Source Code Exploration with Memcached

A beginner's guide to understanding and exploring open-source code

Praveen Raj

Prashanth Raghu

www.bpbonline.com

First published: 2024

Published by BPB Online
WeWork
119 Marylebone Road
London NW1 5PU

UK | UAE | INDIA | SINGAPORE

ISBN 978-93-55518-873

www.bpbonline.com

Dedicated to

My parents ***Sri Rajavardhan*** *and* ***Sharadaraj*** *and*
uncle ***S Muthuraj*** *and* ***Aunt Swati***

I would also like to dedicate the work to my grandmother
Late Smt. Lakshmi M
My sisters ***Dr Poornima***, ***Dr Pratima*** *and* ***Priyanka***
My Spiritual Guru Sadhguru ***Jaggi Vasudeva***
My Mentors ***Ananth Narayan S*** *and* ***Amruta Misra***
My Friends

As always we would like to add that all the credits for all the sources would go to the authors of Memcached, Our sincere gratitude goes out to these individuals for their contributions to the Memcached community

— Praveen Raj

My parents ***Sri S Raghu*** *and* ***S Anuradha***
who have always supported me during all endeavors and
helped shape each of my works.

I would also like to dedicate the work to my grandparents
Late Sri G Sampath *and* ***Late Smt. GS Sriragamma***
and
Late Sri SRS Iyengar *and* ***Smt. Jayalakshmi***

My brother ***Sri Jayanth Raghu*** *and sister in law* ***Smt. Veena Rao*** *and*
my companions in joy ***Siddhanth*** *and* ***Vedanth***

As always we would like to add that all the credits for all the sources would go to the authors and contributors of Memcached, we have only summarized the learnings for the good of all and easy readability.

— Prashanth Raghu

About the Authors

- **Praveen Raj** has been actively involved in the open-source community for the past seven years, primarily contributing through Intel® Security Libraries. His journey in the world of open source has been characterized by a dedicated commitment to the development and enhancement of security-related technologies, and he has contributed to the container runtime community focussed on security such as containerd, ocicrypt, ocicrypt-rs and katacontainers. Currently is working for Intel® Trust Authority. He has completed his master's from the Rashtreeya Vidyalaya College of Engineering, Bangalore, and his Bachelor's of Engineering from PES Institute of Technology Bangalore.

- **Prashanth Raghu** is an open source enthusiast who believes in the transformational capability of Open source software in one's learning journey and enabling him to make solid decisions during architectural designs. He believes in writing books that can be considered documentation of large code bases to enable and accelerate one's learning. He is currently on a startup to create products for children and provide solutions to startups. He is a Carnatic flutist by passion and is actively mentored by Guru Vid Sri S Venkatesh and Karnataka Kalashree Vid Sri HS Venugopal. He has completed his master's from the National University of Singapore and his Bachelor's of Engineering from PES Institute of Technology, Bengaluru.

About the Reviewer

Jayanth Raghu has been working as a principal engineer at Infinera for over 12 years, specializing in network planning and management tools. He possesses a wealth of experience in full-stack development. Jayanth has a strong interest in engaging with customers to address their problems and continuously seeks to explore the latest technological trends.

In his spare time, Jayanth enjoys playing and watching sports, and his free time is often occupied by his children.

Acknowledgements

- I would like to thank my technical guru Prashanth Raghu, for encouraging me to co-author this book, I could never have started to work on this book without him. Special thanks to my sister Dr Poornima and mother Sharadaraj, whose love is an unending wellspring of tenderness, and whose sacrifices have paved the path to my ambitions. This dedication is a token of gratitude for the countless sacrifices they have made, the lessons they have imparted, and the unwavering support they have provided throughout my journey. Special thanks to family members Rajavardhan, S Muthuraj, Swati, Dr Pratima R, Dr Abhilash V, Priyanka R and Jagadeesh H.

  I recognize that my accomplishments and experiences are not solely a result of my efforts but are also gifts from the divine. The grace of God has been the silent partner in my endeavors, the reassuring whisper in moments of doubt, and the gentle hand that has steered me away from adversity.

  With profound respect and boundless love, this book is also dedicated to my guiding light, my spiritual guru Sadhguru Jaggi Vasudeva, whose presence in my life has been an eternal blessing. Many thanks to my other spiritual gurus, Paramahansa Yogananda, Swami Vivekananda and Swami Sivananda.

