DIMACS Technical Report 96-02
February, 1996

Master-Key Cryptosystems

by

Matt Blaze
AT&T Research
Room 2T-316
600 Mountain Avenue
Murray Hill, NJ 07974
`mab@research.att.com`

Joan Feigenbaum[1]
AT&T Research
Room 2C-473
600 Mountain Avenue
Murray Hill, NJ 07974
`jf@research.att.com`

F. T. Leighton
Math Department
MIT
Cambridge, MA 02139
`ftl@math.mit.edu`

[1]Permanent Member of DIMACS

DIMACS is a cooperative project of Rutgers University, Princeton University, AT&T Bell Laboratories and Bellcore.

DIMACS is an NSF Science and Technology Center, funded under contract STC–91–19999; and also receives support from the New Jersey Commission on Science and Technology.

ABSTRACT

We initiate the study of a new class of secret-key cryptosystems, called *master-key cryptosystems*, in which an authorized third party possesses a "master key" that allows efficient recovery of the cleartext without knowledge of the session key. One motivation for this study is that master-key cryptosystems could provide a less cumbersome alternative to *key escrow* in situations in which third-party access is required. We demonstrate that designing a master-key cryptosystem with acceptable performance is roughly equivalent to designing a public-key cryptosystem in which encryption is much faster than is possible with current public-key techniques.

These results were presented at the Crypto rump session, Santa Barbara CA, August 1995.

Master-Key Cryptosystems

We initiate the study of a new class of secret-key cryptosystems, called *master-key cryptosystems* and abbreviated MKCSs, in which an authorized third party (hereinafter called "the government," although it need not literally be one) possesses a "master key" that allows efficient recovery of the cleartext without knowledge of the session key. Otherwise, an MKCS appears and is used by ordinary users (i.e., all users except the government) exactly as any secret-key cryptosystem is used. In particular, pairs of ordinary users must agree on a shared key before they can communicate. MKCSs should be secure against ordinary attacks; knowledge of only the algorithm, without either the session key or the master key, should not allow recovery of cleartext. (Ciphers that are merely weak, however obscure the attack, do not meet this last requirement.)

One reason for the study of MKCSs is that they could provide a less cumbersome alternative to key escrow [3] when third-party access is required, since the "escrow" occurs automatically when the cryptosystem is designed. Another stems from the argument we present below that efficient MKCSs are roughly equivalent to public-key cryptosystems (PKCSs, [1]) in which encryption is very fast and key-generation is slow. Every time the government puts forth an "official" secret-key cryptosystem, there is speculation that the system contains a secret "back door" that allows it to read all ciphertexts, i.e., that the system is in fact an MKCS. The rough equivalence we present makes this speculation less credible: Designing an MKCS with acceptable performance is tantamount to designing a new PKCS in which encryption is faster than is possible with currently known systems.

In an MKCS, there is a "set-up" or "generation" algorithm G. The input to G is a sequence of random bits, and the outputs are a master key K_m and a secret-key cryptosystem (E, D). The system (E, D) works exactly as any secret-key cryptosystem, except that the government can decrypt all ciphertexts by using K_m.

We now show a rough equivalence between master-key cryptosystems and public-key cryptosystems. First assume the existence of an MKCS. The generator G can be used as the key-generation algorithm in a PKCS. Every user A of the PKCS is assigned an instance $(K_m, (E, D))$ of the MKCS. K_m is A's private key; E is A's public key. If B wants to send x to A, he generates a random session key k and sends $y = E(x, k)$; by definition, A can decrypt y with K_m.

Next assume the existence of a PKCS. We give two constructions for MKCSs in which the government can recover the session key given the master key and a single block of known plaintext.

Our first construction is a master-key stream-cipher. Let MP and MS be the government's (public-key, private-key) pair. Users A and B create a session key k and then use the encryption (with the government's public-key MP) of the session key and a counter as a key-stream generator: $E_{MP}(\langle k, 1\rangle)$, $E_{MP}(\langle k, 2\rangle), E_{MP}(\langle k, 3\rangle), \ldots$. The government can use one block of known plaintext to recover one block of key-stream, which it can decrypt with MS to get k and generate the rest of the key-stream with MP.

Our second, more general construction yields a master-key block cipher (which could then be converted to a stream-cipher via one of the usual block-chaining methods). We use the public-key encryption algorithm as a keyed hash function in a variation on the 3-round Luby-Rackoff block cipher construction [2]. (This simplest version of the cipher is secure against known plaintext attacks but, like many similar constructions, is vulnerable to a number of well known chosen plaintext/ciphertext attacks. The simple version serves our purpose of establishing an inherent connection between MKCSs and PCKSs. More secure (and more complicated) versions of the cipher are obtainable but not necessary for this purpose.) Blocks of text are divided into left (L) and right (R) halves. In the encryption algorithm below, H is the hash function that is invertible by the government (using its private key), k is the session key, and $\oplus$ is exclusive-or.

$$\begin{aligned} R &= R \oplus H(L) \\ L &= L \oplus H(R \oplus k) \\ R &= R \oplus H(L \oplus k) \end{aligned}$$

The decryption function follows directly and is omitted.

Given one plaintext block (L_{plain}, R_{plain}), the corresponding ciphertext block (L_{cipher}, R_{cipher}), and the ability to invert H, the government can find

$$k = H^{-1}(R_{cipher} \oplus H(L_{plain}) \oplus R_{plain}) \oplus L_{cipher}$$

This "rough equivalence" implies that designing a practical MKCS entails designing a PKCS in which encryption is very fast. (G might be relatively slow, however, which could mean that one needs the resources of a government to generate an instance of an MKCS. Note that key-generation is indeed slower than encryption in some existing PKCSs, e.g., RSA [4]). Fast encryption might be achieved via short block-length. Is there a (fast and secure) PKCS with short blocks?

References

[1] W. Diffie and M. E. Hellman, *New Directions in Cryptography*, IEEE Trans. Inf. Theory, IT-22 (1976), 644–654.

[2] M. Luby and C. Rackoff, *How to Construct Pseudorandom Permutations from Pseudorandom Functions*, SIAM J. Comput., 17 (1988), 373–386.

[3] National Institute for Standards and Technology, *Escrowed Encryption Standard*, Federal Information Processing Standards Publication 185, US Dept. of Commerce, 1994.

[4] R. Rivest, A. Shamir, and L. Adleman, *A Method of Obtaining Digital Signatures and Public-Key Cryptosystems*, Comm. ACM, 21 (1988), 120–126.