ISSN 2250-3153

Secure File Transfer Using USB

Prof. R. M. Goudar, Tushar Jagdale, Ketan Kakade, Amol Kargal, Darshan Marode

Computer Department, Maharashtra Academy of Engineering, University of Pune,
Ganeshkhind, Pune, India

Abstract- Universal Serial Bus (USB) devices are high transmission speed external devices. USB is the speediest external device available today. The access to USB is very simple, convenient and fast compared to other external devices such as CD, DVD, Floppy drive etc. Thus USB have become most popular interface standard for hardware connection. But from security point of view, USB devices lacks security as no user is restricted to access USB device. If we restrict the use of USB devices, then the data access through external devices would be as chaotic as before. Thus a system which would provide fast access as well as security is required. In this paper, we propose a system which deals with mutual authentication and key agreement in order to provide security for access through USB devices.

Index Terms- Authentication, USB, Cryptography

I. INTRODUCTION

USB devices are well known for the speed they provide while accessing data through them. The USB devices are easy to connect. Thus USB devices provides with speed as well as easiness. This high transmission speed devices are more convenient than any other external devices. But USB device lacks security. There is no authentication process provided by USB. Thus USB devices are very insecure.

USB can be considered as an unprotected gate, through which any data can be transferred without authentication. This is a serious problem when it comes to security of important data stored in the computers. One way to avoid this is to block all USB ports. But then no user will be able to access the information through USB ports. This will again cause the data transfer to be slow as it was in earlier days. Even we cannot ignore the security issues of USB. In many applications such as business or bank applications, the data should be securely stored in PC's and should be accessible to valid users only. Also the system should be fast enough to cope with today's competitive world.

Thus system should provide USB access to valid users but should also restrict unauthorised users to transfer the files stored in PC's by providing mutual authentication. The data stored in USB devices is first encrypted and then stored in USB devices. This is useful to secure files in case the USB device is lost or stolen malevolently. The data is thus secure due to encryption.

For key agreement, we used the key exchange agreement proposed by Rivest, Shamir and Adleman. The objective of using RSA for key exchange is that the key generated by the Authentication server (AS) must be transferred to client safely when the two sides try to communicate. Subsequently, the keys generated by RSA algorithm can be used for encrypting a message for transmission. However, this protocol is subject to man-in-the-middle attacks. Suppose an attacker exists between the sending end and the receiving end. The attacker poses as the sending end to transmit a public key exponent and modulus to the receiving end; without identity confirmation, the receiving end cannot ensure that this message is sent from the sending end. Many scholars have recently presented solutions to this problem. The most widely adopted of these is that of using a user password for identity confirmation on the two sides. We developed a system based on password verification that combined Schnorr's digital signature scheme and the RSA algorithm to realize system security and convenience. Section II explains about the current trends and practices in providing security which is followed by section III which provides general description about various cryptographic algorithms. Section IV contains System design and parameters used and section V contains Security analysis which explains security of system with respect to some general attacks.

II. SYSTEM OVERVIEW

In this proposed system, we have designed a control protocol which would provide security to USB devices along with speed. The proposed system implements user authentication and key exchange agreement.

User authentication is done by providing username and password to the user during registration phase of the system. Thus each user should be first registered to the system before any access to the USB devices. Thus user needs username and password to achieve mutual authentication with the system.

Every time, the USB device is connected to the system, the user is first verified to check whether the user is valid or not. If he is a valid user then he is able to access the USB device. All files stored on the USB device are first encrypted using a key which is generated every time in verification process. The user has to acquire same session key at the time of decryption of the files stored on the USB devices so as to open or read the files. For each file a unique session key is generated based on username, password, filename and private key of AS.

The proposed system is very secure as only valid users can access the USB devices. This is provided by using username and password to valid users during registration phase. Even if the USB device is lost, then the files stored in USB devices are encrypted and thus cannot be decrypted without keys used at the time of encryption. The legal user, if wants to distribute files to

other peoples malevolently, then also the file is secure, as legal file owner cannot obtain agreement key used for decryption until authentication server suspends his account.

