An Analysis and Comparison of Clustered Password Crackers

Christian Frichot
School of Computer and Information Science, Edith Cowan University,
Bradford Street, Mt Lawley, Western Australia 6050
cfrichot@student.ecu.edu.au

Abstract

Password policies alone do not stand a chance of securing computer systems which rely on the use of secret-based, password authentication methods. The enforcement of "strong" passwords via pro-active password strengthening utilities and password cracking tools should be employed by system administrators to reduce the number of "weak" passwords in a computer system. With the availability of low-cost computer and networking hardware, clustered solutions for large computational tasks, such as password cracking, are no longer restricted to larger organisations. John the Ripper and Cisilia are two open-source password cracking programs which have the ability to run in a clustered environment. This paper intends to make a comparison of John the Ripper to Cisilia in a clustered environment utilising the OpenMosix and Beowulf styles of parallel computation. Unfortunately, due to problems with Cisilia an in-depth comparative analysis could not be performed, but the analysis of the John the Ripper results does highlight some issues in regards to clustered password cracking.

KEYWORDS

Beowulf, Cluster, Password Cracking, John the Ripper, Cisilia

INTRODUCTION

Password cracking software can be used by system administrators to discover weak passwords on computer systems. The process of password cracking usually relies on different methods of attack; some of the more common attacks include the dictionary and brute-force attack. The dictionary attack relies on the use of a wordlist, which is used to guess passwords until a successful match is found, the brute-force attack on the other hand will iterate through every single password combination looking for matches. The task of cracking passwords is computationally intensive. To improve upon the time taken to crack passwords, computer clusters have been utilised to share the load amongst several interconnected computers, when the work is split between multiple computers the time taken to crack passwords can be shortened.

Two such password cracking tools which are able to run on a cluster of computers are John the Ripper and Cisilia. It is the intention of this paper to make a comparison between these two tools, and the efficiency of each at cracking Samba passwords.

BACKGROUND

Passwords

There are three main types of computer and network authentication: Things that you know, such as passwords or PINs; things that you are, biometrics; and things that you have, such as a USB-dongle or pass-card (Liu & Silverman, 2000). Of these three types, authentication which relies on something you know is the predominant method. Previous research into password usage has indicated that they are predominant because of their acceptability, ease of use, and their simplicity to system designers (Irakleous, Furnell, Dowland, & Papadaki, 2002; Wood, 1996).

To secure passwords that are stored on computer systems, they are encrypted with a one-way function (Pfleeger, 1997, p. 259). These functions are also known as a hashing function, or a one-way hashing function. The principle behind a one-way function is that it accepts an arbitrary length input, and outputs a digest or hash which is irreversible, based upon an algorithmic manipulation of the input (Schneier, 1996, p. 431). Some of the more common algorithms utilised to encrypt passwords are the MD5 algorithm (Rivest, 1992), and algorithms based off the DES algorithm (Schneier, 1996, p. 265).

Reusable passwords are susceptible to many different forms of attacks. A categorisation of different password attack methods have been summarised by Neumann and Pfleeger (1994; 1997) and included in table 1.

Exhaustive Attacks	Attempting passwords by random trial and error
Educated guessing of passwords	The use of default passwords, dictionary based words, or passwords which have a logical association with a particular user
Deriving passwords (Dictionary)	Pre-emptive dictionary attacks
Capturing unencrypted passwords	Capturing passwords by Trojan horses or other snooping techniques
Bypassing authentication	Using exploits in different parts of the system to access higher-level constructs
Brute force attacks	Similar to exhaustive attacks, but instead of using random trial and error the cracking mode tries all possible passwords.

Table 1. Password Attack Methods

Some of these attacks rely on the attacker having access to the file which stores the passwords. On modern systems this file is usually only accessible by the system administrator, if at all (Hare & Siyan, 1996; Schneier, 2000).

