iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 5 **Issue:** IV **Month of publication:** April 2017

DOI: http://doi.org/10.22214/ijraset.2017.4238

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)

Self Destruction of Data using Active Storage

Ashwini Kakade[1], Dinesh Gawande[2], Roshan Thakur[3]
[1]PG Student,[2,3]Asst. Professor of CSE Dept. DBACER, Nagpur, India

Abstract*: Nowadays internet is the strongest way to connect the people. Internet encompasses banking, messaging, sharing data social networking etc. By developing cloud technology maximum applications are using cloud interface, and people usually trust all the services provided by cloud, users can store personal information on cloud. Later that information can be cached, copy, archived by the CSPs' (Cloud Service Provider). Self destructing data is considering as spy proof future of the internet, by using this data is like a time bomb on the network, it means that after certain time interval. Primarily self destruction of data was introduced on DHT (Distributed Hash Table), since low cost Sybil attack on DHT are the problems. Self destructing data is now on active storage framework (SeDas) because it giving higher security and performance. SeDas is having a proof concept that performance is improved by 60 to 72 percent but it deals with text data only, no multimedia data is considered for this system. Also uploading and downloading of the file having very poor performance especially with the large files. This paper is going to address these two problems so that privacy can be achieved in public network.

Keywords*: Self Destructing Data, Data Privacy, Active Storage, Cryptography, Network Security.

I. INTRODUCTION

Storage is cheap, Data lives forever [2], The Internet never forgets. These truths thoroughly affect the interactions between people and modern computing system. Self destruction of data is considered spy proof future of the Internet. By using self destructing data privacy can be maintained in the public network. Self Destruction of data goal is use to destroy data after certain period of time [1], apart from where the data is stored or archived, in spite of technology that may make such deletion challenging. Consequently, such systems like self destruction prevent the retrieval of "old" data that is past its useful life. Self destruction is implemented by encrypting data with a key and then retrieving the information needed to reconstruct the decryption key with one or more third parties. Assuming that the key reconstruction information disappears from the retrieval with trust from third parties at the intended time, encrypted data will become permanently unreadable : (1) even if an attacker access a copy of the encrypted data and the user's cryptographic keys and passphrases after the timeout, (2) without the user or user's agent taking any precise action to delete it, (3) with no need to alter any stored or archived copies of that data, and (4) without the user relying on secure hardware.

A. Self-Destructing Data System

To Control over data lifetime will become more and more important as more public and private activities are captured in digital form, whether in the cloud or on personal devices. After certain period of time Self destructing data systems can help users to preserve some control, by ensuring that data becomes permanently unavailable. The self-destructing data system in the Cloud environment should meet the following requirements: i) How to destruct all copies of the data at the same time and make them unreadable in case the data is out of control? A local data destruction approach will not work in the Cloud storage because the number of backups or archives of the data that is stored in the Cloud is not known, and some nodes containing the backup data have been offline. The clear data should become enduringly unreadable because of the loss of encryption key, even if an attacker can retroactively get hold of a copy in its original condition; ii) User cannot perform delete action, or any arbitrator storing that data; iii) No need to modify any of the stored or archived copies of that data; iv) No use of secure hardware but support to completely erase data in Hard Drives and Storage Drives, respectively. As nowadays people getting more dependent on the Internet and Cloud technology, need to keep their private data secure and safe. First when data is being transformed, processed and stored by the current node or network node must cache, copy or archive it. These copies are important for systems and the network. However, people are unknown about these copies and cannot control them, so these copies may getaway their privacy. And second their privacy also can be leaked via Cloud Service Providers (CSPs') carelessness, hackers' intrusion or some legal actions. These problems present dreadful challenges to protect people's privacy [1]. Limitations of the existing system:

1) SeDas system only deals with text data, no multimedia data is considered.
2) Uploading and downloading of the large files having some performance issue.

