Brute Force Grid

Project Proposal – CTP 2009

Project ID:

Group members:

M.F.F.Faraj	DCN07C3 / 0619	14451524
W.A.H.Jayawilal	DCN07C4 / 0775	14451825
W.A.R.Lasitha	DCN07C3 / 0634	14451553

Supervised by:

……..
Mr. Lakmal Rupasinghe

Date of submission: 20/05/2009

Abstract

Following pages describe a smart and specialized approach that facilitates one to recover a password in a very efficient manner. It is a grid network implementation which delivers right information quickly, giving the user a unique experience along with the maximum user satisfaction. Unlike the Old-school method of single CPU concept which disheartens its usage due to its slow execution of time, this will really encourage the user due to its revolutionary speed up of the recovery of the password.

The reason we decided to come up with this research is very important, it is not having a password recovery approach implemented using brute force and grid computing concept so far. Brute force method is widely used as a decrypting mechanism in recovering passwords, even though it takes a lot of CPU processing, but yet there is no multi-CPU solution over a public network to enhance its functionality. So we decided to use above mentioned technologies (Brute forcing and Grid computing) to come up with a versatile, scalable and efficient Solution to recover passwords.

First we implement a distributed Grid overlay network as the basic platform which the brute forcing is done. We decided to design our own Grid overlay network, rather using free grids available on internet so that we can give more specialization in brute forcing with simple infrastructure features to build the main platform. Then we will design a peer software and plug-ins to do the brute forcing on top of the grid overlay network. Current technologies like P2P and hashes will be influenced when developing the Grid overlay network, developing the peer software and developing the decryption plug-ins.

Thus with the speeded up processing and the fast response of the grid network, the user will be benefited by being able to get his work done in very efficient way. The simple infrastructure and user-friendly software will suggest a truly refreshing experience. The full details of this fascinating solution are discussed in detail in the pages to come.

Table of Contents

1. Introduction

Brute force Grid is the project that we have planned to do for the final year of Curtin university undergraduate program. This project consists of the grid network which is used to share the CPU power of the specific host with other.

Grid computing is the application of several computers to a single problem at the same time. Usually to a scientific or technical problem that requires a great number of computer processing cycles or access to large amounts of data. [1]

A **brute force attack** is a method of defeating a cryptographic scheme by trying out the large number of possibilities of passwords or keys. In almost most of the cryptographic schemes, the brute force attacking key recovery is possible. But it is set up in such a way that it would be computationally infeasible to carry out in a single CPU. The infeasibility relies on the key size and the encryption mechanism used. [2]

We have used the Grid technology to build our focused project which is specialized for brute force attacking for password recovery. Project consists of the grid computing modules which are developed specifically for the project. This underlying overlay network technology is used to implement the application of brute forcing. We have planned to implement our own grid computing network to minimize the overheads those were involved.

We have customized the basic grid computing network modules to work out efficiently in the project of brute force password recovery. So we have consumed the technology used in the peer to peer file sharing to facilitate the distributed manner of the network. This Is much important in this project, as we are targeting the underling network of this grid overlay network is the public internetwork of internet. The security issues and the possibility of accessing the host peers at each end of the earth should be done efficiently as well as with much care. [6]

As the peer to peer file sharing technology is the primary successor of the today's internet world. We thought it might be the exact sample that we should take into account while building the public grid network on the application layer.

Application layer overlay network differ from the actual grid network which is created normally at the network layer. This is because we should have to use the different network topology underneath the actual grid network which we have planned to implement on top. [3]

This method of developing the grid network gives the designer more elegant and handy perspective to develop the overlay network as he or she wants it to be done. We haven't use any existing grid network frameworks as to minimize the overheads of that network technology implemented. This may lead to number of disadvantages as well as many advantages too. These advantages and disadvantages will be discussed along the proposal.

The project is focused on the brute force attack and the grid network. But we actually want to demonstrate the grid network is effective in the brute force attack scenario, which is implemented in the public network, as a distributed overlay network.

