Counter Mode Security: Analysis and Recommendations

David A. McGrew
Cisco Systems, Inc.
mcgrew@cisco.com

November 15, 2002

Abstract

In this document we describe Counter Mode (CM) and its security properties, reviewing relevant cryptographic attacks and system security aspects. This mode is well understood and can be implemented securely. However, we show that attacks using precomputation can be used to lower the security level of AES-128 CM below the recommended strength for ciphers if the initial counter value is predictable. For this reason, AES-128 CM counter values should contain a 64-bit unpredictable field. We describe how this can be easily done, and make other implementation recommendations.

1 Counter Mode

In Counter Mode (CM), a block cipher is used as a keystream generator, and the keystream is bitwise exclusive-ored into the plaintext to produce the ciphertext. Symbolically,

$$c_i = p_i \oplus s_i \tag{1}$$

where c_i, p_i, and s_i denote the i^{th} ciphertext bit, plaintext bit, and keystream bit, respectively. CM keystream is generated by applying the block cipher to a set of distinct input blocks, as illustrated in Figure 1. Trailing keystream bits not needed for encryption are discarded. In the decryption process, the keystream is exclusive-ored into the ciphertext to give the plaintext. Successive input blocks are generated by a next-counter function, which can be a simple operation such as an integer increment modulo 2^w, where w is a convenient register width. The details of the next-counter function are unimportant; that function does not provide any security properties other than the uniqueness of the inputs to the block cipher[1]. These details could be important if the block cipher is a legacy cipher (e.g. DES), but they are not important when considering ciphers like AES [1] that are believed to be secure.

[1]This is good engineering practice, since the design of a next-counter function that would ensure security even when the block cipher itself was insecure would be difficult at best. In contrast, if the block cipher is secure, counter mode is secure regardless of the details of the next-counter function.

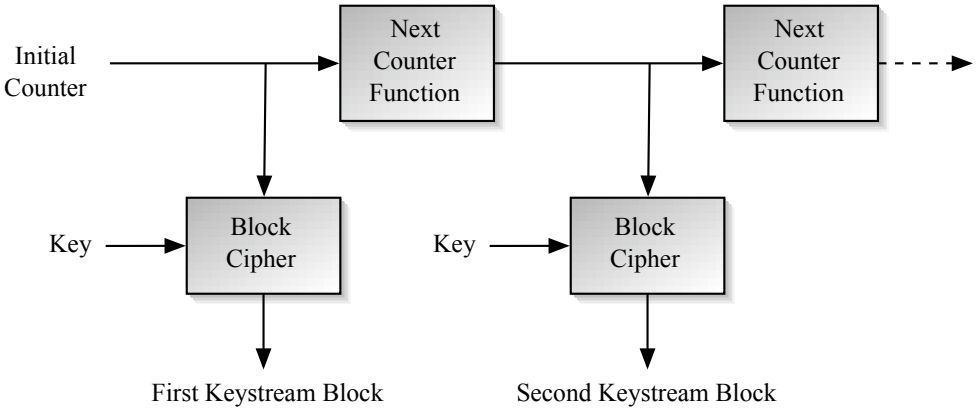

Figure 1: A schematic illustration of keystream generation using counter mode.

CM has significant implementation advantages that are well described elsewhere [7]. We do not discuss these advantages in order to concentrate on security issues. In the following, we first review several system-security properties: malleability, key uniqueness, counter uniqueness, the multiple-sender case, and random IV considerations. We then discuss cryptographic security, including relevant security proofs, precomputation attacks, and means of defending against those attacks. Finally, we conclude with recommendations.

2 System Security

There are some aspects of CM that can affect the security of systems that use it. These aspects are not unique to CM; they apply to all additive ciphers, including RC4 [11] and SEAL [12].

2.1 Malleability

Counter mode is malleable: the ciphertext can be manipulated so that it decrypts to any message of an attacker's choosing, whenever the attacker knows the plaintext. This property is easy to see from Equation 1; to flip a bit of the decrypted plaintext, flip the corresponding bit of the ciphertext. Because of this property, CM should be used in conjunction with a message authentication code.

2.2 Unintentional Key Reuse

For each particular CM key, CM generates a particular keystream. Thus, the use of the same CM key value two distinct times will result in the use of the same keystream to encrypt two distinct plaintexts. Since such a 'two-time pad' situation reveals information about the plaintexts[2], it is

[2]The NSA's VENONA project [6] provides a dramatic and historically important illustration of the exploitation of a two-time pad.

imperative that unintentional re-use of the cryptographic keys be avoided. For this reason, it is desirable that CM be used with an automated key management system.

