UNITED STATES DISTRICT COURT
FOR THE NORTHERN DISTRICT OF ILLINOIS
EASTERN DIVISION

Kove IO, Inc., Plaintiff, v. Amazon Web Services, Inc., Defendant.	Civil Action No. COMPLAINT **DEMAND FOR JURY TRIAL**

COMPLAINT

Plaintiff Kove IO, Inc. ("Kove") files this Complaint against Defendant Amazon Web Services, Inc. ("AWS"), and in support thereof states as follows:

NATURE OF THE ACTION

1. Kove's inventors developed breakthrough technology enabling high-performance, hyper-scalable distributed "cloud" storage years before the advent of the cloud. They were awarded patents for their innovations. Five years after this breakthrough, the *MIT Technology Review* recognized that the type of distributed data storage technology described in Kove's patents was one of the top 10 emerging technologies that would change the world.[1]

2. Kove's technology became essential to AWS as the volume of data stored on its cloud grew exponentially and its cloud storage business faced limitations on the ability to store and retrieve massive amounts of data. An AWS engineer explained that "we have a history of 10 years of using distributed hash tables to run systems and that becomes very powerful and that

[1] Balakrishnan, H., Distributed Storage, 10 Emerging Technologies That Will Change Your World, MIT Technology Review February 2004 (Attached as Ex. 1).

algorithm makes things work at massive scales and pretty much nothing else does."[2] Once able to remove the barriers to massive data storage and retrieval present in previous systems, AWS became the first large-scale vendor of economical cloud infrastructure and services, allowing businesses and individuals around the world access to the cloud without having to set up their own servers, software, and functionality. The ability to offer cloud services on this scope and scale was made possible through infringement of Kove's patents, paving the way for AWS to become what is believed to be Amazon's largest profit center.[3]

3. Kove is a Chicago-based pioneer in high-performance computer storage and data management technologies. It sells hardware and data management technology, creating solutions that overcome technological limitations in modern server architecture. It is a small, innovative product company competing in a field of behemoths, like AWS. It cannot outspend these dominant global players—respect for its intellectual property, as the law requires, is essential to fair competition. Companies such as AWS have little incentive to do business with small companies that have patented (and therefore disclosed) technology if they are able to take it without meaningful consequences. The disclosure of innovation in patents is not intended to facilitate unauthorized use, but rather to incentivize public disclosure for the benefit of all, in return for the promise to inventors of exclusive rights for a limited period of time.

4. Through this case, Kove seeks to stop AWS's unauthorized use of Kove's patented technology, and its noncompliance with the patent laws. This case is about ensuring a level playing

[2] AWS re:Invent 2017: How DynamoDB Powered Amazon Prime Day 2017 (DAT326), YouTube (November 30, 2017), available at https://www.youtube.com/watch?v=83-IWlvJ__8&feature=youtu.be&t=38m38s.

[3] Wingfield, N., Amazon's Profit Swells to $1.6 Billion, Lifted by Its Cloud Business, New York Times, (April 26, 2018), available at https://www.nytimes.com/2018/04/26/technology/amazon-prime-profit.html (Attached as Ex. 2).

field so smaller competitors like Kove can compete fairly on the basis of their hard work and protected innovations.

THE PARTIES

5. Kove was founded in 2004 with an emphasis on high performance storage solutions. Kove has maintained a lean team of no more than 50 employees over the course of its history. The Kove team has included those with strong backgrounds from premier tech companies, including Intel, Sun, Cisco, Seagate, Quantum, NASA, and many others. Kove's headquarters are located in Chicago, Illinois. It owns the patents asserted in this case. Dr. John Overton and Dr. Stephen Bailey are the named inventors, and Dr. Overton serves as Kove's Chief Executive Officer.

6. Kove is a corporation organized and existing under the laws of the State of Delaware and is registered to do business in the State of Illinois. Kove's principal place of business is located at 14 North Peoria Street Suite 2H, Chicago, Illinois 60607.

7. AWS is a corporation organized and existing under the laws of the State of Delaware and is registered to do business in the State of Illinois. Upon information and belief, AWS has its principal place of business at 410 Terry Avenue North Seattle, Washington 98109 and a regular and established place of business in this District at AT&T Center, 227 W. Monroe Street, Chicago, Illinois 60606.

8. AWS develops and provides cloud storage products and services to customers, including customers in this District.

JURISDICTION AND VENUE

9. This Court has subject matter jurisdiction over this case under 28 U.S.C. §§ 1331 and 1338. Venue is proper in this Court under 28 U.S.C. §§ 1391 and 1400(b).

10. This Court has personal jurisdiction over AWS. AWS has continuous and systematic business contacts with the State of Illinois. AWS, directly and/or through subsidiaries or intermediaries, conducts its business extensively throughout Illinois, by shipping, distributing, offering for sale, selling, and advertising (including the provision of interactive web pages) its products and services in the State of Illinois and in this District. AWS, directly and/or through subsidiaries or intermediaries, has purposefully and voluntarily placed its infringing products and services into this District and into the stream of commerce with the intention and expectation that they will be purchased and used by consumers in this District. AWS has offered and sold and continues to offer and sell these infringing products and services in this District.[4] On information and belief, AWS, for example, sells and offers to sell the infringing products and services to developers, partners or, customers in this District, such as Amazon.com, Inc., University of Chicago, City of Chicago, and Federal Home Loan Bank of Chicago. AWS has committed acts of infringement in this District and has a regular and established place of business in this District.

BACKGROUND

11. Drs. Overton and Bailey, who met at the University of Chicago while working on their PhDs, are the named inventors of the three patents-in-suit: U.S. Patent Nos. 7,814,170 ("'170 Patent"); 7,103,640 ("'640 Patent"); and 7,233,978 ("'978 Patent") (collectively, "patents-in-suit").

12. Today, cloud storage is so ubiquitous that it has become part of the common vernacular. When the inventors were studying together at the University of Chicago in the 1990's, the cloud as we know it was years away. The inventors foresaw the advent of the cloud—

[4] AWS, Customer Success Stories, available at https://aws.amazon.com/solutions/case-studies/startups/ and https://aws.amazon.com/solutions/case-studies/government-education/all-government-education-nonprofit/ (Attached as Ex. 3).

distributed, large-scale storage networks—and they identified a key roadblock that would need to be overcome in order to sustain viability.

13. In particular, data storage management has become increasingly critical for companies as the amount of data produced every day grows exponentially—for example, today, there are 2.5 quintillion ($2.5 * 10^{18}$) bytes of data created each day.[5] In the 1990's, the inventors foresaw that data storage requirements would grow beyond the capabilities of conventional computer networks. Dependency on traditional ways of indexing the information in a distributed storage network was identified as a key issue. For example, as information was added to a network, previous technology would require that it be indexed in some hierarchical fashion, and those indices would periodically have to be updated. While these indices worked well to a certain point, the distributed network of data servers and data would become so vast and complex that ordinary hierarchical indices no longer could handle the load without high levels of inefficiency.

14. Further, to store a piece of information (e.g., a data file), a storage system must store not only the data file itself but also its corresponding location information, which records where the data file is located on the network of servers and computers. Without the corresponding location information, a storage system would not know where to find the data file (i.e., which server on the network to access and from which to retrieve the data file).

15. Location information of data files was traditionally stored on a single centralized server, and it was retrieved from that specific centralized server. Thus, then-existing systems allowed a client seeking location information associated with a data file to send a request to a

[5] Bernard Marr, How Much Data Do We Create Every Day? The Mind-Blowing Stats Everyone Should Read (May 21, 2018), available at https://www.forbes.com/sites/bernardmarr/2018/05/21/how-much-data-do-we-create-every-day-the-mind-blowing-stats-everyone-should-read/#2329790360ba.

server, and typically, only data statically associated with that server was returned. The search was conducted only where the system knew in advance to look.

16. In the 1990's, such a server may have been sufficient; however, the inventors knew distributed storage systems would someday contain so many unique data files that it would become impractical—if not impossible—to store the corresponding location information in one place.

17. Drs. Overton and Bailey addressed these challenges. They realized, among other things, that storing location information associated with data files across multiple servers would reduce the processing time to find a data file. Likewise, the inventors understood the need to efficiently identify which of the multiple location information servers stored the location information for a particular data file.

18. To achieve this, Drs. Overton and Bailey invented the claimed technology of the patents-in-suit. For example, hash values corresponding to the location information of data files would be stored and distributed on the network. The distributed hash values would reveal which of the location information servers contained the location information of particular data files. Each hash value may point to the location information server storing the location information of a data file. Each location server would have the ability to point to other servers based on the hash values. While the first server may not have the location information for the file the user is looking for, it will reroute—within milliseconds—to another server that does have that location information. This fundamental technology allowed for the efficient organization of location information even as files are constantly added, deleted, modified, and moved. It enabled hyper-scalable cloud storage and improved upon the scalability limitations of conventional storage systems.

19. The claimed inventions were ahead of their time—distributed storage networks had not yet grown large enough. Years after the inventors filed the earliest of the original patent

applications, an article published in the *MIT Technology Review* described the use of distributed hash tables (as discussed above) as one of "10 Emerging Technologies That Will Change Your World," explaining that "[h]ash tables provide a quick way to organize data: a simple mathematical operation assigns each file its own row in a table; the row stores the file's location," and that "distributed hash tables [are] the coming future of networked storage" and have a "huge variety of applications. . . . Not very many technologies have such broad potential."[6]

THE KOVE PATENTS-IN-SUIT

20. On September 5, 2006, the U.S. Patent and Trademark Office duly and legally issued the '640 patent, entitled "Network Distributed Tracking Wire Transfer Protocol," with Drs. Overton and Bailey as inventors. The earliest application related to the '640 patent was filed on July 8, 1998. A true and correct copy of the '640 patent is attached as Exhibit 4.

21. On June 19, 2007, the U.S. Patent and Trademark Office duly and legally issued the '978 patent, entitled "Method and Apparatus for Managing Location Information in a Network Separate From the Data to Which the Location Information Pertains," with Drs. Overton and Bailey as inventors. The earliest application related to the '978 patent was filed on July 8, 1998. A true and correct copy of the '978 patent is attached as Exhibit 5.

22. On October 12, 2010, the U.S. Patent and Trademark Office duly and legally issued the '170 patent, entitled "Network Distributed Tracking Wire Transfer Protocol," with Drs. Overton and Bailey as inventors. The earliest application related to the '170 patent was filed on July 8, 1998. A true and correct copy of the '170 patent is attached as Exhibit 6.

23. Kove is the sole and exclusive owner of all rights, title, and interest to the patents-in-suit necessary to bring this action, including the right to recover past and future damages.

[6] *See supra* n. 1. (Balakrishnan, Ex. 1).

24. The patents-in-suit are enforceable and valid.

AWS'S INFRINGING PRODUCTS AND SERVICES

25. Upon information and belief, AWS has infringed and continues to infringe, directly and indirectly, one or more claims of the patents-in-suit.

26. The accused products include without limitation cloud-based products and services provided by AWS, such as Amazon Simple Storage Service ("Amazon S3"), DynamoDB, and other related products and services (collectively, the "AWS Accused Products").

