An Integrity Check Value Algorithm for Stream Ciphers

Richard Taylor

Telematic and System Security, Telecom Australia Research Laboratories,
P.O. Box 249, Clayton, Victoria 3168, AUSTRALIA, r.taylor@trl.oz.au

Abstract. A method of calculating an integrity check value (icv) with the use of a stream cipher is presented. The strength of the message integrity this provides is analysed and proven to be dependent on the unpredictability of the stream cipher used. A way of efficiently providing both integrity and encryption with the use of a single stream cipher is also explained. Note that the method of providing message integrity, used with or without encryption, is not subject to a number of attacks that succeed against many conventional integrity schemes. Specifically any legitimate message-icv pair that is copied or removed and subsequently replayed will have an appropriately small small chance of deceiving the receiver. Furthermore, any message-icv pair generated by an attacker and injected into the communication channel will have an appropriately small chance of escaping detection unless the attacker has actually broken the stream cipher. This is the case even if the attacker has any amount of chosen messages and corresponding icvs or performs any number of calculations.

1 Introduction

An integrity check value (icv) refers to a function of a secret key and variable length input messages. For a given key the function maps variable length input messages into a fixed length output. The secret key is shared between the message sender and receiver. The icv of a message is calculated by the sender and appended to the message before being sent. The receiver calculates the icv for the received message and compares the calculated icv with the received one. The message is deemed to be intact if the received and calculated icvs match. Assume that a hostile third party (or attacker) can tap into the communication channel and so can accumulate messages and their corresponding icvs. The integrity protection afforded against such an attacker is a measure of their difficulty in generating a new message and legitimate icv. Note that the term integrity check value as used here is also referred to in the literature as keyed hash function, cryptographic checksum or message authentication code.

The majority of integrity check value algorithms that have appeared in the literature are based on the use of block ciphers (see for example [10], [8] or the standards document [4]). Other work on the related notion of hash functions without secret keys has been motivated by the desirability of digital signatures for public-key cryptosystems (see [3], [12], [5]). Unfortunately there appears to

D.R. Stinson (Ed.): Advances in Cryptology - CRYPTO '93, LNCS 773, pp. 40-48, 1994.

be a lack of methods to efficiently extend the functionality of stream ciphers used for encryption so that both encryption and integrity are provided. Motivated by the need for such a method we propose an integrity check value algorithm based on the use of stream ciphers that is both efficient and secure. This algorithm is particularly suitable as part of an integrated method of providing both encryption and integrity with the use of a single stream cipher. Note that an alternative method of using a stream cipher for calculating an icv has recently been proposed (see [9]). This method can be successfully attacked with high probability however by altering a message and icv in transit (for example if the last bit of a message is altered then one of only two bits in the corresponding icv need be altered to create the matching icv).

Note that the integrity check value algorithm presented is related to certain unconditionally secure authentication schemes (see [1], [2] and [14]). However the use of the polynomial function (2) is new as far as the author is aware, as is the proposed use of the integrity method together with encryption from a single stream cipher.

2 Integrity

Consider any stream cipher with output stream $Z = (z_0, z_1, z_2, ...)$ where each output z_i consists of w bits. Thus each z_i may be considered as an integer from 0 to $2^w - 1$. We show how the output of the stream cipher may be used to provide message integrity by the calculation of an icv that is appended to a message and sent with it. In the following for positive integers t and u we shall write $t[u]$ to represent the unique integer satisfying

$$t[u] \equiv t(mod u) \text{ and } 0 \leq t[u] \leq u - 1.$$

To calculate the icv select a message block length b and prime number $p > 2^w$. It is suggested that p be chosen to be close to a power of 2 for efficient calculation of products modulo p (see [6] for example). Some examples are $p = 2^{31} - 1, 2^{61} - 1, 2^{89} - 1, 2^{107} - 1$ and $2^{127} - 1$. Let a message string $M = (m_0, m_1, ...)$ consisting of integers between 0 and $2^w - 1$ be partitioned into blocks $M_0, M_1, ..., M_s$ each containing at most b integers so that

$$M_j = m_{bj}, m_{bj+1}, ..., m_{b(j+1)-1}, j = 0, 1, ..., s - 1,$$
$$M_s = m_{bs}, m_{bs+1}, ..., m_{bs+t}, \text{ for some } t \leq b - 1.$$

