# Automatic Encryption

**Automatic Encryption RAISE SATA 6Gb/s Interface**

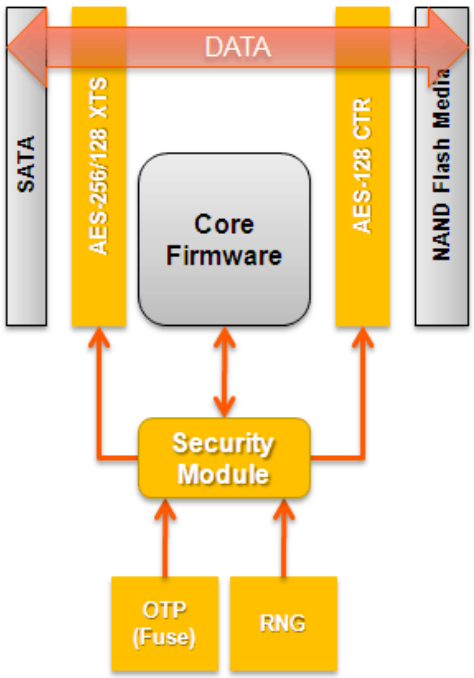
**Latest SandForce SSD Processors Double Encrypt Data**

Security is a hot topic for most industries and storage is not immune. Nearly all SSDs today store data directly to the flash memory without performing any encryption. These systems support password protection techniques that prevent a would-be thief from accessing the data on the SSD. However, the flash memory of an SSD can be accessed directly with a special “clip” in the hands of a skilled technician, unlike a HDD with rotating media. Only if the host spends time encrypting the data will it be secure, but this will consume valuable resources and slow the path to the storage.

The SandForce SF-2000 SSD Processor solves this problem by embedding two encryption engines using AES-256 & AES-128 to protect the information it stores on the flash and prevent any unauthorized access. This is done at the drive level without any host dependency and without slowing down the data transfer.

| Data Security Features | SF-1000 | SF-2000 |
|---|---|---|
| **Hardware encryption of data stored in flash memory** | | |
| • AES-128 engine in CTR mode | Yes | Yes |
| • AES-256 engine in XTS mode | - | Yes |
| **Implemented multiple standards** | | |
| • TCG Enterprise | - | Optional<br>Enterprise only |
| • TCG Opal | - | Optional<br>Client only |
| **ATA Secure Erase** | Yes | Yes |
| **Military Erase protocol** | Yes<br>Enterprise only | Yes<br>Enterprise only |
| **Read-only Operation** | Yes<br>Enterprise only | Yes<br>Enterprise only |
| **Optional disk password (during boot)** | Yes | Yes |

SF-2000 Security Block Diagram



**Additional features**

- Fuse-based OTP (one time programming memory) for unique master key
- Hardware non-deterministic random number generator
- FIPS-197 certification of AES engine (SF-1000)

The total security strength of the Encryption Scheme used in the SF-2000 SSD Processors was calculated by Aspect Labs to be equivalent of **300 Bit** encryption engine. **See the full report.**

Aspect Labs,
3080 Olcott Street, Suite 110-A
Santa Clara, California, USA 95054
Phone: 1-888-347-7140
Fax: 1-408-492-1419
Website: www.aspectlabs.com

APRIL 20, 2011

SECURITY STRENGTH DETERMINATION AND REPORT

SandForce, Inc. ("Vendor")

Dear Vendor,

In April, 2011, Aspect Labs, a division of BKP Security, Inc. ("Laboratory") was asked to conduct the total strength analysis of data protection by two subsequent AES engines, XTS 256 and CTR 128 ("Encryption Scheme"), operating independent keys with the assumption that the data path is hardware protected. The goal of this analysis was to estimate the security strength of the encryption. We have performed an independent detailed analysis of Encryption Scheme based on the current state of the art in the cryptographic science.

*The total security strength of the Encryption Scheme was calculated to be **300 Bits.***

A detailed mathematical report for the crypto analysis is attached to this Letter.

With best regards,

Stan Kladko, Ph.D., Director, Aspect Labs FIPS 140-2 and Common Criteria Lab

DETAILED SECURITY STRENGTH ANALYSIS.

1. General.

Encryption Scheme utilizes two AES encryption engines, XTS 256 and CTR 128 that use independently generated cryptographic keys. SandForce uses a strong non-deterministic key generator for each key. The independent security strength of the XTS 256 engine is then 256 bits and the security strength of the CTR 128 engine is 128 bits. SandForce does satisfy the requirements of uniqueness for the counters present in both engines.

2. Double encryption.

As the encryption engines apply encryption to data subsequently, the data is, therefore, independently encrypted twice. The purpose of this analysis is to calculate the encryption strength of the corresponding double encryption. The effective strength of encryption when several ciphers are used in sequence has been subject to intensive research in the cryptographic science, starting from the 1970s. In the remainder of this document, we review the corresponding approaches and calculate the effective encryption strength, considering a known-plaintext attack, where the attacker possesses a ciphertext-plaintext pair and needs to recover the key.

