Automated Provisioning of High Performance Clusters for Enhanced Computational Power

Ayush Tejwani, Anurag Deotale, Vedant Deshmukh
Vishwakarma Institute of Technology, Pune, Maharashtra
Corresponding author: Ayush Tejwani, Email: tejwani.ayush@gmail.com

Most of the industry needs fast processing power in the form of supercomputers for their research and development work, but the high capital, as well as maintenance cost of supercomputers, make them somewhat unfeasible. This is where clusters are preferable. Clusters are groups of independent computers that are linked together to perform a common task and are considered an alternative to supercomputers. These individual computers distribute tasks and work together inorder perform better. We have used similar clusters to conduct brute force attacks on Linux user passwords to evaluate the improvement in compute capability when compared to a single machine. Our system also automates the provisioning i.e. setup of the cluster to make it easy to use.

Keywords: Password cracking, clusters, bruteforce, Ansible, Vagrant, MPI.

2023. In Saroj Hiranwal & Garima Mathur (eds.), *Artificial Intelligence and Communication Technologies*, 705–711. Computing & Intelligent Systems, SCRS, India. https://doi.org/10.52458/978-81-955020-5-9-67

1. Introduction

Passwords remain the most popular means of authentication, although other forms of authentication like biometrics or OTPs (One-time passwords) are steadily gaining popularity due to their ease of use. Many users choose a password that can be easily guessed or broken in. Along with this, many users reuse the same passwords in other places which makes it even more insecure. Various password cracking techniques exist. One of them is the brute-force attack in which the attacker tries every possible combination to break the password. To thwart such attacks, it is recommended to select a long password containing random characters and symbols. The selected characters should not be sourced from any dictionaries. Many tools are available that carry out brute-force attacks at the rate of thousands of hashes per second, but sometimes in doing so, depending on the password, they can take days or sometimes even months. Therefore it becomes infeasible to run the attack for a long time[8]. Hence for this, multi-node clusters[10][3][4][7] come into the picture. These clusters consist of various computing nodes which are all connected and carry out the attack whilst utilizing the entire computing power of all the nodes, thereby significantly reducing the time taken to carry out the attack. Along with the added computing power, as a result of the addition of new nodes in the setup, the ability to conveniently scale the cluster[11] depending on the user's requirement is also important. The thought of manually setting up each node and also provisioning it should not act as a deterrent to scalability. To overcome that, we propose a setup where the process of creating and provisioning the nodes will be automated. Thus by doing so, the user now has the freedom to conveniently upscale or downscale the setup depending on the requirement whilst also removing the redundancy involved in setting up and configuring the setup, for every new node attached to the setup.

2. Literature Review

Cracking passwords by launching attacks through the use of tools is widespread, not only among law- enforcement/government agencies but also among cybercriminals. Although there are many tools which are available that can be used to launch attacks on passwords, John the Ripper[12][12] remains one of the most popular among them. It also supports password cracking using MPI[13] along with different types of attacks on passwords apart from brute-force attacks. It can also crack hashes in various other formats like MD5 etc.

The idea of using clusters[11] to launch such aforementioned attacks has been around for some time. The obvious advantages are that of performance[6][9][14] and the time taken. The feasibility of such clusters has also been evaluated[12] where a Linux based High-Performance Cluster (HPC)[1][2] where a password cracking tool using the Message Passing Interface was built. An analysis and comparison of clustered password crackers was made where two tools, John the Ripper and Cisilia, two tools that can run in a clustered environment were tested[12]. The cluster with Cisilia could not give consistent results. Hence John the Ripper emerged as the alternative.

Thus it was established that an HPC cluster[2] could be used to crack passwords in significantly less time but most of these clusters required a lot of effort while setting them up. Though the performance of the cluster[6][9][14] would increase with the increase in the number of nodes in the setup, there didn't exist a setup that could do so, and that did not require manually configuring each node of the setup. The need to scale up the cluster, as and when required by the user along with the ease of having to bypass the need for individual configuration is what we try to accomplish in our setup.