  I am also thankful for my managers at Intel Uttam Shetty, Mrittika Ganguly, Amruta Misra, Raghu Yelluri, Sudhir Bangalore, Shubha R, Jaswant Jasrotia and colleagues Dinkar Dhawale, Arijit Gosh, Arvind Rawat, Raghavender N, Vartika D and my best friends Krishna Desai, Chandra Kumar, Dattatreya, Sudarshan Prabhu, Sankalp Shettar, Manjunath and Naveen Kumar.

  I would also like to thank my professors Prof. NS Kumar, Prof. Ramamurthy Sreenath, Prof. Chidambara Kalamanji, Prof. Nagegowda KS, Prof. D Krupesha, Prof. Rajkumar Rangaswamy, Prof. Nitin.V.Pujari, Prof. Sridhar Seshadri. Extended thanks to my school teachers Miss Pallavi, Miss Daneshwari, Ravi and Rajkumar, Yoga teacher Dilip.

  I am incredibly grateful to all the amazing people who took time out of their busy schedule to review this book in so much detail. Many thanks to BPB staff who gave us insightful feedback, reviews, encouragement and helpful during the process of authoring this book.

  — Praveen Raj

❍ This book would not have been possible without the infinite support of my Family and the lord almighty Sri Ramar. First and foremost my grandfather Mr. G. Sampath a leading advocate in Bangalore, India who led the baton in the Central Excise and Customs department to become one of the authorities on the subject matter in India. I would like to acknowledge the infinite support of my father Mr. S Raghu also an advocate in the same subject matter. He was the one who pulled me out of extreme rough weather in my life and stood as a the shining lighthouse amidst all storm. His everlasting smile is a great inspiration during all my endeavours. I would like to extend my greatest regards to my mother Mrs. S Anuradha who has taken all the pain in life to support me during all my endeavours. It would be incomplete if I did not mention the role of my brother Jayanth Raghu who stood like a ladder for me to climb to the top guiding me and supporting me at every step. I still remember the first day of the undergraduation where he remarked "ಪಚ್ಚ python ಕಲಿತರೆ Google ನಲ್ಲಿ ಕೆಲಸ ಸಿಗುತ್ತೆ. ಕಣೋ. ಚೆನ್ನಾಗಿ ಕಲಿಥಿಕೋ" (Brother, learn python well it is what will help you get into Google. (Original in Kannada)). My aunt Smt. Radha Sreekanth and uncle Sri. Sreekanth and Sri. Lakshmi Narasimhan, Smt. Jayashree, Sri. Raghava Simhan and Smt. Sumana Raghava Simhan. I would also like to thank Gangadhar Gauribidanur who was more than a family member to us, always willing to help and ever entertaining us through delightful stories right from our childhood. I would also like to thank my respected professors D. A. Srinivas, Prof. NS Kumar, Prof. Ramamurthy Sreenath, Prof. Chidambara Kalamanji, Prof. Nagegowda KS, Prof. D Krupesha, Prof. Rajkumar Rangaswamy , Dr Juergen Ehrensberger, Dr Chan Mun Choon and Prof. Nitin.V.Pujari, Prof. Dr K Shridhar. I would like to also mention my friends Shiva Shankar, Paresh Nadig, Harsha SK, Nithin, Ashwin, Vijay Kumar, Ajay Kumar, Umesh Rao, Sumanth Rao, Susheela, Pallavi Pai, Shruthi K, Chandrakanth K, Shiva Kumar, Shashikiran RP, Pranav HU, Siddharth Goel, Veeresh Hiremath, Raveendra Pai, Praveen Raj, Sanjeev Reddy, Vishwanath, Yashaswi Chaudhary, Suhas Bharadwaj, Vinod Kumar M, Praveen Poojari, Prashanth Rajendran, Rajeev Ravi and Padmanabha Venkatagiri Sheshadri, Xiang Fau, Manjunatha Doddavenkatappa, Girisha D'Silva, Nimanatha Baranasuriya who have been my prime pillars of support always. I would like to also thank Elvis D'Souza and Umair Z Ahmed who have helped me during my under graduation. I would like to also thank everyone who helped me on the irc channel with all my doubts. I would like to also thank my mentors Kiran Hathwar, Hariom Srivastava, Kshitij Saxena, Rahul Bendre, Mr. Krishnakumar, Mr. NS Nagaraja, Mr. Madhu Iyengar, Mr. Srinivas Thonse, Mr. Kurt Griffiths, Mr. Alejandro Cabrera, Mr. Fabio Percoco, Supreeth Kumar, Arun Das, Siva Ekambaram, Rajith Noojibial, Sunil Patro, Souvik Dutta, Neelanshu Goel, my teammates Rajath