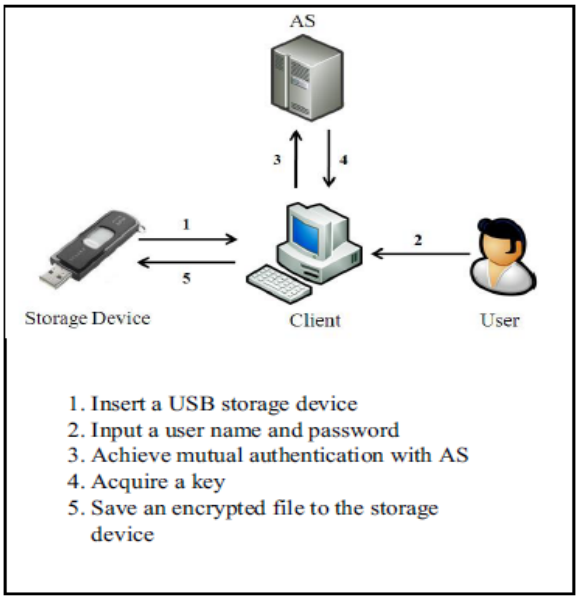

Figure 1: Overview of System

The operational flow of the system is as shown in figure 1. User first inserts a USB device. The system then forces user to pass through authentication process in order to access USB device. The user is thus forced to input username and password. This username and password is verified by authentication server called as AS. If the username and password is matches, then the person is valid user and thus AS provides him with the session key. If username and password does not match, then the user is invalid and is restricted to access USB device. The session key which is acquired by the valid user is used to encrypt the file to be stored in USB devices. This file is then stored on USB device securely. In order to decrypt the file, user has to go through the same verification process mentioned above.

III. CRYPTOGRAPHIC ALGORITHMS

Encryption algorithm is used to encrypt/decrypt the files in order to save them on USB devices. This provides security to files in case the USB device is lost. In general, there are two types of encryption algorithms used in cryptography. They are symmetric and asymmetric algorithms.

Symmetric encryption algorithm includes AES(Advanced Encryption Standards), DES(Data Encryption Standards), IDEA(International Data Encryption Algorithm), Triple-DES etc whereas Asymmetric encryption algorithm includes Diffie-Hellman, RSA(Rivest, Shamir and Adleman), DSA(Digital Signature Algorithm) etc. Hybrid algorithm is formed by combining Symmetric and Asymmetric algorithms depending on their pros and cons. [1]

The difference between Symmetric and Asymmetric algorithms is listed below in Table 1. [2]

CHARACTERISTIC	SYMMETRIC KEY CRYPTOGRAPHY	ASYMMETRIC KEY CRYPTOGRAPHY
Key Used	Public	Public and Private
Speed	Very Fast	Slow
Size of Resulting Ciphertext	Same/Less than Plain text	More than Plain text
Key Agreement	Big Problem	No problem at all
Used for	Encryption/Decryption	Encryption/Decryption and digital signature

Table 1: Difference between Symmetric and Asymmetric Cryptography

In this proposed system, we will be using asymmetric cryptography as it provides secure key exchange agreement. The differences among various asymmetric algorithms are given in Table 2. [2]

CHARACTERISTIC	DIFFIE-HELLMAN	RSA	DSA
Proposed By	Whitfield Diffie and Martin Hellman	Rivest, Shamir and Adleman	NIST
Speed	Fast in Key Generation and slow in verification	Slow in Key Generation and fast in verification	Fast in Key Generation and very slow in verification
Primarily used for	Key Generation and Encryption/ Decryption	Key Generation and Encryption/ Decryption	Key Generation

Table 2: Difference among Diffie-Hellman, RSA and DSA Algorithm

In this proposed system, we will be using RSA algorithm which was developed in MIT by Rivest, Shamir and Adleman in 1977 and first published in 1978. It will be used for only key exchange agreement. For mutual authentication, we will be using Digital Signature proposed by Schnorr in 1989.