3. Architecture

The focus of the system is to create a High Performing Cluster(HPC)[10] which does not require manual configuration. To create Virtual Machines we need a tool for controlling the hypervisor, a tool that controls the tool controlling the hypervisor and a tool that can provision our created VMs.

In our system, as shown in Fig 1, we have used Virtual Box, the tool controlling the hypervisor for bringing up the Virtual Machines. Vagrant, the tool which controls Virtual Box using a Perl script that automates the creation of VMs in real-time. Vagrant provides us with the ability to spawn VMs dynamically in one single go, in contrast to manually spawning VMs using Virtual Box.

As shown in Fig 1, the VMs created by Vagrant through Virtual Box are vanilla images of the OS, to use them for our specific computing purpose we would need them to have a distributed computing tool as well as tools required for our specific computational purpose which is, password cracking in our case. In our setup, we have used 3 VMs, 1 master and 2 compute nodes, however, a data centre requires much more computational capabilities hence will have several VMs spawned, to have all these VMs in the same state we will need to automate their provisioning, which is done using Ansible. Ansible is run on a single machine i.e. Master node which has SSH access to all compute nodes and when an ansible playbook is run provisioning of all the compute nodes begins

We use Ansible to install a tool in every node so that they can talk to each other and distribute tasks amongst themselves without the need for Ansible. This tool needs to be lightweight hence we have chosen OpenMPI for this purpose. The Ansible Playbook will contain OpenMPI[5], its base packages and the computational tool we will use to test our system which in our case is the JohnTheRipper Jumbo version, used for password cracking.

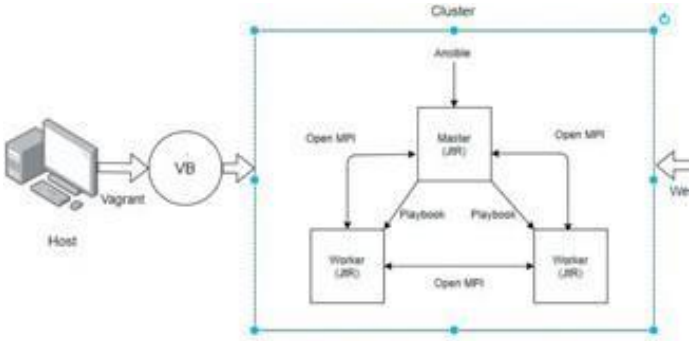

Fig. 1. System design of the HPC

4. Methodology

Installation of Vagrant and Ansible on the host machine is the basic requirement, to begin with. As established we require a cluster of virtual machines and to create and manage this cluster, Vagrant can be used, for this demonstration 3 virtual machines with 1 master node and 2 compute nodes are set up in the cluster, as shown in Fig 2. The configuration of the machine in the cluster is provided to Vagrant in the form of a Vagrant file. Vagrant utilizes a Vagrant box which is a packaged Virtual Machine. Here the vagrant box is Ubuntu. Each node is assigned a private IP and a hostname as an identifier. Ansible is installed on the master node, for this vagrant provide a Vagrant provisioner. The Vagrant Ansible provisioner allows you to provision the guest using Ansible playbooks by executing ansible-playbooks from the Vagrant host. Using this Configuration fixed in the Vagrant file cluster is set up.

```
$ vagrant status
Current machine states:

master                    running (virtualbox)
slave1                    running (virtualbox)
slave2                    running (virtualbox)
```

Fig 2. Status of nodes Created by Vagrant

The next step is to install all the required libraries and tools on all the provisioned nodes, as shown in Fig 3, now as Ansible is installed only on the master node, ansible will require some connection details of the other nodes for installation of libraries, this is provided using an inventory file which contains all the IPs of the machine to be considered for installation.