27. As an example, Amazon S3 allows users to store data in the cloud. In fact, it is described as "storage for the internet." Specifically, it launched in 2006 and was designed to provide highly scalable, reliable, and low-latency data storage infrastructure at very low costs, where users are able to "[w]rite, read, and delete objects containing from 1 byte to 5 gigabytes of data each" and where "[e]ach object is stored and retrieved via a unique developer-assigned key."[7] In the first seven years of operation, Amazon S3 experienced a 20,000% increase in the number of data objects stored, reaching 2 trillion ($2*10^{12}$) data objects by 2013.[8] Amazon S3 is considered "one of the wonders" of AWS.[9]

28. As another example, DynamoDB is described as a "fully managed NoSQL database service that provides fast and predictable performance with seamless scalability. DynamoDB lets

[7] Amazon Web Services Launches, Press Release (March 14, 2006), at *1, available at http://phx.corporate-ir.net/phoenix.zhtml?c=176060&p=irol-newsArticle&ID=830816 (Attached as Ex. 7); Feloni, R., How A Regular Employee Helped Put Amazon On The Path To Billions Of Dollars, Business Insider (Jul. 22, 2014), available at http://www.businessinsider.com/benjamin-black-and-amazon-web-services-2014-7 (Attached as Ex. 8).

[8] Barr, J., AWS News Blog, Amazon S3 – Two Trillion Objects, 1.1 Million Requests / Second, available at https://aws.amazon.com/blogs/aws/amazon-s3-two-trillion-objects-11-million-requests-second/ (Attached as Ex. 9).

[9] Hern, A., Amazon Web Services: the secret to the online retailer's future success, The Guardian (February 2, 2017), available at https://www.theguardian.com/technology/2017/feb/02/amazon-web-services-the-secret-to-the-online-retailers-future-success (Attached as Ex. 10).

you offload the administrative burdens of operating and scaling a distributed database, so that you don't have to worry about hardware provisioning, setup and configuration, replication, software patching, or cluster scaling. With DynamoDB, you can create database tables that can store and retrieve any amount of data, and serve any level of request traffic. You can scale up or scale down your tables' throughput capacity without downtime or performance degradation, and use the AWS Management Console to monitor resource utilization and performance metrics."[10]

29. "DynamoDB powers the next wave of high-performance, internet-scale applications. . . . DynamoDB is used by Lyft to store GPS locations for all their rides, Tinder to store millions of user profiles and make billions of matches, Redfin to scale to millions of users and manage data for hundreds of millions of properties, Comcast to power their XFINITY X1 video service running on more than 20 million devices, BMW to run its car-as-a-sensor service that can scale up and down by two orders of magnitude within 24 hours, Nordstrom for their recommendations engine reducing processing time from 20 minutes to a few seconds, Under Armour to support its connected fitness community of 200 million users, Toyota Racing to make real time decisions on pit-stops, tire changes, and race strategy, and another 100,000+ AWS customers for a wide variety of high-scale, high-performance use cases."[11]

30. On information and belief, AWS uses the AWS Accused Products for its own business purposes. In addition, AWS regularly conducts testing and troubleshooting of the AWS

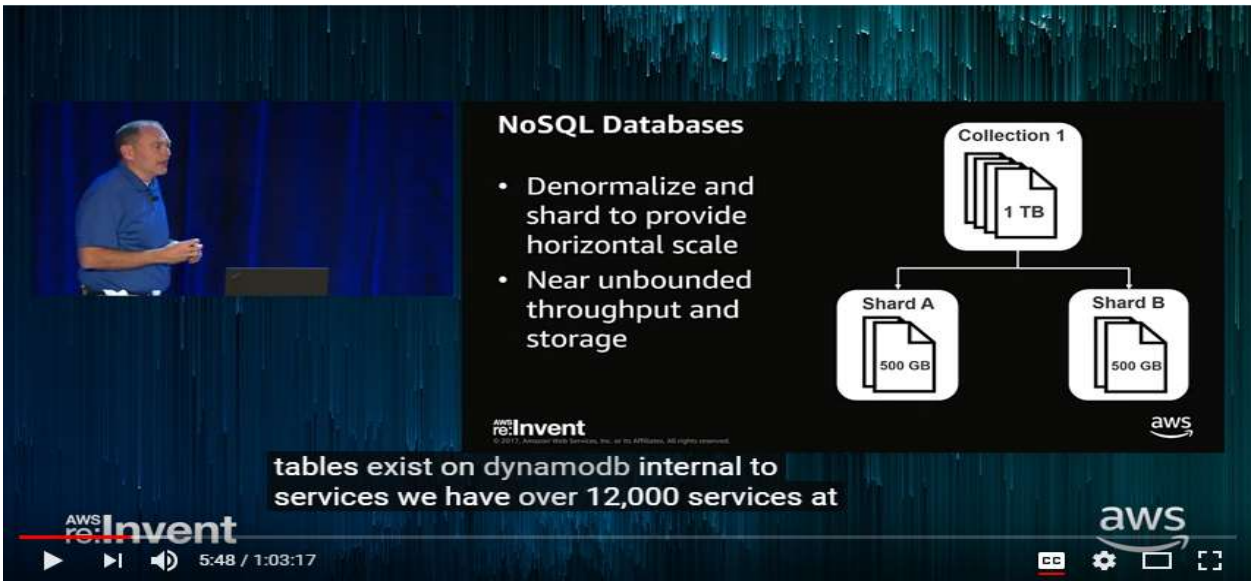
---

[10] DynamoDB Developer Guide (API Version 2012-08-10), at *1, available at https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/dynamodb-dg.pdf (While the full document is incorporated by reference herein, an excerpted version of the document is attached as Ex. 11); *see also* Amazon's Web Version of the DynamoDB Developer Guide at https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/Introduction.html.

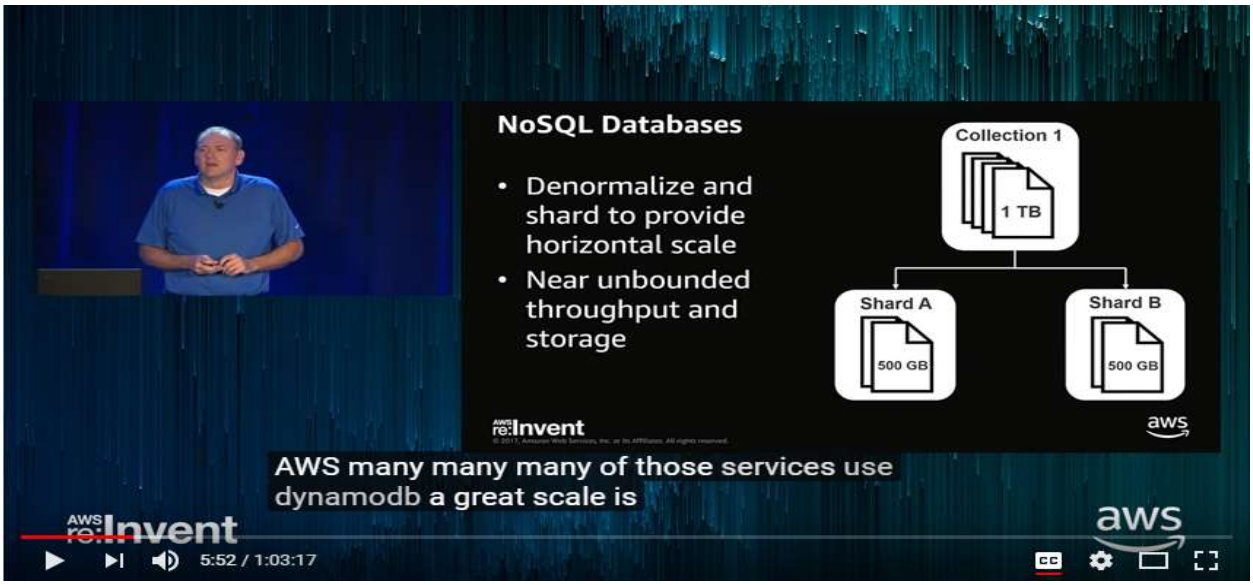
[11] Vogels, W., A Decade of Dynamo: Powering the net wave of high-performance, internet-scale applications, All Things Distributed, at *2, available at https://www.allthingsdistributed.com/2017/10/a-decade-of-dynamo.html (Attached as Ex. 12).

Accused Products. Further, Kove is informed and believes companies related to AWS (e.g., Amazon.com, Inc. and its subsidiaries) use the AWS Accused Products.

31. On information and belief, the features discussed herein are not limited to any one of the AWS Accused Products. For example, as an AWS senior practice manager explained: "[T]ables exist on DynamoDB internal to services we have over 12,000 services at AWS many many of those services use DynamoDB . . . ."[12]



[12] AWS Youtube Videos, AWS re:Invent 2017: DynamoDB adaptive capacity: smooth performance for chaotic workl (DAT327), available at https://youtu.be/kMY0_m29YzU.



32. Indeed, on information and belief, the AWS Accused Products are based on a common platform, like the "Dynamo" technology: "Dynamo is internal technology developed at Amazon to address the need for an incrementally scalable, highly-available key-value storage system. The technology is designed to give its users the ability to trade-off cost, consistency, durability and performance, while maintaining high-availability."[13] Dynamo is used "to power parts of [AWS products], such as S3."[14] DynamoDB too is based on Dynamo technology and is built on AWS products such as S3.[15]

33. On information and belief, AWS offers an easily integrated platform, consisting of the AWS Accused Products, that uses the features discussed herein as shown below:[16]

---

[13] Vogels, W., Amazon's Dynamo, All Things Distributed, at *1, available at https://www.allthingsdistributed.com/2007/10/amazons_dynamo.html (Attached as Ex. 13).

[14] *Id.*

[15] Vogels, W., Amazon DynamoDB – a Fast and Scalable NoSQL Database Service Designed for Internet Scale Applications, All Things Distributed, at *1, available at https://www.allthingsdistributed.com/2012/01/amazon-dynamodb.html (Attached as Ex. 14).

[16] Agnihotri, P., Amazon Web Services – Backup, Archive and Restore Approaches Using AWS, at *5, *16 available at https://d0.awsstatic.com/whitepapers/Backup_Archive_and_Restore_Approaches_Using_AWS.pdf (Attached as Ex. 15).

