



Keywords: SHA, SHA-1, DS1963S, DS1961S, DS2432, MAC, cryptanalysis, attack, copy, replay, eavesdrop, A-B-A, emulation, brute force, microprobe, competitor, security, access control, ecash, authentication, challenge, response Jun 07, 2002

APPLICATION NOTE 1098

White Paper 3: Why are 1-Wire SHA-1 Devices Secure?

Abstract: This document shows why 1-Wire® and iButton® SHA-1 devices are secure by introducing possible attack scenarios and explaining how the SHA-1 devices counter the attack in hardware or in a recommended usage procedure. Data and hardware authentication with 1-Wire devices are the key concepts presented for various types of service applications including access control and electronic cash. Since 1-Wire/iButton SHA-1 devices are portable tokens that can create a message authentication code (MAC) based on a random challenge using a cryptographically good hash; they are an ideal choice for the above-mentioned service applications.

Introduction

Before reading this document, please review White Paper 8: [1-Wire SHA-1 Overview](#) for a description of SHA-1 and the available 1-Wire SHA-1 devices. The overview also presents these devices in the context of a service provider for electronic cash (eCash) tokens.

This document introduces possible attack scenarios and explains how the 1-Wire devices counter the attack in hardware or in a recommended usage procedure. (*Special terms, commands, or codes are shown in italics for clarity.*) A glossary of terms can be found in White Paper 4: [Glossary of 1-Wire SHA-1 Terms](#).

Attacks

What is an attack? An attack is any malicious intent to subvert a system to defraud it. The motive for this varies with the service provided. In an electronic cash system (such as for vending), the motive may be free snacks. If the service is access control to a top-secret lab, the motive may be entirely different. The following attacks are ordered approximately from easiest to hardest. All of these attacks assume the attacker has access to a valid user token that is part of the service, meaning the authentication secret is set and that the data in the device is valid.

Copy Attack

The *Copy Attack* is done by copying valid service data from a device that is part of the service and writing it to another device that may or may not be part of the service. A device is part of the service if it has the correct authentication secret installed. The purpose of this attack is to take a valid token and create another one from it, thus creating two valid tokens. This could also be done to copy the attributes such as identity or monetary value (eCash) to another device to create more money or give access where none was provided.

| Conditions | Device | How the Attack is Defeated |
|---|--------------------------|--|
| Target device is <i>not</i> part of system the (authentication secret <i>not</i> set) | DS1963S, DS1961S, DS2432 | Since the authentication secret of the target device is not set to the correct value, then the device will not be accepted by the service no matter what the data is. |
| Target device is part of the system (authentication secret set correctly) | DS1963S | The target device will pass the authentication test. However there is a second MAC (message authentication code) which is embedded in the service data. This MAC (<i>Service Data Signature</i>) has the token's <i>ROM ID</i> as part of the SHA-1 calculation. Since the target device has a different <i>ROM ID</i> , then the data will be considered invalid and the device will be rejected. |
| Target device is part of the system (authentication secret set correctly) | DS1961S, DS2432 | The target device requires a valid MAC to write data to it. Since the attacker does not know the secret, writing to the device is not possible. |

Replay Attack

The *Replay Attack* attacker makes a copy of the service data of a valid device. The attacker then uses up the eCash or other usage fields. The saved data is then copied back to the device. This way, the old data is *replayed*.

| Conditions | Device | How the Attack is Defeated |
|------------|-----------------|--|
| None | DS1963S | Each page used for secure data has a read-only non-rolling-over page write-cycle counter associated with it. While the device still has a correct secret to generate the authentication MAC, the second MAC (<i>Service Data Signature</i>) that is embedded in the service data will be invalid. This MAC includes the write-cycle counter and a page number that it resides on. Writing old data back into that page or another page will invalidate it. Even rewriting the same data will invalidate it since the write-cycle counter advances. |
| None | DS1961S, DS2432 | The target device has a write protection feature that requires a valid MAC to copy data to it. Since the attacker does not know the secret, copying to the device is not possible. |

Eavesdrop Attack

The *Eavesdrop Attack* is a technique where the 1-Wire communication is monitored to reveal the secret or a repeating pattern that could be replicated.

