International Journal of Computer Applications (0975 – 8887)
Volume 169 – No.2, July 2017

Parallel High-Performance Windows 7 Password Recovery using Weak Migration in Intra-Platform Mobile Agents Technology

Dhuha Basheer Abdullah
Department of Computer Science
College of Computer Science and Mathematics
University of Mosul, Iraq

Manar Talaat Ahmed
Department of Computer Science
College of Computer Science and Mathematics
University of Mosul, Iraq

ABSTRACT

The quest for developing computer architectures in terms of accuracy, efficiency and cost is unending to tackle a very large-scale system, that have a performance problem, using the parallel processing. The mature technology of the mobile agents is gaining more momentum to be adopted with the parallel processing over a grid computing in contrast to the traditional parallel applications, that base on either a dedicated hardware or software. In this paper, the mobile agents technology was proposed as a new approach to achieve a grid computing-based parallel processing for Windows 7 password recovery problem by using the brute-force procedure and realizing how the mobile agents technology can be used to execute complicated applications from the High performance computation community. In this work a weak migration in intra-platform mobile agent was used. The experimental results demonstrate the computational power of the proposed system.

Keywords

HPC, Mobile agent, Parallel Processing, password recovery problem.

1. INTRODUCTION

Password plays an important role in an information security to protect the privacy and confidentiality of data on computers, despite of the increasing use of alternative techniques such as smart card, biometrics and graphical passwords [1]. It tends to be the only line of defense against the intruders of computer machines [2].

The users must use an easy to remember passwords such as using family names, hobbies, company names, etc. [3]. Forgetting the password is a common problem that the users of the computers may face, since the users can be confused among passwords which are used in different areas such as in the computer systems and web services [4]. A study of yahoo users found that the average of the used password for each user is seven different passwords. The same study reported that over a three-month period 2,149 out of 50,100 users (~ 4.28 %) forgot their passwords. Another study of students reported that one-of-third peoples did not remember their passwords, those people had an average 4.45 different passwords to use them to access 8.18 distinct applications [5]. Therefore, the unavailability of passwords requires a specific procedure to recover the password. In this case, the users will need to what is called a High Performance Computation (HPC) to implement this complicated problem [6]. The main goal of this work is to accomplish a reliable and a high performance password recovery system of Windows 7 based on the parallel processing and the mobile agent techniques.

1.2 Related Works

Several methods have been suggested over the past few years in the field of developing a high performance system for password recovery or password cracking problems, but to the best of the author's knowledge, there is no work combining the parallel processing with the mobile agents technology to address the brute-force technique-based password recovery problem.

In 2007, Johnny Bengtsson introduced a feasibility study in a High Performance Computing cluster (HPC-cluster) for Linux password-based cracking process. In this study, computer forensics analysis tools were developed for parallel cracking using two major techniques: dictionary based-attack and brute-force based-attack using the Message Passing Interface (MPI). This study intended to build a cheap cluster for setting up a HPC using the MPI [7]. Moreover, in 2009, Kostas Theoharoulis, Charalampos Manifavas and Ioannis Papaefstathiou designed a hardware system using the Field Programmable Gate Array (FPGA) to achieve a fast rainbow table attack based Windows password recovery. The proposed system can be more than 1000 times faster than the corresponding software using a highly parallel architecture depending on a fine-grained pipeline. The main drawback of the proposed system is that up to a couple of months may be needed to create an efficient rainbow table on a single machine. Besides that, the recovery process of this system based on the rainbow table attack where this attack does not coverage the overall password possibilities as the brute-force attack [8]. Besides that, in 2011, Martijn Sprengers developed and improved Graphical Processing Unit (GPU)-based an efficient and exhaustive search approach for attacking the authentication mechanisms that use password hashing schemes. The proposed approach was proved to be very efficient to lunch a massive parallel search compared with the traditional CPU. The experimental results showed that the proposed approach is more than 30 times faster than equally CPU implementation [9]. In 2013, Hsien-Cheng Chou, Hung-Chang Lee, Hwan-Jeu Yu, Fei-Pei Lai, Kuo-Hsuan Huang and ChihWen Hsueh proposed a model to generate probabilistic passwords sorted in decreasing order. This model called a TDT model since it consisted of a Training set, a Dictionary set and a Testing set using to reduce the search space of the password. The simulation results prove that the proposed model is 1.43 times higher than the John-the-Ripper attack software. Also, the authors designed a hybrid cracking system that combined many attacking techniques. The experimental results showed that the proposed hybrid system is more efficient than the TDT model for cracking passwords of UNIX operating system [1]. Also in the same year, Chrysanthou Yiannis proposed an optimized method of password recovery using various password attack techniques.

This method was suggested to use of Markov Chains and Dynamic Self-Adjusting attack in combination with the various types of attacks such as the brute-force attack, dictionary attack, hybrid attack and etc., using each type for its benefits. The proposed method was tested on the Phpbb password database that leaked in 2009. The experimental results showed that the proposed attack can run on the personal computer and recover over 75% of passwords in less than one hour [10].

2. PARALLEL PROCESSING

Parallel processing, also known as a concurrent processing, refers to a set of independent processors (also called as processing elements) which is collaborated to solve complex problems by partitioning these problems into independent fragments and distributing these fragments among the processors of the system [11][12]. Parallel processing has become engrained into different applications such as engineering and design applications, scientific applications, commercial applications and computer systems applications.

