INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH & DEVELOPMENT

ISSN 2278 – 0211 (Online)

Implementing Morpheme-based Compression Security Mechanism in Distributed Systems

Rockson Kwasi Afriyie
Senior Lecturer, Department of Information and Communication Technology,
Wa Polytechnic, Ghana
Michael Asante
Professor, Department of Computer Science,
Kwame Nkrumah University of Science and Technology, Ghana
Edeh Michael Onyema
Lecturer, Department of Mathematics and computer Science,
Coal City University, Nigeria

Abstract:
This paper presents a morpheme-based text compression security mechanism to secure information confidentiality in distributed systems. The effectiveness of the security mechanism is illustrated using English Language text data. The results indicated that the proposed security mechanism can preserve the confidentiality of information. Additional desirable property of the technique is its ability to reduce the size of communicated information, which translates into a gain in terms of time, CPU, and storage resources, leading to improved performance in data transmission.

Keywords: *Morpheme, compression, confidentiality, distributed systems, security mechanism*

1. Introduction

A distributed system is a collection of autonomous computers linked by a computer network and equipped with software to perform a single task or to provide a single service [4]. The software enables computers to coordinate their activities and to share the resources of the system hardware, software, and data. Resources are to be secure to serve their purpose in distributed systems. One of the fundamental issues in distributed systems is security [17]. The multiple components composition of distributed systems makes it easier to compromise the security of the resources [4]. The risk of exposure of confidential information, the many opportunities for network staff to gain access to transmitted information, the deliberate attempts to disrupt a service and the probability of satellite or point-to point radio link to be intercepted with the appropriate equipment are some of the security challenges that can be exploited to attack a distributed system [14]. The exploitation can be triggered intentionally or by accident [1]. A trigger by accident means that there is no ill intention by the source, however, it may have adverse effect on the system, resources or services being distributed. The existence of the vulnerabilities in hardware, networks, operating systems, and applications or software [1] and the possibility to explore them requires putting in place extra security mechanisms to secure the data communicated in distributed systems.

Providing security is perceived to be more difficult in distributed systems. System vulnerabilities can be triggered intentionally or inadvertently changing them from potential to actual to cause security breaches [1]. This makes users and owners of digital information reluctant to distribute their data in an insecure environment because they fear that their information could be mishandled by anyone who has access to the network [15]. Unauthorised access can compromise the confidentiality and integrity of information in distributed systems. Reference [13], reported that security failures are experienced when there is a change in the application domain of security mechanism undermining a security model. Further, security mechanisms that are adequate and effective in a restricted environment often experience failures upon exposure to a more general environment. This study therefore seeks to explore morpheme-based data compression technique as a security mechanism to secure the confidentiality of information for safe storage and distribution. The study will also determine the adequacy and effectiveness of the mechanism in different application domains. Further, the contribution of the technique to the overall performance of distributed systems will be evaluated.

1.1. Data Compression Algorithms

Data compression refers to coding that represents the same information using fewer bits. The primary purpose of data compression is to reduce the number of bits used to store or transmit information. Based on the requirements of reconstruction, data compression schemes can be categorised into two broad classes: lossless compression algorithm and lossy compression algorithm [3], [5], [6], [9]. The lossless class of algorithms preserve the original data exactly, and lossy algorithms discard portion of the data [15], though not observable by the natural eye. The two main classes span into a plethora of techniques including, morpheme-based compression technique, which are desirable to specific areas such as text, images and audio. Generally, all the algorithms operations are premised on the fact that information contains excess data making it redundant. They therefore exploit this redundancy to reduce the size of the data.

1.2. Morpheme-based Compression Technique

A morpheme is the smallest grammatical unit of a language that has its own meaning; either a word or a part of a word [3, 10]. Though morpheme can be a word or part of a word, a word is independent unit of language. Thus, a word is freestanding language unit. Every language has a morpheme and every word comprises one or more morphemes. This means that the text data transmitted or communicated in a distributed system are made of morphemes. This study is leveraging on the composition of text data to implement morpheme-based text compression technique to secure the data within distributed systems.

