International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)
Web Site: www.ijettcs.org Email: editor@ijettcs.org
Volume 6, Issue 5, September- October 2017 ISSN 2278-6856

A Prototype Implementation for Public Key Infrastructure Based on Transport Layer Security

Dr.P.Sumalatha[1], Dr. C.KrishnaPriya[2]

[1]Department of Computer Science and Technology, Sri Krishnadevaraya University, Anantapuramu, Andhra Pradesh, India

[2]Department of Computer Science and Technology, Sri Krishnadevaraya University, Anantapuramu, Andhra Pradesh, India

Abstract: ***Information security has to be given paramount importance in communication networks. Besides the ability to share documents, the networks should provide complete security measures in order to ensure that the documents are accessed by intended recipients with privacy and integrity. The common security requirements like authentication, confidentiality, privacy and non-repudiation are to be fulfilled in network. Public key infrastructure (PKI) has been around for digital signatures to ensure secure communications. The security primitives and components in the PKI can enforce trust among the communicating parties using digital signatures. In this paper we review the PKI and its applications in the real world besides implementing it practically. We have made experiments with a standalone prototype that demonstrates the proof of concept. The empirical results revealed that the PKI when implemented with care can provide fool proof security to communications over any network.***

Keywords: Digital signatures, public key infrastructure, TLS, certificate authority.

1. INTRODUCTION

Cryptography provides secure communications over various networks. Since the security of data plays an important role, the cryptography came into existence. The adversaries are using new means of breaking systems. Cryptography has two different ways of encryption mechanisms. They are private key encryption and public key encryption. The private key encryption is simple. In private key encryption Key exchange is the main problem involved. Because in private key encryption the sender and receiver use the same key for encryption and decryption processes respectively. So it is essential for the sender to send the key to receiver. As the key is sent in plain text format, it causes potential risk and hence, this kind of encryption mechanism cannot provide complete security. The private key encryption is also known as symmetric cryptography. Figure 1 visualizes the private key encryption and decryption mechanisms.

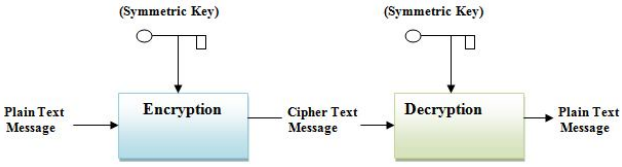

Figure 1 – Symmetric cryptography [1]

As can be seen in Figure 1, the symmetric key used by the sender for the process of encryption is used by the receiver for decryption. When the key sharing can be avoided, it would be more secure. Thus the public key cryptography came into existence.

The public key cryptography on the other hand follows different approach in which the key sharing is effectively avoided. In this kind of cryptography both the sender and receiver have a pair of keys. The pair consists of a private key used for decryption and a public key used for encryption. The public key is informed to others while the private key is kept confidential. The pair of keys is highly cohesive in nature. In other words, they are related. When encryption is done with public key of any participant, only private key of that participant can decrypt the encrypted message. Since the parties involved in communication have two keys, they can avoid key sharing problem. This kind of cryptography is known as asymmetric cryptography. Figure 2 illustrates the encryption and decryption mechanisms as part of public key cryptography.

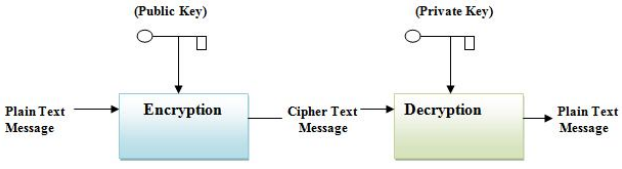

Figure 2 – Asymmetric key cryptography

As can be seen in Figure 2, it is evident that the sender of the message uses public key of the receiver in order to encrypt message. The encrypted message is sent to the intended recipient. The recipient will be able to decrypt it as the message was encrypted by his public key. This intuitive relationship between private key and public key makes this kind of cryptography more secure when compared with the private key cryptography. In fact this

kind of security is widely used in the real world. The security of public key cryptography depends on its mathematical complexity. It is proven that it is mathematically so complex and adversaries cannot afford to spend time to break it. However, the public key cryptography is broken easily when keys are stolen by or known to others. This problem can be overcome by using more sophisticated security infrastructure known as public key infrastructure (PKI).