The normal way of doing the brute force attack is doing it in the brute force attack is by using a single PC or main frame computer. The enterprise versions include the grid networking system under one roof to crack down the passwords and keys. [2]

But here we are doing a complete different approach to the grid network password recovery. We use the same old Grid network which is used in the enterprise version password recovery solutions, but the project uses that grid network concept in the public internet arena, and utilized to the specific task of brute force attacking only.

The new approach that is being planned to take over by the project makes the brute force attack key recovery, more efficient quick and simply has no limitation over the internet to spread over. In contrast with the other enterprise solutions, Brute force Grid may provide a free friendly environment for the password recovering parties all over the world to make them hold hand in hand and work in a group to make the success.

The concept of humanity is mainly focused in the project. The users are expected to conduct only legal password recoveries over the system. And this system is solely dedicated to ethical hackers and the student educational purposes. Any illegal act done using this system may not make the designers of the system to be liable of.

2. Background

Grid computing is the use of several computers to execute a solution for a single problem at the same time. And Grid (Distributed) computing is a type of parallel computing. This is used in problems that need great amount of computer processing power which deal with a large amount of data. In case of solving a problem using grid technology have to distribute the program among computers, the software that using in grid divide and apportion the pieces of a program among several computers and that computers do parallel processing(apportion pieces). This gird technology is used in commercial enterprises for such diverse application as economic forecasting; seismic analysis and drug discovery. The primary advantage in this technology is this distributed computing can produce similar computing resources to a multi-processor supercomputer at lower cost. [1] [5]

Password cracking is the process of recovering password to access data or transmitted by computer system. The purpose of this cracking is to recover forgotten password or to gain unauthorized access to a system. There are some main attacking methods dictionary attack, guessing, brute force attack and pre-computation .In guessing method; password can be retrieved by a human who knows personal information well. In dictionary attack, user chosen passwords are derived by cracking programs armed with dictionaries, and the user personal information's. Pre-computation involves hashing each word in the dictionary and storing the word and its computed hash in way that enables lockup on the list of computed hashes. In this method password recovery is very fast and very useful for dictionary attacks where salt is not used properly. A last resort is to try every possible password within a given range, that's called brute force attack. In nowadays brute force attack is done only on a single machine. [2]

In our project we are going to reduce the time taken to recover the password from brute force attack by increasing the resources. The cost goes high with the increase of the resources. So we implement a method to do the brute forcing using a grid network, Likewise said before gird network can produce similar computing resources to a multi-processor supercomputer at lower cost. So it save time taken to recover password using more resources than a single machine and no fee if we use 3rd party grid network. Brute force attack in distributed manner is not used before so this could be new improvement in password recovering

3. Objectives

The project brute force grid is planned to design to meet some needs of sub targets. These sub targets may centralize in to the main target of the project.

If we have the top down approach in organizing the objectives and targets of the project, the project brute force grid is primarily designed to provide the free service of password and key recovery. This key recovery from the hashes was done using the brute force attack. But apart from the traditional brute force attack we have use the new way of approach to this issue.

The new approach is to use the grid network system that overlays the physical network underneath. The physical network is the internet. So the projects objective is to be able to be run on both the LAN and public network environment to provide the logical overlay grid network. This is use to provide the distribution of the process load between the hosts within the gri network to provide the efficient password and key recovery from the hashes.[5]

The objectives of the project further moves in to the field of decryption. We are planning to design the plug-in based decryption mechanism which is used to decrypt the hashes and gain the keys. This plug-ins has the standard way of interacting with the grid peer software which is planned to be designed separately in java. This plug-ins should be much efficient as they are the ones which actually interact with the processing part of the project.

Users get the ability to create the plug-ins that they want. So the system should be able to work independently for any plug-ins that is used for the above mentioned purpose. The grid peer shouldn't have any idea of how the plug-in is working. And this may lead to the transparency of the developer of the grid network and the plug-in developer.

So now we can conclude the major objective of the project by defining that, the project is to be created in order to gain the quicker key recovery from the hashes using the technology of the Grid networking. The hash recovery should be done using the plug-in modules which is transparent to the main system of the grid.