2.3 Counter Uniqueness

Using a counter more than once with the same key will cause CM to generate two identical keystream segments, resulting in a two-time pad situation. For this reason, it is imperative that a CM implementation ensure the uniqueness of the counter values. A conservative system should include counter generation within the core cryptographic module.

2.4 Multiple Senders

If CM is used in a system in which multiple senders are performing encryption with the same key, then it is imperative that the system ensure that all counter values used with that key are unique, across all senders. A simple way to accomplish this is to include a sender identifier field in the counter. Unique sender identifier values can be established by the key management system.

However, having multiple senders with a single key can be a false economy. Most cryptographic systems require anti-replay protection, and that protection requires a per-sender data context. In addition, a system with multiple senders using the same key must coordinate the key usage limit (Section 3) across those senders.

2.5 Initialization Vector

Counter mode as described above (and as conventionally implemented) does not use a random initialization vector (IV). In contrast, the commonly-used CBC mode is conventionally implemented with a random IV, and thus does not have the unique counter requirement described above. The avoidance of a random IV is in some cases a security advantage for CM. Systems for which the generation of a per-packet random IV is onerous often use pseudorandom methods, sometimes leading to failures of the cipher[3]. By not requiring a random per-packet input, CM avoids this potential failing.

3 Security Proofs

The security of CM has been proven by Bellare et. al. [3], using the assumption that the block cipher is indistinguishable from a random permutation (as is conventional in the analysis of block cipher modes of operation). Their analysis shows that the security of CM can be *better* than that

[3]An example of this is the notorious chosen-plaintext weakness of CBC when the last ciphertext block of one packet is used as the IV of the next packet.

of CBC. It provides an upper bound on the number of keystream blocks that can be generated using a single CM key, and provides us with confidence to use CM up to that limit.

The standard definition of confidentiality in theoretical cryptography is that of *indistinguishabity from random*: a cipher is considered secure against an adversary if that adversary cannot distinguish its ciphertext from the output of a truly random source. The capability of the adversary to distinguish the cipher from random is captured in the definition of *advantage*, which is the difference of the adversary's true positive probability and false positive probability[4].

To apply the analysis of Bellare et. al. to AES CM, we need to tie together several results. The counter mode cipher in that paper (the XOR and CTR methods of Theorems 11 and 13) assumes that the encryption function is a pseudorandom function (PRF), not a pseudorandom permutation (PRP) like AES[5]. To apply that analysis to a PRP, it's necessary to apply Proposition 8 from that paper:

$$\text{Adv[PRF]}(t, q) \leq \text{Adv[PRP]}(t, q) + q^2 2^{-l-1}. \tag{2}$$

Here q is the number of known plaintext blocks, t is the time used in the attack, l is the bit-length of the cipher block, and the advantage when distinguishing mechanism X from random is denoted as Adv[X]. Plugging in the block width $l = 128$ for AES and the value $\epsilon = \text{Adv[PRF]}$ and assuming that $\text{Adv[PRP]}(t, q) = 0$ (that is, AES can't be distinguished from a PRP), we get

$$q = \sqrt{\epsilon} \cdot 2^{64.5} \tag{3}$$

In words, as long as we don't generate more than $q \simeq 2^{64}$ keystream blocks, then the adversary's advantage is limited to ϵ.

It is easy to see where the upper bound on the number of keystream blocks comes from. A true random source would likely produce two identical blocks after generating a total of 2^{64} blocks, while AES CM will not. Thus the cipher can be distinguished from randomness after about 2^{64} blocks have been output. The fact that there are no better distinguishing attacks as long as this bound is respected is shown in [3].

4 Precomputation Attacks

Two important attacks that can be effective against CM and other additive ciphers are the key collision (KC) attack [4] and Hellman's time-memory tradeoff (TMTO) attack [2]. These attacks are shortcuts over exhaustive key search that trade a storage requirement off against decreased computational effort. They can be used against *any* cipher, even ones that are not statistically defective[6].

[4] A true positive is the case in which the adversary correctly identifies a cipher, and a false positive is the case in which the adversary believes that a random source is the cipher.

