Slip

Slip

Key Exchange Algorithm

Based on Diffie-Hellman

- First published in 1976 by Whitfield Diffie and Martin Hellman
- Conceptualized by Ralph Merkle

Slip

- Invented in 2019 by Karl Zander
- Reference implementation in Python requring pycrypto

Design Goals

- To be resistant to DLP attacks
- To have smaller key lengths than traditional Diffie-Hellman

Key Generation

- Generate 2 N bit primes, let them be N and M and let them not be equal
- Let M be the private modulus
- Let N be the public modulus
- Choose a integer in N between 1 and N – 1 and let that be the secret key, SK

Key Exchange Setup

- Alice and Bob generate SK, N, M
- Alice and Bob send their public modulus N to each other and compute U as the produce of NA and NB
- They compute S as product of the private modulus M and U, they keep S secret
- Alice and Bob both generate a base key Y between 1 and S – 1 and agree upon using a single Y value

Key Exchange Setup

- Alice and Bob select a temporary key T in the space of 1 to U - 1

Key Exchange Phase 1

- Alice and Bob both raise Bob's Y to their temporary exponent modulo their secret S. They exchange phase 1.

Key Exchange Phase 2

- Alice and Bob compute phase1 raised to the temporary exponent arriving at the secret modulus.

Key Exchange Phase 3

- Alice and Bob raise Alice's Y to the power of their secret key modulo the secret modulus and exchange phase3

Key Exchange Phase 4

- Alice and Bob raise each other's phase3 to their secret exponent modulo the secret modulus and arrive at the shared key

Cryptanalysis

- Phase2 may be derived using the discrete log
- Phase4 cryptanalysis - in-progress