ENCRYPTION OF STORED DATA IN NETWORKS: ANALYSIS OF A TWEAKED BLOCK CIPHER

IAN F. BLAKE[1,2], CYRIL GUYOT[1], CLEMENT KENT[1] AND V. KUMAR MURTY[1,2]

To Yufeng Ding.

ABSTRACT. Although encryption can protect the confidentiality of stored data, attacks on storage networks allow modes of use of oracles which render some ciphers weak. Some of these weaknesses are described, and potential solutions in the form of two classes of secure tweakable block ciphers are proposed. One class may be favored in some hardware implementations, while the second is preferred for software and will be competitive in some hardware solutions. These ciphers are candidates for standardization as IEEE/ANSI secure block storage ciphers. We also present theoretical analysis of the security of tweakable block ciphers which is a slight refinement of the work of Liskov, Rivest and Wagner [14].

1. INTRODUCTION

Although encryption has long been used to protect the confidentiality of transmitted data (data in transit), its use to protect stored data (data at rest) has been more limited, due in part to the perceived high cost of encrypting large amounts of data and in part to the lack of standards for such encryption. The progress of Moore's Law has now rendered encryption cheap enough that its use to protect terabytes of data at rest is now practical. However, the lack of standards, and of rigorous analysis of weaknesses in common storage environments, remains a barrier to widespread adoption of storage encryption.

A number of storage systems manipulate data at the sector level, rather than at the file level. There are challenges to providing confidentiality and resistance to data corruption which are harder to meet when the unit of work is the sector, rather than the file, and when existing storage systems must be used. For instance, a sector-level encryptor cannot add hashes, signatures, or in any other way expand the space occupied by the encrypted data. Thus, strengths of some existing file-level encryption systems ([3], [11]) do not carry over to the sector level.

Recently, the IEEE formed the Security in Storage Working Group with a goal of proposing standards for encryption of block data ([15],[2]). Analysis of proposed ciphers has begun [13] and has revealed interesting weaknesses. A number of these are summarized in this paper. There is an urgent need for sector-level cipher systems which are cryptographically strong and yet are feasible to implement given today's hardware and software.

[1] Kasten Chase Applied Research.
[2] University of Toronto.

Two classes of tweakable block ciphers are proven to have certain properties in the paper of Liskov, Rivest, and Wagner [14]. In this paper we note that only a subset of one of these classes resists attacks in the networked storage environment, and we present data on the relative efficiency of software and hardware implementations of the two classes. Members of either class may be suitable for an IEEE standard for sector-level encryption; characteristics of the operating environment may determine the choice between the two.

We also prove a result about the security of tweakable block ciphers which is a slight refinement of [14], Theorem 2.

2. Attacks on Data in Storage Networks

The analysis in this paper focuses on data in Storage Area Networks, or SANs. These networks grew as a way of providing high throughput (currently 2Gb/sec per channel) movement of data encapsulated as SCSI commands and payloads through a switched network which provides high reliability and quality of service guarantees [5]. They are used as a means of decoupling server computers from their disk and tape storage devices. This allows great flexibility in adding, sharing, and managing storage in large data centers, but it creates new modes of attack.

In a switched network all servers can potentially access all storage devices. Various means of limiting this access exist, such as zoning and LUN masking [4], [6], but a determined attacker can sometimes defeat these measures. A typical scenario is when an attacker has elevated privileges on a server that give him raw access to the storage devices; that is to say, the ability to read and write sectors on the storage device without mediation through a file system.

The most destructive short term use of such raw access is a Denial of Service attack in which large quantities of data are destroyed or corrupted. Annoying as such an attack may be, large SAN systems typically mirror and back up data with multiply redundant systems, so it is not too hard to recover data from before the time of a destructive DOS attack. In addition such attacks are likely to be noticed promptly as applications fail. A longer term solution to such attacks relies on strong authentication of storage users by the storage devices themselves and is beyond the scope of this paper (see [1], [16]).

A more subtle and more dangerous attack is to steal high-value data (topical examples being the theft of customer credit card numbers [9] or bank account PINs [7]). Alternatively, the stored data may be modified in some way advantageous to the attacker (e.g. by changing a salary field in a human resources database). Encryption of the stored data is commonly regarded as an appropriate protection of such confidential data. However, not all encryption systems do provide this protection against certain raw mode attacks.

Some of the simplest modes, such as ECB mode, are trivially vulnerable to dictionary attacks. Others, such as CTR mode, allow the attacker to observe changes in the ciphertext and deduce changes in plaintext. Recall that, given a counter i (based for instance on sector and cipherblock number and unique over the range of sectors on that storage device), the ciphertext is given as

$$C_i = E_K(i) \oplus P. \tag{1}$$

Later, new plaintext P' is written to the same location. Then the attacker forms

$$C_i \oplus C'_i = P \oplus P'. \tag{2}$$

This is a very large advantage to the attacker for very small effort. In addition, if the plaintext at a location i is known at one time, the value $E_K(i)$ is immediately derived, exposing all future data written to that location.

Other cipher modes are stronger than ECB and CTR, but are still open to other attacks. These attacks fall into two groups: malleability, and copy and paste attacks.

2.1. **Malleability.** If an attacker is able to change the ciphertext in a sector to achieve a desired effect in the decrypted plaintext, (such as a pay raise), the cipher may be called malleable. A nonmalleable cipher is one in which an attacker who lacks the key for the cipher has as probability ϵ of achieving the desired plaintext change through changing ciphertext, of same order of magnitude as the probability of guessing the plaintext given the ciphertext.

