DATABASES / OPEN SOURCE / STORAGE

Navigating the Path From Redis to Valkey

By carefully planning your migration to Valkey, you can ensure minimal disruption while enhancing your infrastructure for future growth.

Dec 20th, 2024 8:00am by Martin Visser

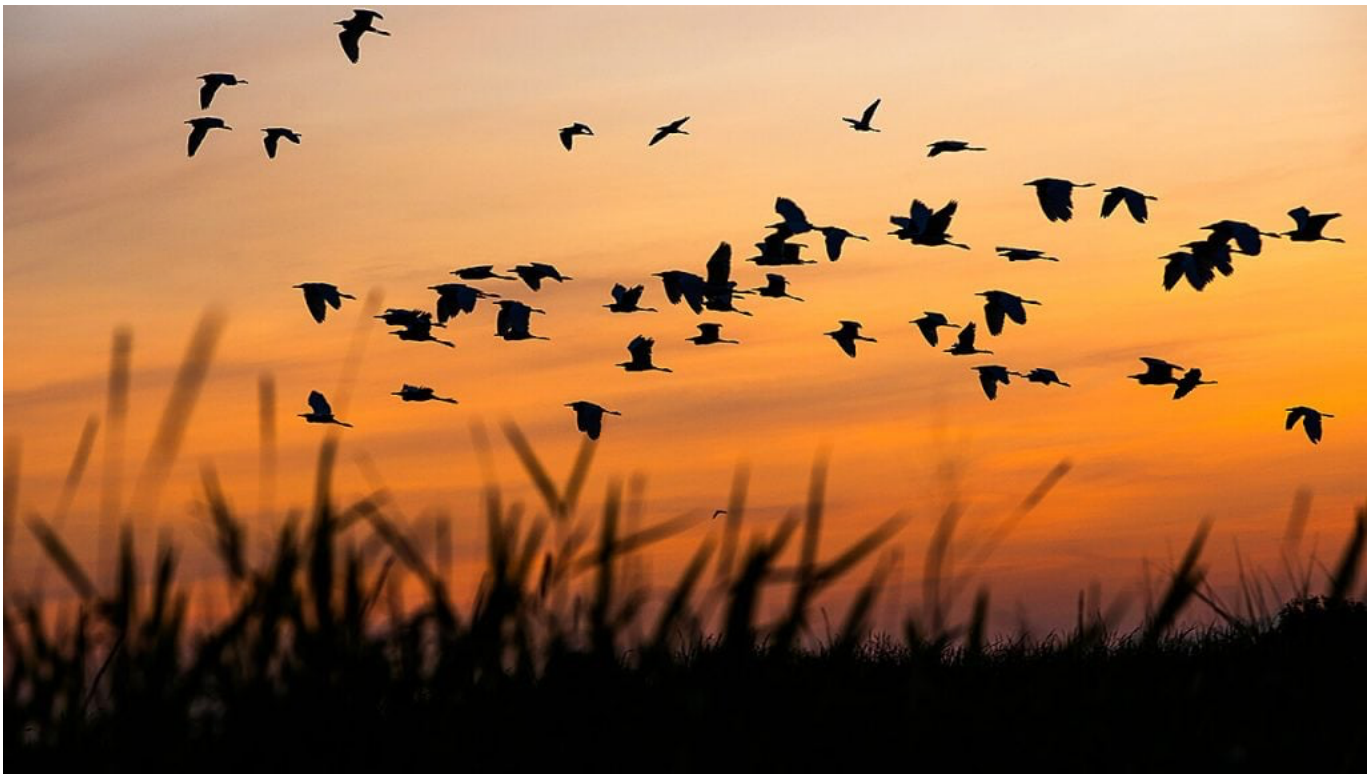

Image from Tonktiti on Shutterstock.

Whatever you think of Redis' decision to change the software license for its in-memory data store, it

certainly had a huge impact on the open source community that exists around Redis. Amid the frustration, Valkey was quickly created as an open source alternative, considering how critical the project is to so many developers.

But this also represents a serious inflection point for organizations that have relied on Redis as their go-to project for in-memory data storage and caching. Should you stick with what you know, and how should you plan a migration if you decide to make that change?

The Emergence of Valkey: A Community-Driven Response

Developers are concerned about this change for Redis because it affects the way they think about a critical component of their applications. Redis is the most commonly deployed key-value store, deployed by 67% of study respondents in a recent survey. The license change has made a lot of developers consider their future approach, as 70% of respondents in our survey with Redis deployments said the shift in Redis' licensing has motivated them to seek alternatives.

Valkey is a reaction from the open source community that did not want to see another project taken away and made into something that they did not agree with, or that members of the community could not use as they saw fit. Rather than putting up with the issue, the community decided to create its own fork from the last open source version of Redis, then continue development to suit users' needs.