| Conditions | Device | How the Attack is Defeated |
|---|--------------------------|---|
| Monitor and log 1-Wire communication during a valid transaction | DS1963S, DS1961S, DS2432 | When authenticating a device to verify whether it is part of the service, a random challenge is presented to be included in the MAC calculation. This random value (salt) changes the 1-Wire communication traffic significantly with each transaction. Also, at no time is the secret ever sent over the 1-Wire except when installing the secrets in a protected environment. |

A-B-A Attack

The *A-B-A Attack* tries to play two different monetary SCU (service control unit) off each other to fool them into giving more product than was paid for. This attack starts with the token being presented to the first SCU 'A'. It is then removed before the debit can be performed but close enough to the end so that the SCU thinks it was complete but without verification. This could be accomplished by monitoring the 1-Wire communication and interrupting the sequence at the appropriate time. SCU 'A' will then wait for the token to be reintroduced to verify the debit was complete. The token is then presented to another SCU 'B' and a complete debit is performed with product produced. The token is then taken back to the first SCU 'A' and it is allowed to verify that the transaction was complete. This fools 'A' into thinking it did the debit and product is given again. Consequently, two products were delivered for only one debit.

| Conditions | Device | How the Attack is Defeated |
|---|--------------------------|---|
| The transaction taking place on SCU 'A' must be stopped at the critical time. | DS1963S, DS1961S, DS2432 | A random value called a <i>Transaction ID</i> is part of the service data to make every monetary instance unique. When SCU 'A' goes back to verify that it's debit was complete, it will fail due to an incorrect <i>Transaction ID</i> . |

Emulation Attack

Use a microprocessor to emulate the behavior of the 1-Wire token. The emulator must be fast enough to respond to the 1-Wire master as if it was a real device. As shown below there is no risk as long as the attacker does not know the authentication secret. This risk is further reduced by the technique of making each authentication secret unique by including the ROM ID as a component in the calculation of the secret (*Unique Authentication Secret*).

| Conditions | Device | How the Attack is Defeated |
|-------------------------------------|--------------------------|--|
| Authentication secret is not known. | DS1963S, DS1961S, DS2432 | The emulated device can not create the correct MAC for a given challenge so would not be accepted as part of the service. |
| Authentication secret is known. | DS1963S, DS1961S, DS2432 | None Communicate with the SHA device at the fastest possible data rate. The 1-Wire master (SCU) will expect a SHA computation in 1ms to 2ms. This makes creating an emulator very challenging. Never disclose or expose the authentication secret. |

Brute Force Attack (MAC)

A *Brute Force Attack* is an inelegant enumeration of all possibilities to get a desired result. The MAC variation of this attack consists of enumerating through all of the challenge-response pairs and keeping the results in a database to be used by an emulator. This requires access to a valid 1-Wire device that is part of the service. The random challenge is three bytes long giving the number of possible challenge-response pairs at over 1.6 million. Since the resulting SHA-1 MAC is 20 bytes long, the data created from this would be: 0xFFFFFFFF (challenge) * 20 (size of MAC) = 335,544,300 bytes (320MB). This creates a pre-computed MAC look-up table that could be used by an emulator.

| Conditions | Device | How the Attack is Defeated |
|--|--------------------------|---|
| Dynamic data such as eCash. Each time the device is presented, a new service record with a <i>Service Data Signature</i> is created and written. | DS1963S, DS1961S, DS2432 | The new service record must be reread and authenticated after it is written to the token. Since there is a different MAC for authentication of the new service record, the emulator would have to know this also. The service record should contain a random value called a <i>Transaction ID</i> so the emulator will not be able to predict what the new service record will be so it could not brute force this new MAC. |
| Static data. Data is not changed, it is only authenticated. | DS1963S, DS1961S, DS2432 | After authentication of the static data, write random data to an unused page and do another authentication from that page. The emulator will not know this new MAC. |

Brute Force Attack (Secret)

