Dr. Shamsudeen E. International Journal of Engineering Research and Applications
www.ijera.com
ISSN: 2248-9622, Vol. 11, Issue 6, (Series-I) June 2021, pp. 48-51

RESEARCH ARTICLE OPEN ACCESS

Distributed Applications: Virtual Machine, BitTorrent and Blcokchain

Dr. Shamsudeen E
Assistant Professor of Computer Applications
EMEA College of Arts and Science, Kondotty, Kerala, India

ABSTRACT
Applications of Distributed computers have been widely used nowadays.Distributed system, also known as distributed computing, is a system with multiple components located on different machines that communicate and coordinate actions in order to appear as a single coherent system to the end-user. Blockchain is a storage technology for public decentralized databases. The emergence of blockchain technology makes it possible to solve the trust problem of distributed system nodes within the wide area network. In this paper it is described how much related blockchain transactions with distributed computing. The paper also deals with the BitTorrent and virtual machine and how they are applied the distributed concept.
Keywords:distributed computing, blockchain, virtual machine, BitTorrent.

Date of Submission: 25-05-2021 Date of Acceptance: 08-06-2021

I. INTRODUCTION

With the ever-scaling technological expansion of the world, distributed systems are becoming more and more used and has significance. They are a sophisticated field of study in computer science and applications.

A distributed system[1], also known as distributed computing, is a system with multiple components located on different machines that communicate and coordinate actions in order to appear as a single coherent system to the end-user. It does not reveal the existence of multiples systems. i.e., the user feels that all services are get done by the system which is located on his table.

The machines that are a part of a distributed system may be computers, physical servers, virtual machines, containers, or any other node that can connect to the network, have local memory, and communicate by passing messages.

1. Working principle of a distributed system

There are two general ways that distributed systems function:

1. Each machine works toward a common goal and the end-user views results as one cohesive unit.
2. Each machine has its own end-user and the distributed system facilitates sharing resources or communication services.

3. Why distributed systems?

There are three reasons to implement distributed systems

3.1. Horizontal Scalability

Since computing happens independently on each node, it is easy and generally inexpensive to add additional nodes and functionality as necessary.

3.2. Reliability

Most distributed systems are fault-tolerant as they can be made up of hundreds of nodes that work together. The system generally doesn't experience any disruptions if a single machine fails.

3.3. Performance

Distributed systems are extremely efficient because work loads can be broken up and sent to multiple machines.

4. Challenges of distributed systems

Distributed systems are not without challenges. Complex architectural design, construction, and debugging processes that are required to create an effective distributed system.

Three challenges are,

4.1. Scheduling

A distributed system has to decide which jobs need to run, when they should run, and where they should run. Schedulers ultimately have limitations, leading to underutilized hardware and unpredictable runtimes.

4.2. Latency

The more widely your system is distributed, the more latency you can experience with communications. This often leads to teams making tradeoffs between availability, consistency, and latency.

4.3. Observability
Gathering, processing, presenting, and monitoring hardware usage metrics for large clusters is a significant challenge.

5. How a Distributed System Works and its types

In a distributed system everything must be interconnected through the communication system.
Distributed systems are generally classified into four basic categories based on their architecture.
5.1. Client-server
Clients sends the request to the server to get some service from the server site. Client acts as a requester and sever acts as a master as it fulfills the requests of the client.
5.2. Three-tier
Information about the client is stored in a middle tier rather than on the client site to simplify application deployment. This architecture model is most common for web applications.
5.3. n-tier
Generally used when an application orforward requests to additional enterprise services on the network.
5.4. Peer-to-peer
Here all the systems gets the equal priority. Responsibilities are uniformly distributed among machines in the system, known as peers[2], which can serve as either client or server.

6. Distributed Applications

If you roll up more than one Rails servers behind a single load balancer and all are connected to one database, could you call that a distributed application? Yes because, as per definition of distributed systems, a group of computers working together to appear as a single coherent system.

A system is distributed only if the nodes communicate with each other to coordinate their action. In this example all systems are connected through a communication network.