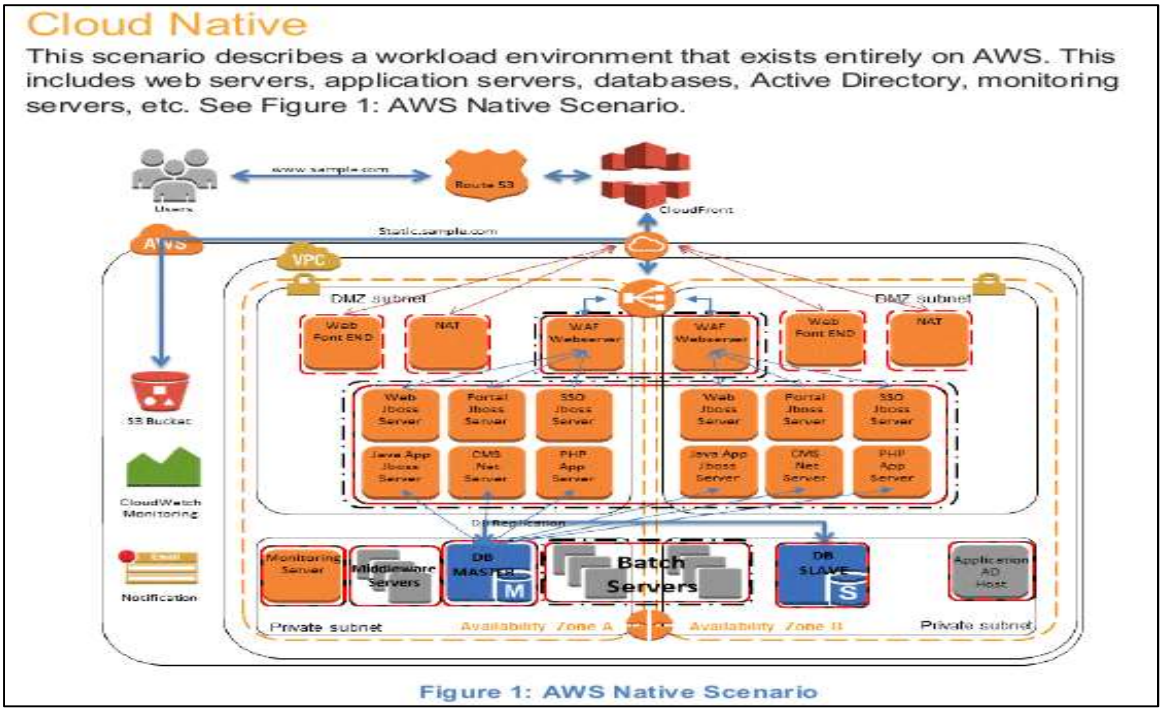


Figure 1: AWS Native Scenario

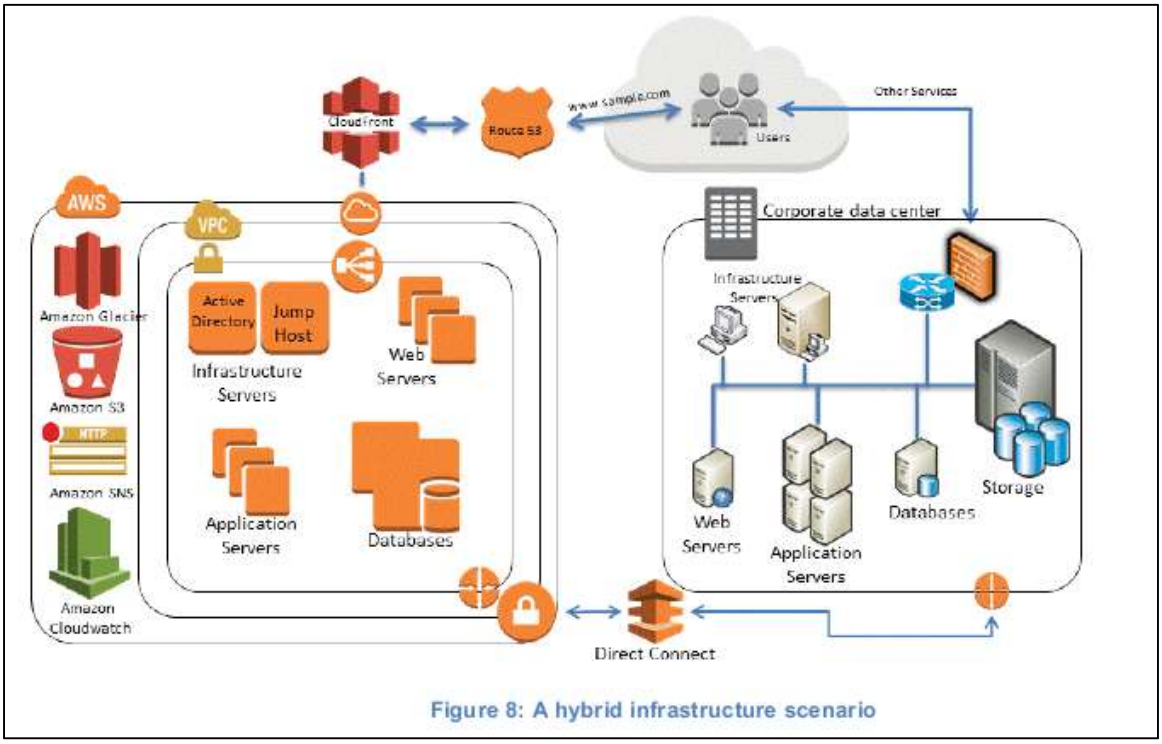


Figure 8: A hybrid infrastructure scenario

COUNT I

AWS'S INFRINGEMENT OF THE '170 PATENT

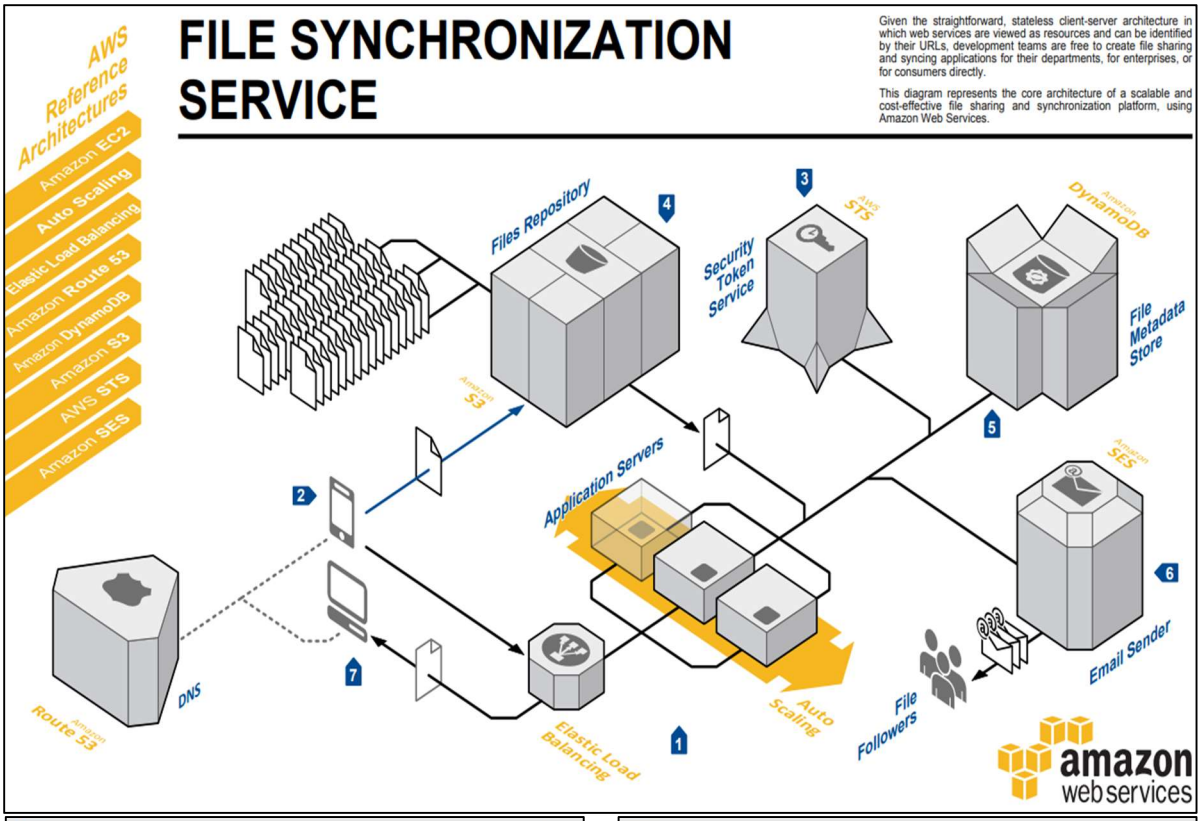
34. Kove incorporates by reference the foregoing paragraphs of this Complaint as if fully set forth herein.

35. On information and belief, AWS has infringed and will continue to infringe the '170 patent. AWS directly infringes the '170 patent under 35 U.S.C. §271(a) by making, using, selling, offering for sale, and/or importing in this District and into the United States products and/or methods covered by one or more claims of the '170 patent, including, but not limited to, the AWS Accused Products. As an example, the AWS Accused Products infringe at least claim 1 of the '170 patent.

36. Claim 1 is directed to a "system for managing data stored in a distributed network." Claim 1, for example, recites a "data repository configured to store a data entity, wherein an identifier string identifies the data entity." Claim 1 further recites a "data location server network" where "data location information for a plurality of data entities is stored." Further still, claim 1 recites a "hash function used to organize the data location information across the . . . data location servers."

37. The AWS Accused Products manage data stored in a distributed network. For example, as shown below in the Amazon S3 Reference Architecture diagram ("Architecture Diagram"), "[u]sers upload files into Amazon Simple Storage Service (Amazon S3), a highly durable storage infrastructure designed for mission-critical and primary data storage. Amazon S3 makes it easy to store and retrieve any amount of data, at any time."[17]

[17]Amazon Web Services, Inc., https://media.amazonwebservices.com/architecturecenter/AWS_ac_ra_filesync_08.pdf (modified from original) Attached as Ex. 16).



1 The file synchronization service endpoint consists of an **Elastic Load Balancer** distributing incoming requests to a group of application servers hosted on **Amazon Elastic Compute Cloud** (Amazon EC2) instances. An **Auto Scaling** group automatically adjusts the number of **Amazon EC2** instances depending on the application needs.

2 To upload a file, a client first needs to request the permission to the service and get a security token.

3 After checking the user's identity, application servers get a temporary credential from **AWS Security Token Service** (STS). This credential allows users to upload files.

4 Users upload files into **Amazon Simple Storage Service** (Amazon S3), a highly durable storage infrastructure designed for mission-critical and primary data storage. **Amazon S3** makes it easy to store and retrieve any amount of data, at any time. Large files can be uploaded by the same client using multiple concurrent threads to maximize bandwidth usage.

5 File metadata, version information, and unique identifiers are stored by the application servers on an **Amazon DynamoDB** table. As the number of files to maintain in the application grows, **Amazon DynamoDB** tables can store and retrieve any amount of data, and serve any level of traffic.

6 File change notifications can be sent via email to users following the resource with **Amazon Simple Email Service** (Amazon SES), an easy-to-use, cost-effective email solution.

7 Other clients sharing the same files will query the service endpoint to check if newer versions are available. This query compares the list of local files checksums with the checksums listed in an **Amazon DynamoDB** table. If the query finds newer files, they can be retrieved from **Amazon S3** and sent to the client application.

