PGP: An Algorithmic Overview

David Yaw

11/6/2001

VCSG-482

Introduction

The purpose of this paper is not to act as a manual for PGP, nor is it an in-depth analysis of its cryptographic algorithms. It is intended to provide additional information about PGP's algorithms for encryption and hashing.

Encryption Algorithms

Diffie-Hellman / ElGamal

Of the algorithms presented here, ElGamal is the only public-key encryption. It is listed here as Diffie-Hellman / ElGamal because ElGamal is based on the Diffie-Hellman problem and the keys that PGP uses for ElGamal encryption are called Diffie-Hellman keys. A Diffie-Hellman key consists of four values: p, a large prime, g, a primitive element under mod p, d, a random number between 1 and p-2, and e, which is g^d mod p. Only d, the decryption key, is kept secret. In fact, p, the large prime, can be used by all users, and it does not significantly reduce the security of the system. (PGP accomplishes this with optional pre-computed primes. With pre-computed primes, all keys that use a

particular prime could be broken at the same time, but since it is still impossible to compute a database of all discrete logarithms in mod *p*, this is a non-issue.)

Let's say Alice wants to share a secret key (*k*) with Bob. Alice would retrieve Bob's public key (p_b, g_b, e_b) off of a public key server. Alice would then compute a random number *r*, and the following message:

$$m1 = g_b{}^r \bmod p_b$$

$$m2 = ke_b{}^r \bmod p_b$$

The message *m*1*m*2 would then be sent, and Bob would decrypt using

$$k = m2\left(m1^{d_b}\right)^{-1} \bmod p_b$$

Substituting *m*1 and *m*2 gives us

$$k = ke_b{}^r \left(g_b{}^{r^{d_b}}\right)^{-1} = kg_b{}^{rd_b}\left(g_b{}^{-rd_b}\right)$$

Key *k* would then be used for conventional encryption of the actual text of the message to be communicated.

CAST

CAST a conventional symmetric key cipher, similar to DES in its overall construction. It uses a variable length key, but PGP uses the maximum of 128 bits. The basic algorithm is this:

INPUT: plaintext m1...m64; key K = k1...k128.

OUTPUT: ciphertext c1...c64.

1. Key Schedule Compute 16 pairs of subkeys $\{K_{mi}, K_{ri}\}$ from K.

2. (L0,R0) ← (m1...m64). Split the plaintext into left and right 32-bit halves L0 = m1...m32 and R0 = m33...m64.

3. 16 rounds: for i from 1 to 16, compute Li and Ri as follows:

$$L_i = R_{i-1}$$

$$R_i = L_{i-1}^{f(R_{i-1},K_{mi},K_{ri})}$$

4. c1...c64 ← (R16,L16). Exchange final blocks L16, R16 and concatenate to form the ciphertext. [ADAMS]

CAST has good resistance to both differential and linear cryptanalysis, and has no weak or semi-weak keys. CAST is currently the default cipher for PGP.

AES (Rijndael)

Rijndael is a symmetric key cipher, recently selected by the National Institute of Standards and Technology to be the new AES. It has a variable key length of 128, 192 or 256 bits, and a variable block length of 128, 192, or 256 bits. Unfortunately, since Network Associates chose not to publish the source to PGP 7.0, there is no easy way to tell which of the nine available modes of AES is being used.

AES uses a variable number of rounds, depending on the key and block lengths. Each of the rounds consists of four steps. [SAVARD]

1. Byte substitution. Each byte is replaced with a value generated from a S-box.

2. Shift Row. Each byte is shifted according to this table:

From			
1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

To			
1	5	9	13
6	10	14	2
11	15	3	7
16	4	8	12

3. Mix Column. Each byte, in the arrangement above, is multiplied by this matrix in $GF(2^8)$.

2	3	1	1
1	2	3	1
1	1	2	3
3	1	1	2

4. Add Round Key. This is simply an XOR of the subkey for the current round. These subkeys are generated using left shifts and byte transformations using the same S-box as was used in Step 1.

AES has several advantages that make it good for a wide variety of applications. It is simple to implement, takes up very little memory, and is very fast.

