

CIPHER DEVICES AND CIPHER METHODS

Cross-reference to Related Applications

[0001] The present application claims the benefit of the United States provisional patent application No. 61/715,324 filed on 18 October 2012, the entire contents of which are incorporated herein by reference for all purposes.

Technical Field

[0002] Embodiments relate generally to cipher devices and cipher methods.

Background

[0003] More and more data are being stored in electronic form. The number of data breaches and the associated risks have increased tremendously. Thus, data encryption and key management are the essential solutions in securing data. However, a process of encryption may seriously affect performance. Thus, there may be a need for efficient encryption.

Summary

[0004] According to various embodiments, a cipher device may be provided. The cipher device may include: a data collector configured to receive data to be processed; a cipher circuit configured to perform at least one of encryption or decryption of the data; and a control circuit configured to determine whether the amount of data received by the

data collector fulfills a pre-determined criterion, and configured to instruct the cipher circuit to perform the at least one of encryption or decryption of the data if the amount of data received by the data collector fulfills the pre-determined criterion.

[0005] According to various embodiments, a cipher method may be provided. The cipher method may include: receiving data to be processed; performing at least one of encryption or decryption of the data; determining whether the amount of data received by the data collector fulfills a pre-determined criterion; and instructing the at least one of encryption or decryption of the data if the amount of data received fulfills the pre-determined criterion.

Brief Description of the Drawings

[0006] In the drawings, like reference characters generally refer to the same parts throughout the different views. The drawings are not necessarily to scale, emphasis instead generally being placed upon illustrating the principles of the invention. In the following description, various embodiments are described with reference to the following drawings, in which:

FIG. 1A shows a cipher device according to various embodiments;

FIG. 1B shows a cipher method according to various embodiments;

FIG. 2 shows a computing system according to various embodiments;

FIG. 3 shows the building blocks of the proposed hardware cipher device driver design to accelerate Data Protection for Large Scaled Shared Storage System according to various embodiments;

FIG. 4 shows the data flow according to various embodiments;

FIG. 5 shows the workflow of the data accumulator process combined with pipelining process according to various embodiments; and

FIG. 6 illustrates how the high throughput with pipelined data cipher can be achieved according to various embodiments.

Description

[0007] Embodiments described below in context of the devices are analogously valid for the respective methods, and vice versa. Furthermore, it will be understood that the embodiments described below may be combined, for example, a part of one embodiment may be combined with a part of another embodiment.

[0008] In this context, the cipher device as described in this description may include a memory which is for example used in the processing carried out in the cipher device. A memory used in the embodiments may be a volatile memory, for example a DRAM (Dynamic Random Access Memory) or a non-volatile memory, for example a PROM (Programmable Read Only Memory), an EPROM (Erasable PROM), EEPROM (Electrically Erasable PROM), or a flash memory, e.g., a floating gate memory, a charge trapping memory, an MRAM (Magnetoresistive Random Access Memory) or a PCRAM (Phase Change Random Access Memory).

[0009] In an embodiment, a "circuit" may be understood as any kind of a logic implementing entity, which may be special purpose circuitry or a processor executing software stored in a memory, firmware, or any combination thereof. Thus, in an embodiment, a "circuit" may be a hard-wired logic circuit or a programmable logic circuit such as a programmable processor, e.g. a microprocessor (e.g. a Complex

Instruction Set Computer (CISC) processor or a Reduced Instruction Set Computer (RISC) processor). A "circuit" may also be a processor executing software, e.g. any kind of computer program, e.g. a computer program using a virtual machine code such as e.g. Java. Any other kind of implementation of the respective functions which will be described in more detail below may also be understood as a "circuit" in accordance with an alternative embodiment.

[0010] More and more data are being stored in electronic form. The number of data breaches and the associated risks have increased tremendously. Thus, data encryption and key management are the essential solutions in securing data. However, a process of encryption may seriously affect performance. Thus, there may be a need for efficient encryption.

[0011] One of the measures to protect information on a computer system's hard disk drive is by encrypting the information through Full Disk Encryption solutions. It has become very important especially for enterprises with central information storage systems to employ FDE technologies on their data centers.

