кода, написан у програмском језику *Erlang*, јавно доступан од 2007. године. Имплементира неколико различитх протокола међу којима су *AMQP 0-9-1* и *MQTT*.

Архитектура *RabbitMQ* брокера порука изгледа тако да се с једне стране налазе генератори порука, а са друге примаоци порука. Генератор поруке креира поруку којој придодаје кључ за трасирање (енгл. *routing key*) који шаље ка брокеру, а брокер прима поруку кроз компоненту за размену порука (енгл. *exchange*), односно размењивач порука. За размењивач порука потенцијално може бити везано много редова чекања уз помоћ различитих техника увезивања које зависе од конкретног типа размењивача поруке.

Табела 1. Табеларна компарација неперформатних категорија посматраних брокера порука

Особина	***NATS***	***RabbitMQ***	***RocketMQ***
Година настајања	2011	2007	2012
Језик	*Go*	*Erlang*	*Java*
Гаранција доставе	Најмање један/највише један/тачно један	Најмање један/ највише један	Најмање један
Гаранција редоследа	ФИФО редослед по генератору порука	Редослед у реду чекања, ФИФО редослед	Редослед у реду чекања, ФИФО редослед
Доступност	Висока	Висока	Висока
Трансакције	Не	Да	Да
Скалабилност	Висока	Слаба	Добра
Протоколи	*NATS* протокол	*HTTP*, *MQTT*, *AMQP*, *STOMP*	*MQTT*
Компаније [5] [6]	*Nutanix, MasterCard*	*Reddit*, *CircleCI*	*Alibaba*, *ByteDance*

Размењивач порука има задатак да дистрибуира копије порука ка редовима чекања у зависности његовог типа, од тога како су редови чекања увезани са њим, као и кључа за трасирање који је придодат порукама које се трасирају. С друге стране, примаоци порука претплаћени су на редове чекања и конзумирају поруке у њима.

4. *Apache RocketMQ*

RocketMQ је брокер порука креиран од стране компаније *Alibaba* како би решили проблеме перформанси које су имали са *ActiveMQ* брокером порука, а који су дошли до изражаја тек онда када је знатно порасла потражња за услугама ове компаније. Према [4], *RocketMQ* добио је на популарности услед његове једноставне архитектуре, великој лепези функционалности, као и изузетној скалабилности. Аутори овог брокера порука сматрају да је он постао индустријски стандард када је реч о порукама у којима се налазе финансијски подаци за које је неопходна изузетна доза сигурности преноса и обраде.

5. Поређење неперформантних особина

У табели 1, дато је поређење најзначајних неперформантних особина брокера порука које могу имати утицај на избор конкретног брокера порука за имплементацију у неки софтверски систем.

6. Тестирање перформанси брокера порука

За потребе компаративне анализе перформанси брокера порука упоређене су две перформантне категорије: кашњење порука (енгл. *latency*), као и проток порука (енгл. *throughput*). Како би се по ове две категорије добиле конкретне и упоредиве метрике, сваки брокер порука био је подвргнут серији тестова на пларформи *Azure*. Тестови су извршавани употребом алата посебно направљеног само за потребе овог рада.

6.1. Хардверско-софтверско окружење

За потребе тестирања перформанси брокера порука, коришћене су виртуелне машине *Standard D4ds v4* које нуди платформа *Azure* у склопу *Azure Virtual Machines* сервиса. Овај тип виртуелних машина нуди следећу спецификацију: 4 vCPUs / 16GiB RAM / 30 GiB HDD / 6400 IOPS / Ubuntu 22.04. Као физичка локација машина, изабран је регион северне Европе, у зони 3.

Зарад олакшаног покретања изолованих окружења у којима су се брокери порука извршавали, на машину су били инсталирани алати *Docker v27.3.1* и *Docker Compose v2.29.6*. *Docker* контејнери нису били условљени ограничењима ресурса.

Тестирање се вршило на једној виртуелној машини како би се обезбедило добијање фер резултата на које не утиче кашњење мреже.

6.2. Параметри тестирања

Као предмети тестирања, у обзир су се узели најзначајније перформантне метрике у свету брокера порука [7], а то су кашњење порука, као и проток порука. За сваку од поменутих категорија тестирања, тестирање се вршило за три могуће величине поруке како би се установило да ли и како величина порука утиче на перформансе брокера порука. Величине порука:

1. Мала порука – 1KB
2. Средња порука – 100KB
3. Велика порука – 1MB

Свака порука је садржала низ унапред насумично генерисаних карактера (*ASCII* карактери, цифре, као и знакови интерпункције).