©IJRASET: All Rights are Reserved

II. LITERATURE SURVEY

The goal of creating data that self-destructs or vanishes automatically after it is no longer useful. Moreover, it should do so without any explicit action by the users or any party storing or archiving that data, in such a way that all copies of the data vanish simultaneously from all storage sites, online or offline. More generally, self-destructing data is broadly applicable in today's Web centered world, where users' sensitive data can persevere "in the cloud" indefinitely (sometimes even after the user's account termination). With self-destructing data, users can regain control over the lifetimes of their Web objects, such as personal messages on Facebook and documents like Google Docs [3]

Self destructing system can enduringly delete data after certain period of time:

A. Even if an attacker can retroactively get an unspoiled copy of that data and any relevant persistent cryptographic keys and passphrases from before that timeout, possibly from stored or archived copies.
B. Without the use of any explicit delete action by the user or the parties storing that data.
C. Without needing to modify any of the stored or archived copies of that data.
D. Without the use of secure hardware.
E. Without relying on the introduction of any new external services that would need to be deployed (whether trusted or not).

Self destruction of data was introduced by University of Washington author was Roxana Geambasu et al [2] in Vanish: Increasing Data Privacy using Self Destruction of Data, 2009, in this paper message automatically get "self-destruct" after a period of time and it was done over DHT (Distributed Hash Table). After that in 2009 itself Scott Wolchok et al [1] present two Sybil attacks against the vanish implementation, which supplies its encryption keys in the million-node Vuze DHT. These attacks work by continuously attacking the DHT and saving each stored value before it goes off. They can efficiently recover keys for more than 99% of Vanish messages. SeDas paper shows that the foremost cost of these attacks is network data transfer, not the memory usage as the Roxana expected, and that the total cost is two orders of magnitude less than Roxana estimated. While the consideration is potential defences, Roxana conclude that public DHTs like Vuze [2] probably cannot provide strong security for Vanish. In 2010 SafeVanish was introduced by Lingfang Zang et al. [4] SafeVanish is based on extending the length range of the key shares and applying the public-key cryptosystem, to multiply the hopping attack cost, including storage requirement and the network bandwidth, and to avoid the sniffing attack.

Though Vanish is an interesting approach but it is having important privacy problem. To address the hopping attack problem of Vanish discussed above, Lingfang Zang et al. [4] gives a new scheme, called SafeVanish. Hopping attack is one type of the Sybil attacks and it is addressed by extending the length range of the key shares to enlarge the attack cost significantly, and some enhancement on the Shamir Secret Sharing algorithm implemented in the Vanish system. Also, proposed an improved approach not in favour of sniffing attacks by way of using the public key cryptosystem to avoid sniffing operations.

Tang et al. [3] proposed FADE which is built upon standard cryptographic techniques and assuredly deletes files to make them unrecoverable to anyone upon revocations of file access policies. Wang et al. [4] utilized the public key based homomorphism authenticator with random mask technique to achieve a privacy-preserving public auditing system for Cloud data storage security and uses the technique of a bilinear aggregate signature to support handling of multiple auditing tasks. Perlman et al. [5] present three types of assured delete: ending time known at file creation, deletion of individual files as require, and formation of custom keys for classes of data. However, the use of P2P features still is the serious weakness both for Vanish and SafeVanish, because there is a precise attack against P2P methods (e.g., hopping attacks and Sybil attacks).

In addition, for the Vanish system, the endurance time of key realization is determined by DHT system and not convenient for the user. Based on active storage framework, Lingfang Zang et al. [1] propose a distributed object-based storage system with self-destructing data function. SeDas system combines a positive approach in the object storage techniques and method object, using OSD's data processing capabilities [6] to achieve data self-destruction. User can specify the time to live of the key of distribution and use the settings of stretched interface to export the life cycle of a key, allowing the user to manage the subjective life-cycle of private data.

The whole scenario of the literature survey is given in a table 1 with key points. Key points are describing the resources and features that are hold by respective papers.