TripleDES

TripleDES is an extension of the well-known DES. It consists of three applications of DES, with the second application actually a decryption operation. TripleDES operates on a 64-bit block, and has a 3*56 = 168 bit key length, but is only thought to be equivalent to a 112 bit cipher.

Since DES is a very well known cipher, and TripleDES is simply multiple applications of DES, I will not go into detail about DES's, and therefore TripleDES's, construction.

IDEA

IDEA is a symmetric cipher that uses 128-bit keys and 64-bit blocks. The claimed strength of the cipher is from “mixing operations from different algebraic groups”, specifically, multiplication (in mod $2^{16}+1$), addition (in mod 2^{16}), and exclusive-or. The algorithm is quite simple to explain, so I will go into full detail.

Multiplication is probably the most difficult operation to explain. Since zero is not a useful value in modulo arithmetic, a value of 0x0000 is taken to represent 2^{16}. And since $2^{16}+1$ is prime, each multiplication has an inverse.

First, the 128 bit key is split into eight 16 bit subkeys. Subsequent subkeys are generated by rotating the key 25 bits to the left, and then splitting the new key into subkeys. A total of 52 subkeys are generated in this way.

For each of eight rounds the following steps are run: (p? = plaintext 16 bits. s? = subkey)

p1 x s1 --> d1

p2 + s2 --> d2

p3 + s3 --> d3

p4 x s4 --> d4

d1 XOR d3 --> d5

d2 XOR d4 --> d6

d5 x s5 --> d7

d6 + d7 --> d8

d8 x s6 --> d9

d7 + d9 --> d10

d1 XOR d9 --> d11

d3 XOR d9 --> d12

d2 XOR d10 --> d13

d4 XOR d10 --> d14

Then, d12 and d13 are swapped for input to the next round. After eight rounds, the four blocks are transformed one more time using:

d109 x s49 --> c1

d110 + s50 --> c2

d111 + s51 --> c3

d112 x s52 --> c4 [ANON]

For decryption, the same algorithm is run, but with a different set of subkeys. Each subkey used for multiplication is replaced with its multiplicative inverse (and since the modulo, $2^{16}+1$ is prime, these values are unique), and each subkey used for addition is replaced with its additive inverse.

IDEA is considered strong against differential and linear cryptanalysis, but does have one major weakness: there is a large class (2^{51}) of weak keys, but since there are 2^{128} keys to choose from, the chances of getting a weak key are 2^{-77}.

IDEA was originally the default cipher in PGP, but not any longer due to patent issues with Swiss firm Ascom. (Commercial use of IDEA requires a license.)

Twofish

Twofish was one of the AES finalists. It uses a 128, 192, or 256-bit key with a 128-bit block size. Although Twofish was not included in PGP until 7.0, for which source was not released, comments in version 6.5.8 (the last version to have its source released) indicate that it is being used with a 256-bit key. [SOURCE, CLprefs.c]

The Twofish algorithm is too complex to try to summarize here, so I will simply mention that the cipher is "a 16-round Feistel network with a bijective F function made up of four key-dependent 8-by-8-bit S-boxes, a fixed 4-by-4 maximum distance separable matrix over GF(2^8), a pseudo-Hadamard transform, bitwise rotations, and a carefully designed key schedule." [SKWWHF98]

There has been much discussion about the inclusion of Twofish in PGP, a mere eight months after Twofish was introduced. I agree with the comments of many, who feel that the decision to include Twofish was premature, based on assumptions about Twofish, rather than proven strength, and the strength of Blowfish, a related predecessor but with totally different properties. [SIMPSON2]

Hashing Algorithms

SHA-1 / DSS

SHA-1 is a hashing algorithm that takes a message of less than 2^{64} bits in length and generates a 160-bit value. (The reason for the "less than 2^{64}" part is that part of the data hashed is the length of the data, represented as a 64-bit number.) A change of even one character will yield a completely different hash, as shown in this example. [BELLDANDY]

```
"This is a test: 0"
5f6a87e9 406b03a7 5a5d3862 14fbe16c 76f061cb

"This is a test: 1"
c180ec10 093b02eb e4abb557 f2a60d71 9efd1391
```

Note that only one bit in the string changed, changing the ASCII "0" (0x30) to "1" (0x31), but the entire hash changed.