Као додатан параметар тестирања употребљен је број порука како би се установило да ли постоје промене у перфомансама за различите редове величина броја порука. За сваку величину порука, три различите количине порука су биле тестиране:

1. Мали број порука - 1.000
2. Средњи број порука - 10.000
3. Велики број порука - 100.000

За сваку комбинацију поменутих параметера тестирања, извршено је најмање три теста. У случају да се међу извршена три теста нашао резултат који значајно одступа од осталих резултата, извршено је још три теста. Као крајњи резултат узимала се просечна вредност три најбоља резултата.

6.3. Спецификација брокера порука

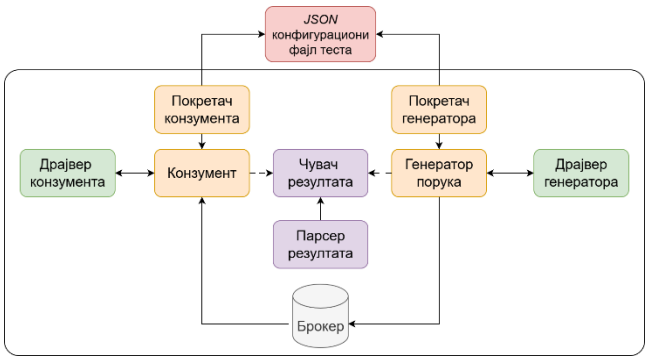
У наставку су дате тачне верзије коришћених брокера порука:

- *NATS* v2.10.14
- *RabbitMQ* v3.13.2
- *Apache RocketMQ* v5.2.0

Код свих сценарија тестирања генератор порука, брокер и прималац порука су били заједно на једној машини. Модел комуникације који је био коришћен је „објава/претплата“.

6.4. Начин тестирања и бележења резултата

За потребе свих описаних тестирања и зарад максимално објективних резултата, креиран је посебан алат у програмском језику *Python*. Први корак у раду са алатом јесте конфигурисање параметара теста кроз *JSON* конфигурациони фајл. Кроз овај фајл, корисник има опцију да подеси жељени брокер порука, величину и број порука. Корисник задаје команду за покретање конзумента порука који се повезује са брокером порука и креће да ослушкује поруке. У том тренутку, конзумент у терминалу исписује поруку да је спреман и корисник може приступити покретању генератора порука. На основу параметара из конфигурационог фајла који се односе на тип брокера порука, број и величину порука, генератор порука се инстацира и креће да генерише поруке. Конзумент порука их прима и обрађује. Оба процеса задужена су за вођење бриге о прикупљању метрика током извршавања теста, а потом и њихово чување. Након извршења теста, потребно је покренути парсер добијених метрика ради извлачења информација везаних за мерене категорије. Визуелни приказ архитектуре описаног алата дат је на слици 1. Код је доступан на платформи *GitHub* и може му се приступити путем [8].



Слика 1. Визуелни приказ архитектуре алата за извршавање тестова

7. Резултати

У наставку су дати резултати брокера порука у контексту кашњења и протока порука.

7.1. Кашњење

У табели 2, приказани су брокери порука сортирани по резултатима које су остварили у сваком тесту.

Табела 2. Табеларни приказ резултата кашњења порука брокера порука

	Мала порука			Средња порука			Велика порука		
Број порука	1к	10к	100к	1к	10к	100к	1к	10к	100к
RabbitMQ	1	1	1	2	3	3	3	3	2
NATS	3	3	2	3	2	2	2	2	1
RocketMQ	2	2	3	1	1	1	1	1	3

Оно што се може закључити евалуацијом резултата мерења кашњења порука јесте, да је целокупно гледано, *Apache RocketMQ* показао најбоље резултате са убедљиво најмањим кашњењем код средње и велике величине величине порука за све количине порука (уз недостатак резултата са велику величину порука и обим од 100.000 порука), док је за мале величине порука показао минимално лошије резултате од најбољег *RabbitMQ* брокера порука. Разлог за веома ниско кашњења се првенствено огледа у томе што су аутори *Apache RocketMQ* брокера уложили велике напоре да направе што више оптимизација у извршавању процеса. Према [9] [10], знатна унапређења су направљена на пољу оптимизације извршавање процеса унутар *Java* виртуелне машине, као и на пољу смањења кашњења у закључавању ресурса (енгл. *lock latency*) узевши у обзир да је закључавање ресурса једна од основних потреба у вишенитним окружењима. *RabbitMQ* је показао најбоље резултате код мале величине порука, док је код средње и велике величине порука показао веома лоше резултате. Разлог за његове лошије резултате може се тражити у томе да је *RabbitMQ* фокусиран на процесирање порука у групама (енгл. *batches*) како би се побољшао проток на уштрб кашњења. *NATS* брокер порука, иако није показао најбоље резултате ни у једном тесту, био је конзистентан у резултатима.

