

VALTRUS INNOVATIONS LTD. and
KEY PATENT INNOVATIONS LTD.,

V.

Defendants.

JURY TRIAL DEMANDED

THE PARTIES

IPR2026-00039 EX2005 001

2. KPI is the beneficiary of a trust pursuant to which Valtrus owns, holds, and asserts the Patents-in-Suit as defined in Paragraph 30. KPI is an Irish entity duly organized and existing under the laws of Ireland. The address of the registered office of KPI is: The Glasshouses GH2, 92 Georges Street Lower, Dun Laoghaire, Dublin A96 VR66, Ireland.

3. Defendant The Home Depot, Inc. is a company organized under the laws of the State of Delaware, with a principal place of business at 2455 Paces Ferry Road SE, Atlanta, Georgia 30339. The Home Depot, Inc. may be served through its registered agent for service, Corporation Service Company located at 251 Little Falls Drive, Wilmington, Delaware 19808. Upon information and belief, The Home Depot, Inc. does business in Texas, directly or through intermediaries, and offers its products and/or services, including those accused herein of infringement, to customers and potential customers located in Texas, including in the Judicial District of the Eastern District of Texas.

4. Defendant Home Depot U.S.A., Inc. is a corporation organized under the laws of the State of Delaware, with its principal place of business located at 2455 Paces Ferry Road SE, Atlanta, Georgia 30339. Home Depot U.S.A., Inc. may be served through its registered agent for service, Corporation Service Company, D/B/A/ CSC-Lawyers Incorporated, located at 211 E. 7th Street, Suite 620, Austin, Texas 78701.

5. Home Depot U.S.A., Inc. is a wholly-owned subsidiary of The Home Depot, Inc. As used herein, “Home Depot” or “Defendants” refers collectively to Home Depot U.S.A., Inc. and The Home Depot, Inc. The assets, liabilities, income, and expenditures of Home Depot U.S.A., Inc. are included in the consolidated financial statements of The Home Depot, Inc. Home Depot U.S.A., Inc., on behalf of itself and The Home Depot, Inc., operates Home Depot retail

stores in this District and throughout the United States and the Home Depot retail website, www.homedepot.com, among other websites.

6. Home Depot U.S.A., Inc. is controlled and managed by The Home Depot, Inc. in connection with Defendants' infringing activities pleaded herein. Defendants function as an integrated organization and a single business enterprise in connection with those activities. Defendants hold themselves out as a single business enterprise in their advertising and in connection with the trademark "The Home Depot" in promoting the sale of products through Home Depot retail stores and homedepot.com, without any apparent distinction regarding which Defendant is offering or would deliver those products and services.

7. Defendants control, participate in the commission of, and have a direct financial interest in the infringing acts set forth herein.

JURISDICTION AND VENUE

8. This is an action for patent infringement arising under the patent laws of the United States, 35 U.S.C. §§ 1, *et seq.* This Court has jurisdiction over this action pursuant to 28 U.S.C. §§ 1331 and 1338(a).

9. Home Depot's online e-commerce platform works hand-in-hand with its physical stores and fulfillment centers, including those located within this District:

We are also continuing to expand our fulfillment network, investing in a significant number of new fulfillment facilities to drive speed and reliability of delivery for our customers and to help us ultimately meet our goal of reaching 90% of the U.S. population with same or next day delivery for extended home improvement product offerings, including big and bulky products. These facilities include omni-channel fulfillment centers, which deliver product directly to customers, and market delivery operations, which function as local hubs to consolidate freight for dispatch to customers for the final mile of delivery, with a focus on appliances. . . . In addition to our distribution and fulfillment centers, we leverage our stores as a network of convenient customer pickup, return, and delivery

fulfillment locations. Our premium real estate footprint provides a distinct structural and competitive advantage.¹

10. Home Depot's online platforms support its interconnected retail strategy: "Online sales, which consist of sales generated online through our websites and mobile applications for products picked up at our stores or delivered to customer locations, represented 14.2% of net sales and grew by 7.4% during fiscal 2022 compared to fiscal 2021. The increase in online sales in fiscal 2022 was a result of customers continuing to leverage our digital platforms and reflects our ongoing investments to enhance these platforms and related fulfillment capabilities, which support our interconnected retail strategy."²

11. Home Depot intends for its websites to complement its stores: "Our online product offerings complement our stores by serving as an extended aisle, and we offer a significantly broader product assortment through our websites and mobile applications, including homedepot.com, our primary website"³

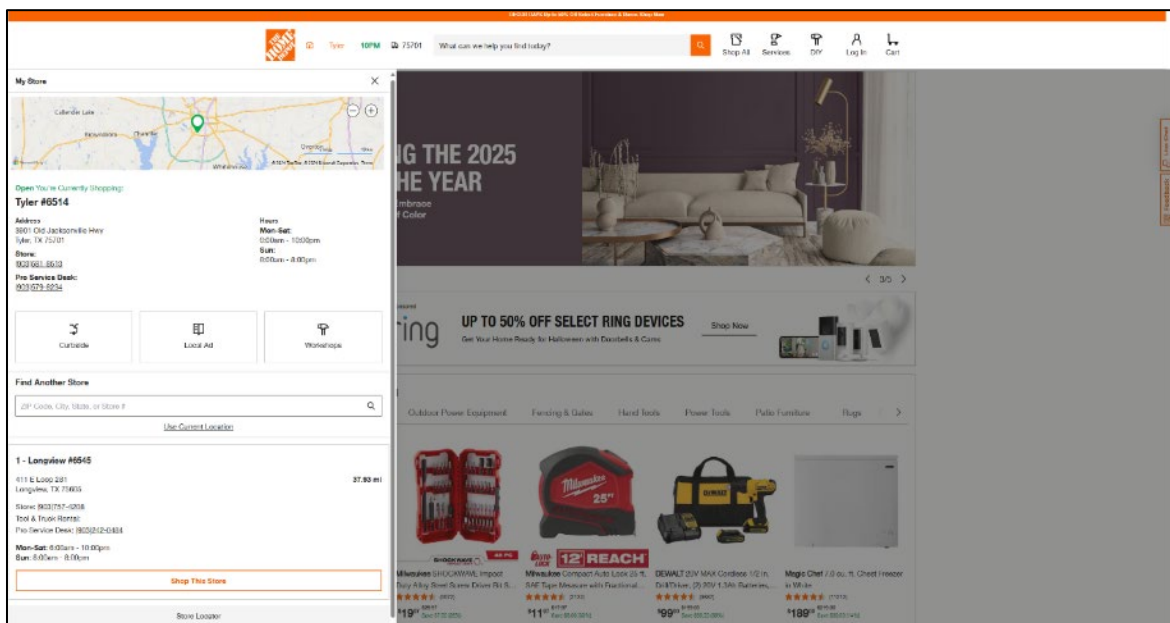
12. When a customer who resides in the Eastern District of Texas visits the Home Depot website, www.homedepot.com, the website directs that customer to a specific store in this District. For example, when a customer residing in Tyler, Texas shops on the Home Depot website, the Tyler Home Depot address, 3901 Old Jacksonville Highway, populates on the customer's screen, indicating where the customer is shopping and providing information relating to that location. In the image below, the webpage recommends the Tyler Home Depot in Tyler,

¹ The Home Depot 2022 Annual Report at 5, available at https://ir.homedepot.com/~/_media/Files/H/HomeDepot-IR/2023/2022-HD-Annual-Report.pdf (hereinafter, "Home Depot 2022 Annual Report")

² Home Depot 2022 Annual Report at 28

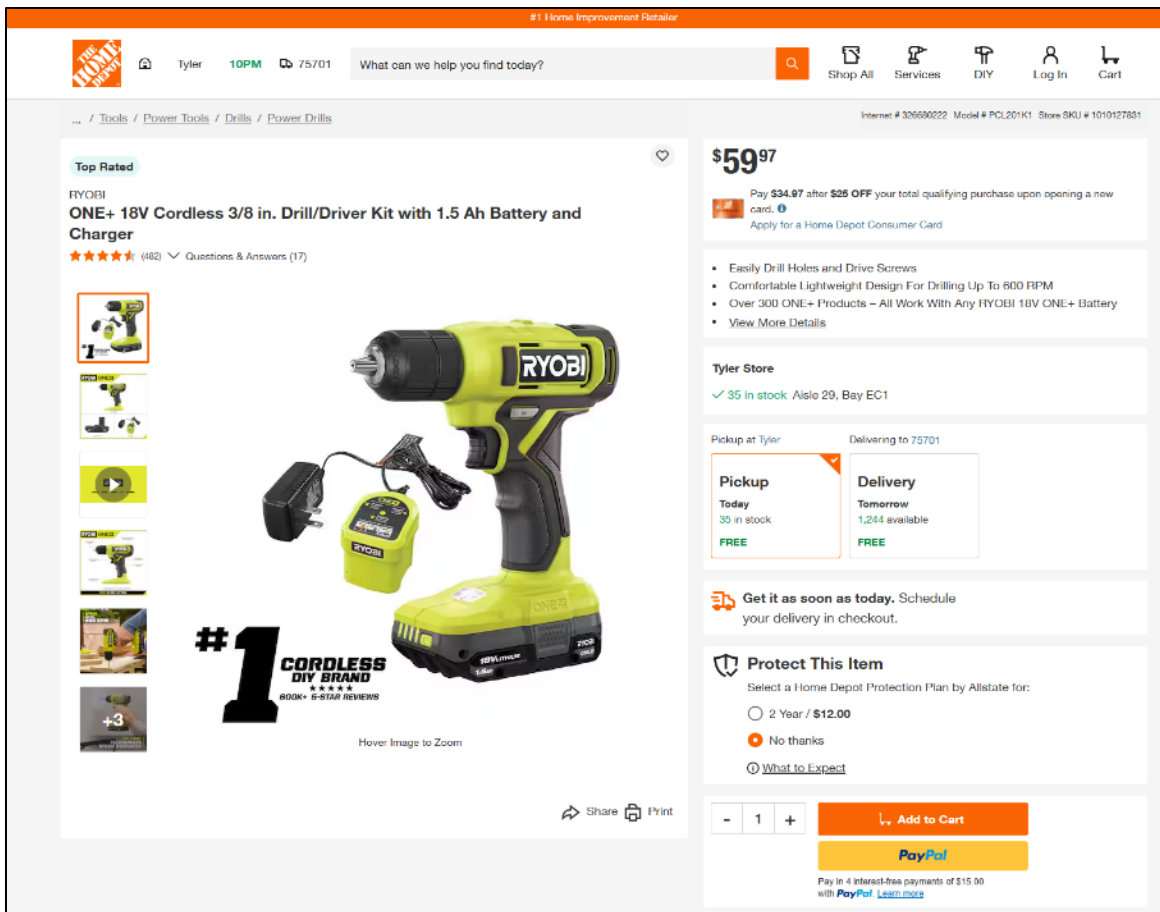
³ *Id.* at 2.

Texas as the physical Home Depot store where the customer's online shopping choices will be fulfilled.



13. As shown in another screenshot of an online transaction conducted by a customer located in this District, when a customer residing in the Eastern District of Texas selects a product, the website directs the customer to a local store (the Tyler store in this example) in this District where the customer will receive that item. The webpage even informs the customer of the specific location within the store in this District, e.g., the aisle and bay in the aisle (Aisle 29, Bay EC1 in the image below) where that product will be located within the store for the customer to retrieve it.

⁴ https://www.homedepot.com/?mtc=SEM-BF-CDP-BNG-Multi-Multi-NA-Multi-NA-RSA-NA-NA-NA-BT1-NA-NA-NA-THD_CORE&cm_mmc=SEM-BF-CDP-BNG-Multi-Multi-NA-Multi-NA-RSA-NA-NA-NA-BT1-NA-NA-NA-THD_CORE-71700000002466128-58700008684477045-43700079706214486--&gclid=2cafb58be368125d8578d28ba880a2c2&gclsrc=3p.ds&ds_rl=5041&msclkid=2cafb58be368125d8578d28ba880a2c2



14. The Home Depot stores, and the computers located in these stores in this District, not only provide relevant details regarding the location of particular products in the stores but also display the available products and amount in stock, all as part of Home Depot's infringement of the Patents-in-Suit. The provisioning of this product information originates from Home Depot's stores located in this District, and also serves as part of Home Depot's infringement of the Patents-in-Suit in this District.

15. Home Depot's in-store servers located in this District and/or edge servers controlled by Home Depot located in this District and elsewhere in Texas process information about products

⁵ <https://www.homedepot.com/p/RYOBI-ONE-18V-Cordless-3-8-in-Drill-Driver-Kit-with-1-5-Ah-Battery-and-Charger-PCL201K1/326680222>

in this District so they can be delivered to customers in this District, all of which constitutes material conduct that is part of, and helps establish, Home Depot's infringement of the patented shopping system and techniques.

16. Home Depot conducts business from and operates from at least the following locations in this District where online orders can be fulfilled: (1) 411 East Loop 281, Longview, Texas 75605; (2) 3901 Old Jacksonville Highway, Tyler, Texas 75701; (3) 2530 S Jefferson Avenue, Mount Pleasant, Texas 75455; (4) 4110 St. Michael Drive, Texarkana, Texas 75503; and (5) 3120 NE Loop 286, Paris, Texas 75460.

17. This Court has personal jurisdiction over Defendants. Defendants regularly conduct business and have committed acts of patent infringement and/or have induced acts of patent infringement by others in this Judicial District and/or have contributed to patent infringement by others in this Judicial District, the State of Texas, and elsewhere in the United States. Defendants, directly and/or through subsidiaries or intermediaries, have committed and continue to commit acts of infringement in this District by, among other things, making, using, importing, offering to sell, and/or selling products that infringe the Patents-in-Suit as defined in Paragraph 30.

18. This Court has general jurisdiction over Home Depot due to its continuous and systematic contacts with the State of Texas, including through its operation of approximately 182 physical retail stores in the State of Texas, its targeting of Texas residents with the homedepot.com website, its ownership and/or lease of land in the State of Texas, and other business activities throughout the State of Texas.

19. Home Depot is subject to the specific personal jurisdiction of this Court because Plaintiffs' patent infringement claims against Home Depot. specifically arise from Home Depot's

acts of infringement in the State of Texas. These acts of infringement include directing and controlling the operation of the homedepot.com website using the patented inventions, and specifically targeting residents of Texas with this website to further the sale of products and services to those customers online and at Home Depot's physical stores in this District.

20. Home Depot U.S.A., Inc.'s presence and acts of infringement committed in this District are attributable and imputed to The Home Depot, Inc. for venue purposes. The Defendants have not maintained corporate separateness. On information and belief, the Defendants function as an integrated organization in the operation of the homedepot.com website, its retail stores with respect to the infringing actions complained of herein. For example, Home Depot states in its annual report, "When we refer to 'The Home Depot,' the 'Company,' 'we,' 'us' or 'our' in this report, we are referring to The Home Depot, Inc. and its consolidated subsidiaries."⁶

21. Upon information and belief, as shown above, Home Depot operates and/or contracts with third parties to operate in-store servers, in-store routers, and/or edge servers in this District to receive, transmit, process, and/or collect information about the searches and purchases made from customers' devices located in this District constituting material conduct as part of Home Depot's infringement of the patented shopping system and techniques.

22. Further, Defendants hold themselves out as a single business enterprise under the trademark "The Home Depot" in connection with the infringing actions without any apparent distinction regarding which Defendant is performing those actions. For example, Defendants' stores are branded as "The Home Depot" with no apparent distinction over whether the store is operated by The Home Depot, Inc. or Home Depot U.S.A., Inc. Home Depot touts the significance of its brand name: "Our business has one of the most recognized brands in North America. As a

⁶ Home Depot 2022 Annual Report at 1

result, we believe that The Home Depot® trademark has significant value and is an important factor in the marketing of our products, e-commerce, stores and business.”⁷

23. Venue is proper in this District under 28 U.S.C. §§ 1391(b)-(d) and 1400(b). Defendants are registered to do business in the State of Texas, have offices in the State of Texas, have transacted and continue to transact business in the Eastern District of Texas, and have committed and continue to commit acts of direct and indirect infringement in the Eastern District of Texas.

24. Defendants are subject to this Court’s jurisdiction pursuant to due process and/or the Texas Long Arm Statute due at least to their substantial business in this State and Judicial District, including (a) at least part of their past infringing activities, (b) regularly doing or soliciting business in Texas, and/or (c) engaging in persistent conduct and/or deriving substantial revenue from goods and services provided to customers in Texas.

PATENTS-IN-SUIT

25. On March 8, 2011, the United States Patent and Trademark Office duly and legally issued U.S. Patent No. 7,904,686 (the “’686 Patent”) entitled “Data Security For Use With A File System”. A true and correct copy of the ’686 Patent is attached hereto as Exhibit A.

26. On December 29, 2009, the United States Patent and Trademark Office duly and legally issued U.S. Patent No. 7,640,332 (the “’332 Patent”) entitled “System and Method for Hot Deployment/Redeployment in Grid Computing Environment”. A true and correct copy of the ’332 Patent is attached hereto as Exhibit B.

27. On June 27, 2006, the United States Patent and Trademark Office duly and legally issued U.S. Patent No. 7,068,597 (the “’597 Patent”) entitled “System and Method for Automatic

⁷ *Id.* at 3.

Load Balancing in a Data-Over-Cable Network”. A true and correct copy of the ’597 Patent is attached hereto as Exhibit C.

28. On October 10, 2006, the United States Patent and Trademark Office duly and legally issued U.S. Patent No. 7,120,832 (the “’832 Patent”) entitled “Storage Device Performance Monitor”. A true and correct copy of the ’832 Patent is attached hereto as Exhibit D.

29. On September 12, 2006, the United States Patent and Trademark Office duly and legally issued U.S. Patent No. 7,107,326 (the “’326 Patent”) entitled “Method and System for Integrating IP Address Reservations with Policy Provisioning”. A true and correct copy of the ’326 Patent is attached hereto as Exhibit E.

30. Plaintiffs are the sole and exclusive owner of all legal right, title, and interest in the ’686 Patent, the ’332 Patent, the ’597 Patent, the ’832 Patent, and the ’326 Patent (collectively, the “Patents-in-Suit” or “Asserted Patents”), and hold the exclusive right to take all actions necessary to enforce their rights to the Patents-in-Suit, including the filing of this patent infringement lawsuit. Plaintiffs also have the right to recover all damages for past, present, and future infringement of the Patents-in-Suit and to seek injunctive relief as appropriate under the law.

FACTUAL ALLEGATIONS

31. The ’686 Patent generally relates to a system and apparatus for providing data security for use with a file system. The technology described in the ’686 Patent was developed by Ranganath G. Iyengar at Hewlett-Packard Development Company, L.P. For example, the technology is implemented by Home Depot’s use of Apache Software Foundation’s Hadoop Distributed File System (“HDFS”); see apache.org.

32. The ’332 Patent generally relates to technology for hot deployment and/or redeployment in a grid computing environment, where the grid computing environment adds a

new version of an application from a repository server, determining which grid node is running the application affected by the new version of the software, notifying the client application manager associated with one or more of the affected grid nodes, and hot deploying/redeploying the new version of the application to the affected grid nodes. The technology described in the '332 Patent was developed by Sanjay Dahiya at Hewlett-Packard Development Company, L.P. For example, the technology is used by Home Depot and/or is implemented in systems using Kubernetes for rolling updates.

33. The '597 Patent generally relates to a system and method for load balancing in a network system, such as a data-over-cable system. The technology described in the '597 Patent was developed by John G. Fijolek, Irene Gilbert, Ali Akgun, Shahidur Khan, and Vikram Swamy. For example, the technology is implemented by Home Depot's use of the Google Cloud Platform featuring AMD EPYC processors.

34. The '832 Patent generally relates to technology for monitoring performance of a storage device by intercepting communications between a computer system and said storage device, analyzing the intercepted communications, and reallocating at least some of said data on said storage device to enhance the performance of the storage device. The technology described in the '832 Patent was developed by Kevin Collins and Michael P. Fleischmann at Hewlett-Packard Development Company, L.P. For example, the technology is used by Home Depot and/or is implemented in systems that use Apache Kafka Streams using RocksDB Universal Compaction and also those using Apache Cassandra.

35. The '326 Patent generally relates to a system and method for policy provisioning and access managing on a data-over-cable system. The technology described in the '326 Patent was developed by John G. Fijolek, Irene Gilbert, and Jaideep Kulkarni at Hewlett-Packard

Development Company, L.P. For example, the technology is implemented by Home Depot's use of the Google Cloud Platform, such as through Identity and Access Management ("IAM") for Cloud Load Balancing, Backend service-based external TCP/UDP Network Load Balancing, and External HTTP(S) Load Balancing.

36. Home Depot has infringed the '832 Patent, the '326 Patent, the and the '597 Patent by making, using, selling, offering to sell, and/or importing, and by actively inducing others to make, use, sell, offer to sell, and/or import the infringing products identified above.

37. Home Depot has infringed and is continuing to infringe the '686 Patent and the '332 Patent by making, using, selling, offering to sell, and/or importing, and by actively inducing others to make, use, sell, offer to sell, and/or import the infringing products identified above.

COUNT I
(Infringement of the '686 Patent)

38. Paragraphs 1 through 37 are incorporated by reference as if fully set forth herein.

39. Plaintiffs have not licensed or otherwise authorized Defendants to make, use, offer for sale, sell, or import any products that embody the inventions of the '686 Patent.

40. Defendants have directly infringed and continue to directly infringe the '686 Patent, either literally or under the doctrine of equivalents, without authority and in violation of 35 U.S.C. § 271, by making, using, offering to sell, selling, and/or importing into the United States products that satisfy each and every limitation of one or more claims of the '686 Patent. Such products include the HDFS.

41. For example, Defendants have directly infringed and continue to directly infringe at least claim 1 of the '686 Patent by making, using, offering to sell, selling, and/or importing into the United States products that include HDFS.

42. Upon information and belief, Home Depot has used the HDFS since at least 2017.



Data Engineer

The Home Depot

Jul 2018 - Jul 2019 · 1 yr 1 mo

Austin, Texas Area

I was a leader in the migration of several data warehouses processes—each handling terabytes of data daily—from on-premise Hadoop to Google Cloud Platform's BigQuery. I also trained team members in technologies such as CI/CD, Apache Airflow, and Google BigQuery.

8



Sql Server Database Administrator

The Home Depot

Oct 2019 - Sep 2022 · 3 yrs

United States · Remote

- + Managed and optimized multiple database systems, including MySQL, PostgreSQL, and Microsoft SQL Server.
- + Responsible for all servers running in SQL platform (SQL Server 2005/2008/2008 R2/2012/2014/2016/2019/2022).
- + Spearheaded end-to-end migration of SQL servers from on-premises infrastructure to cloud platforms (Azure/GCP).
- + Proficient in setting up and configuring SQL Server, including the implementation of AlwaysOn Availability Groups (AO AG) and Windows Server Failover Clustering (WSFC) for high availability and disaster recovery.
- + Competent in active directory integration w/SQL Server and automating infrastructure management using Terraform.
- + Optimized SQL queries, CDC processes, stored procedures, functions, indexes, & views to enhance DB performance.
- + Developed multiple Nodes (Active/Passive) in SQL Servers to add failover clustering features for high availability.
- + Utilized Big Data technologies like Hadoop and Spark to process and analyze large datasets.
- + Configured Replication for production servers to provide high availability & supervised using Replication Monitor.
- + Established scheduled backups (transaction log, differential, and full), along with recovery sessions (Simple and Full).

9

⁸ <https://www.linkedin.com/in/will-hudgins/>

⁹ <https://www.linkedin.com/in/surbhi-agrawal-789670ba/>



Hadoop Architect

HomeDepot

Jan 2017 - Nov 2017 · 11 mos

Atlanta, GA

Formulated procedures for installation of Hadoop Patches, Updates and version upgrades and automated processes for troubleshooting, resolution and tuning of Hadoop Clusters

Developing and designing Data migration migration methods in spark using Scala to enhance the performance of spark over HIVEsql methods .

NiFi is built to automate the flow of data between systems.

Worked with HortonWorks Dataflow process to automate and manage the flow of analytical information between systems to create derived datasets, streaming data.

Scaling and tuning a Cassandra cluster

Used NiFi REST API/UI to obtain or alter data on the NiFi host system using the NiFi OS credentials.

Installed and configured Apache Hadoop, Hive and Pig environment on Amazon EC2 and assisted in designing, development and architecture of Hadoop and HBase systems

Horton works HDFS , Map Reduce, Hive, Pig, Java JDK 1.6 , AWS, Cent OS 6.4, Shell Scripting, Flume, Apache, Sqoop, base, Red Hat Linux 6.4, My SQL 5.5

Worked with automated Hortonworks Data Flow using Ambari 2.6.

HDF integrated with Streaming Analytics Manager via sinks – Cassandra/Solr and source as HDFS.

Knox proxy integration via NiFi

Implementation of the technical components of solutions, including tool selection, data architecture, database administration, ETL (extract, transform and load), reporting and integration.

Built data platforms, pipelines, storage systems using the Apache Kafka, Apache Storm and search technologies such as Elastic search. Migrated data from DB2 and Teradata to Hadoop.

Migrated data from hadoop to Google Cloud Storage.

Written automated scripts to deploy compressed files to BigTables via BigQuery in GCP

Environment HortonWorks Hadoop, Apache NiFi, DataFlow, IBM BigInsights, Google Cloud Platform(GCP),

Google BigQuery, PCF, Kafka, Flume, Linux scripting, Spark, DB2, TeraData, SQL Server, SQL, PL/SQL

10



Hadoop Developer

The Home Depot · Contract

Aug 2018 - Jul 2019 · 1 yr

11



Hadoop Developer

The Home Depot · Contract

Mar 2017 - Dec 2018 · 1 yr 10 mos

Atlanta, GA

12

¹⁰ <https://www.linkedin.com/in/vipul-mehta-7a985a184/details/experience/>

¹¹ <https://www.linkedin.com/in/rohith-s-032b131ab/>

¹² <https://www.linkedin.com/in/tanmai-b-b58611220/>



The Home Depot
 4 yrs 11 mos
 Greater Atlanta Area

- Senior Software Engineer**
 Jan 2017 - Apr 2017 · 4 mos
 On-site
 - Enabled and automated data pipelines for moving over 10 GB of data from Oracle and DB2 source tables to Hadoop and Google BigQuery using GitHub for source control and Jenkins.
 - Facilitated rebuilding of 1+ TB of Cloud inventory table through automated data transfer from Oracle / DB2 source tables to Hadoop HDFS / Hive and Google BigQuery using parameterized Sqoop jobs, Jenkins, and GitHub.
 - Collaborated with cross-functional teams to analyze user requirements, translating them into technical specifications and ensuring seamless integration with existing systems.
 - Mentored and coached junior engineers, providing guidance on coding best practices, design patterns, and industry standards, resulting in a 20% improvement in team productivity and code quality.


