Volume 5, Issue 1, January 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Towards A Cryptographically- Secure Cloud Security Solution

Chinedu, Paschal Uchenna[1], Ugwuegbulam, Chidiebere[2], Akagha, Chinaka[3]
[1] Department of Information Management Technology, Federal University of Technology, Owerri, Nigeria
[2, 3] National Open University of Nigeria, Owerri Centre, Nigeria

As with any emerging technology, cloud computing offers a rare opportunity to rework security and IT controls for a better tomorrow. In an environment where information systems security and privacy has become paramount to enterprise customers, risk of unauthorized access to information in the cloud poses a significant concern to cloud computing stakeholders. In order to combat the unavoidable threats concerns of the cloud computing stakeholders, this research paper advocated that proper implementation of security; privacy and forensic measures should not just be seen as the cloud providers' sole concern, but the responsibilities of all consumers of the services. Thus, the research prescribes a security technique to remedy some of these risks and threats through the provision of novel cloud-based data encryption techniques that proposes AES-256 encryption using Rijndael encryption Algorithm, and a one-way Hash Algorithms referred to as "Deciv Algorithm", which could help cloud users maintain control of their data at rest or in transit within the cloud networks rather than outsource to external vendors as usual.

Key Words: Cloud Computing, Cloud Security, Encryption, Algorithms

I. INTRODUCTION

1.1 Background of Study

Encryption has been used as a security measure to render data unintelligible to unauthorized parties. Encryption is gaining popularity as social and community computing (such as the cloud) is gaining momentum.

Data is the most important resource to a user, and in a public cloud where communal computing and multitenancy is practices, encryption must be inevitable to ensure confidentiality and integrity of data and data bank. Encryption is implemented with a hope to curb impersonation, wiretapping, piracy, spoofing and data diddling which are common abuse in a multi-user environment. There has been Blowfish, Rijndeal, AES, Python, Crypt and so many other encryption algorithms presently in existence to combat some of these known threats.

However, this hope has not been achieved as Hackers are all out with their botNets and Rainbow tables decrypting all the known algorithms. Users with confidential data are gripped with fear of insecurity, even the service providers are not confidently sure of the data security despite the encryption used. And where data security measures are implemented by cloud providers, users are not sufficiently assured sole ownership of control of their useful, confidential or classified sensitive data. This challenge which raised questions within the constituencies of consumers of cloud services has become the concern of this research paper.

1.2 Problem Statement

In view of these challenges, this research paper's concern remains what security and privacy measures should be taken to ensure sole control of data by cloud users? Which cloud security solution would be less vulnerable to hacker's attack or attack penetration? Why are hackers always breaking encryptions? Is there a way to stop unauthorized decryption? Any solution?

In suggesting answer to this, the research is not limited to the direct deployment of cloud services provider or vendor security solutions but instead it further emphases a difference with deeper advocacy that proper implementation of security, privacy, and forensic measures should be deployed with maximum participation from all consumers of the services within the various constituencies described by the deployed cloud computing model.

1.3 Research Aim

Therefore, the research aims is to design and propose a novel cloud-based data encryption solution that deploys AES-256 encryption technique using Rijndael encryption Algorithm, and a one-way Hash Algorithms (referred to as "Deciv Algorithm"), which would help cloud users maintain control of their meaningful, confidential and sensitive data (at rest or in transit within the cloud networks) rather than outsource to external vendors as usual. The system if implemented or algorithm adopted, would improve the existing state of data privacy, and security of cloud data and application environment as it would effectively hide meaningful user data from all external parties to a virtual network- even from the service provider

II. CLOUD COMPUTING

2.1 What is Cloud Computing?

A cloud has been defined as a pool of virtualized computer resources [2]. In their paper, [2]. argued that a cloud is more than a collection of computer resources owing to the fact that it provides a mechanism to manage those resources. Management here includes provisioning, change requests, re-imaging, workload rebalancing, de-provisioning, and monitoring.

