Research Paper Computer Science

An Investigation Into the Field of Cryptography and Cryptographic Algorithm Protocols

*Pratik A Vanjara **Dr. Kishor Atkotiya

* Department Of Computer Science & I.T., Shree M & N Virani Science College, Rajkot – 360 002

** Head, Department of Computer Science,J. H. Bhalodia Women's College, Rajkot - 360 002

ABSTRACT

The field of Information Security and the subfield of cryptographic Algorithm protocols are both vast and continually evolving and expanding fields. The use of cryptographic protocols as a means to provide security to web servers and services at the transport layer, by providing both Encryption and authentication to data transfer, has become increasingly popular. I intend To discuss the need for research into cryptography and to look at existing cryptographic Algorithms, cryptographic protocols and related concepts. Finally I intend to look at some related work in detecting encrypted applications.

Introduction

This paper introduces and defines concepts relating to cryptography, cryptographic algorithm protocols, issues relating to cryptography and the development of software frameworks. Cryptography is the discipline, art and science of ensuring that messages are secure from possible “attacks”, whether these “attacks” be eavesdropping, impersonation or corruption. Cryptography provides security through a number of mathematical transformations that can be proven to be mathematically secure provided some optimum conditions. We however need to cognizant that cryptography on its own is insufficient to ensure a high level of security within an organization, that is to say that cryptography is not the silver bullet to solve all information security issues and should be used in conjunction with good security practices. Cryptography, like the Information Security field itself, is an incredibly broad field involving many existing disciplines such as abstract algebra to provide mathematical proofs for the guaranteed correctness of an algorithm, statistics for analysis of cryptographic algorithms and quantum physics for quantum based random number generation for quantum cryptography. In this literature review I intend to discuss some cryptographic principles, cryptographic algorithms and the related processing and security costs of employing these algorithms.

Cryptographic algorithm protocols are a vital component of Information Security as a means of securing modern networks against would-be attackers by providing data integrity, encryption and authentication to network traffic at the transport layer. Sensitive information, such as banking details, that transverses networks will most likely do so through an encrypted tunnel provided by the cryptographic algorithm protocol; it is thus imperative that both the protocol itself is secure and the applications use of the protocol is correct and sensible. A recent paper by Lee et al. shows that in a study of over 19000 web servers, 98.36% of the servers provided support for TLS and 97.92% provided support for SSLv3.0 and 85.37% provided support for SSLv2.0. These statistics serve to show the prevalence of SSL/TLS and the need to support these protocols.

Cryptography

Cryptography is a common component of any Information Security infrastructure; whether it before the encryption of large files for secure long term storage or ensuring that communication lines are safe for the transfer of confidential information. In this section I discuss two basic schemes ofCryptography, symmetric cryptography and public key cryptography, also outlining cryptographic Hash functions.

Symmetric Cryptography

Symmetric cryptography, also known as secret key cryptography, has been in use since ancient times and has a wide variety of different implementations ranging from simple substitution ciphers such as Caesars Cipher to complex and supposedly “mathematically unbreakable” algorithms such as AES. Symmetric key encryption makes use of a single key that must be kept secret, this key is used for both the encryption and decryption of messages to be sent or stored. I will outline some of these functions, how they work and the relative amount of work required to perform each.

The Data Encryption Standard (DES)

The Data Encryption Standard was developed by IBM and was selected in 1976 as an official

Federal Information Processing Standard for the United States. The original DES algorithm used a 64-bit key, of which 8-bits are used for parity and the remaining 56-bits are used to encrypt the plain-text. The required computations for brute forcing a DES key would be 255 operations, given a 64-bit plain-text and 64-bit DES key. While the DES algorithm itself is considered to be resistant to cryptanalysis, the actual keys used for encryption are considered to be fairly weak. The DES algorithm consists of three phases.

Phase 1

The first 64-bits of plain-text, which we will call collectively, x, run through an Initial Permutation function, which we shall denote as IP, returning 64-bits of output, which we will call x0. We can mathematically represent this as x0 = IP(x): The output is separated into equal length sections, obviously consisting of 32-bits each. We will represent this separation as L0R0, where L0 represents the first 32-bits and R0 represents the remaining 32-bits. Further we define an inverse function of the Initial Permutation function, which we call IIP.

Phase 2

The output then undergoes 16 repetitions of a computation that is key dependent using some cipher function, which we shall call f, making use of a key scheduling function which we shall call KS. A key scheduler calculates all the sub-keys for each round or iteration. The output of each iteration or round can be represented as xi = LiRi with $1 \leq i \leq 16$ with Li = Ri - 1 and Ri = Li f(Ri – 1,Ki). The Ki’s are 48-bit blocks that can be derived from the original 56-bit String using KS.