At the level of a single cipherblock, any strong symmetric cipher such as AES-128 is nonmalleable. However, at the sector level, malleability may arise. For instance, suppose that the attacker knows his salary is stored in cipherblock i and the company president's in cipherblock j . (Such knowledge may be deducible from file or database system structures or from traffic analysis). Simply transposing the two cipherblocks on the disk achieves a considerable pay increase in most cases.

One approach to preventing data translocation attacks is to encrypt each sector of data using an iterative mode such as CBC mode, and to use a unique IV (initialization vector) based on the sector number. This defeats translocation both within and between sectors, but leaves open another malleability attack.

Recall that in CBC mode we have:

$$C_{i+1} = E_K(C_i \oplus P_{i+1}). \tag{3}$$

where C_i and P_i are ciphertext and plaintext for cipherblock i, and $C_0 = IV$.

If the attacker knows his salary is in cipherblock $i+1$, and that flipping bit b in the plaintext will raise his salary by 2^b, he will flip bit b in C_i, e.g.

$$C'_i = 2^b \oplus C_i. \tag{4}$$

A decrypt of C_{i+1} now gives the desired result, although decrypt of C'_i yields pseudorandom text.

In order to forestall such elementary attacks, Hughes [12] asked whether sector nonmalleable ciphers could be defined, such that an attacker's change to any of the ciphertexts of the sector has a probability less than ϵ of yielding a desired plaintext in any location within the decrypted sector. Hughes has also stated a stronger version in which, with probability $1-\epsilon$ the decrypted plaintext of each cipherblock of the sector is indistinguishable from pseudorandom text. That is to say, almost any change to the encrypted sector results in complete gibberish in the entire decrypted sector.

In response to this, Halevi and Rogaway [**?**] have proposed a tweakable sector-level cipher which is thought to be sector nonmalleable. This property is achieved at some cost in efficiency of the resulting cipher.

Sector nonmalleable ciphers provide to disk sector encryption systems an analogue to the integrity properties of systems which add a signed hash to stored data, without requiring extra storage space. Replay attacks, in which an old ciphertext for a sector replaces newer ciphertext, are not prevented however.

2.2. **Copy and Paste.** A second class of attacks may be called copy and paste attacks. In these attacks, ciphertext is copied from one location, possibly modified, and pasted into a new location. In some attacks, the goal is to decipher the data, in others it is to make a known change to the data.

A typical environment for copy and paste attacks is a SAN with two servers. Server 1 is well protected by normal security procedures. On server 1 the attacker is an ordinary user whose access to the server's data is restricted by the operating and file systems' access controls. The attacker wishes to decipher or modify data belonging to another user, which he cannot normally see or change. The attacker has elevated privileges on server 2 which have allowed him to gain raw mode access through the SAN to all encrypted data on Server 1.

The data on Server 1 is protected by an encryption/decryption oracle which grants its services only to legitimate requests from server 1. In most attacks we assume the attacker has legitimate access to at least one sector S_A from the data for Server 1, and so can use the oracle for that sector.

A simple example of a copy and paste attack allows the attacker to decipher the first cipherblock of any sector S_B encrypted with CBC mode with a known IV function. Note that since we are assuming the cipher system has no extra space in which to save additional parameters such as a random IV for each sector, the IV must be some reproducible function $IV_B = F(B)$. If the attacker knows F, then he may copy the ciphertext for the first cipherblock of sector S_B to the beginning of his own sector S_A using raw mode, then ask the oracle to read the plaintext of S_A. The result will be

$$P_B \ \oplus \ F(B) \ \oplus \ F(A) \tag{5}$$

and since the attacker knows $F(B)$ and $F(A)$ he trivially finds the plaintext P_B.

We may try to block such attacks by making the IV function F unknown to the attacker. A very simple, and secure, choice is

$$F(A) = E_K(A), \tag{6}$$

which is exactly equivalent to prepending A to the sector plaintext and then doing CBC mode encryption of the extended text using an IV of 0; thus in systems optimized to do CBC in hardware this is a very easy secure IV to use.

In the most general case this protects the first cipherblock of CBC encrypted text from copy and paste. However, there is a special case which will unfortunately be quite common in disk storage. This is that at some time the attacker knows plaintext P_B in a number of locations B on the disk, due either to analysis of how file systems structure the disk on initial format or due to other sources of information such as traffic analysis. At that time the attacker performs the above copy and paste and observes the value $P_B \oplus F(B) \oplus F(A)$, and since P_B is known deduces the value

$$DIF(A,B) = F(B) \oplus F(A) \tag{7}$$

which is stored for future use. At some future time a new, unknown value P'_B is encrypted at location B as C'_B. The attacker copies C'_B to location A, decrypts through a legitimate read operation, and xors the stored value $DIF(A,B)$ to yield P'_B.

This attack may also be reversed; any chosen plaintext $P \oplus DIF(A,B)$ may be legitimately written to location A, and the resulting ciphertext may be copied to location B. When read by the legitimate owner of location B, it will yield plaintext P, as desired by the attacker.

Even if the value of plaintext P_B is not known to the attacker, the copy and paste attack described renders the difference between new and old text in a location easily deducible. The difference in values of (5) above for P_B and later P'_B is exactly $P_B \oplus P'_B$ so the first cipherblock of CBC mode is always as weak as CTR mode.

Independent of the IV function and the existence of known plaintext, copy and paste always allows the decryption of 97% of the data in a sector, since only the decrypt of the first cipherblock is dependent on the IV. The decrypt of the remaining 31 cipherblocks in a 512 byte sector yields the correct plaintext so long as the preceding cipherblock's ciphertext is unchanged, since

$$P_{i+1} \ = \ C_i \ \oplus \ D_K(C_{i+1}). \tag{8}$$

Thus, so long as the contents of sector B are moved without change to sector A and then read through the oracle, 31 out of 32 cipherblocks decrypt correctly.