3. MOBILE AGENT TECHNOLOGY

Software agent is one of the most striking topics in computer science, and it is seen as a promising technology for designing and developing complex and distributed software systems [13][14]. An agent is defined as "a person whose job is to act for, or manage the affairs of, other peoples" [15]. In the context of computers, software agent refers to "an executable entity that performs certain tasks on behalf of the users" [16][17]. Software agent are endowed with the property of mobility to acquire the ability of migration the execution environment in a network aware fashion and roaming through network nodes to carry out tasks on behalf of users for software agents [15]. The software agent that exploits this propriety is called mobile agent. This propriety can be exploited to reduce the complexity of such contemporary applications. Moreover, mobile agents are supported by many underlying platforms such as Voyager, Tromoso and Cornell Moving Agents (TACOMA), P2P Interactive Agent eXtensions (PIAX), Trinity mobile agent framework, Float Mobile Agent (FMA), Secure Mobile Agent Rapid Development (SMARD), TAgent (TA), Aglets and Mobile Agent Framework (MAF) [18].

Some of the underlying platforms support collaboration and communication but still lacks special support for parallelism. Java Agent DEvelopment framework (JADE) was chosen as a software framework to facilitate the development of the mobile agents-based parallel applications in compliance with the Foundation for Intelligent Physical Agents (FIPA) specifications for interoperable intelligent Multi- Agents System (MAS).

4. WINDOWS PASSWORD RECOVERY TECHNIQUES

Password recovery is the process of hashing the guess words or phrases using a specific algorithm and then it is compared the hashing result with the hash value that is stored in a computer system. If the two passwords are match, then the guess is correct. Otherwise, the guess is incorrect. Specially, there are three basic methods of password recovery: ***dictionary method***, ***rainbow tables method*** and ***brute-force method*** [19].

4.1 Password on Windows 7 Operating System

Windows operating system stores passwords of its users in a hash format by using a *hash algorithm*, which is also called a *one-way hash function*. Hash algorithm digests a certain input of a variable length and it returns a deterministic fixed size hash value. This function cannot be reserved, since the output hash value cannot be used to determine the original password [20][21][22].

Starting with Windows NT until new versions of Windows operating systems, the user account and its password are stored in the Security Account Manager (SAM) database file [23]. The SAM file is located in the C:\Windows\System32\config folder and also stored in the registry at HKEY_LOCAL_MACHINE\SAM. SAM file cannot accessible when the operating system is booted because the operating system holds a lock on the SAM file [24].

Moreover, NT LAN Manager Hash (NTLM hash) was released with the Windows NT operating system, (hence the name, NT hash), then it was used with the Windows 2000, XP, Vista and 7. NTLM hash is used as a mechanism to encrypt the password of the new versions of Windows operating systems and it supports passwords up to 128 bits in length [23]. NTLM hash uses the Message Digest 4 algorithm (MD4) to create the hash value of the password after converting the password with the American Standard Code for Information Interchange (ASCII) representation of the password to a Unicode with a Little Endian format (UTF-16LE) representation [21].

Generally, the password hash value can be extracted from the SAM database file by using software tools [24]. This research uses the fgdump software tool, which is a command-line tool using it to extract the hash values .

5. THE PROPOSED SYSTEM

In this work, mobile agent technology was adopted to introduce the proposed parallel system over a grid of personal computers where the tasks can be decomposed and encapsulated to a family of individual and collaborative mobile agents. The proposed system addresses the password recovery problem of Windows 7 operating system. More specifically, the brute-force technique, which is the most efficient method to recover the password, was used to perform an exhaustive search on the hashes for a chosen character set and a range of the password length. Theoretically, a complete brute-force technique that includes all letters, numbers, special and printable characters can guarantee a 100% success rate but it is very time consuming because the sample space of every permutation and combination is extremely large. Thus, bigger passwords can take many years to recover using this password recovery technique. Due to these reasons, this research presents a proposed system which introduces brute-force-based three efficient methods of password recovery for Window 7operating system as a proof of parallel processing boosted by the inherited benefits of using Java mobile agent technique powered by the Java multithreading capability where JADE provides a single thread per agent.

5.1 System's Architecture

The proposed system's architecture was built based on the Master-Slaves model, which means that the physical infrastructure (system's platform) of this work is organized in a master side and slaves side. Where it consists of a controller computer, which is represented the master node, and a set of peripheral computers, which are represented the slave nodes. The whole system's computers are distributed on a LAN (Local Area Network) and they are connected as a star topology as shown in the Figure 1. The choice of the star topology to build the system architecture is due to the absence of any agent to transfer among the parallel computers as all agents are transferred through the controller computer. Therefore, this system is a fully parallel and there is no dependency between the agents of the system. The used type of parallelism is a fine grain because each mobile agent in the proposed system uses the iterative password generator method to recover the password where the number of instructions in each mobile agent does not exceeded 500 instructions. Based on the Flynn's Taxonomy of Computer Architecture, the Multiple Instruction multiple Data (MIMD) multicomputers architecture was used to built the Master-Slaves model to improve the development of the proposed system.

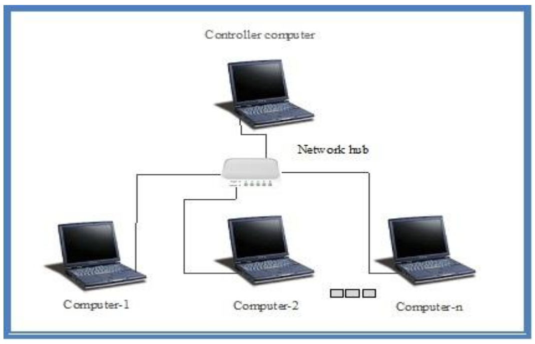


Fig 1: Architecture of the proposed system

5.2 System's Platform

The proposed system's platform is composed by a team of collaborative agents that have been organized to provide a series of services required to manage the distributed platform of the system. The collaborative agents of the proposed system are distributed between the master side and the slaves side as shown in the Figure 2.