©IJRASET: All Rights are Reserved

Volume 5 Issue IV, April 2017
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

TABLE I Literature Survey

Sr. No	IEEE Paper Title	Year of Publish	Key Points
1	SeDas: A Self-Destructing Data System Based on Active Storage Framework [1]	JUN 2013	Active Storage, Cloud Network, Shamir Key Distribution, TTL (time to live), Metadata Server.
2	Vanish: Increasing Data Privacy with Self-Destructing Data [2]	AUG 2009	DHT, P2P, Client based app^n, VDO (Vanish Data Object), Locator.
3	Defeating Vanish with Low-Cost Sybil Attacks Against Large DHTs [3]	SEP 2009	DHT, P2P, Advance hopping implementation.
4	SafeVanish: An Improved Data Self-Destruction for Protecting Data Privacy [4]	NOV 2010	DHT, P2P, Hopping attack, expanded range of key shares.

III. IMPLEMENTATION DETAILS

A. *System Architecture*

The implementation of this system encompasses various modules into a single architecture, ensuring encryption and decryption, key distribution and sharing and active storage. The following diagram shows system architecture of the self destructing data system using active storage. The self destructing data system consists following steps:

1) Sender (Message)
2) Message Encryption
3) Key slicing
4) Key Distribution (on active storage nodes)
5) Key Integration / Reconstruction
6) Decryption (Message with reconstructed key)

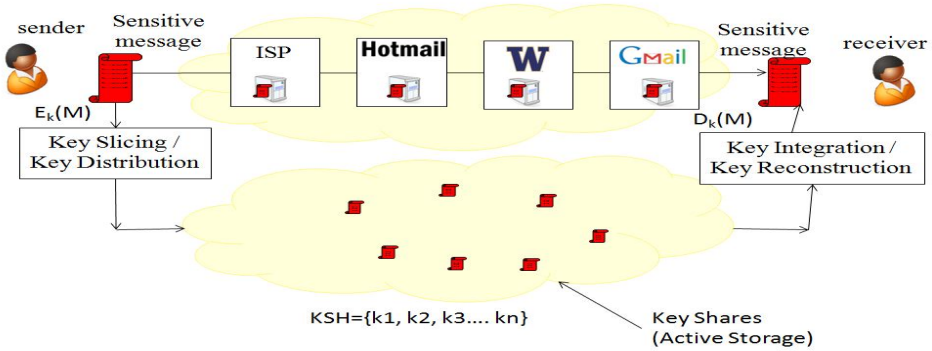


Fig. 1. System Architecture

There are total five functions that are being under consideration. And these functions are also being considered as the modules of the project

Encryption and decryption process can be done by using any standard algorithm. For better consideration as we have key size into an account we can go for AES (Advance Encryption Standards) [7] [9].

The next part is “key slicing”. And it is totally on Shamir Secret sharing algorithm [8]. Up till now there is no challenge for this algorithm.

Shamir secret sharing is based on (k, n) threshold scheme, and the formulae are:

N= 2K -1 is the robust system.

K= (N -1) / 2 to calculate k for the system. Where,

N is total number of key shares,

K is minimum number of key shares to reconstruct key.

©IJRASET: All Rights are Reserved

Volume 5 Issue IV, April 2017
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

Next module is the key distribution on active storage. For our system we just required a key storage system that works on the active storage.

The next part comes into account is the reconstruction of the keys that we do need at the receiver side. At the receiver end time out will be checked first, it is timeout then nothing to do, if there is no timeout then message should be decrypted with the specified keys.

To implement these modules we need some components, and they are follows:

Firefox extension

Messaging media

Active storage

B. Mathematical Model

Let S be the system which provides a self-destructing message using active storage.

S={M,E,K,KS,KD,KI,RK,D,KSH,Er|f1,f2,f3,f4,f5}

M - Message

E - Encryption of Message

KS – Key Slicing

KD – Key Distribution

KI – Key Integration / Reconstruction

RK – Reconstructed Key D – Decryption of Message

KSH – Distributed key shares on active storage Er – Error set in output

Set Theory:

KSH = {k1, k2, k3.....kn}

Calculated by (1)

Er = {Er1, Er2, Er3.....Ern}

Functions:

f1 - Message Encryption with a key

f2 - Key Slicing

f3 – Key Share Distribution

f4 – Key Integration / Reconstruction

f5 – Decryption of message with a reconstructed key

C. Data Flow Diagram

The data flow diagram is basically a diagram which describes how the data is flow and processed inside a system. The data flow diagram (DFD) is a visual representation of data flow and processing of a system.