In summary, CBC mode with a known IV function is vulnerable in the first cipherblock of a sector to trivial decipherment, or to replacement with ciphertext yielding a chosen plaintext. If the IV function is unknown to the attacker, any structure to the file systems placement of data on the disk that allows the attacker over time to observe the ciphertext of known plaintexts in cipherblock 1 still allows attacks. Any difference in the plaintext value at a cipherblock 1 location can be observed even if no single plaintext is known. And, any cipherblock 2 or higher is vulnerable to a malleability attack and to decryption using copy and paste. Thus, CBC mode is not recommended for sector-level ciphers in storage networks.

The above arguments allow us to immediately dispose of several other possible simply tweaked ciphers. As before with IVs, we may generate a tweak for any cipherblock i of any sector N using any tweak function $T = F(N,i)$, and then form either of two tweaked ciphers:

$$C_{N,i} \ = \ E_K(P_{N,i} \ \oplus \ T)(\text{tweak before}) \tag{9}$$

$$C_{N,i} \ = \ E_K(P_{N,i}) \ \oplus \ T(\text{tweak after}) \tag{10}$$

Such ciphers have been proposed in the past to protect standalone storage systems, in which an attacker may have raw mode access or partial oracle access but not both together. As they can be parallelized they can be highly efficient in hardware implementations, and if the tweak function is not known to an attacker either protects well against dictionary analysis on a stolen disk. Clearly in a networked storage environment copy and paste renders these simply tweaked ciphers unsafe.

One may ask whether tweaking before and after encryption is safe against copy and paste. The resulting cipher is:

$$C_{N,i} \;=\; T \;\oplus\; E_K(P_{N,i} \;\oplus\; T). \tag{11}$$

As this is the construction in Theorem 2 of [14], it might be thought that this construction has been proven secure so long as the function F is one of the ϵ-almost 2-xor-universal (ϵ-AXU2) set of functions of [14], such as hash127. However, by the arguments above, if the attacker knows the tweak function F, (11) is completely vulnerable to copy and paste.

If the tweak function F is not known to the attacker, however, (11) resists copy and paste even when the attacker has known plaintexts. This may be contrasted with the statement *"The tweak is not intended to provide additional uncertainty to an adversary. Keeping the tweak secret need not provide any greater cryptographic strength."* [14]. When subject to copy and paste attacks, cipher (11) must have at least one secret parameter for the tweak or it will be insecure. This seems to be implicitly acknowledged in section 3.1 of [14] where, when discussing the efficiency advantages of (11), the authors state it does require a longer key. Halevi (personal communication) has suggested that the tweak secret may be derived from the symmetric encryption key, for instance as $E_K(0)$. This obviates the need for additional secrets. The effect of this on the strength of (11) is not known.

An interesting question is what advantage the attacker may gain from creating the collection of values:

$$E_K(P_j \;\oplus\; T) \;\oplus\; E_K(P'_j \;\oplus\; T) \tag{12}$$

which the attacker generates by writing chosen plaintexts P_j and P'_j to one of his own locations and combining the resulting ciphertexts. As the location is constant, T is constant but unknown, and the attacker can observe the difference in the result of changing any specific bit in the input text, although this text is not known. Does a form of differential cryptanalysis apply here?

3. Two Classes of Tweakable Block Ciphers

The notion of a tweakable block cipher was introduced by Liskov et al [14]. They observe that most modes for a block cipher can be viewed as a mapping

$$E : K \times V \times M \to C$$

where $K \in \{0,1\}^k$ is a k-bit key, $V \in \{0,1\}^v$ is an initialization vector, $M \in \{0,1\}^*$ is a message of arbitrary length and $C \in \{0,1\}^*$ is the corresponding ciphertext. The feedback modes offer the property that the same message block will map to a different ciphertext block if its position within the plaintext differs. On the other hand they may be vulnerable to a copy-and-paste attack.

The idea of a tweakable cipher is to add a level of variability in the ciphertext in a more elegant and efficient manner than feedback modes. The mechanism is by adding an easily and efficiently computable 'tweak'. Thus the previous equation becomes:

$$\tilde{E} : K \times T \times P \rightarrow C$$

where $K \in \{0,1\}^k$, $T \in \{0,1\}^t$ and $P,\ C \in \{0,1\}^n$ are the key, tweak, plaintext and ciphertext respectively. Changing the tweak should be a less costly operation than changing the key and, in essence, any given setting of the tweak should give rise to a different family of block ciphers. As we have observed above, to avoid copy and paste attacks, it is necessary that there be some element of secrecy in relation to the tweak. The differentiation between uncertainty provided by the tweak and security provided by the key, and the clearer separation of the two concepts in that work [14], is an important contribution to the study of block ciphers. Previous work such as [8], [18] also inherently have the notion of a tweak although not so clearly delineated.

Several authors have addressed the security issue with particular reference to storage including [8], [10] and [17]. In the next section the key result of [14] is discussed and one particular version of the tweaked block cipher is introduced. Section 3 considers the security of this scheme by establishing results on the equidistribution of certain binary n-tuples. The final section comments on the results in terms of application to storage networks.

