FIPS 140-2 Security Policy

for

Centrify Corp.

Centrify Cryptographic Module

Software Module Version: 1.0

Document Version Number: 1.1

1. Module Description

Centrify Cryptographic Module provides cryptographic functionality to Centrify software applications. For the purposes of FIPS 140-2 the module is classified as a software module. This module includes the following component:

- fipscanister.o

The module is installed into a GPC. Since the GPC where the module is installed is a multi-chip standalone device, the module is qualified as a multi-chip standalone module. The GPC provides the physical cryptographic boundary.

The main purpose of the module is to provide cryptographic functions.

FIPS 140-2 conformance testing of the module was performed at Security Level 1. The following configurations were tested by the lab:

Software Component Version	**Operating Systems**
1.0	Mac OS X 10.6.5 RedHat Enterprise Linux ES v5

The following table summarizes FIPS 140-2 compliance claims

Security Requirements Section	Security Level
Cryptographic Module Specification	1
Module Ports and Interfaces	1
Roles, Services and Authentication	1
Finite State Model	1
Physical Security	N/A
Operational Environment	1
Cryptographic Key Management	1
EMI/EMC	1
Self-Tests	1
Design Assurance	1
Mitigation of other attacks	N/A

2. Cryptographic Boundary

The logical cryptographic boundary of the module includes the software binary (software component). The physical cryptographic boundary is the boundary of the GPC where the module is installed.

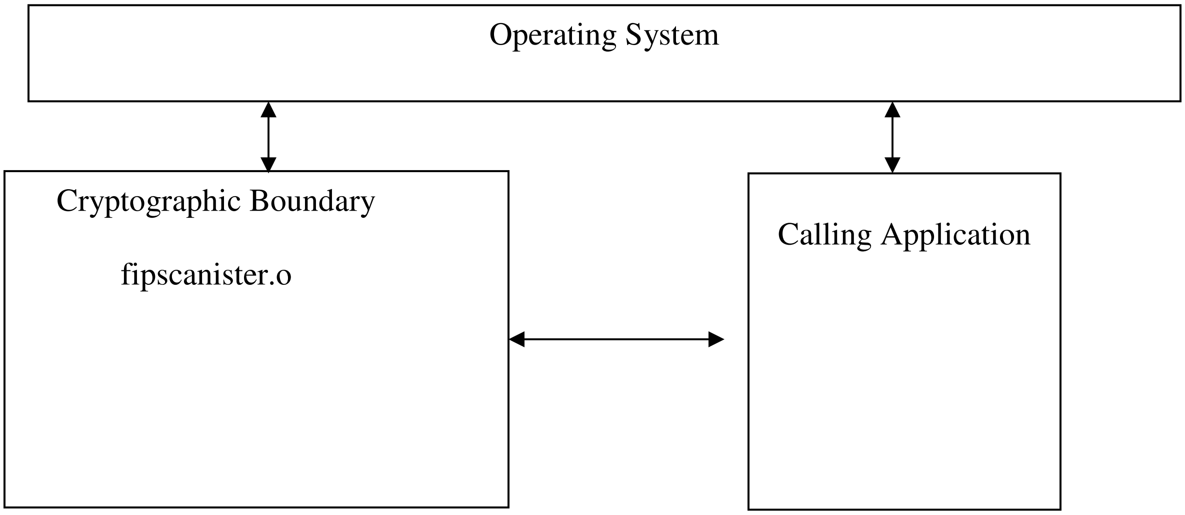
The module includes the following logical interfaces:

- Control Input Interface: software API commands and command parameters used to control and configure module operation.
- Status Output Interface: return values from software API commands used to obtain information on the status of the module.
- Data Input Interface: data inputs to the software API commads.
- Data Output Interface: data outputs of the software API commands.

All module interfaces, inputs and outputs are provided by the software component.

The block diagram for the module is provided below.

Figure 1. Block Diagram



3. Roles and Services

The module provides the following roles:

1. User.
2. Crypto Officer.

The Crypto Officer installs the module.
The User employs the cryptographic services provided by the module.

