Logic

-In our master machine (process id 0) we will read shadow.txt file and look for the username inputted by the user, once we find the information corresponding to our required user we will use tokenize function to parse the sentence to get salt and encrypted key

-Then we send the length of salt, length of key, salt and encrypted key to each of the slave processes to use to search for password.

-We will dynamically divide passwords among the slaves based on password lengths e.g passwords of length 1-2 will go to one process, 3-4 will go to another and so on

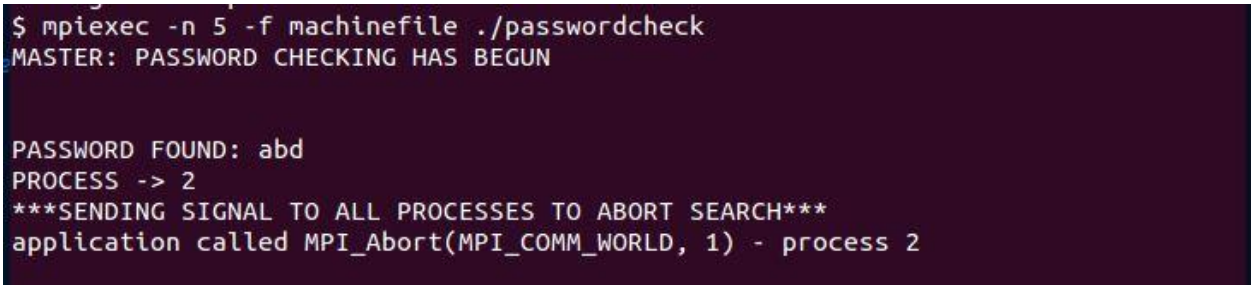
-In case of 8(max length) not being divisible by number of processes the remainder will be taken up by the master itself to search. e.g if number of slaves is 7, passwords of length 8 will remain so master will search through them itself.

-To search the password a function runs which tests all possible combinations of alphabets for its allotted length and returns true or false if password is found or not.

-If password is found, the slave calls MPI_Abort to stop all other processes from continuing their search.

Screenshots with explanation:

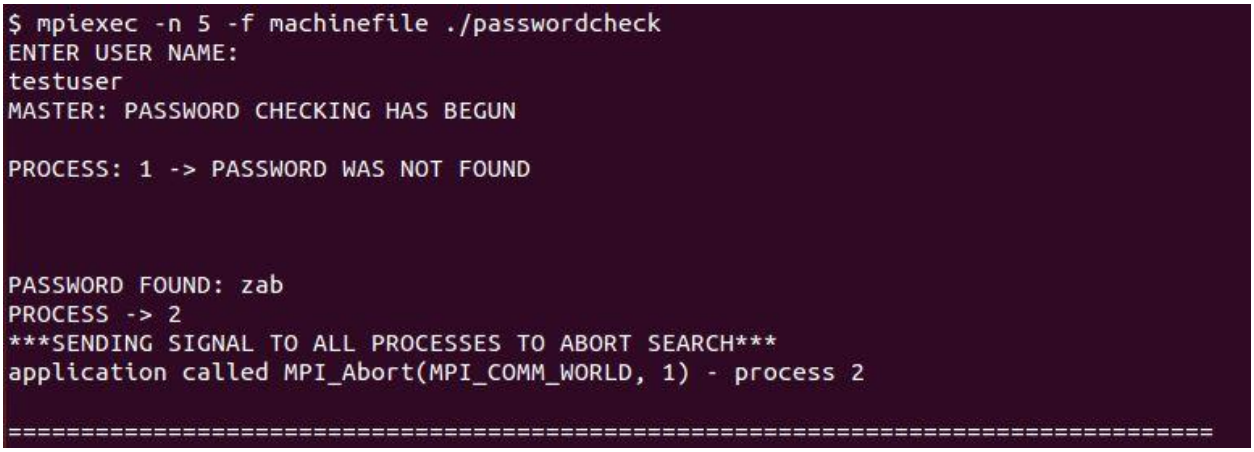
Screenshot 1:



Explanation:

Above is the screenshot for password 'abd' using 3 slave VMs. Passwords search was divided between 5 processes, one of which being the master which did not conduct any search.
Process 1 search passwords of length 1-2, process 2 searched 3-4, 3 searched 5-6 and 4 searched 7-8. Thus 'abd' lies in range 3-4 thus was found by process 2.
Username was hardcoded in this example.

