Overview of Security Algorithms

Saloni Goyal
Chameli Devi Group of Institution
166/1 Moti Bungalow
Balaji Niwas, Dewas (M.P)

ABSTRACT

In ancient times we did not have technology to transfer messages (data) electronically and hence only the receivers receives it. No third could intercept it. Nowadays web apps have become popular and widely used in fields that require high level of security such as finance, military systems, E-mails, medicals, etc. With increase in this, increase in malicious activities and attacks have been observed. Hosts have become botnets[1], number of host scam phishing sites have increased.

Actions without a prior thought and thinking without acting can now not be given any place. It's high time to secure our data. This paper comprises of an overview of security algorithms that are a shield to our electronic data.

General Terms

Security Algorithms, Cryptographic methods.

Keywords

Secret key cryptography, Public key cryptography, RSA, cryptographic algorithms, AES, ECC.

1. INTRODUCTION

We do not want someone to intrude our privacy and so the techniques to keep our information hidden, known as cryptography are used. Cryptography uses CIPHERS which are like virtual locks to keep our information safe and owner of the key is the receiver who decrypts the message. This method of encryption and decryption also include authentication and integrity check and the method of non-repudiation (mechanism to check that sender really sent this message).To prevent our data from various attacks[2] like Cipher text Only Attacks(COA), Known Plaintext Attack (KPA), Dictionary Attack, Birthday Attack**,** etc, various cryptographic schemes are used.

2. CRYPTOGRAPHIC SCHEMES

1. Secret key cryptography or symmetric cryptography.
2. Public key cryptography or asymmetric cryptography.
3. Hashing.

3. SECRET KEY CRYPTOGRAPHY (SKC)

Symmetric cryptography or secret key cryptography is a single key cipher. Single key is used for both encryption and decryption. Sender uses some set of rules to convert the plain text into cipher text and receiver applies the same set of rules (key) to decrypt the cipher text. The risk in this system is that if either party loses the key or the key is intercepted, the system is broken and messages cannot be exchanged securely. Symmetric-key encryption plays an important role in SSL communication, which is widely used for authentication, tamper detection, and encryption over TCP/IP networks.

Secret-key ciphers generally fall into one of two categories:

1. Block cipher: applies a private key and the algorithm to a 'block' of data.
2. Stream cipher: applies a private key and the algorithm to a 'bit' of data.

Various algorithms are implemented under SKC. Some are explained below.

3.1 Electronic Code Book Mode (ECB)

Data encryption standards are used to convert messages into cipher by applying the same key on each block of plain text.

C[i] = Ek(P[i])

P[1] ---E(k) ---C[1]

P[2] ---E(k) ---C[2]

And so on…

Here E(k) is the key, C[i]=blocks of cipher text, P[i]=plain text divided in "i" number of blocks. So the current cipher text has no dependency on the previous block of plain text.

Analysis:
Similar blocks of plain text result into similar cipher text and therefore reducing the security level. Attackers can guess alike patterns by hit and trial and hence it is not an efficient method.

3.2 Cipher Block Chaining Mode (CBC)

The first block of plain text is XORed with initialization vector and then encrypted using a key into cipher text. Now the next plain text will be XORed with previous cipher text and then encrypted to result in a new block of cipher text. This process continues till the last block of plain text.

If the first block has index 1, the mathematical formula for CBC encryption is:

Ci = Ek (P[i] XoR C[i-1])
C0=initializing vector

While the mathematical formula for CBC decryption is:
Pi= Dk(Ci XORed (i-1))
C0= initializing vector

Analysis:
CBC creates a new cipher text even for the similar plain text hence removing disadvantage of ECB mode. Although it is a slow process and cannot be parallelized since the result shows partial dependency on previous cipher. Error in any bit of

plain text or initializing vector may result into wrong encryption and data will not be received correctly.

Decrypting initialization vector wrong may result in corrupted first block of plain text but that doesn't affect other plain texts.

