International Journal of Science Engineering and Advance Technology

Implementation and Design of SHA-1 Algorithm

[1]Chandra sekhar murala,[2]M.Purnasekhar
[1]Research scholar(M.Tech), [2]Asst. professor
Dept.of CSE,Nova College of Engineering & Technology, Vegavaram,Jangareddygudem
chandumurala@gmail.com

Abstract:
The substantive technologies are having so plenty algorithms for crypto systems. All the technologies are famous for message authentication or message compression techniques like DSA, RSA and SHA-1.These are majorly streamed on compressed techniques only not on Authentication Requirements like Masquerade, Content Modification, Sequence Modification and Timing Modification. However the previous SHA-1 failure using brute force attack in 2^{80} operations and Collision failure found in 2005 in 2^{33} operations. So we address the above problem and focused on SHA-2 cryptography algorithms.
Keywords: SHA-2, plenty, cryptography, masquerade.

I Introduction:

Hashing is a method of inserting data into a table. Tables can be implemented in many ways.Examples include a fixed array (limiting number of elements), array of linked lists (potentially unlimited number of elements)There is the potential to retrieve data faster.Using the proper hash function will distribute the elements throughout the table. To retrieve the element, apply the hash function until it is found or it is clear that it was not found. We represent a table for crypography primitives.

	DES	AES
Date	1976	1999
Block size	64 bits	128 bits
Key length	56 bits	128, 192, 256, ... bits
Encryption primitives	Substitution and permutation	Substitution, shift, bit mixing
Cryptographic primitives	Confusion and diffusion	Confusion and diffusion
Design	Open	Open
Design rationale	Closed	Open
Selection process	Secret	Secret (accepted public comm
Source	IBM, enhanced by NSA	Belgian cryptographers

Table 1: cryptography primitives.

II Related Works

Every mechanism based on

- **Message Digest Functions**

Protect integrity Create a message digest or fingerprint of a digital document like MD4, MD5, SHA **Message Authentication Codes** (MACs) Protect both integrity and authenticity Produce fingerprints based on both a given document and a secret key.

Checksums → fingerprint of a message

If message changes, checksum will not match Most checksums are good in detecting accidental changes made to a message They are not designed to prevent an adversary from intentionally changing a message resulting a message with the same checksum. Message digests are designed to protect against this possibility.

Secure Hash Functions:

The first secure hash function is SHA-0 is established in 1993,then SHA-1 is designed in the year of 1995,our mechanism SHA-2 established in 2002.This document specifies a Secure Hash Algorithm, SHA-1, for computing a condensed representation of a message or a data file. When a message of any length < 2^64 bits is input, the SHA-1 produces a 160-bit output called a message digest. The message digest can then, for example, be input to a signature algorithm which generates or verifies the signature for the message. Signing the message digest rather than the message often improves the efficiency of the process because the message digest is usually much smaller in size than the message. The same hash algorithm must be used by the verifier of a digital signature as was used by the creator of the digital signature. Any change to the message in transit will, with very high probability, result in a different message digest, and the signature will fail to verify In SHA-2 we are having bellow bit padding versions. These are SHA-224, SHA-256, SHA-384, SHA-512.

Padding : the total length of a padded message is multiple of 512 . Every message is padded even if its length is already a multiple of 512 .Padding is done by appending to the input A single bit, 1 Enough additional bits, all 0, to make the final 512 block exactly 448 bits long A 64-bit integer representing the length of the original message in bits. Once these steps have been performed on each

512-bit block (B_1, B_2, …, B_n) of the padded message.

	Output size (bits)	Internal state size (bits)	Block size (bits)	Max message size (bits)	Word size (bits)	Rounds	Operations	Colli fo
SHA-0	160	160	512	$2^{64}-1$	32	80	+, and, or, xor, rot	Y
SHA-1	160	160	512	$2^{64}-1$	32	80	+, and, or, xor, rot	N (2^{52} a
SHA-2	256/224	256	512	$2^{64}-1$	32	64	+, and, or, xor, shr, rot	N
	512/384	512	1024	$2^{128}-1$	64	80	+, and, or, xor, shr, rot	N

Table 2: Sha Algorthms Representations.