Databases, Root Cause and +24 skills

13

43. The HDFS performs a method of providing data security for use with a file system. For example, the Apache Hadoop framework provides security features to protect data present in Hadoop clusters. Hadoop uses HDFS file system (a file system) to store data. HDFS comprises a name node and multiple data nodes.

Hadoop, an open-source framework used for distributed storage and processing of big data, provides various security features to protect data and ensure secure access to Hadoop clusters. Here are some of the ways Hadoop is secured:

2. Authorization: Hadoop uses Access Control Lists (ACLs) to manage permissions for users and groups accessing data on the Hadoop Distributed File System (HDFS). Additionally, Hadoop provides fine-grained authorization through the use of Apache Ranger, which allows administrators to create policies that define access control for specific resources and actions.

14

¹³ <https://www.linkedin.com/in/brice-massala-a4949b2/>

¹⁴ https://medium.com/@big_data_landscape/hadoop-security-5dac0ffe209e

NameNode and DataNodes

HDFS has a master/slave architecture. An HDFS cluster consists of a single NameNode, a master server that manages the file system namespace and regulates access to files by clients. In addition, there are a number of DataNodes, usually one per node in the cluster, which manage storage attached to the nodes that they run on. HDFS exposes a file system namespace and allows user data to be stored in files. Internally, a file is split into one or more blocks and these blocks are stored in a set of DataNodes. The NameNode executes file system namespace operations like opening, closing, and renaming files and directories. It also determines the mapping of blocks to DataNodes. The DataNodes are responsible for serving read and write requests from the file system's clients. The DataNodes also perform block creation, deletion, and replication upon instruction from the NameNode.

15

The Communication Protocols

All HDFS communication protocols are layered on top of the TCP/IP protocol. A client establishes a connection to a configurable TCP port on the NameNode machine. It talks the ClientProtocol with the NameNode. The DataNodes talk to the NameNode using the DataNode Protocol. A Remote Procedure Call (RPC) abstraction wraps both the Client Protocol and the DataNode Protocol. By design, the NameNode never initiates any RPCs. Instead, it only responds to RPC requests issued by DataNodes or clients.

16

44. The HDFS performs the step of applying a mapping function to data block numbers that are associated with a file, wherein the data block numbers are contained in an index node associated with said file. For example, NameNode performs various operations, such as opening and closing of files, and mapping data block locations (e.g., data block number). Mapping (e.g., mapping function) is the process of determining which data block is stored at which location.

If you take another look at Figure 4-3, you can see the NameNode daemon running on a master node server. All mapping information dealing with the data blocks and their corresponding files is stored in a file named `fsimage`. HDFS is a journaling file system, which means that any data changes are logged in an edit journal that tracks events since the last *checkpoint* — the last time when the edit log was merged with `fsimage`. In HDFS, the edit journal is maintained in a file named `edits` that's stored on the NameNode.

¹⁵ http://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html#NameNode_and_DataNodes

¹⁶ <https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html>

The client opens the file it wishes to read by calling `open()` on the `FileSystem` object, which for HDFS is an instance of `DistributedFileSystem` (step 1 in Figure 3-2). `DistributedFileSystem` calls the namenode, using remote procedure calls (RPCs), to determine the locations of the first few blocks in the file (step 2). For each block, the namenode returns the addresses of the datanodes that have a copy of that block. Furthermore, the datanodes are sorted according to their proximity to the client (according to the topology of the cluster's network; see "Network Topology and Hadoop" on page 70). If the client is itself a datanode (in the case of a MapReduce task, for instance), the client will read from the local datanode if that datanode hosts a copy of the block (see also Figure 2-2 and "Short-circuit local reads" on page 308).

17

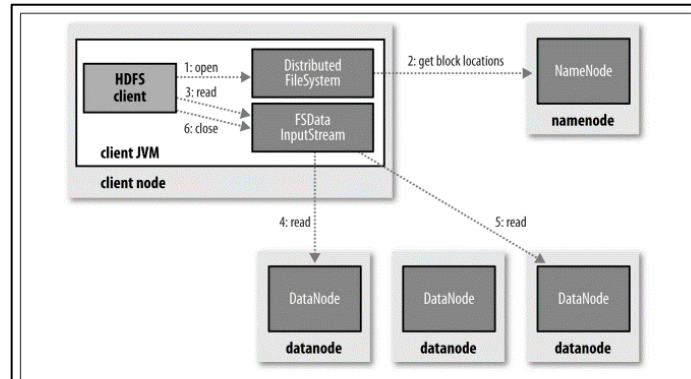


Figure 3-2. A client reading data from HDFS

18

NameNode and DataNodes

HDFS has a master/slave architecture. An HDFS cluster consists of a single NameNode, a master server that manages the file system namespace and regulates access to files by clients. In addition, there are a number of DataNodes, usually one per node in the cluster, which manage storage attached to the nodes that they run on. HDFS exposes a file system namespace and allows user data to be stored in files. Internally, a file is split into one or more blocks and these blocks are stored in a set of DataNodes. The NameNode executes file system namespace operations like opening, closing, and renaming files and directories. It also determines the mapping of blocks to DataNodes. The DataNodes are responsible for serving read and write requests from the file system's clients. The DataNodes also perform block creation, deletion, and replication upon instruction from the NameNode.

19

45. By way of further example, data blocks are associated with a file. The data block locations are stored in the NameNode's (e.g., index node) memory.

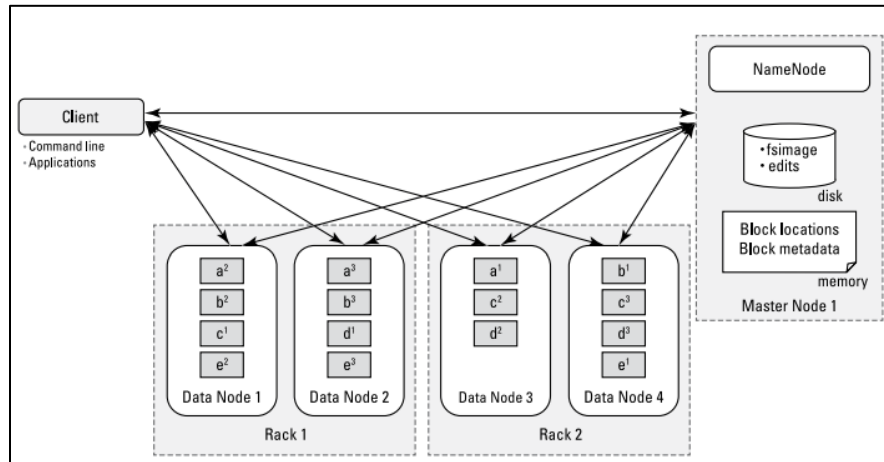
17

<https://ia800201.us.archive.org/2/items/HadoopForDummiesDirkDeRoos2014/Hadoop%20For%20Dummies%20-%20-%20Dirk%20deRoos%202014.pdf>, at Page 74 of 411

¹⁸ <https://grut-computing.com/HadoopBook.pdf>, at Page 97 of 756

¹⁹ <https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html>

After the startup process is completed, the NameNode has a complete picture of all the data stored in HDFS, and it's ready to receive application requests from Hadoop clients. As data files are added and removed based on client requests, the changes are written to the slave node's disk volumes, journal updates are made to the `edits` file, and the changes are reflected in the block locations and metadata stored in the NameNode's memory (see Figure 4-4).



Throughout the life of the cluster, the DataNode daemons send the NameNode heartbeats (a quick signal) every three seconds, indicating they're active. (This default value is configurable.) Every six hours (again, a configurable default), the DataNodes send the NameNode a block report outlining which file blocks are on their nodes. This way, the NameNode always has a current view of the available resources in the cluster.

20

46. The HDFS performs the step of obtaining mapped data block numbers after applying the mapping function, wherein the mapped data block numbers are addresses of data of the file in a storage device. For example, NameNode determines “the addresses of the data nodes” (e.g., mapped data block numbers) on which the data block is stored. Data nodes (e.g., storage device(s)) store the data blocks of a file.

The client opens the file it wishes to read by calling `open()` on the `FileSystem` object, which for HDFS is an instance of `DistributedFileSystem` (step 1 in Figure 3-2). `DistributedFileSystem` calls the namenode, using remote procedure calls (RPCs), to determine the locations of the first few blocks in the file (step 2). For each block, the namenode returns the addresses of the datanodes that have a copy of that block. Furthermore, the datanodes are sorted according to their proximity to the client (according to the topology of the cluster's network; see “Network Topology and Hadoop” on page 70). If the client is itself a datanode (in the case of a MapReduce task, for instance), the client will read from the local datanode if that datanode hosts a copy of the block (see also Figure 2-2 and “Short-circuit local reads” on page 308).

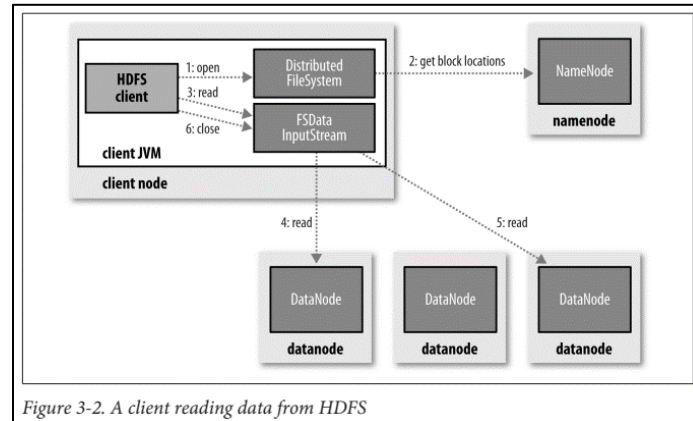


Figure 3-2. A client reading data from HDFS

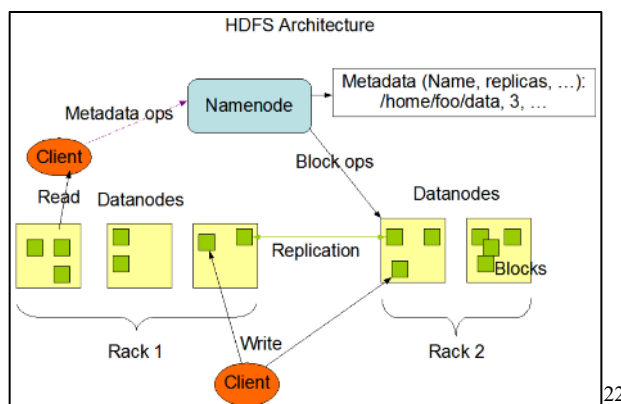
One important aspect of this design is that the client contacts datanodes directly to retrieve data and is guided by the namenode to the best datanode for each block. This design allows HDFS to scale to a large number of concurrent clients because the data traffic is spread across all the datanodes in the cluster. Meanwhile, the namenode merely has to service block location requests (which it stores in memory, making them very efficient) and does not, for example, serve data, which would quickly become a bottleneck as the number of clients grew.

21

NameNode and DataNodes

HDFS has a master/slave architecture. An HDFS cluster consists of a single NameNode, a master server that manages the file system namespace and regulates access to files by clients. In addition, there are a number of DataNodes, usually one per node in the cluster, which manage storage attached to the nodes that they run on. HDFS exposes a file system namespace and allows user data to be stored in files. Internally, a file is split into one or more blocks and these blocks are stored in a set of DataNodes. The NameNode executes file system namespace operations like opening, closing, and renaming files and directories. It also determines the mapping of blocks to DataNodes. The DataNodes are responsible for serving read and write requests from the file system's clients. The DataNodes also perform block creation, deletion, and replication upon instruction from the NameNode.

²¹ <https://grut-computing.com/HadoopBook.pdf>, at Pages 97-98 of 756



22

47. Home Depot directly infringes the '686 Patent through its implementation and use of the HDFS, performing at least the steps of claim 1.

48. Defendants have indirectly infringed and continue to indirectly infringe one or more claims of the '686 Patent by knowingly and intentionally inducing others, including Home Depot customers and end-users, to directly infringe, either literally or under the doctrine of equivalents, by making, using, offering to sell, selling, and/or importing into the United States products that include the infringing technology.

49. Defendants, with knowledge that these products, or the use thereof, infringe the '686 Patent at least as of February 8, 2024,²³ knowingly and intentionally induced, and continue to knowingly and intentionally induce, direct infringement of the '686 Patent by providing these products to end-users for use in an infringing manner. Additionally, on information and belief, Defendants have adopted a policy of not reviewing the patents of others, including specifically those related to Defendants' specific industry, thereby remaining willfully blind to the '686 Patent at least as early as the issuance of the Patents-in-Suit.

²² <https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html>

²³ Plaintiffs sent Defendants correspondence in February 2024 notifying them of their infringement of the '686 Patent. The February 2024 correspondence was preceded by multiple efforts by Plaintiffs to contact Defendants regarding at least some of the Asserted Patents dating back to April 2022. To date, Defendants have not responded to any of Plaintiffs letters or emails.

50. Defendants have induced infringement by others, including end-users, with the intent to cause infringing acts by others or, in the alternative, with the belief that there was a high probability that others, including end-users, infringe the '686 Patent, but while remaining willfully blind to the infringement. Defendants have induced, and continue to induce, infringement by their customers and end-users by supplying them with instructions on how to operate the infringing technology in an infringing manner, while also making publicly available information on the infringing technology via Defendants' website, product literature and packaging, and other publications.²⁴

51. Plaintiffs have suffered damages as a result of Defendants' direct and indirect infringement of the '686 Patent in an amount to be proven at trial.

52. Plaintiffs have suffered, and will continue to suffer, irreparable harm as a result of Defendants' infringement of the '686 Patent, for which there is no adequate remedy at law, unless Defendants' infringement is enjoined by this Court.

53. Defendants have committed and continue to commit acts of infringement that Defendants actually knew or should have known constituted an unjustifiably high risk of infringement of at least one valid and enforceable claim of the '686 Patent. Defendants' direct and indirect infringement of the '686 Patent has been and continues to be willful, intentional, deliberate, and/or in conscious disregard of rights under the patent. Plaintiffs are entitled to an award of treble damages, reasonable attorney fees, and costs in bringing this action.

COUNT II
(Infringement of the '332 Patent)

54. Paragraphs 1 through 37 are incorporated by reference as if fully set forth herein.

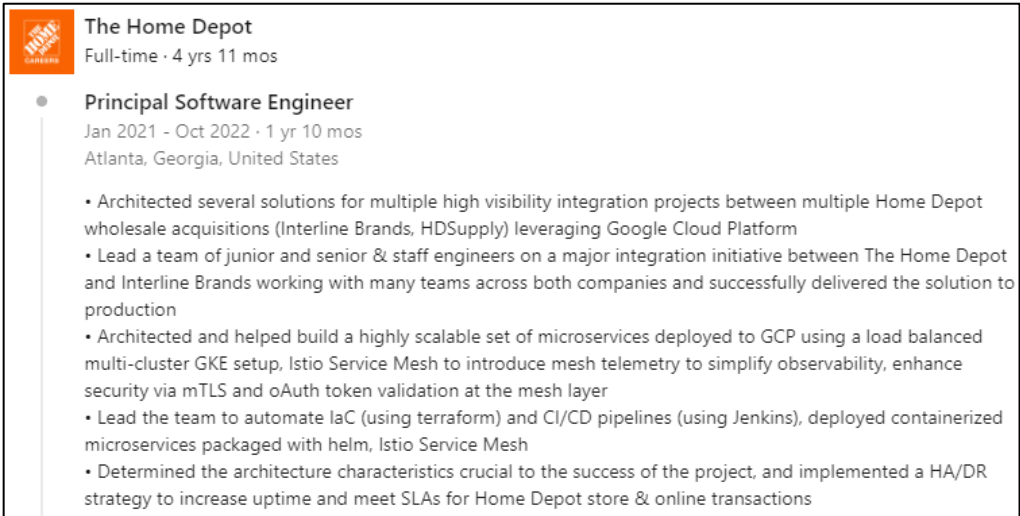
²⁴ See, e.g., https://www.homedepot.com/c/customer_service;
https://www.homedepot.com/c/SF_Mobile_Shopping

55. Plaintiffs have not licensed or otherwise authorized Defendants to make, use, offer for sale, sell, or import any products that embody the inventions of the '332 Patent.

56. Defendants have infringed, and continue to directly infringe, the '332 Patent, either literally or under the doctrine of equivalents, without authority and in violation of 35 U.S.C. § 271, by making, using, offering to sell, selling, and/or importing into the United States products that satisfy each and every limitation of one or more claims of the '332 Patent. Such products include systems using Kubernetes for rolling updates.

57. For example, Defendants have infringed, and continue to directly infringe, at least claim 1 of the '332 Patent by making, using, offering to sell, selling, and/or importing into the United States products that include systems using Kubernetes for rolling updates.

58. Upon information and belief, Home Depot has used Kubernetes (alternatively referred to as “K8s”, see kubernetes.io) since at least 2019. “GKE” is an acronym for Google Kubernetes Engine (see <https://cloud.google.com/kubernetes-engine>).



The Home Depot
Full-time · 4 yrs 11 mos

● **Principal Software Engineer**
Jan 2021 - Oct 2022 · 1 yr 10 mos
Atlanta, Georgia, United States

- Architected several solutions for multiple high visibility integration projects between multiple Home Depot wholesale acquisitions (Interline Brands, HDSupply) leveraging Google Cloud Platform
- Lead a team of junior and senior & staff engineers on a major integration initiative between The Home Depot and Interline Brands working with many teams across both companies and successfully delivered the solution to production
- Architected and helped build a highly scalable set of microservices deployed to GCP using a load balanced multi-cluster GKE setup, Istio Service Mesh to introduce mesh telemetry to simplify observability, enhance security via mTLS and OAuth token validation at the mesh layer
- Lead the team to automate IaC (using terraform) and CI/CD pipelines (using Jenkins), deployed containerized microservices packaged with helm, Istio Service Mesh
- Determined the architecture characteristics crucial to the success of the project, and implemented a HA/DR strategy to increase uptime and meet SLAs for Home Depot store & online transactions

²⁵ <https://www.linkedin.com/in/ehsan-mirhosseini-0536478/> (“Architected and helped build a highly scalable set of microservices deployed to GCP using a load balanced multi-cluster GKE setup.”)



Technology Specialist - Platform Engineering

The Home Depot · Full-time

Dec 2020 - Present · 3 yrs 11 mos

Florida, United States

Working as Technologist at Home Depot "Enterprise Search as Service" team.

Design, development, test and continuous integration/deployment of micronaut based REST API. following TDD and BDD design principles for quality control.

Re engineering current microservices for high performance, high availability and resilient under GCP using various open source technology stack. Implementing DevSecOps using Concourse pipeline. Configuration management using github and vault. Experience with Solr and Elasticsearch engines indexing. Application monitoring using kibana, grafana dashboards. Continuous deployment of applications into GCP VM's using pipeline and diverting traffic post cucumber based regression tests. CD of K8S applications using Spinnaker. Using Terraform to create K8S, Elastic Deployment cluster. Mentoring junior developers and collaboration with different teams.

26



Java Full Stack Developer

The Home Depot · Contract

Jul 2019 - Sep 2021 · 2 yrs 3 mos

Virginia, United States · Remote

- Involved in reviewing business requirements and functional designs for application and participated in creating Wire Frames.
- Involved in Agile process, bi-weekly Sprints, and daily Scrums to discuss the development and achieve TDD (Test driven development approach).
- Worked with Angular 7/6 Flex Layout that provides a sophisticated layout API using Flexbox CSS and media Query. This module component layout features using a custom Layout API, media Query observables, and can be injected to DOM.
- Developing single page application by using Covalent open-source framework, which is built on Angular 7/6, Angular-cli, Typescript, Material design, Karma, Jasmine, d3, NGX, SCSS.
- Used Angular 7/6 and Confidential Covalent UI as framework to create a Single Page Application (SPA) which can bind data to specific views and synchronize data with server.
- Involved in implementing the complete Application in the in-build MVC Architecture with Angular 7/6.
- Implemented Microservices Architecture with Spring Boot based RESTful services and consumed SOAP based Web Services.
- Configure backup, alerts, repairs and monitoring of Cassandra clusters using Opscenter.
- Administered, monitored and maintained multi data-center Cassandra cluster using OpsCenter and Nagios in production.
- Used MongoDB internal tools like Mongo Compass, Mongo Atlas Manager & Ops Manager, Cloud Manager etc.
- Worked on Big Data Integration & Analytics based on Hadoop, SOLR, Spark, Kafka, Storm and web Methods.
- Worked on analyzing Hadoop cluster using different big data analytic tools including Flume, Pig, Hive, HBase, Oozie, Zookeeper, Sqoop, Spark and Kafka.
- Deploying and maintaining Micro services using Docker and experienced on Docker, AWS ECS, Kubernetes
- Lead daily stand-ups and scrum ceremonies for two scrum teams.

27

²⁶ <https://www.linkedin.com/in/rajpal-marpu-1b398b52/>

²⁷ <https://www.linkedin.com/in/shiva-krishna-thota/>



Cloud Infrastructure Engineer

The Home Depot

Jul 2022 - Present · 11 mos

Remote

Skills: Apache Kafka · Docker · Gitlab · Ansible · Maven · Python (Programming Language) · Kafka Streams · Terraform · Azure Kubernetes Service (AKS) · Microsoft Azure · Amazon Web Services (AWS) · Jenkins · Kubernetes

59. Kubernetes performs a method for hot deployment and/or redeployment in a grid computing environment, wherein the grid computing environment includes one or more grid nodes. For example, Kubernetes uses a method of rolling updates to allow for hot deployment and redeployment (rollouts and rollbacks). Using rolling updates, application containers (e.g., pods) are updated and scheduled on worker nodes of the cluster.

Kubernetes progressively rolls out changes to your application or its configuration, while monitoring application health to ensure it doesn't kill all your instances at the same time. If something goes wrong, Kubernetes will rollback the change for you. Take advantage of a growing ecosystem of deployment solutions.

When you deploy Kubernetes, you get a cluster.

A Kubernetes cluster consists of a set of worker machines, called nodes, that run containerized applications. Every cluster has at least one worker node.

Users expect applications to be available all the time and developers are expected to deploy new versions of them several times a day. In Kubernetes this is done with rolling updates. **Rolling updates** allow Deployments' update to take place with zero downtime by incrementally updating Pods instances with new ones. The new Pods will be scheduled on Nodes with available resources.

²⁸ <https://www.linkedin.com/in/pavan-k-devops-engineer/>

²⁹ <https://kubernetes.io/>

³⁰ <https://kubernetes.io/docs/concepts/overview/components/>

³¹ <https://kubernetes.io/docs/tutorials/kubernetes-basics/update/update-intro/>

60. Kubernetes performs the step of adding a new version of an application release bundle in a repository server. For example, container images are software bundles stored in a registry. To roll out a new version of software, a container image with the new version is formed.

Once you have a running Kubernetes cluster, you can deploy your containerized applications on top of it. To do so, you create a Kubernetes **Deployment** configuration. The Deployment instructs Kubernetes how to create and update instances of your application. Once you've created a Deployment, the Kubernetes control plane schedules the application instances included in that Deployment to run on individual Nodes in the cluster.

When you create a Deployment, you'll need to specify the container image for your application and the number of replicas that you want to run. You can change that information later by updating your Deployment; Modules 5 and 6 of the bootcamp discuss how you can scale and update your Deployments.

32

A container image represents binary data that encapsulates an application and all its software dependencies. Container images are executable software bundles that can run standalone and that make very well defined assumptions about their runtime environment.

You typically create a container image of your application and push it to a registry before referring to it in a Pod.

If you don't specify a registry hostname, Kubernetes assumes that you mean the Docker public registry.

33

61. Kubernetes performs the step of determining by a discovery services module which of the one or more grid nodes are running an application associated with the added new version of the application release bundle upon adding the new version of the application release bundle in the repository server. For example, a control plane has a deployment controller that ensures that the new container image is deployed all across the ReplicaSet of the cluster.

³² <https://kubernetes.io/docs/tutorials/kubernetes-basics/deploy-app/deploy-intro/>

³³ <https://kubernetes.io/docs/concepts/containers/images/>

Users expect applications to be available all the time and developers are expected to deploy new versions of them several times a day. In Kubernetes this is done with rolling updates. **Rolling updates** allow Deployments' update to take place with zero downtime by incrementally updating Pods instances with new ones. The new Pods will be scheduled on Nodes with available resources.

Initially, the pods run the first version of your application—let's suppose its image is tagged as v1. You then develop a newer version of the app and push it to an image repository as a new image, tagged as v2. You'd next like to replace all the pods with this new version. Because you can't change an existing pod's image after the pod is created, you need to remove the old pods and replace them with new ones running the new image.

34

A *Deployment* provides declarative updates for Pods and ReplicaSets.

Each time a new Deployment is observed by the Deployment controller, a ReplicaSet is created to bring up the desired Pods. If the Deployment is updated, the existing ReplicaSet that controls Pods whose labels match `.spec.selector` but whose template does not match `.spec.template` are scaled down. Eventually, the new ReplicaSet is scaled to `.spec.replicas` and all old ReplicaSets is scaled to 0.

35

Once you have a [running Kubernetes cluster](#), you can deploy your containerized applications on top of it. To do so, you create a Kubernetes **Deployment**. The Deployment instructs Kubernetes how to create and update instances of your application. Once you've created a Deployment, the Kubernetes control plane schedules the application instances included in that Deployment to run on individual Nodes in the cluster.

