Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®
Computer Standards & Interfaces 27 (2005) 101–104
COMPUTER STANDARDS & INTERFACES
www.elsevier.com/locate/csi

Protection for software in measuring instruments

Ales Vobornik*

KET, ZCU, Petatricatniku 14, Plzen 30614, Czech Republic

Received 24 May 2004; accepted 28 May 2004
Available online 25 June 2004

Abstract

A protection of measuring instrument control programs and control system data against unauthorized change is necessary for provision of functionality of entire measuring instrument. Cyclic codes or hashing functions can be used as such protection. Application of the MD5 hashing function is illustrated on practical example.

Keywords: Control program; Cyclic code; Digital signature; Hashing function

1. Introduction

A basis of modern measuring instruments is microprocessor that controls all functions of the instrument. Thus software of this microprocessor imminently affects not only functions of the instrument but also its parameters because measured signals are processed by means of its program. Hence the software should be properly protected against accidental and also intentional unauthorized change. A principle of this protection would be the integrity check—not only for control program itself but especially for all data, i.e. correction and calibration coefficients. These data are preferred to be stored separately, out of control program, by reason of easy calibration of the instrument.

The protection against software copying is understood as subsidiary problem because the reengineering and copying of the measuring instrument is distinctively more complicated a problem than copying a control program only.

2. Feasible solutions

The integrity check can be performed by many ways. One of the usually used possibilities is application of the cyclic codes—CRC that are utilized at data transmission. Another, more exacting possibility is the utilization of hashing functions that are used in various cryptosystems.

The cyclic codes create an extensive group of the safety codes. Their advantage is excellent protection characteristics and easy technical realization. Mathematical calculation is firstly based on dividing of a polynomial by polynomial. So called ‘‘generator polynomial’’ takes up the essential position. Its convenient choice can affect a probability of software change detection.

If the generator polynomial will have more than one term, one changed bit in the block will be always

* Tel.: +42-0377634571; fax: +42-0377634502.
E-mail address: vobornik@ket.zcu.cz (A. Vobornik).

0920-5489/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.csi.2004.05.005

detected. Double change will be detected every time when the generator polynomial will have a three-membered coefficient. The odd number of changes will be detected every time when the generator polynomial will have a type $(x+1)$ coefficient. A burst of changes will be detected every time when they will be shorter than generator polynomial. Otherwise the change will be detected with probability less than 1. Thus length of the generator polynomial determines the length and probability of burst changes detection. Consequently, the cyclic codes can be highly used to protect data transmission but actual security of data integrity would lead to very long and therefore impractical code.

The hashing functions were directly developed for input data copy generation and they are used for messages protected by electronic signature, for passwords protection, etc. The input data for hashing function is the message (data block) with variable and practically unrestricted length. The hashing function output is hashing value with fixed and relatively short length. The strict safety requirements are imposed to the hashing function. First of all this function must not be reversible. It means that the input data cannot be obtained from the hashing value. Further phenomenon of the hashing function is its collision immunity, which means that it is very difficult to find two messages with the same hashing value. One can thus expect that the hashing function will comply with all the requirements for provision of measuring instrument data integrity.

There exist many hashing functions. Most of them can be classified to MD*x*, RIPEMD-*x* and SHA-*x* classes.

To the MD*x* class functions belongs function MD2 that is very slow and obsolete with considerable risk of collision. Further, to the MD*x* class functions belong function MD4, which is very fast but during its application, collisions were insisted. From aspect of safety this function is thus unsuitable. The latest MD*x* class function is function MD5, which is still fast but its author [1] does not recommend it from the point of view of collision possibility. The function MD5 was yet recently very often used without detection of any collision.

To the RIPEMD-*x* class functions belong functions RIPEMD, RIPEMD-128 and RIPEMD-160. These functions are more complicated and consequently significantly slower. The SHA-*x* class functions, to which belong functions SHA-0, SHA-1, SHA-256 and SHA-512, are more complicated and consequently slower than the functions of class MD*x*. Function RIPEMD-160 and all SHA-*x* class functions, with exception of function SHA-0, are considered as safety.

Comparisons of algorithms relative speeds are in Table 1. Table 1 illustrates brisk fall of the speed in accordance with output hashing code rising length. The control microprocessors used at present in measuring instruments are mainly of 16-bit architecture but there exist solutions with powerful 8-bit microprocessors. Further comparison was performed with microprocessor of class 8051 with clock frequency 12 MHz, which is probably the lowest class of microprocessor being used in measuring instruments. The processing speeds and code lengths of corresponding functions, which also affect possibilities of practical utilization, are in Table 2. The testing programs were written and debugged in the C programming language. The speed was tested on a block of 32 kB length.

The cyclic functions can be realized by hardware but in this case they are realized by software in two ways. Both ways realize one function but in the second, faster case the calculation is simplified by using table. A rise of speed caused by utilization of the table of values is more apparent for shorter CRC codes. Slight accelerations for longer CRC codes are caused especially by the necessity to process numbers that are longer than the length of microprocessor word. The CRC codes longer than CRC32, i.e. CRC64, CRC128, were not tested because additional processing deceleration, i.e. processing speeds lower than at function MD5, can be expected.

Table 1
Hashing algorithms relative speeds according to Ref. [5]

Algorithm	Code length [*b*]	Relative speed
MD4	128	1
MD5	128	0.68
RIPEMD-128	128	0.39
SHA-1	160	0.28
RIPEMD-160	160	0.24
SHA-256	256	0.12
SHA-512	512	0.03