1.1 Public Key Infrastructure

Public key infrastructure refers to a set of components that are involved in complete fool proof security in digital communications over private or public networks. Especially this is required in untrusted networks like Internet. PKI enables users to exchange data securely. Here digital certificates are used for authentication and the communication between two parties will be highly secure. The digital certificates provided by PKI play an important role in identifying parties involved in communication. The PKI uses public key cryptography along with digital certificates for user authentication. Thus it provides high security besides eliminating the need for sharing secret keys. A public key infrastructure has many components namely certificate authority (CA), a registration authority (RA), digital certificate, a certificate revocation list (CRL), an online certificate status protocol (OCSP), a digital signature, and the secure token (ST3). Figure 3 illustrates the general security mechanisms of PKI.

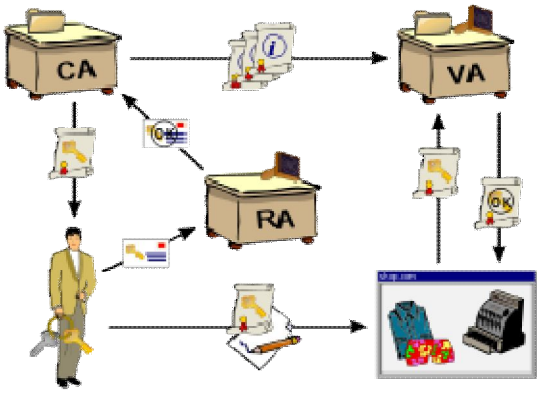

Figure 3 – Illustrates security mechanisms and components involved in PKI [2]

As can be seen in Figure 3, it is evident that the CA is responsible to issue and verify digital certificates to online users. RA will help CA for certificate verification. In fact it acts as verifier for CA. The VA is responsible to verify the validity of security certificate using a list of invalid certificates. Digital certificate contains identify information of a user along with binding it with public key. The CRL represents a list of certificates that have been revoked. OCSP verifies the certificate validity even when the certificate is subjected to tampering or revocation. Digital signature is the data that is used to prove the identity of a user. With all these components playing specific roles the PKI is made highly secure infrastructure that makes use of public key cryptography and digital certificates.

In this paper we studied the PKI and its components. We have implemented a PKI architecture using Java programming language. The remainder of the paper is structured as follows. Section II reviews literature pertaining to PKI and its evolution and applications. Section III resents our implementation of PKI and a prototype application. Section IV concludes the paper.

2. RELATED WORK

Diffie and Hellman [3] originally conceived the concept of public key cryptography while the Rivest, Shamir and Adleman [4] involved in designing RSA algorithm based on public key cryptography.

2.1 Evolution of PKI Standards

PKI is the result of evolution process. Initially the X.509 provided recommendations that enable to understand the formats and mechanisms for public key cryptography with digital signatures provided by CAs. However, it does not provide details information about the sub fields that are to be provided as part of digital certificates. The efforts of this standard resulted in PKI of X.509 version 3 with certificates besides certification revocation lists in version 2. Many revisions were witnessed before Internet PKI came into existence in the form of RFC 2459 [5]. In order to make use of the standard many algorithms came into existence. Later on PKI management protocols were subjected to number of iterations. RFC 2510 standard was proposed to specify message protocol that securely exchanges information between components of PKI [6]. Extended procedures were developed later that includes LDAP v3 for storing information [7]. Afterwards PKI repositories are used to hold certificates using FTP and HTTP protocols as specified in the RFC 2585 [8]. In [9] the working of PKI is described theoretically.