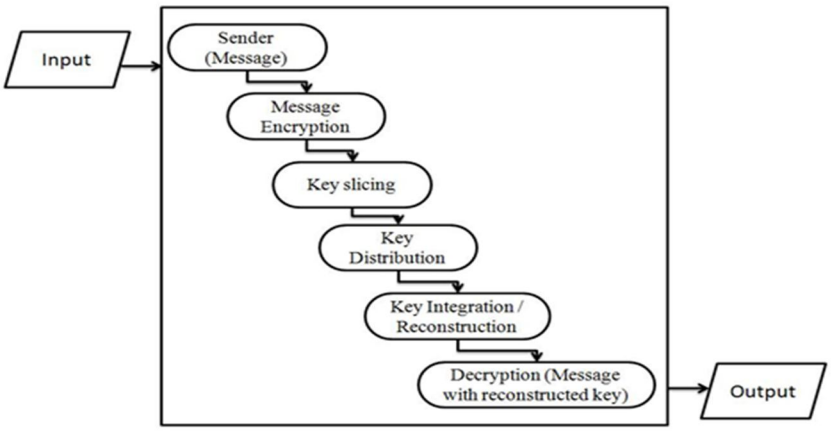


Fig 2. Level of DFD diagram

©IJRASET: All Rights are Reserved

D. Encryption and Decryption

The need of AES is only because of key sizes that we get from AES algorithm that is 128, 192, 256 bits. If we use DES here we get only 56 bit key [12]. If we used 56 bit key then for next module of key sharing we get relatively very small key, therefore for key sharing purpose we need AES algorithm, also its is 1021 times faster and secure than 3DES[11].

Algorithm 1: AES Encryption

```
byte state[4,Nb]
state = in
AddRoundKey(state, keySchedule[0, Nb-1])
for round = 1 step 1 to Nr–1  {
SubBytes(state)  ShiftRows(state)
MixColumns(state)
AddRoundKey(state,keySchedule[round*Nb,(round+1)*N b-1])
    }
SubBytes(state)
ShiftRows(state)
AddRoundKey(state, keySchedule[Nr*Nb, (Nr+1)*Nb-1])
out = state
```

Algorithm 2: AES Decryption

```
byte state[4,Nb]
state = in
AddRoundKey(state, keySchedule[Nr*Nb, (Nr+1)*Nb-1])
for round = Nr-1 step -1 down to 1 {
InvShiftRows(state)
InvSubBytes(state)
AddRoundKey(state,keySchedule[round*Nb,(round+1)*N
b-1])
InvMixColumns(state)
  }
InvShiftRows(state)
InvSubBytes(state)
AddRoundKey(state, keySchedule[0, Nb-1])
out = state
```

Where,

Nb – Number of columns in the State

For AES, Nb = 4

Nk – Number of 32-bit words in the Key

For AES, Nk = 4, 6, or 8

Nr – Number of rounds (function of Nb and Nk)

For AES, Nr = 10, 12, or 14

E. Key Slicing and Distribution

In cryptography [16], secret sharing is an approach of distributing a secret shares in a group of nodes (participants), each of which given a share of the secret. The secret can only reconstruct when the threshold numbers of shares are collected together; any individual share is of no use. It gives tight control and removes single point vulnerability and individual key share holder cannot change/access the data.

Goal is to divide some data D (e.g., the safe combination) into n pieces D1, D2….Dn in a way that:

1) Knowledge of any k or more D pieces makes D easily computable.

2) Knowledge of any k -1 or fewer pieces leaves D completely undetermined (in the sense that all its possible values are equally likely).

©IJRASET: All Rights are Reserved

This scheme is called (k, n) threshold scheme. If k=n then all shares are required together for reconstructing the secret.