Cloud computing is a term used to describe both a platform and type of application. A cloud computing platform dynamically provisions, configures, reconfigures, and deprovisions servers as needed. Servers in the cloud can be physical machines or virtual machines. Advanced clouds typically include other computing resources such as storage area networks (SANs), network equipment, firewall and other security devices [2].

2.2 Risk Assessment of the Cloud Model

The cloud model can be thought of as being composed of three **service models** (**Table 2.2.1**), four **deployment models** (**Table 2.2.2**) and five essential **characteristics** (**Table 2.2.3**). Overall risks and benefits will differ per model and it is important to note that when considering different types of service and deployment models, enterprises should consider the risks that accompany them [6].

Table 2.2.1—Cloud Computing Service Models

Service Model	Definition	To Be Considered
Infrastructure as a Service (IaaS)	Capability to provision processing, storage, networks and other fundamental computing resources, offering the customer the ability to deploy and run arbitrary software, which can include operating systems and applications. IaaS puts these IT operations into the hands of a third party.	Options to minimize the impact if the cloud provider has a service interruption
Platform as a Service (PaaS)	Capability to deploy onto the cloud infrastructure customer-created or acquired applications created using programming languages and tools supported by the provider.	• Availability • Confidentiality • Privacy and legal liability in the event of a security breach (as databases housing sensitive information will now be hosted offsite) • Data ownership • Concerns around e-discovery
Software as a Service (SaaS)	Capability to use the provider's applications running on cloud infrastructure. The applications are accessible from various client devices through a thin client interface such as a web browser (e.g., web-based e-mail).	• Who owns the applications? • Where do the applications reside?

Table 2.2.2—Cloud Computing Deployment Models

Deployment Model	Description of Cloud Infrastructure	To Be Considered
Private cloud	• Operated solely for an organization • May be managed by the organization or a third party • May exist on-premise or off-premise	• Cloud services with minimum risk • May not provide the scalability and agility of public cloud services
Community cloud	• Shared by several organizations • Supports a specific community that has shared mission or interest. • May be managed by the organizations or a third party • May reside on-premise or off-premise	• Same as private cloud, plus: • Data may be stored with the data of competitors.
Public cloud	• Made available to the general public or a large industry group • Owned by an organization selling cloud Services	• Same as community cloud, plus: • Data may be stored in unknown locations and may not be easily retrievable.
Hybrid cloud	A composition of two or more clouds (private, community or public) that remain unique entities but are bound together by	• Aggregate risk of merging different deployment models • Classification and labelling of

	standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load balancing between clouds)	data will be beneficial to the security manager to ensure that data are assigned to the correct cloud type.

Table 2.2.3—Cloud Computing Essential Characteristics

Characteristic	Definition
On-demand self service	The cloud provider should have the ability to automatically provision computing capabilities, such as server and network storage, as needed without requiring human interaction with each service's provider.
Broad network access	According to NIST, the cloud network should be accessible anywhere, by almost any device (e.g., smart phone, laptop, mobile devices, PDA).
Resource pooling	The provider's computing resources are pooled to serve multiple customers using a multitenant model, with different physical and virtual resources dynamically assigned and reassigned according to demand. There is a sense of location independence. The customer generally has no control or knowledge over the exact location of the provided resources. However, he/she may be able to specify location at a higher level of abstraction (e.g., country, region or data center). Examples of resources include storage, processing, memory, network bandwidth and virtual machines.
Rapid elasticity	Capabilities can be rapidly and elastically provisioned, in many cases automatically, to scale out quickly and rapidly released to scale in quickly. To the customer, the capabilities available for provisioning often appear to be unlimited and can be purchased in any quantity at any time.
Measured service (Pay as you go)	Cloud systems automatically control and optimize resource use by leveraging a metering capability (e.g., storage, processing, bandwidth and active user accounts). Resource usage can be monitored, controlled and reported, providing transparency for both the provider and customer of the utilized service.

As can be observed in the characteristics listed in **Table 2.2.3**, there are many approaches and challenges to cloud computing. Benefits to the enterprise, as well as risks, will vary depending on the types of service and deployment models selected [6].