Use the stream cipher to generate $s + 2$ outputs $z_i, z_{i+1}, z_{i+2}, ..., z_{i+s+1}$. The icv is calculated as

$$icv(M, b, p, z_i, z_{i+1}, ..., z_{i+s+1})$$
$$= (f(M_0, z_i) + f(M_1, z_{i+1}) + ... + f(M_s, z_{i+s}) + z_{i+s+1})[p] \quad (1)$$

where for any message string $N = (n_0, n_1, ..., n_r)$, and integer x

$$f(N, X) = (...((n_0 x + n_1)x + n_2)x + ... + n_r)x[p]. \quad (2)$$

Equivalently $f(N,x)$ may be expanded as

$$f(N,x) = (n_0x^{r+1} + n_1x^r + n_2x^{r-1} + ... + n_rx)[p].$$

Example 1. Let $w = 30, p = 2^{31} - 1, b = 20$. Let the cipher be in state 56 (the last output produced being z_{55}). Let $M = (m_0, m_1, ..., m_{108})$ be a message string of length 109 that requires integrity. M is divided into blocks $M_0, M_1, ..., M_4$ of length 20 where $M_i = (m_{20i}, m_{20i+1}, ...m_{20i+19}), i = 0, 1, ..., 4$, and one block of 9 integers $M_5 = (M_{100}, m_{101}, ..., m_{108})$. The cipher is used to generate 7 outputs $z_{57}, z_{58}, ..., z_{62}$, and the integrity check value

$$icv(M, 20, 2^{31} - 1, z_{56}, z_{57}, z_{58}, ..., z_{62})$$

is calculated according to (1) and (2). The transmitted message is then

$$m_0, m_1, ..., m_{108}, icv.$$

3 Analysis

The following Theorem and Corollaries establish a clear link between the strength of the integrity mechanism and the strength of the stream cipher from which it is constructed.

Theorem 1. *Let $p > 2^w$ be prime, and the function f be defined by (2). Let M and M' be any two unequal message strings of length b, and y any fixed integer. Then if x is a uniformly distributed random variable in the range 0 to $2^w - 1$,*

$$Probability(f(M,x) - f(M',x) \equiv y(modp)) \leq \frac{b}{2^w}.$$

Proof. Let $M = (m_0, m_1, ..., m_{b-1})$ and $M = (m'_0, m'_1, ..., m'_{b-1})$. By expanding (2),

$$\begin{aligned} &f(M,x) - f(M',x) \\ &\equiv ((m_0 - m'_0)x^b + (m_1 - m'_1)x^{b-1} + ... + (m_{b-1} - m'_{b-1}x)(modp). \end{aligned}$$

Thus

$$f(M,x) - f(M',x) \equiv y(modp)$$

if and only if

$$(m_0 - m'_0)x^b + (m_1 - m'_1)x^{b-1} + ... + (m_{b-1} - m'_{b-1})x - y \equiv 0(modp).$$

By a standard result of elementary number theory (see [11] p58) such an equivalence has at most b solutions in x, from which the result follows. □

Corollary 2 is a straightforward consequence of this theorem.

Corollary 2. *Let M and M' be any two unequal message strings, and y any fixed integer. Let the function icv() be defined as in (1) and (2). Then if $z_i, z_{i+1}, z_{i+2}, \ldots, z_{i+s}$ are independent and uniformly distributed random variables in the range 0 to $2^w - 1$,*

$$Probability(icv(M, b, p, z_i, z_{i+1}, ..., z_{i+s+1}) - icv(M', b, p, z_i, z_{i+1}, ..., z_{i+s+1}) \equiv y(modp)) \leq \frac{b}{2^w}.$$

Corollary 3 indicates the strength of the integrity mechanism in terms of the likelihood of replacing, in transit, a message and the corresponding icv with a legitimate, but different, message-icv pair.