4. Tweakable block ciphers

The key results needed from [14] are reviewed. Using the notation of [14], define the security of a tweaked block cipher $\tilde{E}$ with block, length n and key K as $\mathrm{Sec}_E(q,t)$, the maximum advantage that an adversary can obtain when trying to distinguish between the tweaked encryption $\tilde{E}_K(\cdot)$ with a randomly chosen key K and a random permutation $\tilde{\Pi}(\cdot)$ on the space of binary n-tuples, when the adversary is allowed q queries to an unknown oracle (either $\tilde{E}_K(\cdot)$ or $\Pi(\cdot)$) when allowed computation time t. The advantage is defined as the difference between the probability the adversary outputs 1 when given oracle access to $\tilde{E}_K$ and the probability the adversary outputs 1 when given access to $\tilde{\Pi}$. It is argued that the block cipher may be considered secure when $\mathrm{Sec}_{\tilde{E}}(q,t)$ is sufficiently small.

A stronger version of this notion is $\mathrm{Sec}'_{\tilde{E}}(q,t)$, the maximum advantage when trying to distinguish between the pair of oracles $\tilde{E}_K(\cdot),\ \tilde{D}_K(\cdot)$ and the random permutations $\tilde{\Pi},\ \tilde{\Pi}^{-1}$ when allowed q queries and time t. The notions apply to a tweakable block cipher and we say that the tweakable block cipher is chosen-ciphertext secure if $\mathrm{Sec}_{\tilde{E}}(q,t)$ is sufficiently small, in which case it is referred to as a strong tweakable block cipher.

The key result of [14] of interest here is as follows. Let $\epsilon > 0$ and let $\mathcal{H}$ denote a family of functions $\{0,1\}^t \rightarrow \{0,1\}^n$ with the property that for any $x, y \in \mathbb{F}_{2^t}$ and any $z \in \mathbb{F}_{2^n}$

$$\mathrm{Pr}_h[h(x) \oplus h(y) = z] \leq \epsilon \tag{13}$$

where the probability is taken over $h \in \mathcal{H}$ chosen uniformly.

If $z = 0$, we have to exclude $x = y$ for otherwise the probability on the left of (13) is 1. This does not seem to be stated explicitly in [14] but is clearly a necessary restriction.

Following [14], we shall call such a family an $\epsilon - AXU_2$ (or ϵ-almost 2-xor-universal). The condition is a form of equidistribution to prevent an attacker to gain information from any skewness. We then have:

Theorem ([14] *Theorem 2) Let $\tilde{E}_{K,h}(T,M) = E_K(M \oplus h(T)) \oplus h(T)$ and let $\mathcal{H}$ be an $\epsilon - AXU_2$ family with $\epsilon \geq 1/2^n$. Then $\tilde{E}$ is a strong tweakable block cipher. Specifically*

$$Sec'_{\tilde{E}}(q,t) \leq Sec'_E(q,t) + 3\epsilon q^2.$$

We can actually work with a variant of the $\epsilon - AXU_2$ space of functions. For $\epsilon > 0$, and for each y, and for any choice of $\{z_{x,y}\}$, consider the condition

$$\sum_{x \neq y} Pr_h[h(x) \oplus h(y) \ = \ z_{x,y}] \ \leq \epsilon(q-1)$$

where the summation is over values of x excluding y. Let us call a function h satisfying this condition ϵ-pseudo 2-xor-universal ($\epsilon - PXU_2$). Clearly an $\epsilon - AXU_2$ function is also an $\epsilon - PXU_2$ function. However, the converse need not hold. We then have the following result.

Theorem 4.1. *Let $\tilde{E}_{K,h}$ be as above and let $\mathcal{H}$ be a $\epsilon - PXU_2$ family with $\epsilon \geq 1/2^n$. Then $\tilde{E}$ is a strong tweakable block cipher and*

$$Sec'_{\tilde{E}}(q,t) \leq Sec'_E(q,t) + 4\epsilon q^2.$$

For the tweaks of interest here we will write $h_k(x)$ to indicate that the tweak depends in some simple way on a secret key k and storage location information x and thus the averaging indicated in the theorem, for a fixed form of tweak, is uniform over the set of possible secrets k.

We have in mind here the storage scenario with the parameters N, the sector number on a hard drive (and may typically assume values up to approximately 2^{40}) and i is the cipher block number of the plaintext within the sector. Clearly the situation is completely general but for illustrative purposes we will also assume we are using AES with block and key size of 128 bits and for a sector length of 512 bytes there are 32 blocks to a sector - hence i is in the range $[0,31]$. Thus in these tweaks k is a secret element which will be taken to be an element in the finite field $\mathbb{F}_{2^{128}}$ with 2^{128} elements, realized as the set of polynomials $\mathbb{F}_2[x]/(p(x))$ where $p(x)$ is an irreducible polynomial of degree 128 where $\mathbb{F}_2$ is the finite field with two elements. For efficient arithmetic one might choose $p(x)$ to be a polynomial of low weight - in this case, since there is no irrecducible trinomial over $\mathbb{F}_2$ the pentanomial $x^{128} + x^7 + x^2 + x + 1$ will be used. We associate with the binary 128-tuple $(a_0, a_1, a_2, \ldots, a_{127})$ the "polynomial"

$$\sum_{i=0}^{i=127} a_i x^i \in \mathbb{F}_{2^{128}} \sim \mathbb{F}_2[x]/(p(x)).$$

With the tweak $h_k(\cdot)$ the encryption process is then

$$\text{Ciphertext} = E_K(\text{Plaintext} \oplus h_k(i,N)) \oplus h_k(i,N)$$

for the fixed and secret k and the corresponding decryption is

$$\text{Plaintext} = D_K(\text{Ciphertext} \oplus h_k(i,N)) \oplus h_k(i,N)$$

where $E_K(\cdot)$ and $D_K(\cdot)$ are the block encryptions and decryptions with key $K \in \{0,1\}^n$ (and for AES, we assume $n = 128$). One can easily see that this algorithm is only secure if the original random element k is kept secret.