2.2 Applications of PKI

Public key infrastructure (PKI) provides complete secure solutions in communication networks. The PKI is used in various real time applications. Vic Patel et al. [1] applied PKI for air traffic management. Malan et al. [10] adapted PKI for effective key distribution in TinyOS. They used the infrastructure based on elliptic curve cryptography (ECC). The PKI with ECC has proved to give more security. Their experiments also proved that the ECC with PKI is a viable alternative. It is interesting here to know the memory size and the performance of this PKI when size of security key is changed. Figure 4 (a) and Figure 4 (b) show the results of the experiments made by Malan et al. [10].

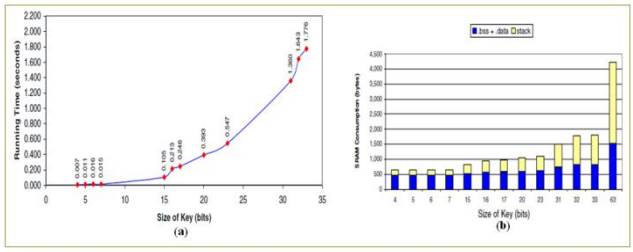

Figure 4 – (a) size of key vs. runtime (b) size of key vs. memory consumption [10]

As can be seen in Figure 4 it is evident that the runtime is increased when size of the key is increased. In the same fashion, the memory consumption is increased when the size of keys is increased. When the results are observed, the time taken and memory consumed can be viable at the time is less while the memory consumption to this extent is not much overhead to the system considering the amount of RAM being make available in the configuration of modern systems [10].

Smart and Muller [11] explored PKI for wearable public key infrastructure. Cristiano and Liu [12] made experiments on splitting public keys for making use of them in PKI. Thus they made it to be CA independent with smaller certificates and increased key size. Zhang et al. [13] implemented a public key infrastructure that is known as Anonymous and Certificates PKI (AC-PKI) that provide certificate less public keys to make the PKI simple and effective. Hao et al. [14] built fast authentication PKI for MANETs which makes the authentication process faster. These researchers focused on trusted computing technology illustrated in [15]. Toorani and Shirazi [16] implemented a light weight PKI for mobile networking environments. Their research has assumed significance as the mobile networks are resource constrained. Smith [17] illustrated the PKI for various companies that need end to end security. He also explored the challenges that arise while building a PKI infrastructure in various environments. Harn and Ren [18] proposed a generalized digital certificate as part of PKI for user authentication to make the key establishment procedure simple. For successful user authentication and establishing keys securely these researchers proposed integer factoring and discrete logarithm based protocols.

Comer, Singh and Vasudevan [19] proposed a hybrid scheme to implement PKI. The scheme combines the historical authentication mechanisms and external authority for improved effectiveness. Mehrasa et al. [20] proposed an algorithm that transmits partial private key thus eliminating certificates in PKI. They used identity based encryption for certificate less public key cryptography (CL-PKC). Vatra [21] proposed a new PKI for Romania. Lee et al. [22] proposed a hybrid PKI that is based on both certificates and identity based cryptography. Their PKI is known as unified PKI. With identity based approach it makes the usage of the PKI simple and effective. Benantar [23] provides the complete description of PKI and its real world usage. Coronado-Garc´ıa, Hern´andez-L´opez and P´erez-Legu´ızamo [24] explored Autonomous Decentralized System architecture for implementing PKI which is highly reliable in nature. Wang and Zhang [25] introduced the implementation of PKI that makes use of TLS (Transport Layer Security).

3. IMPLEMENTATION OF PKI

In this paper we implement PKI based on transport layer security. The implementation of this is done with a prototype application built in Java platform. Swing is used for pluggable look and feel UI that also exhibit MVC (Model View Controller) architecture. The functionality is made using Java language with security related API. The environment used to build the application includes a PC with 4 GB RAM, core 2 dual processor running Windows 7 operating system. The PKI implemented by us has the components such as certificate authority, certificate repository, registration authority, distribution system and PKI enabled applications. The overview of our PKI is illustrated in Figure 5.