A. RSA Algorithm

- *Key Generation*

1. Choose two distinct large prime numbers p and q.
2. Compute $n = pq$ where n is used as modulus.
3. Compute $\varphi(n) = (p - 1)(q - 1)$, where φ is Euler's totient function.
4. Choose an integer e such that $1 < e < \varphi(n)$ and e and $\varphi(n)$ are co-prime.
    - e is released as the public key exponent.

5. Determine $d = e^{-1} \bmod \varphi(n)$ i.e. compute d given $(d*e) \bmod \varphi(n) = 1$.
    - *d* is kept as the private key exponent.
6. The public key consists of the modulus *n* and the public (or encryption) exponent *e*. The private key consists of the modulus *n* and the private (or decryption) exponent *d* which must be kept secret.

- *Encryption*

1. Message M is to be transmitted.
2. Convert M into an integer m, such that $0 < m < n$ by padding scheme then computes the ciphertext *c* corresponding to

$c = m^e \pmod n$.

3. Then transmits *c*.

- *Decryption*

1. Recover *m* from *c* by using private key exponent *d* via computing

$m = c^d \pmod n$.

2. Given *m*, we can recover the original message M by reversing the padding scheme.

B. Digital Signature

Schnorr signature scheme is used to limit the number of signatures. Schnorr signature scheme employs a subgroup of order q in Z*p , where p is some large prime number. The method also requires a hash function H : {0, 1}* –> Zq.

- *Key Generation Algorithm*

1. Select prime numbers q and p with the property that q divides (p − 1).
2. Select a random integer x such that $1 \le x \le q - 1$.
3. Compute $y = g^x \bmod p$.
4. A's public key is (p, q, α, y), and A's secret key is x .

- *Signature Algorithm*

1. Select a random secret integer k, $1 \le k \le q - 1$
2. Compute $r = g^k \bmod p$, $e = H(m||r)$, and $s = x \cdot e + k \bmod q$
3. A's signature for m is the pair (s, e).

- *Signature Verification*

1. Compute $v = g^s \cdot y^{-e} \bmod p$, and $\bar{e} = H(m||v)$
2. Accept the signature if and only if $e = \bar{e}$ [3][4].

IV. System Design

User has to first register to the system. After registration phase, when user connects USB device and have to go throw verification and data encryption phase where session key will be generated which is used to encrypt/decrypt file.

A. Parameters and Symbols

1. *p, q* : Two large primes *p* and *q*, where *q* | *p*-1.
2. *g, G* : *g* is an element chosen from Z**p* and having an order of *q*; *G* is the cyclic group generated by *g*.
3. *id, pw* : User account (user name) and password.
4. *x, Y*: Server's private key and public key; $Y = gx \bmod p$.
5. *h*(.), *H*(.) : One way collision-resistant hash functions; *h*(.) maps arbitrarily long strings to strings of fixed length, and *H*(.) maps to elements of the cyclic group *G*.
6. ||: Concatenate operate.
7. *Fn* : Filename for encryption.
8. *File* : File for encryption.
9. *EK*[.] : Symmetric encryption function with respect to a key *K*.
10. *DK*[.] : Symmetric decryption function with respect to a key *K*.

B. Registration Phase

A user needs to register with the system before accessing the USB device. During the registration phase, the user first inserts a USB storage device and then chooses one set of *id* and *pw*. The *pw* is substituted into the one way hash function to calculate $hpw = H(pw)$, and then *id* and *hpw* are sent to the authentication server. When receiving this registration message, the server will choose a random number *k* and generate $r = hpw^k \bmod p$ and $r1 = g^k \bmod p$. It then computes $e = h(id \| r \| r1)$, use its private key *x* to calculate $s = (k - e \cdot x) \bmod q$, and save (*e*, *s*) to the user's storage device. After receiving the triplet (*e*, *r*, *s*), the user checks whether *e* is equal to $h(id \| r \| g^s y^e \bmod p)$ [5]. A valid check concludes the registration phase. Data transmission in this phase is done under a secure channel. In addition, in order to avoid password guessing attacks, the protocol will force the user to choose password with sufficient complexity.