Password authentication is often one of the only lines of defence for computer systems (Wood, 1996). In many cases after a user account has been compromised by an attacker, they can do anything that the compromised-account is authorised to do, including discovering ways to get more privileged access to the system (Klein, 1990). To make this situation even worse, previous research into user-chosen passwords indicates that the majority of user-chosen passwords are susceptible to dictionary-attacks (Klein, 1990; Spafford, 1992).

An important aspect of the security of passwords is a password's strength. In regards to users, a strong password is one which is remembered easily, but not easily guessable. Mathematically a password's strength may be defined by the length of the password and the number of possibilities per character in the password (Beverstock, 2003). For example, a 2 character password using only single-case alphabetic characters has 676 possible passwords.

$$26^2 = 676$$

An 8 character password using a mixture of upper and lower case characters has 53.5 trillion possibilities.

$$52^8 = 53.4597 \times 10^{12}$$

A 14 character password using a mixture of upper and lower case characters, numbers, symbols and punctuation has approximately 5000 septillion possibilities.

$$95^{14} = 48.7675 \times 10^{26}$$

Breaking passwords

Password cracking software usually implements either the dictionary or brute-force attack method, in some cases hybrid methods may be utilised. System administrators might use these tools to legitimately break password files to discover weak passwords so they can notify the user. Common implementations of dictionary-attacks will take wordlists as input, permute the words through filters, encrypt the permuted word and compare it to a password hash. Some example permutations are included in table 2.

Invert the word	"Password" becomes "drowssaP"
Try all different case combinations of alphabetic characters	"secret" could become "SeCReT" or "sECREt", for example.
Convert the word to plural	"Dog" becomes "Dogs"
Swap alphabetic characters to numeric characters	"passcode" could become "p455c0d3"
Word pairs	"Dog" and "Cat" could become "DogCat"

Table 2. Word Permutation Rules

Some of the more popular tools include Crack, John the Ripper, L0phtcrack, Distributed Keyboard Brute-Force (DKBF) and Cisilia. Crack is one of the original UNIX, dictionary-attack based password crackers written by Alec Muffet (1991) which utilised the C crypt function to create password hashes. John the Ripper (JtR), by Solar Designer (2002), is an open-source, multiple platform password cracker that can crack multiple password types, and can perform multiple different types of attacks. L0phtcrack by @Stake software (2002) is a Microsoft Windows password cracker, similar to JtR it utilises multiple methods of attack. DKBF is a distributed version of L0phtcrack, but can only perform brute-force attacks. Cisilia is a multi-process password cracker, designed to break Microsoft Windows based passwords using brute-force attacks on load-balancing clusters (C.I.S.I.ar, 2003).

Compute Clusters

A cluster of computers is a collection of low-cost computers, connected together within a Local Area Network (LAN). The goal of a cluster is to combine the computational power of individual nodes to create a logical super-computer at a much cheaper cost (Dinquel, 2000). There are many different configurations and styles of computer clusters, two such styles are the Beowulf class of clusters, and load-balancing clusters.

Beowulf clusters were originally developed by NASA in 1994 to create a system which could perform as well as a super-computer at a much lower cost (Ridge, Becker, Merkey, & Sterling, 1997). To reach this goal, Beowulf clusters traditionally are characterised to:

- Use easily available, off the shelf components
- Be scalable
- Use a freely available software base
- Use dedicated nodes

Beowulf clusters rely on software being specifically designed and implemented to work in parallel. The Message Passing Interface (MPI) specification outlines C and FORTRAN function definitions allowing programs to pass messages to each other. Another message passing standard is the Parallel Virtual Machine (PVM) specification.

A load-balancing cluster, such as an OpenMosix cluster, relies on algorithms to automatically migrate active processes, or jobs, to other nodes in the cluster. OpenMosix specifically is a “Linux kernel extension for single-system image clustering” (Bar, 2004). A computer which starts a task in an OpenMosix cluster does not have to specify how to share the load; the cluster will automatically transfer the work to different nodes if it believes the first computer to be too busy. One of the benefits of an OpenMosix cluster compared to a Beowulf cluster is that software does not have to be specifically written to work in parallel on an OpenMosix cluster.