If an attacker manages to build a table of $h_k(i,N) \oplus h_k(j,N')$ then a copy and paste attack would still be successful. To see this suppose an attacker has access to the decryption oracle at (j,N') and wants to decrypt the ciphertext at (i,N). The attacker first copies the ciphertext $E_K(\text{Plaintext}\oplus h_k(i,N))\oplus h_k(i,N)$ and XOR's it with $h_k(i,N)\oplus h_k(j,N')$. The attacker now has $E_K(\text{Plaintext}\oplus h_k(i,N))\oplus h_k(j,N')$ and can apply the decryption oracle to get

$$D_K(E_K(\text{Plaintext} \oplus h_k(i,N)) \oplus h_k(j,N') \oplus h_k(j,N')) \oplus h_k(j,N')$$

$$= \text{Plaintext} \oplus h_k(i,N) \oplus h_k(j,N').$$

The attacker now has only to XOR this with $h_k(i,N) \oplus h_k(j,N')$ to retrieve the Plaintext.

As a consequence and the previously quoted theorem, it is clear that the security of this algorithm requires that

$$(i, j, h_k(i,N) \oplus h_k(j,N')) \text{ for } i,j \in [0,\text{Max}]$$

be "equidistributed". The equidistirbution properties of the tweaks of interest are considered inthe following section.

5. Equidistribution properties for the tweak

Clearly a great many tweaks are possible and to prove there security it is sufficient to consider their equidistribution proerties in the sense specified by the previous theorems. The security of one (or two) specific tweaks will be considered in this section.

5.1. **A multiplicative tweak.**

Consider the function defined by the equation:

$$h_k(i,N) \;=\; k \cdot f(i,N).$$

In the following f is taken to be injective over the range of N and i. A simple example would be $f(N,i) = (256N + i)$.

The computation of f is one of integer multiplication and addition modulo $2^{128}-1$ and then converted to its binary expansion to yield an element of $\mathbb{F}_{2^{128}}$.

To establish the security of the tweaked system it is necessary to establish the distribution properties of the system:

$$\{i, j, k \cdot i \oplus k \cdot j\} \;\text{ for }\; i,j \in \mathbb{F}_{2^{128}}.$$

Notice first that there is a natural association between elements of $\mathbb{F}_{2^{128}}$ and the integers: for the element $\beta \in \mathbb{F}_{2^{128}},\ \beta = \sum_i b_i x^i$, with $b_i \in \mathbb{F}_2$, viewing the b_i to be integers in $\{0,1\}$, we can associate to each β a unique integer:

$$\beta \ \mapsto \ \omega(\beta) \ = \ \sum_{i=0}^{n-1} b_i 2^i.$$

Thus

$$0 \leq \ \omega(\beta) \ < 2^n.$$

We consider equidistribution of the following sequences. The first is

$$(i,j) \ \mapsto \alpha i + \alpha j.$$

Here, i, j are given as functions

$$i = f(N_i, \ell_i),\ j = f(N_j, \ell_j)$$

and where $1 \leq N_i, N_j \leq 2^{40}$ represents the sector number and $1 \leq \ell_i, \ell_j \leq 32$ is the cipher block number within a sector. Each cipher block consists of 128 bits and a sector is 512 bytes.

We need to study

$$Pr_h[h(x) \oplus h(y) = z]$$

for

$$h \in H, x, y \in \mathbb{F}_{2^{128}}$$

For this specific "tweak", this is equivalent to studying

$$Pr_k[kx \oplus ky = z]$$
$$= Pr_k[k(x \oplus y) = z] = Pr_k[kt = z]$$

If $z = 0$, by assumption $x \neq y$ and so $t \neq 0$. The only possible value of k is $k = 0$. If $z \neq 0$, then $t \neq 0$ and $kt = z$ is equivalent to $k = zt^{-1}$ and

$$Pr_h[h(x) \oplus h(y) = z]$$
$$= \frac{\#[k \in \mathbb{F}_{2^{128}} | kt = z]}{\#[k \in \mathbb{F}_{2^{128}}]} = \frac{1}{2^n}$$

5.2. **An exponential tweak.** The multiplicative tweak discussed above has the advantage of being very simple and efficient to implement. Mathematically, there are clearly many variants that one could still produce security estimates for and use.

One particular example is an exponential version of the multiplicative tweak discussed above. In this we set

$$h_k(i) \ = \ k^{256N+i}.$$

In this case, we have the obvious estimate

$$Pr_k[h_k(i) \oplus h_k(j) \ = \ z] \ \leq \ \max(i,j)/2^n.$$

Thus, as long as q is small compared to

$$2^{n/2}/\max(i,j)^{\frac{1}{2}}$$

the tweak is secure. Unfortunately, i and j can be as large as 2^n and so this essentially means that the number of queries will have to be very small. It is an interesting question to ask whether the above estimate can be improved. Some heuristics seem to suggest that for "most" values of i and j, the bound can be significantly improved. It would be useful to have a proof of this together with an estimate of the exceptional set. However, such a proof, as well as an improved bound valid for all i and j, seem to be difficult mathematical problems.

We suggest that the strong tweakable block cipher in (11) above resists copy and paste when the tweak function is unknown to the attacker. In [14] it is noted that ϵ-AXU2 functions such as hash127 would allow (11) to run in about 1.5 times the cycles required for a single AES-128 encryption. In this paper, we discuss allowing other multiplicative tweak functions such as

$$F(N,i) = k \cdot (N + 2^c i) \tag{14}$$

where multiplication is in the finite field, k is a 128 bit tweak secret, and the addition in the brackets is addition in the finite field of binary n-tuples with the same bit pattern as the integers N and $2^c i$. We show that (11) is also a strong tweakable block cipher with secret k. We call this cipher LRW-2 below.