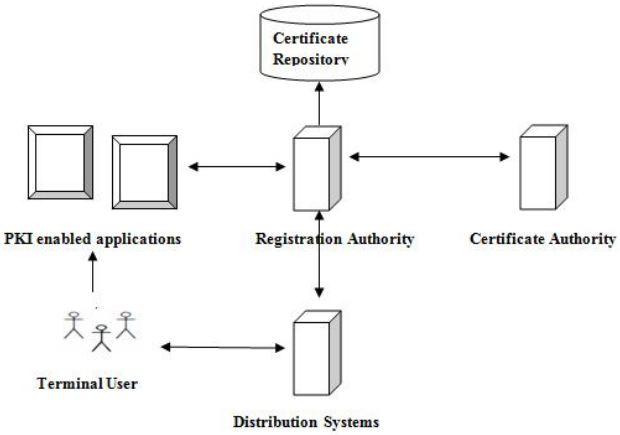

Figure 5 – Illustrates the overview of PKI Architecture

As can be seen in Figure 5, it is evident that the terminal users use the applications which are PKI enabled. The registration authority, certificate authority and certificate repository are used to have secure authentication of the users while using PKI enabled applications. The PKI is based on the transport layer security. For more information on transport layer security reader can visit the paper [25]. We used PKI API provided by Java programming language [26]. The API is shown in Table 1.

Table 1 – PKI API used in prototype application.

PURPOSE	API USED
Basic certification path	CertPath, CertificateFactory, CertPathParameters
Certification path validation	CertPathValidator, and CertPathValidatorResult
Certification path building	CertPathBuilder and CertPathBuilderResult
Certificates and CRL storage	CertStore, CertStoreParameters, CertSelector, CRLSelector

As can be seen in Table 1, the PKI is used from Java API. The certification path is illustrated in Figure 6.

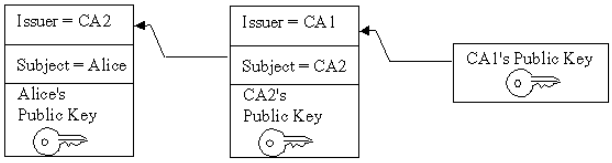

Figure 6 – Illustrates certification path [26]

In our prototype application Tomcat Server is used for secure communication. Web based clients are used for mutual authentication and communication. The TLS is used by configuring Tomcat. The server.xml file which is one of the important configuration files in Tomcat is modified in order to use PKI using TLS. The following changes are made to server.xml file.

```
<Connector port="9090"
maxThreads=" 150" minSpareThreads="25"
maxSpareThreads="75"
enableLookups="false" disableUploadTimeout="true"
acceptCount=" 1 00" debug="O" scheme="https" secure="true"
clientAuth= "true" sslProtoco 1= "TLS"
keystoreFile=" conf/tomcat.jks" keystorePass=" 4321"
keystoreType=" JKS"
truststoreFile=" conf/truststore .jks"
truststorePass="4321" truststoreType="JKS"/>
```

Listing 1 – Illustrates the configuration of TLS in server.xml

The certification path generation code is used to generate certification path which makes use of the standard java.security.cert.X509Certificate. The sample code is as shown in listing Figure 7.

```
// open an input stream to the file
FileInputStream fis = new FileInputStream(filename);
// instantiate a CertificateFactory for X.509
CertificateFactory cf = CertificateFactory.getInstance("X.509");
// extract the certification path from
// the PKCS7 SignedData structure
CertPath cp = cf.generateCertPath(fis, "PKCS7");
// print each certificate in the path
List<Certificate> certs = cp.getCertificates();
for (Certificate cert : certs) {
    System.out.println(cert);
}
```