III. DATA ENCRYPTION

Encryption is an information security measure that renders data unintelligible to unauthorized readers. It is a coded transformation of data into a form unreadable to intruders and interlopers who lack the appropriate key to decrypt the encoded data [5].

Encryption involves using a cryptographic algorithm and a cryptographic key in order to transform a plaintext into a ciphertext or not obvious text [1]. The figure gives us diagrammatical illustration of a basic encryption system.

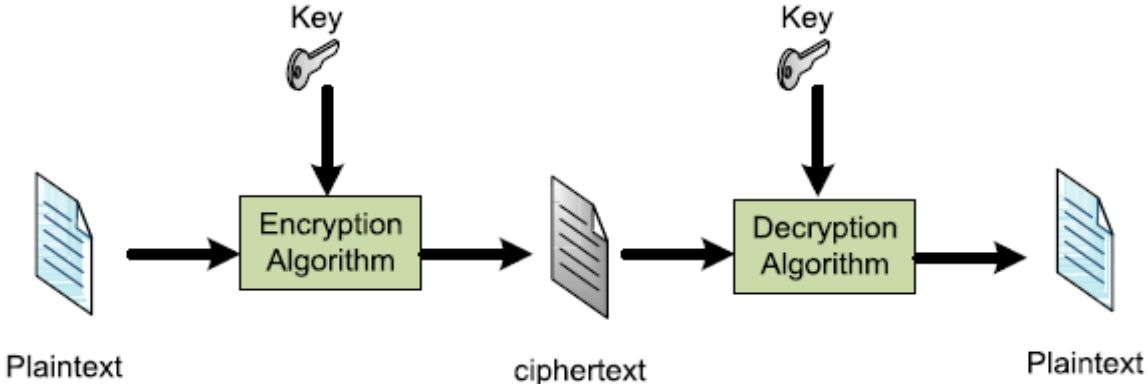

Figure 3. Basic Encryption [1].

Encryption is gaining popularity as social and community computing (such as the cloud) is gaining momentum. According to [8], Encryption Technique is important not just for the data but also for database controls and communication channels such as the Secure Socket Layer (SSL). In a public cloud where communal computing and multi-tenancy is practiced, encryption must be inevitable to ensure confidentiality and integrity of information and data store. As a mitigation technique that could sufficiently address the risk of information disclosure threat type, [12] elucidated that Encryption is implemented to curb impersonation, wiretapping, piracy, spoofing and data diddling which are common abuse in a multitenant environment.

3.1 Basic Encryption Techniques:

The two most important techniques for encryption are the Symmetric Encryption and Asymmetric encryption. Both are vital and applicable as a hybrid encryption for secure implementation of a public cloud.

3.1.1 Symmetric Cryptography

This is a private-key encryption technique that is based on shared secrete between the two communicating parties [10]. This party generates a key that allows them to encrypt and decrypt messages and the key is kept secret. No one can read the message except with the key. The key is just an algorithmic seed (called $Salt in PHP) that will turn data into rubbish. That same 'seed' is required to turn the rubbish into the original data [4].

Single user, such as an organization in a cloud can choose a particular key, use it to encrypt his information base and keep the key very secret. Such information-base will continue to be confidential until the organization let the key out to someone else or intruders hacked-in and retrieve the key.

3.1.2 Asymmetric Cryptography:

This is a public key encryption that is based on matched key-pair. [10] argued that the participant to such secure communication generates two keys. One portion is made private and kept secret, while the order is published to the public. The public can use the key to decrypt any information encrypted with the private key-pair. Although this is called public, yet it's only those that have they key-pair (public) that can decode and read the ciphered information.

In a public cloud, service provider can have a cloud-wide public key which the cloud- tenants can use to decrypt information within the cloud environment while outsiders and intruders cannot read their information since they do not have the key. This will also make it easy to detect foreign and fabricated information because the cloud-public key cannot decrypt the foreign information injected into the platform by interlopers and cyber criminals.