3. Original Meet-In-The-Middle approach.

Meet-In-The-Middle approach was first introduced by Diffie and Hellman in 1977, initially for the case, where both ciphers in the double-encryption have the same key strength (see *W. Diffie and M. E. Hellman (June 1977), "Exhaustive Cryptanalysis of the NBS Data Encryption Standard," Computer 10 (6): 74–84*). The original Diffie-Hellman approach does not directly apply to the SandForce Encryption Scheme, as in the SandForce case the two ciphers used for double encryption have different encryption strengths.

In the following we extend the Diffie-Hellman approach for the case when the two ciphers have different strengths.

Lets us denote the SandForce AES engine XTS 256 as A and the SandForce AES engine CTR 128 as B, with the corresponding keys K1 and K2, with lengths L1 and L2.

Then we have:

C = A(K1, B(K2, P))

Where C is the plaintext and P is the cipher text.

Let M to be the intermediate value:

M = B(K2, P)

Then:

C = A(K1, M)

And

M = A^{-1} (K1, C)

Then by iterating over the space of all keys K1, one can build a table mapping M to K1, which will take 2^{L1} operations. One will then sort this table, which will take of the order of L1 2 ^ {L1} operations for the mergesort algorithm. The memory requirement to hold the table is of the order of 2^{L1}.

Then, having the sorted table, one can iterate over all keys K2, which will take approximately 2^{L2} operations, and for each resulting M = B(K2, P) look up the table, to see if the corresponding M is there. This lookup will take approximately L1 operations for an ordered table. If M is in the table, it will give K1, so we recover the pair K2, K1. This second step will take approximately 2^{L2} operations.

As we see, one needs the total of about 2^{L1} memory storage locations and Max(2^{L1}, 2^{L2}) operations.

We can change the above considerations replacing decryption with encryption using:

P = B^{-1}(K2, A^{-1}(K1, C))

this will give us an algorithm requiring 2^{L2} memory storage locations and Max(2^{L1}, 2^{L2}) operations.

As in our case L1 > L2, it is beneficial to choose the second algorithm, that will require 2^128 locations and 2^256 operations. As allocating 2^128 memory locations is infeasible for the modern storage technology , the original Meet-In-The-Middle approach of Diffie-and-Hellman does not apply to the case of SandForce.

3. Modified Meet-In-The-Middle approach.

A modified Diffie-Hellman approach which requires less memory was considered by many authors. See for example *S. Even and O. Goldreich, On the power of cascade ciphers see both ACM Transactions on Computer Systems, vol. 3, pp. 108–116, 1985,* and *Hamid R. Amirazizi, Martin E. Hellman, "Time-memory-processor trade-offs," IEEE Transactions on Information Theory 34(3): 505-512 (1988).*

Essentially, in the modified approach the space of all keys K2 is split into R subspaces by fixing the first G bits of the key K2, so R = 2 ^ G. The number of keys in a given subspace is 2^{L2 – G}

One iterates over the subspaces. For a given subspace a table is generated and one proceeds as in the original approach, having the running time of Max{2^{L1 - G}, 2^{L2}} = 2^{L2} and the size of the table 2^{L2 - G}. After a new subspace is selected, the old table is replaced with a new one.

As one iterates over all subspaces, the total running time is 2^{L1+G}, and the memory requirement is MEM = 2^{L2 - G}.

For modern technology, the realistic maximum of MEM = L2 - G = 40 is feasible, which corresponds to a terabyte of memory. For L2 = 128 this corresponds to G = 88. Then one has the running time of the algorithm of the order of 2^{256 + 88} = 2^344.

*Therefore, the strength of the algorithm as provided by the modified Meet-In-The-Middle approach is* ***344 bits.***

4. Pollard Rho Method.

An improved approach based on Pollard Rho method was introduced by Oorschot and Wiener. (See *"Improving Implementable Meet-in-the-Middle Attacks by Orders of Magnitude", CRYPTO, Vol. 1109, Springer (1996), p. 229-236, and Journal of Cryptology, 1999, vol. 12, 1, pp. 1-28*).

The attack is based on the parallel collision search, which is, in turn, based on Pollard's rho method, as described, for instance, in *D.E. Denning, Cryptography and Data Security, Addison-Wesley, 1982*.

While the cited papers use advanced mathematics, detailed description of which is outside of the scope of this report, application of the method above to the case of SandForce amounts to applying formula (8) on page 17 of *"Improving Implementable Meet-in-the-Middle Attacks"*, which can be approximately specified in the case of SandForce as the running time of the algorithm being equal to

T ~ (L1 Sqrt(L2/MEM))

For L1 = 2^{256}, L2 = 2^{128} and MEM = 2^{40} this gives

T ~ 2^{300}

*Therefore, the approximate strength of the SandForce technology using Pollard Rho Method is* ***300 bits.***

This again, assumes that the attacker has access to about 1 TByte of relatively fast memory to hold the tables (such as flash memory).

5. Further Improvements.

More than 15 years have passed since the original paper of Oorschot and Wiener. Since that time there was no significant progress in the field of attacks on double encryption. Therefore, we conclude that the estimate of **300 bits**, as given by the Oorschot-Wiener formula, is probably going to hold for an extensive period of time. This concludes our analysis.