1.3. Data Compression and Distributed Systems

Data compression involves a wide variety of software and hardware compression techniques with important applications in the areas of file storage and distributed systems [5]. The main purpose of compression, reducing redundancy, is however in conflict, with the field of distributed systems which considers redundancy or information replication as a means of ensuring reliability in providing services and resources to clients. Distributed systems believe that having information replicated, the impact of hardware and software faults on users can be reduced [4]. Redundancy in a message however, takes extra bit to encode, requiring more of commodity hardware such as processor and memory. When we successfully get rid of that extra information, we will have reduced the size of the message to reduce the requirement of extra storage and processing resources [5].

Data Compression has numerous desirable applicability in distributed systems. For instance, the ever-increasing disparity between processor speeds and I/0 rates and the growth of new applications demanding enormous data rates are pushing data compression implementation to the forefront of distributed systems [2]. Compression is used to reduce the demand for storage resource, network bandwidth, and to extend physical memory [2], [3]. The important role data compression plays in distributed systems is reported [2]. The concept has been used in file storage and distributed systems to reduce the amount of data being transferred leading to improved performance. In a situation where a network is run on "pay as you use" in proportion to amount of data transfer basis, compressing data before transfer lowers cost. Any communication involving networks – wired or wireless is likely to benefit from data compression [2]. Mobile computing has the greatest potential of benefitting from the use of compression, since they are likely to execute applications that do not fit comfortably in physical memory [2].

The application of data compression can be used to store information in less space or transfer it less expensively and at a reduced latency in distributed systems. These often go hand in hand, for example, when compressed data is written to a disk: the disk I/O takes less time, since less bits are being transferred, and less disk space is occupied on the disk storage after the transmission. A study [6], reports the benefits of compression in reducing the size of the original data may translate into a gain in one or more of time, CPU, memory or static storage. It however cautioned that time to compress the data can exceed the savings from transferring less data, resulting in degradation. A study [2] indicates that data compression actually improves overall performance in distributed systems.

1.4. Threats

According to [1], threats against systems are entities that can intentionally exploit or inadvertently trigger specific system vulnerabilities to cause security breaches. An attack is an intentional exploitation of vulnerabilities, and an accident is an inadvertent triggering of vulnerabilities. Both materialise threats, changing them from potential to actual [1]. Therefore, users and owners of digital information are reluctant to distribute their data in an insecure environment because they fear that their information could be mishandled by anyone who has access to the network [15]. Unauthorised access can compromise the confidentiality and integrity of information. Examples of security threats in distributed systems are interception, interruption, modification and fabrication, normally perpetuated by adversaries or attackers. Attackers in a distributed system can be active or passive. Passive attacker mainly observes data or information without modifying or compromising services. They represent the interception and interruption forms of security threats. An active attacker modifies or deletes data and may cause service to be denied to authorised users.

Because of its networked nature, the communication channel presents a vulnerability in distributed systems [4]. Communication related active attacks attempt to modify the data sent over a communication channel. Therefore, distributed systems threats are mainly on the attack of their communication channels. Common types of attacks on communication channels are eavesdropping attacks involve obtaining copies of messages without authorization, masquerading attacks involve sending or receiving messages using the identity of authorized users without their authority, message tampering attacks involve the interception and modification of messages so that they have a different effect than what was originally intended. Replay attacks involve resending intercepted messages at a later time in order to cause an action to be repeated, and denial of Service attacks is saturating a communication channel with data to deny others access to a resource [18].

1.5. Security Mechanisms in Distributed Systems

Information can be stored or transmitted protected in a distributed system by implementing the appropriate security mechanisms. A security mechanism refers to a process or a device incorporating such a process to detect, prevent, or recover from a security attack [16]. Some of the mechanisms that can be employed to protect against security threats include encryption, authentication, authorisation, and auditing [4].