AIM OF THE PAPER

The research aimed to perform a quantified comparison of two different clustered password cracking tools, John the Ripper-MPI and Cisilia. To make a comparison, experiments were performed using the different utilities on a single set of input data. This input data was a representation of passwords, ranging from weak passwords to strong passwords. Unfortunately, in early tests of Cisilia on an OpenMosix cluster problems started to surface which prohibited a comparative analysis of the two different password crackers.

RESEARCH METHOD

Hardware Configuration

Two separate hardware configurations were used for the experiments. Both of these configurations were trying to bring the effect of different hardware to a minimum. The basis of the setup relied on x86 Intel based processors, inter-connected with 100Mbps networking hardware.

The Beowulf configuration used for testing John the Ripper-MPI was comprised of 13 nodes. One of the 13 nodes was used as the head-node and did not perform any of the work, but was required in the configuration as all the program and data files were hosted on this machine.

Processor Make	Intel Pentium III
Processor Clock Speed	866MHz

Memory	256MB of SDRAM
Hard Drive	20GB IDE
Network Interface Card	100Mbps

Table 3. Beowulf Node Configuration

The head-node was configured with an extra 128MB of memory, providing it with 384MB of SDRAM, and also a faster SCSI based disk drive. The nodes were inter-connected with a Baystack 10/100 24 port managed switch. Combining the processor speed and memory of the 12 compute-nodes resulted in a Beowulf cluster which had approximately 10.4GHz clock speed, and 3GB of memory.

The OpenMosix configuration used for testing Cisilia was comprised of 14 nodes, a head-node, 6 compute-nodes with hard drives and 7 compute-nodes without hard drives. Even though there was a head-node, computational work was still performed on this machine as well.

Processor Make	Intel Pentium III
Processor Clock Speed	733MHz
Memory	256MB of SDRAM
Hard Drive	20GB IDE
Network Interface Card	100Mbps

Table 4. OpenMosix Node Configuration

The disk-less nodes had the same configuration, just without the 20GB hard drive.

Combining the processor speed and memory of the 14 nodes resulted in an OpenMosix cluster which had approximately 10.3GHz clock speed, and 3.5GB of memory.

Software Configuration

Two different software configurations were used for the experiments. Both of these relied on the use of the Linux operating system, not just because that is what the software was designed on, but also because the separate clustering-systems, Beowulf and OpenMosix, relied on the use of Linux.

The Beowulf clusters operating system was built using the Rocks Cluster distribution (NPACI Rocks, 2004) version 3.1.0 which is based on top of the Red Hat Enterprise Linux 3 Advanced Workstation distribution. The software installation allowed the setup of the head-node, followed by a network boot and auto configuration of the remaining 12 compute-nodes. The kernel version of the system was 2.4.21, compiled for i686.

The research used a modified version of John the Ripper called John the Ripper-MPI (JtR-MPI) (Lim, 2004) and Cisilia to perform its tests because both utilities perform brute-force attacks, both can crack Microsoft Windows LAN Manager or Samba based passwords, and both can perform these attacks using a cluster of computers. One of the differences between the two utilities and their style of breaking passwords is their parallel paradigm. JtR-MPI is designed specifically to use message passing between computers to run parallel. Cisilia uses a different method of parallelisation, it is designed to spawn multiple copies of itself, which when run on a load-balancing cluster will split the load between computers and therefore run in parallel.

The version of JtR-MPI used was 1.6.36-mpi which is John the Ripper version 1.6.36 which has been modified to work in a Beowulf environment using MPICH for inter-node communication. MPICH version 1.2.5.2 was used for both the compilation and the execution of JtR-MPI. SSH version 3.7.1 was used to start the individual JtR-MPI processes on each of the nodes in the Beowulf cluster.

