nToken Security Policy

Version: **2.1.2**

Date: **6 June 2008**

Patents

UK Patent GB9714757.3. Corresponding patents/applications in USA, Canada, South Africa, Japan and International Patent Application PCT/GB98/00142.

Contents

The nToken is a FIPS 140-2 level 2 module with level 3 physical security. It is designed to protect a single DSA key used to identify a host to an nCipher NetHSM.

The only purpose of the nToken is to sign a message containing a nonce NetHSM to prove to the NetHSM that the session was instigated by a client running on the host.

Unit ID	Model Number	RTC NVRAM	SEE	Potting	EMC	Crypto Accelerator	Overall level
nToken	nC2023P-000	No	No	Yes	A	None	2

The module runs firmware provided by nCipher. There is the facility for the administrator to upgrade this firmware. In order to determine that the module is running the correct version of firmware they should use the Show Status service which reports the version of firmware currently loaded. The validated firmware versions is 2.33.60.

All modules are now supplied at build standard "N" to indicate that they meet the latest EU regulations regarding ROHS.

Provided that the nToken is only used with the FIPS approved firmware, it can only be operated in its FIPS approved mode of operation. It is possible to load new firmware. The administrator should ensure that any new firmware is FIPS validated before they load it into the module.

The nToken connects to the host computer via a PCI bus. The nToken must be accessed by a custom written application.

The nToken can be connected to a computer running one of the following operating systems:

- Windows 2000 and 2003
- Solaris
- HP-UX
- AIX
- Linux x86
- FreeBSD x86

Windows XP Professional SP2 and a Debian Linux distribution with kernel version 2.2.26 were used to test the module for this validation. .

Initializing the nToken

When the module is initialized it generates a random AES key for use as a module key. This key is stored in the modules EEPROM and is never revealed. This step is usually performed in the nCipher factory.

In order to enrol an nToken, the administrator runs the **nTokenEnrol** utility on a host computer, that is outside the security boundary.

The utility performs the following steps:

1. generates a DSA key pair (*Generate DSA key* service)
2. wraps the private half as a key blob protected by the module key (*Wrap key* service) and exports this blob to the host's hard disk.
3. exports a certificate containing the public key and the nToken's Electronic Serial number to the nCipher NetHSM. (*Export* service)
4. displays the SHA-1 hash of the DSA public key on the host computer's display, to enble the administrator to verify that they are enrolling the correct nToken. (*Hash* service)

The utility then sends this data to the NetHSM that will rely on the nToken. Once administrator confirms that the hash shown on the NetHSM's display is the same as that disaplyed at step 4, the NetHSMadds the nToken public key and serial number and the identity of the host in which it is installed to the it's configuration file.

Using the nToken

Once the nToken is enrolled - whenever the nCipher server is started it loads the key blob created when the module was initialized obtaining a key ID for the signing key.

The nToken is used when a operator wishes to open a connection from a host application to a NetHSM via the nCipher server. When the operator attempts to open such a connection, the nCipher server obtains a nonce from the NetHSM and has the nToken sign a message containing this nonce to confirm the identity of the computer the application is running on. The nCipher server sends this message to the NetHSM. The NetHSM verifies the signature and can then determines whether the host is authorized.

Although the nToken uses the same firmware image as nShield modules, nToken modules are factory configured so that they can only perform a limited subset of operations. See nCipher Master Feature Enable Key for more details.

The nCipher server uses the Show Status service to determine which attached modules are nTokens and which are nShields. If the operator requests a service that the nToken cannot perfrom, the server ensures the command is routed to an nShield and not an nToken.

Upgrading Firmware

Although nCipher do not expect that there will ever need to update the firmware in an nToken, nCipher provide a utility to perform this.

Verifying firmware

The administrator or operator can use the **fwcheck** utility to check that the nToken has been programmed with valid firmware. This utility takes a copy of the signed and encrypted firmware file and performs a zero knowledge check to ensure the firmware loaded on the module is the same as the copy on disk.

Keys

For each type of key used by the nToken, the following section describes the access that a operator has to the keys. The nToken refers to keys by their handle, an arbitrary number, or by its SHA-1 hash.

Module key

The Module key is an AES key used to protect other keys. This key is generated by the module key when it is initialized. If the module is re-initialized, the AES key is cleared and a new key must be generated before the module can be started in operational mode. nTokens are initialized by nCipher before they are supplied to the customer and do not normally need re-initializing. The module key is guaranteed never to have been known outside this module.

Authentication Key

When the nToken is enrolled it generates a DSA key pair used for signature generation. The public half of this key is exported in plain text and has to be transferred to the NetHSM.

The private half is encrypted under the module key and exported as an nCipher format key blob which is stored on the host computer's hard disk.

In order to use the signature key, the operator loads the key blob. When the key is loaded in the module it is stored in RAM and identified by a random identifier that is specific to the operator and session. The operator can then have the module sign messages by providing this identifier.

Firmware Integrity Key

All firmware is signed using a DSA key pair. A module only loads new firmware if the signature decrypts and verifies correctly.

The private half of this key is stored at nCipher.