Once the application instances are created, a Kubernetes Deployment controller continuously monitors those instances. If the Node hosting an instance goes down or is deleted, the Deployment controller replaces the instance with an instance on another Node in the cluster. **This provides a self-healing mechanism to address machine failure or maintenance.**

36

62. Kubernetes performs the step of notifying a client application manager associated with one or more of the determined grid nodes about adding the new version of the application release bundle along with a type of data transfer protocol to use. The deployment controller

³⁴ <https://kubernetes.io/docs/tutorials/kubernetes-basics/update/update-intro/>

³⁵ <https://kubernetes.io/docs/concepts/workloads/controllers/deployment/>

³⁶ <https://kubernetes.io/docs/tutorials/kubernetes-basics/deploy-app/deploy-intro/>

notifies Kubelet processes in worker nodes. Each Kubelet process is, therefore, notified of the desired configuration, including the new application and associated protocol information.

A *Deployment* provides declarative updates for *Pods* and *ReplicaSets*.

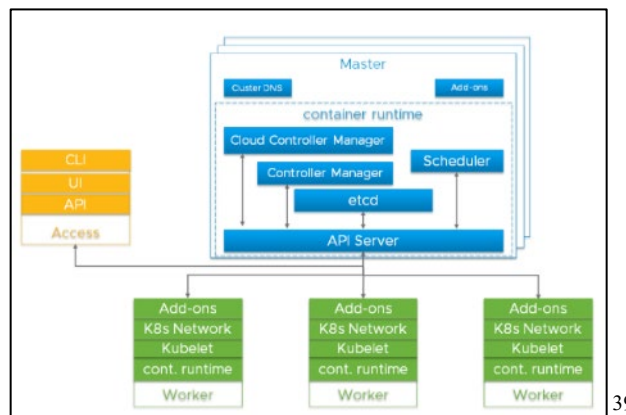
Each time a new *Deployment* is observed by the *Deployment* controller, a *ReplicaSet* is created to bring up the desired *Pods*. If the *Deployment* is updated, the existing *ReplicaSet* that controls *Pods* whose labels match `.spec.selector` but whose template does not match `.spec.template` are scaled down. Eventually, the new *ReplicaSet* is scaled to `.spec.replicas` and all old *ReplicaSets* is scaled to 0.

37

An agent that runs on each node in the cluster. It makes sure that containers are running in a *Pod*.

The kubelet takes a set of *PodSpecs* that are provided through various mechanisms and ensures that the containers described in those *PodSpecs* are running and healthy. The kubelet doesn't manage containers which were not created by Kubernetes.

38



39

³⁷ <https://kubernetes.io/docs/concepts/workloads/controllers/deployment/>

³⁸ <https://kubernetes.io/docs/concepts/overview/components/#kubelet>

³⁹ <https://kubernetes.io/blog/2018/08/03/out-of-the-clouds-onto-the-ground-how-to-make-kubernetes-production-grade-anywhere/>

```

Name:          nginx-deployment
Namespace:     default
CreationTimestamp: Thu, 30 Nov 2017 10:56:25 +0000
Labels:       app=nginx
Annotations:  deployment.kubernetes.io/revision=2
Selector:     app=nginx
Replicas:     3 desired | 3 updated | 3 total | 3 available
StrategyType: RollingUpdate
MinReadySeconds: 0
RollingUpdateStrategy: 25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=nginx
  Containers:
    nginx:
      Image:   nginx:1.16.1
      Port:    80/TCP
      Environment: <none>
      Mounts:    <none>
      Volumes:   <none>

```

```

Conditions:
  Type           Status  Reason
  ----           -
  Available      True    MinimumReplicasAvailable
  Progressing    True    NewReplicaSetAvailable
OldReplicaSets: <none>
NewReplicaSet:  nginx-deployment-1564180365 (3/3 replicas created)

```

```

Events:
  Type     Reason             Age   From                      Message
  ----     -
  Normal   ScalingReplicaSet  2m    deployment-controller     Scaled up replica set nginx-
  Normal   ScalingReplicaSet  24s    deployment-controller     Scaled up replica set nginx-
  Normal   ScalingReplicaSet  22s    deployment-controller     Scaled down replica set ngin
  Normal   ScalingReplicaSet  22s    deployment-controller     Scaled up replica set nginx-
  Normal   ScalingReplicaSet  19s    deployment-controller     Scaled down replica set ngin
  Normal   ScalingReplicaSet  19s    deployment-controller     Scaled up replica set nginx-
  Normal   ScalingReplicaSet  14s    deployment-controller     Scaled down replica set ngin

```

63. Kubernetes performs the step of hot deploying/redeploying the new version of the application release bundle running on one or more application servers in an associated grid node using an appropriate hot deployment plug-in based on the data transfer protocol by a respective one of the client application managers. For example, rolling updates of the new software are made with the rollout command of the Kubelet interface. The image is deployed in the worker nodes according to the desired configuration by the Kubelet process.

⁴⁰ <https://kubernetes.io/docs/concepts/workloads/controllers/deployment/>

Users expect applications to be available all the time and developers are expected to deploy new versions of them several times a day. In Kubernetes this is done with rolling updates. **Rolling updates** allow Deployments' update to take place with zero downtime by incrementally updating Pods instances with new ones. The new Pods will be scheduled on Nodes with available resources.

41

A Deployment provides declarative updates for Pods and ReplicaSets.

You describe a *desired state* in a Deployment, and the Deployment Controller changes the actual state to the desired state at a controlled rate. You can define Deployments to create new ReplicaSets, or to remove existing Deployments and adopt all their resources with new Deployments.

- **Declare the new state of the Pods** by updating the PodTemplateSpec of the Deployment. A new ReplicaSet is created and the Deployment manages moving the Pods from the old ReplicaSet to the new one at a controlled rate. Each new ReplicaSet updates the revision of the Deployment.

- The `template` field contains the following sub-fields:

- The Pods are labeled `app: nginx` using the `.metadata.labels` field.
- The Pod template's specification, or `.template.spec` field, indicates that the Pods run one container, `nginx`, which runs the `nginx` Docker Hub image at version 1.14.2.
- Create one container and name it `nginx` using the `.spec.template.spec.containers[0].name` field.

42

```
Name:          nginx-deployment
Namespace:     default
CreationTimestamp: Thu, 30 Nov 2017 10:56:25 +0000
Labels:        app=nginx
Annotations:   deployment.kubernetes.io/revision=2
Selector:      app=nginx
Replicas:      3 desired | 3 updated | 3 total | 3 available
StrategyType:  RollingUpdate
MinReadySeconds: 0
RollingUpdateStrategy: 25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=nginx
  Containers:
    nginx:
      Image:      nginx:1.16.1
      Port:       80/TCP
      Environment: <none>
      Mounts:      <none>
      Volumes:     <none>
```

⁴¹ <https://kubernetes.io/docs/tutorials/kubernetes-basics/update/update-intro/>

⁴² <https://kubernetes.io/docs/concepts/workloads/controllers/deployment/>

Conditions:		
Type	Status	Reason
----	-----	-----
Available	True	MinimumReplicasAvailable
Progressing	True	NewReplicaSetAvailable
OldReplicaSets:	<none>	
NewReplicaSet:	nginx-deployment-1564180365 (3/3 replicas created)	

Events:				
Type	Reason	Age	From	Message
-----	-----	-----	-----	-----
Normal	ScalingReplicaSet	2m	deployment-controller	Scaled up replica set nginx-
Normal	ScalingReplicaSet	24s	deployment-controller	Scaled up replica set nginx-
Normal	ScalingReplicaSet	22s	deployment-controller	Scaled down replica set nginx-
Normal	ScalingReplicaSet	22s	deployment-controller	Scaled up replica set nginx-
Normal	ScalingReplicaSet	19s	deployment-controller	Scaled down replica set nginx-

43

64. Home Depot directly infringes the '332 Patent through its implementation and use of Kubernetes, performing at least the steps of claim 1.

65. Defendants have indirectly infringed and continue to indirectly infringe one or more claims of the '332 Patent by knowingly and intentionally inducing others, including Home Depot customers and end-users, to directly infringe, either literally or under the doctrine of equivalents, by making, using, offering to sell, selling, and/or importing into the United States products that include the infringing technology.

66. Defendants, with knowledge that these products, or the use thereof, infringe the '332 Patent at least as of February 8, 2024,⁴⁴ knowingly and intentionally induced, and continue to knowingly and intentionally induce, direct infringement of the '332 Patent by providing these products to end-users for use in an infringing manner. Additionally, on information and belief, Defendants have adopted a policy of not reviewing the patents of others, including specifically those related to Defendants' specific industry, thereby remaining willfully blind to the '332 Patent at least as early as the issuance of the '332 Patent.

⁴³ <https://kubernetes.io/docs/concepts/workloads/controllers/deployment/>

⁴⁴ Plaintiffs sent Defendants correspondence in February 2024 notifying them of their infringement of the '332 Patent. The February 2024 correspondence was preceded by multiple efforts by Plaintiffs to contact Defendants regarding at least some of the Asserted Patents dating back to February 2024. To date, Defendants have not responded to any of Plaintiffs letters or emails.

67. Defendants have induced infringement by others, including end-users, with the intent to cause infringing acts by others or, in the alternative, with the belief that there was a high probability that others, including end-users, infringe the '332 Patent, but while remaining willfully blind to the infringement. Defendants have and continue to induce infringement by their customers and end-users by supplying them with instructions on how to operate the infringing technology in an infringing manner, while also making publicly available information on the infringing technology via Defendants' website, product literature and packaging, and other publications.⁴⁵

68. Plaintiffs have suffered damages as a result of Defendants' direct and indirect infringement of the '332 Patent in an amount to be proven at trial.

69. Plaintiffs have suffered, and will continue to suffer, irreparable harm as a result of Defendants' infringement of the '332 Patent, for which there is no adequate remedy at law, unless Defendants' infringement is enjoined by this Court.

70. Defendants have committed and continue to commit acts of infringement that Defendants actually knew or should have known constituted an unjustifiably high risk of infringement of at least one valid and enforceable claim of the '332 Patent. Defendants' direct and indirect infringement of the '332 Patent has been and continues to be willful, intentional, deliberate, and/or in conscious disregard of rights under the patent. Plaintiffs are entitled to an award of treble damages, reasonable attorney fees, and costs in bringing this action.

COUNT III
(Infringement of the '597 Patent)

71. Paragraphs 1 through 37 are incorporated by reference as if fully set forth herein.

72. Plaintiffs have not licensed or otherwise authorized Defendants to make, use, offer

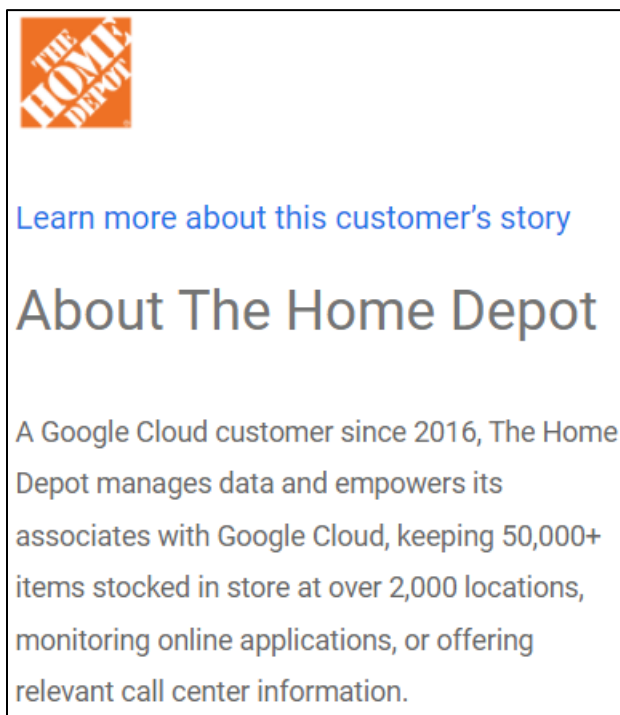
⁴⁵ See, e.g., https://www.homedepot.com/c/customer_service;
https://www.homedepot.com/c/SF_Mobile_Shopping

for sale, sell, or import any products that embody the inventions of the '597 Patent.

73. Defendants have directly infringed the '597 Patent, either literally or under the doctrine of equivalents, without authority and in violation of 35 U.S.C. § 271, by making, using, offering to sell, selling, and/or importing into the United States products that satisfy each and every limitation of one or more claims of the '597 Patent. Such products include the Google Cloud Platform ("GCP"), such as through IAM for Cloud Load Balancing, Backend service-based external TCP/UDP Network Load Balancing, and External HTTP(S) Load Balancing.

74. For example, Defendants have directly infringed at least claim 1 of the '597 Patent by making, using, offering to sell, selling, and/or importing into the United States products that include the GCP, such as through IAM for Cloud Load Balancing, Backend service-based external TCP/UDP Network Load Balancing, and External HTTP(S) Load Balancing.

75. Upon information and belief, Home Depot has used the GCP since at least 2016.



[The Home Depot](#) (THD) is the world's largest home-improvement chain, growing to more than 2,200 stores and 700,000 products in four decades. Much of that success was driven through the analysis of data. This included developing sales forecasts, replenishing inventory through the supply chain network, and providing timely performance scorecards.

However, to compete in today's business world, THD has taken this data-driven approach to an entirely new level of success on [Google Cloud](#), providing capabilities not practical on legacy technologies.

The pressures of contemporary growth that drove much of the work are familiar to many businesses. In addition to everything it was doing, THD needed to better integrate the complexities in its related businesses, like tool rental and home services. It needed to better empower teams, including a fast-growing data analysis staff and store associates with mobile computing devices. It wanted to better use online commerce and artificial intelligence to meet customer needs, while maintaining better security.

46

**The Home Depot**
Full-time · 10 yrs 8 mos

- **Technology Director - Data Management & Analytics**
Jun 2015 - Present · 6 yrs 10 mos
Greater Atlanta Area

Responsible for the data engineering and analytic development for The Home Depot enterprise including finance, merchandising, supply chain, online, marketing, pro, services & store operations.

Led data engineering team responsible for the migration of on-premise data warehouse to Google bigquery. Migration included developing batch and streaming data pipelines to move 5+ TB data daily, integrating 10+ PB of data consumable to analyst community in bigquery, migrating multiple reporting and analytics products to the Google Cloud Platform, and enabling the migration of 1000+ data analysts' work to bigquery. Migration completed summer of 2019 with the full retirement of the on-premise data warehouse.

47

⁴⁶ <https://cloud.google.com/customers/the-home-depot/>

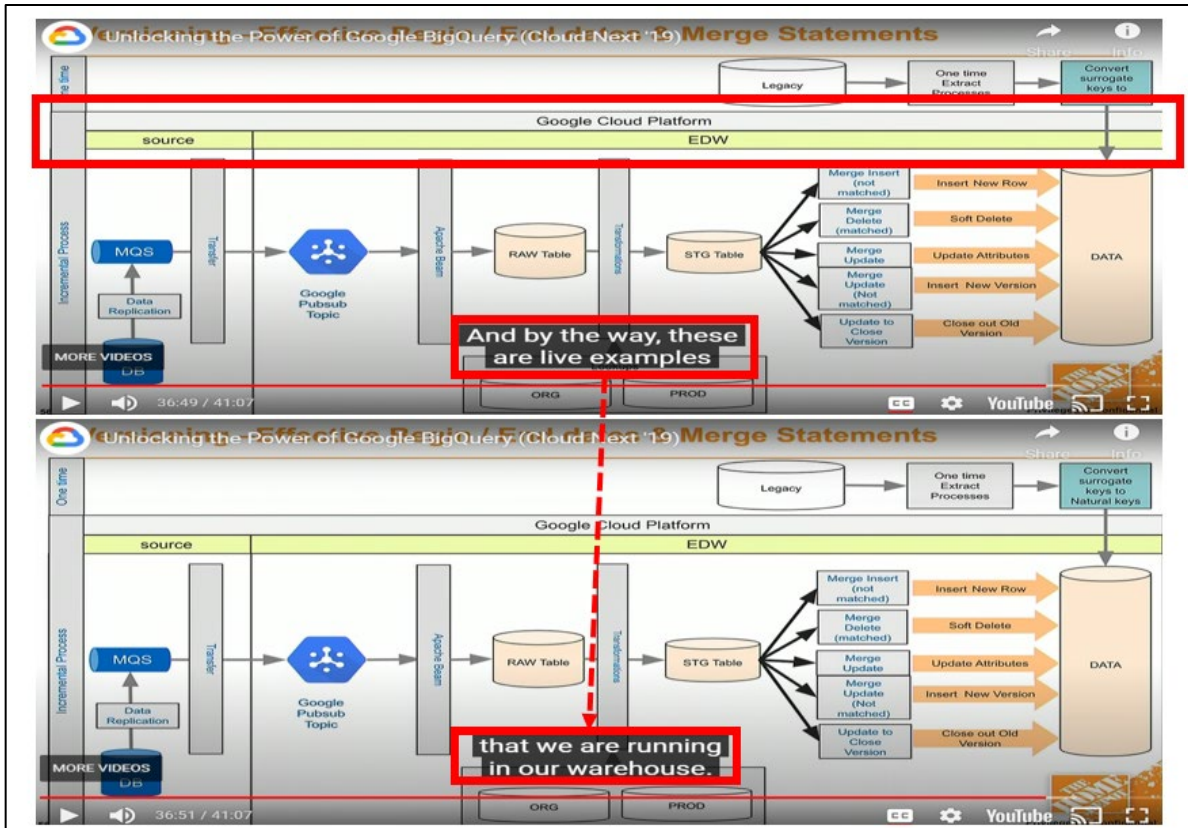
⁴⁷ <https://www.linkedin.com/in/rick-ramaker-b5b5876/>



A YouTube video player interface showing a slide titled "Performance wins". The slide contains a table with the following data:

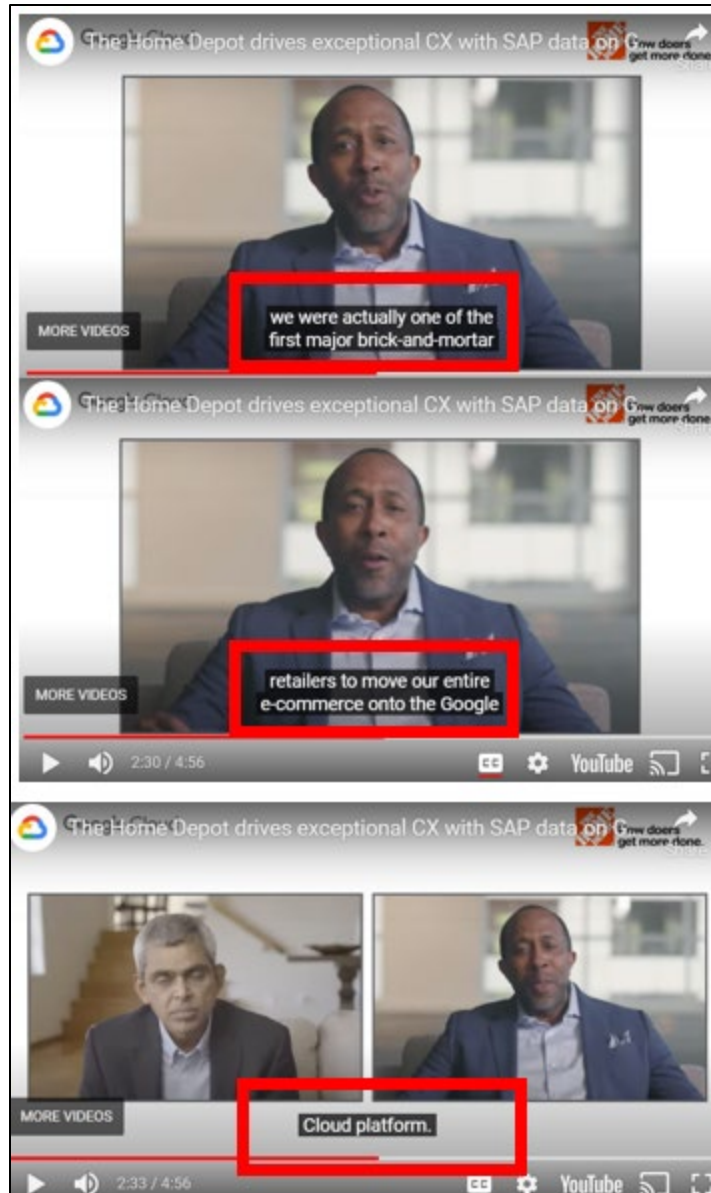
	Before Execution Time	BigQuery Time	Reduction %
Analytical workload 1	8 hours	5 minutes	99%
Analytical workload 2	12 hours	7 minutes	99%
Analytical workload 3	9 hours	12 minutes	98%

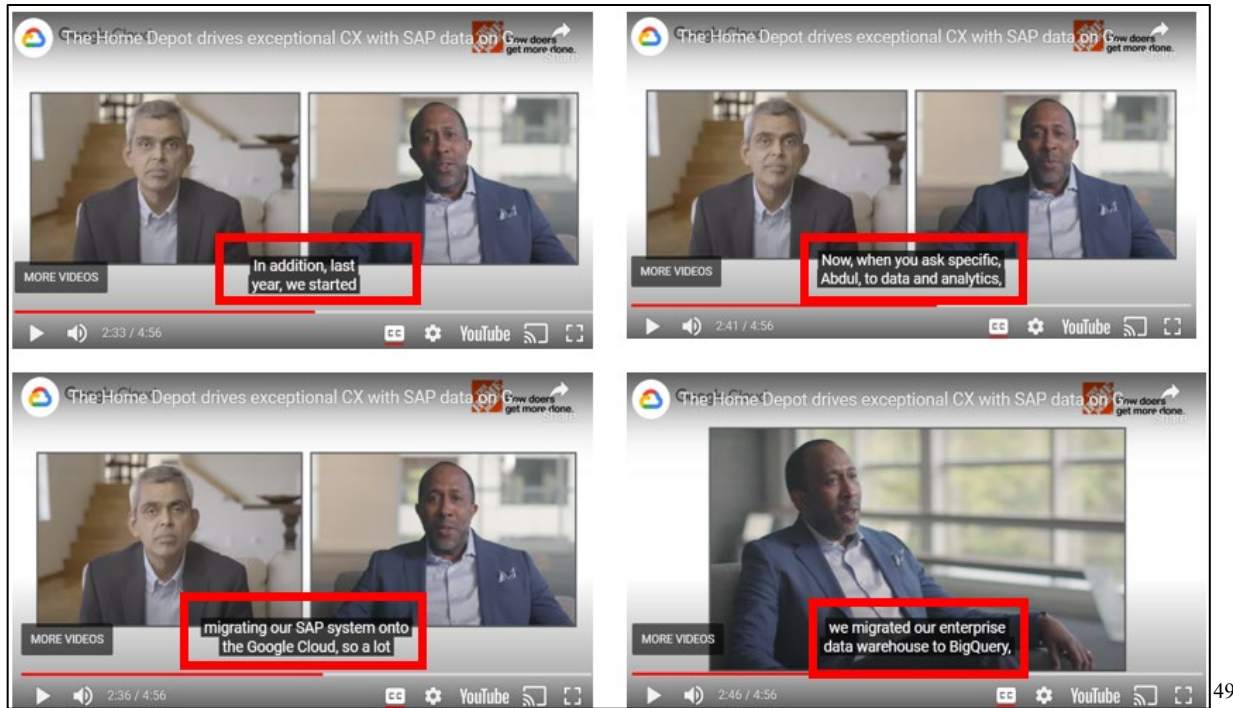
Below the table, the text reads: "Able to execute workloads that would not complete on legacy platform". The video player shows a progress bar at 37:53 / 46:32 and the word "Performance wins". The NEXTON logo is visible at the bottom.



48

⁴⁸ <https://cloud.google.com/customers/featured/the-home-depot>





49

76. Upon further information and belief, the GCP includes IAM for Cloud Load Balancing, Backend service-based external TCP/UDP Network Load Balancing, and External HTTP(S) Load Balancing.

Backend service-based external TCP/UDP
Network Load Balancing overview

⁴⁹ <https://cloud.google.com/blog/products/sap-google-cloud/how-the-home-depot-migrated-to-sap-on-google-cloud>

Google Cloud external TCP/UDP Network Load Balancing (after this referred to as Network Load Balancing) is a regional, pass-through load balancer. A network load balancer distributes TCP or UDP traffic among virtual machine (VM) instances in the same region.

A network load balancer can receive traffic from:

- Any client on the internet
- Google Cloud VMs with external IPs
- Google Cloud VMs that have internet access through Cloud NAT or instance-based NAT

Network Load Balancing has the following characteristics:

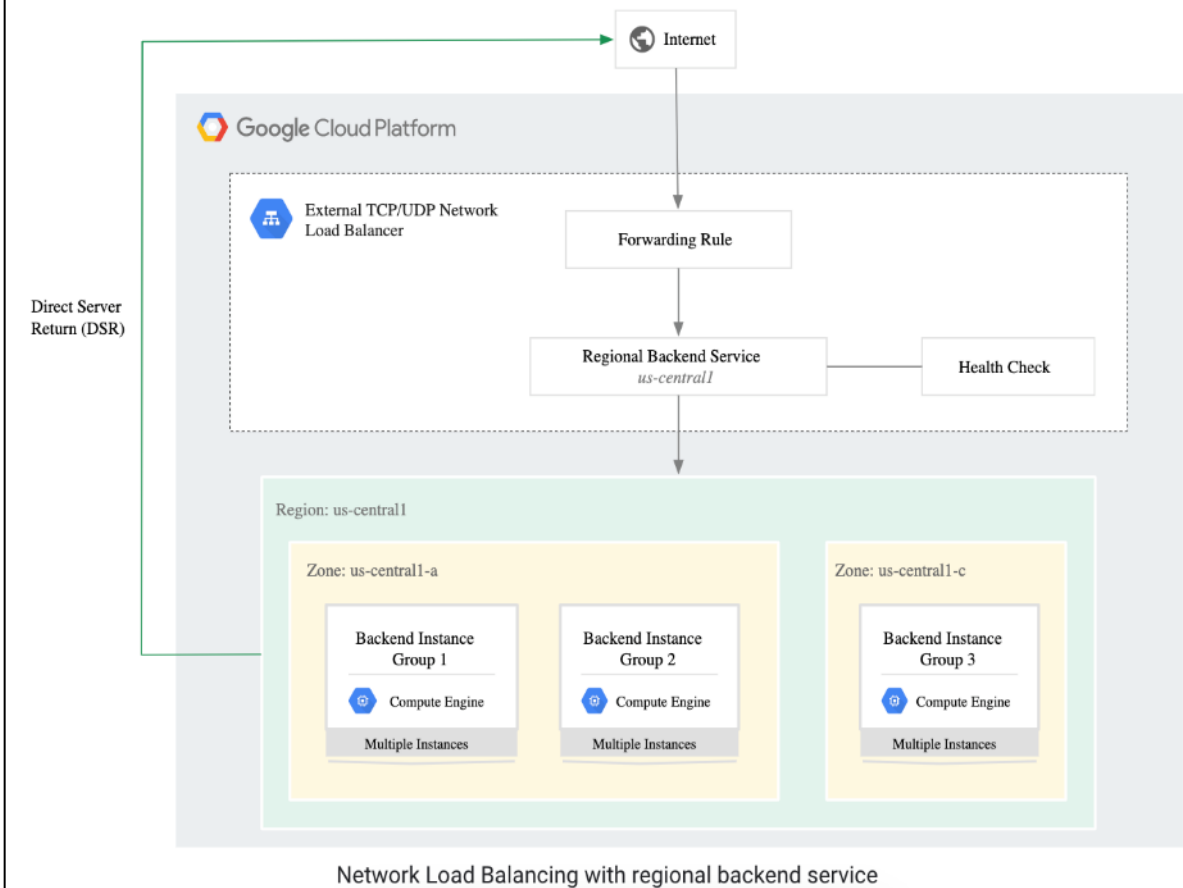
- Network Load Balancing is a managed service.
- Network Load Balancing is implemented by using [Andromeda virtual networking](#) and [Google Maglev](#).
- The network load balancers are not proxies.
 - Load-balanced packets are received by backend VMs with their source IP unchanged.
 - Load-balanced connections are terminated by the backend VMs.
 - Responses from the backend VMs go directly to the clients, not back through the load balancer. The industry term for this is *direct server return*.