4. Procedures

The project Brute Force Grid is system which is deployed to ease the process of password recovery using the Grid network system. But other than currently presented Grid systems for brute forcing, this brute force grid is using the public internetwork as the working arena.

The system was to be developed in multiple sections. We are planning to fragment the project into several main components and to sub divide the main components into several modules. Then develop the modules separately and integrate into one final component. Each component is tested individually thus to go for the maximum accurate throughputs. As we consider that finding the bugs in the smaller sectors reduces the complexity of the program when it comes to a whole bunch of coding at last. So the all components will be tested module-wise in each development cycle and will be finally debugged as a one complete product after the component integration.

The project Brute Force Grid has several main components. As I am planning not to be specific in the technical terms, the need of the technical description is considered as a minimal need to show the working procedure.

4.1 Technical specification

The project totally consists of the software section. There is no hardware section available in this project. The system runs in the java platform to make it platform independent. The major drawback of the system when we do it using the java is the speed. The speed is a must when it comes to hash decryption. The module which is used to do the decrypting of hashes should be done in higher efficiency. This efficiency is maintained by using the programming language of C to design. So the project consists of two programming languages; Java to deploy the grid network system and the C to deploy the decryption modules.

We have fragmented the modules of the grid network in to two main parts.

- **Grid peer**
  - Is the peer end software which is acting as the agent for that specific host to that grid network
- **Server**
  - Server is used by the grid peers to initiate the connection with other grid peers as there is no other mechanism to identify the newly connected grid peer to know what the other neighboring peers are.

4.1.1 Grid peer

Grid peer plays the major role in the network. It's fragmented in to several components to ease the process of development. The main components are Database and Working components.

4.1.1.1 Database

1. Grid Peer List (GPL)
   - Used to store the currently active peers this includes the socket to that peer and the connection ID which is used by other storage components in the Grid Peer,

2. Report Server List (RSL)
   - Logical database of the peers in GPL which is currently serving the local job.

3. Report Client List (RCL)
   - logical database of the peers in GPL which is being served by the local host,

4. Report IP List (RIL)
   - This is the physical storage which is stored in a specific file for long term use. This consists of the IPs of the host which is considered as an up-peer of the grid, which can be connected to get into the grid network when the peer software is newly awakened.

5. Remote Job List (RJL)
   - The Job identifier list, which the jobs were received from the remote locations to the local host grid peer.

6. Local Job List (LJL)
   - The jobs that were created by the local host peer in order to execute the brute force in the grid network.

7. Common register
   - Use to store the constant values that are used to initialize and continue the working procedure of the Local Grid Peer. These values are also stored in the physical media to conservation for next boot of the Grid Peer.

4.1.1.2 Working components

1. Remote Peer Connector
   - Check out the IPs in the RIL and try to connect to the. If succeeded, add the peer to the list of GPL. This also handles the unreachable IPs for a specific time period and removes it from the list.

2. RIL filler
   - Use to create the list of RIL using the web server's list and by the GPL list available in the neighboring grid peers.

3. Remote Request Collector
   - This component is used to accept the remote requests from other grid peers in order to do the remote jobs of them. The component is required to call the plug-in check, take the information's of the current CPU usage and the maximum allowed remote requests which were preset by the local user.

4. Job Scheduler
   - This component is use to schedule the remote jobs and local jobs, giving the priority to the local jobs. Uses a Round Robin(RR) type scheduling algorithm.

5. Information requester
   - This component is use to request the remote peer about the given jobs state. The information responder component in the other side of the network responds to it.

6. Information responder
   - This component is responsible to give the appropriate state information about the remote jobs to the peer that the job is received from.

7. Local Job intake
   - This component works in local host peers only. This is responsible to intake the local user jobs to the local peer and enqueue them in the LJL in the state of waiting. This module checks the ADL before in taking any jobs, and the local user is responsible for selecting the appropriate hashing mechanism which is used in all the other network peers to do that job.

8. Local Job starter
   - Local job starter is used to enable the enqueued jobs in the LJL. This component check the GPL and get the available peer list, and create the job requests for the remote peers, after checking the maximum connection allowed in the local peer. This maximum count is store in the Common register.

