2004

Observations on the Message Integrity Code in IEEE802.11 Wireless LANs

Jianyong Huang
University of Wollongong, jyh33@uow.edu.au

Willy Susilo
University of Wollongong, wsusilo@uow.edu.au

Jennifer Seberry
University of Wollongong, jennie@uow.edu.au

Publication Details
Observations on the Message Integrity Code in IEEE802.11 Wireless LANs

Abstract
This paper surveys the security of the message integrity code used in the IEEE802.11 Wireless LANs. To address the security flaws of Wired Equivalent Privacy (WEP), the IEEE802.11i draft defines two data confidentiality and integrity protocols, Temporal Key Integrity Protocol (TKIP) and Counter-Mode-CBC-MAC Protocol (CCMP). TKIP is based on RC4, and CCMP is based on the AES cipher. TKIP includes a keyed hash functions, called Michael, as the message integrity code. The aim of this paper is to summarise the recent research results of Michael and analyse the practicability of two attacks against Michael.

Disciplines
Physical Sciences and Mathematics

Publication Details

This conference paper is available at Research Online: http://ro.uow.edu.au/infopapers/511
Observations on the Message Integrity Code in IEEE802.11 Wireless LANs

Jianyong Huang, Willy Susilo and Jennifer Seberry
School of Information Technology and Computer Science
University of Wollongong
Australia
Email: {jyh33, wsusilo, jennie}@uow.edu.au

Abstract

This paper surveys the security of the message integrity code used in the IEEE802.11 Wireless LANs. To address the security flaws of Wired Equivalent Privacy (WEP), the IEEE802.11i draft defines two data confidentiality and integrity protocols, Temporal Key Integrity Protocol (TKIP) and Counter-Mode-CBC-MAC Protocol (CCMP). TKIP is based on RC4, and CCMP is based on the AES cipher. TKIP includes a keyed hash functions, called Michael, as the message integrity code. The aim of this paper is to summarise the recent research results of Michael and analyse the practicability of two attacks against Michael.

1 Introduction

The IEEE802.11 standard [3] defines a WEP protocol to protect authorised users of a wireless LAN from eavesdropping and other attacks. WEP uses RC4 [11] as the encryption and decryption algorithm, and employs the CRC-32 (Cyclic Redundancy Check) as the Message Integrity Code (MIC). Recent research showed that WEP fails to provide confidentiality, access control and data integrity [5, 7, 12]. The authentication protocol in WEP was attacked by [4]. There are open source tools available on the Internet to break the WEP system, for example, Airsnort [8] and WEPCrack [13].

The IEEE802.11i draft [1] specifies two protocols, TKIP and CCMP, to address the weaknesses of WEP. The WEP protocol is implemented in hardware in most existing IEEE 802.11 devices, and these devices are still used by enterprises and home users at present. TKIP defines a set of algorithms, which wraps WEP, to allow the current WEP implementation to remain unchanged while addressing the security flaws of WEP. Two keyed hash functions, Michael and Temporal Key Hash (TKH), are employed by TKIP to enhance the WEP encryption (illustrated by Figure 1). Designed to prolong the usage of current IEEE802.11 devices, TKIP is subject to the constraints of the legacy hardware implementation. TKIP is considered only as a short-term solution. The long-term solution is CCMP, and the encryption algorithm in CCMP is the Advanced Encryption Standard (AES) [2]. The implementation of CCMP will require new hardware. In this paper, we only focus on the security issues of the message integrity code in TKIP, and refer the reader to [9] for TKH and to [1] for CCMP to find out more details.

Our Contributions: We summarise the recent research results of Michael, and provide remarks and comments on the latest attacks.