Encryption refers to encoding a specified information with the aim of hiding its content [20]. Cryptography provides several secure algorithms for encrypting and decrypting messages using secrets called keys [16]. A cryptographic key is a parameter used in an encryption algorithm in such a way that the encryption cannot be reversed without knowledge of the key [16]. The two main classes of encryption algorithms are the use of shared secret keys – the sender and the recipient are the only users who share a knowledge of the key. The other class of encryption algorithms is the one that practices public/private key pairs. The message source uses a public key – one that has already been published by the receiver – to encrypt the message. The receiver uses matching private key to decrypt the message. Although many principals may examine the public key, only the recipient can decrypt the message, because they have the private key. Both classes of encryption algorithm are extremely useful and are used widely in the construction of secure distributed systems [16].

Authentication verifies the claimed identity of an entity (a user, server, client). In order to determine if an entity is authorised to access particular data or services it is necessary to know who the entity is. Examples of authentication mechanisms include passwords, chip cards, and biometric identification [4].

Authorisation permits entities to only access those resources that they are entitled to. This requires entities identification, tracking their permissible resources, monitoring and prevention of unauthorised access to resources. Authorisation is a key requirement for confidentiality and integrity of information.

Auditing security mechanism detects what was illegitimately accessed by an entity. The mechanism therefore, provides audit trails of activities transpired in the system to enable user accountability. it provides evidence of who to hold accountable for an incident, and when the incident occurred. Auditing does not provide direct protection against security threats by preventing attacks, however, it is necessary in determining what went wrong when an attack could not be prevented [4].

2. Related Works

To situate this study, morpheme-based compression security mechanism in distributed systems, with respect to related work, reference is made to a number of data compression approaches, involving energy saving in distributed systems, data compression and security, and compression and encryption [15]. Data compression algorithms have been developed to operate over distributed, networked nodes and achieve significant energy savings with little or no informational loss.

A survey [1], discussed vulnerabilities and threats in distributed systems and outline a number of mechanisms for mitigating them. Among the mechanisms mentioned are using trust in Role-Based Access Control (RBAC). However, trust only cannot be used as a guarantee to secure a system since this can easily suffer betrayal. There could be an accidental trigger of vulnerability as indicated above. Users of distributed systems are highly unpredictable. Further, there is reported inadequacy in identity-based approaches to access control and in open systems [1].

Literature [15], reports that exploring data compression and security is a new research direction where more testing and the development of algorithms are being studied. The study sought to answer two interesting questions: "What is the cost of encryption in terms of file size after performing compression?" And "how bad is performing first encryption and then compression?". The study findings indicated that it is inefficient to perform first encryption and then compression. The file size increased and, in some cases, the resulting output far larger than the original input. It however, demonstrated that the cost of security is negligible if compression is performed first before encryption. This study indicated that data compression has a relation with data security through encryption in spite of the conflicting acceptance of randomness between encryption and compression [15].

A survey [19], concerning Lempel–Ziv data compression on parallel and distributed systems, obtained an approximation scheme which is suitable for a small-scale system and adaptively worked on a large-scale parallel system when the file size is large. The study used input data blocks to achieved a satisfying degree of compression effectiveness and obtaining the expected speed-up on a real parallel machine. The approach improves compression effectiveness, and scalability of a parallel implementation of sliding window compression on a distributed system with low communication cost seems to guarantee robustness only on very large files. They show that LZW compression is scalable and robust [19].
Data communication is implemented in distributed systems as a technique to reduce the amount of data exchanged between nodes and results in energy saving [7]. The critical role of data compression in saving energy of data storage, wireless communication and distributed system is reported [7]. The study reported that data compression is used naturally in wireless sensor networks (WSN) node energy savings. The study which evaluated energy efficiency of data compression in wireless sensor networks concluded that there is overall energy efficiency of communications energy consumption [7]. The lossless compression examples they analysed indicated that selecting the algorithm that has better compression performance to compress data can save more energy.

By local data reduction and transformation, [8] reported that edge mining can reduce the number of packets sent, energy usage and remote storage requirements. Edge mining is data mining that takes place on mobile and smart technology devices located at the edge points of the Internet of Things [8]. Their study indicated edge mining potential to reduce the risk to personal privacy through embedding of information requirements at the sensing point, limiting inappropriate use [8].

