Internet Draft
September 2001
Expiration Date: April 2002

J. Walker, Intel
R. Moskowitz, TruSecure

The AES128 CTR Mode of Operation and Its Use With IPsec
draft-moskowitz-aes128-ctr-00.txt

Status of this Memo

Table of Contents

1. Abstract

This document describes the use of the AES Cipher Algorithm with 128 bit key in Counter (CTR) Mode, with an implicit counter, as a

confidentiality mechanism within the context of the IPsec Encapsulating Security Payload [ESP].

CTR is a parallelizable block-cipher mode of operation. It uses the underlying block cipher as a stream cipher. Accordingly, great care must be exercised to utilize it appropriately within IPsec.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC-2119].

3. Introduction

Counter mode (CTR) is a block-cipher mode of operation. It uses a block cipher as a stream cipher. Although it has not yet been standardized, it is one of the oldest modes of operation. It is particularly easy to implement, and parallelizable. The size of the generated ciphertext increases only by the size of the counter value over that for the plaintext. Furthermore, it only uses the block cipher encryption primitive for both encryption and decryption. Counter mode also guarantees a high level of privacy when the underlying block cipher does and it is used correctly. Thus, counter mode has many properties that make it attractive for use with AES and high-speed networking.

Counter mode also has properties that render it harder to use than other modes of operation. Being a stream cipher, any reuse of the counter (from which the mode is named) with the same key is catastrophic, in that it immediately leaks information about the encrypted plaintext. Hence it is inappropriate to use this mode of operation with statically configured keys, unless the implementation takes extraordinary measures to prevent reuse of a counter value with the key. Also, it is trivial to use any valid ciphertext to forge other valid ciphertexts, so it is equally catastrophic to use the mode of operation without message authentication.

Counter mode is easy to describe abstractly. Assume the cipher block E uses a block size of B bits; for AES, B = 128. The encrypting party establishes a counter C when the session key is fixed:

```
        C := 0
```

To encrypt a message M, the sender partitions M into B-bit blocks M = M[1] M[2] à M[n-1] M[n]. Each block of M is then XORÆd with an encrypted counter value:

```
        C[0] := C
        for i := 1 to n-1 do
                C := C + 1
```



```
                C[i] := M[i] XOR E(C)
        C := C + 1
        C[n] := M[n] XOR Chop(E(C))
        output C[0] C[1] à C[n] as the ciphertext
```

Here, if block M[n] consists of k bits, the function Chop() truncates its argument to the k most significant bits. To decrypt a counter mode-encrypted ciphertext C[0] C[1] à C[n], this process is reversed:

```
        C := C[0]
        for i := 1 to n-1 do
                C := C + 1
                M[i] := C[i] XOR E(C)
        C := C + 1
        M[n] := C[n] XOR Chop(E(C))
        output M[1] à M[n] as the recovered plaintext
```

4. CTR rules in ESP

For CTR to be used in ESP, four factors MUST be standardize. These are the padding rules, the construction of the counter, the keying material, and data authentication.

4.1. Padding

CTR mode does not require padding of the cleartext. However, ESP does. ESP uses padding to 32-bit word-align the authentication data. The padding, Pad Length, and the Next Header MUST be concatenated with the cleartext before encrypting, as per ESP rules.

4.2. The Counter

Each ESP datagram must convey the counter used to encrypt the payload. When AES is used as the block cipher, the counter consists of 128 bits. This specification defines the CTR mode counter implicitly:

Counter ::= (0x0000000000000 || SPI || Seq# || 0x000)

where SPI identifies the security association, Seq# is the ESP sequence number, represented as a big-Endian integer value, and the 12 least significant bits of the counter begins with the value 0, again represented as a big-Endian integer value. The first 128-bit block of the datagram plaintext is encrypted by XORing the plaintext block with the value AES(Counter+1), the second by XORing the second block of plaintext with AES(Counter+2), etc. This construction permits each datagram to consist of up to 2^12 = 4096 128-bit blocks, or 65536 bytes of total encrypted data, including padding.