III Proposed Work:

SHA-2 operates in much the same way as SHA-1, but because values of 256, 384 and 512-bits are possible, the primary differences are more initial blocks of messages (8 instead of 5), fewer numbers of rounds (64 instead of 80), the use of right shifts as well as left shifts (SHA-1 only uses left shifts), constants for each round instead of blocks of rounds and the use of 64-bit inputs on the 512-bit function over the 32-bit for all others [7]. Messages are initially padded from 16 to 64 rounds using logical calculations based on the inputted message and each round is calculated using six variables based on the eight message blocks and 64 rounds. Unlike SHA-1, where there are four round calculations, SHA-2 only uses one. Its inherent strength comes from the use of 64 round constants over the four in SHA-1. This greatly reduces the risk of collisions and to date none have been found.Although SHA-2 is increasing in popularity, SHA-1 is still significantly used, not least for its incorporation into the Trusted Platform Module (TPM) ASIC [4]. However, TPM are currently investigating the use of SHA-2 in later modules [21]. If this takes place, the ASIC will also have to contain backwards-compatibility to SHA-1 for communication with older systems. Therefore, it is important that an ASIC can manage creation of SHA-1 and SHA-2 message digests at a speed where delay would not be noticed on a high-speed internet connection ([22] suggests a speed of 40Gbit/sec for a Fibre-optic line).

IV Conclusion:

The actual hash results outputted were not correct, which is believed to be an issue within the firmware program. This program was created at a low-level without the use of a complier or syntax checker. Therefore, a small error could have been generated which has not beensuccessfully debugged and would be difficult to find without a time consuming line-by-line simulation of the program. If an established microcontroller or FPGA were used instead of a custom-designed ASIC, the program could be created in a higher level language such as C and compiled into an assembly instruction set. If the goal still were to create an ASIC with no existing compiler, a program such as AWK or the C Pre-processor could be used to compile the program from a high-level language. This would make the code more user-readable and allow debugging in a more methodical and easier manner. With extra time to perform these tasks, it is highly likely that the faults within the program could be identified and removed, giving correct hash functions for SHA-2.

References:

[1] Bruce Schneier, “Applied Crptography”, John Wiley and Sons, Inc.
Press, 1996.Ding, W. and Marchionini, G. 1997 A Study on Video Browsing Strategies. Technical Report. University of Maryland at College Park.
[2] NIST, “Secure Hash Standard”, FIPS PUB 180-1, May 1993.
[3] NIST, “with change notice Secure Hash Standard”, FIPS PUB 180-2 August 2002.
[4] NIST, “Digital Signature Standard (DDS)”, FIBS PUB 186 May 1994.
[5] “An Overview of Cryptographic Hash Functions and Their Uses”, SANS Institute, 2003
[6] Federal Information Processing Standards. Digital Signature Standard (DSS). FIPS PUB 186-2, January 27, 2000.
[7] “Advanced Hash Calculator”, www.filesland.com, Version 2.33.
[8] Yong Kyu Kang, Dae Won Kim, Taek Won Kwon, Jun Rim Choi, “An Efficient Implementation of Hash Function Processor for IPSEC”, In Proceedings of the IEEE Asia- Pacific Conference on ASIC, pp. 93-96, August. 2002.
[9] N. Sklavos, G. Dimitroulakos, O. Koufopavlou, “An Ultra High Speed Architecture for VLSI Implementation of Hash Functions”, Proceedings of the 10th IEEE International Conference on Electronics, Circuits and Systems (ICECS 2003) pp. 990-993 Vol.3, 2003.
[10] Harris Michail, Athanasios P. Kakarountas, OdysseasKoufopavlou, Costas E. Goutis, “A Low Power and High Throughput Implementation of the SHA-1 Hash Function”, IEEE International Symposium on Circuits and Systems (ISCAS 2005), pp. 4086-4089 Vol.4.,2005.