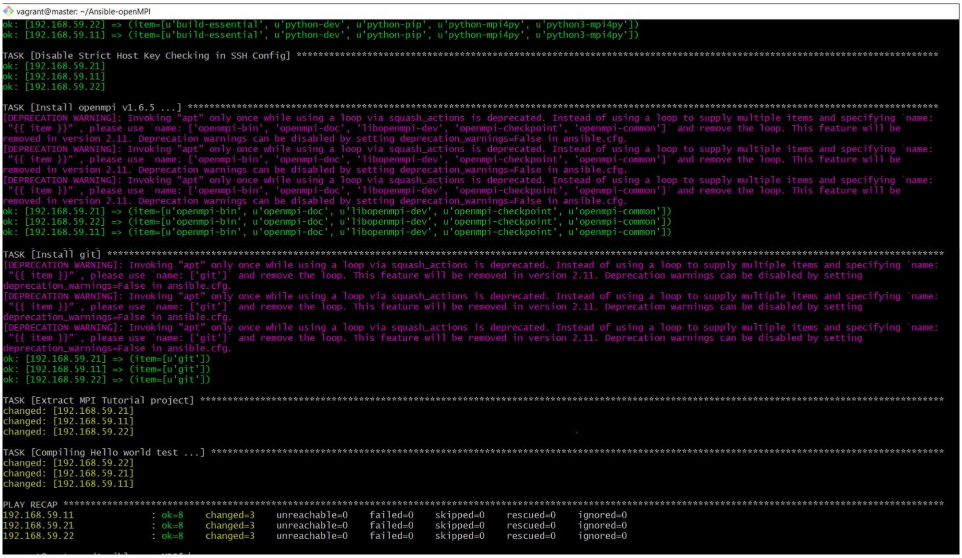

Fig 3. Ansible playbook running on multiple nodes

Also just specifying the IPs will not suffice for a swift connection between ansible installed nodes and other nodes, passwordless login using an SSH key will be a suitable solution for establishing a swift connection. To enable passwordless login we create tasks in an ansible playbook and run it on all machines, as shown in Fig 4.

Now the next task in line is to install OpenMpi Library, OpenMPI[5] is an open-source library that allows all the nodes in the cluster to communicate with each other. This communication is a crucial part of this high-performance cluster[11] as to achieve higher computation power the nodes need to work together by sharing tasks and working as a team.

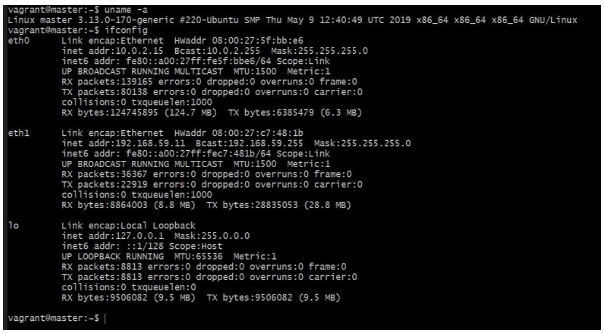

Fig 4. Master node enabled for password less SSH into compute mode

With the OpenMPI library[5] the brute-forcing tool for password cracking to be chosen need to be OpenMpi compatible to leverage the computational power of all virtual machine together.JohnTheRipper(JtR)[12] is one such security auditing and password recovery tool that serves the capability of employing the brute-force method and is also OpenMpi capable. Installation of OpenMpi library and JtR[12] on all nodes is achieved using Ansible playbooks and inventory filesWith all the requirements met and the system configured, the main job of password cracking using the computational power of a cluster of VMs can be done.

5. Results

As observed in Fig 5, for a three-lettered Linux password, the cluster setup cracked the password three times faster than what a single node could achieve. The time required to crack the password has significantly dropped. Subsequently the time required to crack the password can be further reduced by creating and utilizing more worker nodes in the setup. Thus, the time to crack the password is directly proportional to the number of worker that are attached in the setup.

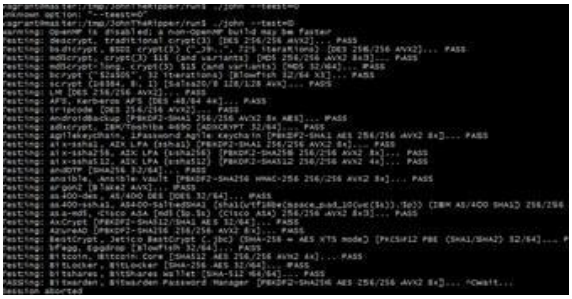

Fig 5. Time taken by single node to compute hashes

6. Conclusion

In line with the results above we can confidently conclude, use of distributed systems can provide exponential compute capabilities as compared to single machine. Moreover, automated provisioning and spawning of the system on real time user demand makes the entire process much more efficient. The proposed system is capable of automated spawning and provisioning of VMs which can be used for compute-intensive processes. The above results draw out a comparison between single machine and a cluster on computing power as well as compute time. The results suggest that in line with the number of VMs spawned we have exponential growth in computing power and an exponential decrease in compute time. The system can be used in an enterprise setup and computing which used to take days and months with this setup will be reduced to hours.