5.2.1 Master Side

- *Directory Facilitator Agent (DF***):** DF agent, who is located on the master side, offers a yellow pages services to other agents in the proposed system's platform.
- *Agent Management System Agent (AMS):* AMS agent, which is also located on the master side, provides a white pages services and represents the authority of the proposed system's platform.
- *Mobile Agent:* Mobile agent is a generic component that hosts a particular parallel task. Moreover, the mobile agent is dynamically created by the controller agent. The proposed system can split up the workload by cloning the mobile agent. The cloned agents implement the mobility capability in order to travel from the master side to the slave nodes where the available resources and environment are suitable to execute their parallel tasks.
- *Controller Agent:* The controller agent, which is located on the master side, has the following main tasks:
- The controller agent is responsible of managing and monitoring the overall proposed system.
- The controller agent is responsible of sending REQUEST messages to all agents on the system's platform, those are registered their *Create-Container* service with the DF agent, to create a number of empty agent containers to host the delivered cloned agents.
- The controller agent can send a REQUEST message with the *query-platform-locations* action to the AMS agent to get the all available containers on the system's platform including the main container and the remote containers which are distributed among the grid computers. Moreover, the controller agent can present the retrieved containers on the User Interface (UI) for the end-user.
- The controller agent is responsible of initiating and cloning the mobile agent as well as is responsible of delegating the parallel tasks to the cloned agents. The cloned agents should be deployed by the controller agent among the agencies, which are distributed on the network, to execute their parallel tasks and to obtain the final result that will be sent back to the controller agent by a CONFIRM message.
- The controller agent should present the final result on the UI to the end-user and it should send a REQUEST message with a *kill container* action to the AMS agent to kill of the all containers which are distributed on the salves side.
- ▪ Also, the controller agent is responsible to send another REQUEST Message to the AMS agent but with the *shutdown-platform* action to end the platform of the proposed system.

5.2.2 Slaves Side

- *Stationary Agents:* During the proposed system lifecycle, a set of stationary agents should be dynamically setup. Concretely, each stationary agent should be registered with the DF agent to publicize its *Create-Container* service to the controller agent. During the system runtime, the controller agent may send REQUEST messages to the all stationary agents to create a number of empty agent containers in order to allow the future hosting of the cloned agents. After that, each stationary agent should send an INFORM message to the controller agent to inform it of creating the requested number of empty containers.

6. SYSTEMS'S METHODS

This research introduced three proposed efficient methods based on the mobile agents technology, which are supported with different degrees of parallelism and they are powered with the Java multithreading capability, for higher-performance Windows 7 password recovery.

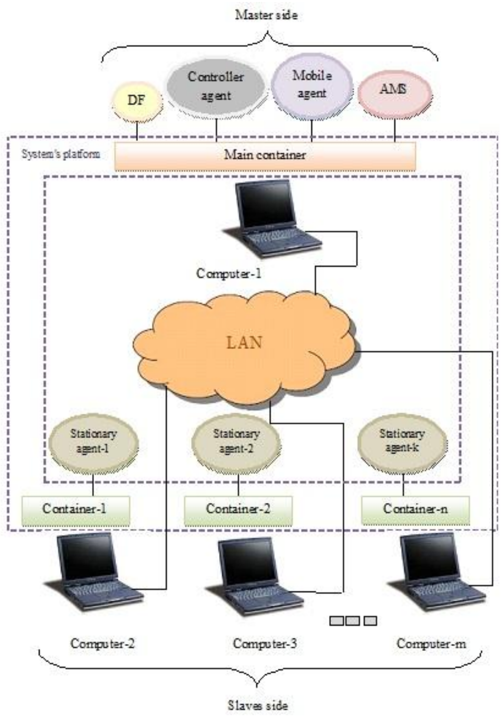


Fig 2: Platform of the proposed system

6.1 First Method Algorithm

Topologically, this method requires a single controller computer and three peripheral computers in which tasks will be executed in parallel. Dynamically, a single container should be setup on each peripheral computer and it should be hosted a predefined stationary agent. The algorithm of this method can take any range of the password length as long as the range achieves the condition of the Equation 1:

The input range = ((Maximum length – Minimum length) + 1) / 3 (1)

Furthermore, this algorithm splits up all possible combinations of each possible password length into two segments where each segment will be encapsulated on a single thread per cloned agent. The total number of the possible combinations of each thread can be calculated according to the Equation 2.

Total combinations of each thread = (character set ^ password length) / 2 (2)

The controller agent is responsible of dispatching the whole constructed cloned agents to execute the password recovery process on the parallel slave nodes simultaneously. In addition, to increase the efficiency of this algorithm, each slave node should run an equal number of threads (cloned agents) where the total number of cloned agents, which are dispatched to the parallel slave nodes, is calculated depending on the Equation 3.

Total number of the cloned agents = ((Maximum length – Minimum length) + 1) * 2 (3)

Figure 3 explains the algorithm of the first proposed method procedure.

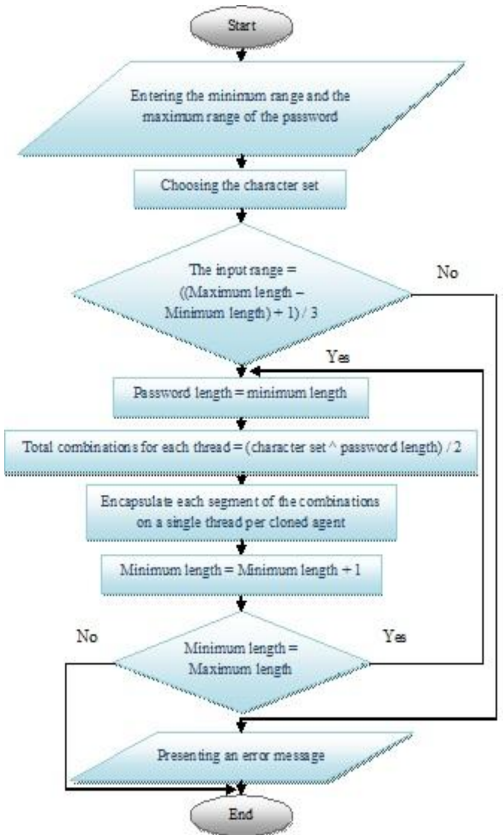

Start
Entering the minimum range and the maximum range of the password
Choosing the character set
The input range = ((Maximum length – Minimum length) + 1) / 3
No
Yes
Password length = minimum length
Total combinations for each thread = (character set ^ password length) / 2
Encapsulate each segment of the combinations on a single thread per cloned agent
Minimum length = Minimum length + 1
No
Yes
Minimum length = Maximum length
Presenting an error message
End