38. AWS stores data entities, wherein an identifier string identifies the data entities. In particular, Amazon S3 stores objects in a "Files Repository," and each Amazon S3 object, is "uniquely identified within a bucket by a key (name) and a version ID. . . . Amazon S3 can be thought of as a basic data map between 'bucket + key + version' and the object itself. Every object in Amazon S3 can be uniquely addressed through the combination of the web service endpoint, bucket name, key, and optionally, a version."[18]

39. On information and belief, AWS includes a data location server network where data location information for a plurality of data entities is stored. For example, Amazon S3 maintains an index subsystem that manages the metadata and location information of all S3 objects in the data location servers as illustrated below.[19]

> We'd like to give you some additional information about the service disruption that occurred in the Northern Virginia (US-EAST-1) Region on the morning of February 28th, 2017. The Amazon Simple Storage Service (S3) team was debugging an issue causing the S3 billing system to progress more slowly than expected. At 9:37AM PST, an authorized S3 team member using an established playbook executed a command which was intended to remove a small number of servers for one of the S3 subsystems that is used by the S3 billing process. Unfortunately, one of the inputs to the command was entered incorrectly and a larger set of servers was removed than intended. The servers that were inadvertently removed supported two other S3 subsystems. One of these subsystems, the index subsystem, manages the metadata and location information of all S3 objects in the region. This subsystem is necessary to serve all GET, LIST, PUT, and DELETE requests. The second subsystem, the placement subsystem, manages allocation of new storage and requires the index subsystem to be functioning properly to correctly operate. The placement subsystem is used during PUT requests to allocate storage for new objects. Removing a significant portion of the capacity caused each of these systems to require a full restart. While these subsystems were being restarted, S3 was unable to service requests. Other AWS services in the US-EAST-1 Region that rely on S3 for storage, including the S3 console, Amazon Elastic Compute Cloud (EC2) new instance launches, Amazon Elastic Block Store (EBS) volumes (when data was needed from a S3 snapshot), and AWS Lambda were also impacted while the S3 APIs were unavailable.

40. On information and belief, Amazon S3 stores data location information for S3 objects in a DynamoDB table.[20] It has been stated that "[t]he heart of the S3 object index is a

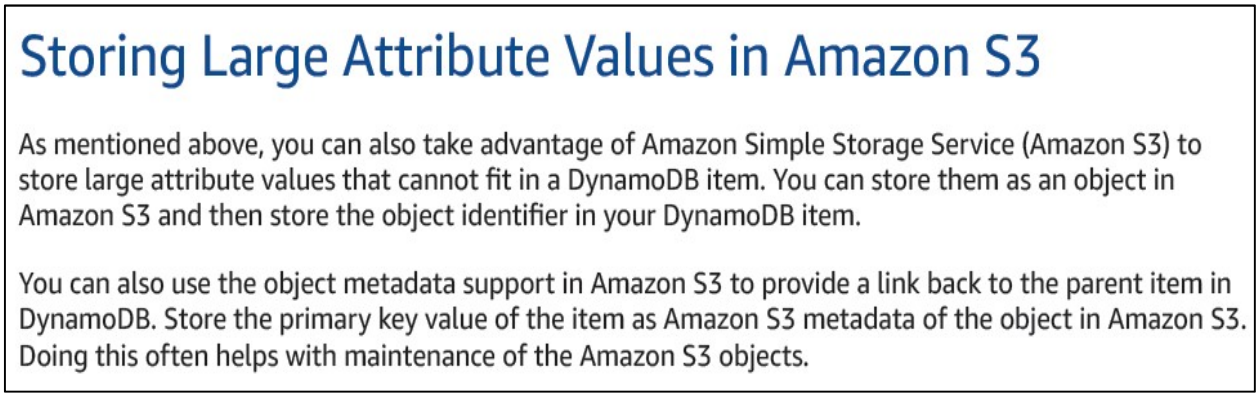
---

[18] Amazon Simple Storage Service Developer Guide, (API Version 2006-03-01), at *3–4, available at http://docs.aws.amazon.com/AmazonS3/latest/dev/s3-dg.pdf (While the full document is incorporated by reference herein, an excerpted version of the document is attached as Ex. 17); *see also* Amazon's Web Version of the S3 Developer Guide at https://docs.aws.amazon.com/AmazonS3/latest/dev/Welcome.html.

[19] Summary of the Amazon S3 Service Disruption in the Northern Virginia (US-EAST-1) Region, available at https://aws.amazon.com/message/41926/ (emphasis added) (Attached as Ex. 18).

[20] *See supra* n. 10 at *523 (DynamoDB Developer Guide, Ex. 11).

DynamoDB table with one item per object, which associates various attributes with the object's S3 key. Each item contains the S3 key, the size of the object, and any additional attributes to use for lookups."[21]



Storing Large Attribute Values in Amazon S3

As mentioned above, you can also take advantage of Amazon Simple Storage Service (Amazon S3) to store large attribute values that cannot fit in a DynamoDB item. You can store them as an object in Amazon S3 and then store the object identifier in your DynamoDB item.

You can also use the object metadata support in Amazon S3 to provide a link back to the parent item in DynamoDB. Store the primary key value of the item as Amazon S3 metadata of the object in Amazon S3. Doing this often helps with maintenance of the Amazon S3 objects.

41. AWS's "Frequently Asked Questions About Amazon DynamoDB" explains that "file pointers (possibly to Amazon S3 objects) are best saved in Amazon DynamoDB," as opposed to in Amazon S3.[22] AWS allows "the object metadata support in Amazon S3 to store the primary key value of the corresponding item as Amazon S3 object metadata. This use of metadata can help with future maintenance of your Amazon S3 objects."[23]

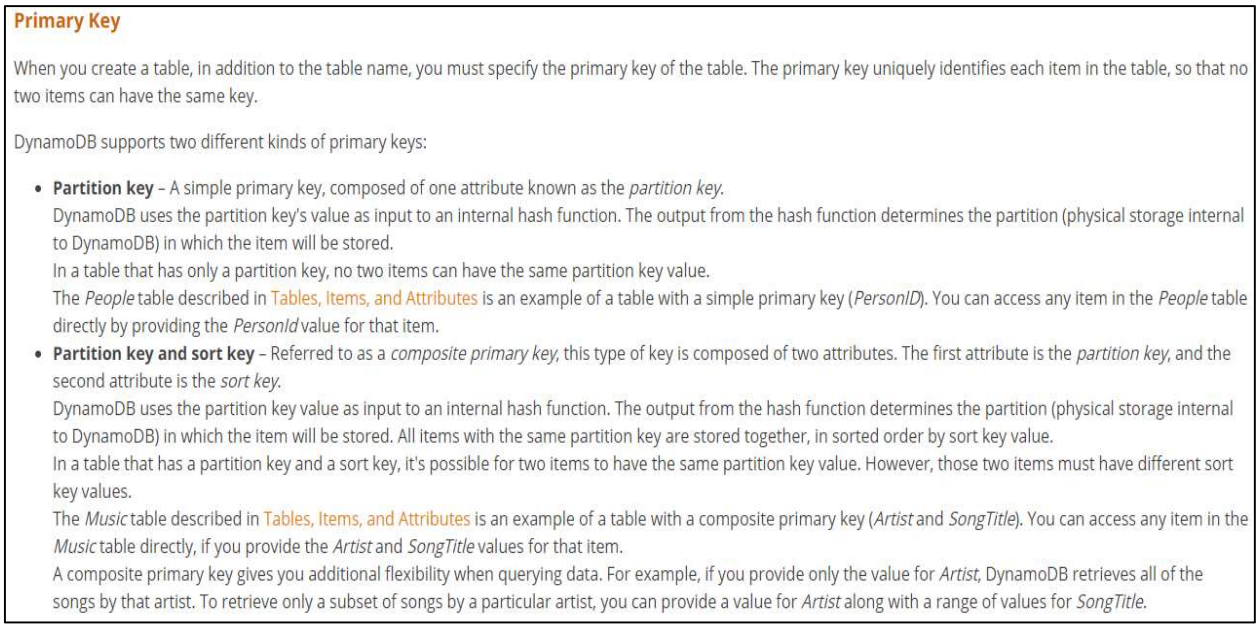
42. On information and belief, AWS uses a hash function to organize data location information across the data location servers. For example, DynamoDB tables organize data location information across servers based on a hash function. AWS has described DynamoDB as a system that "uses its hash function to determine where to store a new item . . . . Note that the

[21] Deck, M., AWS Big Data Blog, Building and Maintaining an Amazon S3 Metadata Index without Servers, available at https://aws.amazon.com/blogs/big-data/building-and-maintaining-an-amazon-s3-metadata-index-without-servers/ (Attached as Ex. 19).

[22] Amazon DynamoDB FAQs, at *4–5, available at https://aws.amazon.com/dynamodb/faqs/ (Attached as Ex. 20).

[23] *See supra* n. 10 at *523 (DynamoDB Developer Guide, Ex. 11).

items are not stored in sorted order. Each item's location is determined by the hash value of its partition key."[24]



Primary Key

When you create a table, in addition to the table name, you must specify the primary key of the table. The primary key uniquely identifies each item in the table, so that no two items can have the same key.

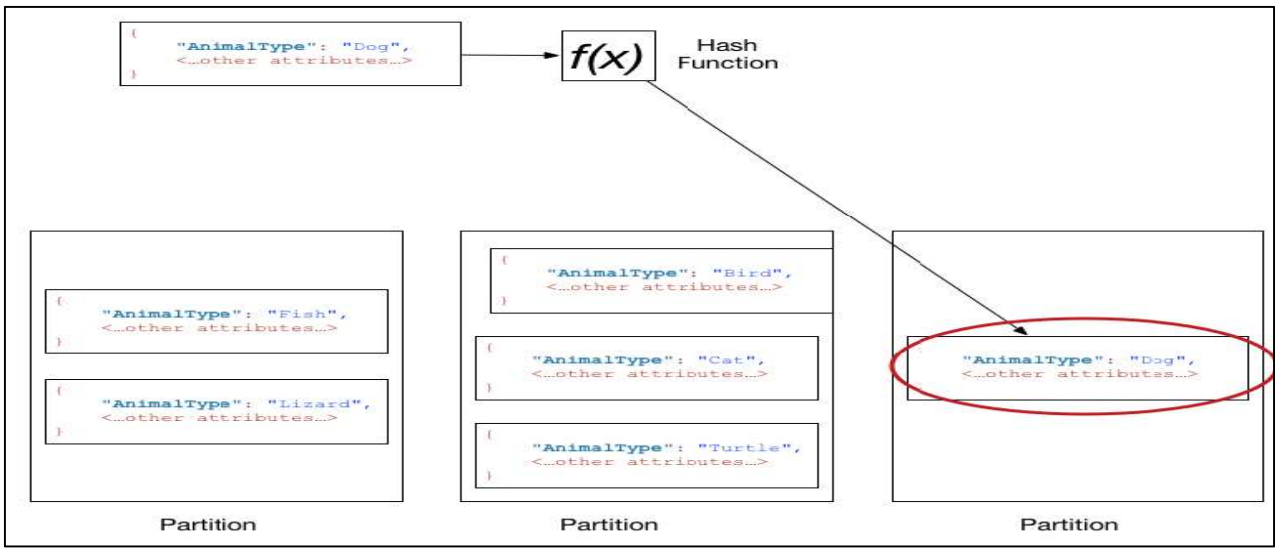
DynamoDB supports two different kinds of primary keys:

- **Partition key** – A simple primary key, composed of one attribute known as the *partition key*.
  DynamoDB uses the partition key's value as input to an internal hash function. The output from the hash function determines the partition (physical storage internal to DynamoDB) in which the item will be stored.
  In a table that has only a partition key, no two items can have the same partition key value.
  The *People* table described in Tables, Items, and Attributes is an example of a table with a simple primary key (*PersonID*). You can access any item in the *People* table directly by providing the *PersonId* value for that item.
- **Partition key and sort key** – Referred to as a *composite primary key*, this type of key is composed of two attributes. The first attribute is the *partition key*, and the second attribute is the *sort key*.
  DynamoDB uses the partition key value as input to an internal hash function. The output from the hash function determines the partition (physical storage internal to DynamoDB) in which the item will be stored. All items with the same partition key are stored together, in sorted order by sort key value.
  In a table that has a partition key and a sort key, it's possible for two items to have the same partition key value. However, those two items must have different sort key values.
  The *Music* table described in Tables, Items, and Attributes is an example of a table with a composite primary key (*Artist* and *SongTitle*). You can access any item in the *Music* table directly, if you provide the *Artist* and *SongTitle* values for that item.
  A composite primary key gives you additional flexibility when querying data. For example, if you provide only the value for *Artist*, DynamoDB retrieves all of the songs by that artist. To retrieve only a subset of songs by a particular artist, you can provide a value for *Artist* along with a range of values for *SongTitle*.