Singh, Sachin Mannur and Anirjit Bannerjee as well as my colleagues from InMobi and Zeta albeit for a short duration. I would also like to thank my non technical mentors, Respected Vid Sri S Venkatesh and Vid Sri HS Venugopal who have been a guiding light in my life. I would like to also acknowledge the support of all my cousins Ramyashree, Ranjitha, Abhijna, Prathiha and Bharath as well as Tejaswini Pillai and Sunil Mudambi.

As we say in Bharat:

" Sarve Bhavanthu Sukhinaha , Sarve Santhu Niramayaha, Sarve bhadrani Phashyantu, Ma Kaschit dukha baarbhaveth. Om Shanthi, Shanthi, Shanthihi".

"Om, May All become Happy, May All be Healthy (Free from Illness) May All See what is Auspicious, May no one Suffer in any way."

I would like to extend my humblest regards to all the great saints and philosophers of my country from whom great thoughts have emerged and spread to all over the world today. If anyone asks me why I am proud of my country I would always say " Bharat is the first country which spoke about looking inwards for happiness".

I am also grateful to BPB Publications for their guidance and expertise in bringing this book to fruition. It was a long journey of revising this book, with valuable participation and collaboration of reviewers, technical experts, and editors. Special thanks to the team who have coordinated with us flawlessly.

Last but not the least to my wonderful friend and co author Sri Praveen Raj who is an epitome of dedication and silent commitment to his craft. This book is brought to life greatly by his commitment as well as my brother Sri Jayanth Raghu who was also the technical reviewer for the same.

— Prashanth Raghu

Preface

Open-Source development has been the single largest contributor to the development community and the contributions are used across the IT landscape including corporations, universities, schools etc. Despite being highly popular, contributing to open source and navigating open source systems has always been a challenge and some reasons could be due to most open source developers holding other day jobs which might also be their regular work.

Memcached (**https://memcached.org/**) is one of the largest commercially and non-commercially used caching systems in the IT world, with a relatively smaller codebase that enables developers to understand the basic skelitals of source code structures, while really enabling developers to hone their skills towards development of high quality production grade, enterprise ready open source software. Most server based technologies

Open source development skills enable developers to understand intricacies of softwares enabling powerful decision making while choosing technologies over trends and making mature decisions on a day to day development basis. This books is for all programming enthusiasts and is designed to be a Source code 101 level book but can help developers elevate to architect level decision making as well as architects to make solid software decisions while building sustainable and scalable systems.

Chapter 1: Source Code Explorations in Open-Source Systems - Navigating large open source code bases have both been tricky and difficult to understand the intent of the created code. Despite this, it happens to be one of the most sought after skills to develop and architect high performance code. In today's age of scalable systems some of the toughest problems contain the answers in understanding the codebases of the used systems. This chapter explores some common ways to begin and navigate the journey and understanding differences and similarities between different open source systems.

Chapter 2: Getting Started with Memcached Design - Most server based systems to accommodate for high network IO capability on commodity hardware and resource utilization adopt the usage of the event loop based systems to manage both the IO connections as well as network management. Once the data is input into the system, the protocol manager reads the data to identify the type of data coming from the client. The data input is then placed on the LRU cache with the management of the expiry of individual data units. The network manager manages the connection lifecycle from clients as well as managing the timeouts and their termination if needed.

Chapter 3: Design of Event Loop - Asynchronous Socket driven systems contain an event loop that drives connections to the server using a shared thread that handles both connection and data management to the application. The event loop can be created using operating system primitives which help application developers minimize the time to both create and manage the event loop and the operations associated with it.