Fig 3: Flowchart of the first method algorithm

6.2 Second Method Algorithm

Similarly, the second method also requires a single controller computer and three peripheral computers in which tasks will be executed in parallel. Also, a single container should be setup dynamically on each peripheral computer and it should be hosted a predefined stationary agent. In addition, the algorithm of this method accepts any range of the password length that is satisfied the condition of Equation 1. The principle of this algorithm is to encapsulate the all possible combinations of each possible length in a single thread per cloned agent according to the Equation 4.

Total combinations for each thread = (character set ^ password length) (4)

The whole constructed cloned agents should be dispatched by the controller agent to execute the password recovery process on the parallel slave nodes simultaneously. Moreover, each slave node should run an equal number of threads (cloned agents) to strengthen this algorithm. The total number of cloned agents, which are distributed to the parallel slave nodes, is calculated depending on the Equation 5.

Total number of the cloned agents = ((Maximum length – Minimum length) + 1) (5)

Figure 4 explains the main functions of the algorithm of the second proposed method.

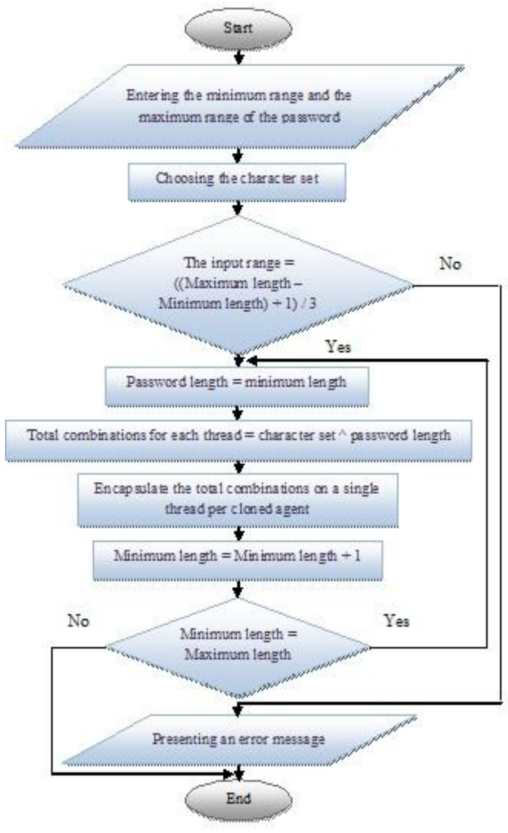

Start
Entering the minimum range and the maximum range of the password
Choosing the character set
The input range = ((Maximum length – Minimum length) + 1) / 3
No
Yes
Password length = minimum length
Total combinations for each thread = character set ^ password length
Encapsulate the total combinations on a single thread per cloned agent
Minimum length = Minimum length + 1
No
Yes
Minimum length = Maximum length
Presenting an error message
End

Fig 4: Flowchart of the second method algorithm

6.3 Third Method Algorithm

The physical structure of the third proposed method requires also a single controller computer and only two peripheral computers in which tasks will be executed in parallel. In addition, the algorithm of this method accepts only a single possible password length that is predefined by the end-user. Besides that, the total number of all possible combinations of predefined possible length, which can be calculated according to the Equation 2, is split up into two segments where each segment will be encapsulated on a single thread per cloned agent.

According to the this algorithm, the controller agent dispatches only two cloned agent to execute the password recovery process on two parallel slave nodes simultaneously which are distributed on the system's platform. This means that there is no multithreading concept applied on each of the distributed slave node. Figure 5 illustrates the flowchart of the algorithm of the third proposed method.

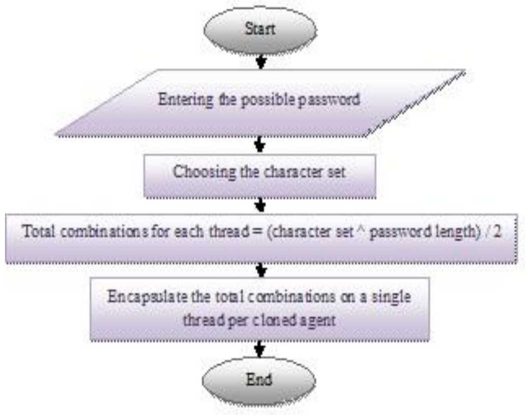

Start
Entering the possible password
Choosing the character set
Total combinations for each thread = (character set ^ password length) / 2
Encapsulate the total combinations on a single thread per cloned agent
End

Figure 5: Flowchart of the third method algorithm

7. SYSTEM’S MAIN PHASES

The proposed system was designed with four sequential phases to be followed by the three proposed methods to recover the password of Windows 7 operating system.

7.1 First Phase: Registering and Retrieving of Stationary Agents

This phase of implementation consists of two processes: the registering process and the retrieving process of the stationary agents. In the registering process, each stationary agent, which is distributed on the remote container, should enroll *a Create-Container* service on the yellow page of the DF agent during its setup process. On the other hand, in the retrieving process the controller agent should retrieve all stationary agents, who are publicized their *Create-Container* service during the registering process, from the DF agent. Moreover, the controller agent can present the retrieving stationary agents on the user interface.