43. As AWS illustrates in its DynamoDB Developer Guide, "[t]he following diagram [below] shows a table named PETS, which spans multiple partitions. The table's primary key is ANIMALTYPE (only this key attribute is shown). DynamoDB uses its hash function to determine where to store a new item, in this case based on the hash value of the string DOG. Note that the items are not stored in sorted order. Each item's location is determined by the hash value of its partition key."[25]

[24] AWS Documentation, DynamoDB Core Components, available at https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/HowItWorks.CoreComponents.html#HowItWorks.CoreComponents.PrimaryKey; *see also supra* n. 10 at *2–9.

[25] *See supra* n. 10 at *18–20.



44. "To read that same item from the PETS table, DynamoDB calculates the hash value of DOG, yielding the partition in which these items are stored."[26] On information and belief, these partitions (e.g., nodes) reside on AWS servers.

45. It has been further described that "DynamoDB automatically spreads the data and traffic for your tables over a sufficient number of servers to handle your throughput and storage requirements, while maintaining consistent and fast performance. All of your data is stored on solid state disks (SSDs) and automatically replicated across multiple Availability Zones in an AWS region, providing built-in high availability and data durability."[27]

46. Moreover, on information and belief, AWS receives read and store (i.e., "get" and "put") operations for the keys, e.g., the ANIMALTYPE example discussed above: "Both get and put operations are invoked using Amazon's infrastructure-specific request processing framework over HTTP. There are two strategies that a client can use to select a node: (1) route its request through a generic load balancer that will select a node based on load information, or (2) use a partition-aware client library that routes requests directly to the appropriate coordinator nodes. . . . A node

[26] *Id.* at *20.
[27] *Id.* at *1.

handling a read or write operation is known as the coordinator. Typically, this is the first among the top N nodes in the preference list. If the requests are received through a load balancer, requests to access a key may be routed to any random node in the ring." [28]

47. Further still, it has been described that "partitioning and placement information also propagates via the gossip-based protocol and each storage node is aware of the token ranges handled by its peers. This allows each node to forward a key's read/write operations to the right set of nodes directly."[29]

48. Based on the above, the AWS Accused Products directly infringe at least, but not limited to, claim 1 of the '170 patent.

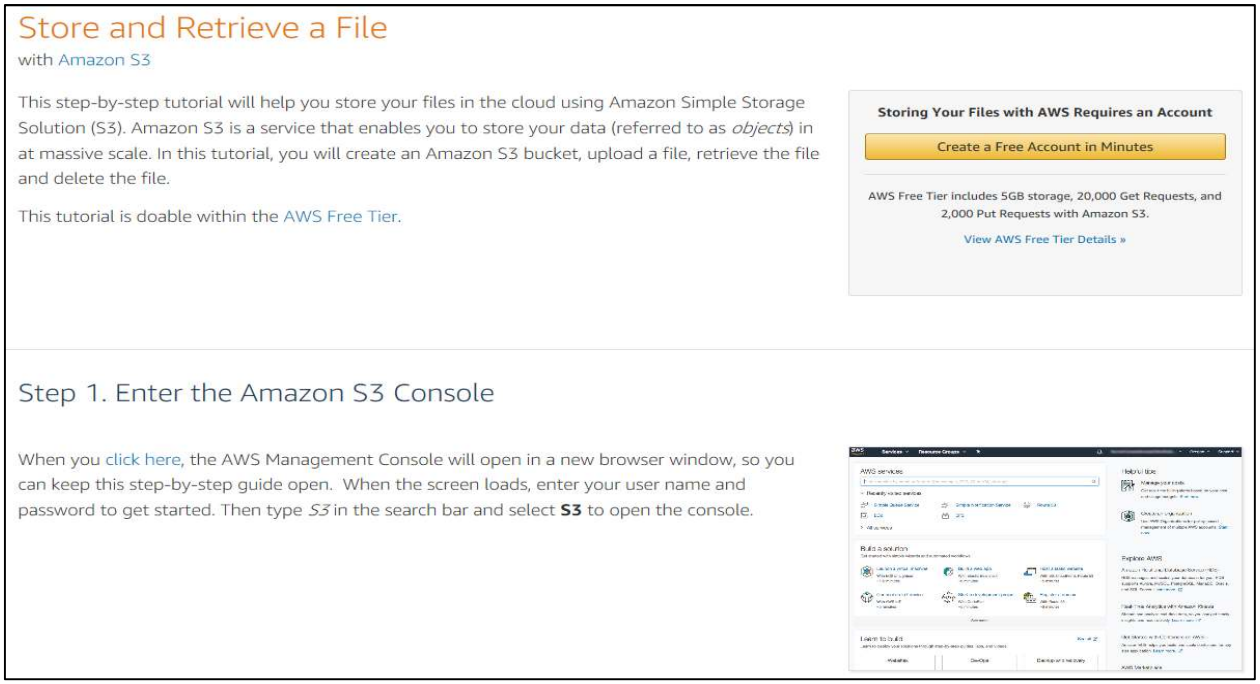
49. AWS also indirectly infringes the '170 patent by inducing others to infringe and/or contributing to the infringement of others, including third party users of the AWS Accused Products in this District and elsewhere in the United States. Specifically, on information and belief, AWS has had knowledge of the '170 patent since at least the time it was served with this Complaint. AWS nevertheless continues to act in wanton disregard of Kove's patent rights.

50. Kove is informed and believes, and thereon alleges, that AWS has actively induced the infringement of the '170 patent under 35 U.S.C. § 271(b) by actively inducing the infringing use of the AWS Accused Products by third party users in the United States. Kove is informed and believes, and thereon alleges, that AWS knew or should have known that its conduct would induce others to use the Accused Products in a manner that infringes the '170 patent. Kove is informed and believes, and thereon alleges, that these third parties infringe the '170 patent in violation of 35 U.S.C. § 271(a) by using the AWS Accused Products.

---

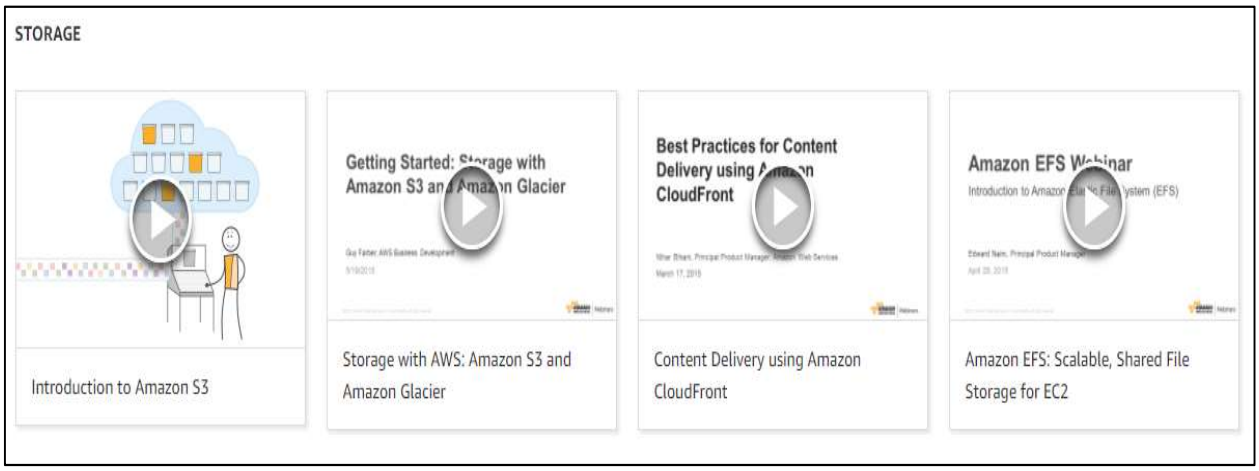
[28] DeCandia, G., et al., Dynamo: Amazon's Highly Available Key-value Store, Amazon.com (Oct. 14–17, 2007) at *211 (Attached as Ex. 21); *see supra* n. 13.
[29] *Id*. at *213.

51. For example, AWS provides several support websites instructing third parties on the use of the AWS Accused Products, including, without limitation: "Getting Started Resource Center"[30]; "Cloud Solutions"[31]; "AWS Documents"[32]; "AWS Support"[33]; and "AWS Customer Success."[34] These exemplary instructional documentations explain how to use the AWS Accused Products to store and retrieve data. In addition, AWS provides a "step-by-step tutorial [that] will help you store your files in the cloud using Amazon Simple Storage Solution (Amazon S3). Amazon S3 is a service that enables you to store your data (referred to as objects) in at massive scale. In this tutorial, you will create an Amazon S3 bucket, upload a file, retrieve the file and delete the file."[35]



[30] Getting Started Resource Center, available at https://aws.amazon.com/getting-started/.
[31] Cloud Solutions Find the right solution by application or industry, available at https://aws.amazon.com/solutions/.
[32] AWS Documentation Find user guides, developer guides, API references, tutorials, and more, available at https://aws.amazon.com/documentation/.
[33] AWS Support, available at https://aws.amazon.com/premiumsupport/.
[34] AWS Customer Success, available at https://aws.amazon.com/solutions/case-studies/.
[35] Store and Retrieve a File with Amazon S3, available at https://aws.amazon.com/getting-started/tutorials/backup-files-to-amazon-s3/.

52. AWS also provides answers on its website to popular topics, posts, and questions that developers, users, or operators may have about the AWS Accused Products.[36] For instance, AWS News Blog provides updates on Amazon S3.[37] Likewise, AWS educates users about features and developments of the AWS Accused Products: "AWS holds events both online and in-person to bring the cloud computing community together to connect, collaborate, and learn from AWS experts."[38] Users can also get help by directly asking AWS support developers.[39] AWS further provides instructional videos to teach users how to use the AWS Accused Products.[40]



53. In addition, AWS provides S3 and DynamoDB manuals to help the customer understand, setup, and use the features of the AWS Accused Products, along with user manuals for all AWS products and services, which may be accessed through links at https://aws.amazon.com/documentation/. AWS also provides Tools for using the AWS Accused

[36] AWS News Blog, available at https://aws.amazon.com/blogs.
[37] Barr, J., Amazon S3 Update: New Storage Class and General Availability of S3 Select, available at https://aws.amazon.com/blogs/aws/amazon-s3-update-new-storage-class-general-availability-of-s3-select/ (Attached as Ex. 22).
[38] AWS Events & Webinars available at https://aws.amazon.com/about-aws/events/.
[39] *See* AWS Support, *supra* n. 33.
[40] Getting Started Videos, https://aws.amazon.com/getting-started/videos/; Amazon S3, http://amzn.to/2iNk9IA; Introduction to Amazon S3, https://youtu.be/rKpKHulqYOQ.