In the most general case Galois field multiplication of 128 bit elements is significantly slower than hash127, so LRW-2 would seem an inefficient choice of cipher. However, if sufficient memory is available to store a table of the values $k \cdot (2^j)$, for $j = 0$ to $c + 5$ (all of which are calculated on the first encryption using k), then subsequent tweaks for successive cipherblocks may be calculated from the tweak for the preceding cipherblock by a few xor operations using values from the table. In one software implementation, this resulted in the cycles to encrypt a sector using LRW-2 being about 1.3 times the cycles to do AES-128 ECB alone. In addition, if the encryptor mostly works on one or a few storage devices, so that the above tables may be cached, calculations of the tweak for later sector writes will be even shorter (the small average Hamming weight of $N \oplus N'$ when N and N' are close means few XORs are required to calculate $k \cdot N'$ from $k \cdot N$). This reduces the ratio between sector encryptions using LRW-2 and simple AES to between 1.1 and 1.2.

Thus, for software implementations, and for some hardware systems that can use the above optimizations, LRW-2 may have a reasonable average cost. If, however, the implementation has little memory for the tables (about 1KB per k value) or must service many different storage devices in random order, LRW-2 may be less efficient.

5.3. **Encrypt-Tweak-Encrypt.**

An alternative strong tweakable block cipher is given in Theorem 1 of [14]:

$$C \;=\; E_K(T \;\oplus\; E_K(P)). \tag{15}$$

This cipher is called LRW-1 below. In software implementations it will take more than twice the cycles needed for a simple encryption, and so is less efficient than (11) using hash127 or (14). However,the coding is minimal if the tweak function is simple, and in hardware (see below) it may be quite efficient.

Unfortunately, Joux has shown [13] that if the tweak T is known to the attacker, a copy and paste attack can be mounted against LRW-1. We illustrate a simple case and refer the reader to [13] for details of the more general attack. Suppose the tweak function is the disk location itself, that is $T_B = B$. The attacker is assumed to have legitimate access to locations A_0 and A_1. He attempts to acquire access to location $A_2 = A_0 \oplus A_1 \oplus B$. If granted, he performs:

[1] Copy C_B to A_0 - raw mode copy and paste B to A_0
[2] Read $P_0 = D_K(A_0 \oplus B \oplus P)$ - oracle read from A_0
[3] Write $C_1 = E_K(A_0 \oplus A_1 \oplus B \oplus P)$ - oracle write to A_1
[4] Copy C_1 to A_2 - raw mode copy and paste
[5] Read $P_2 = D_K(A_0 \oplus A_1 \oplus A_2 \oplus B \oplus P) = P$ - oracle read from A_0

The more general attack of Joux succeeds against any tweak function T known to the attacker, if the attacker has access to 2 MB of disk.

We therefore conclude that LRW-1 must use a tweak function $T = F(N, i)$ unknown to the attacker to resist copy and paste. In addition, as the above simple attack shows, the function F must not be linear in the location, which rules out tweaks such as (14) above. The question of which functions yield tweaks for LRW-1 secure against copy and paste is not answered in this paper; in the calculations below we assume a function like hash127 is used.

5.4. **Efficiency of Tweakable Block Ciphers.** In pipelined hardware implementations of AES-128, 10 rounds will be performed each in its own silicon. If the time to perform 1 round is t, the time to encrypt 32 cipherblocks in ECB mode will be 42t that is, a full sector can be encrypted in 4.2 times the cycles required to encrypt one cipherblock.

For a fully pipelined implementation of LRW-1 with a tweak function requiring time $M_1 t$ to calculate, twice as many gates will be required as for ECB support, and time $(53 + M_1)t$ is required to encrypt one sector. So using twice as many gates, a fully pipelined LRW-1 system encrypts a sector in

$$(53 + M_1)/42 \simeq 1.26 + M_1/42 \tag{16}$$

more cycles than a fully pipelined ECB sector encryptor. For AES-256 the ratio is

$$61/46 + M_1/46 \simeq 1.33 + M_1/46.$$

Thus pipelining reduces performance impacts of LRW-1 at the expense of using twice as many gates.

For LRW-2 with AES-128, assume time $M_2 t$ is required to do the initial Galois field multiply, 2 cycles for tweaking before and after, and that calculation of tweaks for successive cipherblocks can be done in parallel with other steps. Then the ratio between LRW-2 and pipelined sector ECB will be

$$(M_2 + 44)/42. \tag{17}$$

The value of M_2 will depend on specifics of the system design but for processing of multiple sectors with a table cache M_2 may average as low as 4 or 5, and would thus be about as fast as pipelined LRW-1. For AES-256 the ratio is

$$(M_2 + 48)/46. \tag{18}$$

For low average M_2 pipelined LRW-2 may be more efficient than pipelined LRW-1.

6. Timings for the tweak

The time needed for a multiplication over $\mathbb{F}_{2^{128}}$, when the irreducible polynomial is chosen to be $X^{128} + X^7 + X^2 + X + 1$ is about 0.7 microseconds on a Pentium IV-M 2GHz. The time spent on doing one field addition on our machine is about .07 microseconds. On the same machine AES ran at about 90MB/s, corresponding to .17 microseconds spent on each block.

Since the tweak needs to be done for all the blocks, it is clear that computing the field multiplication for every block is not efficient enough to make it useable. - Such an implementation would give a throughput of 17MB/s. -

As a consequence, one has to use precomputations. For example if one knows $k \cdot$"i", then $k \cdot$"$(i+1)$" which is the next block is easy to compute and can be done in one addition. Note that $i + 1$ is an addition of integers, whereas the multiplication is done in $\mathbb{F}_{2^{128}}$.

