DALLAS SEMICONDUCTOR MAXIM

1-WIRE® DEVICES

May 13, 2005

White Paper 9: Are SHA-1 Devices Still Secure Enough?

Recently, a group of Chinese researchers attacked the strength of the Secure Hash Algorithm (SHA1). This white paper discusses that attack and shows that, although the algorithm is slightly less collision-resistant than previously thought, the security of the SHA-1 memory devices from Dallas Semiconductor Maxim is not affected. Thus, the company's SHA-1 memory devices (DS1963S, DS1961S, and DS2432), will continue to provide a low-cost, effective solution for accessory/peripheral authentication and tamper-proof, verifiable memory applications.

Introduction

The SHA-1 memory devices from Dallas Semiconductor/Maxim provide a low-cost and highly effective solution for accessory/peripheral authentication and tamper-proof, verifiable memory applications. The ability to authenticate these SHA-1 devices is a feature which is useful in applications where there is a need to keep counterfeiters away, such as high-volume consumables, high-value hardware, hardware license management, building access control, or vending.

Ultimately, the utility of these devices depends on the robustness and security of the Secure Hash Algorithm, as defined by the U.S. National Institute of Standards and Technology (NIST) in Federal Information Processing Standards Publication 180-1 (FIPS PUB 180-1) and ISO/IEC 10118-3. Recently, a group of Chinese researchers attacked the strength of this algorithm (see Note 1). This application note shows that, although some applications which use SHA-1 might need to reevaluate the security provided, the security of Dallas Semiconductor/Maxim SHA-1 memory devices is not compromised by this research claim.

The Attack on the SHA-1 Digest

The FIPS PUB 180-1 defines SHA-1 as a standard for computing condensed representations of data in a secure way. As defined in the document, the algorithm is considered secure because of two claims, "(1) it is computationally infeasible to find a message which corresponds to a given message digest, or (2) to find two different messages which produce the same message digest." The first assertion implies both that the resulting output of the SHA-1 algorithm does not contain enough information to discover the full text of the input to the algorithm (i.e., the algorithm is irreversible), and that it would take vast resources and time to find a corresponding message (input) if only the digest (output) was available. The second assertion implies that it would take vast resources and time to find two unique inputs which resulted in the same output (i.e., that the algorithm is resistant to collisions). These assumptions do not claim that there are no two messages which share the same digest; they just mean that it is prohibitively difficult to find them.

The theoretical bound for the number of hash operations required to find a collision (a pair of messages which share the same digest) is 2^{80} operations (see Note 2). The researchers' recent attack on SHA-1 claims to lower this bound to 2^{69} operations. This latter claim weakens the second SHA-1 assertion by, in effect, reducing the "computational infeasibility" by an order of 2^{11}. This does not mean that it is no longer "computationally infeasible to find two different messages which produce the same message digest," only that it is slightly less infeasible than previously thought. Furthermore, the researchers' claim does not mean that it is no longer "computationally infeasible to find a message which corresponds to a given message digest." This is because this new attack depends on the ability to select both input messages carefully. The only proven SHA-1 security attack for finding a message—not necessarily THE message—which corresponds to a given digest requires a brute-force search with 2^{160} operations.

Although the second SHA-1 algorithm assertion is weakened by this new Chinese research, there is no reason to suspect that the research will lead to any attacks on the first SHA-1 claim. Thus in summary, SHA-1 is still irreversible, but perhaps slightly less collision-resistant. This could, nonetheless, be an alarming result for applications that depend on digital signatures, such as time-stamping or document notarization. As much of the information in the input message is contextual for these applications, it still remains to be seen if there will be effective, application-specific attacks to come from the Chinese research.

The Message Authentication Code

The Dallas Semiconductor/Maxim SHA-1 memory devices depend on a Message Authentication Code (or MAC) in both directions of communication. Computing the MAC simply requires taking a publicly visible input string (made up of the memory contents, the device's unique serial number, and a random challenge) and combining it with a secret key as the input message for the SHA-1 algorithm. The resulting digest (or hash) is referred to as the MAC. Transmitting the MAC along with the message provides a secure means of proving that you know the secret key and that nothing tampered with the data during transmission. During a read operation, the SHA-1 memory device responds with a MAC, proving that it is authentic and that the host received the data correctly. During a write operation, the host provides a MAC to prove that it is authorized to make changes to the device's memory contents and that the device received the new memory contents correctly.

A successful attack on the algorithm of this MAC-based security system would require discovery of the secret key. In most of the available SHA-1 memory devices, this key is a 64-bit write-only value. (Newer devices may soon be available with an even larger key.) An attacker could issue a challenge to a device, read the resulting MAC, and then execute a brute-force search of the full 64-bit value until it found a MAC that matched. This action would require 2^{64} SHA-1 operations, which would require more than a decade's worth of computing time on a 64 CPU Cray X1 supercomputer (see Note 3).

Finding a message, which matches the digest of any given input message, requires 2^{160} operations (far more than the 2^{64} required to find the secret key). Because the length of our input message is fixed at 512 bits, and 448 of those bits are known public data, the most direct approach is to search for the correct value of the remaining 64 bits (i.e., the secret-key value). As long as it is "computationally infeasible to find a message which corresponds to a given message digest," there can be no attack more successful than a brute-force search for the secret-key value.

Note that although the complexity of 2^{64} operations for a secret-key search is less than the 2^{69} operations required to find a pair of messages that collide, there is no comparison in terms of the class of the attack. If the research team found a method of finding collisions in SHA-1 in 2^{50} operations, the secret-key search would still require 2^{64} SHA-1 operations to find the secret-key value. Consequently, the new attack for finding a collision between any two input messages can not be used to find a collision for a given, fixed input message because it requires selecting the input messages carefully.

Summary

There have been well-documented attacks on the SHA-1 memory devices that are related to the system in which those devices are used (see, *Whitepaper 3: Why are SHA-1 Devices Secure?*). However, using the publicly readable MAC to find the hidden secret key is the only known attack protected by the strength of the chosen algorithm. In the case of SHA-1, we know that the algorithm was defined with two distinct strengths: collision-resistant and irreversibity. A recent attack showed that the SHA-1 algorithm is slightly less collision-resistant than previously thought, but that attack did not affect the security of the SHA-1 memory devices from Dallas Semiconductor/Maxim.

NOTES:

1. Collision Search Attack on SHA-1 (PDF)

2. Finding collisions in SHA-1 has an upper bound of 2^{80} because of the "birthday paradox." Basically, this paradox says that if you are attempting to match any two elements with n-bits of output, you only have to consider $2^{(n/2)}$

elements, not $2^{(n)}$ elements, to have an extremely high probability of finding a match. This is a well-known cryptographic property of all hashes and is determined only by the number of bits in the output.

3. The SHA-1 algorithm performs about 1740 basic arithmetic operations on the message block elements. Assuming 20% overhead for additional manipulation, a complete turn of the algorithm will run in 2100 clock cycles. Using a Cray X1 supercomputer with 64 CPUs (currently Cray's largest scale, single-cabinet configuration) operating at peak performance of 819 gigaflops, it would require 12.4 years of continuous operation to generate a complete look-up table. Cray's largest advertised X1 system, with 64 cabinets, would take over two months. That much computing power makes this type of attack cost-prohibitive.

More Information

DS1961S:	QuickView	-- Full (PDF) Data Sheet	-- Free Samples
DS1963S:	QuickView	-- Full (PDF) Data Sheet	-- Free Samples
DS2432:	QuickView	-- Full (PDF) Data Sheet	-- Free Samples