7.2. Проток порука

У табели 3, приказани су брокери порука сортирани по резултатима које су остварили у сваком тесту мерења протока слања и конзумирања порука. Резултати су код оба предмета мерења били идентични.

Табела 3. Табеларни приказ резултата протока слања и конзумирања порука

	Мала порука			Средња порука			Велика порука		
Број порука	1к	10к	100к	1к	10к	100к	1к	10к	100к
RabbitMQ	2	2	2	1	1	1	1	1	1
NATS	1	1	1	2	2	2	2	2	2
RocketMQ	3	3	3	3	3	3	3	3	3

Евалуацијом резултата мерења протока порука се може закључити да је *Apache RocketMQ* показао убедљиво најгоре резултате у оба посматрана случаја протока порука. Објашњење за овакво понашање би се могло пронаћи у томе што *RocketMQ* чува сваку поруку на диск, док то није предефинисано понашање код остала два брокера порука. *NATS* је приказао изузетно добре резулате са врло високим протоком порука код мале величине порука. Са друге стране, *RabbitMQ* приказао је конзистентне и најбоље резултате код свих мерења за средњу и велику величину порука што је и логично узевши у обзир његову усмереност на обраду у групама.

8. Закључак

У овом раду, извршено је поређење *NATS*, *RabbitMQ* и *Apache RocketMQ* брокера порука кроз њихове перформантне и неперформантне категорије. У оквиру перформатних категорија, акценат је стављен на кашњење и проток порука, као две најзначајније категорије код брокера порука. Како би се тестирало што више варијација реалних сценарија коришћења самих брокера, сваки брокер порука тестиран је кроз два параметра тестирања – величина и број порука. Зарад добијања што објективнијих резултати, креиран је посебан алат у програмском језику *Python* који омогућава инстанцирање примаоца и генератора порука на основу конфигурационих параметара теста дефинисаних кроз *JSON* конфигурациони фајл, бележење метрика, њихово чување, као и парсирању у смислене резултате који се могу записати и тумачити. Сви чиниоци алата били су покренути на виртуелној машини *Azure* платформе на којој су тестови и извршавани. Оно што се може закључити из крајње анализе резултата мерења јесте да ниједан од посматраних брокера порука није идеалан за све сценарије употребе. Приликом одабира коришћења једног од посматраних брокера порука, препорука аутора јесте да се првенствено сагледа перформантна категорија која је битнија за систем који се развија, као и очекивана величина порука. На основу ова два параметра може се направити следећа матрица одлука:

1. Битност – минимално кашњење поруке
    a. Мала величина порука – *RabbitMQ*
    b. Средња и велика величина порука – *Apache RocketMQ*
2. Битност – максималан проток порука
    a. Мала величина порука – *NATS*
    b. Средња и велика величина порука – *RabbitMQ*

У овом раду, акценат је стављен на извршавање на једној машини како би се искључио утицај кашњења мреже на перформансе. Стога би будући правац развоја могао бити мерење перформанси брокера порука у дистрибуираном окружењу.

9. Литература

[1] L. Magnoni, "Modern Messaging For Distributed Systems," 2015.

[2] K. S. E. Philippe Dobbelaere, "Kafka versus RabbitMQ," 2017.

[3] "Oficijalna veb stranica NATS dokumentacije," [Online]. Available: https://docs.nats.io/nats-concepts/overview.

[4] "Zvanična dokumentacija Apache RocketMQ brokera poruka," [Online]. Available: https://rocketmq.apache.org/docs/. [Accessed 03 2024].

[5] S. Raje, "Performance Comparison of Message Queue Methods," 2019.

[6] "HG Insights," [Online]. Available: https://discovery.hgdata.com/. [Accessed 05 2024].

[7] "Benchmarking Apache Pulsar, Kafka, and RabbitMQ," 21 08 2020. [Online]. Available: https://www.confluent.io/blog/kafka-fastest-messaging-system/. [Accessed 05 2024].

[8] O. Kuzmanovic, "Programski kod korišćen za izvšavanje testova".

[9] Y. Z. A. G. Y. Guo Fu, "A Fair Comparison of Message Queuing Systems," *IEEE Access,* 2020.

[10] A. Shi, "DZone," [Online]. Available: https://dzone.com/articles/apache-rocketmq-how-did-we-lowered-latency. [Accessed 10 2024].

Кратка биографија:

Огњен Кузмановић рођен је 1998. године у Новом Саду. Основну школу „Светозар Марковић Тоза” завршио је 2013. године као носилац дипломе „Вук Караџић”. Исте године уписује природно-математички смер у гимназији „Јован Јовановић Змај” коју завршава 2017. године. Потом, уписује Факултет техничких наука, одсек Рачунарство и аутоматика. Све предвиђене испите положио је са просечном оценом 9,68.