4.3. Keying Material for AES128-CTR

Provide the 128 most-significant bits as the encryption key size to the keying material extraction process. The remaining bits are applied to the ESP authentication method.

4.4. Data Authenticity for AES128-CTR

Since it is trivial to construct one valid ciphertext from any other valid ciphertext when counter mode is used, implementations MUST require the use of a non-NULL ESP authentication method with counter mode.

5. Security Considerations

When used properly, AES-CTR mode provides strong confidentiality guarantees. Bellare et. al. show in [MODES] that the privacy guarantees under counter mode are at least as strong as those for CBC mode when using the same block cipher for both.

However, it is very easy to misuse this construction. If a counter value is ever reused with a key, the confidentiality guarantees are voided. This is very easy to see: if the same counter value i is used to encrypt two plaintexts P[1] and P[2], then it is trivial to recover information about the plaintexts:

```
C[i] := E(i)
C[1] := P[1] XOR C[i]
C[2] := P[2] XOR C[i]
C[1] XOR C[2] = (P[1] XOR C[i]) XOR (P[2] XOR C[i])
              = P[1] XOR P[2]
```

Practically, this implies this mode of operation should not be used with statically configured keys. ESP implementations therefore SHOULD NOT support CTR mode with statically configured keys; if it does, the implementation MUST take other precautions to assure the implementation will never reuse a counter value with a key.

Similarly, data forgery is trivial with CTR mode. The argument is very similar to the privacy case: if a known plaintext P[1] is counter mode encrypted under i, then it can be replaced with plaintext P[2], since

```
C[i] := E(i)
C[1] := P[1] XOR C[i]
C[2] = (P[2] XOR P[1]) XOR C[1]
```

Accordingly, implementations MUST NOT allow the use of CTR mode without ESP authentication.

6. IANA Considerations

IANA has assigned ESP transform number XX to AES128-CTR.

7. ICANN Considerations

There are no ICANN considerations here.

8. References

[RFC-2119], Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, March 1997.

[ESP], Kent, S., and Atkinson, R., "IP Encapsulating Security Payload", RFC 2406, November 1998.

[MENEZES], Menezes, A., van Oorschot, P., and Vanstone, S., "Handbook of Applied Cryptography", CRC Press, 1997.

[MODES], Bellare, M, Desai, A., Jokipii, E., and Rogaway, P., ôA Concrete Security Treatment of Symmetric Encryption: Analysis of the DES Modes of Operationô, Proceedings 38th Annual Symposium on Foundations of Computer Science, 1997