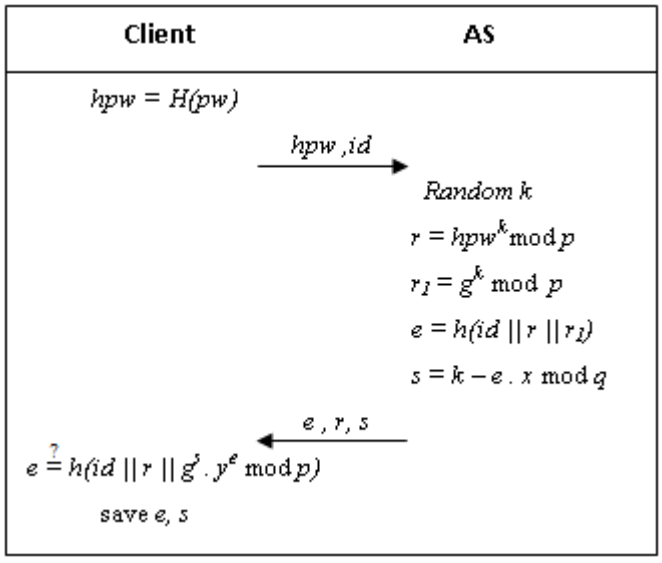

Figure 3: Registration Phase

C. Verification and Data Encryption Phase

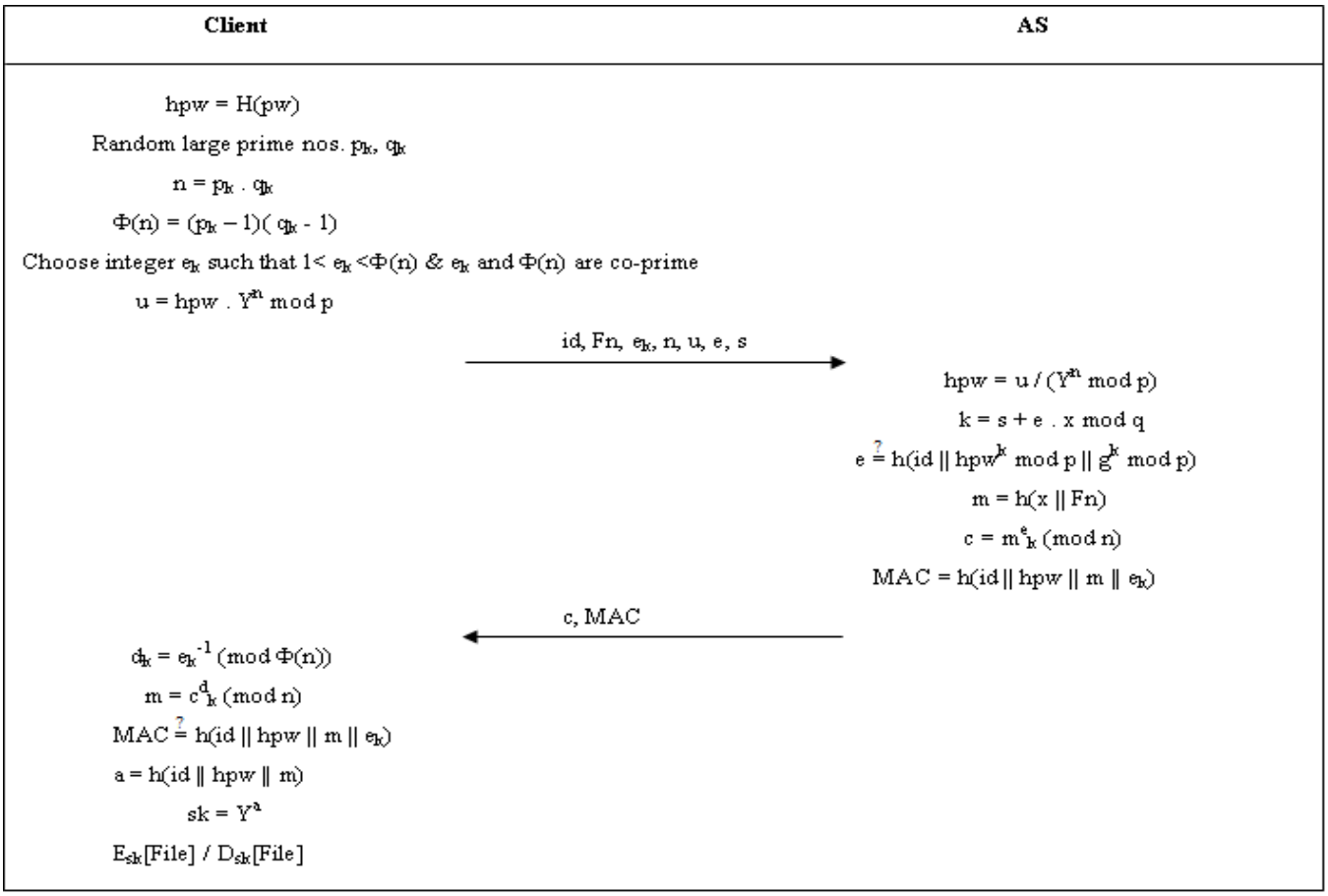

Figure 4: Verification and Data Encryption Phase

After completing the registration phase, and when accessing the USB storage device, the user needs to achieve mutual authentication with the authentication server using the id and pwd, which generates an encryption key. The communication procedure is described in detail below
Figure 4 shows the full procedure of verification and data encryption phase.

• Step 1:
The user attaches the USB storage device to a computer through a normal procedure and inputs the correct id and pw. At this moment, the user (client) will use pw to calculate hpw using the one way hash function H(.). Then the user chooses a random large prime numbers pk and qk and calculate modulus n=pk.qk. Euler's totient function Φ(n)=(pk-1)(qk-1) is also calculated. Public key exponent ek co-prime with Φ(n) is chosen such a that 1<ek<Φ(n). Calculates u = hpw . yn mod p. Finally, the user will send messages of {id, Fn, ek, n, u, e, s} to the authentication server.

• Step 2:
After receiving {id, Fn, ek, n, u, e, s}, the AS will use its long term private key x to calculate hpw = u / (y n mod p) and k = s + e . x mod q. Then, the authentication server will employ parameters it generated to verify whether e = h(id || hpwk mod p || g s y e mod p). If yes, then the user in this communication is legal. If not, the communication is terminated. Subsequently, the authentication server will use the received file name Fn and the long-term private key x to calculate m = h(x || Fn), and perform encryption on m, to generate ciphertext c= mek mod n. Finally, the authentication server calculates a message authentication
code MAC = h(id || hpw || m || ek) and sends the generated message {c, MAC} to the user.