The core of SHA-1 is 80 rounds for each 160 bits of plaintext. The 160 bits are divided into 5 32-bit values, labeled *a* through *e*. Each round is as follows: [SAVARD]

- Change *a* by adding the current constant to it. The constants are, in hexadecimal:
    - For rounds 1 to 20: 5A827999
    - For rounds 21 to 40: 6ED9EBA1
    - For rounds 41 to 60: 8F1BBCDC
    - For rounds 61 to 80: CA62C1D6
- Change *a* by adding the appropriate subkey for this round to it.
- Change *a* by adding *e*, circular left-shifted 5 places to it.
- Change *a* by adding the main f-function of *b*, *c*, and *d* to it, calculated as follows:
    - For rounds 1 to 20, it is (*b* AND *c*) OR ((NOT *b*) AND *d*).
    - For rounds 21 to 40, it is *b* XOR *c* XOR *d*.
    - For rounds 41 to 60, it is (*b* AND *c*) OR (*b* AND *d*) OR (*c* AND *d*).
    - For rounds 61 to 80, it is again *b* XOR *c* XOR d.
- Change *d* by giving it a circular right shift of 2 positions (or, for consistency, a circular left shift of 30 places).
- Then swap pieces, by moving each piece to the next earlier one, except that the old *a* value is moved to *e*. (That is, *a* = *b*, *b* = *c*, *c* = *d*, *d* = *e*, *e* = *a*.)

In PGP, signatures are done using DSS, Digital Signature Standard. Currently, the key size for DSS keys is a constant 1024 bits. Since there are different keys for encryption/decryption and for signing, it makes the potential weakness of the encryption key a non-issue, since a new Diffie-Hellman key can be generated for the same DSS key. (This is why the full name of the key type in PGP is DH/DSS.)

Others

Although Diffie-Hellman keys are now the standard for PGP, PGP originally used RSA keys. Although Diffie-Hellman keys are slightly more secure per bit than RSA keys, there is a more compelling reason for the switch to Diffie-Hellman keys. The Diffie-Hellman PGP key certificate contains more than just large numbers: it also has information on preferred and accepted symmetric ciphers. The RSA PGP certificate only allows one type of symmetric cipher, IDEA. While this does not break the RSA certificates, the lack of flexibility to choose a stronger symmetric cipher and patent issues of IDEA both present a compelling case for switching to Diffie-Hellman keys.

In addition, RSA key certificates only support the older MD5 hashing algorithm. While MD5 is not considered to be broken, it is possible that two different, but closely related messages could hash to the same value. While this may not seem like a big deal, changing a “1” to a “9”, a change of only 1 bit, might give the same hash value, if circumstances are right. If signing a multi-million dollar contract, this risk is unacceptable. SHA-1, however, does not have this risk, and can be used with confidence.

References

[ADAMS] ADAMS, CARLISLE. The CAST-128 Encryption Algorithm.

http://finecrypt.tripod.com/cast.html.

[ANON] ANONYMOUS. The IDEA Algorithm.

http://www.momentus.com.br/PGP/doc/idea.html.

[BELLDANDY] "DON". Belldandy: SHA-1 calculator.

http://66.27.62.81/~omoikane/belldandy_zip.html.

[RSAFAQ] RSA Laboratories Cryptography FAQ. http://www.rsa.com/rsalabs/faq/.

[SAVARD] SAVARD, JOHN J. G. A Cryptographic Compendium.

http://home.ecn.ab.ca/~jsavard/crypto/jscrypt.htm.

[SIMPSON1] SIMPSON, SAM. PGP DH vs. RSA FAQ. 1999.

http://www.scramdisk.clara.net/pgpfaq.html.

[SIMPSON2] SIMPSON, SAM. A question on Twofish / AES / PGP, PGP-Users mailing list.

8th Mar 1999. http://cert.uni-stuttgart.de/archive/ietf-openpgp/1999/03/msg00012.html.

[SKWWHF98] SCHNEIER, B., KELSEY, J., WHITING, D., WAGNER, D., HALL, C.,

FERGUSON, N. Twofish: A 128-bit Block Cipher. 1998.

http://www.counterpane.com/twofish-paper.html.

[SOURCE] Source code.

http://www.pgpi.org/cgi/download.cgi?filename=pgpsrc658win32.zip.