Corollary 3. *Let M and M' be any two unequal message strings, and y, g any fixed integers. Let the function icv() be defined as in (1) and (2). Then if $//z_i, z_{i+1}, z_{i+2}, ..., z_{i+s+1}$ are independent and uniformly distributed random variables in the range 0 to $2^w - 1$,*

$$Probability(icv(M', b, p, z_i, z_{i+1}, ..., z_{i+s+1}) \equiv y(modp) \mid icv(M, b, p, z_i, z_{i+1}, ..., z_{i+s+1}) \equiv g(modp)) \leq \frac{b}{2^w}. \quad (3)$$

Proof. Clearly the left hand side of the inequality in (3) is equal to

$$Prob(icv(M, b, p, z_i, z_{i+1}, ..., z_{i+s+1}) - icv(M', b, p, z_i, z_{i+1}, ..., z_{i+s+1}) \equiv g - y(modp) \mid icv(M, b, p, z_i, z_{i+1}, ..., z_{i+s+1}) \equiv g(modp)).$$

However from (1)

$$icv(M, b, p, z_i, z_{i+1}, ..., z_{i+s+1}) - icv(M', b, p, z_i, z_{i+1}, ..., z_{i+s+1}) \equiv (f(M_0, z_i) + f(M_1, z_{i+1}) + ... + f(M_s, z_{i+s}) - f(M'_0, z_i) + f(M'_1, z_{i+1}) + ... + f(M'_s, z_{i+s}))(modp)$$

is independent of z_{i+s+1} while

$$icv(M, b, p, z_i, z_{i+1}, ..., z_{i+s+1}) \equiv g(modp))$$

if and only if

$$z_{i+s+1} \equiv (g - f(M_0, z_i) - f(M_1, z_{i+1}) - ... - f(M_s, z_{i+s}))(modp).$$

Thus the events described in the conditional probability of (3) are independent and so the left hand side of the inequality (3) is equal to

$$Prob(icv(M, b, p, z_i, z_{i+1}, ..., z_{i+s+1}) - icv(M', b, p, z_i, z_{i+1}, ..., z_{i+s+1}) \equiv (g - y)(modp)).$$

The result now follows by Corollary 2. □

It follows from Corollary 3 that with an idealised stream cipher (with outputs that are uniformly distributed independent random variables) if any message and its integrity check value were to be altered in transit the new message and integrity check value would register as valid by the receiver with probability at most $b/2^w$. Moreover this is quite independent of whatever calculating ability or amount of chosen messages and corresponding icvs the party altering the message has. Thus we refer to $log_2(2^w/b)$ as the *effective icv length*. For the effective icv length to be a meaningful indicator of integrity strength with a practical deterministic cipher however, clearly the stream cipher key length must be at least as large as the effective icv length. On the other hand the contrapositive of Corollary 3 shows that if there is some way of altering or substituting message-icv pairs that goes undetected with a probability of more than $b/2^w$ then there must be some corresponding level of predictability in the stream cipher output. We illustrate this point with an example.

Example 2. Consider a stream cipher with a set V of initial vectors that determine the starting state of the cipher. For v in V let z_i^v denote the ith output of the stream cipher with initial vector v. Let the block length b equal 1. Assume that for some cipher position i there are a pair of unequal single integer messages m and m' and a function F that can calculate the icv of m' from that of m for all initial vectors v (so the integrity mechanism can be successfully attacked with probability 1). Furthermore assume that F can be evaluated in a reasonable amount of time (for example F should not embody a search through all initial vectors). Thus

$$icv(m', 1, p, z_i^v, z_{i+1}^v) = F(icv(m, 1, p, z_i^v, z_{i+1}^v)), \text{ for all } v \in V. \tag{4}$$

Subtracting the icv of m from both sides of (4) and expanding the l. h. s. according to (1) and (2)

$$\begin{aligned}(m'z_i^v + z_{i+1}^v)[p] - (mz_i^v + z_{i+1}^v)[p] = \\ F(icv(m, 1, p, z_i^v, z_{i+1}^v)) - icv(m, 1, p, z_i^v, z_{i+1}^v)\end{aligned}$$

which implies that

$$(m' - m)z_i^v[p] = F'(icv(m, 1, p, z_i^v, z_{i+1}^v)) \tag{5}$$

for a suitable function F'. Let S be the set of numbers s from 0 to $2^w - 1$ for which the equation

$$(m' - m)s[p] = F'(t)$$

has exactly one solution for t among $0, 1, ..., p - 1$. Then since $(m' - m)s[p]$ is a one to one function of s on $0, 1, ..., 2^w - 1$, it follows that S contains at least $2^w - k - 2$ integers. We may then define the inverse of F' on S, and by rearranging (5)

$$icv(m, 1, p, z_i^v, z_{i+1}^v) = F'^{-1}((m' - m)z_i^v[p]),$$

provided z_{i+1}^v is in S. By expanding the icv function from (1) and (2) and isolating z_{i+1}^v

$$z_{i+1}^v = (F'^{-1}((m'-m)z_i^v[p]) - mz_i^v)[p] \text{ for all } v \in V, z_i^v \in S.$$

Thus z_{i+1}^v can be obtained from z_i^v unless z_i^v happens to be one of at most $k+1$ values not in S. If the function F'^{-1} can be evaluated in a reasonable amount of time this amounts to an effective attack on the stream cipher.

