White Paper

Securing Data at the Speed of Light with gKrypt

V1.2

Table of Contents

Introduction

As enterprise and consumer software transitions to the Cloud, there's an acute need for painless data security solutions which help protect your conversations, data and transactions in all their states. Every year, billions of dollars go down the drain as a result of data breaches. The centerpiece for any data security solution is encryption, which scrambles data, rendering it useless for an intruder and represents the last line of defense. Strong encryption schemes like AES, however, put an undesired burden on the processor, something known as "performance tax" in the industry. gKrypt eliminates encryption induced performance degradation by hyper-accelerating encryption on graphic chips or GPUs, all the while supporting hardware accelerated CPU based encryption.

While startups may decide to compromise the security of their consumer data through unencrypted data storage on the cloud, enterprise is still reluctant to buy into the promise of PaaS. Regulations like HIPAA, HITECH and PCI-DSS etc. have developed 'teeth', exponentially increasing the cost of a data breach, both financial and the loss of business credibility.

Cryptography is at the core of any effective data security solution since it scrambles data data-in-motion illegible. When *all* data is encrypted, in an event of security breach, the confidentiality of the data will not be compromised even if the physical storage medium is stolen and under current regulations, if the stolen data is encrypted, the company does not has to reveal the breach to its customers which ultimately saves its business credibility and customer confidence.

As lucrative as cryptography may sound, it comes at a cost, commonly referred to as "performance tax" in the industry because data encryption is a compute intensive process which requires dedicated processor power. The most widely adopted cipher is AES, approved by US Government in 2001 and incorporated by NIST as FIPS 197 publication. It is offered by almost every *crypto module*. Its widespread adoption was primarily fuelled by the fact that up to this day, no AES crack has been reported.

Introduction

Challenge	• **Immunize confidential data:** Prevent compromising confidential data in all its states, a) Data at rest b) In motion and c) data in application, as a last shield of defense. Achieve this with no impact on performance or increased TCO.
Solution	• **Secure all data with AES:** Implement AES cipher as the core component of data security strategy. Achieve zero impact on performance by utilizing gKrypt for AES encryption and decryption.
Implications	• **Customer confidence:** Business credibility and customer confidence is reinforced when you make your security policy compliant with relevant regulation/s, immunizing sensitive data even in the case of data breach. • **Lower implementation, operating costs and TCO:** The use of gKrypt eliminates the need for acquiring new high-end CPUs like Intel® Xeon processors for encryption.

This paper presents gKrypt as an attractive solution to secure data without compromising performance and lowering the implementation and operational costs at the same time. I will walk through usage scenarios in both states of data: At rest and in motion. gKrypt's architecture is discussed and finally performance gains are shown when compared with Intel® AES-NI™, the gold-standard in AES performance.

Some useful definitions before we begin.

Advanced Encryption Standard (AES)

AES is a variant of Rijndael block cipher which limits the data block size to 128 bits and permits three key sizes, 128-bit, 192- bit and 256-bit. AES has been adopted as encryption standard by U.S. government since 2001[1].

Block Cipher

An encryption algorithm which works on fixed-length groups of bytes, called blocks, is referred as Block Cipher. The inverse of block cipher is stream cipher which can encrypt and decrypt data with arbitrary length.

Deflate

Deflate is a lossless data compression algorithm that uses a combination of the LZ77 algorithm [2] and Huffman coding to achieve better compression ratio compared to either of its building blocks. It was originally defined by Phil Katz for version 2 of his PKZIP archiving tool, and was later specified in RFC 1951[3]. Deflate is currently being used in PKZIP, gzip, zlib and 7-zip compression tools.

Huffman Coding

Huffman is an entropy encoding scheme which achieves compression by assigning smaller codes to frequently used symbols and longer codes to less frequent symbols. Huffman coding is frequently used for lossless data compression.

Hash Function

Cryptographic hash functions are algorithms that take an arbitrary block of data (message) and return a fixed-size bit string, referred as hash value or message digest, such that an accidental or intentional change of data will change the hash value with very high probability. Cryptographic hash functions have many information security applications, notably in digital signatures, message authentication codes (MACs), and password verification.

Lempel–Ziv (LZ77)

LZ-77 is a lossless data compression algorithm, published by Abraham Lempel and Jacob Ziv in 1977; hence the name LZ77. The algorithm operate by searching for matches between the text to be compressed and a set of strings contained in a data structure (called the 'dictionary') maintained by the encoder. When the encoder finds such a match, it substitutes a reference to the string's position in the data structure and achieves compression.