Chapter 4: Server Initialization - Initialization of the server includes parameters for the users to configure the server for both performance and scalability as well as few parameters specific to the underlying system.

Chapter 5: LRU Cache - The LRU Cache is the design of the internal cache memory that holds users data as Key Value pairs. Memory being limited and memcached being a caching system, the unused key value pairs have to be evicted from the system. The LRU Cache contains mechanisms to remove these unused KV pairs to help keep the cache leaner.

Chapter 6: Slab Management - Allocation of memory to user's key value pairs on a demand basis, requires multiple calls to operating system primitives. Memcache uses the slab allocation system to preallocate memory as well as divide the data into pages according to the requested size. Preallocation of memory reduces the number of memory allocations to be made in the runtime. The memory allocator is also protected by semaphore locks to prevent memory corruption during allocation, reallocation and memory management.

Chapter 7: Server Authentication - Authentication to Memcached is not mandatory but a suggested step for connecting to Memcached. The server provides a standard SASL auth allowing users and connections with a username / password to be able to login to the system. Additionally memcached also supports optional SSL auth to allow higher level of security to connections trying to manipulate key data.

Chapter 8: Protocol Definitions - Memcached uses a custom protocol over TCP / UDP to provide operations to its clients. The chapter deals with the design of the protocol, headers, request and response bodies, as well as the status codes for success and unsuccessful responses.

Chapter 9: Background Processes - Memcached uses background operations to perform a few non synchronous critical tasks such as slab rebalancing, dumping memcached data to files. This chapter deals with the flow of background processes and their generic structures.

Chapter 10: Proxy Server Design and External Storage - Memcached provides backdoor entry for administrators to dynamically configure the server as well as examine server statistics and vitalities as well as live user data.

Chapter 11: Using Memcached at Scale - Memcached has been chosen as the tool of choice owing to its commercial success as well as the simplicity of code along with the usage of most common architecture patterns for client server based open source systems. Use cases of caching softwares could range for a wide range of cases right from storage of audio, video and other multimedia formats to textual data ranging from database results, user session data to be distributed across different machines. This chapter covers many usages at large scale in organizations such as facebook, twitter, pinterest etc. One of the major drawbacks of memcached is the lack of support for clustering and hence this chapter talks on the workarounds created in these systems to support clustering at such large workloads.

Chapter 12: Continuation of the Exploration Journey - Memcached has been chosen as the tool of choice owing to its commercial success as well as the simplicity of code along with the usage of most common architecture patterns for client server based open source systems. The chapter covers identification of common patterns between different tools as well as understanding the same to help continue the journey.

Coloured Images

Please follow the link to download the
Coloured Images of the book:

https://rebrand.ly/698cc8

We have code bundles from our rich catalogue of books and videos available at **https://github.com/bpbpublications**. Check them out!

Errata

We take immense pride in our work at BPB Publications and follow best practices to ensure the accuracy of our content to provide with an indulging reading experience to our subscribers. Our readers are our mirrors, and we use their inputs to reflect and improve upon human errors, if any, that may have occurred during the publishing processes involved. To let us maintain the quality and help us reach out to any readers who might be having difficulties due to any unforeseen errors, please write to us at :

errata@bpbonline.com

Your support, suggestions and feedbacks are highly appreciated by the BPB Publications' Family.

Piracy

If you come across any illegal copies of our works in any form on the internet, we would be grateful if you would provide us with the location address or website name. Please contact us at **business@bpbonline.com** with a link to the material.

If you are interested in becoming an author

If there is a topic that you have expertise in, and you are interested in either writing or contributing to a book, please visit **www.bpbonline.com**. We have worked with thousands of developers and tech professionals, just like you, to help them share their insights with the global tech community. You can make a general application, apply for a specific hot topic that we are recruiting an author for, or submit your own idea.

Reviews

Please leave a review. Once you have read and used this book, why not leave a review on the site that you purchased it from? Potential readers can then see and use your unbiased opinion to make purchase decisions. We at BPB can understand what you think about our products, and our authors can see your feedback on their book. Thank you!

For more information about BPB, please visit **www.bpbonline.com**.