4 ICV Length

In many conventional integrity mechanisms the icv is calculated as a function of a fixed key and the message. In such systems different messages that share the same icv (called collisions) may be found by an attacker who collects sufficiently many message-icv pairs. According to the so called Birthday Paradox (see [15]) the number of message-icv pairs required is approximately equal to the square root of the number of possible icvs. For this reason the length of the icv is usually chosen to be quite large (typically 128 bits) so that finding collisions is not feasible. In the method described in this article however collisions do not exist in the sense described above because the integrity function (1) and (2) uses fresh output from the stream cipher for each new message. Because of this it is suggested that the icv length may be much smaller than in conventional integrity systems.

Notwithstanding these remarks we describe how to modify the integrity method to increase the effective icv length by any required factor h. As in Section 2 let a message string $M = (m_0, m_1, ...)$ be divided into blocks $M_0, M_1, ..., M_s$ each containing at most b integers. Use the stream cipher to generate $h(s+2)$ outputs $z_i, z_{i+1}, z_{i+2}, ..., z_{i+h(s+2)-1}$. The integrity check value icv_h is defined as the concatenation of h integers between 0 and $p-1$ calculated according to

$$\begin{aligned}
&icv_h(M, b, p, z_i, z_{i+1}, ..., z_{i+h(s+2)-1}) = \\
&(f(M_0, z_i) + f(M_1, z_{i+1}) + ... + f(M_s, z_{i+s}) + z_{i+s+1})[p]|| \\
&(f(M_0, z_{i+s+2}) + f(M_1, z_{i+s+3}) + ... + f(M_s, z_{i+2s+2}) + z_{i+2s+3})[p]|| \\
&. \\
&. \\
&(f(M_0, z_{i+(h-1)(s+2)}) + ... + f(M_s, z_{i+h(s+2)-2}) + z_{i+h(s+2)})[p],
\end{aligned}$$

where the function f is given by (2). If $p < 2^{w+1}$, this provides an icv of length $h(w+1)$ bits with an effective icv length of

$$log_2\left(\left(\frac{2^w}{b}\right)^h\right) = hlog_2\left(\frac{2^w}{b}\right) = h(w - log_2 b). \tag{6}$$

Thus, for example, if $b = 20, w = 30, p = 2^{31} - 1, h = 4$ then the icv has length 124 bits while the effective icv length is

$$4(30 - log_2 20) \approx 102.7.$$

Note that this method of increasing the icv length by a factor of h requires the same increase factor in the amount of icv calculations required. As well the output from the stream cipher must be increased by a factor of h in order to complete the icv calculations. This however may be compensated for by increasing the block length b so the number of stream cipher outputs per message length remains approximately constant.

The other obvious way to increase the icv length by a factor of h is to increase the message and cipher integers to $w' = hw$ bits and take a prime $p' > 2^{w'}$. If p' can be chosen close to a power of 2 (say $p' = 2^v - 1$) then this method enjoys several advantages over the above method. Firstly the effective icv length becomes $hw - log_2 b$ which is larger than the corresponding $h(w - log_2 b)$ from (6). Also the output from the stream cipher required to process a given amount of message is not increased since a given message is divided into $1/h$ as many integers as before. Finally the multiplication of numbers modulo p' may take as much as h^2 times as much calculation as multiplication modulo p. However only $1/h$ as many multiplications are required to process a given amount of message. Thus the amount of calculations required to calculate the icv is increased by a factor of at most h (this figure may be further reduced for large h, see [7] pp 278-301 for a discussion of the complexity of multi-precision multiplication).