• Step 3:
After receiving the message {c, MAC}, the user uses the public key exponent ek and Φ(n) to calculate private key exponent dk. m is retrieved by decrypting c using d to calculate a used in generation of session key sk, Next the user will verify whether MAC = h(id || hpw || m || ek). If yes, then mutual authentication is achieved between the
user and the authentication server, and the user will calculate a = h(id || hpw || m) and generate an encryption key sk using the equation sk = (Y) a = g x . a mod p.

• Step 4:
After the user and the authentication server complete these steps, the session key sk can be calculated by sk = gx . a mod p. When

a user wants to access the storage device via the USB interface, this encryption key, can be used to encrypt the File, i.e., as E sk [File], to protect the file and provide private and secure access to the USB device. For file decryption, the user needs to undergo the same verification steps and obtain the same key sk to decrypt the file (D sk [E sk [File]]) when accessing it on the USB device.

V. SECURITY ANALYSIS

System Analysis means determining whether the project is economically, socially, technologically and organizationally feasible.

- *Correctness*

Our protocol will prevent any confidential file loss via USB removal storage device. In our protocol design file transfer via USB interface is blocked till the user does not passes through authentication procedure. If the user is valid then the required files are transferred to peripheral device (USB) in encrypted format. The key used for encryption is computed using Username, Password and filename. After encryption if user want to read that file he has to first decrypt it. For decryption, user has to go through same authentication procedure and have to obtain same key used for encryption.