6.1. Erlang Virtual Machine

Erlang is a high level functional language that has great semantics for concurrency, distribution and fault-tolerance(error detection). The Erlang Virtual Machine[3] itself handles the distribution of an Erlang application.A built-in distribution mechanism enables designers to create a system whose processes may run on different computers. Erlang handles fault detection and recovery in distributed systems and has fault recovery schemes, which can be customized. The Development Environment includes several facilities for creating interfaces between applications written in Erlang and other programming languages.

Its model works by having many isolated lightweight processes all with the ability to talk to each other via a built-in system of message passing. This is called the Actor Model[4] and the Erlang OTP libraries can be thought of as a distributed actor framework

The model is what helps it achieve great concurrency rather simply — the processes are spread across the available cores of the system running them. Since this is indistinguishable from a network setting (apart from the ability to drop messages), Erlang's VM can connect to other Erlang VMs running in the same data center or even in another continent. This swarm of virtual machines run one single application and handle machine failures via takeover (another node gets scheduled to run).

In fact, the distributed layer of the language was added in order to provide fault tolerance. Software running on a single machine is always at risk of having that single machine dying and taking your application offline. Software running on many nodes allows easier hardware failure handling, provided the application was built with that in mind.

6.2. BitTorrent

BitTorrent[5](peer-to-peer protocol)is one of the most widely used protocol for transferring large files across the web via torrents. The main idea is to facilitate file transfer between different peers in the network without having to go through a main server. That is, Each computer that downloads the data downloads it from the web page's central server.

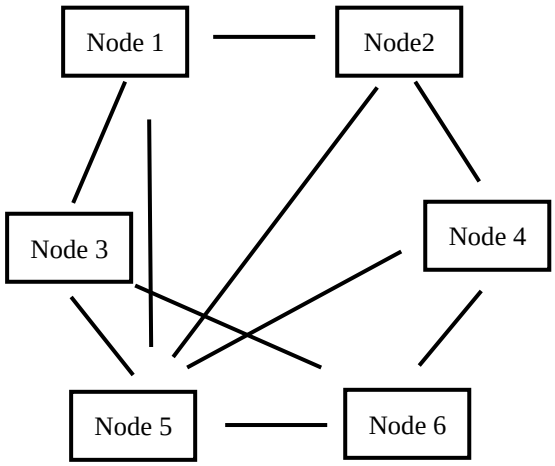

Figure 1

Traditionally, a computer joins a BitTorrent swarm by loading a .torrent file into a BitTorrent client. The BitTorrent client contacts a "tracker"

specified in the .torrent file. The tracker is a special server that keeps track of the connected computers. The tracker shares their IP addresses with other BitTorrent clients in the swarm, allowing them to connect to each other.

Once connected, a BitTorrent client downloads bits of the files in the torrent in small pieces, downloading all the data it can get. Once the BitTorrent client has some data, it can then begin to upload that data to other BitTorrent clients in the swarm. In this way, everyone downloading a torrent is also uploading the same torrent(shown in Fig. 1. This speeds up everyone's download speed. If 10,000 people are downloading the same file, it doesn't put a lot of stress on a central server. Instead, each downloader contributes upload bandwidth to other downloaders, ensuring the torrent stays fast.

Importantly, BitTorrent clients never actually download files from the tracker itself. The tracker participates in the torrent only by keeping track of the BitTorrent clients connected to the swarm, not actually by downloading or uploading data.

6.2.1. Leechers and Seeders

Users downloading from a BitTorrent swarm are commonly referred to as "leechers" or "peers". Users that remain connected to a BitTorrent swarm even after they've downloaded the complete file, contributing more of their upload bandwidth so other people can continue to download the file, are referred to as "seeders". For a torrent to be downloadable, one seeder – who has a complete copy of all the files in the torrent – must initially join the swarm so other users can download the data. If a torrent has no seeders, it won't be possible to download – no connected user has the complete file.

BitTorrent clients reward other clients who upload, preferring to send data to clients who contribute more upload bandwidth rather than sending data to clients who upload at a very slow speed. This speeds up download times for the swarm as a whole and rewards users who contribute more upload bandwidth.