Phase 3
In the final phase, IP is applied to x16to give another 64-bit cipher block which we will call C, i.e. C = IIP(x16) = IIP(R16L16). We note the inverse property applies, that is IIP(IP(x)) = x.

The Cryptographic Hash function, f
Firstly, this function will expand the Ri's from their 32-bit block to a 48-bit block through an expansion permutation. Essentially this function increases the bit length by reusing some of the bits in the R'is, and also re-ordering them making use of a lookup table. We then exclusive-or this output together with Ki. This result is then broken up in 8 blocks of 6-bits each. These 6-bit blocks are then passed through an S-box giving an output of 4-bits. The S-box takes the first bit and the last bit of the input forming a 2-bit binary number. The base10 value of this 2-bit number is used to select a row. The remaining inner 4-bits are used to select a column number. These row and column values are used to index a value from the S-box. The 4-bit output of each of these 8 boxes is then concatenated to yield a 32-bit output which is finally given to the permutation function P which gives a result of 32-bits.

Key Scheduling
The key scheduling function, KS, is used to make the 48-bit Ki's from the original 56-bit key. We note that while DES keys are 64-bit, only 56-bits are actually used to seed the random functions as 8-bits are used for error checking. Every 8th bit (i.e 8th, 16, 24 ... 64) is used for parity. The key scheduling functions consist of two permutation functions, PC1and PC2, where PC stands for

Permutation Choice. To select the Ki's we apply the following algorithm. Given a 64-bit key K, we discard the 8- bits used for parity and apply PC1 to the remainder of the key. This can be represented as PC1(K) = C0D0 where C0 represents the first 28-bits and D0 represents the remainder. PC itself has two components, with the first half determining Ci and the second half determining Di. To calculate the individual CiDi we apply a LSi function, which represents the number of left cylindrical shifts, this is a value which is either 1 or 2, by which Ci or Di is to be shifted. That is Ci = LSi(Ci - 1) and Di = LSi(Di - 1).

The Li function is yet another look up table function. The bits of Ci and Di are then concatenated together and PC2 is applied to the output of the concatenation, that is Ki = PC2(Ci;Di). For decryption the same key is used, but the order of functions applied is reversed.

AES
The AES accepted candidate, Rijndael, was designed by John Diemen and Vincent Rijmen from Belgium and was published in 1998, it is an iterated block cipher allowing for variable key length and allows for a choice from a number of different block size. Rijndael supports block sizes of 128-bits, 192-bits and 256-bits. Rijndael is byte orientated, compared to the bit orientated nature of DES. The number of rounds or iterations applied is dependent on the sizes of the block and the key used. For example if the block size is 128-bits and if we let m be the size of key and r the number of rounds is given by r = k/32+6. At the start a 128-bit block of plain-text is used as the initial state. This initial state will be passed through a number of key-dependent transformations, finally returning a 128-bit block of plain-text. A state is treated as a 4x4 matrix, where Ai;j will represent a single byte with $0 \leq i; j \leq 3$, i referring to the rows and j referring to the columns. For example A0;0 is the first byte and A1;0 is the 5th byte. Rijndael makes use of four basics operators to allow for transformation from one state, say A = (Ai;j), to another state, say. The set of operators used by Rijndael include the following four operators.

Operator 1 : Byte Substitution
This is a non-linear permutation that operates on each byte in the current state independently, allowing for parallelism. In this phase we take 8-bytes of the 16-byte phase a multiply them an 8 x 8 matrix, i.e. matrix multiplication of an 8 x 8 matrix by a 8x1 column vector resulting in a 8x1 column vector. This can be efficiently implemented by making use of a 256-bit lookup table or an

S-box
Operator 2 : Shift Row
This is a cyclic shift of the bytes in a state. his could be represented as say Bi;j = Ai;(j+1)mod4.

The first row will undergo no changes, owever the second row will shift one column, the third row shifts two columns and the third row will shift three columns

Operator 3 : Mix column
Each of the columns Ai undergoes a linear transformation. A transformation is applied to a column at a time and is equivalent to multiplying the columns contents by a 4 x 4 matrix, that is matrix multiplication of a 4 x 4 matrix with a 4 x 1 column matrix containing the columns values

Operator 4 : Round Key Addition
For every round a round key, RK, is generated from the cipher key via the key scheduling function. The round key is the same length as the encryption block and are represented in a 4 x 4 matrix, similar to how the plain-text is represented. We then perform exclusive or the round key with the current state

Cryptographic Algorithm Protocols
We may define a protocol as a series of steps taken in order to achieve some goal. In the case of Cryptographic algorithm protocols, the goal is to allow for the secure communication of parties by agreeing upon some standards that are to be used to encrypt/decrypt the messages sent.