Notations

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TA</td>
<td>Transmitter address</td>
</tr>
<tr>
<td>TTAK</td>
<td>TKIP mixed transmitter address and key</td>
</tr>
<tr>
<td>TK</td>
<td>Temporal key</td>
</tr>
<tr>
<td>IV</td>
<td>Initialisation vector</td>
</tr>
<tr>
<td>DA</td>
<td>Destination address</td>
</tr>
<tr>
<td>SA</td>
<td>Source address</td>
</tr>
<tr>
<td>MAC</td>
<td>Medium access control</td>
</tr>
<tr>
<td>MSDU</td>
<td>MAC service data unit</td>
</tr>
<tr>
<td>MPDU</td>
<td>MAC protocol data unit</td>
</tr>
<tr>
<td>&lt;&lt;&lt;</td>
<td>Left rotation</td>
</tr>
<tr>
<td>&gt;&gt;&gt;&gt;</td>
<td>Right rotation</td>
</tr>
<tr>
<td>⊕</td>
<td>Exclusive-or</td>
</tr>
<tr>
<td>⊕</td>
<td>addition modulo $2^{32}$</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Organisation: The rest of this paper is organised as follows. Section 2 describes the Michael message integrity code. Section 3 documents recent research results of Michael, and contains four subsections: Section 3.1 shows that Michael is not one-way, and Section 3.2 provides a related-message attack. Section 3.3 reveals that Michael is not collision-free, and Section 3.4 demonstrates a packet forgery attack against Michael. Section 4 provides com-
ments on the recent research of Michael. Section 5 con-
cludes this paper.

![TKIP Encryption Diagram](image)

2 The Michael Message Integrity Code

WEP uses CRC-32 as its message integrity code, and
the CRC-32 mechanism fails to provide data integrity [5].
TKIP employs Michael [6] as its message integrity code.

The inputs to Michael include a 64-bit Michael key
and an arbitrarily long message, and the output is a 64-bit
Michael value. The message is padded at the end with a sin-
gle byte with the hexadecimal value 0x5a and then followed
by between 4 and 7 zero bytes. The padding method is to
make the total length of the message plus the padding equal
to a multiple of 4. The last block of the padded message is 0
while the second last block of the padded message is not 0.
The details of Michael are described in Algorithm 2.1 and
2.2.

Algorithm 2.1: \textsc{Michael}((k_0, k_1), (m_0, ..., m_{n-1}))

\textbf{Input}: Key\((k_0, k_1)\)
\textbf{Input}: Padded message\((m_0, ..., m_{n-1})\)
\textbf{Output}: MIC value \((L, R)\)

\(L, R \leftarrow (k_0, k_1)\)
\textbf{for} \(i \leftarrow 0 \text{ to } n - 1\)
\textbf{do} \(L \leftarrow L \oplus m_i\)
\textbf{return} \((L, R)\)

Algorithm 2.2: \textsc{B}(L, R)

\textbf{Input}: \((L, R)\)
\textbf{Output}: \((L, R)\)

\(R \leftarrow R \oplus (L \lll 17)\)
\(L \leftarrow (L + R) \mod 2^{32}\)
\(R \leftarrow R \oplus \text{XSWAP}(L)\)
\(L \leftarrow (L + R) \mod 2^{32}\)
\(R \leftarrow R \oplus (L \ggg 2)\)
\(L \leftarrow (L + R) \mod 2^{32}\)
\textbf{return} \((L, R)\)

The Michael value is computed iteratively by begin-
ning with the Michael key value and applying the block
function \textsc{B} (given in Algorithm 2.2) for every message
block. The block function is an unkeyed 4-round Feistel-
type construction. Michael uses several operations, in-
cluding exclusive-or, left rotation, right rotation, swapping
(\textit{XSWAP}), addition modulo \(2^{32}\). Function \textit{XSWAP}
swaps the position of the two least significant bytes and
the position of the two most significant bytes in a word,
\textit{i.e.,} \(\text{XSWAP}(ABCD) = BADC\) where \(A, B, C, D\) are
bytes.

3 Weaknesses of Michael

This section summarises recent results from two differ-
ent research groups.