The *Secret Brute Force Attack* enumerates through all possible secrets until a correct MAC is produced. A token that is part of the service supplies what a correct MAC is. If the secret is found it could be used in an emulator as long as it has a SHA-1 coprocessor that could do the hash in a millisecond. The secret is eight bytes long (FFFFFFFFFFFFFFFF hex). If it takes one microsecond to compute SHA-1 on a fast computer then it would take 580 thousand years to enumerate through all of the secrets.

| Conditions | Device | How the Attack is Defeated |
|---|--------------------------|--|
| Computing power and a virtually unlimited amount of time. | DS1963S, DS1961S, DS2432 | Using unique secrets (Unique Authentication Secret) in each device mitigates the severity of the problem. This is accomplished by using the devices unique ROM ID as part of the secret generation. If the secret is discovered the system does not have a 'class break'. Only the one device with that particular ROM ID can be emulated. Putting that ROM ID on a black-list will prevent its use forcing the attacker to do another brute force attack on a different device. |

Microprobe Physical Attack (Secret)

The physical attack is an attempt to probe the internal silicon chip to read the *Unique Authorization Secret*. This is very difficult to do, more so with the NV RAM device (DS1963S) than the EEPROM devices. The iButton versions of the 1-Wire SHA-1 devices also add a layer of protection from a probe attack by virtue of their package. The can enclosure is stainless steel with the silicon globbed over. Losing contact with the battery in the DS1963S even for a moment will erase the onboard secrets. As with the brute force attack, if the secret is discovered it could be used in an emulator to make what appears to be a valid device in the service. This emulated device could then be used to replay previous data such as eCash. The conditions and partial solution is the same for this attack as the brute force secret attack above.

Host/SCU Attack (Secret)

Take the 1-Wire SCU that does the authentication and probe it for the *Master Authentication Secret* and *Master*

| Conditions | Device | How the Attack is Defeated |
|---------------------------------------|------------------------|---|
| Must have physical access to the SCU. | none | Keep the SCU secure. For example, use a secure microprocessor like the DS5002 that does not allow the firmware to be read out. |
| Must have physical access to the SCU. | DS1963S as coprocessor | A partial solution is to use the DS1963S as the 'coprocessor' for the device authentication and data validation in the SCU. The secrets are safe from detection. However this coprocessor could be used to create valid <i>Service Data Signatures</i> if the padding data (<i>Initial Signature</i>) is known. |

Competitor Attack

The *Competitor Attack* is an attempt to discredit a system by maliciously and intentionally destroying or disrupting data.

| Conditions | Device | How the Attack is Defeated |
|--|-----------------|---|
| Token must be presented to a malicious reader. | DS1963S | Any data can be overwritten which could invalidate the token. However, writes to a secret can be detected by checking the write-cycle counter associated with it. The device can be reused if it is reloaded with the correct data and secrets. |
| Token must be presented to a malicious reader. | DS1961S, DS2432 | The target device has a protection feature that requires a valid MAC to write data to the user memory area. Since the attacker does not know the secret, writing to the device is not possible. The target device also has a write protect mechanism for the secret. To prevent the attacker from overwriting the secret, this feature must be enabled. |

Conclusion

Data and hardware authentication are the key concepts for various types of service applications including access control and electronic cash. A portable token that can create a message authentication code (MAC) based on a random challenge using a cryptographically good hash is ideal for these applications. The Dallas Semiconductor 1-Wire SHA-1 devices provide an economical and rugged solution.

1-Wire is a registered trademark of Dallas Semiconductor Corp.
iButton is a registered trademark of Dallas Semiconductor Corp.

Dallas Semiconductor is a wholly owned subsidiary of Maxim Integrated Products, Inc.

Application Note 1098: <http://www.maxim-ic.com/an1098>

More Information

For technical questions and support: <http://www.maxim-ic.com/support>

For samples: <http://www.maxim-ic.com/samples>

Other questions and comments: <http://www.maxim-ic.com/contact>

Related Parts

DS1961S: [QuickView](#)

DS1963S: [QuickView](#)

DS2432: [QuickView](#) -- [Abridged Data Sheet](#)

AN1098, AN 1098, APP1098, Appnote1098, Appnote 1098

Copyright © by Maxim Integrated Products

Additional legal notices: <http://www.maxim-ic.com/legal>