Rijndael

Rijndael is a collection of block ciphers proposed by Belgian cryptographers, *Joan Daemen* and *Vincent Rijmen*, for NIST initiated AES selection process. The Rijndael comprises three block ciphers with 128, 192 and 256 bits block sizes. Each of these block ciphers can be used with key sizes of 128, 192 and 256 bits; providing nine possible variants.

Secure Hashing Algorithm (SHA)

SHA is set of cryptographic hash functions designed by National Security Agency (NSA) and published as U.S. Federal Information Processing Standard (FIPS). The first member of SHA family was published in 1993 and was replaced by SHA-1 in 1995, to address a security flaw in the initial version.

HIPAA

The Health Insurance Portability and Accountability Act of 1996 (HIPAA; Pub.L. 104-191, 110 Stat. 1936, enacted August 21, 1996) was enacted by the United States Congress and signed by President Bill Clinton in 1996. Title I of HIPAA protects health insurance coverage for workers and their families when they change or lose their jobs. Title II of HIPAA, known as the Administrative Simplification (AS) provisions, requires the establishment of national standards for electronic health care transactions and national identifiers for providers, health insurance plans, and employers.

HITECH

Health Information Technology for Economic and Clinical Health Act (HITECH Act), enacted as part of the American Recovery and Reinvestment Act of 2009, addresses the privacy and security concerns associated with the electronic transmission of health information. The HITECH Act requires HIPAA covered entities to report data breaches affecting 500 or more individuals to HHS and the media, in addition to notifying the affected individuals.

PCI-DSS

The Payment Card Industry Data Security Standard (PCI DSS) is an information security standard for organizations that handle cardholder information for the major debit, credit, prepaid, e-purse, ATM, and POS cards. Defined by the Payment Card Industry Security Standards Council, the standard was created to increase controls around cardholder data to reduce credit card fraud via its exposure. Validation of compliance is done annually — by an external Qualified Security Assessor (QSA) for organizations handling large volumes of transactions, or by Self-Assessment Questionnaire (SAQ) for companies handling smaller volumes.

gKrypt Overview

gKrypt is the first crypto module customized to use GPUs as accelerators to swiftly secure your applications with increased system responsiveness. gKrypt supports both CUDA and OpenCL technologies to run on a wide range of hardware including Nvidia® Tesla™, Geforce™, Quadro™ as well AMD® Radeon™ and FireStream™ series for both desktop and mobile variants. Furthermore, due to highly optimized parallel pipelines, it delivers many times performance gains compared to current solutions, including Intel® AES-NI™, while retaining the security characteristics. Being fully backward compatible with CPU only machines, you always know that the best resources of your system are utilized without sacrificing compatibility using gKrypt. The added support for multi-GPUs and Nvidia® Kepler™ based GPUs ensures maximum performance for devices supporting PCI Express 3.0 bus giving up to **80Gbps** application throughput per device for AES 256-bit in consumer and enterprise systems.