Figure 7 – Code for certification path generation

Afterwards, fetching certificate chain from a key store needs to be done. This is done using CertificateFactory class in Java. The sample code is presented in Figure 8.

```
// instantiate a KeyStore with type JKS
KeyStore ks = KeyStore.getInstance("JKS");
// load the contents of the KeyStore
ks.load(new FileInputStream("./keystore"),
    "password".toCharArray());
// fetch certificate chain stored with alias "sean"
Certificate[] certArray = ks.getCertificateChain("sean");
// convert chain to a List
List certList = Arrays.asList(certArray);
// instantiate a CertificateFactory for X.509
CertificateFactory cf = CertificateFactory.getInstance("X.509");
// extract the certification path from
// the List of Certificates
CertPath cp = cf.generateCertPath(certList);
```

Figure 8 – Obtaining certificate chains from keystore

The certification path validation is performed using CertPathValidator. The sample code is as presented in Figure 9.

```
// create CertPathValidator that implements the "PKIX" algorithm
CertPathValidator cpv = null;
try {
    cpv = CertPathValidator.getInstance("PKIX");
} catch (NoSuchAlgorithmException nsae) {
    System.err.println(nsae);
    System.exit(1);
}
// validate certification path ("cp") with specified parameters ("params")
try {
    CertPathValidatorResult cpvResult = cpv.validate(cp, params);
} catch (InvalidAlgorithmParameterException iape) {
    System.err.println("validation failed: " + iape);
    System.exit(1);
} catch (CertPathValidatorException cpve) {
    System.err.println("validation failed: " + cpve);
    System.err.println("index of certificate that caused exception: "
            + cpve.getIndex());
    System.exit(1);
}
```

Figure 9 – Sample code for certification validation

As part of the PKI, we have implemented a prototype application that has provisions to end user to communicate with server using TSL that provide complete security in communications. The results revealed that our application is effective and the API used using Java programming language is able to support TSL which ensure fool proof security in PKI.

4. A STANDALONE PKI PROTOTYPE

In this section we provide our standalone PKI prototype which has been built with graphical user interface. The application demonstrates PKI infrastructure that facilitates key generation, digital signature generation, and verification and so on. The key pairs are generated using DSA algorithm. The DSA algorithm also supports generation of digital signatures and verifying them later. All these operations are supported by the prototype which is part of public key infrastructure. KeyPairGenerator class of java.security package is used to build public key infrastructure. The KeyPairGenerator class is capable of supporting algorithms presented in Table 2.

Table 2 – Algorithms supported by KeyPairGenerator

ALGORITHM	PURPOSE
DiffieHellman	Generates key pairs for algorithm named “DiffieHellman”.
DSA	Generates key pairs for the algorithm named Digital Signature Algorithm (DSA).
RSA	Generates key pairs for the algorithm named RSA.
EC	Generates key pairs for the algorithm named “Elliptic Curve”.

In this prototype we used DSA algorithm with 1024 key size. The DSA algorithm is the standard for digital signatures. The algorithm is part of Java security API which lets applications make use of PKI. The prototype application has utilized this standard for public key infrastructure.

Figure 10 – Key and digital signature generation

As can be seen in Figure 10, it is evident that two keys are generated namely private key and public key. The sender as shown creates digital signature after giving some identity information to the system. Figure 11 shows the successful verification of digital signatures.