The OpenMosix clusters operating system was built using CHAOS version 1.2 (FIAT & Gobbledok, 2004) on the compute-nodes, and ClusterKnoppix (Vandersmissen, 2003) version 3.3-2003-11-19 on the head-node. The software installation allowed the setup of the head-node and compute-nodes from live CDs, which meant that the operating-system ran entirely off RAM disks and CD-ROMs. The kernel version of the system was 2.4.22-openmosix-2, compiled for i386. Cisilia version 0.7.3 was used for the tests on the OpenMosix cluster.

Password samples

To test a common set of passwords required the use of a password format that both Cisilia and JtR-MPI could crack. The only matching password format was Microsoft's LAN Manager (LANMAN) or Samba based passwords. The algorithm used to create these passwords has been proven to be quite weak, and is only being used for backwards compatibility with older Microsoft operating systems (Shaffer, 2004). LANMAN passwords are created by:

- Converting all alphabetic characters to their uppercase equivalent
- Truncating or padding the password to 14 characters (bytes)
- Splitting the password into two 7 character (byte) halves
- Using each 7 character half as a key to encrypt a known string using the DES encryption algorithm
- The two cipher-texts are then concatenated and stored on file

To test the two cracking utilities the password samples had to represent the broad spectrum of passwords from weak to strong. To create random passwords that represented this spectrum required the use of a method to create and determine a password and its strength.

The method used to create the password samples was taken from a system previously implemented in IBM's Lotus Notes and Domino (2004) software. Williams' (2001) article on the algorithms used to determine a passwords quality was used as the guidelines for creating passwords which had a rating from 5 to 14. Some of the factors of the algorithm include the passwords strength, its fallibility to dictionary-attacks and if it contains one or more mixed-case, numeric or punctuation characters. Some example rules for a password of quality eight is included in table 5.

A password that contains at least six (6) characters and includes at least two (2) number, mixed-case or punctuation characters	"th31zp", "catHaT", "wo!l,u"
A password that contains at least seven (7) characters and includes at least one (1) number and one (1) mixed-case character	"tNz8lqp", "pOnbjr2", "qnz7mKh"
A password that contains at least eight (8) characters and does not include words from the dictionary	"ibnzlspq", "pmbczxwh", "wxmpfvnx"

Table 5. Example Rules for a Password of Quality Eight

Tests Performed

The first test was performed by JtR-MPI on the Beowulf cluster. A maximum time was set to 3 days to ensure that the results from all the tests could be compared, and also to ensure that the tests could be finished on time. The results collected from this test included:

- Half the cipher-text password
- Half the plain-text, cracked password
- The username
- A digit representing which half of the password was cracked (a 1 or 2)
- A time in the format D:HH:MM:SS

Following this first test, a second test was performed with the same configuration. This test was used to measure the reliability of the utility.

The third test was performed by Cisilia on the OpenMosix cluster. Similar to the JtR-MPI tests, results which took longer than 3 days were discarded. Cisilia was configured to return results which contained:

- A time in the format HH:MM:SS.SS
- The username
- The plain-text, cracked password

During all the tests, no other processor-intensive tasks were being performed on the nodes. For the full 3 day period the only processes running on the clusters, apart from operating-system tasks, were the password cracking utilities.

RESULTS

Problems with Cisilia caused the program to run inconsistently. After decreasing the number of nodes down to 8, the test eventually started correctly. Unfortunately, after 5 hours and 37 minutes, the program stopped working only after having cracked 4 passwords.

A second test on 8 nodes ran for longer, but was manually stopped after having run for 3 days. During this time Cisilia only managed to crack 4 passwords, 2 of which were different from the first test.

Due to instability and inconsistency of Cisilia the results were discarded and a comparison between Cisilia and John the Ripper could not be performed. Fortunately the tests that were performed by John the Ripper were successful and provided the paper with some results that could be analysed.