This page describes how Network Load Balancing works with a *backend service* instead of a target pool. Previously, the only option for a network load balancer was the [target-pool based network load balancer](#). Compared to target pools, backend services give you more fine-grained control over how your load balancer behaves.

- A backend service configuration contains a set of values, such as health checks, session affinity, connection draining timeouts, and failover policies. Most of these settings have default values that allow for easy configuration to get you started quickly.
- Backend service-based network load balancers use health checks that match the type of traffic (TCP, SSL, HTTP, HTTPS, or HTTP/2) they are distributing. Target pool-based load balancers only support legacy HTTP health checks.
- Backend service-based network load balancers support using managed instance groups as backends. Managed instance groups automate certain aspects of backend management and provide better scalability and reliability as compared to unmanaged instance groups. Target pool-based network load balancers support only unmanaged instance groups.

Architecture

The following diagram illustrates the components of a network load balancer:



50

External HTTP(S) Load Balancing overview

[Send feedback](#)

This document introduces the concepts that you need to understand to configure Google Cloud external HTTP(S) Load Balancing.

Google Cloud HTTP(S) Load Balancing is a global, proxy-based Layer 7 load balancer that enables you to run and scale your services worldwide behind a single external IP address. External HTTP(S) Load Balancing distributes HTTP and HTTPS traffic to backends hosted on Compute Engine and Google Kubernetes Engine (GKE).

External HTTP(S) Load Balancing is implemented on [Google Front Ends \(GFEs\)](#). GFEs are distributed globally and operate together using Google's global network and control plane. In the Premium Tier, GFEs offer cross-regional load balancing, directing traffic to the closest healthy backend that has capacity and terminating HTTP(S) traffic as close as possible to your users. With Standard Tier, the load balancing is handled regionally.

⁵⁰ <https://cloud.google.com/load-balancing/docs/network/networklb-backend-service>

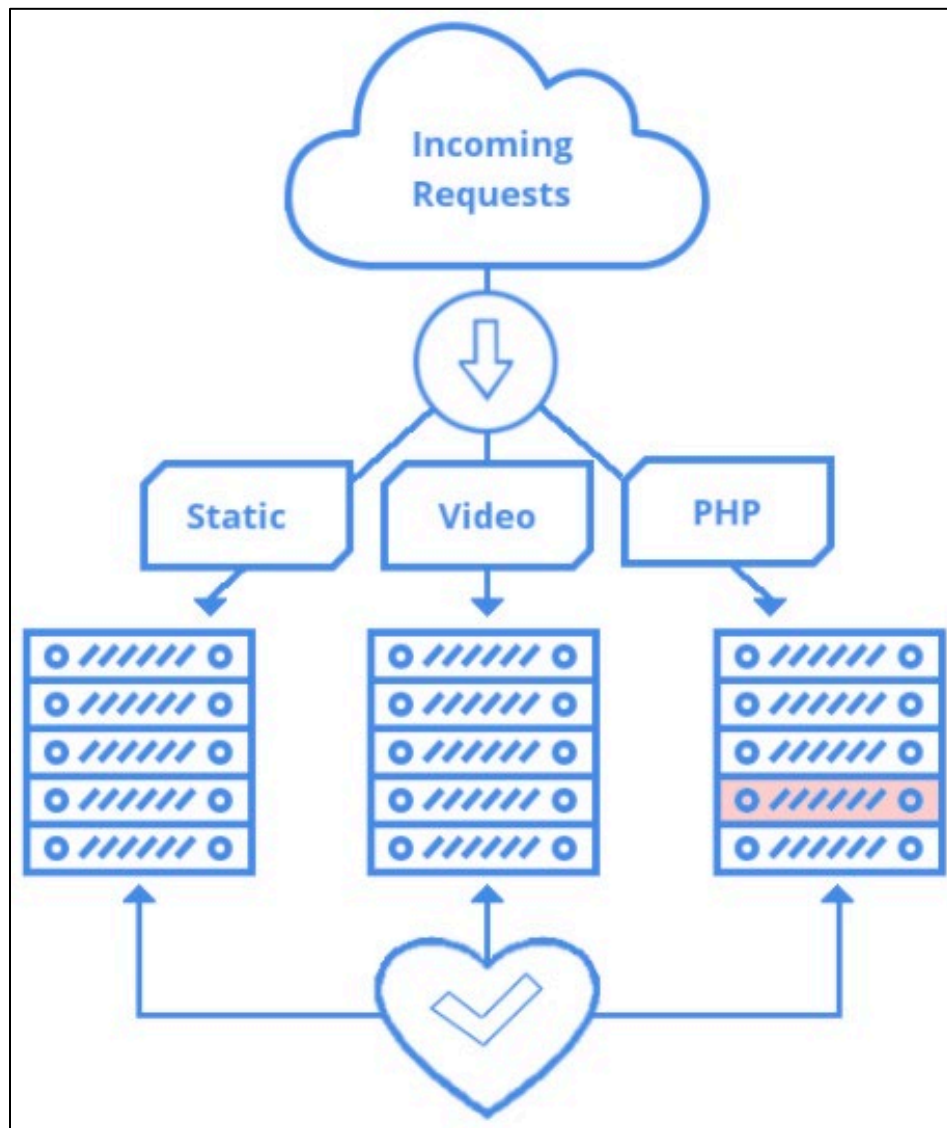
Use cases

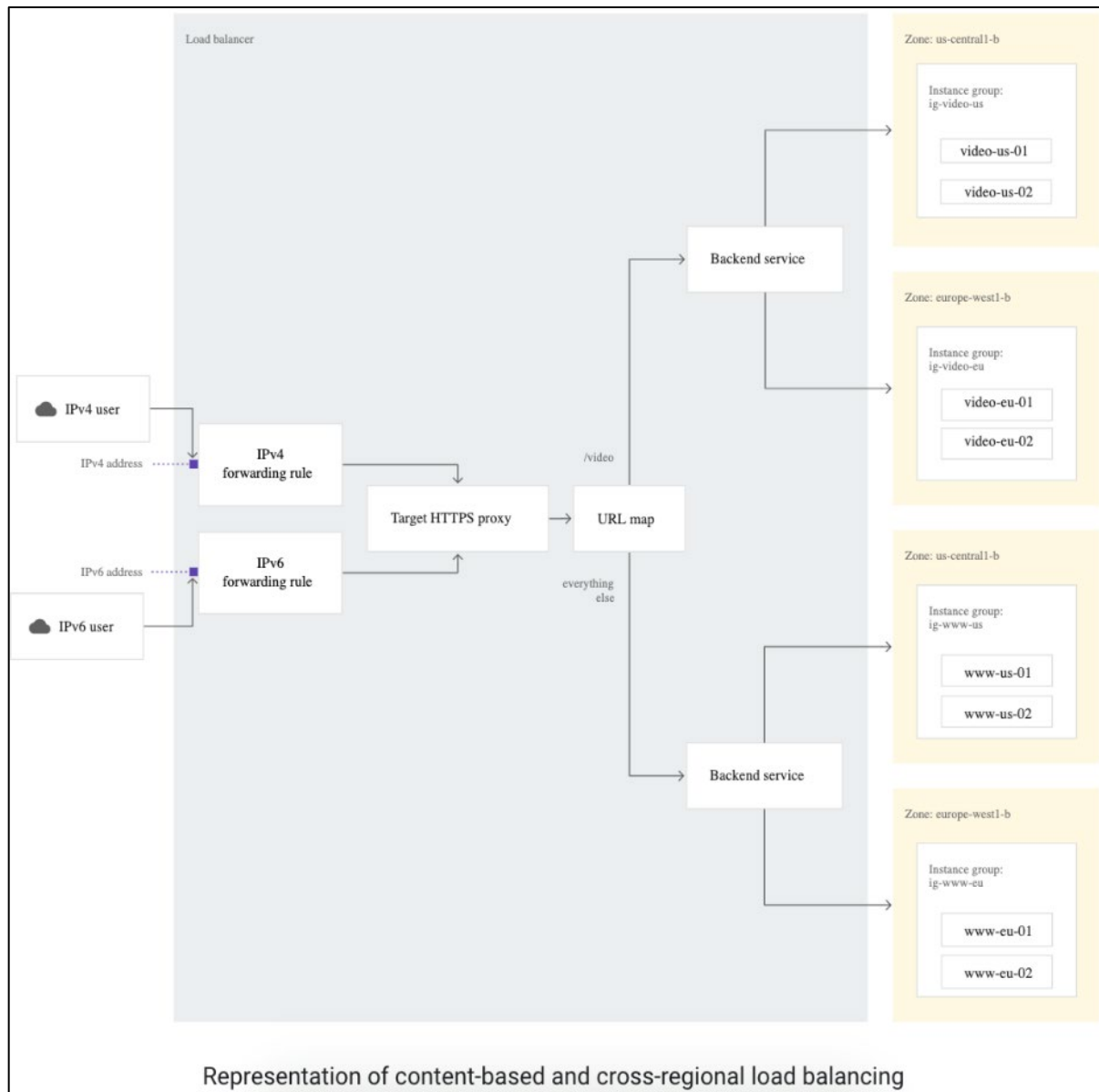
The external HTTP(S) load balancers address many use cases. This section provides some high-level examples.

Three-tier web services

You can use external HTTP(S) Load Balancing to support traditional three-tier web services. The following example shows how you can use three types of Google Cloud load balancers to scale three tiers. At each tier, the load balancer type depends on your traffic type:

- **Web tier:** Traffic enters from the internet and is load balanced by using an [external HTTP\(S\) load balancer](#).
- **Application tier:** The application tier is scaled by using a regional internal HTTP(S) load balancer.
- **Database tier:** The database tier is scaled by using an [internal TCP/UDP load balancer](#).





51

External TCP/UDP Network Load Balancing

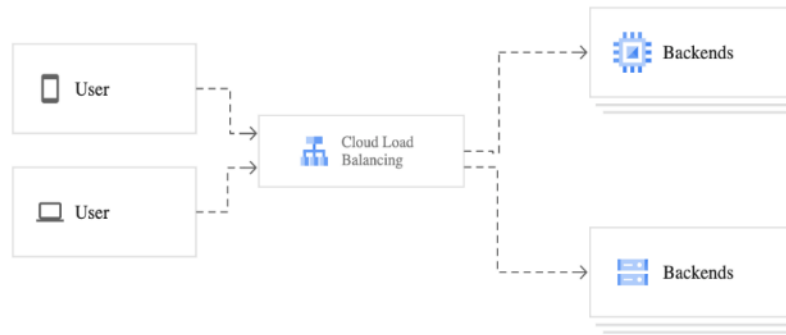
77. The GCP performs a method for dynamic load balancing of requests in a network system. For example, Google Cloud Load Balancing executes backend service-based external TCP/UDP Network Load Balancing.

⁵¹ <https://cloud.google.com/load-balancing/docs/https>

Cloud Load Balancing overview

[Send feedback](#)

A load balancer distributes user traffic across multiple instances of your applications. By spreading the load, load balancing reduces the risk that your applications experience performance issues.



Simple overview of load balancing (click to enlarge)

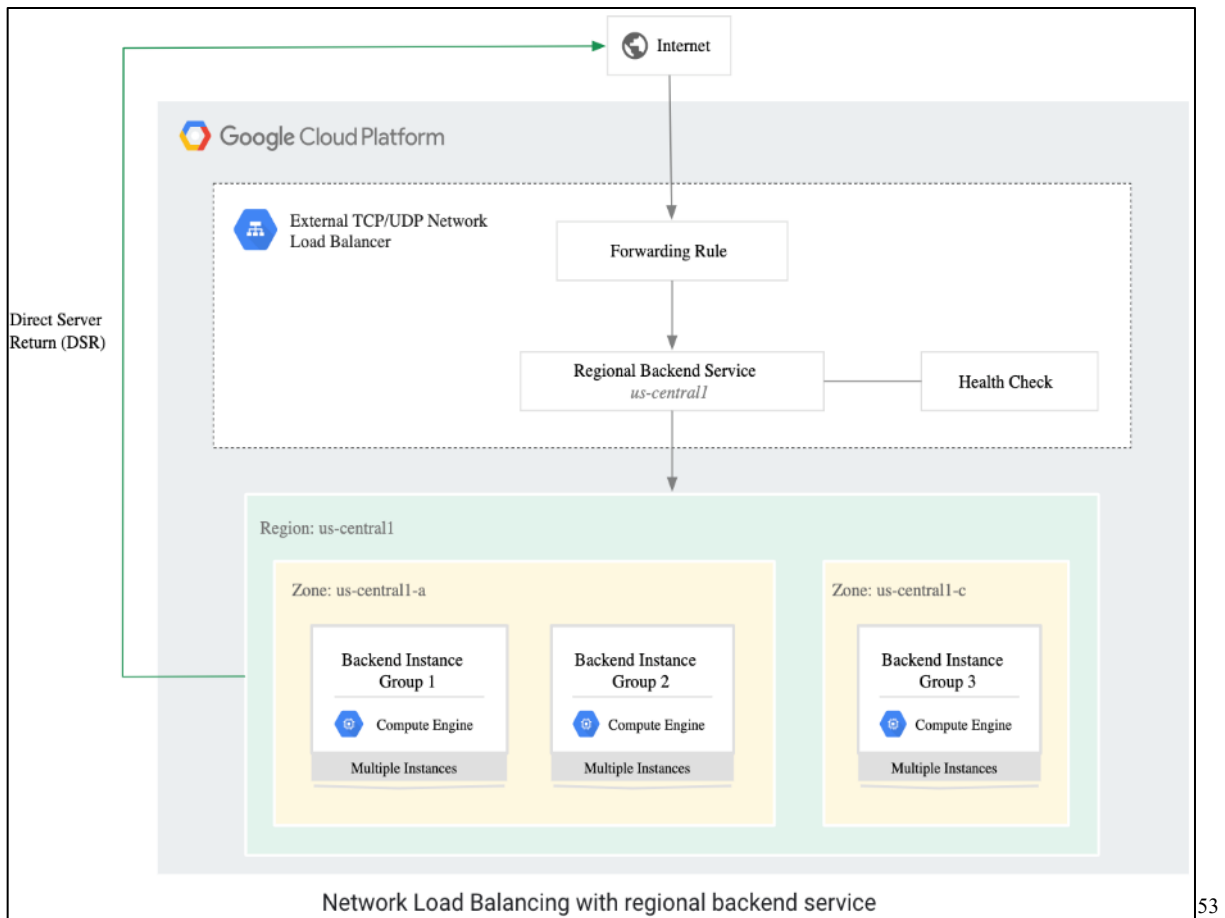
About Cloud Load Balancing

By using Cloud Load Balancing, you can serve content as close as possible to your users on a system that can respond to over one million queries per second.

52

78. The GCP performs the step of receiving a first message on a first network device from a second network device. For example, the network load balancer receives an initial request from a user/client device.

⁵² <https://cloud.google.com/load-balancing/docs/load-balancing-overview>



53

79. The GCP performs the step of receiving a first message on a first network device from a second network device and marking the first message with an identifier of a network access device. For example, the network load balancer receives an initial request and marks the request with the source (e.g., user/client) IP address for matching to a forwarding rule.

⁵³ <https://cloud.google.com/load-balancing/docs/network/networklb-backend-service>

Backend service-based external TCP/UDP Network Load Balancing overview

[Send feedback](#)

Google Cloud external TCP/UDP Network Load Balancing (after this referred to as Network Load Balancing) is a regional, pass-through load balancer. A network load balancer distributes TCP or UDP traffic among virtual machine (VM) instances in the same region.

A network load balancer can receive traffic from:

- Any client on the internet
- Google Cloud VMs with external IPs
- Google Cloud VMs that have internet access through Cloud NAT or instance-based NAT

Forwarding rule

A regional external forwarding rule specifies the protocol and ports on which the load balancer accepts traffic. Because network load balancers are not proxies, they pass traffic to backends on the same protocol and port. The forwarding rule in combination with the IP address forms the frontend of the load balancer.

The load balancer preserves the source IP addresses of incoming packets. The destination IP address for incoming packets is the IP address associated with the load balancer's forwarding rule.

Incoming traffic is matched to a forwarding rule, which is a combination of a particular IP address, protocol, and port(s) or range of ports. The forwarding rule then directs traffic to the load balancer's backend service.⁵⁴

80. By way of further example, the initial request marked with the source IP address is first used in a five-tuple hash for routing the initial request to a backend Virtual Machine (“VM”) and further referenced for directing all subsequent requests originating from the same connection to the same backend for connection tracking.

⁵⁴ <https://cloud.google.com/load-balancing/docs/network/networklb-backend-service>

Connection tracking and consistent hashing

Network Load Balancing uses a connection tracking table and a configurable consistent hashing algorithm to determine how traffic is distributed to backend VMs.

If the load balancer has an entry in its connection tracking table for an incoming packet that is part of a previously established connection, the packet is sent to the backend VM that the load balancer previously determined. The previously determined backend had been recorded in the load balancer's connection tracking table.

When the load balancer receives a packet for which it has no connection tracking entry, the load balancer does the following:

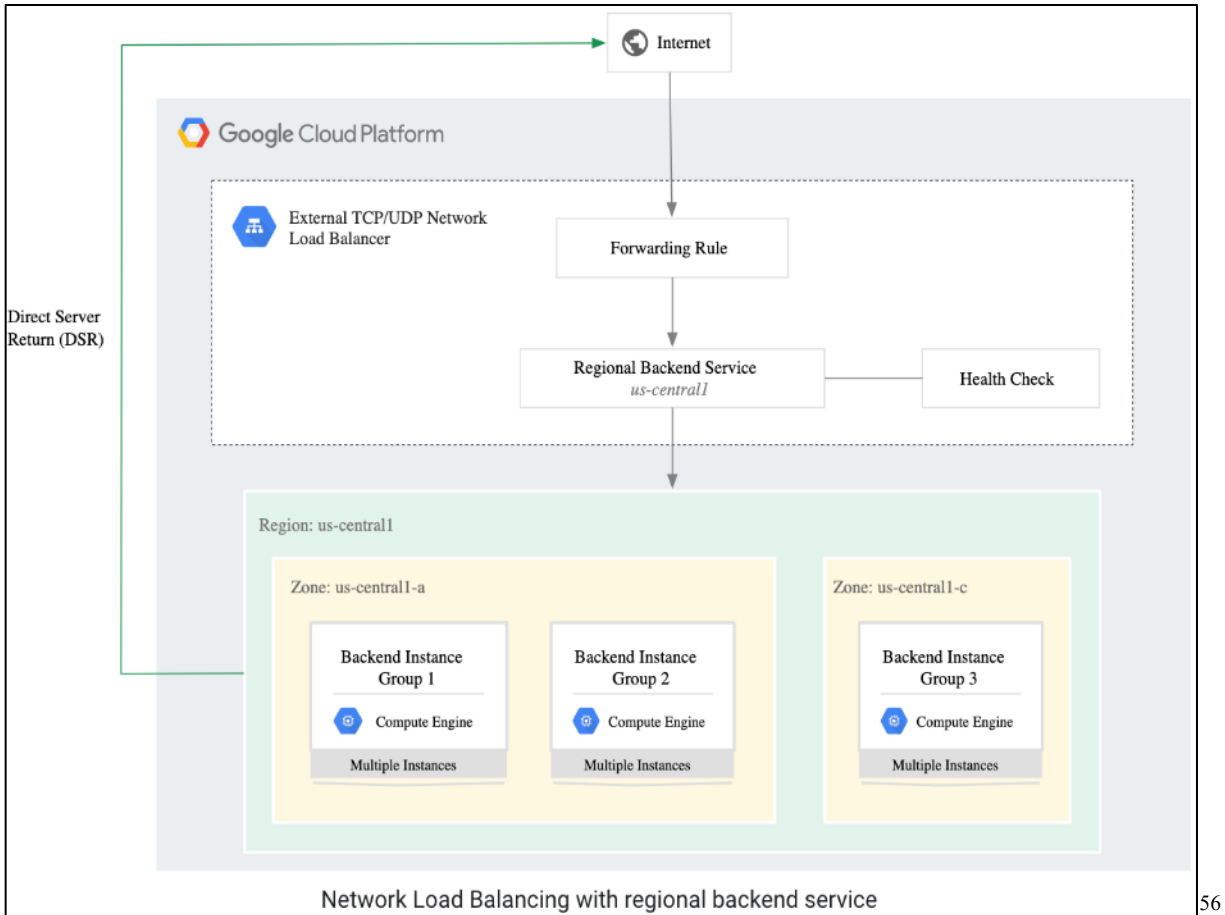
- If no session affinity has been configured, the load balancer creates a five-tuple hash of the packet's source IP address, source port, destination IP address, destination port, and the protocol. It uses this hash to select a backend that is currently healthy.
- If you have configured a session affinity option, the load balancer still creates a hash, but from fewer pieces of information, as described in [session affinity](#).
- If the packet is a TCP packet, or if the packet is a UDP packet where session affinity is set to something other than `NONE`, the load balancer records the selected backend in its connection tracking table.

Connection tracking table entries expire after 60 seconds if they are not used.

55

81. The GCP performs the step of intercepting the first message on a third network device prior to at least one first protocol server receiving the first message, wherein the third network device comprises a set of rules for load balancing of requests between a plurality of channel pairs, each channel pair having predetermined resources for a network device with predetermined capabilities. For example, the network load balancer intercepts the initial requests before the backend servicer receives the request.

⁵⁵ <https://cloud.google.com/load-balancing/docs/network/networklb-backend-service>



56

82. By way of further example, the network load balancer comprises a set of rules, such as defined by the forwarding rule, which in turn contains a set of rules, for load balancing of requests between a plurality of backend instances set up for the user/client device, with connection tracking and with/without session affinity.

⁵⁶ <https://cloud.google.com/load-balancing/docs/network/networklb-backend-service>

Forwarding rule

A regional external forwarding rule specifies the protocol and ports on which the load balancer accepts traffic. Because network load balancers are not proxies, they pass traffic to backends on the same protocol and port. The forwarding rule in combination with the IP address forms the frontend of the load balancer.

The load balancer preserves the source IP addresses of incoming packets. The destination IP address for incoming packets is the IP address associated with the load balancer's forwarding rule.

Incoming traffic is matched to a forwarding rule, which is a combination of a particular IP address, protocol, and port(s) or range of ports. The forwarding rule then directs traffic to the load balancer's backend service.

A network load balancer requires at least one forwarding rule. You can define multiple forwarding rules for the same load balancer as described in the next section.

Multiple forwarding rules

You can configure multiple regional external forwarding rules for the same network load balancer. Optionally, each forwarding rule can have a different regional external IP address, or multiple forwarding rules can have the same regional external IP address.

Configuring multiple regional external forwarding rules can be useful for these use cases:

- You need to configure more than one external IP address for the same backend service.
- You need to configure different protocols, ports or port ranges by using the same external IP address.

When using multiple forwarding rules, make sure that you configure the software running on your backend VMs so that it binds to all the external IP address(es) of the load balancer's forwarding rule(s).

57

83. By way of further example, each backend VM/instance has predetermined resources, such as those to support the balancing mode of CONNECTION, connection tracking, and/or session affinity, for a user/client device with predetermined capabilities, e.g., those of sending TCP/UDP requests.

Balancing mode

The balancing mode determines whether backends of a load balancer can handle additional traffic or are fully loaded. Google Cloud has three balancing modes:

- CONNECTION
- RATE
- UTILIZATION

⁵⁷ <https://cloud.google.com/load-balancing/docs/network/networklb-backend-service>

Connection balancing mode

For `CONNECTION` balancing mode, the target capacity defines a target maximum number of concurrent connections. Except for internal TCP/UDP load balancers and network load balancers, you must use one of the following settings to specify a target maximum number of connections:

- `max-connections-per-instance` (per VM): Target average number of connections for a single VM.
- `max-connections-per-endpoint` (per endpoint in a zonal NEG): Target average number of connections for a single endpoint.
- `max-connections` (per zonal NEGs and for zonal instance groups): Target average number of connections for the whole NEG or instance group. For regional managed instance groups, use `max-connections-per-instance` instead.

58

Connection tracking and consistent hashing

Network Load Balancing uses a connection tracking table and a configurable consistent hashing algorithm to determine how traffic is distributed to backend VMs.

If the load balancer has an entry in its connection tracking table for an incoming packet that is part of a previously established connection, the packet is sent to the backend VM that the load balancer previously determined. The previously determined backend had been recorded in the load balancer's connection tracking table.