9. Remote Job Stop requester
    - The component which is used to stop the remotely active job of the local host peer of the grid. This informs the remote host peer to stop the given job as the password is recovered in some other peers or the job was cancelled in the job initialization peer.

10. Remote Job Stop responder
    - The local component, used to deactivate the selected job to be disabled after the request was received from the remote requestor which initiates the job.

11. Remote connectivity monitor
    - This is an automated component runs in the local host peer in order to check the failed connection to the other peers and inform the connection initiator. This component checks the host currently connected. This is stored in GPL. So this goes through the GPL and find out the down linked peers and remove them from the list. It also alters the RSL, RCL, RJL and LJL according to the need of the altering done in the GPL.

12. Plug-in check and updater
    - Plug-in checker is the component that checks the new plugging available in the internet and downloads them to the local host peer according to the need of the new job which is requested by the remote client. This is invoked by the new request from the remote client. When such an event occur the component check the ADL and if it is not available it downloads from the web server.

13. Web updater
    - This component is use to notify the web server about the present of the grid peer, then which IP may be used to give to other peers which request for active IPs to connect to the Grid network.

4.1.2 Web server

Now if we look at the server, the web server, it's quite simple. We have made the server simple as to minimize the centralized manner of their distributed network.

Server is responsible for maintaining a list of active peers and provides the peers when needed. And the next most important thing that there web server does is, it has the decrypting plug-ins which may be download to the peers when needed. This eliminates the security issues like virus spreads or other mal-ware execution on the grid, if we take the plug-in copy from the remote client which requests.

Server is divided into three components as of the service it's providing

4.1.2.1 Website

- This contains the general website to give the knowledge about the brute force grid to the public. They can download their own free version through this site.

4.1.2.2 Peer list

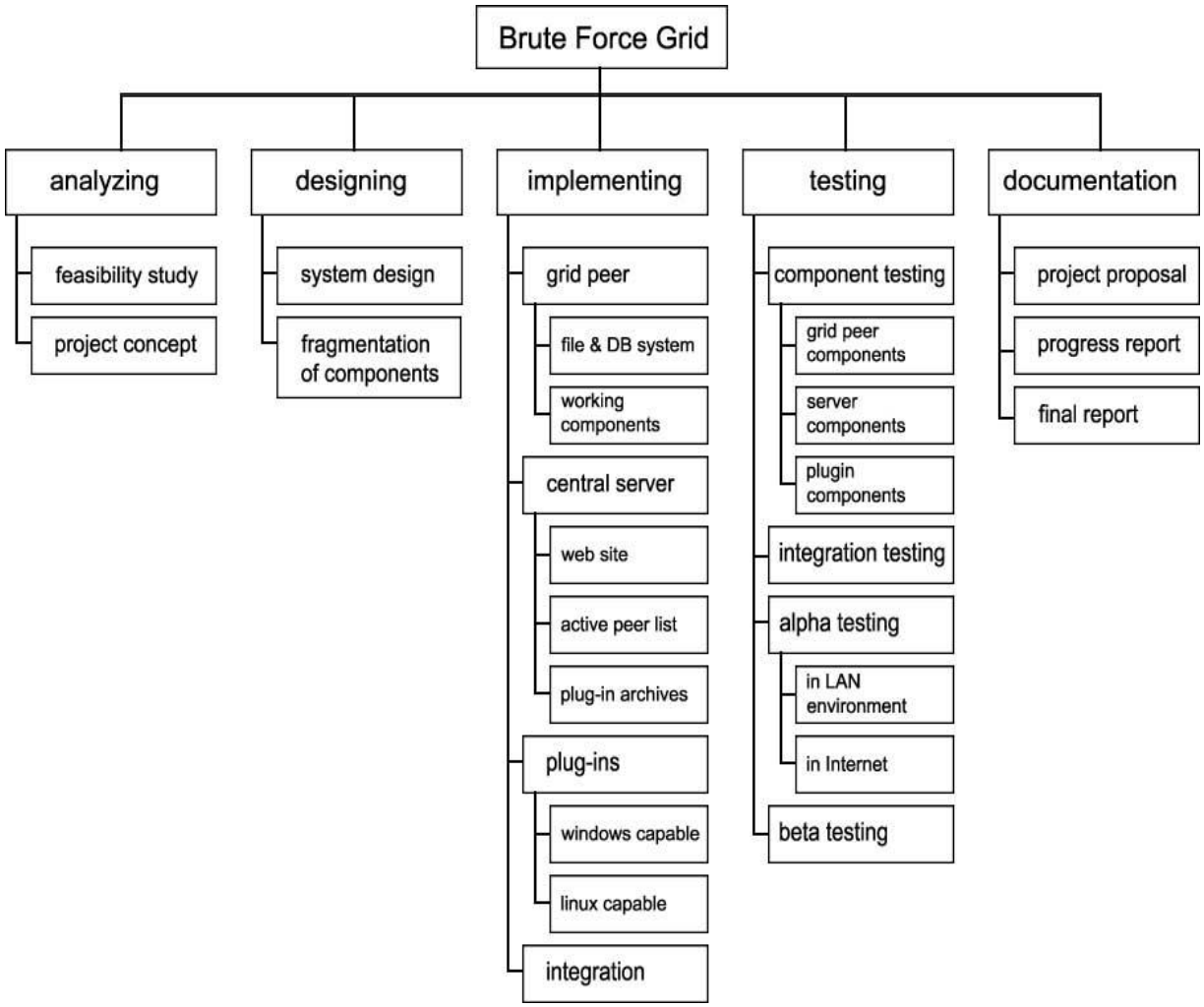
- This contains the list of currently presented peers in the internet. Each peer requests the web server periodically with the information of presence. If the peer fails to presences the information on a given period of time, the server considers it as the dropped peer and removes it from the list.