3.1.3 Hybrid Encryption Scheme:

A Hybrid cryptosystem can be used to crypt and decrypt both private and public keys. Such systems can be used for key distribution and to encrypt bulk data with high speed [12]. This scheme is used today to secure web-based transactions as well as secure email services and other communication systems such as Netscape communicator, secure socket layers and digital signatures. A typical example of such system is the R.S.A machine.

3.2 Encryption Algorithms

Encryption Algorithms are the patterns used to transform the data being encrypted into cipher. It is made up of some mathematical constructs, logics and transformation procedures, reshufflings and alphabetic replacements etc. [11]. Most Algorithms are very secure at the start but with time hackers and cyber Abusers decipher the underbelly of the security code-pattern and use it to break the encryption in order to have access to secured files and virtual resource [3].

Efforts upon efforts have been made, one algorithm after another, but it's just a matter of time before the hacker make a breakthrough. This is apparently because the output of every encryption follows a consistent pattern by just looking at a result of an encryption; you can tell which algorithm produces it.

For instance, every encryption made with Blow-fish starts with $2a& or with &2y&. Also, those made with Unix's UU encode starts with M and can be easily decoded with UU decode ( ) Algorithm. Those just converted to hexadecimal using bin2hex ( ) contain numbers (0 – 9) and letters (a – e) only

So with extra effort and technical knowhow hackers can master these using pattern-matching and apply brute-force attack.

3.2.1 PHP Hash-Code Algorithms

PHP Hash-code algorithm have been commended for their irreversible nature and unbreakableness; because not even the developer of the algorithm can decrypt it In the PhP encryption algorithm history-line, we have sha1, sha256, haval160, MD5( ), Hash_Mac, Blowfish…. The list is bottomless.

Blowfish (BCrypt)

BCrypt has been adjudged the most secure Hash algorithm in the open source; not only because it is one-way but because it is difficult (some say impossible) to guess or break. [13], citing [7] reiterated that It is a common knowledge that Hackers now employ fast-working Robots that can apply brute-force attack on any cipher to decipher it within the shortest time possible. Blowfish, however, won its trophy by applying a salt parameter in its algorithm that slows down any automated cracking attempt targeted on its output.

Bcrypt uses 'salt' to define the format of the encryption. The salt makes the output pattern to be different in each round of encryption. If a particular data is encrypted twice the two output will be different, if different salt were used in the encryption. This is what confuses Hackers and make them surrender, because they cannot match any pattern ie salts can be made of alphabets in upper and lower case, numbers and special characters for more added robustness and impossible-to-guess algorithm. BCrypt has two kinds of salt-pattern: The $2a$xx$ and the $2y$xx$ patterns. The xx in the salt parameter represents a two-digit number that is used to slow down any automated attempt to crack Bcrypts cyphertext, such as those perpetrated by Robots. The two-digit number ranges from 04 to 32 makes the cracking attempts to work n-times slower, where n is the two digit number (xx).

A typical example of Bcrypt output is as below:

$2y$19$r9v96x3F4HQSAios6N5wMu0wzQ7PuposutBku/DEF/YNQzfla2.

This output will slow any automated attack down 19 times the normal speed, making it to take years to crack. A handful of experts believe that using a randomly generated salt will make the cryptext stronger and more difficult to guess.

3.2.2 Advance Encryption System (AES)

AES is a 256-bit key encryption that was formally called Rijndael. It is also a saltable encryption. There are about $116X10^{25}$ possible combinations of values in an AES. That's more than the number of atoms in the Universe [13].

It will take a fast working Brute attack over a million years to successfully crack a Rijndael encryption [13], thus circumventing AES-256 is impractical and impossible. As for the blackHat attackers with their Rainbow table, they will require 700 Hexillion Bytes of Hash-codes and password combinations to be able to overcome AES security [9], and of course the device to contain such a file is yet to be produced. EAS is the world's number one choice of encryption.