Products, such as SDKs (software development kits), IDE (integrated development environment) Toolkits, and Command Line Tools.[41]

54. As yet another example, in this District, AWS promotes collaborating with "partners and customers" and encourages architects to work "with AWS field sales, pre-sales, training and support teams to help partners and customers learn and use AWS services such as Amazon Elastic Compute Cloud (EC2), Amazon Simple Storage Service (S3), Amazon SimpleDB/RDSdatabases, AWS Identity and Access Management (IAM), etc."[42] Accordingly, AWS actively induces third parties to infringe the '170 patent.

55. Upon information and belief, AWS contributorily infringes the '170 patent under 35 U.S.C. § 271(c) by importing, selling and/or offering to sell within the United States the AWS Accused Products (or components thereof) that constitute a material part of the claimed invention and are not staple articles of commerce suitable for substantial non-infringing use. For example, the AWS Accused Products, including Amazon S3 and DynamoDB, are material, have no substantial non-infringing uses, and are known by AWS to be especially made or especially adapted for use in the manner claimed in the '170 patent. Accordingly, AWS contributorily infringes the '170 patent.

COUNT II

AWS'S INFRINGEMENT OF THE '978 PATENT

56. Kove incorporates by reference the foregoing paragraphs of this Complaint as if fully set forth herein.

[41] Tools for Amazon Web Services, available at https://aws.amazon.com/tools/.

[42] Amazon Jobs, Cloud Infrastructure Architect - Chicago - Job ID: 583645 (Attached as Ex. 23).

57. On information and belief, AWS has infringed and will continue to infringe the '978 patent. AWS directly infringes the '978 patent under 35 U.S.C. § 271(a) by making, using, selling, offering for sale, and/or importing in this District and into the United States products and/or methods covered by one or more claims of the '978 patent, including, but not limited to, the AWS Accused Products. As an example, the AWS Accused Products infringe at least claim 17 of the '978 patent.

58. Claim 17 is directed to a "method of scaling at least one of capacity and transaction rate capability in a location server in a system having a plurality of location servers for storing and retrieving location information." Claim 17, for example, recites a method for "providing a transfer protocol configured to transport identifier and location information." The claimed method also recites "storing location information . . . at a first location server." Claim 17's method further requires "receiving an identifier and a location relevant to the identifier at the first location server," "storing the received location in a location store at the first data location server," and "transferring a portion of the identifiers and associated locations to a second data location server when a performance criterion of the first location server reaches a predetermined performance limit."

59. On information and belief, AWS provides for a method of scaling at least one of capacity and transaction rate capability in a location server in a system having a plurality of location servers for storing and retrieving location information. For example, DynamoDB manages multiple partitions and servers that support a soft throughput performance limit of 3000 read-capacity units and 1000 write-capacity units. By managing those partitions and servers, DynamoDB provides "some flexibility in your per-partition throughput provisioning by providing

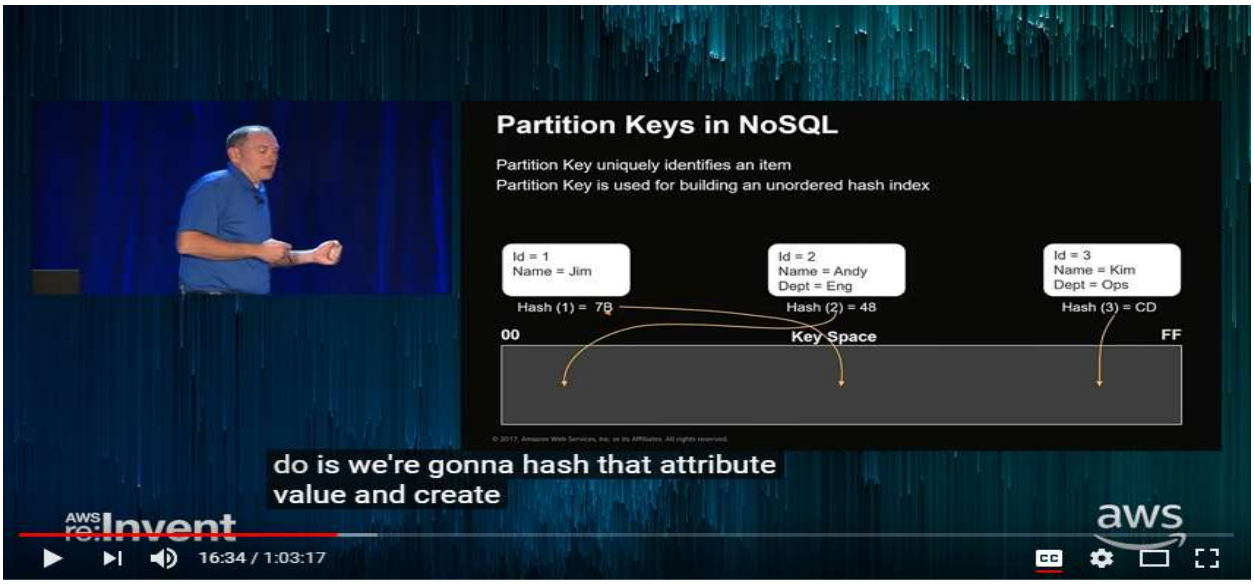
burst capacity, as follows. Whenever you are not fully using a partition's throughput, DynamoDB reserves a portion of that unused capacity for later *bursts* of throughput to handle usage spikes."[43]

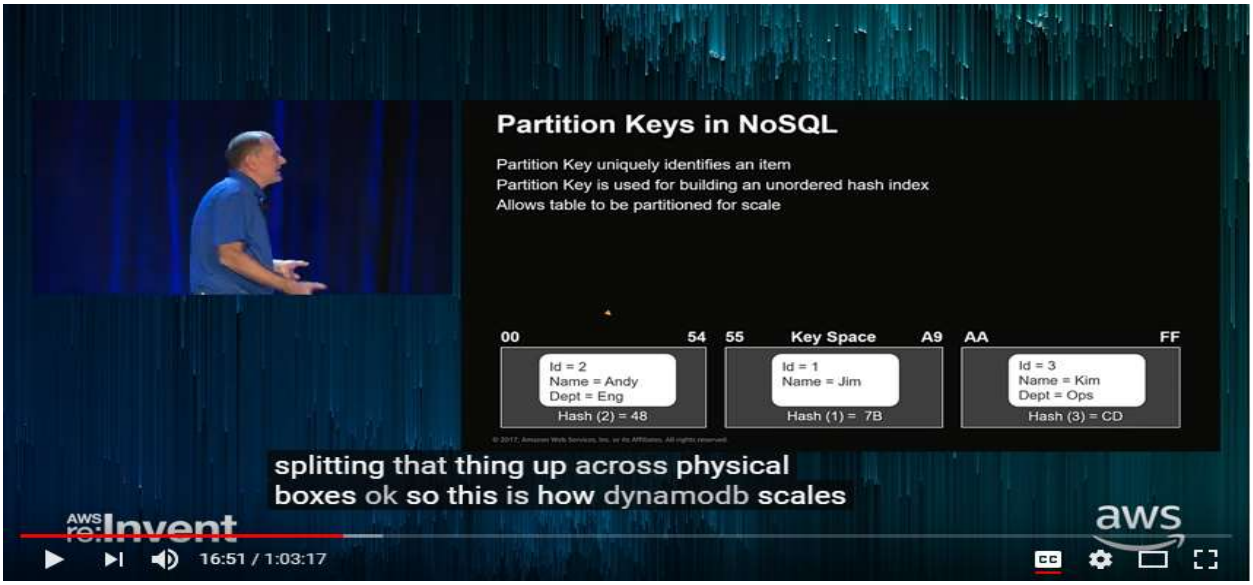
60. As explained in previous paragraphs, AWS stores location information at location servers. For example, DynamoDB stores Amazon S3 object identifiers.[44] "The heart of the S3 object index is a DynamoDB table with one item per object, which associates various attributes with the object's S3 key. Each item contains the S3 key, the size of the object, and any additional attributes to use for lookups."[45]

61. DynamoDB tables organize data location information across servers based on a hash function.[46] For instance, AWS has described DynamoDB as a system that "uses its hash function to determine where to store a new item . . . . Note that the items are not stored in sorted order. Each item's location is determined by the hash value of its partition key."[47]

62. As a further example, it has been explained that DynamoDB hashes location information to create hash tables, which is then split up across a set of servers: "[W]e're gonna hash that attribute value and create this . . . hash index and lay these items out on an arbitrary key space . . . as you add capacity or you increase the storage and add more items into the table we're gonna start splitting that thing up across physical boxes ok so this is how DynamoDB scales . . . ."[48]

[43] *See supra* n. 10 at *517 (DynamoDB Developer Guide, Ex. 11).
[44] *Id.* at *523.
[45] *See supra* n. 21 (Deck, Ex. 19).
[46] *See supra* n. 10 at *18 (DynamoDB Developer Guide, Ex. 11).
[47] *Id.* at *18.
[48] *See supra* n. 12 (DynamoDB Youtube Video).