When the load balancer receives a packet for which it has no connection tracking entry, the load balancer does the following:

- If no session affinity has been configured, the load balancer creates a five-tuple hash of the packet's source IP address, source port, destination IP address, destination port, and the protocol. It uses this hash to select a backend that is currently healthy.
- If you have configured a session affinity option, the load balancer still creates a hash, but from fewer pieces of information, as described in [session affinity](#).
- If the packet is a TCP packet, or if the packet is a UDP packet where session affinity is set to something other than `NONE`, the load balancer records the selected backend in its connection tracking table. Connection tracking table entries expire after 60 seconds if they are not used.

59

⁵⁸ https://cloud.google.com/load-balancing/docs/backend-service#traffic_distribution

⁵⁹ <https://cloud.google.com/load-balancing/docs/network/networklb-backend-service>

Session affinity options

Session affinity controls the distribution of new connections from clients to the load balancer's backend VMs. For example, you can direct new connections from the same client to the same backend VM, subject to the concepts discussed in the connection tracking and consistent hashing section.

Network Load Balancing supports the following session affinity options, which you specify for the entire regional external backend service, not on a per backend instance group basis.

Session affinity	Consistent hashing method	Connection tracking	Notes
None (NONE)	5-tuple hash	5-tuple tracking for TCP only	For TCP, this is effectively the same as <i>Client IP, Client Port, Destination IP, Destination Port, Protocol</i> (5-tuple hash). For UDP, connection tracking is disabled by default.
Client IP, Destination IP (CLIENT_IP)	2-tuple hash of: • packet's source IP address • packet's destination IP address	5-tuple tracking for TCP and UDP	Use this option when you need all connections from the same source IP address to be served by the same backend VM.
Client IP, Destination IP, Protocol (CLIENT_IP_PROTO)	3-tuple hash of: • packet's source IP address • packet's destination IP address • protocol	5-tuple tracking for TCP and UDP	
Client IP, Client Port, Destination IP, Destination Port, Protocol (5-tuple) (CLIENT_IP_PORT_PROTO)	5-tuple hash	5-tuple tracking for TCP and UDP	For TCP, this is equivalent to NONE . For UDP, this option enables connection tracking.

60

84. The GCP performs the step of determining capabilities of the second network device on the third network device. For example, the network load balancer determines the user/client device's capabilities of sending TCP/UDP requests, and accordingly applies the set of rules, such as the forwarding rule(s), to determine an assignment of the user/client device to one of the backend instances.

⁶⁰ <https://cloud.google.com/load-balancing/docs/network/networklb-backend-service>

Regional backend service

Each network load balancer has one regional backend service that defines the behavior of the load balancer and how traffic is distributed to its backends. The name of the backend service is the name of the network load balancer shown in the Google Cloud Console.

Each backend service defines the following backend parameters:

- **Protocol.** A backend service accepts either TCP or UDP traffic, but not both, on the IP address and the ports specified by one or more regional external forwarding rules. The backend service allows traffic to be delivered to backend VMs on the same IP address and ports to which traffic was sent. The backend service and all associated forwarding rules must use the same protocol.
- **Traffic distribution.** A backend service allows [traffic to be distributed](#) according to a configurable [session affinity](#). The backend service can also be configured to enable [connection draining](#) and designate [failover backends](#) for the load balancer.
- **Health check.** A backend service must have an associated regional [health check](#).

Each backend service operates in a single region and distributes traffic to the first network interface (nic0) of backend VMs. Backends must be [instance groups](#) in the same region as the backend service (and forwarding rule). The backends can be zonal unmanaged instance groups, zonal managed instance groups, or regional managed instance groups.

61

85. The GCP performs the step of applying the set of rules to determine an assignment of the second network device to one of the channel pairs based on the capabilities of the second network device, a load factor associated with the channel pair, or a threshold value defining a capacity of the channel pair. For example, the network load balancer determines the user/client device's capabilities of sending TCP/UDP requests, and accordingly applies the set of rules, such as the forwarding rule(s) for the balancing mode of CONNECTION, to determine an assignment of the user/client device to one of the backend instances.

⁶¹ <https://cloud.google.com/load-balancing/docs/network/networklb-backend-service>

Balancing mode

The balancing mode determines whether backends of a load balancer can handle additional traffic or are fully loaded. Google Cloud has three balancing modes:

- `CONNECTION`
- `RATE`
- `UTILIZATION`

Connection balancing mode

For `CONNECTION` balancing mode, the target capacity defines a target maximum number of concurrent connections. Except for internal TCP/UDP load balancers and network load balancers, you must use one of the following settings to specify a target maximum number of connections:

- `max-connections-per-instance` (per VM): Target average number of connections for a single VM.
- `max-connections-per-endpoint` (per endpoint in a zonal NEG): Target average number of connections for a single endpoint.
- `max-connections` (per zonal NEGs and for zonal instance groups): Target average number of connections for the whole NEG or instance group. For regional managed instance groups, use `max-connections-per-instance` instead.

62

External HTTP(S) Load Balancing

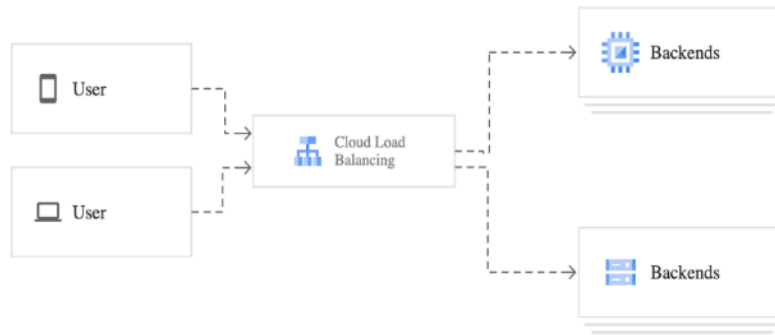
86. The GCP performs a method for dynamic load balancing of requests in a network system. For example, Google Cloud Load Balancing executes external HTTP(S) load balancing.

⁶² https://cloud.google.com/load-balancing/docs/backend-service#traffic_distribution

Cloud Load Balancing overview

[Send feedback](#)

A load balancer distributes user traffic across multiple instances of your applications. By spreading the load, load balancing reduces the risk that your applications experience performance issues.



Simple overview of load balancing (click to enlarge)

About Cloud Load Balancing

By using Cloud Load Balancing, you can serve content as close as possible to your users on a system that can respond to over one million queries per second.

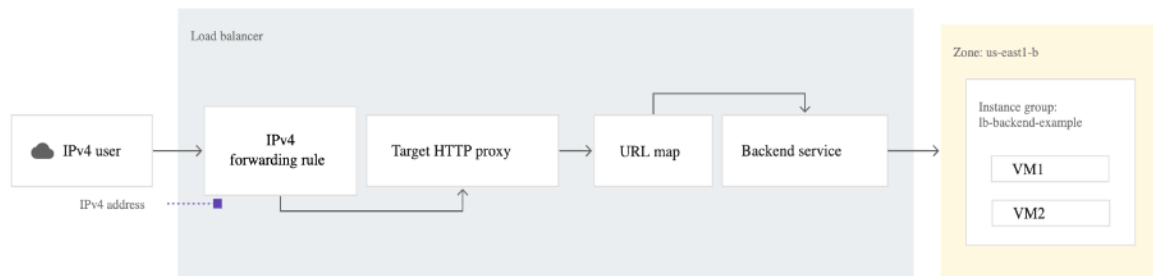
63

87. The GCP performs the step of receiving a first message on a first network device from a second network device. For example, the HTTP(S) load balancer receives an initial request from a user/client device.

⁶³ <https://cloud.google.com/load-balancing/docs/load-balancing-overview>

HTTP load balancer topology

In this guide, you create the configuration that is illustrated in the following diagram.



Simple HTTP Load Balancing (click to enlarge)

The sequence of events in the diagram is as follows:

1. A client sends a content request to the external IPv4 address defined in the forwarding rule.
2. The forwarding rule directs the request to the target HTTP proxy.
3. The target proxy uses the rule in the URL map to determine that the single backend service receives all requests.
4. The load balancer determines that the backend service has only one instance group and directs the request to a virtual machine (VM) instance in that group.
5. The VM serves the content requested by the user.

64

⁶⁴ <https://cloud.google.com/load-balancing/docs/https/ext-http-lb-simple>

External HTTP(S) Load Balancing supports the following backend types:

- [Instance groups](#)
- [Zonal network endpoint groups \(NEGs\)](#)
- [Serverless NEGs](#): One or more [App Engine](#), [Cloud Run](#), or [Cloud Functions](#) services
- [Internet NEGs](#), for endpoints that are outside of Google Cloud (also known as custom origins)
- Buckets in [Cloud Storage](#)

One common use case is load balancing traffic among services. In the following example, external IPv4 and IPv6 clients can request video, API, and image content by using the same base URL, with the paths `/api`, `/video`, and `/images`.

The external HTTP(S) load balancer's URL map specifies that:

- Requests to path `/api` go to a backend service with a VM [instance group](#) or a [zonal NEG](#) backend.
- Requests to path `/images` go to a [backend bucket](#) with a Cloud Storage backend.
- Requests to path `/video` go to a backend service that points to a [internet NEG](#) containing an external endpoint that is located on-premises outside of Google Cloud.

When a client sends a request to the load balancer's external IPv4 or IPv6 address, the load balancer evaluates the request according to the URL map and sends the request to the correct service.

65

88. The GCP performs the step of receiving a first message on a first network device from a second network device and marking the first message with an identifier of a network access device. For example, the HTTP(S) load balancer marks the initial request with the source (user/client) IP address.

Target proxies

[Target proxies](#) terminate HTTP(S) connections from clients. One or more forwarding rules direct traffic to the target proxy, and the target proxy consults the URL map to determine how to route traffic to backends.

The proxies set HTTP request/response headers as follows:

- `Via: 1.1 google` (requests and responses)
- `X-Forwarded-Proto: [http | https]` (requests only)
- `X-Cloud-Trace-Context: <trace-id>/<span-id>;<trace-options>` (requests only)
Contains parameters for [Cloud Trace](#).

⁶⁵ <https://cloud.google.com/load-balancing/docs/https/ext-http-lb-simple>

X-Forwarded-For header

The load balancer appends two IP addresses separated by a single comma to the `X-Forwarded-For` header in the following order:

- The IP address of the client that connects to the load balancer
- The IP address of the load balancer's forwarding rule

If there is no `X-Forwarded-For` header on the incoming request, these two IP addresses are the entire header value:

```
X-Forwarded-For: <client-ip>,<load-balancer-ip>
```

If the request includes an `X-Forwarded-For` header, the load balancer preserves the supplied value *before* the `<client-ip>,<load-balancer-ip>`:

```
X-Forwarded-For: <supplied-value>,<client-ip>,<load-balancer-ip>
```

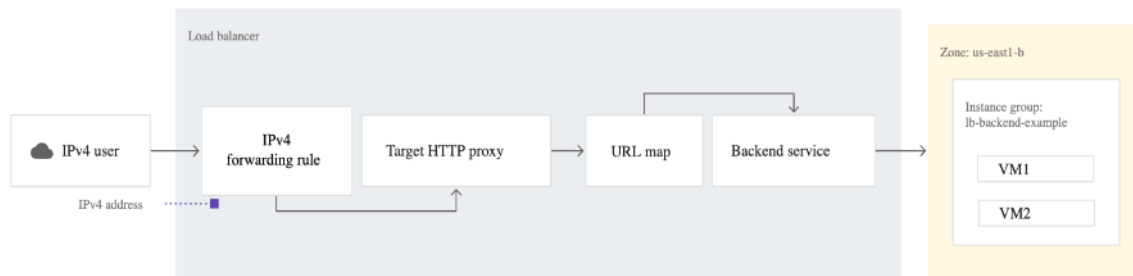
66

89. The GCP performs the step of intercepting the first message on a third network device prior to at least one first protocol server receiving the first message, wherein the third network device comprises a set of rules for load balancing of requests between a plurality of channel pairs, each channel pair having predetermined resources for a network device with predetermined capabilities. For example, the HTTP(S) load balancer intercepts the initial request before a backend server receives the request. By way of further example, the HTTP(S) load balancer comprises a set of rules, such as defined by the forwarding rule, which in turn contains a set of rules, for load balancing of requests between a plurality of backend instances set up for the user/client device, with URL map-based routing of the requests.

⁶⁶ <https://cloud.google.com/load-balancing/docs/https>

HTTP load balancer topology

In this guide, you create the configuration that is illustrated in the following diagram.



Simple HTTP Load Balancing (click to enlarge)

The sequence of events in the diagram is as follows:

1. A client sends a content request to the external IPv4 address defined in the forwarding rule.
2. The forwarding rule directs the request to the target HTTP proxy.
3. The target proxy uses the rule in the URL map to determine that the single backend service receives all requests.
4. The load balancer determines that the backend service has only one instance group and directs the request to a virtual machine (VM) instance in that group.
5. The VM serves the content requested by the user.

67

External HTTP(S) Load Balancing is implemented on [Google Front Ends \(GFEs\)](#). GFEs are distributed globally and operate together using Google's global network and control plane. In the Premium Tier, GFEs offer cross-regional load balancing, directing traffic to the closest healthy backend that has capacity and terminating HTTP(S) traffic as close as possible to your users. With Standard Tier, the load balancing is handled regionally.

⁶⁷ <https://cloud.google.com/load-balancing/docs/https/ext-http-lb-simple>

Forwarding rules and addresses

[Forwarding rules](#) route traffic by IP address, port, and protocol to a load balancing configuration consisting of a target proxy, URL map, and one or more backend services.

Each forwarding rule provides a single IP address that can be used in DNS records for your application. No DNS-based load balancing is required. You can either specify the IP address to be used or let Cloud Load Balancing assign one for you.

- The forwarding rule for an HTTP load balancer can only reference TCP ports 80 and 8080.
- The forwarding rule for an HTTPS load balancer can only reference TCP port 443.

The type of forwarding rule required by external HTTP(S) load balancers depends on which [Network Service Tier](#) the load balancer is in.

- The external HTTP(S) load balancers in the Premium Tier use global external forwarding rules.
- The external HTTP(S) load balancers in the Standard Tier use regional external forwarding rules.

68

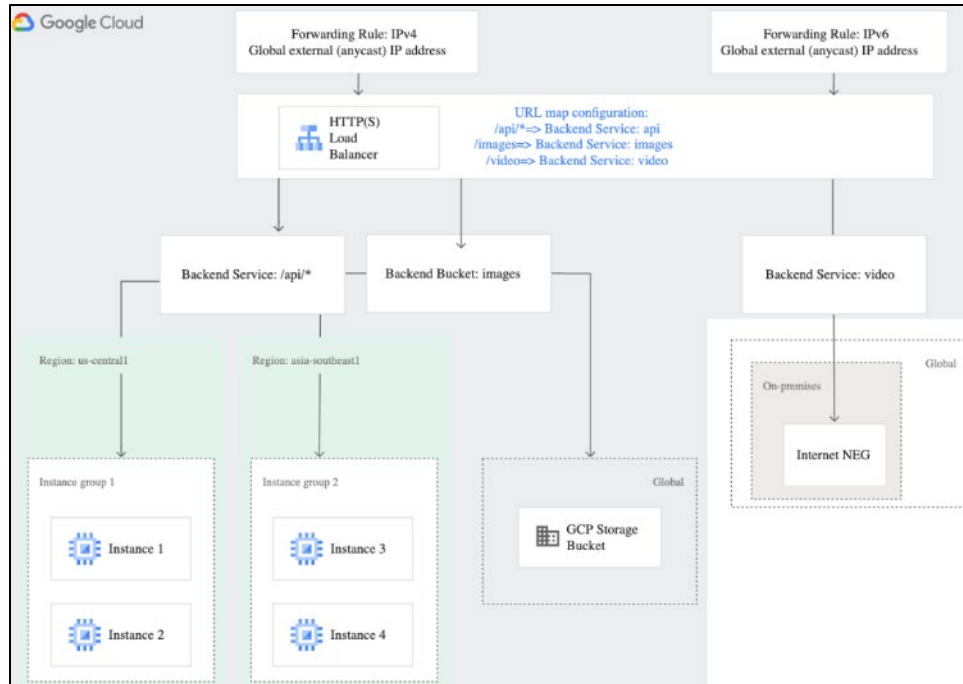
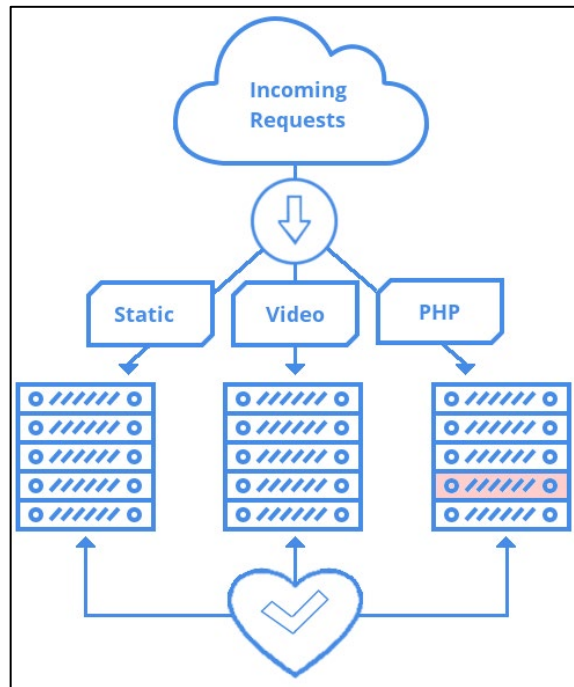
90. By way of further example, each backend instance has predetermined resources, such as those to support content-based routing, along with the balancing of RATE and/or UTILIZATION, for a user/client device with predetermined capabilities, e.g., those of sending HTTP(S) requests using IPv4 or/and IPv6.

One common use case is load balancing traffic among services. In the following example, external IPv4 and IPv6 clients can request video, API, and image content by using the same base URL, with the paths `/api`, `/video`, and `/images`.

The external HTTP(S) load balancer's URL map specifies that:

- Requests to path `/api` go to a backend service with a VM [instance group](#) or a [zonal NEG](#) backend.
- Requests to path `/images` go to a [backend bucket](#) with a Cloud Storage backend.
- Requests to path `/video` go to a backend service that points to a [internet NEG](#) containing an external endpoint that is located on-premises outside of Google Cloud.

⁶⁸ <https://cloud.google.com/load-balancing/docs/https>



⁶⁹ <https://cloud.google.com/load-balancing/docs/https>

Traffic distribution

The values of the following fields in the backend services resource determine some aspects of the backend's behavior:

- A *balancing mode* defines how the load balancer measures backend readiness for new requests or connections.
- A *target capacity* defines a target maximum number of connections, a target maximum rate, or target maximum CPU utilization.
- A *capacity scaler* adjusts overall available capacity without modifying the target capacity.

Balancing mode

The balancing mode determines whether backends of a load balancer can handle additional traffic or are fully loaded. Google Cloud has three balancing modes:

- CONNECTION
- RATE
- UTILIZATION

The balancing mode options depend on the backend service's load balancing scheme, the backend service's protocol, and the type of backends connected to the backend service.

You set the balancing mode when you add a backend to the backend service. Note that you cannot specify a balancing mode when using serverless NEGs or internet NEGs as backends for a load balancer.

70

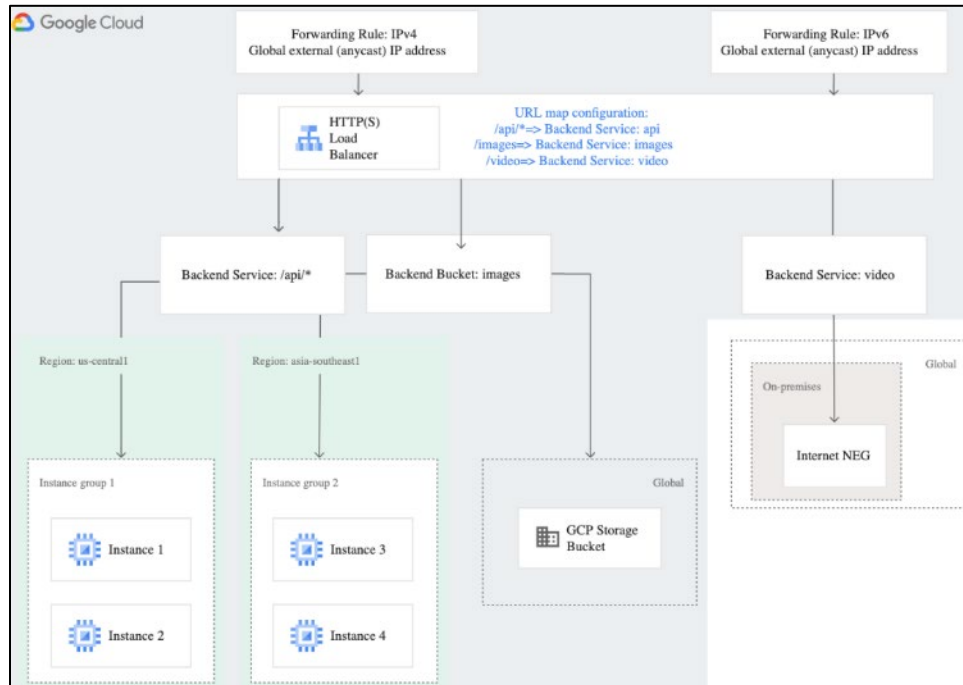
91. The GCP performs the step of applying the set of rules to determine an assignment of the second network device to one of the channel pairs based on the capabilities of the second network device, a load factor associated with the channel pair, or a threshold value defining a capacity of the channel pair. For example, the HTTP(S) load balancer determines the user/client device's capabilities of sending http(s) requests using IPv4 or/and IPv6, and accordingly applies the set of rules, such as the forwarding rule(s), along with using the balancing mode of RATE or UTILIZATION, to determine an assignment of the user/client device to one of the backend instances.

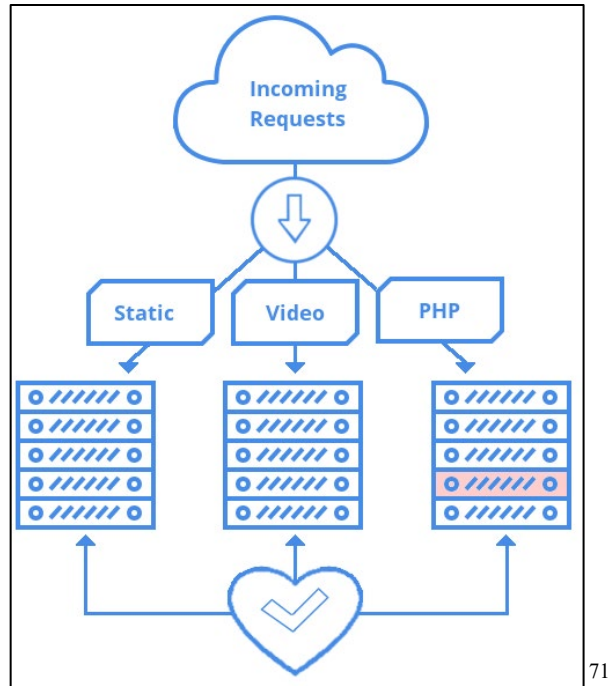
⁷⁰ https://cloud.google.com/load-balancing/docs/https#load_distribution_algorithm

One common use case is load balancing traffic among services. In the following example, external IPv4 and IPv6 clients can request video, API, and image content by using the same base URL, with the paths `/api`, `/video`, and `/images`.

The external HTTP(S) load balancer's URL map specifies that:

- Requests to path `/api` go to a backend service with a VM [instance group](#) or a [zonal NEG](#) backend.
- Requests to path `/images` go to a [backend bucket](#) with a Cloud Storage backend.
- Requests to path `/video` go to a backend service that points to a [internet NEG](#) containing an external endpoint that is located on-premises outside of Google Cloud.





71

⁷¹ <https://cloud.google.com/load-balancing/docs/https>

Traffic distribution

The values of the following fields in the backend services resource determine some aspects of the backend's behavior:

- A *balancing mode* defines how the load balancer measures backend readiness for new requests or connections.
- A *target capacity* defines a target maximum number of connections, a target maximum rate, or target maximum CPU utilization.
- A *capacity scaler* adjusts overall available capacity without modifying the target capacity.

Balancing mode

The balancing mode determines whether backends of a load balancer can handle additional traffic or are fully loaded. Google Cloud has three balancing modes:

- CONNECTION
- RATE
- UTILIZATION

The balancing mode options depend on the backend service's load balancing scheme, the backend service's protocol, and the type of backends connected to the backend service.

You set the balancing mode when you add a backend to the backend service. Note that you cannot specify a balancing mode when using serverless NEG's or internet NEG's as backends for a load balancer.

72

92. Home Depot directly infringed the '597 Patent through its implementation and use of the GCP, such as through IAM for Cloud Load Balancing, Backend service-based external TCP/UDP Network Load Balancing, and/or External HTTP(S) Load Balancing, performing at least the steps of claim 1.

93. Defendants have indirectly infringed one or more claims of the '597 Patent by knowingly and intentionally inducing others, including Home Depot customers and end-users, to directly infringe, either literally or under the doctrine of equivalents, by making, using, offering to sell, selling, and/or importing into the United States products that include the infringing technology.

⁷² https://cloud.google.com/load-balancing/docs/https#load_distribution_algorithm

94. Defendants, with knowledge that these products, or the use thereof, infringe the '597 Patent at least as of April 11, 2022,⁷³ knowingly and intentionally induced, direct infringement of the '597 Patent by providing these products to end-users for use in an infringing manner. Additionally, on information and belief, Defendants have adopted a policy of not reviewing the patents of others, including specifically those related to Defendants' specific industry, thereby remaining willfully blind to the '597 Patent at least as early as the issuance of the '597 Patent.

95. Defendants have induced infringement by others, including end-users, with the intent to cause infringing acts by others or, in the alternative, with the belief that there was a high probability that others, including end-users, infringe the '597 Patent, but while remaining willfully blind to the infringement. Defendants have induced infringement by their customers and end-users by supplying them with instructions on how to operate the infringing technology in an infringing manner, while also making publicly available information on the infringing technology via Defendants' website, product literature and packaging, and other publications.⁷⁴

96. Plaintiffs have suffered damages as a result of Defendants' direct and indirect infringement of the '597 Patent in an amount to be proven at trial.