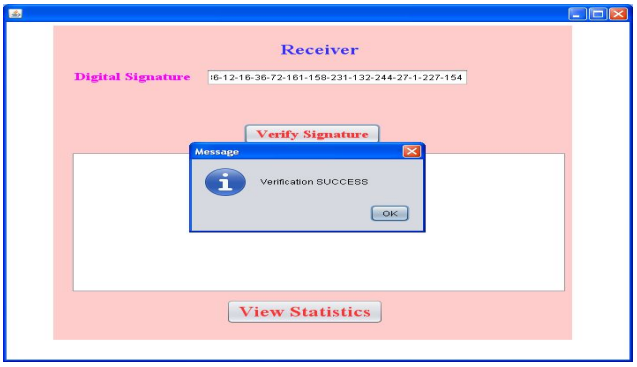

Figure 11 – Illustrates digital signature verification succeeded as digital signature is genuine

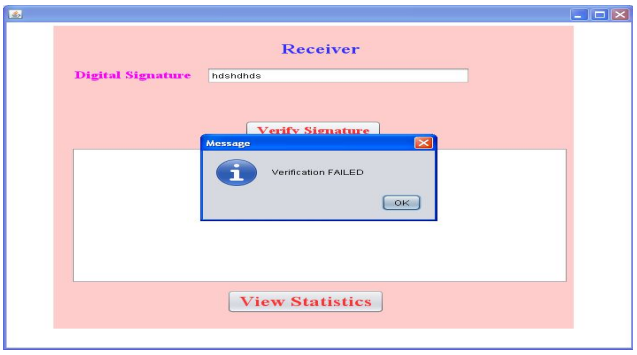

Figure 12 – Illustrates key verification failed as the signature is invalid

The standalone prototype has demonstrated the public key infrastructure with a simple user-friendly simple Java application. This infrastructure can be used in enterprise applications as well where it functionalities are segregated into trusted center, sender and receiver.

5. CONCLUSION

We studied cryptography which is in the form of symmetric cryptography and asymmetric cryptography. Then we also studied PKI systems in the world and their real time usage. The PKI is capable of using digital certificates that guarantee authentication of users who involve in transactions of real world applications. We built a framework for PKI and implemented it using Java programming language. The PKI API provided by Java is used in order to achieve the secure communication. We used Tomcat server to configure TSL for PKI authentication and other security mechanisms. We built a prototype application that demonstrates the proof of concept. We also built a standalone prototype that is user-friendly and simple to shows the efficiency of the public key infrastructure. The empirical results revealed that the application is useful to have secure communications without the need for sharing security keys explicitly.

References

[1] Vic Patel and Tom McParland. (2001). PUBLIC KEY INFRASTRUCTURE FOR AIR TRAFFIC MANAGEMENT SYSTEMS. IEEE. p1-7.
[2] JohnpelQuingua. (2013). Public Key Infrastructure. Available:http://www.johnpelquingua.com/introduction-public-key-infrastructure/ Last accessed 5th Mar 2013.
[3] Diffie, W. and Hellman, M. E.,(1976) New Directions in Cryptography. IEEE Transactions on Information Theory,22 , pp. 644-654.
[4] Rivest, R., Shamir, A. and Adleman, L.,(1978) A Method for Obtaining Digital Signatures and Public Key Cryptosystems. Communications of the ACM, 21, pp. 120-126.
[5] Housley, R., Ford, W., Polk, W., and Solo, D., (1999) RFC 2459 "Internet X.509 Public Key Infrastructure Certificate and CRL Profile"
[6] Adams, C., Farrell, S., (1999) RFC 2510, “Internet X.509Public Key Infrastructure Certificate Management Protocols".
[7] Myers, M., Adams, C., Solo, D., and Kemp, D.,(1999) RFC 2511, "Internet X.509 Certificate Request Message Format".
[8] Housley, R., and Hoffman, P.,(1998) RFC 2585, "Internet X.509 Public Key Infrastructure Operational Protocols: FTP and HTTP".
[9] Indrasenareddy,Bhat and Rajiv,(2011). Establishment of Public key Infrastructure for Digital Signatures, Vol.2,No.6,pp 33-43.
[10] David J. Malan, Matt Welsh, Michael D. Smith. (2004). A Public-Key Infrastructure for Key Distribution in TinyOS Based on Elliptic Curve Cryptography. IEEE. p71-80.
[11] N.P. Smart and Henk L. Muller. (2000). A wearable public key infrastructure (WPKI). IEEE. p127-133.