The initial results from the JtR-MPI tests looked to be very impressive, with the cracking tool reporting 22 cracks within the first hour. In the next two hours of execution, the software successfully performed 8 more cracks. During the next 21 hours a further 4 more hashes were cracked, which meant after the first 24 hours JtR-MPI had broken 34 hashes. After the first day the software slowed down substantially, only cracking a further 4 more hashes in the next 2 days. These results are displayed in figure 1.

At the end of 3 days the utility had cracked 38 hashes.

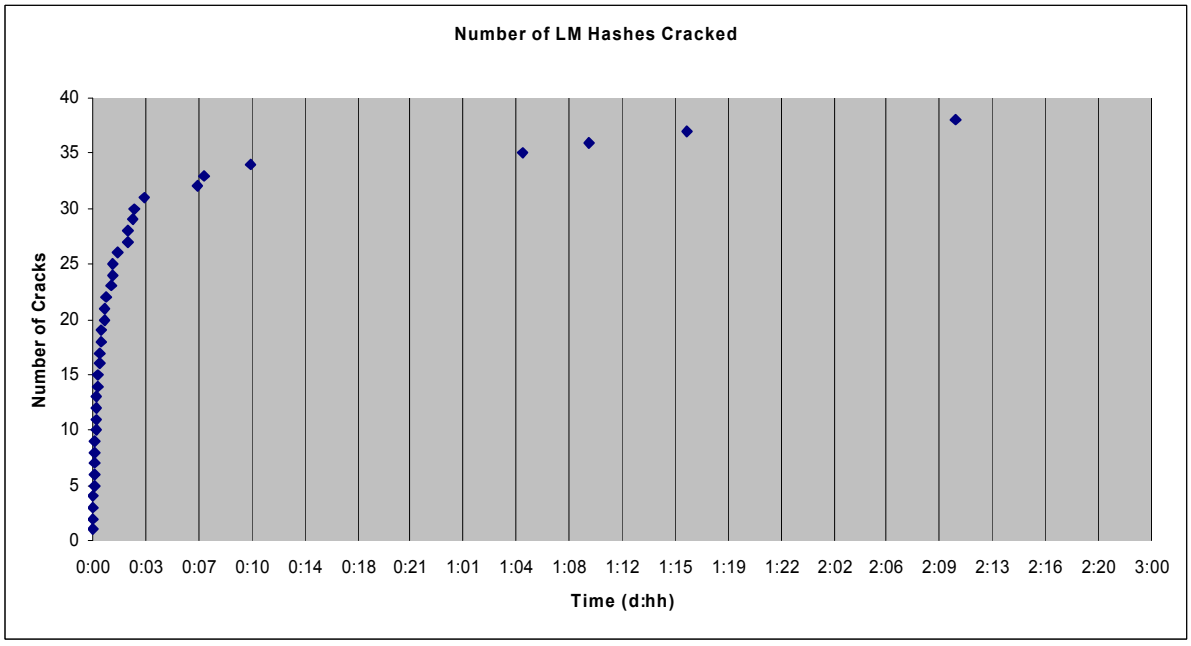

Figure 1. The Number of LM Hashes Cracked

Analysis

Analysis of the results from the successful John the Ripper test show that only 12 passwords were cracked in their entirety, meaning that both of the two separate hashes were cracked correctly. A further 27 passwords had their first half correctly cracked, meaning that the first 7 characters of the password were cracked correctly. This left a final 11 passwords which did not have either half cracked.

Of the 12 passwords which were completely cracked, 10 were cracked within the first 3 hours. The final 2 passwords took an additional 37 hours to be cracked. Of the 26 half-passwords which were successfully cracked, 24 were cracked within the first day.

An inspection of the time each password took to crack and its relative strength rating, according to the article by Williams (2001), revealed that in the case of the test, the rating did not have a strong effect on the order and time each password was cracked. An example of this is in the 2nd and 3rd passwords which were cracked, both which were rated quite highly. This can be seen in figure 2.