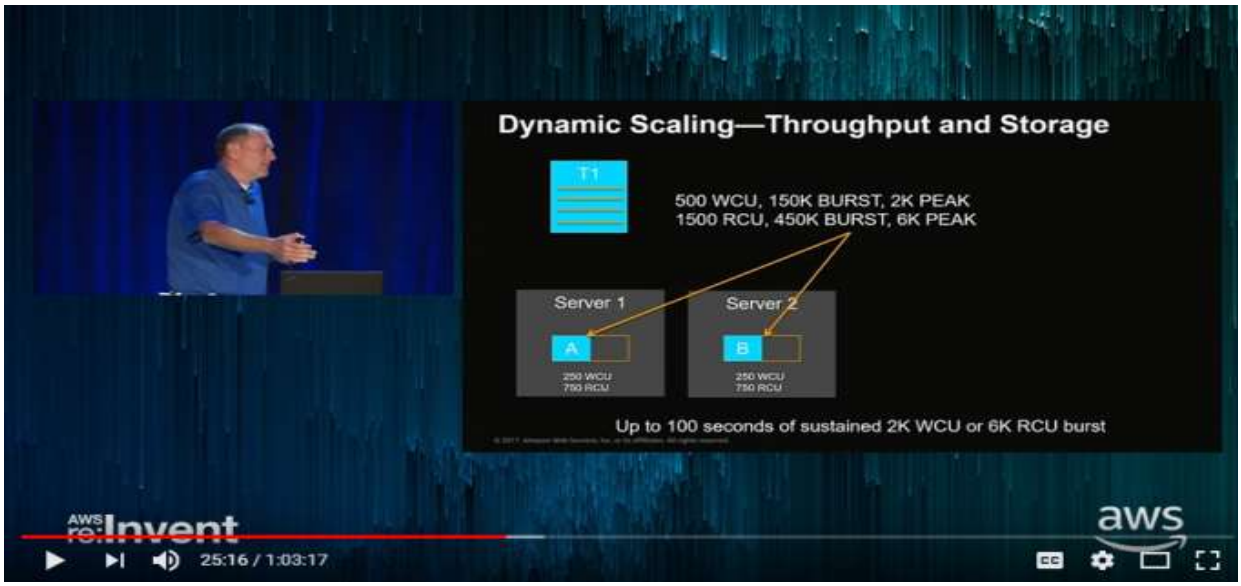


63. In addition, the AWS DynamoDB Developer Guide explains that DynamoDB handles the partition management entirely and the partitions are automatically replicated across multiple Availability Zones within an AWS region.[49] On information and belief, the partition

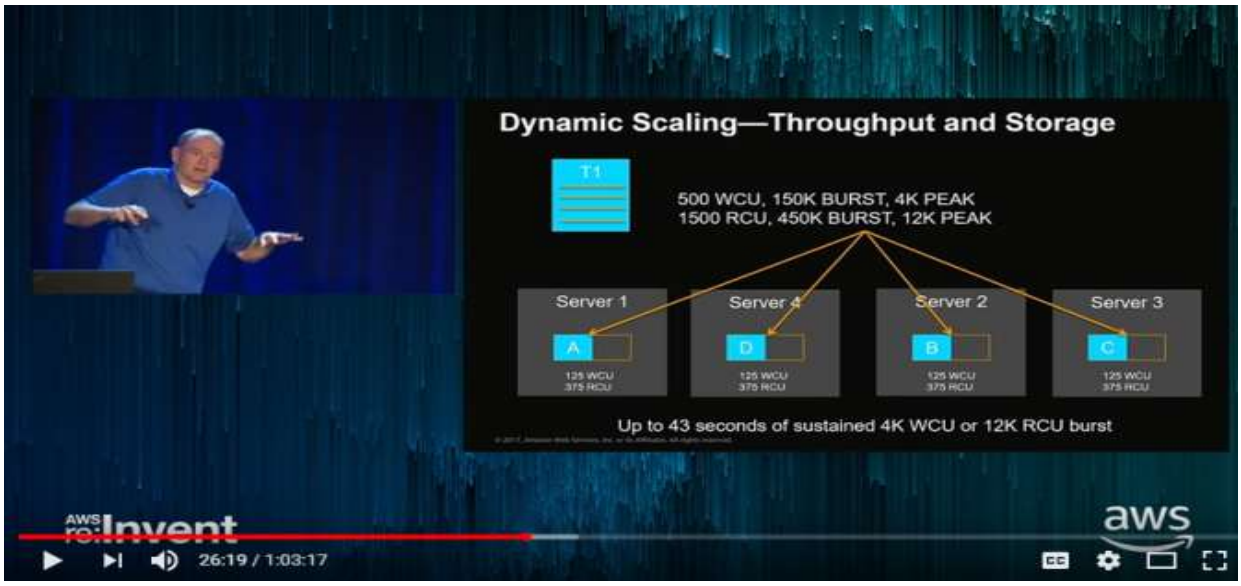
[49] *See supra* n. 10 at *18 (DynamoDB Developer Guide, Ex. 11).

management uses a gossip protocol that allows for "full table routing to other nodes in the system."[50]

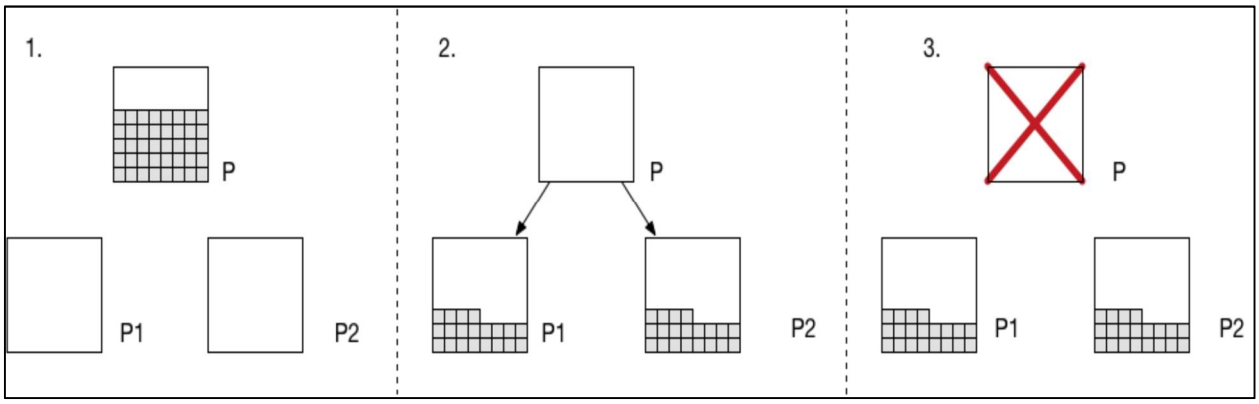
64. Likewise, AWS transfers a portion of the identifiers and associated locations to data location servers when a performance criterion of a location server reaches a predetermined performance limit. DynamoDB's and Amazon S3's best practices for optimizing performance depend on the usage request rates—the throughput performance limit. For example, if the workload request rate grows steadily, Amazon S3 automatically partitions the buckets as needed to support higher request rates.[51] As another example, on information and belief, and as an AWS senior practice manager explained, DynamoDB may use dynamic scaling or burst capacity—throughput and storage—to automatically spread the data location information across more partitions if the throughput performance limit is exceeded[52]:



[50] *See supra* n. 28 at *218 (DeCandia, Ex. 21); *see also supra* n. 13.
[51] *See supra* n. 18 at *535 (S3 Developer Guide, Ex. 17).
[52] *See supra* n. 12 (DynamoDB Youtube Video).



65. Further still, the AWS DynamoDB Developer Guide shows that the partition data is distributed across multiple servers:



66. As depicted, "[t]he large squares represent partitions, and the small squares represent data items in the table. Note that DynamoDB performs partition splits automatically, in the background."[53]

67. Based on the above, the AWS Accused Products directly infringe at least, but not limited to, claim 17 of the '978 patent.

[53] *See supra* n. 10 at *514 (DynamoDB Developer Guide, Ex. 11).

68. AWS also indirectly infringes the '978 patent by inducing others to infringe and/or contributing to the infringement of others, including third party users of the AWS Accused Products in this District and elsewhere in the United States. Specifically, on information and belief, AWS has had knowledge of the '978 patent since at least the time it was served with this Complaint. AWS nevertheless continues to act in wanton disregard of Kove's patent rights.

69. Kove is informed and believes, and thereon alleges, that AWS has actively induced the infringement of the '978 patent under 35 U.S.C. § 271(b) by actively inducing the infringing use of the AWS Accused Products by third party users in the United States. Kove is informed and believes, and thereon alleges, that AWS knew or should have known that its conduct would induce others to use the Accused Products in a manner that infringes the '978 patent. Kove is informed and believes, and thereon alleges, that these third parties infringe the '978 patent in violation of 35 U.S.C. § 271(a) by using the AWS Accused Products. In particular, Kove incorporates by reference paragraphs 51 to 54 as if fully set forth herein.

70. Upon information and belief, AWS contributorily infringes the '978 patent under 35 U.S.C. § 271(c) by importing, selling and/or offering to sell within the United States the AWS Accused Products (or components thereof) that constitute a material part of the claimed invention and are not staple articles of commerce suitable for substantial non-infringing use. For example, the AWS Accused Products, including Amazon S3 and DynamoDB, are material, have no substantial non-infringing uses, and are known by AWS to be especially made or especially adapted for use in manner claimed in the '978 patent. Accordingly, AWS contributorily infringes the '978 patent.

COUNT III

AWS'S INFRINGEMENT OF THE '640 PATENT

71. Kove incorporates by reference the foregoing paragraphs of this Complaint as if fully set forth herein.

72. On information and belief, AWS has infringed and will continue to infringe the '640 patent. AWS directly infringes the '640 patent under 35 U.S.C. § 271(a) by making, using, selling, offering for sale, and/or importing in and into the United States products and/or methods covered by one or more claims of the '640 patent, including, but not limited to, the AWS Accused Products. As an example, the AWS Accused Products infringe at least claim 18 of the '640 patent.

73. Claim 18 is directed to a "system for retrieving data location information for data stored in a distributed network." Claim 18 recites, for example, a "data repository configured to store data, wherein the data is associated with an identifier string." The system also includes a "client responsive to a data query to query a data location server for location information associated with the identifier string." The "data location server network" according to claim 18 contains "location information associated with the identifier string." If the "data location server" does not contain the "location information," "the location server transmits a redirect message to the client, wherein the redirect message contains redirect information for use by the client to calculate a location of a different data location server" containing the location information.

74. AWS provides for a system for retrieving data location information for data stored in a distributed network. For example, as shown above in the Architecture Diagram, "[u]sers upload files into Amazon Simple Storage Service (Amazon S3), a highly durable storage infrastructure designed for mission-critical and primary data storage. Amazon S3 makes it easy to store and retrieve any amount of data, at any time. . . . File metadata, version information, and unique identifiers are stored by the application servers on an Amazon DynamoDB table. As the

number of files to maintain in the application grows, Amazon DynamoDB tables can store and retrieve any amount of data, and serve any amount of traffic."[54]

75. AWS also provides for a data repository configured to store data, wherein the data is associated with an identifier string. For example, as discussed, Amazon S3 stores data in a files repository. Each data entity stored by Amazon S3, called an Amazon S3 Object, is "uniquely identified within a bucket by a key (name) and a version ID. . . . Amazon S3 can be thought of as a basic data map between 'bucket + key + version' and the object itself. Every object in Amazon S3 can be uniquely addressed through the combination of the web service endpoint, bucket name, key, and optionally, a version."[55]

76. Further, as discussed, Amazon S3 stores location information for S3 objects in a DynamoDB table. On information and belief, DynamoDB tables are distributed across multiple servers. AWS has explained, for example, that "DynamoDB stores data in partitions. A partition is an allocation of storage for a table, backed by solid-state drives (SSDs) and automatically replicated across multiple Availability Zones within an AWS Region. Partition management is handled entirely by DynamoDB—you never have to manage partitions yourself."[56]

77. Likewise, location information for Amazon S3 objects is stored in DynamoDB, along with the objects' metadata. The Amazon S3 "index subsystem manages the metadata and location information of all S3 objects in the region,"[57] and it has been explained that "[t]he heart of the S3 object index is a DynamoDB table with one item per object, which associates various attributes with the object's S3 key. Each item contains the S3 key, the size of the object, and any

[54] *See supra* n. 17 (Architecture Diagram, Ex. 16).
[55] *See supra* n. 18 at *3–4 (S3 Developer Guide, Ex. 17).
[56] *See supra* n. 10 at *18 (DynamoDB Developer Guide, Ex. 11).
[57] *See supra* n. 19 (Summary of the Amazon S3 Service Disruption in the Northern Virginia, Ex. 18).