3. Method

The section outlines and discusses the step-by-step methods used in the study. Key areas include the design and generation of a dictionary for the morphemes, the algorithm to decompose the information and to obtain the code representation as well as the implementation of the proposed security mechanism.

3.1. Morpheme Dictionary Generation

The various alphabetic characters can be represented as numbers [11, 12]. The study designed and used morpheme coding system that was used to assign the morphemes to appropriate code representation. Table 1 shows the alphabet number representation.

Char	Code	Char	Code
Blank	0	N	14
A	1	O	15
B	2	P	16
C	3	Q	17
D	4	R	18
E	5	S	19
F	6	T	20
G	7	U	21
H	8	V	22
I	9	W	23
J	10	X	24
K	11	Y	25
L	12	Z	26
M	13		

Table 1: Alphabets Number Assignment

3.2. Morpheme-Coding System

The morphemes were coded by a simple approach of adding the coded numbers for each character [11]. The target morpheme assigned, for instance, the morpheme “ing” was coded as follows:

Step 1: assign the constituent characters in the morpheme with their respective numbers

i = 9

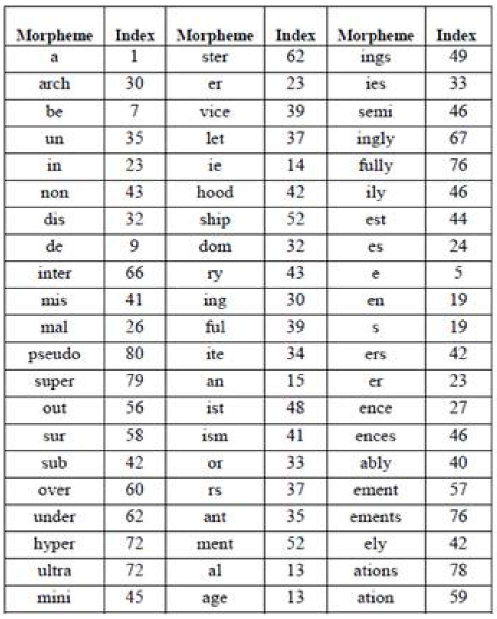
n = 14

g = 7

Step 2: sum them up

9 + 14 + 7 = 30

From the algorithm, the morpheme “ing” would be assigned the correspondent code representation 30. Thus, in the dictionary, the morpheme “ing” would be represented by the code 30. All the other morphemes were coded likewise and assigned code representations. Using the algorithm, the morpheme dictionary was obtained as shown in Table 2.



Morpheme	Index	Morpheme	Index	Morpheme	Index
a	1	ster	62	ings	49
arch	30	er	23	ies	33
be	7	vice	39	semi	46
un	35	let	37	ingly	67
in	23	ie	14	fully	76
non	43	hood	42	ily	46
dis	32	ship	52	est	44
de	9	dom	32	es	24
inter	66	ry	43	e	5
mis	41	ing	30	en	19
mal	26	ful	39	s	19
pseudo	80	ite	34	ers	42
super	79	an	15	er	23
out	56	ist	48	ence	27
sur	58	ism	41	ences	46
sub	42	or	33	ably	40
over	60	rs	37	ement	57
under	62	ant	35	ements	76
hyper	72	ment	52	ely	42
ultra	72	al	13	ations	78
mini	45	age	13	ation	59

Table 2: Morpheme Dictionary
Adopted [3] from Afriyie, et.al (2014)

3.3. Demonstration of the Proposed Algorithm for English Language Text Data

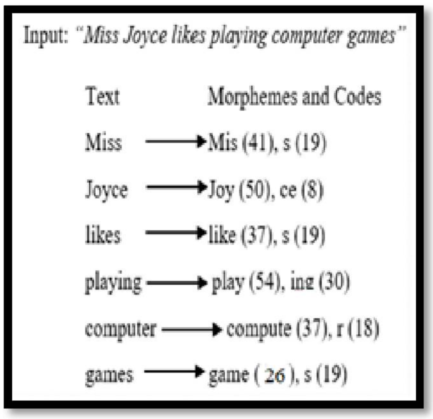
English sentence is selected for the illustration of the proposed approach since this is the language used in the implementation.