4.1.2.3 Plug-in list

- This component is nothing but a small downloadable file portal which is stored inside the web server. The list of plugging currently available is listed in the server. So when the peer want to download a plug-in it goes thought the web list and check the correct file to download. Then creates a socket to the web server and downloads it.

The above mentioned components are the fragmented items available in the project. So when we design the system we hope to provide a work distributed structure and the timeline for each and every component. Then create the grand chart of the all available components which is created up to the time limit. Time to integration and the testing will be also included in the chart to provide the appropriate timing constraint to the development of the system.

4.2 Work breakdown Structure



Below we have mentioned the approximated time required to complete the each task of the components and the person who is going to develop that component.

We are having 3 team members in our project team, including Faraj, Harsha and Ramitha. And designed the work spread into these categories:

4.2.1 Analyzing

- Analyze the project requirement and check out the needs and wants of the consumer side to the project to make it a success. The team discussion on this topic is already done when the project topic s selected. So there is no time allocated in the time line which is used to in the future development stage.

4.2.2 Designing

Design the total project concept. How it's going to work and how we going to deploy the implementation. This was pre designed by the team leader and then submitted to the team members on the discussion. The team's members come to the conclusion after altering the design to optimize the system performance, efficiency and effectiveness.

4.2.3 Component breakdown

Breakdown the project into small components and sub modules to ease the process of development and debugging. This was done by the three team members as a group discussion. The design which was created by the leader and then altered by the team members is used to develop the components to develop the system.

4.2.4 Component development and testing

Develop the components individually as the team leader specifies the team embers the tasks of components and sub modules. This task is done by the whole team members as to target the starting of the integration and testing. After that the integrator gets less component development. Mean while the process of documentation is done parallel and a specific team member is assign to integrate the documentation part which is done by the team members. This will be discussed shortly after the some other topics below.

4.2.5 Integration and testing

The completed components were checked which each and every component is testing individually after the component development stage. These components were integrated in to a single module of the system and tested with dummy components for the ones which are not deployed yet. When the component is developed completely it is use to substitute the dummy component and integrated to the system and tested.

The integration testing is started in the middle of the project time line as there should be some components to initiate the testing. So until the projects main components are developed this phase kept idle. Three members were assigned the project components to develop until the minimal required components were finished. After that a specific team member (Faraj) is assigned the integration and testing while the other two team members do the component development as their major parts.