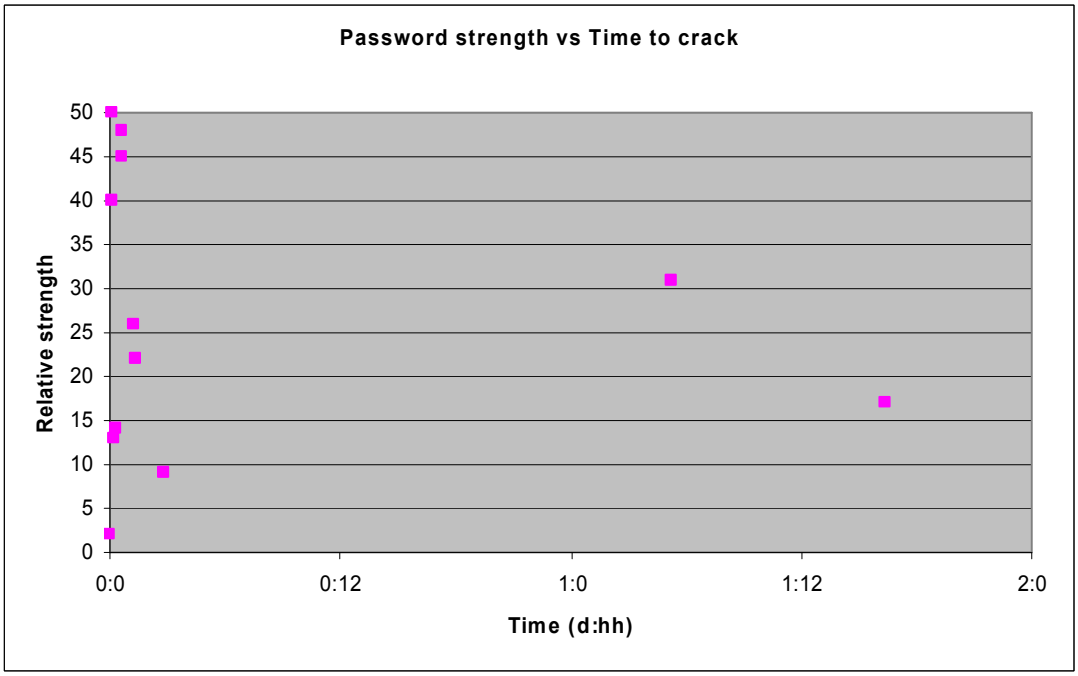

Figure 2. Password Strength vs Time to Crack

These results could be explained by the method in which LANMAN passwords are created. By first splitting the password in half, LANMAN passwords are effectively reduced to two passwords, each with a maximum length of 7. After inspecting the 2nd and 3rd passwords it is easy to see that each is made up of two separate, 7 character-long strings, each consisting of only alphabetic characters. Due to the fact that each separate half was easily cracked, it meant that the entire password could be cracked easily.

The passwords which took longer to crack, even though they had weaker password strength ratings than others, took longer because of their use of non-alphabetic characters such as a "(" character or "+" character. This is the case in the 2 passwords which took longer than 1 day to crack.

Limitations

The main limitation of the research was due to the instability of Cisilia. Without results from Cisilia, a comparison of two different clustered, password cracking utilities could not be performed. Another limitation to the research was the use of LANMAN passwords instead of a different password encrypting algorithm, such as MD5 or even DES based passwords. The LANMAN password format could explain why so many of the passwords were not cracked. Upon inspection of the log files produced by each node in the cluster, it would appear that the 3 day test should have produced many more positive cracks. Some possible answers to this problem could be that during the creation of the LANMAN passwords, the truncating and padding was done incorrectly, or else the method by which John the Ripper pads and truncates possible passwords was done incorrectly. After inspecting the list of cracked passwords this is confirmed, as 95.35% of the passwords had a length of 7 characters. Of the remaining 11 un-cracked passwords, none were a length of 7 characters.