3.3 Cipher Feedback Mode (CFB)

1. To encrypt data we need some inputs, therefore to start we take an initialization vector of n bits which is encrypted using a key and the output so formed has same number of bits out of which there are s significant bits and the remaining n-s bits are discarded. The significant bits are then XORed with the plain text message block to produce s bit cipher text.

2. To generate the next n bit input block we left shift the n-s bits from previous input block (initialization vector in the beginning) and use the s bits of previous cipher text. $x_{i =} x_{(i-1)}$[n-s] concatenated with C_{i-1}

3. S<n and the process continues till the last message block.

Analysis:

Bit error accounts for only one bit error and Decryption can be parallelized but since the bits are prone to flipping and encryption is in a serialized manner with a dependency on previous inputs and ciphers; this is not 100% efficient.

3.4 Output Feedback Mode (OFB)

The only difference between OFB and CFB is that s bits are directly taken from the output and not from the cipher text to produce the next input block.

Analysis:

Unlike CFB it is not dependent on plain text or cipher text.

3.5 Triple Data Encryption Standards (TDES)

This technique was used to remove the problems of (DES) Data encryption standards. It applies DES algorithm thrice to the block of data. The normal DES system uses 56 bits +8 parity bits therefore this system uses 3*56=168 bits.

1. Initial 64 bit plain text is encrypted using key1 and the output is DES cipher. Key is generated using Round key generator which uses Round function and permutations.
2. Decrypt the output of step 1 using single DES with key K_2.
3. Encrypt the output of step 2 using single DES with key K_3.
4. Output of step 3 is the cipher text.

Analysis:

TDES is encrypt-decrypt-encrypt process therefore it is more secured but slower than DES. Cipher Text is completely dependent on plain text so if there is change in any bit of plain text then cipher will also be affected. Despite of all the vulnerabilities, it has managed to survive the market and is still being used in industries like electronic payment.

3.6 Advance Encryption Standards (AES)

Joan Daemen and Vincent Rijmen developed this technique for improving efficiency in both hardware as well as software. Since DES could encrypt only 64 bits therefore a new method was needed with a better key size. AES uses 128/192/254 bit keys and is faster than TDES. This software is implementable in C and java. Number of input bits is same as number of output bits.

1. Initially message undergoes pre round transformation via round generator and unlike DES number of rounds are varying here. Total number of rounds:10 rounds for 128 bit, 12 rounds for 192 bit and 14 rounds for 254 bit.
2. Number of rounds is nothing but loop to process cipher text.
3. 1st round :Sub bytes
4. 2nd round: Shift Rows
5. 3rd round: Mix columns
6. 4th round: Add Round Key
7. These 4 rounds will be present in every loop.
8. Key length for 128 bit is 4 and block size is also 4, key length for 192 bit is 6 and block size is 4 and in case of 256 bit key length is 8 and block size remains same.
9. Internally this algorithm works on a 2-D array of bytes.
10. We can decipher it using inverse cipher mechanism[3] which includes following:
    InvShiftRows();
    InvSubBytes();
    InvMixColumns();
    AddRoundKey();

Applications: It is used in SSl over the internet, in cloud storage system [4], Automatic teller machines since 2003 etc.

Analysis:

AES uses variable key length and is very secure if proper key management is done. No attacks have been recorded against this method.

4. PUBLIC KEY CRYPTOGRAPHY (PKC)

In symmetric algorithm {n*(n-1)}/2 keys were used and to solve this key distribution problem PKC was introduced which included n number of keys in general. There is one public key used for encryption and a private key used for decryption and so the two parties: sender and receiver never have to communicate to send messages over a secure network. This can be implemented using following methods:

4.1 Rivest Shamir Adleman (RSA)

The basic requirement for a PKC method is encryption and decryption key and to generate these we follow the steps below:

1. Take any two random prime numbers p and q.
2. Calculate RSA modulus n = p*q.
3. Calculate Eulers toitent[5] Function $\phi(n) = (p-1)(q-1)$
4. Select the value for a public key e such that $e<\phi(n)$ and e is co-prime to $\phi(n)$ ie. The only common factor between two is 1.
5. Now to calculate decryption key as our private key 'd' we use:
   $d * e = 1 \bmod \phi(n)$

Euclidean Algorithm is used to solve this equation and find d.