The public half is included in all firmware. The firmware is stored in flash memory when the module is switched off, this is copied to RAM as part of the start up procedure.

Firmware Confidentiality Key

All firmware is encrypted using Triple DES to prevent casual decompilation.

The encryption key is stored at nCipher's offices and is in the firmware. The firmware is stored in flash memory when the module is switched off, this is copied to RAM as part of the start up procedure.

nCipher Master Feature Enable Key

The nToken uses the same firmware and basically the same hardware as the nCipher nForce and nShield modules. However most functionality is disabled using nCipher's Feature Enable Mechanism. This controls features using certificates signed by the nCipher Master Feature Enable Key. Presenting such a certificate causes the module to set the appropriate bit in the FRAM.

The nCipher Master Feature Enable Key is a DSA key pair, The private half of this key pair is stored at nCipher's offices. The public half of the key pair is included in the firmware. The firmware is stored in flash memory when the module is switched off, this is copied to RAM as part of the start up procedure.

Roles

The module has two roles which implicitly assumes all services:

Administrator

> The Administrator Role is an implicitly assumed role that is procedurally assumed as part of the setup and initialization procedures of the nToken. The Administrator generates or replaces the secret AES Module Key. This step is performed by nCipher before the module is shipped, but can be repeated by the customer as part of the setup and initialisation process.
>
> Generates the signing key. To assume the administrator role, the administrator must run the nTokenEnrol utility. Running this utility implicitly selects the administrator role. This destroys all stored keys and creates a new key blob that is used to authenticate the user role and exports the public key that can be used to verify messages.
>
> After the module has been configured, and is operating in its FIPS mode, there is no further requirement for the administrator role to interact with the module and all further services interaction is are performed in the implicitly assumed Administrator / User Role.

User

> The user role uses the key to sign messages.
>
> To assume the user role, the operator must present the key blob generated by the administrator. If this blob load correctly the module returns an **ObjectID** for the signing key. To sign a message the operator sends the **Sign** command with the **KeyID** and message. The module returns the signed message.

Services

The following services are provided by the nToken.

Service	Key type	Role	Description
Check Firmware	DSA and Triple DES	Admin / User	Performs a zero knowledge check of the firmware image.
Generate Key	AES	Admin	The module automatically generates a new AES key used as the module key. This key is never exported. This is performed as part of the setup and initialisation procedure.
Generate Key	DSA	Admin	The module generates a DSA key pair, used to sign messages. This is performed as part of the setup and initialisation procedure.
Wrap Key	AES + HMAC	Admin	The private DSA key is wrapped as an nCipher key blob. Wrapping uses AES CBC mode for encryption and SHA-1 HMAC for integrity. This is performed as part of the setup and initialisation procedure.
Export	DSA public	Admin	The public half of the DSA key pair is exported in plain text. This is performed as part of the setup and initialisation procedure.
Hash	SHA-1	Admin	The module exports the SHA-1 hash of the DSA key to enable the key to be identified in other services. The hash can be calculated from either the private or public half of the key. This feature can be used to ensure the correct parts of a key pair are being used. This is performed as part of the setup and initialisation procedure.
Unwrap Key	AES + HMAC	User	The operator presents the wrapped private key at the start of a session. If the key unwraps and the MAC verifies, the operator is authorized.
Sign	DSA private	User	Once the operator has loaded the private key they can use it to sign messages.

Service	Key type	Role	Description
Zero	DSA private	Admin / User	Clears all unwrapped keys. The Module key can be zeroed by repeating the Key Generation Service.
Show Status		Admin / User	Displays status information.
Loads Firmware	DSA and Triple DES	Admin	Replaces a firmware image with new firmware. The firmware is signed and encrypted with keys held at nCipher. The nToken must be re-initialized after loading firmware.

Rules

Object re-use

All objects stored in the module are referred to by a handle. The module's memory management functions ensure that a specific memory location can only be allocated to a single handle. The handle also identifies the object type, and all of the modules enforce strict type checking. When a handle is released the memory allocated to it is actively zeroed.

Error conditions

If the nToken cannot complete a command due to a temporary condition, the module returns a command block with no data and with the status word set to the appropriate value. The operator can resubmit the command at a later time. The server program can record a log of all such failures.

If the nToken encounters an unrecoverable error it enters the error state. This is indicated by the status LED flashing in the Morse pattern SOS. As soon as the unit enters the error state all processors stop processing commands and no further replies are returned. In the error state the unit does not respond to commands. The unit must be reset.

Security Boundary

The physical security boundary is the plastic jig that contains the potting on both sides of the PCB.

The following items are excluded from FIPS 140-2 validation as they are not security relevant.

- status LED
- PCI connector
- mode links
- clear button

Status information

The module has an LED that indicates the overall state of the module.

The module also returns a status message in the reply to every command. This indicates the status of that command.

Physical security

All security critical components of the nToken are covered by potting.

The module has a clear button. Pressing this button put the module into the self-test state, clearing all stored key objects and running all self-tests. The long term security critical parameters, module keys, module signing key can be cleared by re-initializing the nToken, as described above.