[0012] Meanwhile, Linux has been on an uptrend in cutting its share on the server market from Windows and UNIX servers. It has been well accepted because of its stability, security, cost, and flexibility.

[0013] According to various embodiments, devices and methods may be provided which improve the performance aspect of one of Linux' security subsystems called the Crypto API (Application programming interface), which may be used by applications like FDE (Full Disk Encryption) that need different ciphers for encryption/decryption of data. It is to be noted that there are other applications that can use this security subsystem (like

IPSec). However, various embodiments may be also be applied to applications requiring huge amounts of data every time when doing encryption/decryption.

[0014] The Crypto API of Linux may include three logical layers called (1) Transform API, (2) Transform Ops, and (3) Algorithm API. The Transform API may define the methods exported to the applications by the Crypto API. One of the most popular software FDE solutions that use this API is the dm-crypt/LUKS. The Transform Ops may be the glue logic that maps the type of cipher algorithm that the application needs to the actual cipher implementation. The Algorithm API allows registration of any cipher algorithm implementation (using software or hardware accelerators) to the Crypto API.

[0015] In addition, the CryptoAPI of Linux may support three types of Algorithm API's that device drivers can use to get/send data from/to the higher layers, i.e. (1) Stream Cipher, (2) Synchronous Block Cipher, and (3) Asynchronous Block Cipher. In Stream Cipher, the data size that the device driver gets is equivalent to the block size that an encryption algorithm supports, e.g. 16 bytes for AES-256 algorithm. In Synchronous Block Cipher, the device driver gets multiple blocks of data every operation. The maximum number of data that this type of cipher gets when FDE operations are running is 512 bytes. It is synchronous because every encrypt or decrypt operations block until processed data are returned. Asynchronous Block Cipher is similar to Synchronous Block Cipher in terms of the number of data passed on every encrypt/decrypt operation. However, the following encrypt or decrypt operations are not blocked, but instead, allows the device driver to manage queues for encrypt/decrypt.

[0016] Software FDE solutions (e.g. TrueCrypt or dm-crypt/LUKS) may normally try to encrypt/decrypt 512 bytes of data at a time. The reason for this is to seamlessly write/read encrypted/decrypted data per sector of a hard disk drive which is normally 512 bytes. This means that the length of data being processed by the cipher software or hardware will only be 512 bytes of data every time. Using hardware ciphers are generally preferred because encryption/decryption routines are generally CPU resource intensive, thus, by using hardware ciphers, the task is offloaded to the hardware and generally, hardware accelerates the encryption/decryption process. However, using hardware ciphers have their own overheads. For example, in order to do direct memory access (DMA) to/from hardware requires that the necessary configuration and operation registers and memory to be set-up by the hardware device driver before the actual data transfers and cipher operations happen. Therefore, hardware ciphers and device drivers should support large enough data per encrypt/decrypt operation in order to minimize the hardware operations overhead.

[0017] In commonly used methods and devices, doing hardware cipher operations with 512-byte data from the FDE solutions is not efficient.

[0018] According to various embodiments, the performance of traditional block cipher drivers may be improved by 4x using the following approaches. According to various embodiments, a data accumulator process may support large data cipher process by hardware, which may minimize the processing overhead on hardware and speed up the performance by 2x. According to various embodiments, a task pipeline may be provided to schedule the data accumulator process and hardware cipher process, which may further improve the performance by a factor of 4.

[0019] According to various embodiments, the combination of a data collector and pipelining design of hardware cipher may improve the performance of FDE solutions in Linux by 4 times compared to traditional Synchronous Block Cipher solutions. On a test bed, it may be found that the raw performance of hardware cipher is about 5.05 microseconds when processing 512 bytes of data (101MBps) and about 16 microseconds when processing 4096 bytes (256MBps). When a pipeline with just 512 bytes of data is incorporated, the performance may increase by 30% (140MBps). Considering that the 2-buffer pipeline may improve the raw performance by around two-fold, the raw performance may be 455MBps. This may be approximately a 4x improvement from the traditional 512 bytes of Synchronous Block Cipher processing.

[0020] According to various embodiments, devices and methods may be provided for accelerating data protection for large scaled shared storage systems.