6. Cipher text is calculated as C= (m^e)*mod n
7. Message is decrypted by m = (C^d) * mod n

Applications:

1. Bank account security.
2. While buying goods on shopping websites
3. Gmail uses 1024 bit prime number while twitter uses 2048 bit prime number.

Pierre de- Fermat gave a mathematical fact in 17th century for calculating multiples (a^b)-a = n(b) where b is a prime numbers. Example (4^3) – 4 = 60 and this 60 is a multiple of 3.

Analysis:

Its security is based on the difficulty of factoring large integers. This method is used in digital signatures. It is widely accepted and used and is more efficient for encryption than decryption.

Cons: If the factor n is known then even without having decryption key, it is possible to reconstruct message, also the value of e should be large to protect the message from attackers.

4.2 Digital Signature Algorithm (DSA)

National Institute of Standards and Technology has proposed this algorithm in 1991. A signature must be verifiable and non-forgeable and to achieve this, key must be generated: public and a private key.

Public key applied on private key returns the message so the message can be verified as the receiver has the public key. If somebody tries to forge the sign, it means he/she is trying to create a private key similar to the original private key which is the reverse process of verification algorithm and it is very difficult to do so.

Difference between DSA and RSA is:

1. DSA is faster at generating keys and decrypting while RSA is better at encrypting.
2. DSA is faster in signing while RSA is faster at verifying.

Analysis:

DSA and RSA can be used in accordance looking at the needs of client and server.

4.3 Elliptic Curve Cryptography (ECC)

ECC uses Diffie Hellman algorithm[6]. It creates keys using the properties of elliptic curve equation. To understand this we must know addition and multiplication on an elliptical curve:

1. Addition:

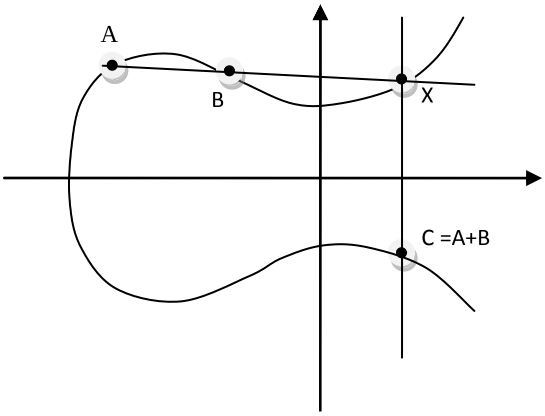

Fig 1: Addition on an elliptic curve

If we draw a line joining two points on the curve, we will always have a third intersecting point on the curve (few are exceptions).To sum the points A and B, draw a line joining A and B to cut the curve at x and then draw a line from x parallel to y axis so as to cut the curve at C. This point C gives the sum. And now if we want to sum C and X, it is not possible because there is no other intersecting point corresponding to C and X therefore they sum up to infinity.

2. Multiplication:

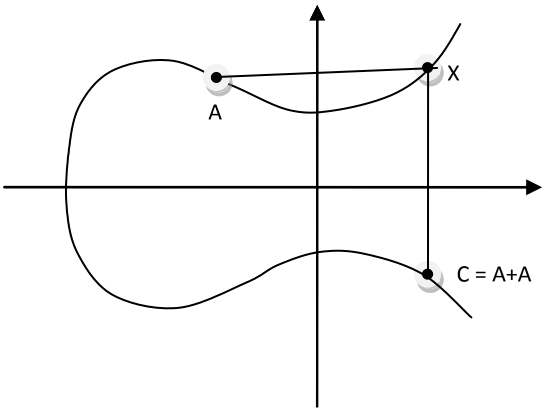

Fig 2: Multiplication on an elliptic curve

Multiplication is same as repetitive addition or adding the same point twice and hence called point doubling. B is found by drawing a parallel line from the Intersecting point and this result in multiplication.