The module provides the following services to the User and Crypto Officer.

Service	Role	Access to Cryptographic Keys and CSPs R- read or use W – write or generate, Z – zeroize N/A – no CSPs are accessed by this service
Run-self tests	User	N/A
Get status of the module	User	N/A
Encrypt/Decrypt	User	R, W (sets encryption key)
Sign/Verify	User	R, W (sets digital signature key)
Perform RSA key wrapping	User	R, W (sets RSA wrapping key)
Calculate Secure Hash	User	N/A
Generate Random Numbers	User	R, W (sets DRBG seed)
Generate Asymmetric Keys	User	R, W (sets DRBG seed) W (sets asymmetric keys)
Calculate HMAC	User	R, W(sets HMAC key)
Zeroize	User	Z (zeroizes keys)
Installation of the module	Crypto Officer	N/A

4. Security Functions

The table below lists approved cryptographic algorithms employed by the module

Algorithm	Certificate #
AES	1554
Triple-DES	1018
HMAC	904
SHS	1375
RSA	755
DSA	480
DRBG	69

The module implements the following non-Approved but allowed cryptographic algorithms: Diffie-Hellman; RSA key wrapping.

Note: to stay in the Approved mode and to comply with Special Publication 800-90, the calling application shall not generate more than 2^19 bits per request to the DRBG generation function. The "Get status of the module" service returns information as to the successful/unsuccessful execution of self-tests. Since the module is validated at Level 1, there are no requirements for indication of the mode of operation.

Note: the module does not implement a bypass capability.

Note: the module supports RSA and Diffie-Hellman keys from 1024 to 10000 bits of length, which corresponds to 80 to 219 bits of effective security strength.

5. Key Management

The following cryptographic keys are supported by the module

Name and Type	Generation or establishment	Usage
AES keys	Set by the application using the module's software API	Encryption/decryption
TDES keys	Set by the application using the module's software API	Encryption/decryption
RSA key pairs	Set by the application using the module's software API	Sign/Verify Key wrapping
DSA key pairs	Set by the application using the module's software API	Sign/Verify

HMAC keys	Set by the application using the module’s software API	Calculate HMAC
HMAC SHA-1 integrity key	Pre-set in the module binary	Used to check integrity of the module at power-on
DRBG Seed	Generated by the module	Used to seed the DRBG

All keys are stored inside the module in plaintext.

To zeroize the keys inside the logical cryptographic boundary one shall power down the computer, which will also power down the module. Since all keys stored in the module are stored in the volatile memory, powering down the module destroys the keys.

Note: the phrase “a key is set by the application” means that the key is passed to the module by the application utilizing an API function call.

Note: the module stores the keys in the volatile-memory and does not utilize non-volatile storage to store keys.

6. Self Tests.

The module runs a set of self-tests on power-up. If one of the self-tests fails, the module transitions into an error state where all data output and cryptographic operations are disabled. The self-test success or error message is output into the log file.

The module runs self-tests for the following algorithms

Power-up tests

- integrity check using HMAC-SHA-1
- AES KAT
- TDES KAT
- SHS KAT
- HMAC KAT
- RSA sign/verify
- DSA sign/verify
- DRBG KAT

2. Conditional tests

- Continuous DRBG Test
- DSA sign/verify test on key generation
- RSA sign/verify test on key generation
- RSA encrypt/decrypt test on key generation

7. Crypto Officer Guidance.

The approved mode of operation is enabled by the Crypto Officer role.

The module is installed as follows:

1. Copy fipscanister.o to the target GPC.

2. Link fipscanister.o with the application code by running the corresponding make script.

3. During application execution, call FIPS_mode_set() function to set the module in the FIPS mode.

Note: the module does not provide a capability to switch between Approved and non-Approved modes of operation.

In order to maintain security of the module operation the Crypto Officer shall verify that the module is installed and executed in a physically secure location.