Architecture of TLS/SSL
It is important to understand the underlying architecture for each of cryptographic protocols. We will consider the architecture of TLS focusing solely on the Handshake Phase, as it is the most Significant to the development of the framework. Firstly, we consider some of the goals of SSL/TLS as these goals dictate the structure of TLS. TLS aims to provide a secure connection between two parties with interoperability, extensibility, allowing for incorporation of encryption algorithms or hashing functions and efficiency provided by caching . We will consider basic architecture of TLS as it is very similar to the architecture of SSL 3.0. For our purposes, we need only to consider the Handshake phase of SSL

The Handshake
During this phase decisions are made as to what cryptographic parameters are to be used for the actual TLS connection. This include deciding on the protocol version, selecting a cipher suite and performing some secret key exchange. The client sends a client hello message to the server. The server then possibly responds with a server hello message. If there is no response then a fatal error occurs and the connection is closed. These hello messages establish: the protocol version to be used, session ID, cipher suite to be used, compression algorithm to use, clientHello.random and ServerHello.random. The actual key exchange may consist of up to four messages containing: the Server Certificate, the Client Certificate, the Server Key exchange and the Client Key exchange. If the Server Certificate is to be authenticated it is sent after the hello messages phase. Following that the Server Key exchange message may be sent if necessary. If the server passes the authentication, it may request the

Client Certificate (if the client has one and if it is required by the cipher suite). The server then sends a Hello Done message back to the client indicating the end of the Hello Message part of the handshake is complete. The server then waits for a for a client response. If the certificate request message was sent then the client needs to respond with a certificate. The client will then send its Client Key exchange message with the contents dependent on the public key encryption algorithm chosen. After the exchanges have taken place a Change Cipher Suite Message is sent from the client to server. The client then sends new messages containing the new algorithms and keys. The server responds by sending a Change Cipher Suite Message back with the new keys and algorithms. The handshake is then complete.

Related tools

SSLdump

SSLdump is an SSL/TLS network protocol analyzer which identifies TCP connections on the chosen network interface and attempts to interpret them as SSL/TLS traffic. When it identifies SSL/TLS traffic it decodes the records and displays them in a textual form to stdout. If given the cryptographic keys involved it can be used to decrypt the traffic passing through.

SSLsniffer

SSL Sniffer provides similar functionality as SSLDump with the exception that it can act as a SSLv3/TLS and SSLv2 proxy server. The issue with these sorts of tools is two-fold, they don't provide any security analysis and further they are protocol specific. I should consider talking about frameworks and development in PHP as well.

REFERENCES

[1] IEEE standard 802.15.4. IEEE Computer Society, October 2003. | [2] S. Biswas and R. Morris. Opportunistic routing in multi-hop Cryptography Algorithm. In Proceedings of the ACM SIGCOMM '05 Conference, Philadelphia, Pennsylvania, August 2005. | [3] R. Braden, T. Faber, and M. Handley. From protocol stack to protocol heap: role-based algorithm. SIGCOMM Comput. Commun. Rev., 33(1):17–22, 2003. ISSN: 0146-4833 | [4] M. Buettner, G. V. Yee, E. Anderson, and R. Han. X-mac: a short preamble mac protocol for duty-cycled cryptography algorithm. In | SenSys '06: | [5] K. K. Chang and D. Gay. Language support for interoperable algorithm. In Proceedings of the 2005 workshop on Software and compilers for embedded systems, | [6] J. I. Choi, J. W. Lee, M. Wachs, and P. Levis. Opening the sensornet black box. In Proceedings of the International Workshop on Cryptography Algorithms, Massachusetts, USA, April 2007. | [7] D. D. Clark, J. Wroclawski, K. R. Sollins, and R. Braden. Tussle in cyberspace: defining tomorrow's internet. IEEE/ACM Trans. | [8] Arch Rock Corporation. A sensor network architecture for the ip enterprise. In Proceedings of the 6th international conference on Information processing in sensor networks, demo session, Cambridge, | Massachusetts, USA, 2007. | [9] J. Crowcroft, S. Hand, R. Mortier, T. Roscoe, and A. Warfield. Plutarch: an argument for network pluralism. In Proceedings of the ACM SIGCOMM workshop on Future directions in network architecture, Karlsruhe, Germany, 2003.