7.2 Second Phase: Creating and Retrieving of System's Containers

This phase of implementation has also two processes: the creating process and the retrieving process of the system's containers. In the creating process, the controller agent should send *REQUEST* messages to the all stationary agents, which were publicized their *Create-Container* service in the previous phase, to create empty agent containers in order to allow future hosting of the delivered cloned agents belong with the containers which are hosted the stationary agents. After that, each stationary agent should send an *INFORM* message to the controller agent to notify it that the process of creating the required number of containers was successfully completed. If the number of INFORM messages equals the number of the stationary agents, the controller agent will send the cloned agents to the hosting containers, otherwise an error message will be shown to the end-user.

In spite of the ability of each empty container to host a single cloned agent, each stationary agent container can also host a single cloned agent. This means that the creating process of this phase is mandatory in the first proposed method of password recovery where the number of the hosting containers and empty containers can be calculated according to the Equation 6.

Total number of empty containers = (((Maximum length – Minimum Length) + 1) * 2) – Number of stationary agent containers

Total number of hosting containers = ((Maximum length – Minimum Length)+1) * 2 (6)

On the other hand, the implementation of the creating process is electively in the second proposed method where the number of the hosting containers and empty containers can be calculated depending to the Equation 7.

Total number of empty containers = ((Maximum length – Minimum Length)+1) – Number of stationary agent containers

Total number of hosting containers = (Maximum length – Minimum Length)+1 (7)

Moreover, the creating process of this phase is not required in the third proposed method because the two cloned agents of this method can be dispatched to the two stationary agent containers which are distributed on the two slave nodes. The hosting containers in this method depend on the Equation 8.

Total number of hosting containers = Number of stationary agent containers (8)

During the retrieving process, the overall system's containers can be retrieved by the controller agent by sending a REQUEST message, which includes the *Query-Platform-Locations* action, to the AMS agent. After that, the AMS agent should send an INFORM message to notify the controller agent about retrieving of the system's containers. Furthermore, the controller agent can present the all retrieved containers on the user interface. Figure 6, illustrates the message passing strategy of the second phase.

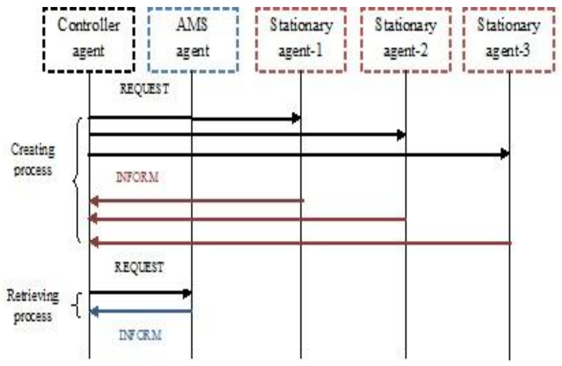


Fig 6: Strategy of message passing of the second phase

7.3 Third Phase: Creating of the Mobile Agent

In this phase, the mobile agent should be created which is named *Recovery-Agent*. As well as, this agent should be hosted on the previously created container which is named *Home* container. In addition, the *Recovery-Agent* must be joined with the *MobileAgent* class which is programmed to be able to contain the parallel process that will be implemented by the cloned agents. The controller agent is responsible of cloning the mobile agent according to the predefined range of the password length and the selected proposed method.

7.4 The Implementation Phase

The implementation phase consists of the following different processes:

- *Choosing one of the three proposed method,* as mentioned previously the proposed system has three different methods for Windows 7 operating system password recovery.
- *Specifying the password length,* the proposed system supports the first and the second proposed method to accept any rang of the password length which is satisfy the Equation 1, while it supports the third method to accept only one possible length.
- *Import the NTLM hash value,* this value is obtained by the fgdump software tool. Where it should be stored in a text file in a specific partition of computer's hard memory.
- *Selecting one of the allowable character set,* the proposed system provides four options of character sets; (0-9), (a-z), (A-Z) and (0-9, a-z and A-Z).
- *Cloning and migrating the cloned Recovery-Agents,* according to the number of parallel tasks, the mobile agent is cloned into many cloned Recovery-Agents. After that, the cloned agents will migrate to remote containers, which were retrieved by the AMS to implement the parallel process of password recovery for Windows 7 operating system. Generally, the time for the migration process depends on the bandwidth, traffic state of the network and the size of the cloned agent.
- *obtaining the final result*, this process consists of two cases:

✓ ***Finding the goal password,*** once the cloned agent finds the goal password, it should send an *INFORM* message, which contains the goal password and the elapsed time of the searching process, to the controller agent that resides on the *main container*. Where the elapsed time can be calculated in milliseconds using the Equation 9.

Elapsed Time = Ending time of the recovery process in milliseconds- Starting time of the recovery process in milliseconds (9)

On the other hand, if the cloned agent could not find the goal password and the controller agent did not receive an INFORM message from any other cloned agent then the controller agent should receive a CANCEL message from this cloned agent. In turn, the controller agent is responsible of killing of all hosting containers by sending a REQUEST message to the AMS with the *kill-container* action and presenting the goal password and the elapsed time on the user interface. After that, the AMS agent will send an INFORM message to notify the controller agent of killing the hosting containers. Figure 7 illustrates the strategy of the message passing in this case.

✓ ***Not finding the goal password***, if the goal **password is not found, each cloned agent should** send a CANCEL message to the controller agent. In this case, the controller agent is responsible only of killing of all hosting containers also by sending a REQUEST message to the AMS with the *kill-container* action. After that, the AMS agent will send an INFORM message to notify the controller agent of killing of all hosting containers. Figure 8 illustrates the strategy of the message passing in this case.