"Miss Joyce likes playing computer games"

- The proposed algorithm is as follows:
- Read the text input
- Partition the string input into possible morphemes
- Extract morphemes
- Scanned the extracted morpheme through the dictionary to do a match if there exist
- Represent the identified morphemes with the appropriate code
- If the string in the input text is not in the morpheme unit, pass string to the output unit text unaltered.

3.4. Theoretical Consideration of the Proposed Algorithm

A sentence under consideration was decomposed morpheme-by-morpheme vis-a-vis the number of morphemes in each word. For example, "joyce" was reconstructed as "joy" and "ce" as identified in the sentence. This algorithm proposes that, if the morpheme "joy" is encoded and represented by a token (code representation) which appears different from the original input word, it will be very difficult for a hacker to guess the original input. The implementation of the proposed algorithm is illustrated in figure 1.



Input: *"Miss Joyce likes playing computer games"*

Text	Morphemes and Codes
Miss	$\longrightarrow$ Mis (41), s (19)
Joyce	$\longrightarrow$ Joy (50), ce (8)
likes	$\longrightarrow$ like (37), s (19)
playing	$\longrightarrow$ play (54), ing (30)
computer	$\longrightarrow$ compute (37), r (18)
games	$\longrightarrow$ game (26), s (19)

Figure 1: Morphemes and Codes Representation Adopted
[3] From Afriyie, et.al (2014)
Output: 41190508037190543007318026l9

3.5. The Decompression

The decompression algorithm is similar to the compression aspect. It uses the same components as the compression algorithm. The coded representation is replaced by the original morpheme strings in the text. The coded information now becomes the input data.

Input: 4119050803719054300731802619

Output: Miss Joyce likes playing computer games

4. Evaluation and Analysis of Results

Input: "Miss Joyce likes playing computer games"

Output: 4119050803719054300731802619

The original input is now represented in codes to hide the actual data or message making it difficult for a third party to understand the message that was transmitted.

Output: 4119050803719054300731802619

The encoded information is rendered useless to the casual observer even when it is intercepted. Only the decompressed information, which is done by the legitimate receiver, can perform the decompression to understand it. Therefore, the confidentiality of the message is preserved. The proposed security mechanism therefore corroborates [4], that the goal of security in distributed system is to protect data and services against adversaries. It can therefore be used in the construction of secure distributed systems [16]. The mechanism can suffice accidental trigger of vulnerability or when trust suffers betrayal and provide adequate protection of information in open systems like the internet.

Further, the number of bits in the original message:

(39 *8, i.e. 8 bits system = 1byte, is 312 bits or 39 bytes) is reduced in the coded representation.

(28 *8, i.e. 8 bits system = 1byte, is 244 bits or 28 bytes).

The reduction in the number of bits or bytes results in producing a smaller size message for speedy transfer because it will require less bandwidth. This implies that deploying the morpheme-based compression security mechanism in distributed systems comes with other benefits. The message confidentiality is not only preserved, but also size of the message is reduced to require less energy in terms of transmission power such as processor, memory and bandwidth. The reduction in the size of the message supports [6] that benefits of compression may translate into a gain in one or more of time, CPU, memory or static storage, leading to improved performance in distributed systems [2].

5. Conclusions

This study has revisited the morpheme-based compression technique proposed in earlier work undertaken by the author [3]. While the basics of the technique have not changed, the technique presented in this study have explored new directions that allow it to be implemented as a security mechanism in distributed systems.

When compression algorithm is applied to data before transmission, hackers find it harder to understand the transformed data. This was realised by implementing the morpheme-based data compression technique. The results show that adopting morpheme-based compression technique, as a security mechanism preserves both confidentiality and integrity of information in distributed computing.