However, Crypt-texts from the same algorithm always have similar pattern with which the algorithm will be recognized. This gives the attackers a fore-knowledge of the Algorithm they are dealing with, and a rainbow table can then be prepared for it. This is indeed a draw back.

IV. CUSTOMIZED ALGORITHM PROPOSED FOR THE CLOUD

Every Encryption algorithm has a uniform pattern in the cyphertext with which it is recognized. Crypto-analysts can build a rainbow table to decrypt the cyphertext even if it takes a long time. Consequently, no encryption is foolproof except the "Deciv Algorithm".

The concept of "Deciv Algorithm" is to use salty encryption algorithms but removed the uniform pattern with which the algorithm can be identified and replace it with something else. The objective is to make the cyphertext unrecognizable to hackers so that no Rainbow table can be built for it and thus it remains undecipherable.

Two Algorithms are proposed. One is a public key encryption which runs on the platform and encrypts very confidential data including virtual databases from intruders. However every tenant of the cloud will have the decrypting key. The other is a private key encryption which only the user knows and keeps secret.

Each of these encryptions does not follow universal open-source algorithms which the hackers can identify, which has a greater than zero probability to be in the rainbow table, which has a potential of being broken someday; rather the proposed encryption algorithms are products of series of encryption sequence and transformations using PHP String functions. The cyphertext has never existed anywhere. They are customized.

4.1 The Private Key Encryption (Proposed)

This symmetric Encryption uses the AES-256 encryption (known as Rijndael Algorithm) as the core, with modifications made to shield its identity from Hackers and also to make it more robust.

The salt is provided by the users so that they can be rest assured of the security of their information. The Information to be secured is first serialized, to generate a storable representation of the information. Next an Initialization Vector (IV) is created to give alternative seed to the encryption routine. The IV here is randomly generated by the Computer to initialize the CBC (Cipher Block Chaining) Mode as illustrated in figure 4.1.