9. Acknowledgments

TBS

APPENDIX A. Test Vectors for AES128 CTR Mode

The following test vectors were computed using the algorithm from Section 3 above. The counter value was constructed as in Section 4.2. These two conditions mean the counter value is incremented and then transformed into a Big-Endian value prior to encryption.

```
key        = 00000000000000000000000000000000
counter    = 1000
plaintext  = 00000000000000000000000000000000
ciphertext = 8eaedd7f7b46339c7589ae9d73498648

key        = c34c052cc0da8d73451afe5f03be297f
counter    = 2000
plaintext  = 8eaedd7f7b46339c7589ae9d73498648
ciphertext = be70f1bd31cf5671269ef79935d7e58b

key        = c98d5fb63b68c027e88317d8233c5b9d
counter    = 3000
plaintext  = be70f1bd31cf5671269ef79935d7e58b
```



```
ciphertext = db7a53c1fe52d21a25048df2e0ef5684

key        = 6a5964499d35cea7e1ac707b37b923ed
counter    = 4000
plaintext  = db7a53c1fe52d21a25048df2e0ef5684
ciphertext = 0293c3e5e8f70561eb875bfd1e8ccd9d

key        = 5f060d3716b345c253f6749abac10917
counter    = 5000
plaintext  = 0293c3e5e8f70561eb875bfd1e8ccd9d
             00000000000000000000000000000000
ciphertext = f86217238848f2428924a5e23a30d41e
             fda94a058c491959ca2bc0b546c612b1

key        = f3ce6e546edba6223b2489f48fc94c5d
```

```
   counter    = 6000
   plaintext  = f86217238848f2428924a5e23a30d41e
                fda94a058c491959ca2bc0b546c612b1
   ciphertext = e18800d55abfa320e7688f847c765fe6
                e61d9e4035caf76839ce1f3f5666d432

   key        = 9591f146c6ff55f71138f822aec4182d
   counter    = 7000
   plaintext  = e18800d55abfa320e7688f847c765fe6
                e61d9e4035caf76839ce1f3f5666d432
   ciphertext = 1bde1ed88f361135eb0af3aac4f04695
                f714185babc5f7f1b373d7a534a67a3d

   key        = 0eb6c75b7a7103eff9d178145b69589d
   counter    = 8000
   plaintext  = 1bde1ed88f361135eb0af3aac4f04695
                f714185babc5f7f1b373d7a534a67a3d
   ciphertext = 40a828ac02241e92677754784010ad83
                85a6c61123e66b50bb1ed9c1894e3b6d

   key        = 2f6f7a25c070604d6a23ad786a05e5ab
   counter    = 9000
   plaintext  = 40a828ac02241e92677754784010ad83
                85a6c61123e66b50bb1ed9c1894e3b6d
                00000000000000000000000000000000
                00000000000000000000000000000000
   ciphertext = a306bd9d74436128027eac3f80c60a55
                e32ed7e95ec3bf14542c594c24191fc5
                1bb2bba73511d1195d65109ce2c08aaf
                d1362f2c8ee6a438ee996f7ec9aa6ef6

   key        = 11ef4d8c58fe48b2ebd10a6d20c87415
   counter    = a000
   plaintext  = a306bd9d74436128027eac3f80c60a55
                e32ed7e95ec3bf14542c594c24191fc5
                1bb2bba73511d1195d65109ce2c08aaf
                d1362f2c8ee6a438ee996f7ec9aa6ef6
   ciphertext = da6816cc4bdbc9e70f182e2ef003764a
```



```
                bc6cc452169986c1aab6bab49a7cfb8c
                2dea11fc8031110fffc699e3286f14f8
                62945e21dc2d9a3bf566f20dea27fd0a

   key        = 10a1acc3425688664fab789cb73ca8a5
   counter    = b000
   plaintext  = da6816cc4bdbc9e70f182e2ef003764a
```

```
                bc6cc452169986c1aab6bab49a7cfb8c
                2dea11fc8031110fffc699e3286f14f8
                62945e21dc2d9a3bf566f20dea27fd0a
   ciphertext = 84f937044a5aa566a5d1792e0591609a
                36cd91ddc2a78d9c7e58d030cd81bf13
                d38385e80804f15ca51cf219918c1c44
                1d1491689e9264edaf40e3a56c052e39

   key        = 1bf58c409996b2f0e5abb93919646154
   counter    = c000
   plaintext  = 84f937044a5aa566a5d1792e0591609a
                36cd91ddc2a78d9c7e58d030cd81bf13
                d38385e80804f15ca51cf219918c1c44
                1d1491689e9264edaf40e3a56c052e39
   ciphertext = 6a8888ff1501f99ed8a23426ffebb918