Some terminologies:

G = generator point on the curve which is public
N = prime order of G
H = cofactor (no. of points on n, we want h to be 1)
Procedure:

1. Sender and receiver chooses a random value d such as 1<=d<= n-1.

2. D * G = (X_g,Y_g) ; even if G,X_g,Y_g is known, attacker cannot figure out the private key D.
3. Example: ABC is transferring his information by p=d*g where p is public and d is private , XYZ is transferring information by q=e*g where q is public and e is private key of XYZ. Now ABC will use public key of XYZ to retain information (R = q*d) and XYZ will use public key of ABC which is (R = e*p) and they both decrypt the same information R securely. Here R=d*e*g which is any point on elliptical curve.

Applications:

1. Credit card numbers.
2. $y^2= x^3$ -3x + b(mod p) is used to generate random numbers

Analysis:
This method is most secure and requires fewer bits in comparison to other algorithms. It is computationally infeasible to calculate all the points on the curve. Factorizing is the only solution found till date to get the multiplicative of generator point.

5. HASH FUNCTIONS

A hash function is a mathematical function that converts a numerical input value into another compressed numerical value. The input to the hash function is of arbitrary length but output is always of fixed length called hash value. Example: If the input was a '.jpg' image file the resulting hash value would effectively be a fingerprint for that file.

An ideal hash function has three main properties: it is easy to calculate the hash value for any given data, it should be extremely difficult to reverse engineer the hash value, and it is extremely unlikely that two different input values, no matter how similar, will generate the same hash value. There are many different types of hash functions, with differing security properties.

5.1 Types of Hash Functions

1. Message Digest (MD): The MD family comprises of hash functions MD2, MD4, MD5 and MD6 out of which MD5 was very popular and most secure used for software check. An analytical attack was reported against MD5 in 2004 after which it was not much used. It was adopted as Internet Standard RFC 1321. It is a 128-bit hash function.

2. Secure Hash Function (SHA): SHA-0, a 160-bit original hash function had some weaknesses so SHA-1 was designed which is widely used of existing functions especially for Secure Socket Layer security.
3. Then came SHA-2 with its different variants. SHA-256 for 256 bits. SHA-3 series had SHA-384 and SHA-512 for 512 bits.
4. SHA-3 originally known as keccak[7] is a sponge function and has efficient performance and good resistance for attacks.
5. The new algorithm include: Skein[8] hash function family whose design is based on threefish[9] block cipher.

5.2 Applications

Password storage (intruder can only see the hashes of password), data integrity check (used to generate checksums on original file but this is be possible only when user is sure about the originality of file). It is used in applications like dropbox. Advance version of hashing along with Hardware Security Module is used by well known companies like Facebook.

6. KEY ANALYSIS

Table 1. Key Length of algorithms

DES	TDES	AES	RSA	ECC
56 bits + 8 parity bits	112 bits (meet in the middle attack)[10]	256 bit maximum	Can range from 768 bit to 1024 and more	160 bit to 512 bit equivalent to maximum of RSA

7. CONCLUSION

Symmetric Algorithm had to face key establishment and trust challenges. Both the sender and receiver need to agree on a secret symmetric key and they must trust each other that the key is safe from attacker and the parties will keep it private to themselves and won't share it with anyone else. To solve these issues Asymmetric algorithm is introduced. Although here also receiver needs to trust that public key he is using is of the sender and not of a malicious third party. This is managed by Public Key Infrastructure. The commonly used algorithms are RSA, TDES, AES, blowfish[11] and the whirlpool[12] hash function which is 512 bit function derived from AES. ECC comes out to be the most secure method but everything about ECC is not yet discovered and researches are continuing on this. Cryptographic methods have improved the security in data transfer over the internet. This is a human issue and everybody must be aware of the keys they use. These policies will work better if users don't leave the keys lying around or chose a common and easily remembered key. Therefore ACT SMART AND KEEP SAFE.