97. Defendants have committed acts of infringement that Defendants actually knew or should have known constituted an unjustifiably high risk of infringement of at least one valid and

⁷³ Plaintiffs sent Defendants correspondence in April 2022 notifying them of their infringement of the '597 Patent. Plaintiffs sent a follow-up letter to the April 2022 correspondence dated December 19, 2022. In addition to these two letters, Plaintiffs sent multiple emails to Defendants regarding at least some of the Asserted Patents, including on April 20, 2023, May 26, 2023, November 16, 2023, and February 8, 2024. To date, Defendants have not responded to any of Plaintiffs letters or emails.

⁷⁴ See, e.g., https://www.homedepot.com/c/customer_service; https://www.homedepot.com/c/SF_Mobile_Shopping

enforceable claim of the '597 Patent. Defendants' direct and indirect infringement of the '597 Patent was willful, intentional, deliberate, and/or in conscious disregard of rights under the patent. Plaintiffs are entitled to an award of treble damages, reasonable attorney fees, and costs in bringing this action.

COUNT IV
(Infringement of the '832 Patent)

98. Paragraphs 1 through 37 are incorporated by reference as if fully set forth herein.

99. Plaintiffs have not licensed or otherwise authorized Defendants to make, use, offer for sale, sell, or import any products that embody the inventions of the '832 Patent.

100. Defendants have directly infringed the '832 Patent, either literally or under the doctrine of equivalents, without authority and in violation of 35 U.S.C. § 271, by making, using, offering to sell, selling, and/or importing into the United States products that satisfy each and every limitation of one or more claims of the '832 Patent. Such products include Apache Kafka Streams using RocksDB Universal Compaction and also those including Apache Cassandra (e.g., DataStax).

101. For example, Defendants have directly infringed at least claim 25 of the '832 Patent by making, using, offering to sell, selling, and/or importing into the United States products that include Apache Kafka Streams using RocksDB Universal Compaction and Apache Cassandra (e.g., DataStax).

102. Upon information and belief, Home Depot has used Apache Kafka Streams using RocksDB Universal Compaction and/or Apache Cassandra since at least 2015.



Cloud Infrastructure Engineer

The Home Depot

Jul 2022 - Present · 11 mos

Remote

Skills: Apache Kafka · Docker · Gitlab · Ansible · Maven · Python (Programming Language) · Kafka Streams · Terraform · Azure Kubernetes Service (AKS) · Microsoft Azure · Amazon Web Services (AWS) · Jenkins · Kubernetes

75



Full-stack Developer

HOME DEPOT USA INC

Jun 2019 - Present · 4 years

Atlanta, Georgia, United States

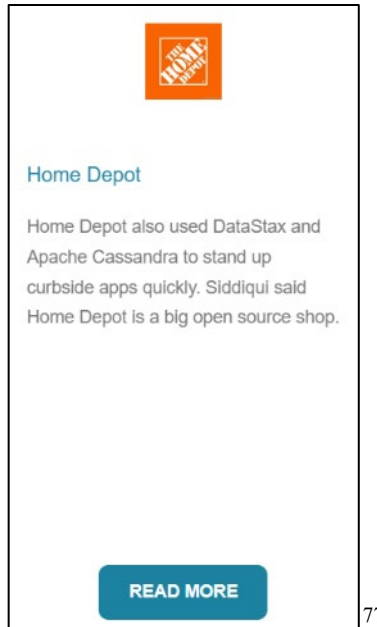
Project: Host Order Injector (HOI), is a Kafka Streams Topology which combines data from Account Binary Stream with a stream of Order data and produces into output messaging streams with the required model for downstream systems.

- Interacting with the Business users, Stake holders, Managers, Product Owners (PO) in understanding the business requirements and converting them into system requirements.
- Worked on Agile Methodology using SAFe principles and following Program Increment (PI) model to meet timelines with quality deliverables.
- Worked on Stateful Set Streams Processing application utilizing Apache Kafka Streams API to filter, map, process, store the data in Rocks DB state store backed by EBS and stream out messages to the downstream systems.

76

⁷⁵ <https://www.linkedin.com/in/pavan-k-devops-engineer/>

⁷⁶ <https://www.linkedin.com/in/sanath-chiluveru-396083273>



Curbside pickup in a hurry

Home Depot has been on a journey in recent years to move to a One Home Depot architecture that connects logistics, delivery, supply chain, customers, digital channels, and associates. One key effort revolved around standing up curbside pickup quickly.

Also: [Walmart, Target, Home Depot and Lowe's: How their digital strategies paid off](#) | [Home Depot digital transformation plan takes longer to pay off](#) | [Home Depot plans to hire 1,000 IT pros as it builds the tech behind One Home Depot strategy](#)

Home Depot also used DataStax and Apache Cassandra to stand up curbside apps quickly. Siddiqui said Home Depot is a big open source shop. ⁷⁸

⁷⁷ https://cassandra.apache.org/_/case-studies.html

⁷⁸ <https://www.zdnet.com/article/how-home-depot-navigated-a-demand-boom-during-covid-19/>

Scaling the Admins: Part 1 - Automating Deployment and Upgrades with Ansible - Sean Durity & Terrell Cole, The Home Depot

[Sign up or log in](#) to save this to your schedule, view media, leave feedback and see who's attending!

<https://sched.co/1Gy9v>

[Tweet](#)

[Share](#)

The Home Depot self-manages a wide variety of critical Cassandra clusters both on-prem and in the cloud. To scale well and to back-up their boast that "we don't do down time," the Database Solutions team required strong automation. Faced with an enterprise move to a new O/S, the team used Ansible to automate both the deployment of new clusters and the upgrading of existing ones. This talk outlines the general approach taken with Ansible and details specific Ansible tips for automating your Cassandra clusters.

Speakers



Sean Durity

Staff Systems Engineer, The Home Depot

Sean Durity administers multiple small and large Cassandra clusters for The Home Depot. He has been a Cassandra fan since his first Cassandra Summit in 2013. As the primary SME, he helped The Home Depot grow from 3 clusters to about 1,000 production nodes across several teams. He... [Read More →](#)

79



Sean Durity

The Home Depot

Staff Systems Engineer

Sean Durity administers multiple small and large Cassandra clusters for The Home Depot. He has been a Cassandra fan since his first Cassandra Summit in 2013. As the primary SME, he helped The Home Depot grow from 3 clusters to about 1,000 production nodes across several teams. He has a passion for teaching, excellent customer service, and scaling technology with a small number of expert administrators.

He is a certified DataStax Administrator and Google Cloud Architect. He was named a DataStax MVP in 2019. His last talk "How to Become the Lord of the Rings" (at DataStax Accelerate 2019) equipped the full room to lead Cassandra fellowships at their companies.

80

⁷⁹ <https://cassandrasummit2023.sched.com/event/1Gy9v/scaling-the-admins-part-1-automating-deployment-and-upgrades-with-ansible-sean-durity-terrell-cole-the-home-depot>

⁸⁰ <https://cassandrasummit2023.sched.com/speaker/seandurity>



The Home Depot

15 yrs 9 mos

- #### Staff Systems Engineer (Cassandra)

Full-time
Nov 2013 - Present · 9 yrs 7 mos
Atlanta, Georgia

Top owner and expert for the Cassandra platform for Home Depot. Cassandra (and DataStax Enterprise) administration for multiple production rings and their lower life cycles - a total of about 500 nodes and growing (current version - DSE 5 - 6.8.x on-prem and GCP). Created a suite of administration tools and scripts. Taught multiple classes on DSE, Cassandra, and data modeling for Cassandra. Help developers define appropriate data models for CQL. Mentor other admins in Cassandra. Building a new, open source Apache Cassandra platform at Home Depot using Ansible and other tools. Earned certification as Cassandra Administrator 09/2015. Earned DataStax Enterprise Certification 02/2017.

81



Software Engineer

The Home Depot
Feb 2018 - Mar 2019 · 1 yr 2 mos
Austin, Texas

- Develop RESTful services from the ground up using Spring Boot/Java 8 for deployment to GCP
- Implement a continuous integration/continuous delivery pipeline using Pivotal Concourse
- Migrate production data from on-site data centers to Cassandra clusters in the cloud using Spring Batch
- Resolve issues of being a remote team of new employees by acting as liaison between the team and HQ in Atlanta

82



Software Engineer - SOQ

The Home Depot · Contract
Jun 2021 - Apr 2022 · 11 mos
Atlanta, Georgia, United States

- Built and maintained automated workflows using Python and SQL for Apache Airflow to process hundreds of thousands of SKUs weekly
- Created and maintained applications for ETL tasks using Java for Google Cloud Dataflow to read from and write to BigQuery and Bigtable
- Designed and developed RESTful APIs using Spring Boot and HBase

83

⁸¹ <https://www.linkedin.com/in/sean-durity-18391/>

⁸² <https://www.linkedin.com/in/yousuplee/>

⁸³ <https://www.linkedin.com/in/michaeldavidsim/>

ca World '17

DEVOPS – CONTINUOUS DELIVERY

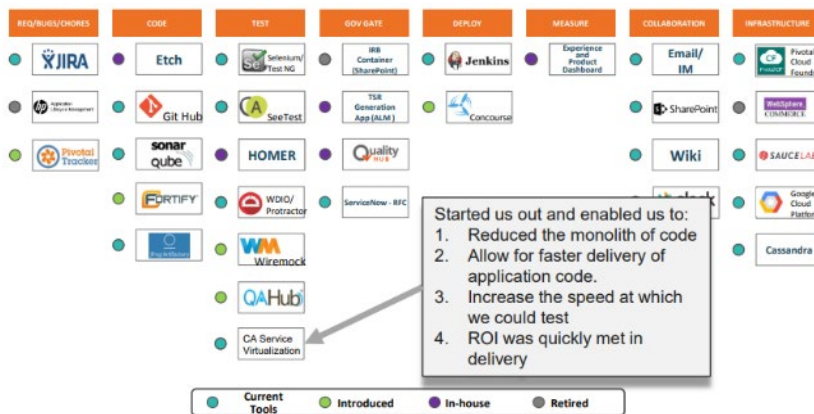
Case Study: How The Home Depot Built Quality Into Software Development.

Matt Mackenny
Director of Software Quality
The Home Depot

DO3T021S

Re-tooling and re-aligning the Quality team

Focused on building a tool stack of in-house with open source & licensed solutions. Here are some example tools we use:

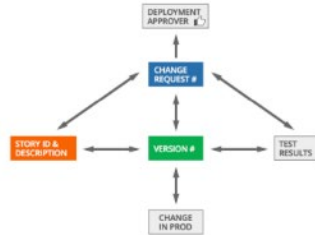


This list does not include all the test, code coverage and other tools we use



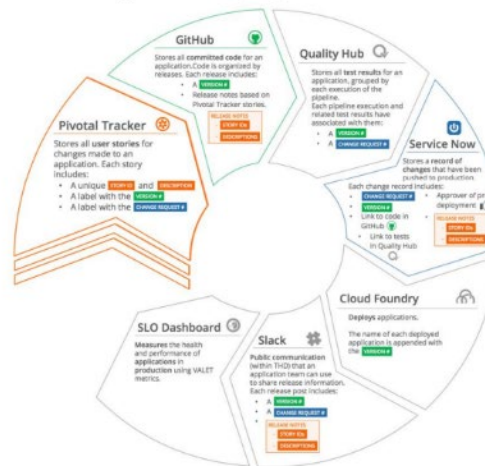
Continuous Delivery

Connecting Components of Deployment



Focused on tracking core data elements across core tools, freeing teams to modify their CD approach as needed and *not* being prescriptive in how development work is done.

Steps of the Deployment Process



84

DATASTAX
PRODUCTS
STORIES
DEVELOPERS
RESOURCES

[CLICK HERE TO WATCH OUR UPCOMING WEBINAR: HOW TO UNDERSTAND YOUR REAL-TIME DATA WITH KASKADA'S TIMELINES](#)

Home
Customer Stories
The Home Depot

Navigating a Demand Boom During COVID-19

As the pandemic struck, the world's largest home improvements chain, with over 2,300 stores across the U.S., pivoted quickly from a mostly bricks-and-mortar approach to one in which it started treating its mobile app and the internet as its "front door to stores." The Home Depot launched curbside pickup with DataStax during the pandemic in <30 days, leading to 100% increase in digital platform sales. Fahim Siddiqui, CTO at Home Depot, talks about e-commerce trends and digital transformation for the retailing giant.

PRODUCTS & SERVICES
DataStax Enterprise
INDUSTRY
Retail/eCommerce/Wholesale
ENTERPRISE
The Home Depot

85

⁸⁴ <https://www.slideshare.net/CAinc/case-study-how-the-home-depot-built-quality-into-software-development-86938724>

⁸⁵ <https://www.datastax.com/enterprise-success/home-depot>



SENIOR SOFTWARE ENGINEER – DATABASE RELIABILITY ENGINEERING (REMOTE)

APPLY

Atlanta, GA
Technology
Corporate
176961

OVERVIEW
QUALIFICATIONS
JOB DESCRIPTION
BENEFITS

Minimum Qualifications:

- Must be eighteen years of age or older.
- Must be legally permitted to work in the United States.

Preferred Qualifications:

- 4-6+ years of relevant work experience.
- Experience in Designing and deploying Highly Scalable Cassandra Database clusters on the Google Cloud Platform.
- Proficient in scripting languages and automation tools like Terraform, Ansible, and Chef required Skills
- Cockroach DB (Distributed SQL Database). So distributed database systems deployment Support and ACID Databases Support knowledge required
- Strong Experience handling common database procedures, such as upgrades, backups, recovery, migration, maintenance, etc.
- Experience in designing, implementing, and deployment of large-scale web and database infrastructures that are highly available, performing, cost-effective, and sustainable.
- Participate in continuous improvement efforts in enhancing performance and providing increased functionality.
- Work with internal and external customers to develop new value-added programs and data solutions with the existing data structure.
- Strong knowledge of Cassandra schema, CQL query, Cassandra command-line utility, and DataStax ops center.
- Experience with GCP Compute Engine, Cloud Storage, and Google-managed databases.(Cloud Spanner, Bigtable, CloudSQL, etc.) is big plus
- Defining and delivering robust monitoring solutions for database tiers that encompass both infrastructure and application-level perspectives.
- Extensive experience with designing and implementing database infrastructure and processes to support true 24x7x365 operations, by reaching the expectation of no downtime for database maintenance.
- Analyzing incidents, performance, metrics, and trends to proactively identify and resolve potential site issues before they develop.
- Proactively analyze performance to identify bottlenecks and handle incidents, bugs, and provide solutions with root cause analysis.
- Performance monitoring and query fine-tuning, Problem determination and troubleshooting, interacting with UNIX and Application group and playing a diverse role with a large team of DBREs.

86

Apache Kafka Streams using RocksDB Universal Compaction

103. Apache Kafka Streams using RocksDB Universal Compaction performs a method for monitoring performance of a storage device.

Optimized for Fast Storage

RocksDB is optimized for fast, low latency storage such as flash drives and high-speed disk drives. RocksDB exploits the full potential of high read/write rates offered by flash or RAM.

RocksDB provides several APIs to read KV pairs from a database, including Get and MultiGet for point lookups and Iterator for sequential scanning. These APIs may result in RocksDB reading blocks from SST files on disk storage. The types of blocks and the frequency with which they are read from storage is workload dependent. Some workloads may have a small working set and thus may be able to cache most of the data required, while others may have large working sets and have to read from disk more often. In the latter case, the latency would be much higher and throughput would be lower than the former. They would also be dependent on the characteristics of the underlying storage media, making it difficult to migrate from one medium to another, for example, local flash to disaggregated flash.

87

⁸⁶ <https://careers.homedepot.com/job/18317075/senior-software-engineer-database-reliability-engineering-remote-atlanta-ga/>

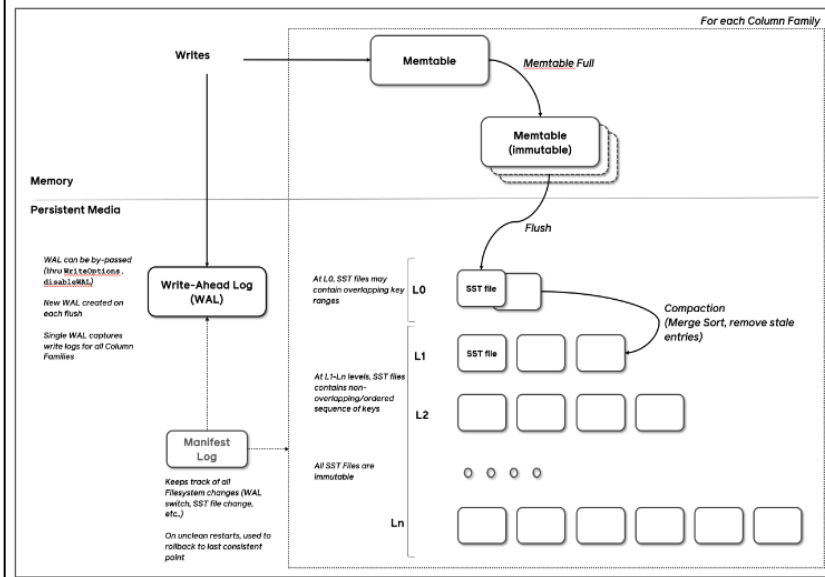
⁸⁷ <https://rocksdb.org/blog/>

3. High Level Architecture

RocksDB is a storage engine library of key-value store interface where keys and values are arbitrary byte streams. RocksDB organizes all data in sorted order and the common operations are `Get(key)`, `NewIterator()`, `Put(key, val)`, `Delete(key)`, and `SingleDelete(key)`.

The three basic constructs of RocksDB are *memtable*, *sstfile* and *logfile*. The *memtable* is an in-memory data structure - new writes are inserted into the *memtable* and are optionally written to the *logfile* (aka. *Write Ahead Log(WAL)*). The logfile is a sequentially-written file on storage. When the memtable fills up, it is flushed to a *sstfile* on storage and the corresponding logfile can be safely deleted. The data in an *sstfile* is sorted to facilitate easy lookup of keys.

The default format of *sstfile* is described in more details [here](#).



88

Compaction Styles

Both Level Style Compaction and Universal Style Compaction store data in a fixed number of logical *levels* in the database. More recent data is stored in Level-0 (L0) and older data in higher-numbered levels, up to Lmax. Files in L0 may have overlapping keys, but files in other levels generally form a single sorted run per level.

Level Style Compaction (default) typically optimizes disk footprint vs. logical database size (space amplification) by minimizing the files involved in each compaction step: merging one file in Ln with all its overlapping files in Ln+1 and replacing them with new files in Ln+1.

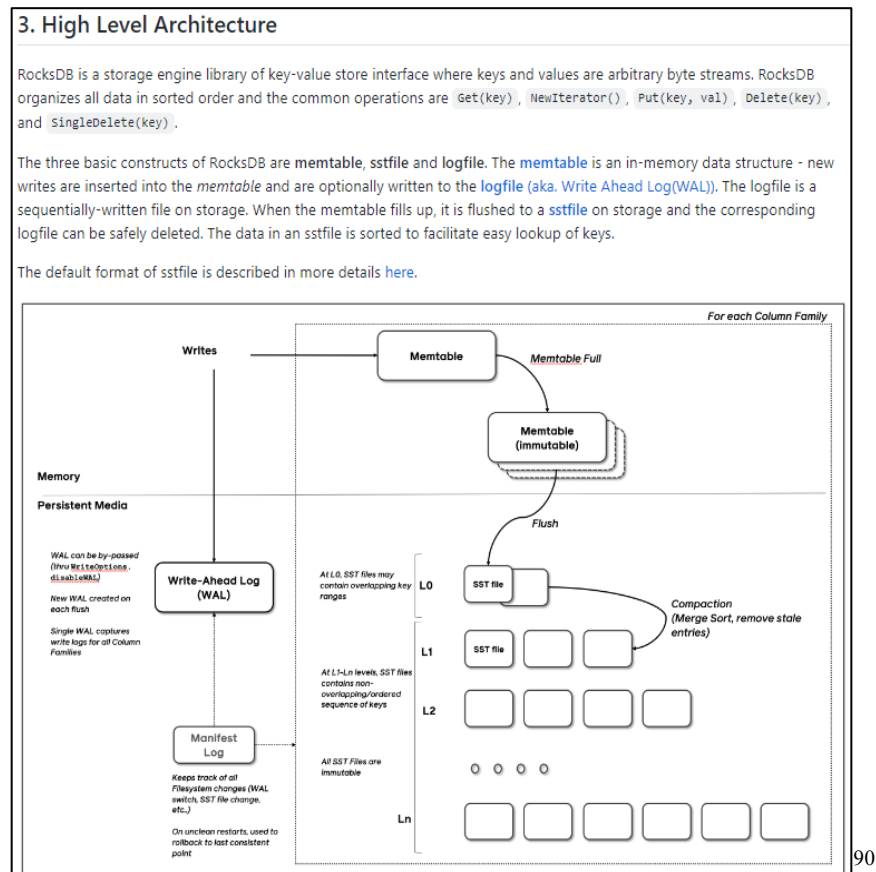
Universal Style Compaction typically optimizes total bytes written to disk vs. logical database size (write amplification) by merging potentially many files and levels at once, requiring more temporary space. Universal typically results in lower write-amplification but higher space- and read-amplification than Level Style Compaction.

89

⁸⁸ <https://github.com/facebook/rocksdb/wiki/RocksDB-Overview>

⁸⁹ <https://github.com/facebook/rocksdb/wiki/RocksDB-Overview>

104. Apache Kafka Streams using RocksDB Universal Compaction performs the step of intercepting communications between a computer system and said storage device.



105. Apache Kafka Streams using RocksDB Universal Compaction performs the step of analyzing said intercepted communications.

⁹⁰ <https://github.com/facebook/rocksdb/wiki/RocksDB-Overview>

3. High Level Architecture

RocksDB is a storage engine library of key-value store interface where keys and values are arbitrary byte streams. RocksDB organizes all data in sorted order and the common operations are `Get(key)`, `NewIterator()`, `Put(key, val)`, `Delete(key)`, and `SingleDelete(key)`.

The three basic constructs of RocksDB are **memtable**, **sstfile** and **logfile**. The **memtable** is an in-memory data structure - new writes are inserted into the *memtable* and are optionally written to the **logfile** (aka. **Write Ahead Log(WAL)**). The logfile is a sequentially-written file on storage. When the memtable fills up, it is flushed to a **sstfile** on storage and the corresponding logfile can be safely deleted. The data in an sstfile is sorted to facilitate easy lookup of keys.

91

Precondition: $n \geq \text{options.level0_file_num_compaction_trigger}$

Unless number of sorted runs reaches this threshold, no compaction will be triggered at all.

If the pre-condition is satisfied, there are four conditions. Each of them can trigger a compaction. They are evaluated in order and the selection stops once a compaction has been scheduled. Before going into detail, the overview for picking a compaction is:

1. Try to schedule a compaction by age of data
2. Else if the space amplification constraint has been violated, try to schedule a major compaction
3. Else try to schedule a compaction (minor or major) while respecting *size_ratio*
4. Else try to schedule a minor compaction without respecting *size_ratio*

92

1. Compaction triggered by age of data

For universal style compaction, the aging-based triggering criterion has the highest priority since it is a hard requirement. When trying to pick a compaction, this condition is checked first.

If the condition meets (there are files older than `options.periodic_compaction_seconds`), then RocksDB proceeds to pick sorted runs for compaction. RocksDB picks sorted runs from oldest to youngest until encountering a sorted run that is being compacted by another compaction. These files will be compacted to bottommost level unless bottommost level is reserved for ingestion behind. In this case, files will be compacted to second bottommost level.

2. Compaction Triggered by Space Amplification

If the estimated *size amplification ratio* is larger than `options.compaction_options_universal.max_size_amplification_percent / 100`, all files will be compacted to one sorted run.

3. Compaction Triggered by number of sorted runs while respecting *size_ratio*

If a compaction is not triggered by the space amplification constraint as described above then a compaction can be triggered by the number of sorted runs. This step respects *size ratio trigger* when selecting the sorted runs to compact.

We calculated a value of *size ratio trigger* as

```
size_ratio_trigger = (100 + options.compaction_options_universal.size_ratio) / 100
```

⁹¹ <https://github.com/facebook/rocksdb/wiki/Universal-Compaction>

⁹² <https://github.com/facebook/rocksdb/wiki/Terminology>

4. Compaction Triggered by number of sorted runs without respecting size_ratio

If none of the previous conditions cause a compaction to be scheduled, then this condition might schedule one. This step does not respect *size_ratio trigger* but the logic is otherwise similar to what was described above for condition 3. If we need to compact more than `options.compaction_options_universal.max_merge_width` number of sorted runs, we cap it to `options.compaction_options_universal.max_merge_width`.

93

106. Apache Kafka Streams using RocksDB Universal Compaction performs the step of reallocating at least some of said data on said storage device to enhance the performance of said storage device based on said analyzed communications.

In the presence of ongoing writes, compactions are needed for space efficiency, read (query) efficiency, and timely data deletion. Compaction removes key-value bindings that have been deleted or overwritten, and re-organizes data for query efficiency. Compactions may occur in multiple threads if configured.

The entire database is stored in a set of *sstfiles*. When a *memtable* is full, its content is written out to a file in Level-0 (L0) of the LSM tree. RocksDB removes duplicate and overwritten keys in the memtable when it is flushed to a file in L0. In *compaction*, some files are periodically read in and merged to form larger files, often going into the next LSM level (such as L1, up to Lmax).

The overall write throughput of an LSM database directly depends on the speed at which compactions can occur, especially when the data is stored in fast storage like SSD or RAM. RocksDB may be configured to issue concurrent compaction requests from multiple threads. It is observed that sustained write rates may increase by as much as a factor of 10 with multi-threaded compaction when the database is on SSDs, as compared to single-threaded compactions.

94

Apache Cassandra

107. Apache Cassandra performs a method for monitoring performance of a storage device. For example, Apache Cassandra monitors performance metrics, such as through “ReadLatency”. By way of further example, Apache Cassandra performs compaction, which results in reclaiming disk space.

ReadLatency	Latency	Local read latency for this table.
-------------	---------	------------------------------------

95

Cassandra Compaction is a process of reconciling various copies of data spread across distinct SSTables. Cassandra performs compaction of SSTables as a background activity. Cassandra has to maintain fewer SSTables and fewer copies of each data row due to compactions improving its read performance. Compaction is a crucial area of Cassandra performance and maintenance.