additional attributes to use for lookups."[58] AWS's "Frequently Asked Questions About Amazon DynamoDB" explains that "file pointers (possibly to Amazon S3 objects) are best saved in Amazon DynamoDB," as opposed to in Amazon S3.[59]

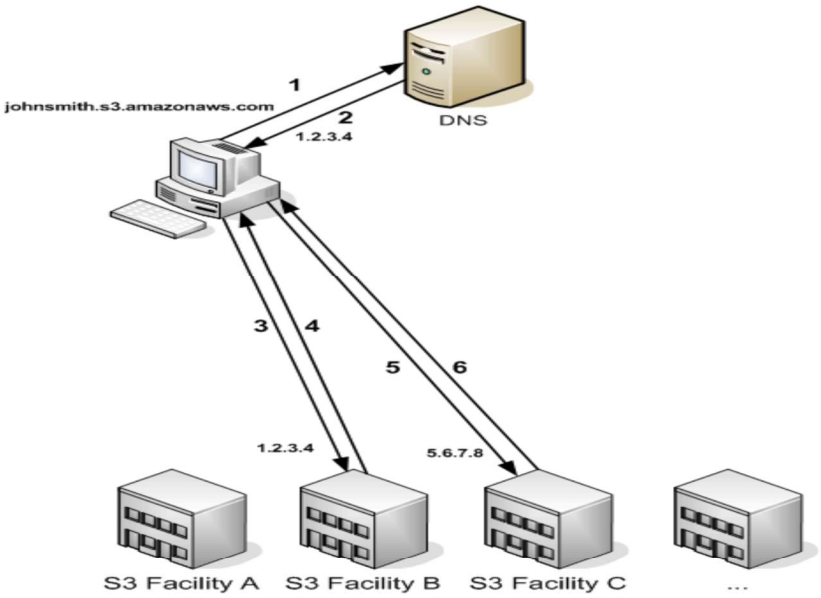
78. On information and belief, DynamoDB tables organize data location information across servers based on a hash function.[60] For instance, AWS has described DynamoDB as a system that "uses its hash function to determine where to store a new item . . . . Note that the items are not stored in sorted order. Each item's location is determined by the hash value of its partition key."[61]

79. Further, Amazon S3 includes the AWS Management Console,[62] which responds to requests to download Amazon S3 objects by querying the locations of the requested objects.[63]

80. Further still, AWS provides for a redirect message that contains redirect information for use to calculate a location of a different data location server. For example, Amazon S3 supports request redirection, as illustrated below, such that if "a request arrives at the wrong Amazon S3 location, Amazon S3 responds with a temporary redirect that tells the requester to send the request to a new endpoint."[64]

---

[58] *See supra* n. 21 (Deck, Ex. 19).
[59] *See supra* n. 22 at *4–5 (Amazon DynamoDB FAQs, Ex. 20).
[60] *See supra* n. 10 at *18 (DynamoDB Developer Guide, Ex. 11).
[61] *Id*.
[62] AWS Management Console, Getting Started with the AWS Management Console, at *1, *9, available at https://docs.aws.amazon.com/awsconsolehelpdocs/latest/gsg/console-help-gsg.pdf (Attached as Ex. 24).
[63] Amazon Simple Storage Service Console User Guide, Uploading, Downloading, and Managing Objects, at *31–41, available at https://docs.aws.amazon.com/AmazonS3/latest/user-guide/s3-user-guide.pdf (While the full document is incorporated by reference herein, an excerpted version of the document is attached as Ex. 25); *see also* Amazon's Web Version of the S3 Console User Guide at https://docs.aws.amazon.com/AmazonS3/latest/user-guide/what-is-s3.html.
[64] *See supra* n. 18 at *530–531 (S3 Developer Guide, Ex. 17).



1	The client makes a DNS request to get an object stored on Amazon S3.
2	The client receives one or more IP addresses for facilities that can process the request.
3	The client makes a request to Amazon S3 Facility B.
4	Facility B returns a redirect indicating the object is available from Location C.
5	The client resends the request to Facility C.
6	Facility C returns a copy of the object.

81. Moreover, on information and belief, AWS receives get and put operations as discussed above: "Both get and put operations are invoked using Amazon's infrastructure-specific request processing framework over HTTP. There are two strategies that a client can use to select a node: (1) route its request through a generic load balancer that will select a node based on load information, or (2) use a partition-aware client library that routes requests directly to the appropriate coordinator nodes. . . . A node handling a read or write operation is known as the coordinator. Typically, this is the first among the top N nodes in the preference list. If the requests are received through a load balancer, requests to access a key may be routed to any random node in the ring. In this scenario, the node that receives the request will not coordinate it if the node is not in the top N of the requested key's preference list. Instead, that node will forward the request to the first among the top N nodes in the preference list. . . . Read and write operations involve

the first N healthy nodes in the preference list, skipping over those that are down or inaccessible. When all nodes are healthy, the top N nodes in a key's preference list are accessed. When there are node failures or network partitions, nodes that are lower ranked in the preference list are accessed."[65]

82. Based on the above, the AWS Accused Products directly infringe at least, but not limited to, claim 18 of the '640 patent.

83. AWS also indirectly infringes the '640 patent by inducing others to infringe and/or contributing to the infringement of others, including third party users of the AWS Accused Products in this District and elsewhere in the United States. Specifically, on information and belief, AWS has had knowledge of the '640 patent since at least the time it was served with this Complaint. AWS nevertheless continues to induce others to infringe in wanton disregard of Kove's patent rights.

84. Kove is informed and believes, and thereon alleges, that AWS has actively induced the infringement of the '640 patent under 35 U.S.C. § 271(b) by actively inducing the infringing use of the AWS Accused Products by third party users in the United States. Kove is informed and believes, and thereon alleges, that AWS knew or should have known that its conduct would induce others to use the Accused Products in a manner that infringes the '640 patent. Kove is informed and believes, and thereon alleges, that these third parties infringe the '640 patent in violation of 35 U.S.C. § 271(a) by using the AWS Accused Products. In particular, Kove incorporates by reference paragraphs 51 to 54 as if fully set forth herein.

85. Upon information and belief, AWS contributorily infringes the '640 patent under 35 U.S.C. § 271(c) by importing, selling and/or offering to sell within the United States the AWS

[65] *See supra* n. 28 at *211 (DeCandia, Ex. 21); *see also supra* n. 13.

Accused Products (or components thereof) that constitute a material part of the claimed invention and are not staple articles of commerce suitable for substantial non-infringing use. For example, the AWS Accused Products, including Amazon S3 and DynamoDB, are material, have no substantial non-infringing uses, and are known by AWS to be especially made or especially adapted for use in manner claimed in the '640 patent. Accordingly, AWS contributorily infringes the '640 patent.

DAMAGES

86. Kove incorporates by reference the foregoing paragraphs of this Complaint as if fully set forth herein.

87. As a result of AWS's acts of infringement, Kove has suffered actual and consequential damages; however, Kove does not yet know the full extent of the infringement and its extent cannot be ascertained except through discovery and special accounting. To the fullest extent permitted by law, Kove seeks recovery of damages at least for reasonable royalties, unjust enrichment, lost profits, and/or benefits received by AWS as a result of infringing the patents-in-suit. Kove further seeks any other damages to which Kove is entitled under law or in equity.

IRREPARABLE HARM TO KOVE

88. Kove incorporates by reference the foregoing paragraphs of this Complaint as if fully set forth herein.

89. Kove has been irreparably harmed by AWS's acts of infringement and will continue to be irreparably harmed unless and until AWS's acts of infringement are enjoined by this Court. Kove has no adequate remedy at law to redress AWS's continuing acts of infringement, and money damages will not suffice to remedy the harms to Kove. The hardships that would be imposed upon

AWS by an injunction are less than those faced by Kove should an injunction not issue. Furthermore, the public interest would be served by issuance of an injunction.

ATTORNEYS' FEES

90. Kove is entitled to recover reasonable and necessary attorneys' fees under applicable law.

DEMAND FOR JURY TRIAL

91. Kove hereby demands a jury trial on its claims for patent infringement.

PRAYER FOR RELIEF

WHEREFORE, Kove respectfully requests that this Court enter judgment in its favor and grant the following relief:

A. A judgment that the AWS Accused Products directly and/or indirectly infringe the '170, '978, and '640 patents;

B. That such infringement is willful;

C. An order permanently enjoining AWS and its respective officers, directors, agents, partners, servants, employees, attorneys, licensees, successors, and assigns, and those in active concert or participation with any of them, from engaging in infringing activities with respect to the '170, '978, and '640 patents;

D. A ruling finding that this case is exceptional and awarding Kove its reasonable attorneys' fees under 35 U.S.C. § 285;

E. A judgment and order requiring AWS to pay Kove's damages in an amount adequate to compensate Kove for AWS's infringement, but in no event less than a reasonable royalty under 35 U.S.C. § 284, including supplemental damages for any continuing post-verdict infringement up until entry of judgment and beyond, with accounting, as needed;

F. An award of enhanced damages pursuant to 35 U.S.C. § 284;

G. In the alternative, in the event injunctive relief is not granted as requested by Kove, an award of a mandatory future royalty payable on each future product sold by AWS that is found to infringe one or more claims of the '170, '978, and '640 patents, and on all future products which are not colorably different from products found to infringe;

H. A judgment and order requiring AWS to pay Kove's costs of this action (including all disbursements);

I. An order for accounting of damages;

J. A judgment and order requiring AWS to pay pre-judgment and post-judgment interest to the full extent allowed under the law; and,

K. Such other and further relief as the Court may deem just and proper under the circumstances.

Dated: December 12, 2018

Respectfully submitted,

/s/ Renato Mariotti
Renato Mariotti

Renato Mariotti (State Bar No. 6323198)
rmariotti@thompsoncoburn.com
Thompson Coburn LLP
55 E. Monroe St., 37th Floor
Chicago, IL 60603
Telephone: (312) 346-7500
Telecopier: (312) 580-2201

Sarah O. Jorgensen
(*pro hac vice* application pending)
sjorgensen@reichmanjorgensen.com
Reichman Jorgensen LLP
1201 West Peachtree, Suite 2300
Atlanta, GA 30309
Telephone: (650) 623-1403
Telecopier: (650) 623-1449

Courtland L. Reichman
(*pro hac vice* application pending)
creichman@reichmanjorgensen.com
Phillip Lee
(*pro hac vice* application pending)
plee@reichmanjorgensen.com
Jennifer P. Estremera
(*pro hac vice* application pending)
jestremera@reichmanjorgensen.com
Joachim B. Steinberg
(*pro hac vice* application pending)
jsteinberg@reichmanjorgensen.com
Michael G. Flanigan (State Bar No. 6309008)
mflanigan@reichmanjorgensen.com
Kate M. Falkenstien
(*pro hac vice* application pending)
kfalkenstien@reichmanjorgensen.com
Reichman Jorgensen LLP
303 Twin Dolphin Drive, Suite 600
Redwood Shores, CA 94065
Telephone: (650) 623-1401
Telecopier: (650) 623-1449

ATTORNEYS FOR PLAINTIFF KOVE IO, INC.