Since encrypting/decrypting files on a hard drive has to be done one sector at a time, and since a sector is usually 512 bytes - or 32 blocks -, then the tweak requires one multiplication and 32 additions per sector. The time spent tweaking on one sector would hence be 32*0.07+0.7 = 2.94 microseconds. The time spent in AES would be 32*0.17 = 5.44 microseconds.

Furthermore, in the case when the file being read spans several sectors, it is not necessary to recompute the field multiplication, and higher throughputs are achievable.

An unoptimized implementation of this algorithm yields a throughput of about 50MB/s for one sector. A similar implementation of the Encrypt-Tweak-Encrypt algorithm yields 37MB/s.

7. Conclusions

Many existing cipher modes such as CBC, ECB, CTR, and others are not suitable for providing secure confidentiality in a networked storage environment, due to malleability and copy and paste attacks. Alternative ciphers, based on the concept of strong tweakable block ciphers, do resist these attacks and can be made reasonably efficient in both software and hardware implementations.

We discussed two such ciphers, labelled LRW-1 (encrypt-tweak-encrypt) and LRW-2 (tweak-encrypt-tweak).

LRW-1 will be 50%-80% slower in software than LRW-2. Its efficiency in hardware will depend sensitively on a choice of a tweak function both efficient and secure against copy and paste.

LRW-2 requires extra code or careful hardware design for Galois field multiplication, but will run faster than LRW-1 in software and in some hardware implementations, especially where many sectors are encrypted with few keys and memory is not limited.

Both ciphers seem, based on present understanding, to be possible candidates for an IEEE/ANSI standard cipher for sector level encryption. Neither seems to be encumbered with patent issues. It is to be hoped that experienced cryptanalysts will challenge these ciphers to determine whether hidden weaknesses exist. Based on our understanding to date, we propose that LRW-2 be the basis for the IEEE/ANSI P1619 standard.

It has been shown that one proposed family of tweaks indeed consists of $\epsilon - AXU_2$ functions and hence the resulting tweaked block ciphers are secure in the sense described. For the multiplicative family each new tweak in going from one sector to the next requires only one addition in $\mathbb{F}_{2^{128}}$ and for the exponential family only one multiplication. As such, the tweaks are very efficient to compute and typically much more efficient than changing the key. They would seem to provide a good solution for the level of variability and security needed for the storage application.

8. Acknowledgements

We would like to thank Donald Beaver, Rob Ewan, Shai Halevi, Serge Plotkin, and Yamian Quintero for very helpful discussions on various aspects of this work. We would also like to thank Jim Hughes who stimulated this avenue of research.

References

[1] Alain Azagury, Ran Canetti, Michael Factor, Shai Halevi, Ealan Henis, Dalit Naor, Noam Rinetzky, Ohad Rodeh, Julian Satran, A Two Layered Approach for Securing an Object Store Network, First International IEEE Security in Storage Workshop, pp. 10-23 Dec. 2002.

[2] Donald Beaver. Network Security and Storage Security: Symmetries and Symmetry-Breaking. First International IEEE Security in Storage Workshop , pp. 3-9 Dec. 2002

[3] Matt Blaze, A Cryptographic File System for Unix. ACM Conference on Computer and Communications Security, Fairfax, VA, November 1993

[4] John Chirillo , Scott Blaul . Storage Security: Protecting, SANs, NAS and DAS, John Wiley and Sons; Dec. 2002

[5] Tom Clark. Storage Area Networks: A Practical Reference for Implementing Fibre Channel SANs. Addison Wesley. August 1999.

[6] Thomas Clark. IP SANS: A Guide to iSCSI, iFCP, and FCIP Protocols for Storage Area Networks. Addison Wesley. Nov. 2001

[7] Consumers, banks clash as ATM fraud escalates. http://www.pittsburghlive.com/x/tribune-review/business/$s_1 30795.html$ April 2003

[8] Paul Crowley, Mercy: A fast large block cipher for disk sector encryption, in *Fast Software Encryption, 7th International Workshop*, Lecture Notes in Computer Science, Springer-Verlag, pp. 49-63, 2000.

[9] Hacker hits up to 8M credit cards http://money.cnn.com/2003/02/18/technology/creditcards/ Feb, 2003; http://www.gartner.com/resources/113200/113282/113282.pdf

[10] Shai Halevi, An observation regarding Jutla's modes of operation, preprint.

[11] J. Hughes, M. O'Keefe, C. Feist, S. Hawkinson, J. Perrault, D. Corcoran, A Universal Access, Smart-Card-Based, Secure File System , Atlanta Linux Showcase , October, 1999, Atlanta, GA

[12] James Hughes. Call for Algorithms, Security in Storage Work Group. www.cryptobroker.net April 2002

[13] Antoine Joux, Cryptanalysis of the EMD Mode of Operation. EuroCrypt 2003, May 2003.

[14] Moses Liskov, Ronald L. Rivest and David Wagner, Tweakable block ciphers, CRYPTO 2002, Lecture Notes in Computer Science, ed. M. Yung, Springer-Verlag, vol. 2442, pp. 31-46, 2002.