6.3. Distributed Ledgers and Blockchain

A distributed ledger[6] is a database that can be shared and synchronized across multiple computers accessible by multiple people concurrently. It allows transactions to have public "witnesses". The participant at each node of the network can access the recordings shared across that network and can keep an identical copy of it. Any changes or additions made to the ledger are reflected and copied to all participants within seconds.

A distributed ledger as it stands across the computers in contrast to a centralized one which is more prone to cyber attacks and fraud, as it has a single point of failure. But, Cyber attacks and financial fraud are reduced by the use of distributed ledgers.

6.3.1. Blockchain

A blockchain[7] is essentially a shared database filled with entries that must be confirmed and encrypted. An easy way to understand is to think of it as a highly secure and verified Office 365 document. Each document entry dependent on a logical relationship to all its predecessors. The name blockchain refers to the "blocks" that get added to the chain of transaction records. To facilitate this, the technology uses cryptographic signatures called a hash.

The most important difference to remember is that blockchain is just one type of distributed ledger. Although blockchain is a sequence of blocks, distributed ledgers do not require such a chain. Furthermore, distributed ledgers do not need proof of work and offer – theoretically – better scaling options.

On the surface, distributed ledger sounds exactly how you probably envision a blockchain. However, all blockchains are distributed ledgers, but remember that not all distributed ledgers are blockchains. Whereas a blockchain represents a type of distributed ledger, it is also merely a subset of them.

For security, the blockchain uses encryption[8]. The owner of a wallet that contributes transactions to the chain has a public and private key. The private key is his own passcode to open the wallet, and a public key is used to distribute to payers who send bitcoins. A private key should never be shared, but the public key is what the name suggests – it's shared with other users as a way to send money only to the intended recipient.

The blockchain can be compared to the Internet. The Internet is not one server where everyone browses content. Instead, it's a massive group of network equipment that work together to connect users to web content. No one person can change the Internet, its infrastructure, or the content it contains. That makes it secure from having anyone entity control its applications

II. CONCLUSION

Distributed systems are the dominant platform for any application, both in academic and industry.This article briefly described the current scenario of blockchain technology.This paper attempts to describe the ideas of blockchain as an application of distributed system, and gives a picture of blockchain for building a wide area

network distributed system.Blockchain basically at its core is a distributed system. More precisely it is a decentralized distributed system.

Distributed systems are a computing paradigm whereby two or more nodes work with each other in a coordinated fashion in order to achieve a common outcome and it's modeled in such a way that end users see it as a single logical platform.

This paper also describes that BitTorrent is a distributed peer-to-peer system which, it is stated, has the potential to change the file distribution. As a distributed system the BitTorrent base its operation around the concept of a torrent file, a centralized tracker and an associated swarm of peers

This paper also illustrates that Distributed system allows creation of a large-scale controlled environment with few physical resources that requires application of network management of virtual machines.

REFERENCES

[1]. Popek, G. and Walker, B. (eds.) (1985). The LOCUS Distributed System Architecture. Cambridge, MA: MIT Press.
[2]. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system. Technical report, Manubot (2019)
[3]. Neiger, G., Santoni, A., Leung, F., Rodgers, D., and Uhlig, R. (2006). Intel Virtualization
[4]. Technology: Hardware support for efficient processor virtualization. j-INTEL-TECH-J, 10(3):167–177.
[5]. Barham, P. et al. (2003). Xen and the art of virtualization. In Proceedings of the ACM Symposium on Operating Systems Principles.
[6]. J.A. Pouwelse, P. Garbacki, D.H.J. Epema and H.J. Sips. A Measurement Study of the BitTorrent Peer-to-Peer File-Sharing System. Delft University of Technology.
[7]. Wattenhofer, R.: Distributed Ledger Technology: The Science of the Blockchain. CreateSpace Independent Publishing Platform (2017)
[8]. Baliga, A.: Understanding blockchain consensus models. Persistent **4**, 1–14 (2017)
[9]. Xue, S., Zhao, X., Li, X., Zhang, G., Xing, C.: A trusted system framework for electronic records management based on blockchain. In: Web Information Systems and Applications - 16th International Conference, WISA 2019, Qingdao, China, 20–22 September 2019 (2019), Proceedings. pp. 548–559 (2019)