F. Active Storage

Today, the active storage system has become one of the most important research domains of intelligent storage systems. There has been several hard works to incorporate active storage technology into the T10 OSD standard [5], in this area by considering data self-destruction, this system is proposing one open source secured storage network in which the data or key shares are to be stored on the different storage nodes. In order to achieve privacy, several changes have to be made as it is going to collaborate with browser plug-in. Therefore to understand the need of data self-destruct, instead of performing whole task of by single server (storage node) this system propose integration of several nodes together controlled by one single plug-in which may be present in both side of communication.

IV. EXPECTED RESULT

This paper proposes self-destructing data system based on active storage. Implementation is in process. Though this system is having traditional algorithms and familiar architectures but we are expecting more in terms of performance as browser plug-ins are part of it and trying to minimize the system overhead of encryption and decryption as AES algorithm is in consideration. Shamir secrete sharing algorithm having (k, n) threshold scheme increases the reliability of the system. In order to deal with security, we are fixing compatibility of browser and active storage.

The result of existing system is as follows in terms of encryption decryption overhead (in seconds) and file size (in MB)

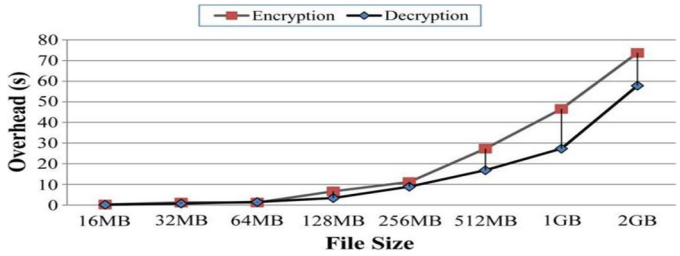


Fig 3. Comparison of overhead for encryption and decryption

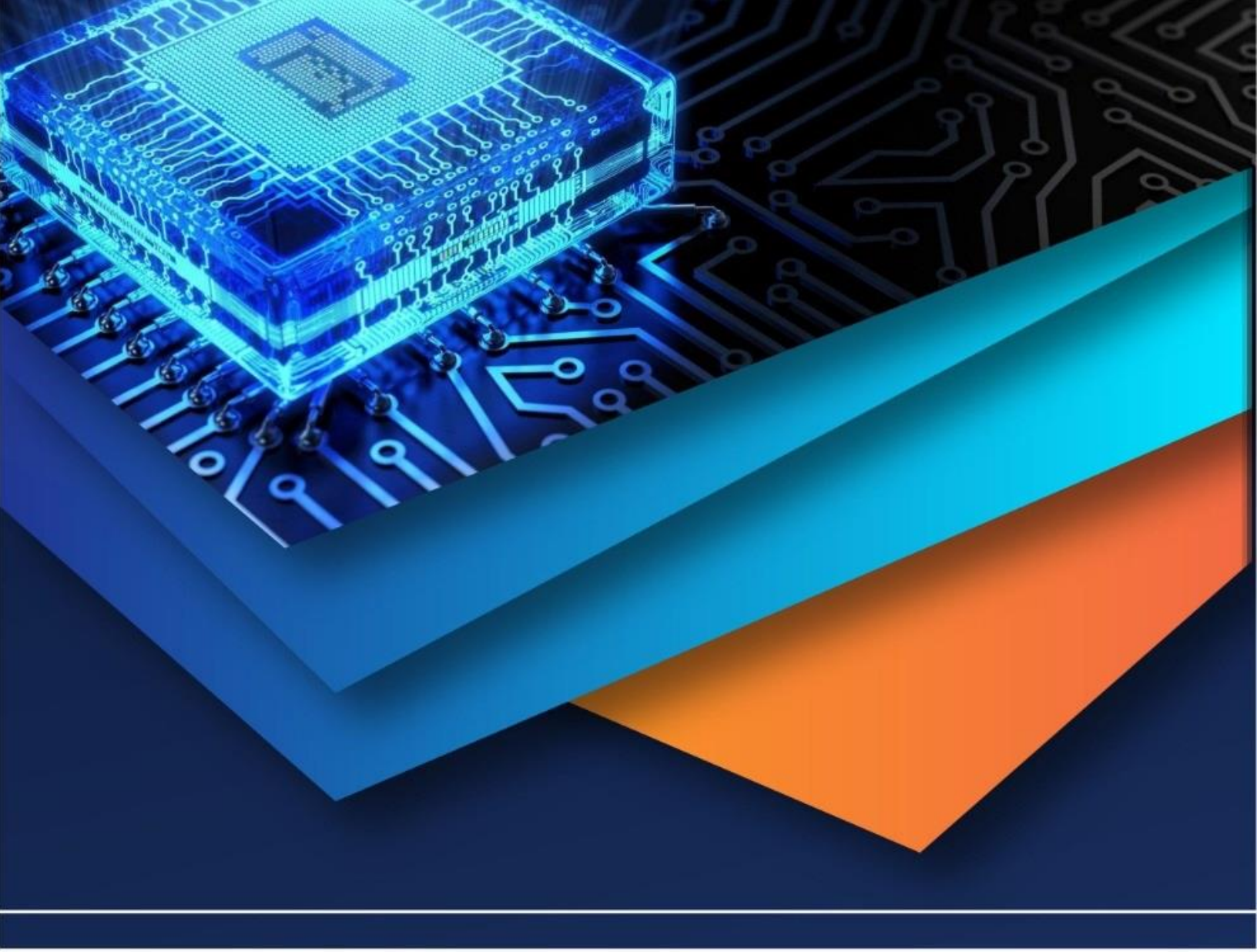
V. CONCLUSION

Data privacy has become extremely important in the Cloud environment. This is a new approach of defending data privacy from attackers, who can retroactively obtain data through various ways because of data lives forever. Also this is motivation for self destructing data system. This approach is using active storage framework for maximum advantage and it is expected that after successful implementation, performance should be more than existing systems. With this privacy can be achieved in a public network.