[12] Salvatore Cristiano and Faye F. Liu. (n.d). ON SPLITTING PUBLIC KEYS FOR THE PUBLIC KEY INFRASTRUCTURE. IEEE. p1-4.
[13] Yanchao Zhang, Wei Liu, Wenjing Lou, Yuguang Fang and Younggoo Kwon. (2005). AC-PKI: Anonymous and Certificateless Public-Key Infrastructure for Mobile Ad Hoc Networks. IEEE. 0 (0), p3515-3519.
[14] Liming HaoXiehua Li Shutang Yang Songnian Lu. (2006). Fast Authentication Public Key Infrastructure for Mobile Ad Hoc Networks Based on Trusted Computing. IEEE. 0 (0), p1-4.
[15] TCG, https://trustedcomputinggroup.org
[16] Mohsen Toorani and Ali AsgharBeheshtiShirazi. (2006). LPKI – A Lightweight Public Key Infrastructure for the Mobile Environments. IEEE. 0 (0), p162-166.
[17] RICHARD GUIDA, ROBERT STAHL, THOMAS BUNT, GARY SECREST, AND JOSEPH MOORCONES Johnson & Johnson. (2004). Deploying and Using Public Key Technology: Lessons Learned in Real Life. IEEE. 0 (0), p67-71.
[18] LeinHarn and Jian Ren. (2011). Generalized Digital Certificate for User Authentication and Key Establishment for Secure Communications. IEEE. 10 (7), p2372-2379.
[19] D. Comer, P. Singh and S. Vasudevan. (2012). Effective border gateway protocol protection that does not require universal adoption of a public key infrastructure. IEEE. 1 (5), p217-228.
[20] SeyedehneginMehrasa, Nazrul M. Ahmad, AlirezaKhorram, Asrul H. Yaacob. (2011). Secure Partial Secret Key Issuing in Certificateless Public Key Infrastructure. IEEE. 0 (0), p79-84.
[21] NicusorVatra. (2010). Public Key Infrastructure for Public Administration in Romania.IEEE. 0 (0), p481-484.
[22] Byoungcheon Lee. (2010). Unified Public Key Infrastructure Supporting Both Certificate-based and ID-based Cryptography. IEEE. 0 (0), p54-61.
[23] M. Benantar. (2001). The Internet public key infrastructure. IBM. 40 (3), p648-665.
[24] Luis Carlos Coronado-Garc´ıa, Carlos Hern´andez-L´opez, Carlos P´erez-Legu´ızamo. (n.d). A Uniqueness Verifying Public Key Infrastructure Based on Autonomous Decentralized System Architecture. IEEE. 0 (0), p1-6.
[25] Ke-feng Wang and Zhi-hong Zhang. (2010). Design and Implementation of a Safe Public Key Infrastructure. IEEE. 0 (0), p298-301
[26] Ke-feng Wang and Zhi-hong Zhang. (2010). Design and Implementation of a Safe Public Key Infrastructure. IEEE. 0 (0), p298-301.

AUTHORS

Dr. P. Sumalatha received her M.Sc(Computer Science) from Sri Krishnadevaraya University, Anantapuramu, A.P., India, in 2007..She did her Ph.D in Computer Networks in the Department of Computer Science and Technology from Sri Krishnadevaraya University, Anantapuramu, A.P., India. Her current research interest includes Computer Networks, Network Security.

Dr. C. KrishnaPriya received her Master of Computer Applications from Sri Krishnadevaraya University, Anantapuramu, A.P., India, in 2007, M.Tech (IT) from KSOU, Mysore, Karnataka in 2011.She did her Ph.D in Computer Networks in the Department of Computer Science and Technology from Sri Krishnadevaraya University, Anantapuramu, A.P., India. Her current research interest includes Computer Networks, Network Security and Intrusion Detection.