

Critical Weaknesses of iaPCBC

Niels Ferguson* Doug Whiting† John Kelsey‡ David Wagner§

November 24, 1999

Abstract

The iaPCBC scheme of Gligor and Donescu does not achieve its stated goal of secure authentication. A trivial attack of constant effort produces corrupted packets that are *guaranteed* not to be detected as such, for packets of sufficient (but practical) length. The existence of such simple attacks implies that the entire approach, including the so-called security "proofs", is fundamentally flawed.

Keywords: cryptography, cryptanalysis, chaining modes, authentication.

1 Introduction

The Integrity-Aware Plaintext Ciphertext Block Chaining (iaPCBC) scheme [GD99] presented by Gligor and Donescu attempts to improve speed by using an encryption chaining mode to provide authentication, without the need to compute a separate MAC. The goal is certainly a laudable one, since the overhead of computing a cryptographic MAC function may be significant in many instances, particularly when the cryptographic computations are performed in software.

The iaPCBC scheme has been also proposed as an experimental RFC for use in IPSEC [BG99], which would allow the ESP transform to be used without requiring HMAC-SHA or HMAC-MD5.

The general iaPCBC approach is as follows. For each packet, a packet-specific value R_0 is generated. The size of R_0 is the same as the block size of the underlying block cipher (e.g., 3DES). In stateless mode, R_0 is generated pseudorandomly. In stateful mode, the value of R_0 is initially chosen at random and then incremented by two for each packet. Accompanying

chaining variables C_0 and P_0 are then generated as

$$C_0 = E_{K_I}(R_0)$$

and

$$P_0 = E_{K_I}(R_0 + 1),$$

where K_I is a secret key. The value C_0 is sent in the clear along with the packet. For each plaintext block P_i , for $i = 1..M$, the associated ciphertext C_i is computed using the following steps:

$$R_i = R_{i-1} + P_{i-1} + C_{i-1}$$

$$C_i = E_{K_D}(P_i \oplus R_i),$$

where K_D is a separate secret key. A known value (e.g., the SPI and sequence number of an IPSEC packet) is appended to the plaintext and encrypted using this method. Upon receiving an iaPCBC packet, the ciphertext is decrypted and the reconstructed known-value plaintext block is checked against its expected value. If the block does not match, an authentication failure occurs. Note that using iaPCBC incurs the cost of three additional encryptions per packet, as well as one addition per block. In many cases, this overhead is considerably less than the cost of computing a MAC.

*Counterpane Internet Security, Inc., 3031 Tisch Way, 100 Plaza East, San Jose, CA 95128, USA; niels@counterpane.com

†Hi/fn, Inc., 5973 Avenida Encinas Suite 110, Carlsbad, CA 92008, USA; dwhiting@hifn.com

‡Counterpane Internet Security, Inc. kelsey@counterpane.com

§University of California Berkeley, Soda Hall, Berkeley, CA 94720, USA; daw@cs.berkeley.edu

2 Attacks

Multiple serious attacks against iaPCBC have been found with fairly minimal effort. They are summarized in the following paragraphs.

2.1 Key Search Resistance

The existence of the ‘hidden’ chaining variable R_i is claimed in [GD99] to improve resistance against certain exhaustive key-search attacks. In the initial version of [BG99], it was claimed that this resistance could allow a weaker cipher such as DES to be used instead of 3DES to achieve security. The following two attacks totally disprove these claims.

In particular, in stateful mode, given the cleartext value of C_0 from two consecutive packets, a key search over K_I can be performed to check whether the associated values for $R_0 = D_{K_I}(C_0)$ are related by an addition of two. The computational load of this key search is only two decryptions and one comparison per key value. It is possible that a false match could occur during the search, in which case a third C_0 value could be used to guarantee that the key is correct. Thus, clearly the key search resistance for K_I is not meaningfully improved.

Also, if only two consecutive plaintexts are known, a simple key search for K_D can be used, both in stateful and stateless mode. Suppose the P_i and P_{i+1} are known in a given packet. Since C_i and C_{i+1} are also known, the chaining variables can be eliminated using the following facts:

$$C_i = E_{K_D}(P_i \oplus R_i)$$

$$R_{i+1} = R_i + P_i + C_i$$

$$C_{i+1} = E_{K_D}(P_{i+1} \oplus R_{i+1})$$

Thus,

$$R_{i+1} = (D_{K_D}(C_i) \oplus P_i) + P_i + C_i$$

and

$$C_{i+1} = E_{K_D}(P_{i+1} \oplus ((D_{K_D}(C_i) \oplus P_i) + P_i + C_i))$$

Note that this final equation involves only known quantities, except for K_D . Thus a simple key search can be performed over K_D , with only one encryption and decryption required per key value, plus a few simple additions and xors. The keysearch resistance for K_D is thus not meaningfully improved.

Clearly, in no way does use of iaPCBC improve the security of the underlying block cipher.

After the results in this section were brought to the attention of the authors of [BG99], the keysearch resistance claims were dropped, but the ease with which these flaws were found suggests extreme caution in evaluating other claims of iaPCBC.