5 Integrity and Confidentiality

To provide both integrity and confidentiality the stream cipher can generate different outputs for both the integrity calculation and for message encryption. In order to prevent the message integrity from being undermined by a known plaintext attack it is important that cipher output that is used in icv calculations by the receiver could never be used for message encryption by the sender. To overcome this problem it is suggested that some integer $d < b$ is chosen and that only those z_i for which i is a multiple of d are used in the icv calculation, the remaining z_i being used for encryption.

Example 3. As in Example 1 let $w = 30, p = 2^{31} - 1, b = 20, M = (m_0, m_1, ..., m_{108})$ and the cipher be in state 56. Further let $d = 10$. To apply integrity and confidentiality the cipher is used to generate 121 outputs $z_{56}, z_{57}, z_{58}, ..., z_{176}$, and the icv is calculated as

$$icv(M, 20, 2^{31} - 1, z_{60}, z_{70}, ..., z_{120})$$

where the icv function is defined by (1) and (2). The transmitted message is

$$(m_0 + z_{56})[2^{30}], (m_1 + z_{57})[2^{30}], ..., (m_3 + z_{59})[2^{30}], (m_4 + z_{61})[2^{30}],$$
$$..., (m_{108} + z_{176})[2^{30}], icv.$$

Finally we report on the performance of the combined confidentiality and integrity scheme using a stream cipher and implemented in computer software. The stream cipher used was constructed from three linear feedback shift registers and a combining function with memory (see [13] for designs of this type). In the icv calculation $w = 30, p = 2^{31} - 1$ and $b = 16$. This led to an effective icv length of 26 (thus the attackers ability to attack the integrity without attacking the stream cipher itself is one chance in 2^{26} or 68288512). A throughput of approximately 4 Mbits/second was attained for the entire system using a PC containing the Intel 486 processor running at 33 Mhz.

Acknowledgement

The permission of the Managing Director, Research and Information Technology, Telecom Australia to publish this paper is hereby acknowledged. Also the Author would like to thank the referees for several constructive comments on the draft version of this paper.

References

1. Brassard, G.: On Computationally Secure Authentication Tags. Advances in Cryptology-CRYPTO'82, proceedings, Springer-Verlag (1983) 79–86
2. Carter, J. L., Wegman, M. N.: Universal Classes of Hash Functions. Journal of Computer and Systems Sciences **18** (1979) 143–154
3. Damgaard, I. B.: A Design Principle for Hash Functions. Advances in Cryptology-CRYPTO'89, proceedings, Springer-Verlag (1990) 416–427
4. ISO/IEC 9797: Data Cryptographic Techniques-Data Integrity Mechanism using a Cryptographic Check Function employing a Block Cipher Algorithm. International Organisation for Standardisation (1989)
5. Jueneman, R. R.: A High-Speed Manipulation Detection Code. Advances in Cryptology-CRYPTO'86, proceedings, Springer-Verlag (1987) 327–346
6. Knobloch, H. J.: A Smart Card Implementation of the Fiat-Shamir Identification Scheme. Advances in Cryptology-EUROCRYPT'88, proceedings, Springer-Verlag (1989) 87–96
7. Knuth, D.: The Art of Computer Programming. Vol. 2, 2nd edition, Addison-Wesley, Reading, Mass. (1981)
8. Lai, X., Massey, J. L.: Hash Functions based on Block Ciphers. EUROCRYPT'92, extended abstracts (1992) 53–67
9. Lai, X., Rueppel, R. A., Woollven, J.: A Fast Cryptographic Checksum Algorithm based on Stream Ciphers. AUSCRYPT'92, abstracts (1992) 8-7–8-11
10. Merkle, R. C.: One Way Hash Functions and DES. Advances in Cryptology-CRYPTO'89, proceedings, Springer-Verlag (1990) 428–446
11. Niven, I., Zuckerman, H. S.: The Theory of Numbers (fourth edition). John Wiley and Sons, New York-Chichester-Brisbane-Toronto (1980)
12. Rivest, R. L.: The MD4 Message Digest Algorithm. Advances in Cryptology-CRYPTO'90, proceedings, Springer-Verlag (1991) 303–311
13. Rueppel, R. A.: Analysis and Design of Stream Ciphers. Springer-Verlag, Berlin (1986)

14. Wegman, M. N., Carter, J. L.: New Hash Functions and their use in Authentication and Set Equality. Journal of Computer and System Sciences **22** (1981) 265–279
15. Yuval, G.: How to Swindle Rabin. Cryptologia **3** (1979) 187–189