                b12b4a52e801a79a3984208577ea0aba
                6909a93e7d9186af2b7f2d39283e8ad1
                479a1979b53a3bbe0e293e3535c433c8

   key        = 68b539de87ad0bd1f94adecbc7d9d1c4
   counter    = d000
   plaintext  = 6a8888ff1501f99ed8a23426ffebb918
                b12b4a52e801a79a3984208577ea0aba
                6909a93e7d9186af2b7f2d39283e8ad1
                479a1979b53a3bbe0e293e3535c433c8
                00000000000000000000000000000000
                00000000000000000000000000000000
                00000000000000000000000000000000
                00000000000000000000000000000000
   ciphertext = 87849e665e824d5b63ebcdd2f3e5f8d9
                7fca7d994f32bd625c8045b6adad1bbf
                d62794d947ac071f9d1891a2ef7b041c
                43b2840a5d86c3294d996b2e08fe8b66
                5dce18e88feca383914967de9770de83
                c25102a433467d712a0c75cb1a537c0a
                8d8bf7452eb508cb108042f46f3c04e6
                173d915cbf1fa272273e56f09ac34353

   key        = 0f5566ab94f6a3e53287113baf8dca7a
   counter    = e000
   plaintext  = 87849e665e824d5b63ebcdd2f3e5f8d9
                7fca7d994f32bd625c8045b6adad1bbf
                d62794d947ac071f9d1891a2ef7b041c
                43b2840a5d86c3294d996b2e08fe8b66
                5dce18e88feca383914967de9770de83
                c25102a433467d712a0c75cb1a537c0a
```

```
              8d8bf7452eb508cb108042f46f3c04e6
              173d915cbf1fa272273e56f09ac34353
ciphertext = 54114ffde7ef68e65861557d8d29949e
              9e5d369d566217dd7e86eee61504099f
              05242d6d1a8c6f1bb747361e145f64ac
              a8cf780493ab5e26156c11e5cc9c9853
              29688343a3b73a68acaeb50dc144c938
              e9d6e8cd8f20c5bd18c001900d49a0bd
              409cd49bc4f08873231eb345328ed32c
              514be68d572ff3f53fbe92f1c8de23d1

key        = 8480339a9135f39e3880ec88fe3f9135
counter    = f000
plaintext  = 54114ffde7ef68e65861557d8d29949e
              9e5d369d566217dd7e86eee61504099f
              05242d6d1a8c6f1bb747361e145f64ac
              a8cf780493ab5e26156c11e5cc9c9853
              29688343a3b73a68acaeb50dc144c938
              e9d6e8cd8f20c5bd18c001900d49a0bd
              409cd49bc4f08873231eb345328ed32c
              514be68d572ff3f53fbe92f1c8de23d1
ciphertext = 8e98f31a8c36a2189c45d2cc820c91e5
              2289d925838a0caeb598d5ee82ea9e76
              cf5e414bdba0f8a33f0df20ce37aba97
              a4a787412871120a36ec7ddb44823e5b
              3cf23e687d5995c3ac2ddd74f68d489c
              f7dd8c73e4506d6fd7f22d7b84900589
              0494c0bfe59a879dd5e75c7a7ae32e19
              e16528837b102df9f3004d48a4eed298

key        = e01f358504f55405eb8682771b8ded9e
counter    = 10000
plaintext  = 8e98f31a8c36a2189c45d2cc820c91e5
              2289d925838a0caeb598d5ee82ea9e76
              cf5e414bdba0f8a33f0df20ce37aba97
              a4a787412871120a36ec7ddb44823e5b
              3cf23e687d5995c3ac2ddd74f68d489c
              f7dd8c73e4506d6fd7f22d7b84900589
              0494c0bfe59a879dd5e75c7a7ae32e19
              e16528837b102df9f3004d48a4eed298
ciphertext = 1d3389c2986ee4f53346ddc8a1539562
              e58e6f8ba0a78cd6de78579caac8b632
              4f5611959f6071a4700245e4500516db
              11e8f73c8588f109557e9a60f1bc8757
              946184932bce426376844ec72a124b3e
              0f4917f01b13b34a16757c4f8348d90b
              00e53f7777ce7889f2d90716add0eaed
              a950aefa0ac515c8955050366194fee3
```

11. Author's Address

Robert Moskowitz
TruSecure Corporation
1200 Walnut Bottom Rd.
Carlisle, PA 17013
Email: rgm@trusecure.com

Jesse Walker
Intel Corporation
2111 N.E. 25th Avenue
Hillsboro, Oregon 97229
Email: jesse.walker@intel.com

16. Copyright Statement