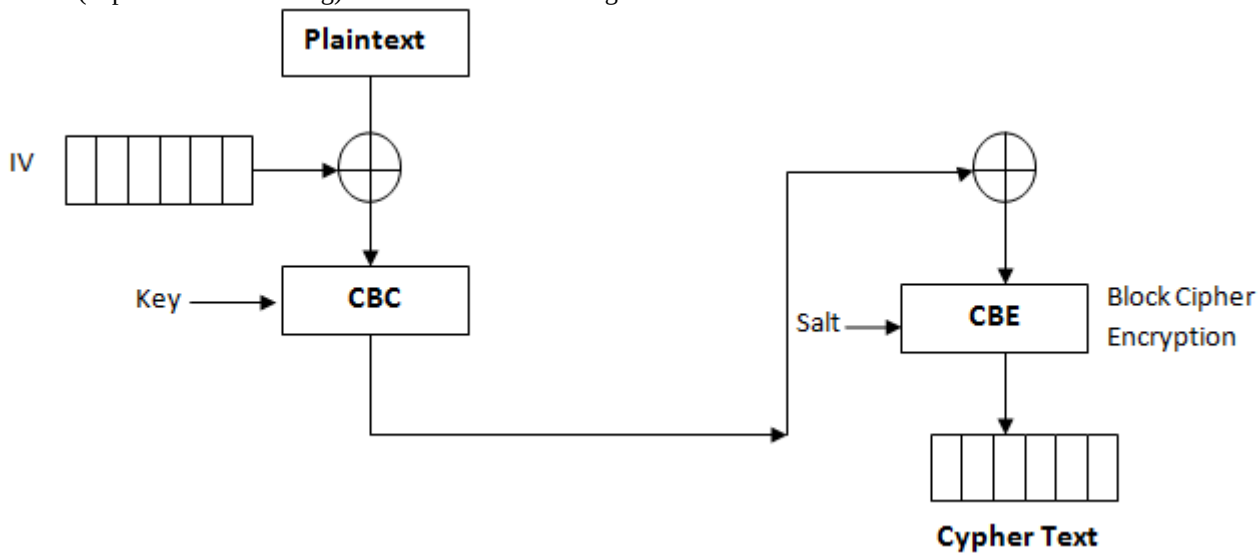

Fig. 4.1 CBC Encryption Mode

The Output (Cyphertext) is appended with a $2y$31$ String. The appendage will not only nullify Oracle padding attacks by confusing the Oracle, but will also deceive the hackers into thinking that the encryption is a blowfish Algorithm. It will take the Hacker time, energy and resources to discover (if ever he can) that the encryption is an AES-256, even at that; it will take over a million years to crack the AES-256 encryption. The Algorithm is as follows:

```
<?php
Define ('ENCRYPTION_KEY', $user_salt);
Function mc_encrypt ($encrypt, $key){
```



```
    $encrypt = serialize ($encrypt);
    $iv = mcrypt_create_iv(mcrypt_get_iv_size(MCRYPT_RIJNDAEL_256,
          MCRYPT_MODE_CBC), MCRYPT_DEV_RANDOM);
    $datacrypt = mcrypt_encrypt(MCRYPT_RIJNDAEL_256, $key, $encrypt,
            MCRYPT_MODE_CBC), $IV);
    $code_it = base64_encode ($datacrypt).'/'.base64_encode ($iv);
    $deciv = '$2y$32$'. $code_it;
    Return $deciv;
}
```

4.2 The Public Key Encryption (Proposed)

The engine of the public key encryption is the Blowfish using salt type $2y$ and hacker slow number of 22. BCrypt is a one-way Hash Algorithm, meaning that it cannot be decrypted. The Encryption uses a 62-digit randomly generated salt to ensure the cyphertext is undecodable. However, after the encryption, the Cyphertext is subjected to PHP Ltrim() function to cut off the harbinger $2y$22$ which shows the hackers the type of Algorithm used. So by cutting off this $2y$22$, the Hackers will not be sure of the hash algorithm used. Who knows, they might be fooled forever. It is impractical to decrypt a Cyphertext without knowing the algorithm that produced it.

```
 <?php
//first we declare the Algorithm as a function for reusability sake.
// the function requires two inputs: the Value to encrypt and the rounds  number of hardness.
Function code_public($input_Val, $rounds){
        $salt = " ";
//the declared salt is filled with the output of 22 random Numbers coined from Alphanumerics
        $salt_chars = array_merge(range('A','Z'),range('a','z'),range(0,9));
         For ($i=0; $i<22;$i++){
                                    $salt. = $salt_chars[array_rand($salt_chars)];
                                  }
//the First 'Deceive' step is taking by reversing the PlainText before encrypting it.
$rev = strrev($input_val);
$hash = crypt($rev, sprint('$2y$22$', $rounds).$salt);
//the 'Decive' Output is done by trimming off the Prefix indicator of the Algorithm.
$deciv = ltrim($hash, $2y$22$);
Return $deciv;
}
```

It is not unreasonable to believe that no encryption, no matter how salty it tastes, will remain undecipherable forever, so long as the algorithm is know; it is only a matter of time before the malicious cryptanalysts will see through it. That's why we have chosen the 'Deciv Algorithm'

The *Deciv algorithm* cannot be fathomed by automatic machines, only human reasoning might decode the deception, yet it must require extra- reasoning and ultra-high amplitude of perception to achieve that, in fact the tendency approaches zero. Meanwhile nowadays, only automated systems are used such as Brute Force, Rainbow table, The Padding Oracle and others; therefore this algorithm is one of the most brute resistant encryption ever to be used by cryptanalysts and hackers.

V. CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

This research paper haven considered the various associated security and privacy issues provides a cryptographically secure cloud algorithm tagged "Deciv Algorithm" which service providers, organizations or consumers could implement as a cryptographically- secure SaaS or PaaS cloud when outsourcing meaningful data, applications, and infrastructure to a virtualized cloud environment. The security of the cloud infrastructure entails protecting the cloud data from unauthorized access, preventing malicious program from corrupting the virtual resource and ensuring the secure cloud data remains unintelligible to any unauthorized access or intrusion by malicious users. Therefore, the research aim of designing a cryptographically secure cloud apps solution which would effectively hide meaningful user data from all external parties to a virtual network- even from the service provider have been achieved. The system if deployed or algorithm adopted, promises to improve the existing state of data privacy, and security of cloud data and application environment.