[15] Project 1619: Standard Architecture for Encrypted Shared Storage Media August 14, 2002 - http://www.siswg.org/

[16] Benjamin C. Reed, Mark A. Smith, Dejan Diklic. Security Considerations When Designing a Distributed File System Using Object Storage Devices. First International IEEE Security in Storage Workshop ,pp. 24-34 Dec. 2002

[17] Phillip Rogaway, The EMD mode of operation (a tweaked, wide-blocksize, strong PRP), submission to IEEE Security in Storage Working Group, (`www.siswg.org`).
[18] R. Schroeppel, The hasty pudding cipher, preprint, (`www.cs.arizona.edu/ rcs/hpc`)

9. Appendix: Proof of Theorem 4.1

Theorem 4.1. Let $\tilde{E}_{K,h}$ be as above and let $\mathcal{H}$ be a $\epsilon-PXU_2$ family with $\epsilon \geq 1/2^n$. Then $\tilde{E}$ is a strong tweakable block cipher and

$$\text{Sec}'_{\tilde{E}}(q,t) \leq \text{Sec}'_{E}(q,t) + 4\epsilon q^2.$$

Proof. The proof of the theorem is very similar to that of [14], Theorem 2. We shall outline the argument and give details only in those aspects that differ from [14]. We let Pr_0 denote the probability measure when the adversary A interacts with the tweaked encryption $\tilde{E}_{K,h}$ and Pr_1 when A interacts with a tweaked random permutation $\tilde{\Pi}$.

We recall that $\tilde{\Pi}$ is a family of random permutations parametrized by T. We assume that A does not make repeated or redundant queries. Each query may be represented by a pair (T, M) where T is the tweak input and M is the plaintext input by A. The oracle responds by returning C. Define the random variables

$$N = M \ \oplus \ h(T)$$

and

$$B \ = \ C \ \oplus \ h(T).$$

Denote the sequence of queries by (T_i, M_i, C_i) for $i = 1, 2, \cdots, q$. As A does not make redundant queries, we may suppose that these are distinct triples.

We will show that if E_K is a randomly chosen permutation Π, then $\tilde{E}_K$ is a secure tweakable block cipher. Once this is established, the same argument as in [14] will show that for an arbitrary E_K that is computationally indistinguishable from Π, $\tilde{E}_K$ is a secure tweakable block cipher.

First, using the fact that $\mathcal{H}$ is a $\epsilon - PXU_2$ family, we establish an estimate for the probability that for some $i \neq j$, we have

$$N_i \ = \ N_j$$

or

$$B_i \ = \ B_j$$

(Following [14], let us call such an event *bad.* More precisely, if $i, j < n$, we call this Bad_n.) This is the case whether we are working in the measure Pr_0 or Pr_1.

Consider Pr_1 and, for example, the first condition. As indicated in [14], Lemma 2, when the oracle is $\tilde{\Pi}$, the choice of function h is independent of the transcript of queries and responses. Hence, we can defer the choice of h until after all the queries are completed. This means that the triples (T_i, M_i, C_i) are all fixed. Now, the above condition is equivalent to having

$$h(T_i) \ \oplus \ h(T_j) \ = \ M_i \ \oplus \ M_j$$

for some $i \neq j$. By the $\epsilon - PXU_2$ condition, the total probability is bounded by

$$\sum_{1 \leq i < j \leq q} Pr_h[h(T_i) \ \oplus \ h(T_j) \ = \ M_i \oplus M_j].$$

From this sum, we can drop those i, j for which $T_i = T_j$ as $Pr_h = 0$ in that case. We thus see that the above sum is

$$\leq \sum_{x \neq y} Pr_h[h(x) \oplus h(y) \ = z_{x,y}] \ \leq \ \epsilon q(q-1).$$

as $\mathcal{H}$ is an $\epsilon - PXU_2$ family. The same estimate holds if the N_i are replaced by the B_i. This shows that under Pr_1, the probability of a bad event is

$$\leq \ 2\epsilon q(q-1).$$

Now consider the same event under the measure Pr_0. For each i, consider the event that $N_1, \cdots, N_{i-1}$ are distinct and $N_i = N_j$ for some $j < i$. It occurs with probability $Pr_0(Bad_i|\overline{Bad_{i-1}})$. We want to estimate

$$\sum_{1 \leq i < q} Pr_0(Bad_i|\overline{Bad_{i-1}}).$$

By [14], Lemma 1, we have

$$Pr_0(Bad_i|\overline{Bad_{i-1}}) \ = \ Pr_1(Bad_i|\overline{Bad_{i-1}}).$$

Under Pr_1, the condition Bad_i conditioned on $\overline{Bad_{i-1}}$ means that for some $j < i$, we have

$$h(T_i) \oplus h(T_j) \ = \ M_i \oplus M_j.$$

As we are using the measure Pr_1, the probability of this happening is

$$\sum_{j<i} Pr_h[h(T_i) \ \oplus \ h(T_j) \ = \ M_i \ \oplus \ M_j]$$

which by the $\epsilon - PXU_2$ condition is

$$\leq \epsilon(q-1).$$

Now summing over i shows that the probability that $N_i \ = \ N_j$ for some $i \neq j$ is $\leq \epsilon q(q-1)$. To complete the argument, it suffices to estimate the probability that $B_i \ = \ B_j$ for some $i \neq j$, but this is done as in [14] and does not involve the $\epsilon - PXU_2$ condition. Assuming that $\epsilon \geq 2^{-n}$, this probability is $\leq \epsilon q(q-1)$. Putting this together gives the estimate

$$Pr_0[Bad] \ \leq \ 2\epsilon q(q-1).$$

By [14], Lemma 1, we know that

$$Pr_0[\overline{Bad}] \ = \ Pr_1[\overline{Bad}].$$

It follows that as in [14],

$$Sec'_{\tilde{E}'}(q,t) \ \leq \ 4\epsilon q(q-1).$$

Now, as in [14], the result follows.

Kasten Chase Applied Research, Mississauga, Ontario, Canada and University of Toronto, Toronto, Ontario, Canada