8. ACKNOWLEDGMENTS

I would like to thank my mentor Mr. Shadab Pasha for his constant support. I am also grateful to the anonymous referees whose numerous suggestions helped to improve this paper.

9. REFERENCES

[1] Neelam Paliwal, Ramesh Singh Rawat, Deepak Singh Rana,2015, Survey of Botnet Based DDoS Attack and Recent DDoS Incidents, International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 4, Issue 5.

[2] Kartikey Agarwal and Dr. Sanjay Kumar Dubey, 2014, Network Security: Attacks and Defence, International Journal of Advance Foundation and Research in Science & Engineering (IJAFRSE), Volume 1, Issue 3.

[3] Ritu Pahal, Vikas Kumar, 2013, Efficient Implementation of AES, International Journal of

Advanced Research in Computer Science and Software Engineering, Volume 3, Issue7.

[4] K.S.Suresh and Prof. K.V Prasad, 2012, Security Issues and Security Algorithms in Cloud Computing, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue 10

[5] Clay S.Turner, 2008, Euler’s Toitent Function and Public Key Cryptography, The Pennsylvania State University

[6] Ekta Lamba and Lalit Garg, 2014, Enhanced Diffie Hellman Algorithm, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 4, Issue 6.

[7] Imad Fakhri Al-shaikhli, Mohammad A. Alahmad and Khanssaa Munthir, 2013, Hash Function of Finalist SHA-3: Analysis Study, International Journal of Advanced Computer Science and Information Technology(IJACSIT), Vol. 2, No.2

[8] Amar Jaffar and Christofer J. Martinez, 2013, Detail Power Analysis of the SHA-3 Hashing Algorithm Candidates on Xilinx Spartan-3E, Vol. 5, No. 4

[9] Littey P.Oomenn and Anas A S,2013, Skein and Threefish Implementation on FPGA, International Journal of Science and Research (IJSR), Vol.4,Issue 5.

[10] Dmitry Khovratovich, Ivica Nikoli´c, and Ralf-Philipp Weinmann, 2009, Meet-in-the-Middle Attacks on SHA-3 Candidates, International Association for Cryptologic Research.

[11] Rashmi A. Gandhi and Atul M.Gosai, 2015, A Study on Current Scenario of Audio Encryption, International Journal of Computer Applications, Vol. 116, No 7.

[12] Rajeev Sobti, G.Geetha, 2012, Cryptographic Hash Functions: A Review, IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 2, No 2.

[13] Rajeev Sobti, G.Geetha, 2012, Cryptographic Hash Functions: A Review, IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 2, No 2

[14] SIGCHI Conference on Human Factors in Computing Systems

[15] Tavel, P. 2007 Modeling and Simulation Design. AK Peters Ltd.

[16] Sannella, M. J. 1994 Constraint Satisfaction and Debugging for Interactive User Interfaces. Doctoral Thesis. UMI Order Number: UMI Order No. GAX95-09398., University of Washington.

[17] Forman, G. 2003. An extensive empirical study of feature selection metrics for text classification. J. Mach. Learn. Res. 3 (Mar. 2003), 1289-1305.

[18] Brown, L. D., Hua, H., and Gao, C. 2003. A widget framework for augmented interaction in SCAPE.

[19] Y.T. Yu, M.F. Lau, "A comparison of MC/DC, MUMCUT and several other coverage criteria for logical decisions", Journal of Systems and Software, 2005, in press.