4.2.6 Include enhancements (optional)

The enhancements are hoped to be added to the system if the time permits. There is lots of enhancement that we are planning to do to our system but we are not mentioning

them in the proposal as in order of having the negative option.

These enhancements are discussed after the projects primary requirement is fulfilled. We keep the space for the future enhancement of each component while we develop the components.

4.2.7 Debugging

After the completion of the project the total project will be debugged for a considerable amount of time. As we think this is the biggest phase of our project we are planning to give more priority to debug the system and fix the maximum possible bugs as could.

This debugging is done by the whole team, and done for a specific amount of time. In this phase no other features are added to the project as enhancements. The components developed by the specific member are given a chance to debug the components.

4.2.8 Specific Testing

The LAN segment is use to test the whole system to find ongoing working bugs. We are using the LAN testing to eliminate the need of having more bugs when we test ht system in the internet. More over the LAN can be seems as a mini under control version of the internet as we look at the perspective of the system. To do this we are planning to choose a LAB in SLIIT and few PCs connected to a specific LAN in it.

The further utilization is taken before deploying the system in the public network for testing purposes. The bugs that were identified were debugged and if the specific module or component should be changed specifically the specific member of the team is given the module or component to debug it.

4.2.9 Public testing and get the feedback

We select the students from the SLIIT campus who has the public IPs of the internet. Or ion there words having the ADSL connections. And let them try out the system. This phase is considered as the alpha testing of our project. The feedback was planned to take from the students and further develop the bug free environment for the brute force grid system.

4.2.10 Final fixation

After the alpha testing in the internet, the final fixation is done to the project, getting the most possible alpha testing feedbacks. Here we consider not including any further

optional to the project but only to fix the correct options problems which were foreseeing the alpha testing phase. This process is done by all three group members.

In this phase no further component base debug is made. The whole system is considered as one unit and the debugging is done. If there occurs an unpredictable situation where there is no chance of doing the fixation at the high level of perspective, then the specific team member is given the component for fixation. This is tried to eliminate in the above mentioned debugging phases to success the final fixation in the high level manner.

4.2.11 Beta testing

At this level the project is almost completed and the beta testing is done in the internet. The project is host in the web server and the general public gets the chance to get their hands in the system.

4.2.12 Documentation and report writing

The other major part in our project is the documentation. As I mentioned earlier this documentation should be done parallel with the project implementation. So the one who designs a specific component is responsible for the documentation of that specific component.

But this may lead to a situation where the documentation style differs from one component to a nether. So initially we are planning to develop a standard method of documentation for each and every component. So the team member who is writing it should follow the initial guild lines.

More over we are also assigning Harsha to take the responsibility of documentation integration. So when a team member completes a component design, he should submit the component code to Faraj and the documentation to Harsha. This documentation is altered for the standard format by Harsha and integrated into the main documentation. The project integration documentation is done by Faraj as he is responsible for the project integration part.

The report is planned to be written when the public testing phase is started. While the alpha testing is on process we are planning to do the project report from the documentation that we have done so far thought out the project development phase and the debugging phase.

4.3 Gantt chart

Gantt Chart

	Task Name	Duration	Start	Finish	Predece
1	Analysing	6 days	Mon 6/1/09	Mon 6/8/09	
2	Designing	10 days	Tue 6/9/09	Sat 6/20/09	1
3	Component breakdown	7 days	Mon 6/22/09	Tue 6/30/09	2
4	**Component development and testing**	**32 days**	**Wed 7/1/09**	**Thu 8/13/09**	3
5	Database and file structures	10 days	Wed 7/1/09	Tue 7/14/09	
6	Grid Peer Components	22 days	Wed 7/15/09	Thu 8/13/09	5
7	Web Server	10 days	Wed 7/1/09	Tue 7/14/09	
8	decryption modules and plugins	20 days	Wed 7/1/09	Tue 7/28/09	
9	Integration and testing	30 days	Wed 7/15/09	Tue 8/25/09	5,7
10	Pre testing	5 days	Wed 8/26/09	Tue 9/1/09	9,7,6,8
11	Include enhancements	10 days	Wed 9/2/09	Tue 9/15/09	10
12	Debuging	18 days	Wed 9/16/09	Fri 10/9/09	11
13	Specific LAN Testing	10 days	Wed 9/16/09	Tue 9/29/09	11
14	Public tesing and get the feedback	10 days	Mon 10/12/09	Fri 10/23/09	13,12
15	Final fixation	6 days	Mon 10/26/09	Mon 11/2/09	14
16	Beta testing and debuging	10 days	Tue 11/3/09	Mon 11/16/09	15
17	Documentation and report writing	101 days	Mon 6/29/09	Mon 11/16/09	