5.2 Recommendation

It is recommended that legislative bodies across the globe follow the internationally known best practices to advance a strong legal framework on data protection with the clearly enhanced Service Level Agreements (SLA) as already in place across Europe and America.

Developers should demystify cloud to relief fears to its adoption by deploying or developing some dependable and trusted user controlled cryptographically- secure PaaS or SaaS cloud, like that depicted by the novel algorithm advanced in this research.

5.3 Directives for Further Research

It is recommended that the follow-up of this research be conducted with the analysis of how to handle security, privacy and forensic issues associated with the hypervisor, multitenant or virtualized environment. The outcome would help enterprises and individuals gain assurance around their cloud computing provider's internal controls and security.
Finally, researches on appreciating the role of virtual world security as a strategy for securing information systems (IS) in a global project. This could also carry along the focus on the cloud computing as an IT strategy for global IS project management.

REFERENCES

[1] Al Beshri, A. M. (2013) *Outsourcing data storage without outsourcing trust in cloud computing*. PhD thesis, Queensland University of Technology. Available online at http://eprints.qut.edu.au/61738/ (Accessed: June 05, 2014)

[2] Boss et al. (2007). *Cloud Computing*: High Performance On Demand Solutions (HiPODS). Version 1.0, Available online at http://www.ibm.com/developerworks/websphere/zones/hipods/ (Accessed: 20 May 2011).

[3] Friedl, S. (2004) *An Illustrated Guide to Cryptographic Hashes*. Unixwiz.net Tech. Available online at http://www.unixwiz.net/techtips/iguide-crypto-hashes.html (Accessed: June 16, 2014)

[4] Graham, R. D. (2011). "Password cracking, mining, and GPUs". erratasec.com. Retrieved 17 August 2011.

[5] Hellman, M. E. (1980). *A cryptanalytic time-memory trade-off.* Information Theory, IEEETransactions on (Volume: 26, Issue: 4) July, 1980

[6] ISACA (2009). Cloud Computing: Business Benefits With Security, Governance and Assurance Perspectives: Emerging Technology White Paper. Available online at http://www.isaca.org/...Center/.../Cloud-Computing-28Oct09-Research.pdf Accessed: 08 June 2011

[7] Ivan (2010). Brute-force attack Available online at http://en.wikipedia.org/wiki/Bruteforce_attack (Accessed: June 14, 2014).

[8] Kelsey et al (1997). RC2. Available online at http://en.wikipedia.org/wiki/RC2 (Accessed: June 14, 2014)

[9] Kingsley-Hughes, A. (2008). *Encryption busted on NIST-certified Kingston, SanDisk and Verbatim USB flash drives* Available online at http://blog.beacontechsolutions.com/encryption-busted-on-nist-certified-kingston-sandisk-and-verbatim-usb-flash-drives/ (Accessed: June 20, 2014)

[10] Paar, C.; Pelzl, J. & Preneel, B. (2010). *Understanding Cryptography: A Textbook for Students and Practitioners*. Springer; July 8, 2010. ISBN 978-3642041006. Available online at http://www.amazon.com (Accessed: June 14, 2014).

[11] Robert, M. O. (1997). *Encryption and the first amendment*. Virginia Journal of Law and Technology. Available online at http://www.vjolt.net/vol2/issue/vol2_art1.pdf (Accessed: June 18, 2014)

[12] Ristic (2010). Internet SSL Survey 2010 Black Hat USA 2010. Availablee online at https://media.blackhat.com/bh-us-10/presentations/Ristic/BlackHat-USA-2010-Ristic-Qualys-SSL-Survey-HTTP-Rating-Guide-slides.pdf (Accessed: August 02, 2014)

[13] www.wikipedia.com/cryptoanalysis/attacks, August 2013