[0021] According to various embodiments, devices and methods may be provided to protect data for a large scaled shared storage system (for example for a cloud storage). According to various embodiments, devices and methods may be provided to enhance data encryption and decryption performance. According to various embodiments, devices and methods may be provided to minimize hardware overheads during cipher processes.

[0022] FIG. 1A shows a cipher device 100. The cipher device 100 may include a data collector 102 configured to receive data to be processed. The cipher device 100 may further include a cipher circuit 104 configured to perform at least one of encryption or decryption of the data. The cipher device 100 may further include a control circuit 106 configured to determine whether the amount of data received by the data collector 102 fulfills a pre-determined criterion. The control circuit 106 may further be configured to

instruct the cipher circuit 104 to perform the at least one of encryption or decryption of the data if the amount of data received by the data collector 102 fulfills the pre-determined criterion. The data collector 102, the cipher circuit 104, and the control circuit 106 may be coupled with each other, like indicated by lines 108, for example electrically coupled, for example using a line or a cable, and/ or mechanically coupled.

[0023] In other words, the cipher device 100 may collect received data until the amount of received data fulfills a pre-determined criterion, and may then carry out at least one of encryption or decryption of the collected data.

[0024] According to various embodiments, the pre-determined criterion may include or may be whether the amount of data is higher than a pre-determined threshold.

[0025] According to various embodiments, the pre-determined threshold may be larger than the amount of data per sector of a hard disk drive for which the cipher device 100 is to provide its (encryption or decryption) services.

[0026] According to various embodiments, the pre-determined threshold may be a value of at least 4 KB (kilobyte).

[0027] According to various embodiments, the cipher circuit 104 may be configured to perform the at least one of encryption or decryption of the data in a pipeline fashion.

[0028] According to various embodiments, the control circuit 106 may further be configured to instruct a further instance of the cipher circuit 104 to perform the at least one of encryption or decryption of the data if the amount of data received by the data collector 102 since the previous instruction to the cipher circuit 104 fulfills the pre-determined criterion, if the cipher circuit 104 did not finish the previously instructed processing.

[0029] According to various embodiments, the cipher device 100 may be configured for bulk encryption.

[0030] According to various embodiments, the cipher device 100 may be configured for bulk encryption of a cloud storage.

[0031] According to various embodiments, the data collector 102 may be configured to receive the data from a higher layer of a crypto application programming interface.

[0032] According to various embodiments, the cipher circuit 104 may further be configured to provide the at least one of encrypted data or decrypted data to the higher layer.

[0033] FIG. 1B shows a flow diagram 110 illustrating a cipher method according to various embodiments. In 112, data to be processed may be received. In 114, at least one of encryption or decryption of the data may be performed. In 116, it may be determined whether the amount of data received by the data collector fulfills a pre-determined criterion. In 118, the at least one of encryption or decryption of the data may be instructed if the amount of data received fulfills the pre-determined criterion.

[0034] According to various embodiments, the pre-determined criterion may include or may be whether the amount of data is higher than a pre-determined threshold.

[0035] According to various embodiments, the pre-determined threshold may be larger than the amount of data per sector of a hard disk drive for which the cipher device is to provide its services.

[0036] According to various embodiments, the pre-determined threshold may be a value of at least 4 KB.

[0037] According to various embodiments, the cipher method may further include performing the at least one of encryption or decryption of the data in a pipeline fashion.

[0038] According to various embodiments, the cipher method may further include instructing a further instance of performing the at least one of encryption or decryption of the data if the amount of data received since the previous instruction of performing the at least one of encryption or decryption of the data fulfills the pre-determined criterion, if the previously instructed at least one of encryption or decryption of the data is not finished.

[0039] According to various embodiments, the cipher method may perform bulk encryption.

[0040] According to various embodiments, the cipher method may perform for bulk encryption of a cloud storage.

[0041] According to various embodiments, the cipher method may further include receiving the data from a higher layer of a crypto application programming interface.

[0042] According to various embodiments, the cipher method may further include providing the at least one of encrypted data or decrypted data to the higher layer.