4.4 Implementation overview

As we have mentioned before the project is about the implementation of the grid network which is implemented over the physical network using the TCP/IP protocol stack under it. This grid uses the overlay networking method to keep track of the other grid peers of the network. This is the same method implemented in the peer to peer file sharing technology in the commonly known P2P applications. We have planned to make the more likely technology used in the P2P file sharing to share the CPU power between the peers. [4] [7]

The data logical and physical database was kept to store the grid peers and the client's role played in the system. These data modules are later used in the all system components to maintain the connectivity and job scheduling.

The central database web server which is used to store the centralized information is presented in the public network. This system contains the currently active IPs. But we are planning to make the system decentralized as possible. So we are planning to reduce the need of the server for minimal level.

The plug-in which were created in C should have separated plug-ins for different operating systems. So the plug-in are to be created as separate versions in order to work in different platforms.

User interface for the system is planned to design in the java, then which can be used to execute in any operating system platform.

5. Evaluation

After the development phase, the product evaluation begins. In doing it, the final product is tested under different platforms and environments. We plan to do the evaluation of the product in two different environments; The LAN and Public Network.

First the brute force grid performance is tested in the LAN environment to make sure the basic functionalities are working properly and how user-friendly the product is. We plan to use the SLIT LAN for this purpose and select random machines establish the gird network, and install the peer software in hosts of the grid network then set the necessary parameters to the software for the execution to get the solution. We plan to get feedback from the users by giving them a questionnaire and allowing them to do comments after they use the gird and get their responses about the performance and other aspects of the grid. We also store password protected data folders in grid network nodes for the further processing. The grid is tested several times using different passwords with different lengths and the time taken to recover is recorded. We also provide user the access to the web server to download our brute force software. After getting user feedbacks, they are thoroughly analyzed to and the user responses are derived. Necessary bug-fixings, modifications and changes are made to create a better, accurate and more user-friendly version.

After testing in LAN environment and correcting defects we came up with, next we go for the next environment to test our brute force grid. We host the grid in the public network (Internet) and allow pre-recognized users in SLIIT to test the grid in the public network platform. The user is given the access to the web server to download the brute force software and feedback web page for each user to get the user responses. The feedbacks are analyzed and the necessary changes are made to the product. This testing concludes the alpha testing phase.

The next phase, the beta testing is done allowing everyone to use the grid. The access to the web server through is given though a sign up process we allow registered users to download our software and run it on the host. Collect the feedback from user accounts and analyze it. We will be able to get a variety of feedbacks since the range of the users is very wide thereby showing us the aspects in which the software to be improved for a better solution.

6. Personal and facilities

Now let's have a specific view on each and every component designing to a specific team member. The work spread to every member of the team is a major part when considering the success of the project. Each and every member in the team has the different ability and specialization in specific parts. So we should identify them and assign the works to them as so for.

Likewise as we had mentioned before we have chosen Harsha to integrate the documentation and Faraj to integrate the components. Ramith develops some major decryption plug-ins; Harsha and Faraj were assigned to create the major grid components as well.

