Secure encryption and cryptographic hashing using ad-hoc random algorithm generation

Linus Lee

October 19, 2015

Abstract

One of the simplest methods of perfectly random encryption is the one-time pad, using a random, infinite sequence of numbers and modular arithmetic to embed plaintext as noise on top of the random sequence. The one-time pad, while theoretically impossible to break, suffers from major flaws that prevent it from being used in practice due to its vulnerability. In this article we introduce a method of generating ad-hoc encryption functions for each instance of encryption from a very small key that effectively produces a completely random output similar to a one-time pad, while eliminating many of the greatest vulnerabilities and practical barriers of the one-time pad encryption. In addition, we also find that when used as a cryptographic hash function, the ad-hoc algorithm is collision-resistant and has advantages over current SHA-1 or MD5 implementations.

Contents

1 The one-time pad and its vulnerabilities

Laying practicality aside and looking at pure cryptographic security, the one-time pad (OTP) is one of the most secure forms of encryption. In a one-time pad, the plaintext is superimposed on top of an equal-size, completely random piece of data, essentially rendering the encrypted data completely indistinguishable from a random data stream. However, OTP itself is unfit for practical cryptographical use because of its requirement of a completely random stream of data as long as the data being encrypted.

In past uses of OTP encryption, re-using "random" data or using a pseudorandom sequence with some predictability has lead to breakable encryption. Besides the complexity of generating a sequence of random data of substantial size, a random sequence megabytes or potentially gigabytes long also poses problems in key transfers, as not only will the keys have to be transferred, but the integrity of the keys will have to be checked, and these keys will have to be transferred for each new transfer of a piece of plaintext. OTP's greatest practial issues in execution are these, that the indefinitely long encryption key will first have to be randomly generated, then be transferred securely and entirely.

OTP's greatest weakness is thus its key, and specifically its length. Were it possible to replicate the security of OTP encryption with keys less than a kilobyte long for any data set, such an encryption method would be functionally identical to OTP while resolving its key problem.

2 Functional encryption

Traditional methods of encrytion uses a pre-determined algorithm, taking the encryption key and the plaintext as parameters. Indeed, traditional encryption methods are defined by such algorithms, from simple MD5 hashes to complex RSA ciphers.

2.1 Methods of instance-specific algorithm composition (ISAC) using bit-flicking

3 Structure of the ISAC0 stack

4 Effective Security

4.1 Overcoming OTP encryption's vulnerabilities

4.2 Statistical security

4.3 Brute-force computational overhead compared to RSA

5 ISAC0 as a cryptographic hash

5.1 Chaotic behavior and collision resistance

5.2 Resistance to time-memory tradeoff

6 Conclusions