[0043] FIG. 2 shows a computing system 200 according to various embodiments. A plurality of servers 202 may, for example via a hub 204, communicate with a data base 206 (which may be large of size, for example 1 PB (petabyte)). A encryption/ decryption server 208 may be provided to encrypt data, like indicated by 212, and to decrypt data, like indicated by 210. Like indicated by 214, the encryption/ decryption server 208 may have a multi-core chip with a plurality of cores, and each core may have assigned one or

more levels of cache. Furthermore, the encryption/ decryption server 208 may have a main memory.

[0044] According to various embodiments, a data accumulator process may be provided to support large data cipher protection by hardware. A hardware cipher device driver design may be provided which may use the Asynchronous Block Cipher to collect multiples of 512-byte data up to the maximum number of bytes that the hardware cipher can support. This may allow the device driver to minimize the hardware operations overhead whenever it starts the encryption/decryption functions.

[0045] According to various embodiments, a task pipeline may be incorporated to schedule the data accumulator process and hardware cipher process. In addition to the data accumulator process, the hardware cipher and the device driver may support two sets of source and destination buffers to form a pipeline. These two sets of buffers for pipelining may allow the hardware cipher device driver to continue collecting the 512xN-byte data while the hardware is doing an encrypt/decrypt operation.

[0046] FIG. 3 shows an illustration 300 of building blocks of the proposed design of Asynchronous Block Cipher Driver for accelerating data protection for large scaled shared storage systems. An architecture of the asynchronous block cipher according to various embodiments is shown.

[0047] A crypto device mapper 302 may be provided. A crypto subsystem 304 may include a transform API 306, transform Ops 308, and an algorithm API 310. An asynchronous block cipher driver 312 (which may operate as a data accumulator) may include an encrypt/ decrypt module 302, a data queue 318, a queue manager 310, and a crypt manager 316. A hardware cipher 322 (which may operate as a data pipeline) may

include a first buffer 324 (Buffer_S1) and a second buffer 326 (Buffer_S2), which may input data to a cipher 328. A third buffer 330 (Buffer_T1) and a fourth buffer 332 (Buffer_T2) may be provided. Buffer S1 and S2 are input buffers and Buffer T1 and T2 are output buffers. The dashed blocks correspond to 614, the hatched blocks with lines from upper left to lower right correspond to 616 and the hatched blocks with lines from lower left to upper right correspond to 618. In a first step, the data buffer 324 is copied from the source buffer to FPGA. In a second step, the cipher FPGA process 328 for data buffer 324 will start. While 328 is starting processing, the next data buffer 326 is copied from the source buffer to FPGA. In a third step, the output from process 328 for buffer 324 is written to output data buffer 330. At the same time, the data buffer 326 is being processed by 328. In a fourth step, the output from process 328 for buffer 326 is written to output data buffer 332. While step 4 is occurring, step 1 can start again and this cycle will go on until the whole encryption job is done.

[0048] FIG. 4 and 5 illustrate the workflow and performance computation according to various embodiments.

[0049] FIG. 4 shows a flow diagram 400 of a method according to various embodiments, including processing of a cryptoAPI encrypt/ decrypt module 402, a queue manager 404 (for example queue manager 314 of FIG. 3), and a crypt manager 406 (for example crypt manager 316 of FIG. 3).

[0050] When the higher layers of the CryptoAPI calls an encryption or decryption process (step 408), the hardware cipher device driver puts the data at the end of the data queue (step 410) and passes back the control to the higher layers, and may also wake up the queue manager (step 412), which may receive the wake-up event in 416; the queue

manager 414 may also have an idle state 414. This may allow the higher layers to keep on queuing data until the specified queue size is filled up. There may be two main threads running according to various embodiments, namely the queue manager 404 and the crypt manager 406.

[0051] The queue manager 404 may check if there is enough data on the queue to let the hardware cipher process (for example, the queue manager 404 may access the data queue in 420, may get the byte-count from the data queue 422, and in 424 may determine whether the byte-count is enough or not). If there is enough data on the queue, or a timeout to fill up the pipeline buffer happens, the queue manager 404 may start the crypt manager 406 in 426, and then the queue manager 404 may start checking for enough data on the data queue again in 418. If not enough data is on the queue, the queue manager 404 may continue processing in 418 after determining whether the byte-count is enough in 424.