Other desirable properties of the mechanism include its ability to reduce information size, translating into a gain in terms of time, CPU, storage resources, leading to improved performance as it makes the information more suitable for quick transfer of large amounts of data often encountered in distributed systems.

6. References

i. B. Bhargava and L. Lilien, “Vulnerabilities and Threats in Distributed Systems. Department of Computer Sciences and Center for Education and Research in Information Assurance and Security (CERIAS), Purdue University, West Lafayette, IN 47907, USA.

ii. F. Douglis “On the role of compression in distributed systems ACM SIGOPS Operating Systems Review. Volume 27 Issue 2, Pages: 88-93, April 1993, ACM New York, NY, USA

iii. K. R., Afriyie, J. B. Hayfron-Acquah, and J. K. Panford, “Optimising Storage Resource using Morpheme-based Text Compression Technique,” International Journal of Computer Applications 93(2), pp.33-42. May. 2014. Available at: http://ijcaonline.org/archives/volume93/number2/16190-5414 doi 10.5120/16190-5414

iv. I. Kuz, M. Manuel, T. Chakravarty and G. Heiser. Distributed Systems. School of Computer Science & Engineering. The University of New South Wales, 2008.

v. M. Nelson and G. Jean-loup, Data Compression Techniques. [CSA215], University of Bahrain, College of Applied Studies.

vi. E. Jeannot, B. Knutsson and M. Bjorkman, Adaptive Online Data Compression. Distributed System Laboratory University of Pennsylvania, USA.

vii. S. Liu, Y. Liu, X. Chen, and F. Xiaoping, A new scheme for evaluating energy efficiency of data compression in wireless sensor networks. International Journal of Distributed Sensor Networks, SAGE Journals 2018. Available at: https://journals.sagepub.com/home/dsn

viii. E. Gaura, J. Brusey, M. Allen, R. Wilkins, D. Goldsmith, and R. Rednic, (2013) Edge mining the internet of things. IEEE Sensors, volume 13 (10): 3816-3825. Available at: http://dx.doi.org/10.1109/JSEN.2013.2266895

ix. S. Khalid Introduction to Data Compression, 3rd ed., Morgan Kaufmann, Elsevier. 500 Sansome Street, Suite 400, San Francisco, CA 94111, 2006

x. Cambridge Dictionary of English, Ed., International Dictionary of English: Low Price Edition, University Press, 2002.

xi. R. K. Afriyie, “Optimising storage resource using morpheme-based text compression technique,” M.Phil. IT, thesis, Kwame Nkrumah University of Science and Technology, Kumasi, Ghana, Jul. 2013.

xii. R. Lafore, Data Structures & Algorithms in Java, 2nd Ed. Sams Publishing, 800 East 96th Street, Indianapolis, Indiana 46240.

xiii. R. J. Anderson, Security Engineering: A Guide to Building Dependable Distributed Systems, 2nd ed. John Wiley & Sons.

xiv. J. D. Moffett, Security & distributed systems. Department of computer science

xv. University of York, England.

xvi. B. Carpentieri, “Efficient Compression and Encryption for Digital Data Transmission”. Security and Communication Networks, Volume 2018, https://doi.org/10.1155/2018/9591768

xvii. I. Shadmanov, K. Shadmanova and R. Rakhimov “A Survey on Security Services and Mechanisms in Distributed Systems” International Journal of Trend in Research and Development, Volume 3(1), pp. 262-266. 2016. Available Online@www.ijtrd.com

xviii. M. Firdhous, “Implementation of Security in Distributed Systems – A Comparative Study”. International Journal of Computer Information Systems, Vol. 2, No. 2, pp. 1-6, 2011

xix. M. Supriya." Detection of Malicious Behaviour of Agent Using Attack Detection Strategies under Multiagent System.” International Journal of Modern Engineering Research (IJMER), vol. 08, no. 05, 2018, pp. 33–41.

xx. [S. De Agostino, “Lempel–Ziv Data Compression on Parallel and Distributed Systems,” Algorithms, vol. 4, pp. 183-199. Sept. 2011. Available at: www.mdpi.com/journal/algorithms