3.1 Michael is Not One-Way

Wool [14] disclosed one serious weakness of Michael: it is \textit{inversible}. Given a known message \(M\) and its corre-
sponding Michael value MIC = \textsc{Michael}(\(K, M\)), the secret
Michael key \(K\) can be recovered, and the details are shown
in Algorithm 3.1 and 3.2.

Algorithm 3.1: \textsc{InvMichael}((v_0, v_1), (m_0, ..., m_{n-1}))

\textbf{Input}: Michael value \((v_0, v_1)\)
\textbf{Input}: Padded message \((m_0, ..., m_{n-1})\)
\textbf{Output}: Key \((k_0, k_1)\)

\textbf{for} \(i \leftarrow n - 1 \text{ downto } 0\)
\textbf{do} \(L \leftarrow L \oplus m_i\)
\textbf{return} \((L, R)\)


Algorithm 3.2: $B^{-1}(L, R)$

Input: $(L, R)$
Output: $(L, R)$
$L \leftarrow (L - R) \mod 2^{32}$
$R \leftarrow R \oplus (L >> 2)$
$L \leftarrow (L - R) \mod 2^{32}$
$R \leftarrow R \oplus (L << 3)$
$L \leftarrow (L - R) \mod 2^{32}$
$R \leftarrow R \oplus \text{XSWAP}(L)$
$L \leftarrow (L - R) \mod 2^{32}$
$R \leftarrow R \oplus (L << 17)$
return $(L, R)$

3.2 A Related-Message Attack

Wool proposed a related-message attack on Michael [14].

Proposition 3.1 Suppose an attacker is able to intercept two TKIP message frames (ciphertexts), $M_1$ and $M_2$, and the following three conditions hold:

1. $M_1$ and $M_2$ are encrypted by the same encryption key and the same IV.
2. $\text{length}(M_1) \geq \text{length}(M_2) + 8$
3. The plaintext $P_1$ of the longer message $M_1$ is known to the attacker.

Then the attacker can recover the Michael key $K$ of the shorter message $M_2$.

Wool’s attack is illustrated in Figure 2. The attacker can compute $K$ by following three steps listed below.

1. $\text{RC4KEY}[i] = M_1[i] \oplus P_1[i], i = 1, ..., \text{length}(M_1) - 8$
2. $P_2 || MIC_2 = \text{RC4KEY}[i] \oplus M_2[i], i = 1, ..., \text{length}(M_2)$
3. $K = \text{InvMichael}(MIC_2, P_2)$

Figure 2. Wool’s Attack

3.3 Michael is Not Collision-Free

Huang, Seberry, Susilo and Bunder (HSSB) [10] proved that the collision status of Michael only depends on the output of its third last round and the second last block message in Theorem 3.1.

Theorem 3.1 Given two pairs of keys and messages, $(K_{ey_1}, M_1)$ and $(K_{ey_2}, M_2)$, Michael($K_{ey_1}$, $M_1$) = Michael($K_{ey_2}$, $M_2$) if and only if the following two conditions hold:

1. $R^4_y = R^4_y - 3$
2. $L^5_x = L^5_y = m_{x-2} \oplus m_{y-2}$

where $M_1$ has $x$ 32-bit blocks, $M_2$ has $y$ 32-bit blocks, $M_1$ is distinct from $M_2$ and both $x$ and $y$ are $\geq 3$.

(Note: $R^4_x$ and $L^5_x$ are the right and left half of the third last round output of Michael($K_{ey_1}$, $M_1$) respectively. $R^4_y$ and $L^5_y$ are the right and left half of the third last round output of Michael($K_{ey_2}$, $M_2$) respectively. $m_{x-2}$ is the second last block of $M_1$ and $m_{y-2}$ is the second last block of $M_2$.)

Furthermore, HSSB [10] showed that Michael is not collision-free in Theorem 3.2 and Theorem 3.3.

Theorem 3.2 Given an arbitrarily length message $M$ and a specific key $K$, a 96-bit block message $M'$ and a key $K'$ can be computed such that Michael($K$, $M$) = Michael($K'$, $M'$), where $M$ has $n$ 32-bit blocks, $n$ is any integer $\geq 3$ and $M'$ is distinct from $M$.