7. Future Scope

To create a High Performing Cluster(HPC)[10] which can be used in an enterprise set up the system should be able to cater to the dynamic, ever-changing requirement of performance[6] [9][14] and infrastructure. This can be further achieved by introducing dynamic spawning of VMs as per user requirements. The user can have the capability to define a threshold within the system according to which the system can spawn VMs for computing and when the requirement changes in real-time this should lead to the spawning and ceasing of VMs as well. This work was carried out on an open-source tool OpenMPI[13] however to make any system MPI capable it needs to be programmed along with MPI. This is a constraint that can be overcome by creating or using a tool that provides similar functionality but is easier to configure and does not require implementation level changes of the computational tool. As mentioned in methodology and architecture, Vagrant was used for spawning and Ansible on the master node to provision. However, this can be further reduced by using Linux[1] as a host machine on which Vagrant itself is tied up Ansible and Vagrant itself can provision the cluster during spawning it.

References

[1] Remya O, Nirali Verma (2021), Linux Cluster for Parallel Computing To Implement Data Mining Algorithm, K-Means with MPI - International Journal of Scientific & Engineering Research

[2] E Alfianto, A Sa'diyah, F Rusydi, I Puspitasari (2020), High-Performance Computing (HPC) design to improve the quality of Introduction of Parallel Computing lectures - IOP Conference Series: Materials Science and Engineering.

[3] S. Karthikeyan , R. Manimegalai Authors Info & Claims (2020), Implementation of multi node Hadoop virtual cluster on open stack cloud environments - International Journal of Business Intelligence and Data Mining.

[4] Huan Zhou, Jose Gracia, Ralf Schneider (2019), MPI Collectives for Multi-core Clusters: Optimized Performance of the Hybrid MPI+MPI Parallel Codes.

[5] Thananon Patinyasakdikul; David Eberius; George Bosilca; Nathan Hjelm (2019), Give MPI Threading a Fair Chance: A Study of Multithreaded MPI Designs.

[6] s. Sampath, Bharat Bhushan Sagar, B.R. Nanjesh, C K Subbaraya (2019), Performance Evaluation of Parallel Applications using MPI in Cluster Based Parallel Computing Architecture.

[7] Dr Raghavender K V (2018), Multi Node Cluster Network over LAN in Analyzing Big Data - Hadoop Frame Work - ISSN 2319 – 1953 International Journal of Scientific Research in Computer Science Applications and Management Studies -.

[8] Dereje Regassa 1, Heonyoung Yeom 1 and Yongseok Son 2, Harvesting the Aggregate Computing Power of Commodity Computers for Supercomputing Applications

[9] Sayma Sultana Chowdhury, Marium-E-Jannat, Abu Awal Md. Shoeb (2019), Performance Analysis of MPI (mpi4py) on Diskless Cluster Environment in Ubuntu - International Journal of Computer Applications.

[10] Karlovassi, Samos (2007) - Parallel Password Cracker: A Feasibility Study of Using Linux Clustering Technique in Computer Forensics.

[11] Tyler Lubeck, Ming Chow, Distributed Password Cracking with John the Ripper Computer Security

[12] Christian Frichot, An Analysis and Comparison of Clustered Password Crackers.

[13] Edward Roderick Sykes, Michael Lin, Wesley Skoczen (2005), MPI Enhancements in John the Ripper.

[14] Chee Shin Yeo, Rajkumar Buyya, Hossein Pourreza, Rasit Eskicioglu, Peter Graham, Frank Sommers (2006), Cluster Computing: High-Performance, High-Availability, and High-Throughput Processing on a Network of Computers.