Fetu Public Key Distribution
by Karl Zander

Fetu PKD, a mathematical system for distributing symmetric encryption keys over an insecure channel. Fetu PKD offers a fresh look at the Diffie-Hellman problem and offers a potentially DLP resistant primitive.

Background

Fetu PKD comes from iterative design variants of the Diffie-Hellman algorithm. The Diffie-Hellman algorithm establishes that one may derive the same secret key by exponentiating a public key by a private key, exchanging the result with the peer and exponentiating the exchange by the private key. The result is a shared secret symmetric key that can be used for encrypting communications.

The algorithm Slip introduced the concept of using a DH secret key as a secret modulus with which to compute a secret key, adding another barrier and exchange in the process for obtaining a shared secret key. Fetu PKD continues to utilize this construction, however, FetuI does not.

Design Goals

There are two primary design goals behind Fetu PKD. The first being that the algorithm must be resistant to discrete log attacks. The second being that it may have smaller key lengths than traditional Diffie-Hellman. Reaching the design goals is something that cannot be properly measured until full cryptanalysis has been performed.

Fetu PKD is the result of a few years of exploring the possibilities of similar hard problems to the DH problem. The Fetu problem is of unknown hardness.

FetuI

FetuI is a reduced version of Fetu PKD. It has the minimum number of steps required and is more closely related to traditional DH. Since FetuI and Fetu PKD both share the same initial steps, I’ll start by describing the former.

Key Generation Stage

A Fetu PKD begins with both parties generating their public and private numbers. Four numbers are generated, a public modulus M, a public key N, a secret key S and a random token in the public modulus, Y.

The public modulus is a prime of N bit size. The public key is also a prime of N bit size. The secret key is a random number between 1 and M minus 1.

Algorithm

Once the public and private numbers are generated, the two parties are ready to derive their secret key. We greet our two famous characters, Alice and Bob. Alice wishes to send a message to Bob over an insecure channel. The party members agree to use Alice's public modulus M and public key N. Both parties make a separate encryption of Alice public key by raising the public key to their private key modulo Alice's public modulus (phase0). In traditional Diffie-Hellman, one would exchange this value, instead, Alice and Bob both raise their result by the random token Y modulo Alice's public key, (phase1).

Alice and Bob both exchange phase1 over the insecure, public channel. Upon receipt, Alice and Bob both raise each other's phase1 to their private key modulo Alice's public modulus arriving at the shared secret key (phase2) that can immediately be used to encrypt a message.

Encryption/Decryption

Alice or Bob may use the shared secret key how they wish. What is most typical is for them to feed the key into a hash function and arrive at an appropriate length key with which to initialize a symmetric key algorithm. Alice or Bob may also use the key to directly multiply the cipher text with the key and the other divide the cipher text by the key.

Fetu PKD

Fetu PKD is an extension of the phase0-phase2 steps in FetuI. The benefit to using Fetu PKD is that once the secret modulus has been established, one has to only make a single exchange to derive a new key in the secret modulus. There is additional complexity and hardness in Fetu PKD vs FetuI.

Key Generation Stage

A Fetu PKD begins with both parties generating their public and private numbers. Four numbers are generated, a public modulus M, a public key N, a secret key S, a temporary secret key T and a random token in the public modulus, Y.

The public modulus is a prime of N bit size. The public key is also a prime of N bit size. The secret key is a random number between 1 and M minus 1. The temporary key is a random number between 1 and M minus 1. The random token is also a random number between 1 and M minus 1.

Algorithm

Once the public and private numbers are generated, the two parties are ready to derive their secret key. We greet, once again, our two famous characters, Alice and Bob. Alice wishes to send a message to Bob over an insecure channel. The party members agree to use Alice's public modulus M and public key N. Both parties make a separate encryption of Alice public key by raising the public key to their temporary key T modulo Alice's public modulus (phase0). In traditional Diffie-Hellman, one would exchange this value, instead, Alice and Bob both raise their result by the random token Y modulo Alice's public key, (phase1).

Alice and Bob both exchange phase1 over the insecure, public channel. Upon receipt, Alice and Bob both raise each other's phase1 to their private key modulo Alice's public modulus arriving at the shared secret modulus (phase2).

Now that the secret modulus has been established, Alice and Bob both make an encryption of the other random token Y, however, this time using their private key S. They both compute Y raised to S modulo M and exchange this value (phase3). Upon receipt of phase3, they can compute phase4 which is phase3 raised to their secret exponent S arriving at the shared secret key (phase4).

Encryption/Decryption

Like FetuI, Alice or Bob may use the shared secret key how they wish with the above mentions being the most common.

Recommended Key Lengths

Fetu PKD – Unknown

FetuI – Unknown

Cryptanalysis

What Fetu hopes to achieve is separation and obfuscation between the private exponent and public exponents used to derive the secret modulus or the shared secret key.

At the time of this writing, Fetu PKD resists attacks under the discrete log problem. Standard algorithms to solve the DLP do not solve for Fetu PKD secret private keys.

Methods to compute either the private exponents, shared secret moduli or shared secret keys have thus far been in vain and the Fetu function cannot be reverted.

There is plenty of room for further investigation into if Fetu PKD can be reversed and the hardness of this feat.

Conclusion

Fetu PKD aims to be a potentially new primitive in the Diffie-Hellman space. Whether it’s hardness is that of Diffie-Hellman or harder, only time and further analysis will tell.