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

https://discord.bpbonline.com

Table of Contents

Chapter 1

Source Code Explorations in Open-Source Systems

Introduction

Welcome to the navigation world, not the navigation done by sailors or the navigation control systems used in satellites, rockets, or other locomotive systems, but in the ocean of open-source code. While we are all sailors looking at the vast sea from the horizon, with numerous sailors operating both on the horizon and the high seas, the maps available to sail to the high seas are only a few. The ones operating there have not been able to create many maps for new sailors to start sailing on their routes.

This brings us to the problem at our hands: navigating open-source code and systems, which are large systems with millions of lines of code, in many cases undocumented, making the learning curve very steep for new enthusiasts to be able to learn and contribute to these systems.

The incentives to learn open-source systems being very high prompts many students and engineers to jump into the arena, often being derailed due to a lack of systematic documentation. This chapter covers the reasons to master, navigate and understand open source code systems and its benefits.

Memcached captures the common essence of most open-source systems and is concise for enthusiasts willing to begin exploring open-source systems. This chapter does emphasize common factors across systems. This book aims to help enthusiasts begin their journey

with a small code base and still be able to understand the core aspects of open-source systems.

Structure

In this chapter, we will discuss the following topics:

- Need to understand open-source code
  - Open-source systems commonly used around us
- Commonalities of different systems
- Common structure of open-source code systems
  - Event loops in network-based systems
  - Memory organization
  - Data structures
- Benefits of understanding, contributing to and extending open-source systems
- Open-source licensing

Objectives

After studying this chapter, you will develop the ability to recognize the importance of understanding open-source code. This involves appreciating why delving into the intricacies of such codebases is crucial. Furthermore, you will gain insights into the structural similarities that underlie different open-source codebases. This will be achieved through the use of illustrative examples that highlight common patterns across various projects. A clear understanding of the significance of documentation within open-source code systems will also be obtained. Additionally, you will acquire the skills necessary to effectively expand upon existing open-source code systems, facilitating their ongoing evolution. Lastly, you will attain a comprehensive comprehension of prevalent open-source system licenses and their far-reaching implications. This knowledge will encompass a deeper understanding of how licenses influence aspects like usage, distribution, contribution, and more.

Need to understand open-source code

Sea swimmers love to enjoy the sights under the sea from the shallower side, commonly referred to as snorkeling. Although this is enjoyable and appreciable, one needs to get to the bottom of the ocean to see the gems. Advanced use cases, performance studies, architecture decisions in high-scalability systems, bug evaluations, and so on require an in-depth understanding of the system being used. Understanding open-source code is one of the most sought-after skills in the industry today. Despite this, the number of professionals

in this area has been limited and continues to be so due to a high entry barrier. It prevents easy onboarding for new contributors to start understanding and contributing. Despite that, the blame cannot be handed to the developers of these systems, as they spend a considerable amount of their free time helping developers around the world at almost no charge. While some open-source systems have enterprise versions or enterprise modules, the efforts are always praiseworthy.

Let us consider the case of an architect; designing a massively scalable system such as Facebook/Instagram needs the selection of a caching system. They are presented with options and are asked for their reasoning for selection. Common solutions would be to study the benchmarks and performance studies provided by individual organizations or other study groups. Although this could help, there could be biases or many points missing. The other approach could be the architect taking up the source codebase of the systems and analyzing the performance in key areas like connection limits, throughput, and so on. This not only helps them understand the system but also gives an in-depth picture of the usability of the system, which can later be used to configure the system as per the requirements of the use case. The team can also fork the main repository and create a version that suits their needs.

An example could be to write an independent memory allocation system, overriding the default considering the known size or format of the data being stored/optimizing for data types like images or videos. These in-depth changes to an open source system or optimizating it require understanding the systems in totality. These are not imaginary use cases, as we will see in *Chapter 11, Using Memcached at Scale,* where such use cases are mentioned in depth. Facebook had changed the routing protocol within Memcached to create a fork, to suit its needs for higher bandwidth availability.

The question could naturally arise as to why large enterprises, such as Facebook, choose to change the code over writing their custom versions of the same. The driving force behind this is the common knowledge base of open-source code developers, which is leveraged over a small set of developers working on software. Open-source software derives a major benefit from the best developers in the world, contributing to creating the most adaptable, scalable, and usable software using state-of-the-art coding and evaluation processes. This results in great systems, especially from a security standpoint, owing to the number of eyes on the code base to catch such errors before they enter into usage. Hence, most large enterprises prefer to use this common knowledge base and provide sponsorship to thank and help the community grow while writing the custom modules needing optimizations within the organization.