- *Offline password guessing*

If the USB is lost or stolen, yet USB access is restricted as for decryption, username and password is required. Thus preventing confidential data stored in USB device. If user tries to guess password, it will be hard to him as it includes solving Discrete Logarithmic Problem [6].

- *Discrete Logarithmic problem*

1. In verification and data encryption phase if attacker tries to guess the value of parameter for that he has to pass through Discrete Logarithmic Problem.
2. Discrete Logarithmic problem where variable have number of solution.
3. eg: X%2=1; to get answer as 1 ,X having number of value(X=3,5,7, ...)

Session Key is generated for each verification message in our protocol. Without knowing p_k and q_k and private key x, attacker cannot decrypt the file. So our protocol resists offline password attack [7].

- *Replay attack and Stolen verifier attack*

If attacker tries to use a captured wiretap login message and he get some parameter but he don't know p_k and q_k, means he don't know m used to calculate session key sk. Even though he finds session key still password is required, therefore our protocol can resist the stolen verifier attack [8].

VI. CONCLUSION

The Proposed System has a secure and efficient control protocol for USB ports. The protocol employs a remote authentication server to verify legal users and uses the Cryptographic algorithm to implement key exchange agreement to protect the privacy of a file transmitted to a storage device. Also the proposed system can resist some general attacks. In terms of protocol communication costs, realizing mutual authentication requires only two rounds of communication sessions. Therefore, the proposed system provides an effective control protocol for USB storage devices which is both secure and efficient.

ACKNOWLEDGMENT

We would like to express our gratitude towards a number of people whose support and consideration has been an invaluable asset during the course of this work.

REFERENCES

[1] W. Diffie and M. Hellman, "New directions in cryptography", IEEE Transactions on Information Theory, Vol. 22, No. 6, pp. 644-654, 1976.

[2] Gustavus J. Simmons, "Symmetric and Asymmetric Encryption", Computing Surveys, Vol. 11, No. 4, December 1979.

[3] Chul-Joon Choi, Zeen Kim and Kwangjo Kim "Schnorr Signature Scheme with Restricted Signing Capability and Its Application".

[4] R. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems", *Communications of the ACM*, Vol. 21, No. 2, pp. 120-126, 1978.

[5] Hyun Sook Rhee, Jeong Ok Kwon, and Dong Hoon Lee, "A remote user authentication scheme without using smart cards", Computer Standards & Interfaces, Vol. 31, No. 1, pp. 6-13, 2009.

[6] Mrs. C. Shoba Bindu, Dr P. Chandra Sekhar Reddy and Dr B.Satyanarayana, "Improved Remote User Authentication Scheme Preserving User Anonymity", International Journal of Computer Science and Network Security, VOL.8 No.3, March 2008.

[7] T. A. ElGamal, "A public-key cryptosystem and a signature scheme based on discrete logarithms", IEEE Transactions on Information Theory, Vol. 31, No. 4, pp. 469-472, 1985.

[8] Lu Zhu, Sheng Yu and Xing Zhang, "Improvement Upon Mutual Password Authentication Scheme", 978-0-7695-3560-9/08 IEEE DOI 10.1109/ISBIM, 2008.

AUTHORS

First Author – Prof. R. M. Goudar, Computer Department, Maharashtra Academy of Engineering, University of Pune, Ganeshkhind, Pune, India
rmgoudar66@gmail.com

Second Author – Tushar Jagdale, Computer Department, Maharashtra Academy of Engineering, University of Pune, Ganeshkhind, Pune, India
jgdtushar@gmail.com

Third Author – Ketan Kakade, Computer Department, Maharashtra Academy of Engineering, University of Pune, Ganeshkhind, Pune, India
ketankakade8@gmail.com

Forth Author – Amol Kargal, Computer Department, Maharashtra Academy of Engineering, University of Pune, Ganeshkhind, Pune, India
aakargal@gmail.com

Fifth Author – Darshan Marode, Computer Department, Maharashtra Academy of Engineering, University of Pune, Ganeshkhind, Pune, India darshanmarode@gmail.com