[0052] In 430, the crypt manager 406 may receive the wake-up event 430 sent by the queue manager 404. The crypt manager 406 may also have an idle state 428.

[0053] In 432, the crypt manager 406 may check for an available pipeline buffer. In 434, the crypt manager 406 may determine whether a pipeline buffer is available. If a pipeline buffer is not available, processing may continue in 432; if a pipeline buffer is available, processing may continue in 436. In 436, the crypt manager 406 may copy the data from the data queue to that buffer (for example the pipeline buffer). After copying, the crypt manager 406 may, in 438, invoke the hardware cipher to start processing the data. In 440 the crypt manager may read the hardware status register. In 442, the crypt manager 406 may determine whether the operation is complete. If the operation is not

complete, processing may continue in 440. If the operation is complete, processing may continue in 444. In 444, after the hardware cipher finishes processing the data, the crypt manager 406 may copy the data to the higher layers.

[0054] While the hardware cipher processes the data, the queue manager 404 may continue to check for available data and may start another instance of the crypt manager 406 to find an available pipeline buffer for hardware to process.

[0055] FIG. 5 shows an illustration of a workflow with big chunk cipher (in other words: big chunk encryption) and pipeline processing. A data accumulator is shown in the upper portion of FIG. 5, and pipeline processing is shown in the lower portion of FIG. 5. A file system 502 and a cryptographic device mapper 504 are shown. A device driver 506 may include an asynchronous block cipher module 508, a batch data collection module 510, and a synchronous data protection module 512. A plurality of CPUs (central processing units) 514 may be provided. The CPUs 514 and the device driver may be in communication with a north bridge 516 and a south bridge 528. The north bridge 516 may communicate with a memory 518, which may include a first source buffer 520 (Src_Buffer1), a first destination buffer 522 (Des_Buffer1), a second source buffer 524 (Src_Buffer2), and a second destination buffer 526 (Des_Buffer2). The south bridge 528 may communicate with a FPGA 534 (Field Programmable Gate Array), which may include an encoding module Enc, a decoding module Dec, a Secure Hash Algorithm module SHA and a random number generator RND), a local partition 530 (which may include an encrypted single partition) and a NW (network) storage 532 (which may include an encrypted single partition). Round K as illustrated in FIG. 5 is the processing of block K while Round K+1 is for block K+1. The processes of each block includes 614,

616 and 618. The process of subsequent block K+1 doesn't need to wait until block K to complete. It starts as soon as 614 of Block K finished.

[0056] According to various embodiments, a data collector layer may be provided before sending to a pipeline (or channel) so that the use of hardware cipher resource is maximized.

[0057] FIG. 6 shows an illustration 600 of high throughput with pipelined data cipher. Processing of a first data block 602 (for example a 4 KB), a second data block 604 (for example a 4 KB), a third data block 606 (for example a 4 KB), and a fourth data block 608 (for example a 4 KB) are illustrated over a time line 612 (indicating micro seconds). Data preparation is shown as blocks 614 bounded by dashed lines. FPGA processing is shown as hatched blocks 616 filled with lines from upper left to lower right. Data write is shown as hatched blocks 618 filled with lines from lower left to upper right. As can be seen from FIG. 6, only the 1st 4KB block 602 of data to be encrypted has a higher latency, and after that, the throughput is 4 KB/ 9 micro seconds.

[0058] According to various embodiments, a data accumulator with pipelining may minimize the hardware operations overhead (DMA operations initialization). According to various embodiments, an increased cipher operations throughput may be provided, for example an about 4x performance increase. According to various embodiments, asynchronous operations may be provided, and the queue manager may start collecting data as soon as the control goes to the crypt thread (which may mean synchronous hardware cipher communications).

[0059] According to various embodiments, devices and methods may be provided to accelerate data protection for large-scale shared storage systems.

[0060] According to various embodiments, a data accumulator process may be provided, for example supporting asynchronous block cipher of CryptoAPI and collect multiples of 512-byte data every encrypt-decrypt call. Upon reaching the maximum data that the hardware can support, a process to start hardware cipher processing may be invoked.