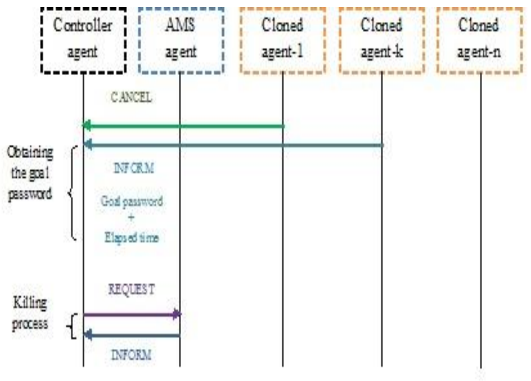


Fig 7: Strategy of the message passing in finding the goal password case

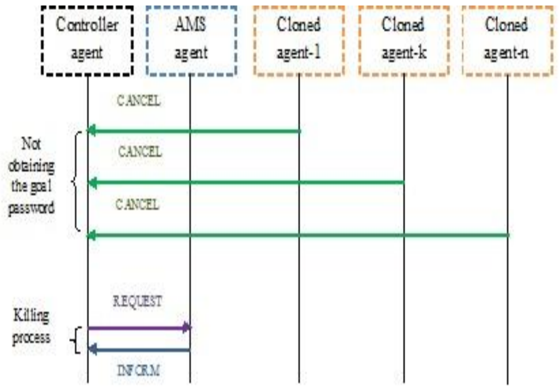


Fig 8: Strategy of the message passing in not finding the goal password case

8. PASSWORD RECOVERY PROCESS

The overall process of password recovery is distributed among the cloned agents which will be dispatched among the distributed computers on the system's platform. Where each cloned agent is responsible to execute its task independently. The internal implementation of password recovery illustrates on the Figure 9 that describes the Flowchart of the password recovery process.

9. PROPOSED SYSTEM IMPLEMENTATION

Java programming language combined with the JADE 4.3.1 framework, which is the most widespread agent-oriented middleware, was used to develop the proposed system. Besides that, the intra-platform mobility service that is provided by the JADE framework was used. Moreover, NetBeans Integrated Development Environment (IDE) 7.4 was used as a Java programming tool to build the proposed system. The main user interface built on the master side to help end-users with using the proposed system as shown in the Figure 10.

On the other hand, the slaves side programs run in the background so they do not have a user interface.

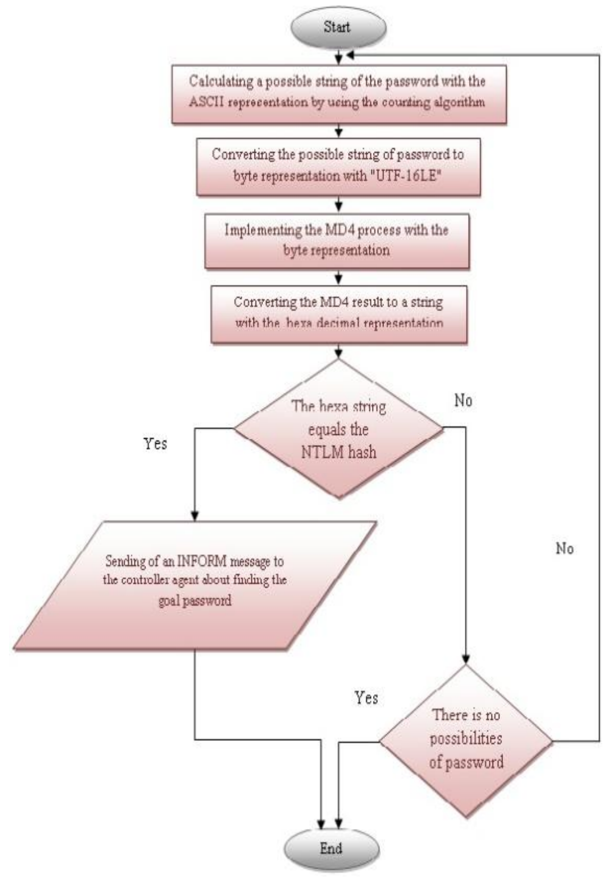
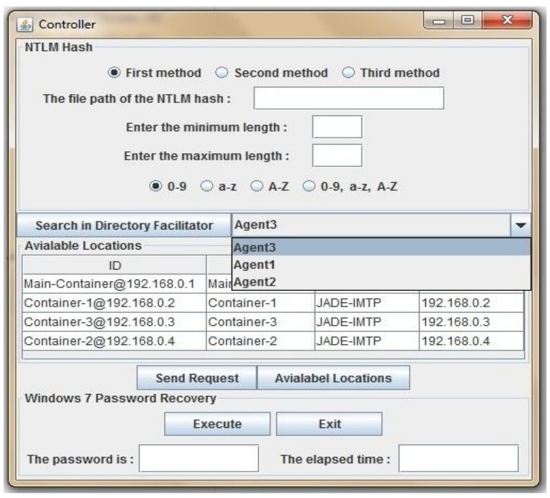

Start
Calculating a possible string of the password with the ASCII representation by using the counting algorithm
Converting the possible string of password to byte representation with "UTF-16LE"
Implementing the MD4 process with the byte representation
Converting the MD4 result to a string with the hexa decimal representation
The hexa string equals the NTLM hash
Yes
No
Sending of an INFORM message to the controller agent about finding the goal password
No
Yes
There is no possibilities of password
End

Fig 9: Flowchart of the password recovery process



ID			
Main-Container@192.168.0.1	Mai		
Container-1@192.168.0.2	Container-1	JADE-IMTP	192.168.0.2
Container-3@192.168.0.3	Container-3	JADE-IMTP	192.168.0.3
Container-2@192.168.0.4	Container-2	JADE-IMTP	192.168.0.4

Fig 10: The user interface of the proposed system

9.1 Experiments and Results

Windows 7 password recovery problem has been successfully implemented using the three different methods that are provided different degrees of parallelism. In the all experiments, the passwords were chosen from the beginning and the middle of the passwords' possibilities using different range of several passwords with the all cases of the character set. Based on the conducted experiments, the three proposed methods have been tested successfully to recover each candidate password with a variety of time as shown in the Table 1. This variation in the time is due to the mechanism of splitting up the task of the password recovery in each proposed method.

