Heatherweight Encryption: Provably Brilliant Crypto

James Heather

No Institute Given**

Abstract. Most fashionable cryptosystems suffer from many drawbacks: they produce inefficiently long ciphertexts, they take a tediously long time to perform their basic operations, and they are vulnerable to any number of troubling attacks. In this paper, we[1] propose a cryptosystem that excels in every respect. We give proofs of optimality of security, algorithmic complexity, and ciphertext size.

1 Introduction

The world has long suffered under the weight of such behemoths as RSA, ElGamal, the so-called 'Advanced' Encryption Standard, the so-called 'Secure' Hashing Algorithm[2], and that one where you wrap a strip of paper round a pencil. Every one of these is so inefficient as to be amusing. They take far too long to encrypt, produce ciphertexts that are big and unwieldy, and can be cracked with minimal effort. A small personal case study will serve to illustrate.

Last year I received an email from Mr Daniel Fardoso, the terminally ill nephew of a deceased Nigerian diplomat, who needed my help in shifting a large quantity of money so that he could live out his remaining months in peace and comfort. Modesty forbids recounting the many reasons that Daniel gave for choosing me to entrust with this task; suffice it to say that he seemed to have every confidence that I had the qualities he was looking for. He asked me to fill in a simple and supposedly secure web page with my bank details, my mother's maiden name, my passport number, and my name and address. Naturally, I concurred; and, when I clicked on the 'Submit' button, a window appeared telling me that it was encrypting the file for safe transmission to Daniel.

Until that point, I had expected that modern cryptographic methods would work securely and efficiently. Much to my disappointment, it was still encrypting the file some three hours later when I retired for the night. In the morning, on discovering that the encryption still had not finished, I sent Daniel an email expressing my concern; he reassured me that there was no sign of any problem at his end.

You can imagine my shock when I received a phone call from the police that afternoon, informing me that £35,000 had been cleaned out of my account! It distresses

** The University of Surrey's legal team has formally requested that this work be submitted in a purely personal capacity.

[1] I use the royal 'we', but the sad fact is that I was unable to persuade anyone to act as co-author. My initial aim was to break the world record for the number of authors on an academic paper, but, owing to what must be considered a terrible short-sightedness on the part of the eighty or ninety people I approached (including my own mother), the current record must stand.

[2] SHA is currently on its 256th version! The first 255 versions go belly up, and they expect us to take SHA256 seriously!

me to say that they tried to blame first Daniel, and then me, for the security breach. My complaint that I had taken every precaution to encrypt the file securely fell on deaf ears.

So much, then, for current encryption systems! Two honest, law-abiding citizens attempt a supposedly secure transaction, and the hackers take full advantage. One cannot help concluding that if the encryption had been faster, then Daniel and I might have had time to stop the theft; and if it had been more secure, then the hackers would not have been able to break it in the first place.

2 Analysis of current systems

Figure 1 gives more technical insight into the inadequacy of currently available cryptosystems. The graph is self-explanatory. Small wonder that hackers are winning the

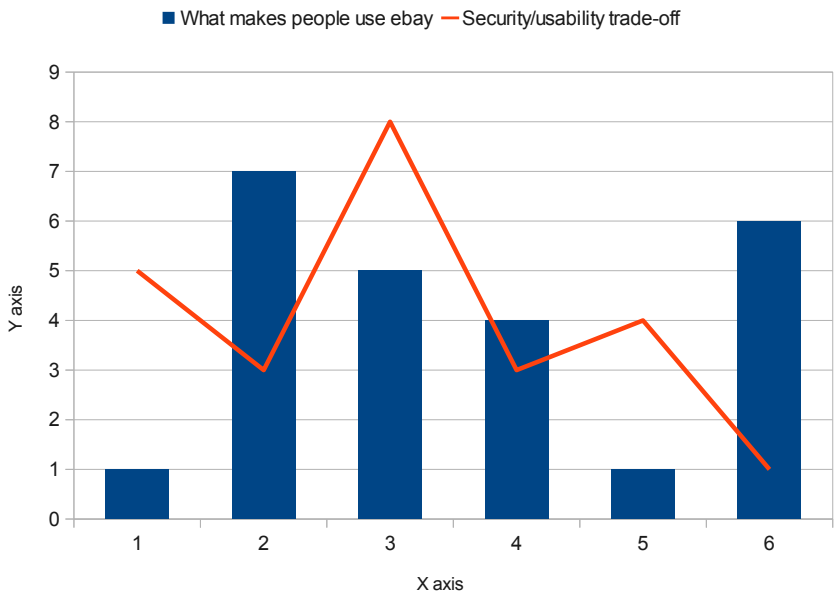

Fig. 1. RSAges-to-encrypt

battle against security experts, the banking system is in crisis, teenagers lurk around every corner drinking Hooch and smirking, and I wake up screaming in the night, scared of my own hands. A radical new approach is needed.