2.2 Truncation Attack

A second problem is that appending a known value block to the end of the plaintext provides well-known opportunities to an attacker, in the case that the value is predictable and the underlying encryption mode allows truncation. (These two properties are indeed present in the experimental RFC [BG99].)

In particular, an attacker can mount a chosen-message attack. The attacker should request the encryption of the message

$$M' = M || KV || \text{Anything},$$

where $||$ denotes concatenation, M is some message the attacker would like to have accepted by the receiver, and KV is the known-value block. The length of M should be a multiple of the block size. The sender will append padding and a known check-value to M' , encrypt it under iaPCBC, and send the resulting ciphertext to the receiver. At this point the attacker should intercept the transmitted ciphertext and truncate it at the block boundary just before the ‘anything’ field. The resulting truncated ciphertext will now decrypt to a padded version of M , followed by the correct known-value block, so the receiver will accept M as a valid and correct message. This is an integrity failure, since the sender only authorized the transmission of M' , not of M .

Although this problem could be easily avoided with minor modifications to the proposed iaPCBC mode (e.g., encryption of the known value under a separate key; prepending the plaintext length; or using a secret internal chaining value from the iaPCBC mode, as was suggested in the paper [GD99]), it again strongly suggests that the scheme has not been well thought through.

2.3 Authentication Failures

This section presents a devastating attack on iaPCBC’s authentication mechanism, which is the chief goal of the scheme.

The chaining relations for decryption have the following form:

$$R_i = R_{i-1} + P_{i-1} + C_{i-1}$$

$$P_i = D_{K_D}(C_i) \oplus R_i,$$

Note that the recurrence relation is almost linear in a single error injected into R_i . In particular, for a given difference in R_i , the same difference occurs in P_i , but that difference in R_{i+1} then almost cancels out, except for the difference between addition and xor.

$$R_{i+1} = R_i + P_i + C_i$$

If all operations were xor, the difference would cancel out *immediately*. Instead, it takes only a few blocks! To see this, suppose that an arbitrary difference is injected into R_i . This can be done by simply making *any* change to C_{i-1} . Let b be the least significant bit of R_i which has a difference (e.g., $b = 1$ implies the LSB). Least significant in this case is measured with respect to the addition operation.

Now, for a difference of b in R_i , the least significant difference in P_i is also b . When the value R_{i+1} is computed, note that the two differences in position b cancel out, possibly moving a difference up to the $2b$ position via the carry bit from the addition operation. So, for each subsequent ciphertext, the difference is *guaranteed* to move up one bit position. If the block size is N bits, then the difference is *guaranteed* to disappear after N blocks!

This property of iaPCBC has been verified empirically with various block ciphers. In practice, for a 64-bit cipher (DES or 3DES), a difference dies out in less than eleven blocks, on average. Thus, for example, any error injected into the C_0 will corrupt several plaintexts but will be *totally undetected* for all packets larger than 512 bytes, and for most packets of 100 bytes or more.

Several minor modifications to the chaining mechanism were tried, including various rotations and other feedback terms. In each case, the modified chaining failed to detect most differences, although not always as spectacularly as in the unmodified iaPCBC.

This type of behavior is totally unacceptable for any authentication scheme.

3 Performance Issues

Great care needs to be taken in evaluating the performance implications of any new authentication scheme. In general, it is believed that something like iaPCBC might improve the performance of IPSEC.

However, it is critical that the performance comparison be done using secure ciphers (i.e., 3DES, not DES). For example, using optimized code on a Pentium-class CPU, the cost of three extra 3DES blocks is on the order of 3000 clocks, while the overhead of HMAC-MD5 for a small packet (less than 128 bytes) is less than 3000 clocks. Thus, it is not clear that the performance gains are significant. For large blocks, the block cipher overhead of 3DES is significantly larger (8-10x) than that of HMAC. While there will be a speed gain, it may not be very large.

In hardware, with some additional logic, the HMAC computation can be easily pipelined with the block cipher, so the overhead of using HMAC is negligible in high-performance hardware systems. Chips already exist that perform HMAC and 3DES at high speed, and HMAC is rarely the performance limitation.

The parallel versions of iaPCBC do allow linear performance gains on large packets when sufficient hardware is available. However, such gains can also be easily achieved using a simple interleaved CBC mode scheme and interleaved MACs. Thus, while such an idea may be useful, it is hardly intrinsic to iaPCBC.

4 Conclusion

These attacks on iaPCBC call into grave doubt the viability of any such scheme as an authentication mechanism. These results were obtained in a matter of only a few hours of work after seeing the initial drafts. The security proofs given in [GD99] are clearly either flawed or meaningless.

References

- [GD99] Gligor and Donescu. Integrity-Aware PCBC Encryption Schemes. November, 1999. See <http://www.research.att.com/~smb/papers/iapcbc.ps>
- [BG99] Bellare and Gligor. An iaPCBC Transform for IPsec. November, 1999. See <http://www.research.att.com/~smb/papers/draft-bellare-gligor-iaPCBC-00.txt>