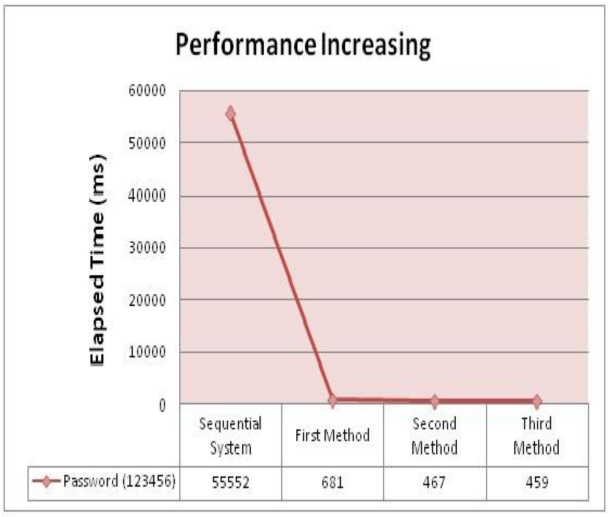
Table 1: The achieved results from the three proposed methods

Password	First Method (Time in milliseconds)	Second Method (Time in milliseconds)	Third Method (Time in milliseconds)
501026	6400	392630	2200
0000000	16411	4696	2140
naae	22121	442193	1919
aaaaabex	7254	5007	2075
NAAK	23150	468321	2060
AAAAABEV	7269	5117	2380
v000	5866	2569558	1981
001jE	27612	13167	3953

According to the speedup measurement, that based on the Amdahl's Law as shown in the Equation 10, and the candidate password 123456 with the possible range 4-6, the first, second and third proposed methods are 82x, 119x, 121x faster than the sequential system respectively as illustrated in the Figure 11.

Speedup = Sequential (Time) / Parallel (Time)

In addition, the achieved results found that the proposed system proved its efficiency and the ability to find any password against the OphCrack software as shown in the Figure 12, that based on the rainbow tables and multithreading techniques to crack passwords. This is due to the probabilistic nature of the rainbow tables technique, while the brute-force technique the brute-force technique tries every possible permutation and combination of characters using the same hash algorithm as the original password utile the correct password is found. Moreover, adopting the parallel processing and the mobile agents technology to recover passwords in the three proposed systems.



	Sequential System	First Method	Second Method	Third Method
Password (123456)	55552	681	467	459

Fig 11: The speedup measurement over the proposed system

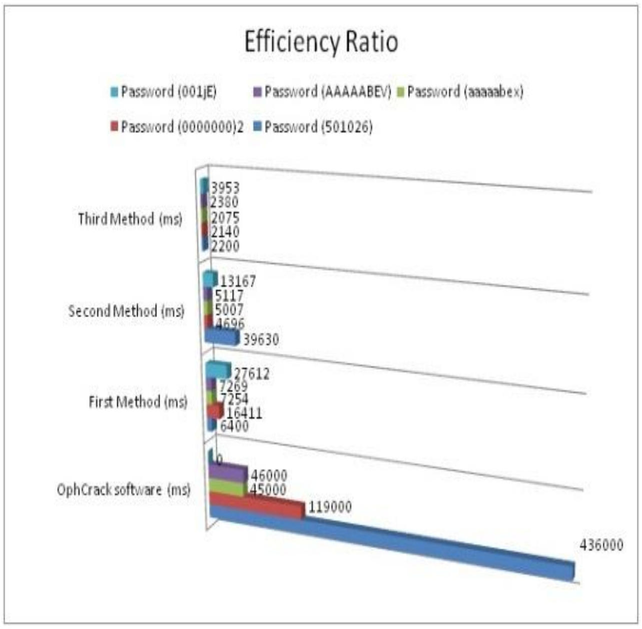


Fig 12: Comparison between the three proposed methods and the OphCrack software

10. CONCLUSIONS

The main focus of this paper is to conjunct the mobile agents technology and the parallel processing to provide several opportunities and promising approach to propose a new system for developing Window 7 password recovery problem. According to the achieved results, the proposed system proved its efficiency and it supports a higher-performance parallel processing to tackle very large-scale systems as Windows 7 password recovery problem. The results clearly show that the first developed method of the proposed system is active than the second method to recover the password when the password is in the second segment of the possible combinations of each possible password length. In contract, the second method is more efficient to recover the password when the password is in the first segment of possible combinations of each possible password length. Besides that, the practical results have also shown that the third method is the fastest one to recover the password since it takes only a single possible password length that is predefined by the end-user. The third method is active only when the user surely knows the length of the forgetting password. Although the proposed system achieve a promising approach, that adopts the mobile agents technology for parallel processing, to address Windows7 password recovery problem, further development could be added in future researches like, the proposed system can be decentralized to provide platform-to-platform mobility for JADE agents using the JADE add-on IPMS. This will offer server replication and a higher degree of robustness. The proposed system can be developed to be deployed on smart phone devices by means of the JADE add-on Lightweight Extensible Agent Platform (LEAP). Load balancing is a worth point to take into consideration in the future works.

11. REFERENCES

[1] Hsien-Cheng C., Hung-Chang L., Hwan-Jeu Yu, Fei-Pei Lai, Kuo-Hsuan Huang and Chih-Wen Hsueh, (February 2013), "Password cracking based on learned patterns from disclosed password", International Journal of Innovative Computing, Information and Control ICIC, No. 2,Volume: 9, Pages:821- 839.