[0061] According to various embodiments, data pipelining may be provided. Hardware support for pipelining may be provided. Two sets of buffer channels may be provided for pipelined processing on the hardware. Data collection may happen while the hardware cipher is processing.

[0062] Table 1 shows a comparison of various ciphering methods.

Hardware Accelerated Cryptographic Solutions	Performance
Synchronous Cipher (512 Bytes Max Data Block) – commonly used	101 MBps
Asynchronous Cipher (512 Bytes Max Data Block) With Pipeline	140 MBps
Asynchronous Cipher With Data Accumulator and Pipelining	455 MBps

Table 1.

[0063] While the invention has been particularly shown and described with reference to specific embodiments, it should be understood by those skilled in the art that various

changes in form and detail may be made therein without departing from the spirit and scope of the invention as defined by the appended claims. The scope of the invention is thus indicated by the appended claims and all changes which come within the meaning and range of equivalency of the claims are therefore intended to be embraced.

Claims

What is claimed is:

1. A cipher device comprising:
 - a data collector configured to receive data to be processed;
 - a cipher circuit configured to perform at least one of encryption or decryption of the data; and
 - a control circuit configured to determine whether the amount of data received by the data collector fulfills a pre-determined criterion, and configured to instruct the cipher circuit to perform the at least one of encryption or decryption of the data if the amount of data received by the data collector fulfills the pre-determined criterion;
 - wherein the pre-determined criterion comprises whether the amount of data is higher than a pre-determined threshold; and
 - wherein the pre-determined threshold is larger than the amount of data per sector of a hard disk drive for which the cipher device is to provide its services.
2. The cipher device of claim 1,
 - wherein the pre-determined threshold is a value of at least 4 KB.
3. The cipher device of claim 1,

wherein the cipher circuit is configured to perform the at least one of encryption or decryption of the data in a pipeline fashion.

4. The cipher device of claim 1,
wherein the control circuit is further configured to instruct a further instance of the cipher circuit to perform the at least one of encryption or decryption of the data if the amount of data received by the data collector since the previous instruction to the cipher circuit fulfills the pre-determined criterion, if the cipher circuit did not finish the previously instructed processing.
5. The cipher device of claim 1,
wherein the cipher device is configured for bulk encryption.
6. The cipher device of claim 5,
wherein the cipher device is configured for bulk encryption of a cloud storage.
7. The cipher device of claim 1,
wherein the data collector is configured to receive the data from a higher layer of a crypto application programming interface.
8. The cipher device of claim 7,
wherein the cipher circuit is further configured to provide the at least one of encrypted data or decrypted data to the higher layer.

9. A cipher method comprising:
 - receiving data to be processed;
 - performing at least one of encryption or decryption of the data;
 - determining whether the amount of data received by the data collector fulfills a pre-determined criterion; and
 - instructing the at least one of encryption or decryption of the data if the amount of data received fulfills the pre-determined criterion;
 - wherein the pre-determined criterion comprises whether the amount of data is higher than a pre-determined threshold; and
 - wherein the pre-determined threshold is larger than the amount of data per sector of a hard disk drive for which the cipher device is to provide its services.
10. The cipher method of claim 9,
 - wherein the pre-determined threshold is a value of at least 4 KB.
11. The cipher method of claim 9, further comprising:
 - performing the at least one of encryption or decryption of the data in a pipeline fashion.
12. The cipher method of claim 9, further comprising:
 - instructing a further instance of performing the at least one of encryption or decryption of the data if the amount of data received since the previous instruction

of performing the at least one of encryption or decryption of the data fulfills the pre-determined criterion, if the previously instructed at least one of encryption or decryption of the data is not finished.

13. The cipher method of claim 9,
wherein the cipher method performs bulk encryption.
14. The cipher method of claim 13,
wherein the cipher method performs for bulk encryption of a cloud storage.
15. The cipher method of claim 9, further comprising:
receiving the data from a higher layer of a crypto application programming interface.
16. The cipher method of claim 15, further comprising:
providing the at least one of encrypted data or decrypted data to the higher layer.