The benefits of understanding the source code are not only for enterprise architects. The other beneficiaries are the student communities who have time and curiosity and can be impacted by learning the systems ground up rather than using manuals or books to understand the functionalities. The students can slowly graduate from developers to architects providing state-of-the-art suggestions and developing enterprise-grade systems with a solid-working knowledge of the systems being used.

Understanding open-source code is necessary for every developer to develop state-of-the-art code for high scalability, performance, and security.

Open-source systems commonly used around us

The world around us is impacted to a greater extent by open-source systems than we realize. From the phones we use and the vehicles we operate to the electric grids around us and the watches we sport, open-source code has impacted people's lives in various ways, much to our unawareness.

Some of the open-source systems used widely are listed here:

- **Android**, one of the most widely used operating systems for smartphones, is known to be an open-source project. Although it could be argued in the strictest sense that the majority of the development happens with Google, the source code is made available to the world to understand the implications and provide quantitative feedback to improve the system.
- **FreeRTOS**, an open-source embeddable operating system with a remarkably low memory footprint, is used in a wide range of systems, from smartwatches and trains to controlling systems and IoT devices, including common household systems like doorbells and emergency fire alarms. These systems are even used aboard airplanes and stand as a testament to the reliability and security of the software developed by the community.
- **Linux** stands as a testament to an individual's vision that could lead to a total change in the thinking of an entire industry and shifting development paradigms to expand to the cloud revolution. The impact of Linux is ubiquitous on almost all systems and software that we use, from operating systems like Android to MacIntosh.
- **MySQL and PostgreSQL**; it would not be wrong to say that if start-ups can spawn up in every corner of the world, the role of open-source software is large. Ranging from programming languages and their libraries to data storage systems like MySQL, everything is offered free for development and enterprise usage. MySQL has revolutionized data management by offering RDBMS solutions for free, which is suitable for operation at a high scale. The impact is seen worldwide in every application sector, from software services to banking and governance.
- **Programming languages** like Python, Lua, C, C++ (compilers), Lua, Rust, database systems, caching systems, web frameworks, and libraries, the availability of open-source code systems has greatly impacted visibility and democratized developer access to enterprise-grade software, resulting in rapid innovation at minimal

cost. The impact of open-source software is greater than we can imagine, and the software mentioned here are just the tip of the iceberg;.

- **Kubernetes** (commonly referred to as K8s) is an open-source container orchestration platform that automates containerized applications' deployment, scaling, and management. Google initially developed it, and it is now maintained by the **Cloud Native Computing Foundation** (**CNCF**).

  Kubernetes, being an open-source project, offers collaborative development, transparency, flexibility, vendor-agnosticism, cost-effectiveness, a rich ecosystem, continuous improvement, and strong community support.

Commonalities from different systems

A programming language contains a parser, symbol table generator, compiler, assembler, and interpreter, to mention a few common phases, along with memory management, thread management, configurations, data structures, and many more. While exploring multiple languages, we can note that the code stages remain almost identical. The parsing is depicted as grammar that is internally a decision tree. It is worth noting that while initial efforts could be needed to understand the working of these systems, the others would soon be a breeze. Although the initial barrier could be high, the commonalities between various systems greatly reduce the time required for understanding other open source systems in the journey.

Common structure of open-source code systems

The open nature of the ecosystem with code accessible through the internet and the liberal nature of the licenses meant developers had access to the ecosystem's knowledge to understand and use common structures, enabling other systems. For example, hashmaps and dictionaries share similar implementation in Python, Redis, and Lua. Such striking similarities can be observed while examining and comparing open-source systems. We can see that most open-source systems use a similar memory management system to reduce the number of OS calls. This commonality greatly helps us understand these systems. In this chapter, we aim to generate higher curiosity by practically using small code snippets to demonstrate such similarities.

Event loops in network-based systems

Network-based systems provide services on a port and are typically served through the internet, with clients connecting from across the world. Most of these systems need clients to connect and access services on a larger scale. Although the concept of event loops will be covered in detail, it could be considered to make several connected clients share a common