[5] A permutation is an invertible function, i.e. a block cipher. In contrast, a random function of 128 bits to 128 will not be invertible, and will have outputs for which there are many inputs.

[6] There are some other generic attacks on stream ciphers that are potentially applicable to counter mode but are not relevant to current proposals, including Babbage's time-memory tradeoff attack [9] and the extension of that attack by Biryukov and Shamir [10].

Attack	AES-128 CM no counter entropy	AES-128 CM 64-bit counter entropy
TMTO	85	128
KC	$128 - \lg M$	128
Exhaustive Search	128	128

Table 1: AES-128 CM effective key sizes against various attacks, where M session keys are being attacked (and $M < 2^{64}$).

In these attacks, the adversary computes a large database prior to attacking any secret keys, then uses this database during the attack stage, potentially attacking many different secret keys. The work of precomputing the database is amortized over many different attacks. An important property of these methods, when applied against an additive cipher, is that they do not require any knowledge of the plaintext during the precomputation stage. In fact, these attacks can be used even when there is uncertainty in the plaintext during the attack stage, using techniques from error-correcting codes [8]. The usefulness of the TMTO is demonstrated by the fact that its use was crucial in the recent subversion of the A5/1 cipher [5].

Precomputation attacks are useful for attacking a system in which many keys will be used. Cryptographic systems (such as those based on SSL and IPsec) typically use many traffic-encryption keys. In many cases, a system should be considered subverted if even a small fraction of the traffic-encryption keys are found by an adversary. These cases provide fruitful ground for precomputation attacks.

The effectiveness of the KC attack depends on the number of secret keys which are attacked. An additive cipher with an n-bit key has an effective key size of $n - \lg M$ against the KC attack when it is used to attack M secret keys. The TMTO has an effective key size of $2n/3$. Thus the AES-128 CM cipher with a predictable counter has an effective key size of $128 - \lg M$ bits and 85 bits against the KC and TMTO attacks, respectively.

The 1996 ad-hoc report on minimal key lengths [13] recommended 75-bit keys. Adding 9 bits to accommodate for Moore's law in the intervening six years, the recommended current strength for ciphers is 88 bits. The AES-128 CM cipher with a predictable counter has effective key sizes against precomputation attacks that is lower than this value. Thus AES CM designs should consider protections against these attacks, in order to ensure that they provide adequate security, especially into the future.

The fact that the TMTO and KC attacks reduce the effective key size of AES CM below the size of the AES key may seem paradoxical in light of the security proof cited above. This apparent paradox is due to the fact that an adversary who can distinguish AES CM from a random source using the TMTO or KC attack can also distinguish AES *itself* from a random permutation.

4.1 Defense Against Precomputation Attacks

A cipher can protect itself against precomputation attacks in several ways. The CBC encryption mode conventionally uses a random IV to serve this purpose. The dependency of the ciphertext on the IV cause the attacker to assume a particular value of the IV during the precomputation stage; the unpredictability of the IV then reduces the probability that the precomputation will be useful. Ciphers which rely on a random IV are sometimes said to perform *randomized encryption*. If there are u bits of unpredictable input into the encryption in addition to the key, then the effective key sizes against the TMTO and KC attacks is increased by u.

Another defense for CM is to have the initial counter value be unpredictable. Adding u bits of unpredictable state to the counter adds u bits to the effective key size. An easy way to accomplish this is to have a field in the counter set to a random value. Since no more than 2^{64} distinct AES outputs can be used, 64 bits suffice to index the counter values. This leaves 64 bits which can be set to a random value. AES-128 CM with a 64-bit unpredictable counter provides 128 bits of strength against precomputation attacks, as long as 2^{64} or fewer keys are attacked. This level of protection is likely to be sufficient the near term.

An easy way to implement CM using an unpredictable component in the initial counter is to consider the field containing the unpredictable data as part of the CM key. In this way, the existing methods for the generation and dissemination of keys using automated key management methods (such as IKE and the TLS key establishment protocol) can be used straightforwardly. Given the fact that CM is recommended to be used in conjunction with an automated key management system (Section 2.2), this solution is a natural one.

Another way to protect against precomputation attacks is to use a predictable but uniformly distributed component in the initial counter. This method is analogous to the use of a public 'salt' value to protect system password files against dictionary attacks. By including some per-sender or per-session data in the initial counter, attacks which amortize their computation across multiple senders or multiple sessions can be thwarted.