This paper solves all these problems. It is no exaggeration to say that the encryption algorithm proposed here does for cryptography what Isaac Newton did for physics, what Alexander Fleming did for medicine, and what Monica Lewinsky did for Bill Clinton.

3 Heatherweight encryption

We have seen that existing algorithms are both heavyweight and cumbersome. In this section, we give the details of Heatherweight[3] encryption, a simple and yet effective reinvention of the field.

The encryption algorithm is perhaps the simplest algorithm one could ask for. To encrypt a message m using key K, we calculate

$$E_K(m) = \langle\rangle$$

In other words, the ciphertext is the zero-length bitstring.

4 Proofs of optimality

We now give proofs that Heatherweight encryption is optimal in terms of ciphertext length, algorithmic complexity of the basic operations, and security.

Theorem 1 (Optimality of length of ciphertext) *Heatherweight encryption produces ciphertexts that are of optimal size for data transmission and data storage.*

Proof. Ciphertexts are zero length, regardless of the length of the input message. This means that the encryption also provides a level of compression that would make Huffman weep horrible tears. Any number of ciphertexts can be transmitted in zero time, and can be archived on even limited capacity storage media without using up any space. Using Heatherweight encryption, one can compress the whole of the Internet, and scribble the resulting ciphertext in full on the back of an envelope—*without even needing a pen!*

Theorem 2 (Optimality of algorithmic complexity) *Heatherweight encryption and decryption are optimal in their time complexity.*

Proof. It has long been assumed that the very best one might hope for, in terms of time complexity of encryption and decryption, is linear, simply because the algorithm needs to examine the whole plaintext to construct the ciphertext. However, the genius of Heatherweight encryption is that it produces ciphertexts that are *independent of the plaintext*. Encryption can therefore be done in constant time:

```
public byte[] encrypt(byte[] key, byte[] plaintext) {
    return new byte[]();
}
```

But what about decryption? It is obvious that for many encryption systems, the main security weakness is the decryption algorithm. Sometimes this is because of a flaw in the algorithm itself that allows an attacker to break the ciphertext without needing the

[3] I am indebted to Peter Ryan for the name 'Heatherweight'. Nonetheless, the University of Luxembourg's legal team has asked me to clarify that 'this does not reflect recommendation or endorsement of any of the rather confused ideas represented in this paper'. One is reminded of Decca Records' decision in January 1962 not to offer the Beatles a recording contract.

key; but there are also timing attacks and suchlike to consider. Even in the absence of these tactics, there is still the possibility that the key will leak, or that the keyholder will be tortured and forced to send the key to the attacker along a rubber hose.

Heatherweight encryption solves these problems by simply *not providing a decryption algorithm*. Security is considerably tightened by this technique, as we shall see.

This means that decryption can also be considered a constant-time operation:

```
public byte[] decrypt(byte[] key, byte[] ciphertext) {
    throw new OperationNotSupportedException("meh");
}
```

Theorem 3 (Perfect security) *Heatherweight encryption provides perfect security.*

Proof. An attacker with access to a decryption oracle is unable to distinguish between $E_K(m_1)$ and $E_K(m_2)$, because all encryptions are the same, so it is impossible to distinguish anything at all. Note that we have not reduced this simply to an underlying hard problem, but to an impossibility.

This even thwarts a 24 attack. Normally Jack is able to beat the decryption key out of anyone, and Chloë can crack any encryption, though usually not until 23:59:57. However, with Heatherweight encryption, even if Jack makes me send him the key down the rubber hose, and Jack then sends it through the hose to Chloë, she is not going to be able to get anything other than `OperationNotSupported` exceptions.

5 Conclusion

Heatherweight encryption changes everything. Just as everyone remembers where they were when they heard that JFK had landed on the moon, so everyone will remember the day when they entrusted all their backups to the power of Heatherweight encryption.

6 Future work

Research is already underway to construct a hash function based on Heatherweight encryption. I conjecture that the Heatherweight encryption algorithm itself could be used directly as a constant-time hash function; but I need to spend some time convincing myself that it would be collision resistant.

Infinite-capacity hard drives with transparent Heatherweight encryption will soon be available. A prototype is already in existence; write speeds are nothing short of phenomenal, though we are having teething trouble with the read operation, which for some reason keeps throwing `OperationNotSupported` exceptions. An investigation is underway.

Acknowledgements

The author is grateful to Antonio Banderas for inspiration. It was while watching his performance as Zorro that the zero function first surfaced as an idea for an encryption algorithm. Previous meditations on his co-star had led to consideration of the zeta function $\zeta(s) = \Sigma_{n=1}^{\infty} \frac{1}{n^s}$, which did not work nearly so well.