REFERENCES

[1] S. M. Metev and V. P. Veiko, Laser Assisted Microtechnology, 2nd ed., R. M. Osgood, Jr., Ed. Berlin, Germany: Springer-Verlag, 1998.
[2] J. Breckling, Ed., The Analysis of Directional Time Series: Applications to Wind Speed and Direction, ser. Lecture Notes in Statistics. Berlin, Germany: Springer, 1989, vol. 61.
[3] S. Zhang, C. Zhu, J. K. O. Sin, and P. K. T. Mok, "A novel ultrathin elevated channel low-temperature poly-Si TFT," IEEE Electron Device Lett., vol. 20, pp. 569–571, Nov. 1999.
[4] M. Wegmuller, J. P. von der Weid, P. Oberson, and N. Gisin, "High resolution fiber distributed measurements with coherent OFDR," in Proc. ECOC'00, 2000, paper 11.3.4, p. 109.
[5] R. E. Sorace, V. S. Reinhardt, and S. A. Vaughn, "High-speed digital-to-RF converter," U.S. Patent 5 668 842, Sept. 16, 1997.
[6] (2002) The IEEE website. [Online]. Available: http://www.ieee.org/
[7] M. Shell. (2002) IEEEtran homepage on CTAN. [Online]. Available: http://www.ctan.org/tex-archive/macros/latex/contrib/supported/IEEEtran/
[8] FLEXChip Signal Processor (MC68175/D), Motorola, 1996.
[9] "PDCA12-70 data sheet," Opto Speed SA, Mezzovico, Switzerland.
[10] A. Karnik, "Performance of TCP congestion control with rate feedback: TCP/ABR and rate adaptive TCP/IP," M. Eng. thesis, Indian Institute of Science, Bangalore, India, Jan. 1999.
[11] J. Padhye, V. Firoiu, and D. Towsley, "A stochastic model of TCP Reno congestion avoidance and control," Univ. of Massachusetts, Amherst, MA, CMPSCI Tech. Rep. 99-02, 1999.
[12] Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification, IEEE Std. 802.11, 1997.

©IJRASET: All Rights are Reserved



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)