⁹³ <https://github.com/facebook/rocksdb/wiki/Universal-Compaction>

⁹⁴ <https://github.com/facebook/rocksdb/wiki/RocksDB-Overview>

⁹⁵ <https://cassandra.apache.org/doc/latest/cassandra/operating/metrics.html>

Cassandra's Write Path

To understand the importance of compactions in Cassandra, you must first understand how Cassandra writes data to a disk. The Cassandra write path in a nutshell:

1. Cassandra stores recent writes in memory (in a structure called the Memtable).
2. When enough writes have been made, Cassandra flushes the Memtable to disk. Data on disk is stored in relatively simple data structures called Sorted String Tables (SSTable). At the most simplified level, an SSTable could be described as a sorted array of strings.
3. Before writing a new SSTable, Cassandra merges and pre-sorts the data in the Memtable according to Primary Key. In Cassandra, a Primary Key consists of a Partition Key (the unique key that determines which node the data is stored on) and any Clustering Keys that have been defined.
4. The SSTable is written to disk as a single contiguous write operation. SSTables are immutable. Once they are written to disk they are not modified. Any updates to data or deletion of data within an SSTable is written to a new SSTable. If data is updated regularly, Cassandra may need to read from multiple SSTables to retrieve a single row.
5. Compaction operations occur periodically to re-write and combine SSTables. This is required because SSTables are immutable (no modifications once written to disk). Compactions prune deleted data and merge disparate row data into new SSTables in order to reclaim disk space and keep read operations optimized.

If you are unfamiliar with Cassandra's write path, please read [The write path to compaction](#) from Cassandra Wiki.

96

⁹⁶ <https://www.instaclustr.com/blog/apache-cassandra-compaction/>

Compaction

Strategies

Picking the right compaction strategy for your workload will ensure the best performance for both querying and for compaction itself.

Size Tiered Compaction Strategy (STCS)

The default compaction strategy. Useful as a fallback when other strategies don't fit the workload. Most useful for non pure time series workloads with spinning disks, or when the I/O from `LCS` is too high.

Leveled Compaction Strategy (LCS)

Leveled Compaction Strategy (LCS) is optimized for read heavy workloads, or workloads with lots of updates and deletes. It is not a good choice for immutable time series data.

Time Window Compaction Strategy (TWCS)

Time Window Compaction Strategy is designed for TTL'ed, mostly immutable time series data.

Types of compaction

The concept of compaction is used for different kinds of operations in Cassandra, the common thing about these operations is that it takes one or more SSTables and output new SSTables. The types of compactions are:

Minor compaction

triggered automatically in Cassandra.

Major compaction

a user executes a compaction over all SSTables on the node.

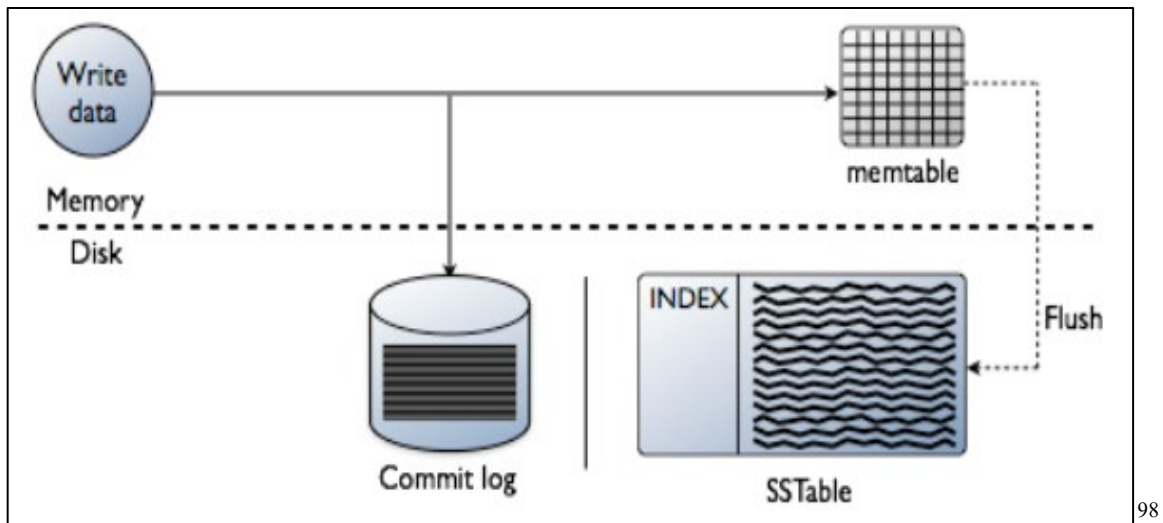
User defined compaction

a user triggers a compaction on a given set of SSTables.

97

108. Apache Cassandra performs the step of intercepting communications between a computer system (e.g., a Cassandra node) and said storage device (e.g., a disk associated with a Cassandra node).

⁹⁷ <https://cassandra.apache.org/doc/latest/cassandra/operating/compaction/>



109. Apache Cassandra performs the step of analyzing said intercepted communications. For example, the step of analyzing can entail monitoring the memtable and tracking the number of SSTables flushed to a disk and comparing it to the parameters `min_threshold` and `max_threshold` to determine whether to perform a minor compaction.

⁹⁸ <https://docs.datastax.com/en/cassandra-oss/3.0/cassandra/dml/dmlHowDataWritten.html>

Common options

There is a number of common options for all the compaction strategies;

`enabled` (default: true)

Whether minor compactions should run. Note that you can have 'enabled': true as a compaction option and then do 'nodetool enableautocompaction' to start running compactions.

`tombstone_threshold` (default: 0.2)

How much of the sstable should be tombstones for us to consider doing a single sstable compaction of that sstable.

`tombstone_compaction_interval` (default: 86400s (1 day))

Since it might not be possible to drop any tombstones when doing a single sstable compaction we need to make sure that one sstable is not constantly getting recomputed - this option states how often we should try for a given sstable.

`log_all` (default: false)

New detailed compaction logging, see `below <detailed-compaction-logging>`.

`unchecked_tombstone_compaction` (default: false)

The single sstable compaction has quite strict checks for whether it should be started, this option disables those checks and for some usecases this might be needed. Note that this does not change anything for the actual compaction, tombstones are only dropped if it is safe to do so - it might just rewrite an sstable without being able to drop any tombstones.

`only_purge_repaired_tombstone` (default: false)

Option to enable the extra safety of making sure that tombstones are only dropped if the data has been repaired.

`min_threshold` (default: 4)

Lower limit of number of SSTables before a compaction is triggered. Not used for `LeveledCompactionStrategy`.

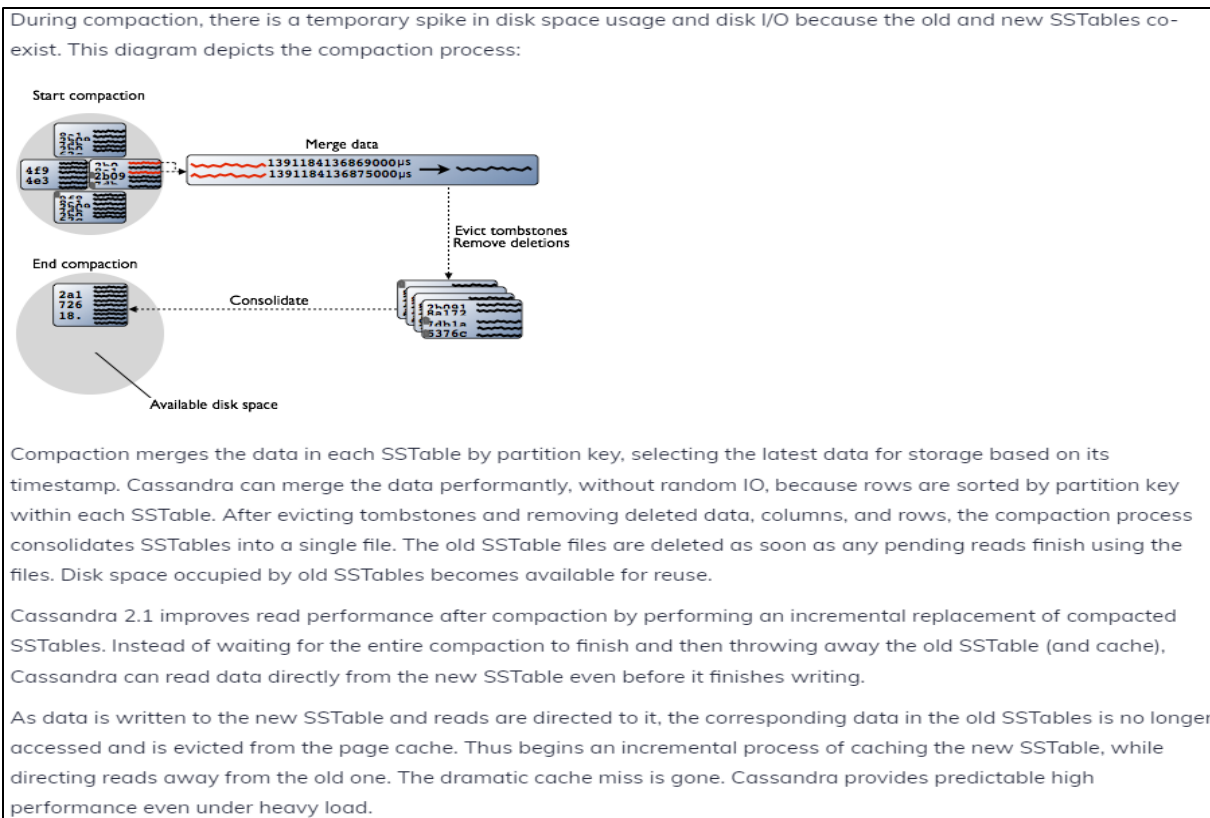
`max_threshold` (default: 32)

Upper limit of number of SSTables before a compaction is triggered. Not used for `LeveledCompactionStrategy`.

99

110. Apache Cassandra performs the step of reallocating at least some of said data on said storage device to enhance the performance of said storage device based on said analyzed communications. For example, the step of reallocating can occur during minor compaction and entails merging SSTables. By way of further example, the step of relocating enhancing performance of the storage device can include reducing access (“read”) time and making additional disk space available for reuse.

⁹⁹ <https://cassandra.apache.org/doc/latest/cassandra/operating/compaction/>



100

111. Home Depot directly infringed the '832 Patent through its use, promotional demonstrations, testing, repairs, and instructional guidance of the Kafka Streams using RocksDB Universal Compaction and/or Apache Cassandra, performing at least the steps of claim 25.

112. Defendants have indirectly infringed one or more claims of the '832 Patent by knowingly and intentionally inducing others, including Home Depot customers and end-users, to directly infringe, either literally or under the doctrine of equivalents, by making, using, offering to sell, selling, and/or importing into the United States products that include the infringing technology.

¹⁰⁰ https://docs.datastax.com/en/cassandra-oss/2.1/cassandra/dml/dml_write_path_c.html

113. Defendants, with knowledge that these products, or the use thereof, infringe the '832 Patent at least as of May 26, 2023,¹⁰¹ knowingly and intentionally induced, direct infringement of the '832 Patent by providing these products to end-users for use in an infringing manner. Additionally, on information and belief, Defendants have adopted a policy of not reviewing the patents of others, including specifically those related to Defendants' specific industry, thereby remaining willfully blind to the '832 Patent at least as early as the issuance of the '832 Patent.

114. Defendants have induced infringement by others, including end-users, with the intent to cause infringing acts by others or, in the alternative, with the belief that there was a high probability that others, including end-users, infringe the '832 Patent, but while remaining willfully blind to the infringement. Defendants have induced infringement by their customers and end-users by supplying them with instructions on how to operate the infringing technology in an infringing manner, while also making publicly available information on the infringing technology via Defendants' website, product literature and packaging, and other publications.¹⁰²

115. Plaintiffs have suffered damages as a result of Defendants' direct and indirect infringement of the '832 Patent in an amount to be proven at trial.

116. Defendants have committed acts of infringement that Defendants actually knew or should have known constituted an unjustifiably high risk of infringement of at least one valid and enforceable claim of the '832 Patent. Defendants' direct and indirect infringement of the '832

¹⁰¹ Plaintiffs sent Defendants correspondence in May 2023 notifying them of their infringement of the '832 Patent. This correspondence was part of a series of correspondence sent from Plaintiffs to Defendants regarding Defendants' infringement of at least some of the Asserted Patents dating back to April 2022, and continuing through at least February 8, 2024. To date, Defendants have not responded to any of Plaintiffs' letters or emails.

¹⁰² See, e.g., https://www.homedepot.com/c/customer_service; https://www.homedepot.com/c/SF_Mobile_Shopping

Patent was willful, intentional, deliberate, and/or in conscious disregard of rights under the patent. Plaintiffs are entitled to an award of treble damages, reasonable attorney fees, and costs in bringing this action.

COUNT V
(Infringement of the '326 Patent)

117. Paragraphs 1 through 37 are incorporated by reference as if fully set forth herein.

118. Plaintiffs have not licensed or otherwise authorized Defendants to make, use, offer for sale, sell, or import any products that embody the inventions of the '326 Patent.

119. Defendants have directly infringed the '326 Patent, either literally or under the doctrine of equivalents, without authority and in violation of 35 U.S.C. § 271, by making, using, offering to sell, selling, and/or importing into the United States products that satisfy each and every limitation of one or more claims of the '326 Patent. Such products include the GCP Cloud Load Balancing, as well as IAM for Cloud Balancing, Backend service-based external TCP/UDP Network Load Balancing, and External HTTP(S) Load Balancing.

120. For example, Defendants have directly infringed at least claim 1 of the '326 Patent by making, using, offering to sell, selling, and/or importing into the United States products that include the GCP Cloud Load Balancing, as well as IAM for Cloud Balancing, Backend service-based external TCP/UDP Network Load Balancing, and External HTTP(S) Load Balancing.

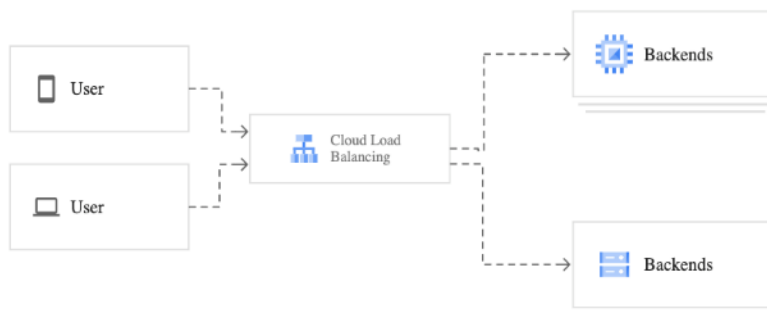
121. Upon information and belief, Home Depot has used the GCP since at least 2016. The allegations in Paragraph 75 are incorporated by reference as if fully set forth herein.

External TCP/UDP Network Load Balancing

122. The GCP comprises a network system for policy provisioning and access managing. For example, Google Cloud with IAM is a network system including backend service-based external TCP/UDP Network Load Balancing used for policy provisioning and access managing (provisioning load balancing policy and managing backend server access).

Cloud Load Balancing overview

A load balancer distributes user traffic across multiple instances of your applications. By spreading the load, load balancing reduces the risk that your applications experience performance issues.



103

Managing Access Control for Cloud Load Balancing using IAM

You can get and set IAM policies using the Google Cloud Console, the IAM API, or the `gcloud` command-line tool. See [Granting, changing, and revoking access to project members](#) for details.

104

Overview of IAM policy

Access to a resource is managed through an **IAM policy**. A policy is a collection of **bindings** that associate a member, such as a user account or service account, with a role. Policies are represented using JSON or YAML.

105

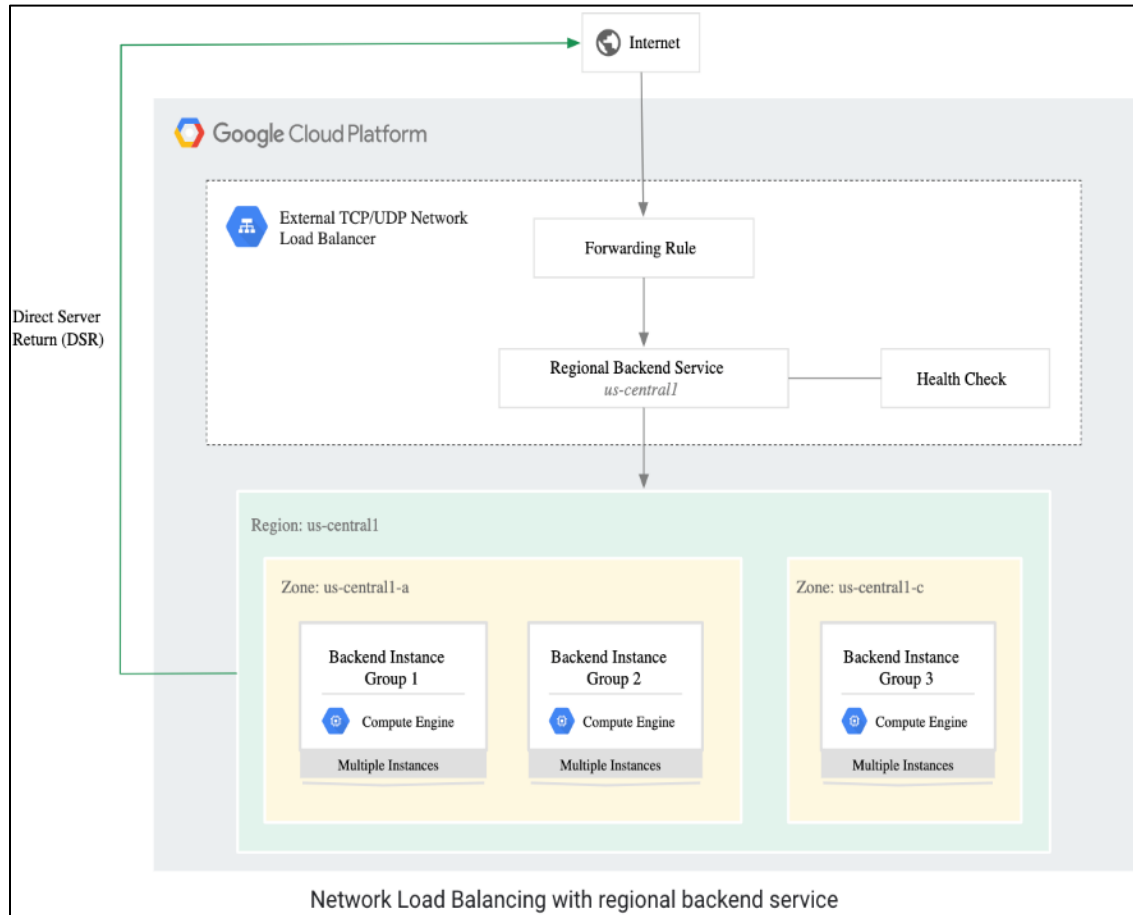
123. The GCP comprises a first network device for marking an incoming message with an identifier of a network access. For example, the network load balancer receives an initial request and marks the request with the source (e.g., user/client) IP address for matching to a

¹⁰³ <https://cloud.google.com/load-balancing/docs/load-balancing-overview>

¹⁰⁴ <https://cloud.google.com/load-balancing/docs/access-control>

¹⁰⁵ <https://cloud.google.com/iam/docs/granting-changing-revoking-access>

forwarding rule.



Backend service-based external TCP/UDP Network Load Balancing overview

[Send feedback](#)

Google Cloud external TCP/UDP Network Load Balancing (after this referred to as Network Load Balancing) is a regional, pass-through load balancer. A network load balancer distributes TCP or UDP traffic among virtual machine (VM) instances in the same region.

A network load balancer can receive traffic from:

- Any client on the internet
- Google Cloud VMs with external IPs
- Google Cloud VMs that have internet access through Cloud NAT or instance-based NAT

Forwarding rule

A regional external forwarding rule specifies the protocol and ports on which the load balancer accepts traffic. Because network load balancers are not proxies, they pass traffic to backends on the same protocol and port. The forwarding rule in combination with the IP address forms the frontend of the load balancer.

The load balancer preserves the source IP addresses of incoming packets. The destination IP address for incoming packets is the IP address associated with the load balancer's forwarding rule.

Incoming traffic is matched to a forwarding rule, which is a combination of a particular IP address, protocol, and port(s) or range of ports. The forwarding rule then directs traffic to the load balancer's backend service.¹⁰⁶

124. By way of further example, the network load balancer receives an initial request and marks the request with the source (user/client) IP address, which is first used in a five-tuple hash for routing the initial request to a backend VM and further referenced for directing all subsequent requests originating from the same connection to the same backend for connection tracking.

¹⁰⁶ <https://cloud.google.com/load-balancing/docs/network/networklb-backend-service>

Connection tracking and consistent hashing

Network Load Balancing uses a connection tracking table and a configurable consistent hashing algorithm to determine how traffic is distributed to backend VMs.

If the load balancer has an entry in its connection tracking table for an incoming packet that is part of a previously established connection, the packet is sent to the backend VM that the load balancer previously determined. The previously determined backend had been recorded in the load balancer's connection tracking table.

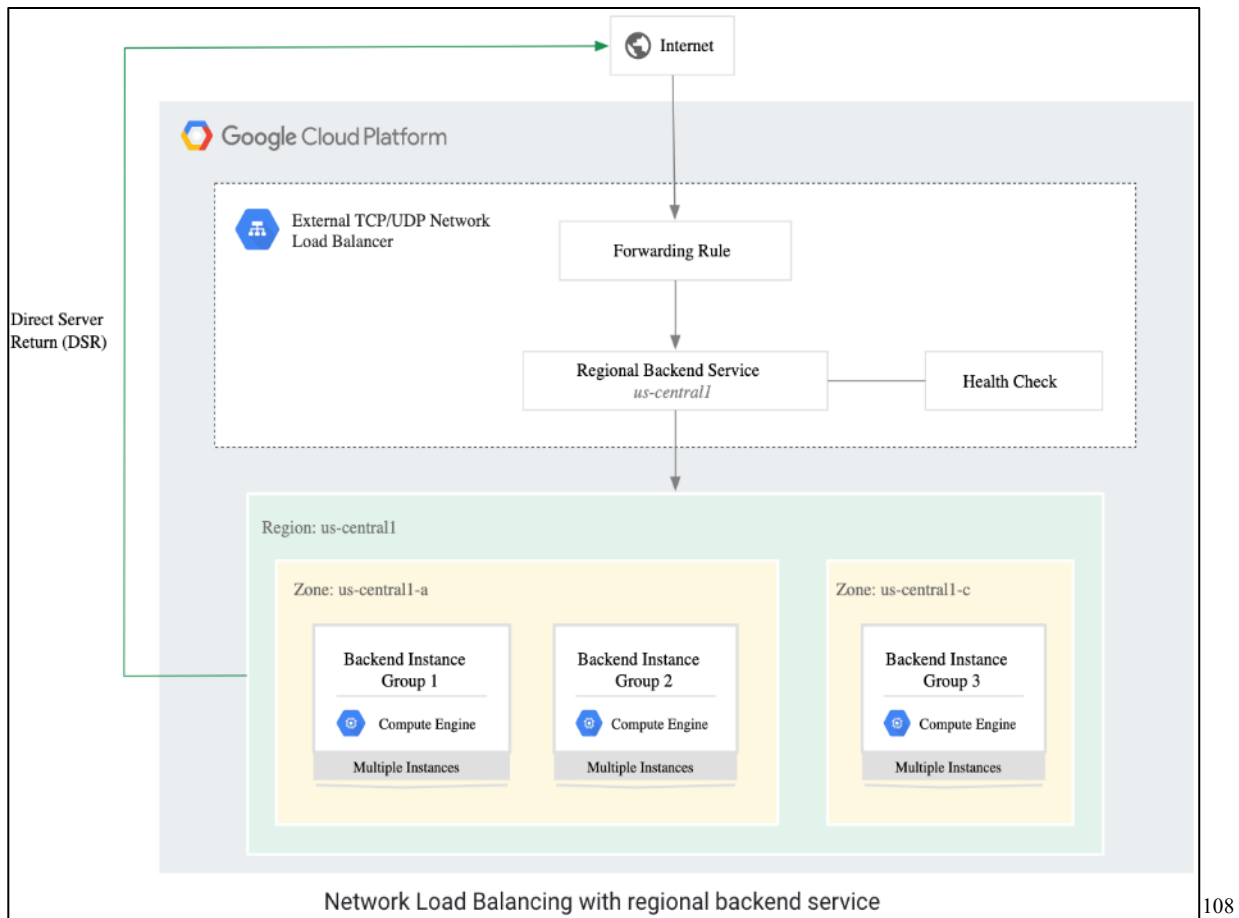
When the load balancer receives a packet for which it has no connection tracking entry, the load balancer does the following:

- If no session affinity has been configured, the load balancer creates a five-tuple hash of the packet's source IP address, source port, destination IP address, destination port, and the protocol. It uses this hash to select a backend that is currently healthy.
- If you have configured a session affinity option, the load balancer still creates a hash, but from fewer pieces of information, as described in [session affinity](#).
- If the packet is a TCP packet, or if the packet is a UDP packet where session affinity is set to something other than **NONE**, the load balancer records the selected backend in its connection tracking table. Connection tracking table entries expire after 60 seconds if they are not used.

107

125. The GCP comprises a second network device for policy provisioning and access managing, wherein the second network device intercepting the incoming message prior to at least one first protocol server receives the incoming message. For example, the network load balancer furnishes policy provisioning and access managing, e.g., load balancing policy provisioning and backend server access managing, according to IAM. The network load balancer intercepts the incoming request before a backend server receives the request.

¹⁰⁷ <https://cloud.google.com/load-balancing/docs/network/networklb-backend-service>



126. By way of further example, the network load balancer furnishes policy provisioning and access managing, e.g., load balancing policy provisioning and backend server access managing, according to IAM. The network load balancer intercepts the incoming request before a backend server receives the request.

Managing Access Control for Cloud Load Balancing using IAM

You can get and set IAM policies using the Google Cloud Console, the IAM API, or the `gcloud` command-line tool. See [Granting, changing, and revoking access to project members](#) for details.