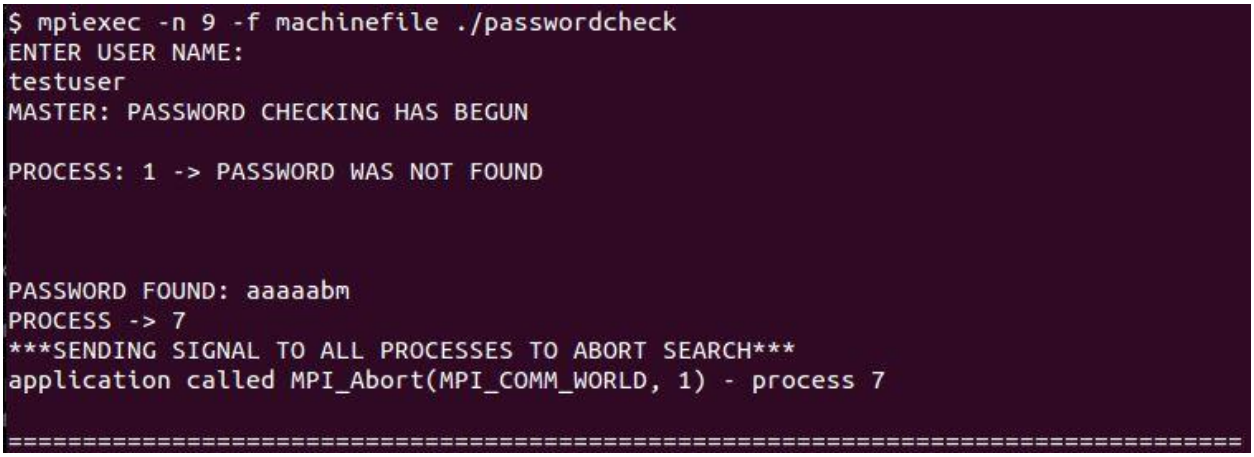
Screenshot 2:



Explanation:

Above is the screenshot for 'zab' using 1 slave VM. Password search was once again divided between 4 slave processes. Username was not hardcoded in this example but instead taken as user input and hash and salt retrieved from shadow file. Answer was found by process 2.

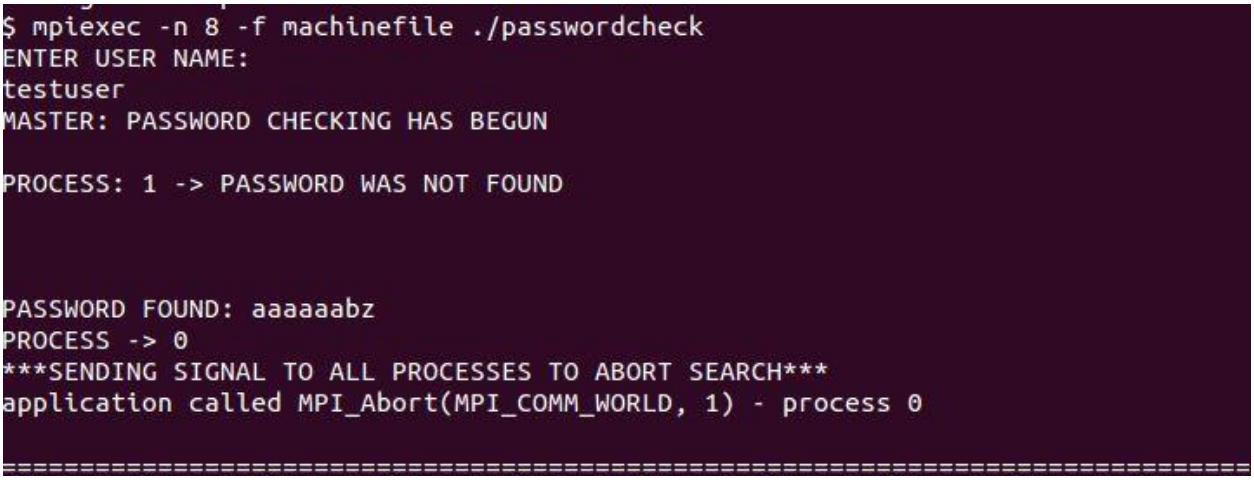
Screenshot 3:



Explanation:

Above is the screenshot for search of password 'aaaaabm', which would be impossible to find in seconds using a simple machine but we were able to find in seconds using a distributed system. We used 2 VMs, and 9 processes, 1 master and 8 slaves and each slave searched through all combinations of a given length. Since our password was of 7 alphabets, process 7 which was assigned length 7 was able to find it.

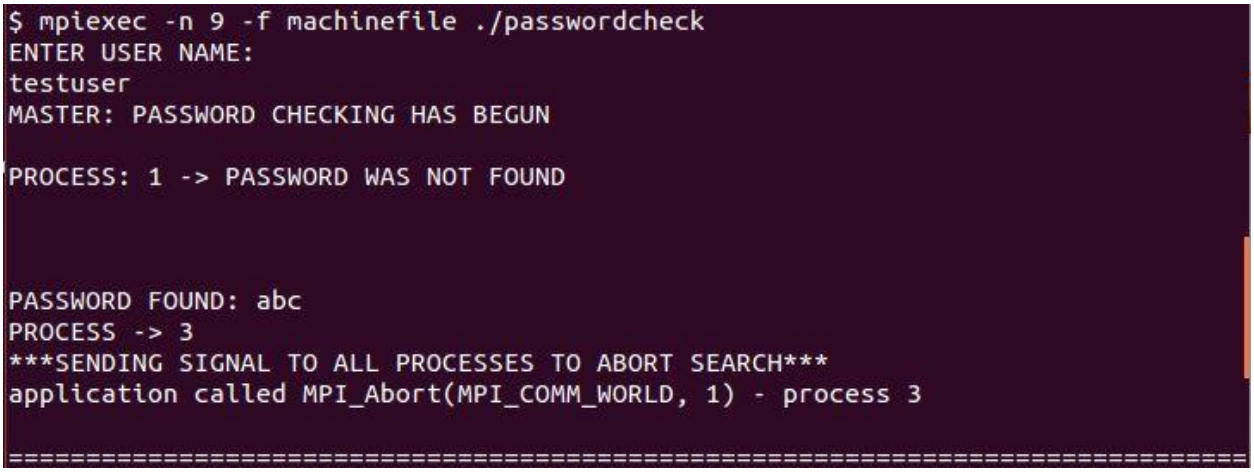
Screenshot 4:



Explanation:

Above is the screenshot for search of password 'aaaaaabz', which is of length 8, using 2 VMs. Since we gave divison between 8 processes there will be 7 slave processes which cant equally divide 8(max length of password) so master process will take the remaining passwords that were not divided which in this case are all passwords of length 8, thus process 0 i.e our master process has found the password as shown.

Screenshot 5:



Explanation:

Above is the screenshot for searching password ‘abc’ for user ‘testuser’ using 2 VMs, and 8 slave processes. Process 3 finds password and search is terminated.

Requirements:

- ✓ Cluster with multiple VMs
- ✓ Retrieving information from shadow file and parsing to get salt and hash
- ✓ Efficient and dynamic task distribution and load balancing
- ✓ Use of proper comments and good code
- ✓ Core functionality completed – password cracked successfully