TRENDING STORIES

1. Rust Will Explode, SBOMs Will Be Duds: Open Source Predictions
2. What Happens to Relicensed Open Source Projects and Their Forks?
3. Kubernetes, Rust, Linux and DOS? The Year in Open Source
4. Open Source in 2025: Strap In, Disruption Straight Ahead
5. How cert-manager Got to 500 Million Downloads a Month

A collaborative effort by tech industry giants, including Google, AWS and Oracle, Valkey is more than a mere technical replica. It represents a principled stand in the ongoing debate about open source sustainability. By inheriting Redis's core functionality and promising continuous community-driven innovation, Valkey aims to provide a stable, transparent alternative that addresses concerns about vendor lock-in and proprietary constraints. This has already been apparent in the recent new major release of Valkey, where significant improvements were contributed by these tech giants and others.

Where Are Redis and Valkey Used?

Redis is much loved for being simple to use and performant at scale. Valkey continues this approach, providing developers with a rich set of native data structures that simplify complex application logic. These data structures allow developers to implement sophisticated algorithms and data manipulations with minimal code. For instance, building a real-time leaderboard or implementing a robust task queue becomes a matter of simple, elegant commands rather than complex database queries.

Beyond raw speed as a database, Redis and Valkey also make it really easy to implement distributed caching for data.

Both Redis and Valkey help developers to improve performance for their applications by speeding up data access. With sub-millisecond response times

and the ability to scale to millions of operations per second, using an in-memory data store as a database can transform the way applications process and retrieve data.

Beyond raw speed as a database, Redis and Valkey also make it really easy to implement distributed caching for data. By storing frequently accessed data in memory, you create a high-performance layer that reduces backend database load and accelerates application response times.

One of the most powerful features is its module system, which allows developers to extend the database's core functionality without compromising performance. This extensibility is particularly compelling for complex domains like financial technology, where standard database operations often fall short of specialized requirements.

For example, you might want to build a financial trading application that can carry out real-time risk calculation and complex derivative pricing from data coming in. Building this functionality in a custom module can transform the database from a simple key-value store into a powerful computational engine without requiring extensive additional data components.

An in-memory data store like Redis can therefore be an essential element in a developer's toolkit for building high-performance, scalable applications. With so many potential ways to use Redis, making a change to another platform — even one that is based on exactly the same code — can be a worry.

TRENDING STORIES

1. Rust Will Explode, SBOMs Will Be Duds: Open Source Predictions
2. What Happens to Relicensed Open Source Projects and Their Forks?
3. Kubernetes, Rust, Linux and DOS? The Year in Open Source
4. Open Source in 2025: Strap In, Disruption Straight Ahead
5. How cert-manager Got to 500 Million Downloads a Month

Migration Planning: Assessing Your Infrastructure

Making any change in your infrastructure can be a challenge. However, for many developers, being in control and able to use software in the way that best suits your needs is essential and nonnegotiable to them. For others, the costs of moving would be too high compared to providing their own support. For others who have been bitten by the issue of lock-in in the past, being beholden to any technology provider due to a change of license is not something they want to repeat. So, planning your migration effectively is critical.

Before initiating a migration, conduct a comprehensive assessment of your current Redis implementation. This should tell you all the ways your applications currently use Redis, how they are configured and where those instances are located. This may throw up some surprises, like applications that have older and out-of-date

versions running and that you would have to patch in any case. How often have we seen Redis undisturbed beavering away in a data center without any operational love? Even when something is this reliable, that doesn't mean it should be neglected or that you should always stick to your current approach.

However, this process is about more than setting down your migration strategy on paper. It can provide an opportunity to identify and implement significant efficiency and performance improvements across a project too. As part of this process, analyze your application's latency, operations throughput and any patterns in transaction volumes over time. Based on this historical view, you can also consider any future demand levels that you expect to see on your application or service, and then plan accordingly.

You can also evaluate how critical that instance of Redis is within the overall application and how your performance overall depends on that component. Does your website latency go through the roof without that component being active? This represents a potential single point of failure, so you can also check that your uptime and availability needs are being met.

Valkey Deployment: On-Cloud, on-Premises or a Mix?

Any migration path will also heavily depend on what you currently have deployed and where. Typically, your primary consideration is minimizing network latency between your application and your new Valkey data store, as unnecessary delays can

significantly affect performance. Valkey can be deployed as part of on-premises applications if you have Redis deployed there or in the cloud. In short, deploy your new Valkey instance where your application resides.

For self-managed cloud deployments with Redis, there are several options you can take. While Valkey is very new, the fact that multiple cloud services with Valkey support are springing up is a good indication of a healthy ecosystem. You may want to look at running your own Valkey instances in your own data center, use a managed service provider to handle it on your behalf or wait for a "Valkey as a service" option to hit the market. Most providers involved in the Valkey community are cloud providers, but there are others (like where I work, Percona) that focus on on-premises deployments too.

Even when something is this reliable, that doesn't mean it should be neglected or that you should always stick to your current approach.