Yet another way to protect CM against precomputation attacks is to use a larger key. However, AES implementations need not include support for the larger AES key sizes. For this reason, and for compactness of implementation, it is undesirable to require the implementation of larger key sizes. An implementation which uses AES-128 CBC would ideally be able to re-use its AES-128 implementation in a counter mode cipher with equivalent security. In addition, larger AES key sizes require more computation than does AES with a 128-bit key.

5 Conclusions

Counter mode has been well-analyzed by the theoretical cryptography community and security practitioners. Its systems security implications are different than those of CBC mode, but are not problematic. Implementors can leverage experience with other additive ciphers (such as RC4 and SEAL) when adding CM to cryptographic systems. We suggest that CM be used in conjunction

with a message authentication code, that it be used with automated key management, and that the counter value by maintained within the cryptographic module implementing CM.

Precomputation attacks are a realistic threat. Adding a 64-bit unpredictable value to the initial counter provides a reasonable level of protection against these attacks, consistent with recommended cipher strengths. This unpredictable value should be considered part of the CM key, for the purposes of key management.

6 Acknowledgements

Thanks to Mats Naslund for fruitful discussions and encouragement.

References

[1] Specification for the Advanced Encryption Standard (AES), *FIPS 197*, U.S. National Institute of Standards and Technology. November 26, 2001. `http://www.nist.gov/aes`.

[2] M.E. Hellman, A cryptanalytic time-memory trade-off, *IEEE Transactions on Information Theory*, July, 1980, pp. 401-406.

[3] M. Bellare, A. Desai, E. JokiPii, and P. Rogaway, A Concrete Security Treetment of Symmetric Encryption : Analysis of the DES Modes of Operation, *Proceedings of the 38th Symposium on Foundations of Computer Science*, IEEE, 1997. A revised version is available online at `http://www-cse.ucsd.edu/users/mihir`.

[4] E. Biham, How to Forge DES-encrypted messages in 2^{28} steps, *Technion Computer Science Department Technical Report CS0884*, 1996. Available online at `http://www.cs.technion.ac.il/~biham/publications.html`.

[5] A. Biryukov, A. Shamir, D. Wagner , Real Time Cryptanalysis of A5/1 on a PC, *Proceedings of the Fast Software Encryption Workshop 2000*, Springer-Verlag, Lecture Notes in Computer Science, 2000.

[6] W. Crowell, *Introduction to the VENONA Project*, `http://www.nsa.gov:8080/docs/venona/index.html`.

[7] H. Lipmaa, P. Rogaway, D. Wagner, Counter Mode Encryption, *Proposal for the NIST Modes of Operation Workshop*, `http://csrc.nist.gov/encryption/modes/proposedmodes/ctr/ctr-spec.pdf`

[8] D. A. McGrew and S. R. Fluhrer, Attacks on Additive Encryption of Redundant Plaintext and Implications on Internet Security, *The Proceedings of the Seventh Annual Workshop on Selected Areas in Cryptography (SAC 2000)*, Springer-Verlag, August, 2000. Available online at `http://www.mindspring.com/~dmcgrew/dam-srf-sac00.pdf`.

[9] S. Babbage, A Space/Time Trade-Off in Exhaustive Search Attacks on Stream Ciphers, *EUROCRYPT '96 Rump Session*, April 1996. Available online at `http://www.iacr.org/conferences/ec96/rump/`.

[10] A. Biryukov and A. Shamir, Cryptanalytic Time/Memory/Data Tradeoffs for Stream Ciphers, *The Proceedings of ASIACRYPT 2000*, Springer-Verlag, 2000, pp. 1-13. Available online at `http://www.wisdom.weizmann.ac.il/~albi/publications.html`.

[11] B. Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, Wiley, 1996. See also the FAQ at `http://www.rsasecurity.com`.

[12] P. Rogaway and D. Coppersmith, A software-optimized encryption algorithm, *Journal of Cryptology*, vol. 11, num. 4, pp. 273-287, 1998. Earlier version in *Proceedings of the Fast Software Encryption Workshop*, Lecture Notes in Computer Science, Vol. 809, R. Anderson, ed., Springer-Verlag, 1993.

[13] M. Blaze, W. Diffie, R. Rivest, B. Schneier, T. Shimomura, E. Thompson, and M. Weiner, *Minimal Key Lengths for Symmetric Ciphers to Provide Adequate Commercial Security*, January 1996. Online at `http://www.counterpane.com/keylength.html`.