¹⁰⁸ <https://cloud.google.com/load-balancing/docs/network/networklb-backend-service>

Forwarding rule

A regional external forwarding rule specifies the protocol and ports on which the load balancer accepts traffic. Because network load balancers are not proxies, they pass traffic to backends on the same protocol and port. The forwarding rule in combination with the IP address forms the frontend of the load balancer.

The load balancer preserves the source IP addresses of incoming packets. The destination IP address for incoming packets is the IP address associated with the load balancer's forwarding rule.

Incoming traffic is matched to a forwarding rule, which is a combination of a particular IP address, protocol, and port(s) or range of ports. The forwarding rule then directs traffic to the load balancer's backend service.

A network load balancer requires at least one forwarding rule. You can define multiple forwarding rules for the same load balancer as described in the next section.

109

Regional backend service

Each network load balancer has one regional backend service that defines the behavior of the load balancer and how traffic is distributed to its backends. The name of the backend service is the name of the network load balancer shown in the Google Cloud Console.

Backend instance groups

An external TCP/UDP load balancer distributes connections among backend VMs contained within managed or unmanaged instance groups.

110

127. The GCP comprises a second network device for policy provisioning and access managing, wherein the second network device intercepting the incoming message prior to at least one first protocol server receives the incoming message, identifying a network device using the identifier and managing an assignment of a plurality of configuration settings based on the identifier. For example, the network load balancer identifies a user/client device using the source (user/client) IP address, and accordingly manages an assignment of configuration settings (such as derived from a connection tracking table and/or session affinity) based on the source IP address.

¹⁰⁹ <https://cloud.google.com/load-balancing/docs/access-control>

¹¹⁰ <https://cloud.google.com/load-balancing/docs/network/networklb-backend-service>

Connection tracking and consistent hashing

Network Load Balancing uses a connection tracking table and a configurable consistent hashing algorithm to determine how traffic is distributed to backend VMs.

If the load balancer has an entry in its connection tracking table for an incoming packet that is part of a previously established connection, the packet is sent to the backend VM that the load balancer previously determined. The previously determined backend had been recorded in the load balancer's connection tracking table.

When the load balancer receives a packet for which it has no connection tracking entry, the load balancer does the following:

- If no session affinity has been configured, the load balancer creates a five-tuple hash of the packet's source IP address, source port, destination IP address, destination port, and the protocol. It uses this hash to select a backend that is currently healthy.
- If you have configured a session affinity option, the load balancer still creates a hash, but from fewer pieces of information, as described in [session affinity](#).
- If the packet is a TCP packet, or if the packet is a UDP packet where session affinity is set to something other than `NONE`, the load balancer records the selected backend in its connection tracking table. Connection tracking table entries expire after 60 seconds if they are not used.

Session affinity options

Session affinity controls the distribution of new connections from clients to the load balancer's backend VMs. For example, you can direct new connections from the same client to the same backend VM, subject to the concepts discussed in the connection tracking and consistent hashing section.

Network Load Balancing supports the following session affinity options, which you specify for the entire regional external backend service, not on a per backend instance group basis.

Session affinity	Consistent hashing method	Connection tracking	Notes
None (NONE)	5-tuple hash	5-tuple tracking for TCP only	For TCP, this is effectively the same as <i>Client IP, Client Port, Destination IP, Destination Port, Protocol</i> (5-tuple hash). For UDP, connection tracking is disabled by default.
Client IP, Destination IP (CLIENT_IP)	2-tuple hash of: <ul style="list-style-type: none">• packet's source IP address• packet's destination IP address	5-tuple tracking for TCP and UDP	Use this option when you need all connections from the same source IP address to be served by the same backend VM.
Client IP, Destination IP, Protocol (CLIENT_IP_PROTO)	3-tuple hash of: <ul style="list-style-type: none">• packet's source IP address• packet's destination IP address• protocol	5-tuple tracking for TCP and UDP	
Client IP, Client Port, Destination IP, Destination Port, Protocol (5-tuple) (CLIENT_IP_PORT_PROTO)	5-tuple hash	5-tuple tracking for TCP and UDP	For TCP, this is equivalent to NONE . For UDP, this option enables connection tracking.

111

128. The GCP comprises a database (e.g., a set of tables derived for connection tracking and/or session affinity) for storing a plurality of configuration information records (e.g., including a user/client IP address and/or derived for session affinity), wherein each record includes an identifier of a network access device and a plurality of configuration information settings (e.g., derived for connection tracking) constructed based on a service level agreement (e.g., IP based session affinity) associated with the identifier of each record.

¹¹¹ <https://cloud.google.com/load-balancing/docs/load-balancing-overview>.

Connection tracking and consistent hashing

Network Load Balancing uses a connection tracking table and a configurable consistent hashing algorithm to determine how traffic is distributed to backend VMs.

If the load balancer has an entry in its connection tracking table for an incoming packet that is part of a previously established connection, the packet is sent to the backend VM that the load balancer previously determined. The previously determined backend had been recorded in the load balancer's connection tracking table.

When the load balancer receives a packet for which it has no connection tracking entry, the load balancer does the following:

- If no session affinity has been configured, the load balancer creates a five-tuple hash of the packet's source IP address, source port, destination IP address, destination port, and the protocol. It uses this hash to select a backend that is currently healthy.
- If you have configured a session affinity option, the load balancer still creates a hash, but from fewer pieces of information, as described in [session affinity](#).
- If the packet is a TCP packet, or if the packet is a UDP packet where session affinity is set to something other than `NONE`, the load balancer records the selected backend in its connection tracking table. Connection tracking table entries expire after 60 seconds if they are not used.

Session affinity options			
<p>Session affinity controls the distribution of new connections from clients to the load balancer's backend VMs. For example, you can direct new connections from the same client to the same backend VM, subject to the concepts discussed in the connection tracking and consistent hashing section.</p> <p>Network Load Balancing supports the following session affinity options, which you specify for the entire regional external backend service, not on a per backend instance group basis.</p>			
Session affinity	Consistent hashing method	Connection tracking	Notes
None (NONE)	5-tuple hash	5-tuple tracking for TCP only	For TCP, this is effectively the same as <i>Client IP, Client Port, Destination IP, Destination Port, Protocol</i> (5-tuple hash). For UDP, connection tracking is disabled by default.
Client IP, Destination IP (CLIENT_IP)	2-tuple hash of: <ul style="list-style-type: none"> • packet's source IP address • packet's destination IP address 	5-tuple tracking for TCP and UDP	Use this option when you need all connections from the same source IP address to be served by the same backend VM.
Client IP, Destination IP, Protocol (CLIENT_IP_PROTO)	3-tuple hash of: <ul style="list-style-type: none"> • packet's source IP address • packet's destination IP address • protocol 	5-tuple tracking for TCP and UDP	
Client IP, Client Port, Destination IP, Destination Port, Protocol (5-tuple) (CLIENT_IP_PORT_PROTO)	5-tuple hash	5-tuple tracking for TCP and UDP	For TCP, this is equivalent to NONE . For UDP, this option enables connection tracking.

112

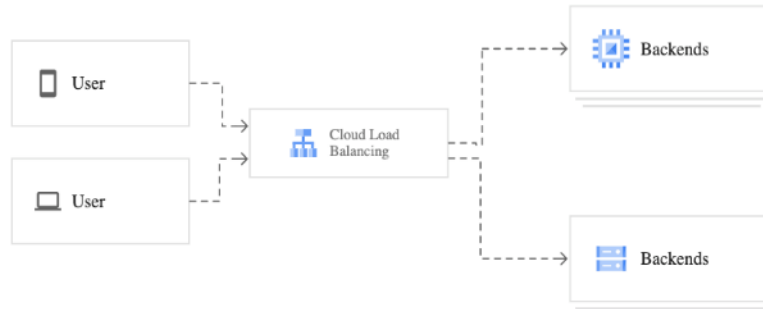
External HTTP(S) Load Balancing

129. The GCP comprises a network system for policy provisioning and access managing. For example, a network system is Google Cloud with IAM and Load Balancing, such as external http(s) Load Balancing, used for policy provisioning and access managing (provisioning load balancing policy and managing backend server access).

Cloud Load Balancing overview

¹¹² <https://cloud.google.com/load-balancing/docs/load-balancing-overview>.

A load balancer distributes user traffic across multiple instances of your applications. By spreading the load, load balancing reduces the risk that your applications experience performance issues.



113

Managing Access Control for Cloud Load Balancing using IAM

You can get and set IAM policies using the Google Cloud Console, the IAM API, or the `gcloud` command-line tool. See [Granting, changing, and revoking access to project members](#) for details.

114

Overview of IAM policy

Access to a resource is managed through an **IAM policy**. A policy is a collection of **bindings** that associate a member, such as a user account or service account, with a role. Policies are represented using JSON or YAML.

115

130. The GCP comprises a first network device for marking an incoming message with an identifier of a network access device. For example, the http(s) load balancer receives an initial request and marks the request with the source (user/client) IP address for matching to a forwarding rule. The http(s) load balancer receives an initial request and marks the request with the source (user/client) IP address for inclusion in the X-Forwarded-For header.

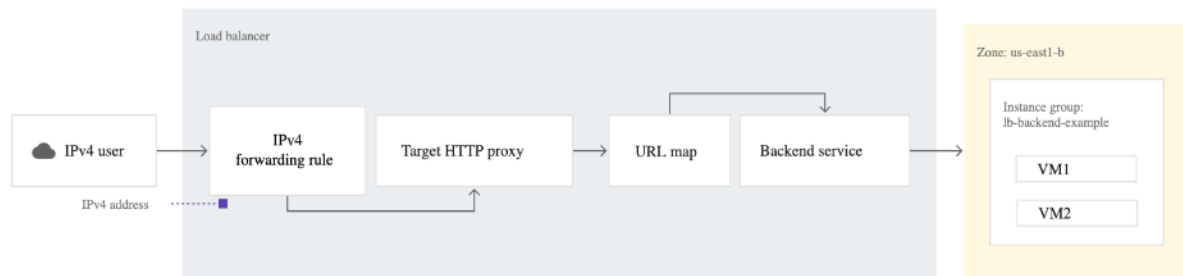
¹¹³ <https://cloud.google.com/load-balancing/docs/load-balancing-overview>

¹¹⁴ <https://cloud.google.com/load-balancing/docs/access-control>

¹¹⁵ <https://cloud.google.com/iam/docs/granting-changing-revoking-access>

HTTP load balancer topology

In this guide, you create the configuration that is illustrated in the following diagram.



Simple HTTP Load Balancing (click to enlarge)

The sequence of events in the diagram is as follows:

1. A client sends a content request to the external IPv4 address defined in the forwarding rule.
2. The forwarding rule directs the request to the target HTTP proxy.
3. The target proxy uses the rule in the URL map to determine that the single backend service receives all requests.
4. The load balancer determines that the backend service has only one instance group and directs the request to a virtual machine (VM) instance in that group.
5. The VM serves the content requested by the user.

116

Target proxies

Target proxies terminate HTTP(S) connections from clients. One or more forwarding rules direct traffic to the target proxy, and the target proxy consults the URL map to determine how to route traffic to backends.

The proxies set HTTP request/response headers as follows:

- **Via:** 1.1 google (requests and responses)
- **X-Forwarded-Proto:** [http | https] (requests only)
- **X-Cloud-Trace-Context:** <trace-id>/<span-id>;<trace-options> (requests only)
Contains parameters for [Cloud Trace](#).

¹¹⁶ <https://cloud.google.com/load-balancing/docs/https/ext-http-lb-simple>

X-Forwarded-For header

The load balancer appends two IP addresses separated by a single comma to the `X-Forwarded-For` header in the following order:

- The IP address of the client that connects to the load balancer
- The IP address of the load balancer's forwarding rule

If there is no `X-Forwarded-For` header on the incoming request, these two IP addresses are the entire header value:

```
X-Forwarded-For: <client-ip>,<load-balancer-ip>
```

If the request includes an `X-Forwarded-For` header, the load balancer preserves the supplied value *before* the `<client-ip>,<load-balancer-ip>`:

```
X-Forwarded-For: <supplied-value>,<client-ip>,<load-balancer-ip>
```

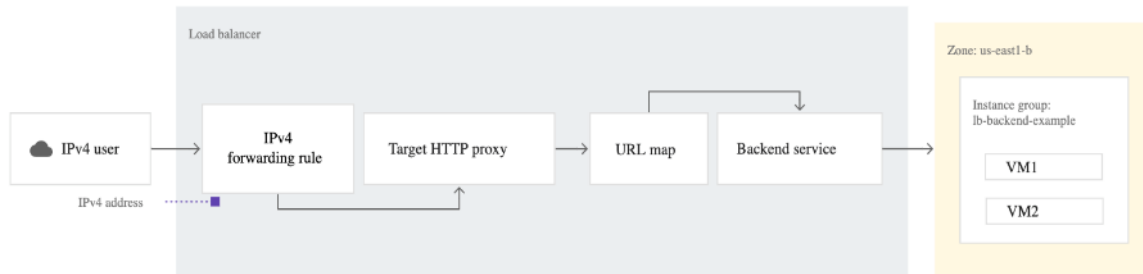
117

131. The GCP comprises a second network device for policy provisioning and access managing, wherein the second network device intercepting the incoming message prior to at least one first protocol server receives the incoming message. For example, the http(s) load balancer furnishes policy provisioning and access managing, e.g., load balancing policy provisioning and backend server access managing, according to IAM. The http(s) load balancer intercepts the incoming request before a backend server receives the request.

¹¹⁷ <https://cloud.google.com/load-balancing/docs/https>

HTTP load balancer topology

In this guide, you create the configuration that is illustrated in the following diagram.



Simple HTTP Load Balancing (click to enlarge)

The sequence of events in the diagram is as follows:

1. A client sends a content request to the external IPv4 address defined in the forwarding rule.
2. The forwarding rule directs the request to the target HTTP proxy.
3. The target proxy uses the rule in the URL map to determine that the single backend service receives all requests.
4. The load balancer determines that the backend service has only one instance group and directs the request to a virtual machine (VM) instance in that group.
5. The VM serves the content requested by the user.

118

Managing Access Control for Cloud Load Balancing using IAM

You can get and set IAM policies using the Google Cloud Console, the IAM API, or the `gcloud` command-line tool. See [Granting, changing, and revoking access to project members](#) for details.

119

External HTTP(S) Load Balancing is implemented on [Google Front Ends \(GFEs\)](#). GFEs are distributed globally and operate together using Google's global network and control plane. In the Premium Tier, GFEs offer cross-regional load balancing, directing traffic to the closest healthy backend that has capacity and terminating HTTP(S) traffic as close as possible to your users. With Standard Tier, the load balancing is handled regionally.

¹¹⁸ <https://cloud.google.com/load-balancing/docs/https/ext-http-lb-simple>

¹¹⁹ <https://cloud.google.com/load-balancing/docs/access-control>

Forwarding rules and addresses

[Forwarding rules](#) route traffic by IP address, port, and protocol to a load balancing configuration consisting of a target proxy, URL map, and one or more backend services.

Each forwarding rule provides a single IP address that can be used in DNS records for your application. No DNS-based load balancing is required. You can either specify the IP address to be used or let Cloud Load Balancing assign one for you.

- The forwarding rule for an HTTP load balancer can only reference TCP ports 80 and 8080.
- The forwarding rule for an HTTPS load balancer can only reference TCP port 443.

The type of forwarding rule required by external HTTP(S) load balancers depends on which [Network Service Tier](#) the load balancer is in.

- The external HTTP(S) load balancers in the Premium Tier use global external forwarding rules.
- The external HTTP(S) load balancers in the Standard Tier use regional external forwarding rules.

120

132. The GCP comprises a second a second network device for policy provisioning and access managing, wherein the second network device intercepting the incoming message prior to at least one first protocol server receives the incoming message, identifying a network device using the identifier and managing an assignment of a plurality of configuration settings based on the identifier. For example, the http(s) load balancer identifies a user/client device using the source (user/client) IP address, and accordingly manages an assignment of configuration settings (such as derived from the URL map configuration and/or session affinity) based on the source IP address.

¹²⁰ <https://cloud.google.com/load-balancing/docs/https>

Enabling session affinity

These procedures demonstrate how to configure a different type of session affinity for each backend service:

- Client IP address session affinity for `web-backend-service`
- HTTP cookie session affinity for `video-backend-service`

When client IP affinity is enabled, the load balancer directs a particular client's requests to the same backend VM based on a hash created from the client's IP address.

When generated cookie affinity is enabled, the load balancer issues a cookie on the first request. For each subsequent request with the same cookie, the load balancer directs the request to the same backend VM or endpoint. For external HTTP(S) load balancers, the cookie is named `GCLB`.

121

Enabling session affinity

These procedures demonstrate how to configure a different type of session affinity for each backend service:

- Client IP address session affinity for `web-backend-service`
- HTTP cookie session affinity for `video-backend-service`

When client IP affinity is enabled, the load balancer directs a particular client's requests to the same backend VM based on a hash created from the client's IP address.

When generated cookie affinity is enabled, the load balancer issues a cookie on the first request. For each subsequent request with the same cookie, the load balancer directs the request to the same backend VM or endpoint. For external HTTP(S) load balancers, the cookie is named `GCLB`.

122

133. The GCP comprises a database (e.g., a set of tables derived from the URL map configuration and/or session affinity) for storing a plurality of configuration information records (e.g., including a user/client IP address), wherein each record includes an identifier of a network access device and a plurality of configuration information settings (e.g., derived from the URL map configuration and/or session affinity) constructed based on a service level agreement (e.g., URL map-based routing and/or source IP session affinity) associated with the identifier of each record.

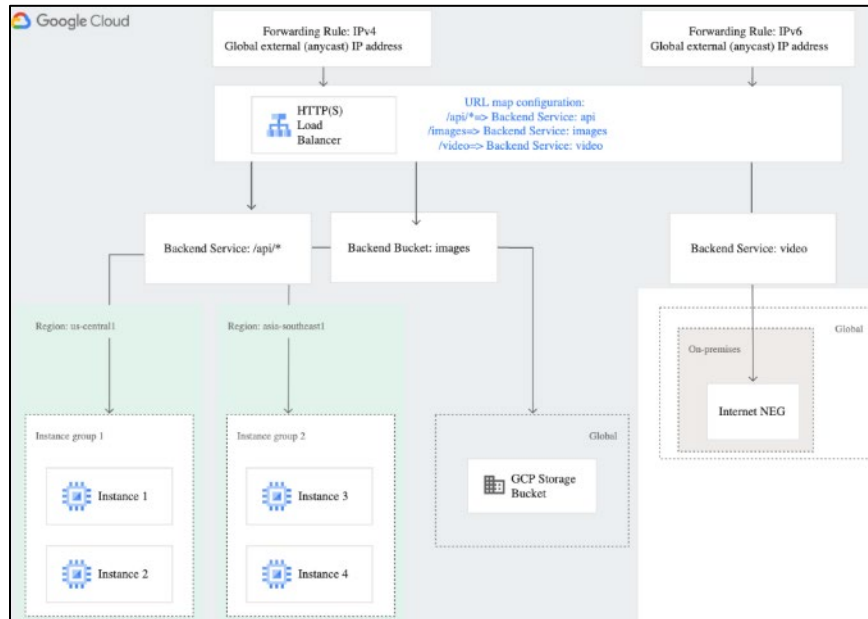
¹²¹ <https://cloud.google.com/load-balancing/docs/https/setting-up-https>

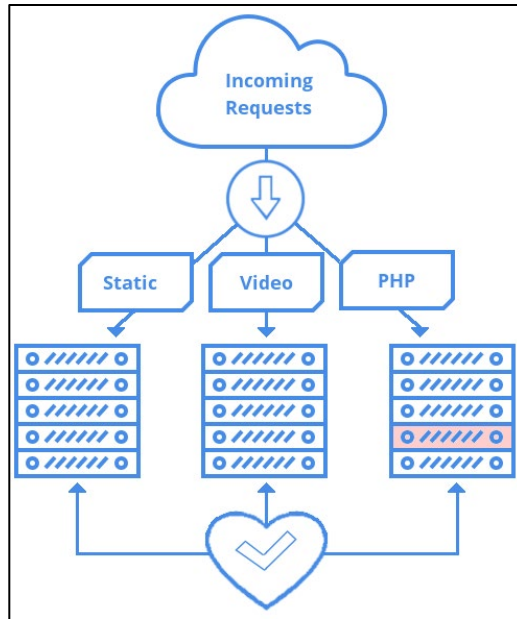
¹²² https://cloud.google.com/load-balancing/docs/backend-service#session_affinity

One common use case is load balancing traffic among services. In the following example, external IPv4 and IPv6 clients can request video, API, and image content by using the same base URL, with the paths `/api`, `/video`, and `/images`.

The external HTTP(S) load balancer's URL map specifies that:

- Requests to path `/api` go to a backend service with a VM [instance group](#) or a [zonal NEG](#) backend.
- Requests to path `/images` go to a [backend bucket](#) with a Cloud Storage backend.
- Requests to path `/video` go to a backend service that points to a [internet NEG](#) containing an external endpoint that is located on-premises outside of Google Cloud.





123

Enabling session affinity

These procedures demonstrate how to configure a different type of session affinity for each backend service:

- Client IP address session affinity for `web-backend-service`
- HTTP cookie session affinity for `video-backend-service`

When client IP affinity is enabled, the load balancer directs a particular client's requests to the same backend VM based on a hash created from the client's IP address.

When generated cookie affinity is enabled, the load balancer issues a cookie on the first request. For each subsequent request with the same cookie, the load balancer directs the request to the same backend VM or endpoint. For external HTTP(S) load balancers, the cookie is named `GCLB`.

124

Cookie-based affinity can more accurately identify a client to a load balancer, compared to client IP-based affinity. For example:

1. With cookie-based affinity, the load balancer can uniquely identify two or more client systems that share the same source IP address. Using client IP-based affinity, the load balancer treats all connections from the same source IP address as if they were from the same client system.
2. If a client changes its IP address, cookie-based affinity lets the load balancer recognize subsequent connections from that client instead of treating the connection as new. An example of when a client changes its IP address is when a mobile device moves from one network another.

125

¹²³ <https://cloud.google.com/load-balancing/docs/https>

¹²⁴ <https://cloud.google.com/load-balancing/docs/https/setting-up-https>

¹²⁵ https://cloud.google.com/load-balancing/docs/backend-service#session_affinity

134. Home Depot directly infringed the '326 Patent through its implementation and use of the GCP Cloud Load Balancing, performing at least the steps of claim 1.

135. Defendants have indirectly infringed one or more claims of the '326 Patent by knowingly and intentionally inducing others, including Home Depot customers and end-users, to directly infringe, either literally or under the doctrine of equivalents, by making, using, offering to sell, selling, and/or importing into the United States products that include the infringing technology.

136. Defendants, with knowledge that these products, or the use thereof, infringed the '326 Patent at least as of April 11, 2022¹²⁶, knowingly and intentionally induced direct infringement of the '326 Patent by providing these products to end-users for use in an infringing manner. Additionally, on information and belief, Defendants have adopted a policy of not reviewing the patents of others, including specifically those related to Defendants' specific industry, thereby remaining willfully blind to the '326 Patent at least as early as the issuance of the '326 Patent.

137. Defendants have induced infringement by others, including end-users, with the intent to cause infringing acts by others or, in the alternative, with the belief that there was a high probability that others, including end-users, infringe the '326 Patent, but while remaining willfully blind to the infringement. Defendants have induced infringement by their customers and end-users by supplying them with instructions on how to operate the infringing technology in an infringing

¹²⁶ Plaintiffs sent Defendants correspondence in April 2022 notifying them of their infringement of the '326 Patent. Plaintiffs sent a follow-up letter to the April 2022 correspondence dated December 19, 2022. In addition to these two letters, Plaintiffs sent multiple emails to Defendants regarding at least some of the Asserted Patents, including on April 20, 2023, May 26, 2023, November 16, 2023, and February 8, 2024. To date, Defendants have not responded to any of Plaintiffs letters or emails.

manner, while also making publicly available information on the infringing technology via Defendants' website, product literature and packaging, and other publications.¹²⁷

138. Plaintiffs have suffered damages as a result of Defendants' direct and indirect infringement of the '326 Patent in an amount to be proven at trial.

139. Defendants have committed acts of infringement that Defendants actually knew or should have known constituted an unjustifiably high risk of infringement of at least one valid and enforceable claim of the '326 Patent. Defendants' direct and indirect infringement of the '326 Patent was willful, intentional, deliberate, and/or in conscious disregard of rights under the patent. Plaintiffs are entitled to an award of treble damages, reasonable attorney fees, and costs in bringing this action.

DEMAND FOR JURY TRIAL

Plaintiffs hereby demands a jury for all issues so triable.

PRAYER FOR RELIEF

WHEREFORE, Plaintiffs pray for relief against Defendants as follows:

- a. Entry of judgment declaring that Defendants have directly and/or indirectly infringed one or more claims of each of the Patents-in-Suit;
- b. Entry of judgment declaring that Defendants' infringement of the Patents-in-Suit is willful;
- c. An order pursuant to 35 U.S.C. § 283 permanently enjoining Defendants, their officers, agents, servants, employees, attorneys, and those persons in active concert or participation with them, from further acts of infringement of the '686 Patent, and the '332 Patent;

¹²⁷ See, e.g., https://www.homedepot.com/c/customer_service; https://www.homedepot.com/c/SF_Mobile_Shopping

d. An order awarding damages sufficient to compensate Plaintiffs for Defendants' infringement of the Patents-in-Suit, but in no event less than a reasonable royalty, together with interest and costs;

e. Entry of judgment declaring that this case is exceptional and awarding Plaintiffs their costs and reasonable attorney fees under 35 U.S.C. § 285; and

f. Such other and further relief as the Court deems just and proper.

Dated: January 27, 2025

Respectfully submitted,

/s/ Vincent J. Rubino, III

Alfred R. Fabricant

NY Bar No. 2219392

Email: ffabricant@fabricantllp.com

Peter Lambrianakos

NY Bar No. 2894392

Email: plambrianakos@fabricantllp.com

Vincent J. Rubino, III

NY Bar No. 4557435

Email: vrubino@fabricantllp.com

FABRICANT LLP

411 Theodore Fremd Avenue

Suite 206 South

Rye, New York 10580

Telephone: (212) 257-5797

Facsimile: (212) 257-5796

***ATTORNEYS FOR PLAINTIFFS
VALTRUS INNOVATIONS LTD. and
KEY PATENT INNOVATIONS LTD.***