For companies using self-managed private cloud services, getting advice on running Valkey as well as migration planning assistance can help save you time and reduce the risk of any project overruns. Getting expert advice can pay off greatly wherever you currently host your systems, particularly if you intend to carry on with your own data center or self-managed environment.

Wherever you run your IT — or if you are thinking of changing your provider — successful migrations require planning, anticipating potential challenges including the need to upgrade at the same time. The golden rule for successful migrations is to isolate changes as much as possible so you can execute step by step and roll back on any mistakes or when unforeseen issues crop up.

Similarly, taking a full backup of any machine or image before you start on your migration makes it easier to recover, so keep the pressure off and ensure you can go back on your actions. With some system updates, the ability to roll back is limited, so a backup is your next best option if something doesn't go the way you thought.

Valkey 8 has major improvements that can bring extra performance, efficiency and stability improvements at no cost.

One of the biggest issues in a migration is where you intend to end up. For Redis and Valkey, the choice of destination is currently limited to Valkey versions 7 and 8. Valkey 8 has major improvements that can bring extra performance, efficiency and stability improvements at no cost, so it makes sense to aim for this where possible. However, you may have multiple versions of Redis in place that you intend to move from. There can be some significant differences between the current Redis version and the target Valkey version.

Many applications are still running on Redis 6, and

substantial advancements have happened in Valkey 7 and 8, so you may have to get your Redis install updated first before moving over to Valkey. Redis' modular structure means that your application could rely upon some specific components or integrations, so you should test that these modules work with Valkey. If they aren't currently supported by Valkey, then look into the community involved in maintaining them, and also where you might be able to provide contributions or support.

Most Redis client libraries will work seamlessly with Valkey, but perform a thorough test in advance to confirm compatibility. Minor configuration tweaks might be necessary depending on your specific implementation.

Getting Ready for Valkey Data Migration

Alongside the general approach to planning migrations, there are some specific points you should bear in mind for Valkey migrations. Connection string updates, which connect application components to the database and support transactions, are one of the biggest challenges. When you implement your new database instance, you will have to modify your application configurations to point to those new Valkey endpoints as well.

Alongside this, you may have to update your authentication process so your application can manage user access. As part of this, you can also carry out some spring cleaning and remove any old accounts that no longer need access to the

system.

The chosen migration strategy will highly depend on downtime and application requirements as well as how Redis is deployed. For more critical database instances, you may have a cluster deployed for high availability and keep the system running through a server failure or instance going down. You will probably want to replicate that same approach with your new Valkey instance. However, you may be able to carry out your migration without requiring a lot of planned downtime.

For example, say you have a proxy like Envoy implemented between the rest of your application and your current Redis instance. In these circumstances, you would not have to make any connection string updates to your system, as the proxy handles these connections for you.

You can also use traffic mirroring to duplicate the writes to the new Valkey cluster so you can avoid extra downtime for your users during the move. Once you have implemented the new cluster, you can apply those transactions into your new production system and have an up-to-date instance faster.

Valkey was formed because the community wanted a project that focused on their requirements while sticking to an open source philosophy. Like any major database instance, moving to Valkey is a significant project, so thinking ahead is essential. By carefully planning your migration, you can ensure minimal disruption while at the same time enhancing your infrastructure for future growth.

Percona will be taking part in State of Open Con, a conference covering open source software, open hardware, open source in finance and banking and mobile/telecommunications. The event will be Feb. 4-5 in London. Alex Williams of The New Stack will moderate a track on the future for open source at the event as well. For more information, visit stateofopencon.com.

TNS

Martin Visser is the Valkey technology lead at open source database company Percona, where he is responsible for the company's product and approach around Valkey. Prior to joining Percona, he worked at companies including VMware, Redis and EnterpriseDB on implementing...

Read more from Martin Visser →

Percona is a sponsor of The New Stack. The New Stack is a media sponsor of State of Open Con 25, which contributed this post.

INSIGHTS FROM OUR SPONSORS

Kubeshop™

[Total Test Automation Awareness Announcement](#)
20 December 2024

[KubeCon NA 2024 Takeaways: GitOps, Platform Eng & Scaling CI/CD](#)
5 December 2024

[Support for JUnit Reports in Testkube](#)
24 October 2024

andela

[Andela | Why Should Python Developers Care About Testing](#)
20 December 2024

[Andela | Mastering Progressive Hydration for Enhanced Web Performance](#)
19 December 2024

[Andela | Insights and Trends from Recent Tech Events: What You Need to Know](#)
18 December 2024

CONFLUENT

[Generative AI Meets Data Streaming (Part III) – Scaling AI in Real Time: Data Streaming and Event-Driven Architecture](#)
23 December 2024

[Generative AI Meets Data Streaming (Part II) – Enhancing Generative AI: Adding Context with RAG and VectorDBs](#)
23 December 2024

[Generative AI Meets Data Streaming (Part I) – Data as the Engine: Building the AI Fundamentals](#)
23 December 2024

SUBSCRIBE