Checking the module

To ensure physical security, the administrator should regularly examine the metal cover over the epoxy resin security coating for obvious signs of damage.

Strength of functions

Attacking ObjectIDs

A operator is authenticated by a key blob, which is encrypted by a 256-bit AES key with integrity provided by a 160-bit HMAC key. It is therefore almost impossible to spoof this authentication.

An attacker may however get the module to sign a message with the stored key by guessing the client and key key identifiers.

Connections are identified by a **ClientID**, a random 32 bit number.

Objects are identified by an **ObjectID** again this is a random 32 bit number.

In order to randomly gain access to a key loaded by another operator, the attacker would need to guess two random 32 bit numbers. The module can process about 2^{16} commands per minute - therefore the chance of succeeding within a minute is $2^{16} / 2^{64} = 2^{-48}$ which is significantly less that the required chance of one in a million (~$2^{-20)}$.

Self Tests

When power is applied to the module it enters the self test state. The module also enters the self test state whenever the unit is reset, by pressing the clear button.

In the self test state the module clears the main RAM, thus ensuring any loaded keys or authorization information is removed and then performs its self test sequence, which includes:

- An operational test on hardware components
- An integrity check on the firmware, verification of a SHA-1 hash.
- A statistical check on the random number generator
- Known answer and pair-wise consistency checks on all algorithms and pRNG
- Verification of a MAC on EEPROM contents to ensure it is correctly initialized.

This sequence takes a few seconds after which the module enters the Pre-Maintenance, Pre-Initialisation, Uninitialised or Operational state; depending on the position of the mode switch and the validity of the the EEPROM contents.

While it is powered on, the module continuously monitors the temperature recorded by its internal temperature sensor. If the temperature is outside the operational range it enters the error state.

The module also continuously monitors the hardware RNG and the approved SHA-1 based pRNG. If either fail it enters the error state.

When firmware is updated, the module verifies a DSA signature on the new firmware image before it updates the images stored in flash.

In the error state, the module's LED repeatedly flashes the Morse pattern SOS, followed by a status code indicating the error. All other inputs and outputs are disabled.

Supported Algorithms

Algorithms marked with an asterisk (*) are not enabled when the module is configured as a nToken.

FIPS approved algorithms:

- AES

  Certificate #599, ECB, CBC and CMAC modes

  GCM* Mode Vendor Affirmed
- Triple DES

  Certificate #570

  Double and triple length keys

  Approved for Federal Government Use - Modes are CBC and ECB
- Triple DES MAC*

  Triple-DES Certificate #570 vendor affirmed
- DSA

  Certificate #233
- ECDSA*

  Certificate #64
- RSA*

  Certificate #274
- SHA-1, SHA-224, SHA-256, SHA-384 and SHA-512

  Certificate #648
- HMAC SHA-1, HMAC SHA-224, HMAC SHA-256, HMAC SHA-384 and HMAC SHA-512

  HMAC certificate #309

- Random Number Generator

  (FIPS 186-2 Change Notice 1 SHA-1 and FIPS 186-2 RNG General Purpose RNG) Certificate #340

Non-FIPS approved algorithms

These algorithms must not be used by a module operating in FIPS approved mode.

Symmetric

- Aria*
- Arc Four (compatible with RC4)*
- Camellia*
- CAST 6 (RFC2612)*
- DES*
- SEED (Korean Data Encryption Standard)*

Asymmetric

- Diffie-Hellman (key agreement, key establishment methodology provides between 80 and 256 bits of encryption strength)*
- Elliptic Curve Diffie-Hellman (key agreement, key establishment methodology provides 192 bits of encryption strength);*
- El Gamal *
- RSA* (key wrapping, key establishment methodology provides between 80 and 256 bits of encryption strength)
- KCDSA*
- EC-MQV* (key establishment methodology provides between 80 and 256 bits of encryption strength)*

Hash

- HAS-160*
- MD5*

- RIPEMD 160*
- Tiger*

MAC

- HMAC (MD5, RIPEMD160, Tiger)*

Other

- SSL/TLS master key derivation*
- PKCS #8 key wrapping.*

Note *TLS key derivation is approved for use by FIPS 140-2 validated modules - though there is as yet no validation test. MD5 may be used within TLS key derivation.*

nCipher addresses

nCipher Corporation Ltd.

Cambridge, UK

Jupiter House
Station Road
Cambridge
CB1 2JD
UK

Tel:	+44 (0) 1223 723600
Fax:	+44 (0) 1223 723601
E-mail:	sales@ncipher.com

nCipher Inc.

Boston Metro Region, USA

92 Montvale Avenue, Suite 4500
Stoneham, MA 02180
USA

Tel:	800-NCIPHER 800-6247437 +1 (781) 994 4000
Fax:	+1 (781) 994 4001
E-mail:	sales@us.ncipher.com

Internet addresses

Web Site:	http://www.ncipher.com/
Support:	http://www.ncipher.com/support/
Online Documentation:	http://active.ncipher.com/documentation/

Note *nCipher also maintain international sales offices. Please contact the UK, or the US, head office for details of your nearest nCipher representative.*