Theorem 3.3 Michael is not collision-free (deduced from Theorem 3.2).

We note that Theorem 3.1 is a necessary and sufficient condition of finding collisions of Michael.

3.4 A Packet Forgery Attack Against Michael

HSSB [10] provided a simple method to find fixed points of Michael, and demonstrated that it is statistically certain that there exist about $2^{32*0.9}$ fixed points of Michael. A fixed point of Michael is a triple $(L_i, R_i, m_i)$ such that Michael($L_i$, $R_i$, $m_i$) = Michael($L_{i+1}$, $R_{i+1}$) = Michael($L_i$, $R_i$), where $L_i$, $R_i$ and $m_i$ are three input parameters to the block function, and $L_{i+1}$ and $R_{i+1}$ are two output parameters of the block function. The feature of fixed points is illustrated in Figure 3. For example, (4987c6d0, 1), 7161872 (hexadecimal numbers) is a fixed point of Michael as Michael((4987c6d0, 1), 7161872) = (4987c6d0, 1).

Based on the property of fixed points, a packet forgery attack was proposed against Michael in Theorem 3.4 [10].
Theorem 3.4 Given a message $M_1$ and an arbitrary key $K$, an attacker can always construct a message $M_2$ such that $\text{Michael}(K, M_1) = \text{Michael}(K, M_2)$ if the following condition holds:

1. The output of the block function of $\text{Michael}(K, M_1)$ in any round is equal to any of the fixed points

where $M_2$ is distinct from $M_1$.

Figure 4 describes the packet forgery attack. After padding, the original message $M_1$ has $n$ blocks, written as $M_1 = (m_0, m_1, ..., m_{n-1})$. Suppose the three inputs to the $(i+1)$-th round, namely $(L_i, R_i, m_i)$ in Figure 4, is a fixed point. Then a multiple of 4 blocks of $m_i$ can be inserted to $(i+2)$-th round without changing the Michael value. In other words, $M_2$ is constructed as $(m_0, m_1, ..., m_i, <m_i, m_{i+1}, ..., m_{n-1}>, m_{i+1}, ..., m_{n-1})$, where the number of inserted blocks of $m_i$ is a multiple of 4. The reason why the inserted blocks of $m_i$ is a multiple of 4 is due to the padding method.

4 Remarks on Recent Attacks on Michael

Michael is a new cryptographic design. Actually, Wool’s related-message attack [14] (described in Section 3.2) is known by the designer of Michael, and this attack is mentioned implicitly in the last paragraph of Page 14 in the design document [6], written as “A known-plaintext attack will reveal the key stream for that IV, and if the second packet encrypted with the same IV is shorter than the first one, the MIC value is revealed, which can then be used to derive the authentication key”. The related-message attack is based on Condition 1, 2 and 3 in Proposition 3.1. The practicability of the fulfilment of condition 1 is related to the Temporal Key Hash [9]. The aim of Temporal Key Hash is to provide per-packet WEP key, therefore it is not practical for the attacker to achieve Condition 1. So, the related-message attack is still in theory.

The packet forgery attack is also not practical as the attacker needs to fulfil Condition 1 in Theorem 3.4 and the TKIP frames are encrypted by RC4. According to [10], there exist about $2^{32} 	imes 0.9$ fixed points for Michael. The output of the block function in any round is 64-bit. Suppose a message $M$ has $n$ 32-bit blocks, then the probability of the fulfilment of Condition 1 in Theorem 3.4 is $\frac{2^{32} 	imes 0.9}{2^{64}}$.

5 Conclusions

Michael is the message integrity code used in TKIP in the latest IEEE802.11i draft. Recent research revealed some weaknesses in this keyed hash function. Michael is neither one-way nor collision-free. Although the related-message attack and the packet forgery attack against Michael are still not practical, we still would like to point out that designing a keyed hash function is not an easy task.

References