[2] Wesley M. E. and Mark A., (Feb. 2000), "Advantages of Parallel Processing and the Effects of Communications Time", NASA Glenn Research Center, Cleveland, OH United States, Report Number CR-209455, Provider: cite seer.

[3] Charles M. W., (2010), "Using probabilistic techniques to aid in password cracking attacks", Ph.D. thesis, The Florida State University College of Arts and Sciences.

[4] Martijn S., (October 2010), "Cracking Password Hashing Schemes using Graphics Processing Units", Ms.c. Thesis Proposal, Kerckhoffs Institute.

[5] Nick V. F. and Haile Sh., (2011), "Technology Corner: Brute Force Password Generation-Basic Iterative and Recursive Algorithms", Journal of Digital Forensics, Security and Law (JDFSL), Volume: 6(3),Pages: 79-86.

[6] David S., Ángel R. and Antonio M., (2007), "Parallel execution of complex tasks using a distributed, robust and flexible agent-based platform", In proceedings of the third workshop on development of multi-agent (DESMA07), II Spanish Congress of Information (CEDI 07), Zaragoza , ISBN 978-84-9732-613-1, Pages: 11- 18.

[7] Johnny B., (2007), "Parallel Password Cracker: A Feasibility Study of Using Linux Clustering Technique in Computer Forensics", Second International Workshop on Digital Forensics and Incident Analysis (WDFIA), IEEE, Pages 75-82.

[8] Kostas Th., Charalampos M. and Ioannis P., (2009), "High-End Reconfigurable Systems for fast Windows' Password Cracking", 17th IEEE Symposium on Field Programmable Custom Computing Machines, Pages: 287-290.

[9] Martijn S., (February 2011), "GPU-based Password Cracking: On the Security of Password Hashing Schemes regarding Advances in Graphics Processing Units", M.Sc. thesis, Radboud University Nijmegen.

[10] Chrysanthou Y., (2013), "Modern Password Cracking: A hands-on approach to creating an optimised and versatile attack", M.Sc. thesis in Information Security at Royal Holloway, University of London.

[11] Norm M., (July 2012), "Programming on Parallel Machines: GPU, Multicore, Clusters and More", University of California, Pages: 410.

[12] Ronald L. K. and Russell D. V., (Jan. 2008), "The CEH Preparing Guide: The Comprehensive Guide to Certified Ethical Hacking", John Wiley & Sons, Pages: 738.

[13] Yan W., (2002-01-08), "Dispatching Multiple Mobile Agents in Parallel for Visiting E-shops", The Third International Conference on Mobile Data Management-MDM, IEEE Computer Society Washinqton, DC, USA, ISBN: 0-7695-1500-2, Page: 61.

[14] Rahul J., (2005), "Mobile agents for e-commerce", KR School of Information Technology, Indian Institute of Technology, Bombay, Roll No.: 99329011.

[15] Parineeth M. R., (2002-07-01), "Mobile agents, Intelligent Assistants on the Internet", Resonance journal, Springer India, Issue 7, Volume: 7, Pages: 35-43.

[16] Uhrmacher A. M. and Gugler K., (2000), "Distributed, Parallel Simulation of Multiple, Deliberative Agents",

Fourteenth Workshop on Parallel and Distributed Simulation, IEEE, Bologne, Pages: 101-108.

[17] Cheng-Zhong X. and Brian W., (Aug. 2000), "A Mobile Agent Based Push Methodology for Global Parallel Computing", In wiley journal: Concurrency: Practice and Experience, Issue: 8, Volume: 12, Pages: 705-726.

[18] Abdelkader O., (November 2009), "Mobile Agents-Based Applications: a Survey", International Journal of Computer Science and Network Security (IJCSNS), Issue: 11, Volume: 9, Pages: 331-339.

[19] Maryam B. and Martin G., (2008), "Distributed Password Recovery", Technical report, Ryerson University.

[20] Torsten I., Frank K., Michael W. and Tilmann K., (2000), "Migration of Mobile Agents in Java: Problems, Classification and Solutions", Proceeding of International Symposium on Multi-Agent and Mobile Agent in Virtual organization and E-Commerce, ICSC, Pages: 362-369.

[21] Sandip V., Bhavesh L. and Shreyansh B., (2011), "A Survey on Agent Communication Languages", International Conference on Innovation, Management and Service, Volume: 14, Page: 237.

[22] Aleksandar K. and Jochem v. K., (May 2011), "Distributed GPU Password Cracking", Research project, University of Van Amsterdam.

[23] Russell E. G., (2008), "High performance password cracking by implementing rainbow tables on nVidia graphics cards (IseCrack)", M.Sc. thesis, Iowa State University, Ames, Iowa.

[24] Jørgen W. B. and Rune W. N., (June 2009), "Procedures and Tools to Reset or Recover the Administrator Password on Popular Operating Systems", M.Sc. thesis, Norwegian University of Science and Technology, Department of Telematics.

12. AUTHOR PROFILE

Prof. Dhuha Basheer Abdullah : head of Computer Science Department, Collage of Computer and Mathematics, University of Mosul. She received her PhD degree in computer science in 2004 in the specialty of computer architecture and operating system. She supervised many Master degree students in operating system, computer architecture, dataflow machines, mobile computing, real time, mobile cloud computing, and distributed database. She supervised three PhD students in FPGA field, distributed real time systems, and Linux clustering. Currently she supervises PhD student in cloud computing field. She leads and teaches modules at both BSc, MSc, and PhD levels in computer science. Also, she teaches many subjects for PhD and master students.

Manar Seyala is a master student in Computer Science Department, College of Computer and Mathematics, University of Mosul. She is interested with network security, operating system and distributed databases subjects.