Below we have summarized mentioned the job assignment for each and every individuals of the team which is so far discussed in the above topics

1. Faraj
    - Database and file structures
        - ✓ ADL – Available Decrypted List
        - ✓ RJL – Remote Job List
        - ✓ Local Job List
        - ✓ Common Register
    - Grid Peer Components
        - ✓ RIL filler
        - ✓ Request collector
        - ✓ Common Interface
        - ✓ Plug-in check updater
        - ✓ Remote connectivity monitor
    - Design the web server and all its components that are to be used as the semi centralized server to the project.
        - ✓ Peer List
        - ✓ Plug-in List
        - ✓ Website
    - Project component integration and module and final testing

2. Harsha
    - Databases and file structures
        - ✓ GPL – Grid Peer List
        - ✓ RSL – Remote Server List
        - ✓ RCL – Remote Client List
        - ✓ RIL – Remote IP List

- Grid Peer Components
  - ✓ Remote Peer Connector
  - ✓ Job scheduler
  - ✓ Job stop requestor
  - ✓ Job stop responder
- Documentation integration
- Module Testing

3. Ramith
   - Grid Peer Components
     - ✓ information requestor
     - ✓ information responder
     - ✓ Job intake
     - ✓ Job starter
   - design of decryption modules and plugging
     - ✓ MD5 Plug-in
     - ✓ AES Plug-in
   - Module Testing

The above mentions workload is planned to distributed as so. But they may vary with other factors while the project execution. But the major parts such as documentation and integration will be done as mentioned above. They may not change.

7. Budget

The project BFG (Brute Force Grid) is budgeted as follows:

- Website and public server hosting costs

  We can get this service free of charge from the free PHP hosting websites such as **http://www.zymic.com/free-web-hosting/** [8]

- Internet costs for purchasing public IP – ADSL

  This cost about 1600 Sri Lankan rupees in SLT. We are using the testers from the SLIIT students, so this will also make no big costing issue to the project. [9]

Other than this we are using the free software modules and software development modules such as

- Netbeans
  - IDE to implement the project
- Java
  - The programming language use to implement the grid network
- dev C++
  - the IDE used to develop the plug-ins

As these products cost no cash values the project budget is set to the minimal level, and only circles around the above mentioned criteria.

8. References

- *[1] Wikipedia, "Grid computing", November 2008, public editable source, web page. Available: http://en.wikipedia.org/wiki/Grid_computing [Accessed: 12th May 2009].*

- *[2] LastBit, "Brute Force Attack", 2005, information gallery, ASP web page, Available: http://www.lastbit.com/rm_bruteforce.asp [Accessed: 12th May 2009]*

- *[3] Ying Zhu, Baochun Li, Member, IEEE, and Jiang Guo, "Multicast with Network Coding in Application-Layer Overlay Networks," application-layer overlay network, vol.22, no.1, 1 - 4, January 2004. [PDF]. Available: http://www.eecg.toronto.edu/~bli/papers/jsac-coding.pdf [Accessed: 12 May 2009].*

- *[4] Aaron Campbell, the p2p concept, November 19, 2005, Web blog, Web page, Available: http://dekita.org/articles/the-p2p-concept. [Accessed: 15th May 2009].*

- *[5] Andreas Mauthe, David Hutchison, "Peer-to-Peer Computing: Systems, Concepts and Characteristics" , Distributed Computing, http://www.wiiw.de/publikationen/PeertoPeerComputingSystems886.pdf [Accessed: 16th May 2009]*

- *[6] E. Adar, B. Hubermann, "Free Riding on Gnutellas", First Monday Peer-Reviewed Journal on the Internet, http://www.firtsmonday.dk/issues/issue5_10/adar/index.html, [Accessed: 16th May 2009]*

- *[7] A. Mauthe, J. F. d. Rezende, D. Hutchison, "N2N Connection Service: A Multipeer Service for Distributed Multimedia Applications", Expert Contribution to ISO SC6/WG4, ref-no. ISO SC6/WG4 4G11, 1996.*

- *[8] Zymic, "free web hosting", 2009, website, Available : http://www.zymic.com/free-web-hosting/ [Accessed: 19th May 2009]*

- *[9] SLT, Internet Access Via SLTNET, information website, Available : http://www.slt.lk/data/forhome/091sltnet_internet.htm [Accessed: 19th May 2009]*

The End