Future Research

Some topics for further research into the topic of clustered password crackers could include an analysis into the effects of using stronger password encryption algorithms, such as MD5 based passwords. Another research topic could be to analyse some other crackers which claim to be able to work in a clustered environment, such as DKBF or even MDCrack.

Another topic of future research would be further analysis into a universal metric for measuring password strength.

Analysing changes to the clusters hardware could also be researched. For example, how well does password cracking scale with larger number of nodes? Or, how well does password cracking scale if utilising 64-bit processors?

REFERENCES:

ATStake. (2002). LC 4 (Version 4).

Bar, M. (2004). OpenMosix, an Open Source Linux Cluster Project. Retrieved April 1, 2004, from http://openmosix.sourceforge.net

Beverstock, D. (2003). Passwords are Dead! (Long live passwords?): SANS Institute.

C.I.S.I.ar. (2003). Cisilia (Version 0.7.3).

Dinquel, J. (2000). Network Architectures for Cluster Computing. Long Beach: California State University.

FIAT, & Gobbledok. (2004). CHAOS (Version 1.2).

Hare, C., & Siyan, K. (1996). Internet Firewalls and Network Security (2nd ed.). Indianapolis: New Riders Publishing.

IBM. (2004). developerWorks : Lotus : Products : Notes/Domino. Retrieved April 10, 2004, from http://www-136.ibm.com/developerworks/lotus/products/notesdomino/

Irakleous, I., Furnell, S. M., Dowland, P. S., & Papadaki, M. (2002). An experimental comparison of secret-based user authentication technologies. Information Management & Computer Security, 10(2/3), 100.

Klein, D. V. (1990). "Foiling the Cracker" A Survery of, and Improvements to, Password Security. Paper presented at the Second USENIX Workshop on Security.

Lim, R. (2004). Parallelization of John the Ripper (JtR) using MPI. Nebraska: University of Nebraska.

Liu, S., & Silverman, M. (2000). A Practical Guide to Biometric Security Technology. Retrieved 16 March, 2004, from http://www.computer.org/itpro/homepage/Jan_Feb/security3.htm

Muffet, A. (1991). Crack v4.1 - A Sensible Password Checker for Unix.

Neumann, P. G. (1994). Risks of passwords. Association for Computing Machinery. Communications of the ACM, 37(4), 126.

NPACI Rocks. (2004). Rocks Cluster Distribution. Retrieved February 1, 2004, from http://www.rocksclusters.org/Rocks/

Pfleeger, C. P. (1997). Security in computing (2nd ed.). Upper Saddle River, NJ: Prentice Hall PTR.

Ridge, D., Becker, D., Merkey, P., & Sterling, T. L. (1997). Beowulf: Harnessing the Power of Parallelism in a Pile-of-PCs. Paper presented at the IEEE Aerospace.

Rivest, R. L. (1992). The MD5 Message Digest Algorithm: RFC 1321.

Schneier, B. (1996). Applied cryptography : protocols, algorithms, and source code in C (2nd ed.). New York: Wiley.

Schneier, B. (2000). Secrets and Lies: John Wiley and Sons, Inc.

Shaffer, G. (2004). NT's Poor Password Encryption. Retrieved April 20, 2004, from http://geodsoft.com/howto/password/nt_password_hashes.htm

Solar Designer. (2002). John the Ripper (Version 1.6).

Spafford, E. H. (1992). Observing Reusable Password Choices. Paper presented at the 3rd UNIX Security Symposium, Berkely, CA.

Vandersmissen, W. (2003). ClusterKnoppix (Version 3.3-2003-11-19).

Williams, C. (2001). Understanding Password Quality. Retrieved April 25, 2004, from http://www-10.lotus.com/ldd/today.nsf/0/098c9f7d4a0cccbd85256abc0011e4f0?OpenDocument

Wood, C. C. (1996). Constructing difficult-to-guess passwords. Information Management & Computer Security, 4(1), 43.

COPYRIGHT