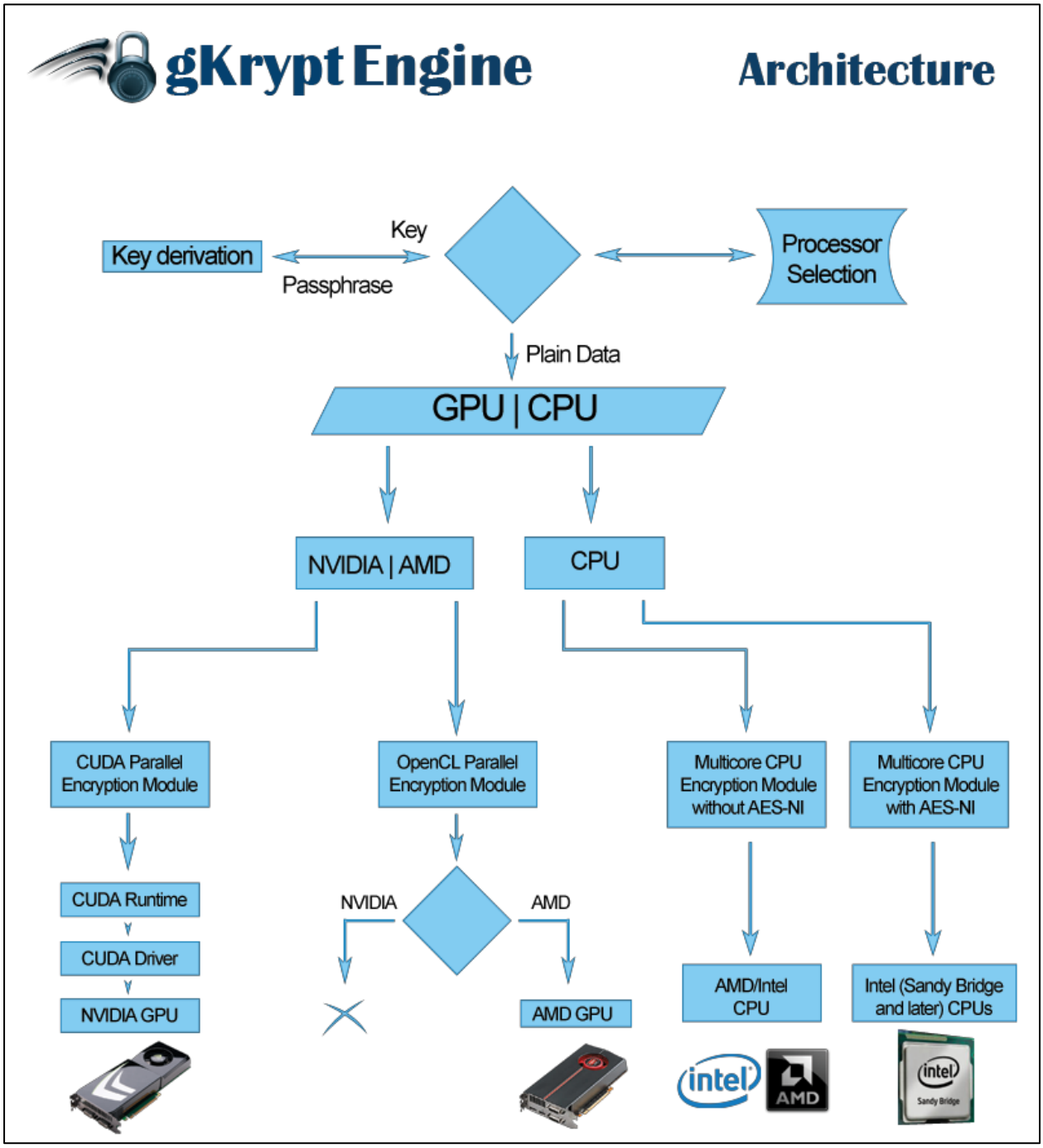
Functionality & Support

Ciphers

gKrypt supports complete set of Rijndael block ciphers with all possible block sizes and key sizes. A brief list of the primary cipher routines is listed below. This is a condensed list which covers the core cipher routines available in gKrypt. For complete list of functionality, refer to the Documentation which comes with the package.

`gktRijndael128[Encrypt/Decrypt]ECB`	Encrypt/Decrypt with Rijndael 128 Supported Key Sizes: 128,192 and 256bits
`gktRijndael192[Encrypt/Decrypt]ECB`	Encrypt/Decrypt with Rijndael 192 Supported Key Sizes: 128,192 and 256bits
`gktRijndael256[Encrypt/Decrypt]ECB`	Encrypt/Decrypt with Rijndael 256 Supported Key Sizes: 128,192 and 256bits
Hashing	SHA-1 and MD5
Compression	LZ-77 variant data compression
Supported GPUs	NVIDIA® and AMD®
Supported CPUs	Intel® and AMD®. AES-NI™ is used for Sandy Bridge and later architectures
GPU APIs	NVIDIA® CUDA™ and Khronos™ OpenCL™
Multiplatform	Microsoft® Windows, Linux and Apple® Mac OS
Multilingual	C/C++, Java*, C#* and Python*

gKrypt Architecture Overview



gKrypt Usage Scenarios

There are two distinct types of data that may be secured with encryption: Data at rest and data in motion. Data in motion is further expanded into data in application and network data. This section lays out potential use cases for different types of data. There are standards and protocols governing data in each of these types and AES encryption is already supported in most if not all of them. With gKrypt, we attain two primary benefits:

- Improved encryption speed.
- Extremely low CPU occupancy which means minimal "performance tax".

The combination of accelerated encryption and no additional workload for the CPU manifests in all three types of data. The following subsections give an overview of protocols and standard used in each type of data.

1. Data at Rest

This consists of all types of data storage mediums including magnetic disks, SSDs etc and the solutions for encrypting data at rest are broadly categorized as FDE (Full Disk Encryption). With the adoption of Cloud model, the demand for data storage has risen exponentially and so is the threat to the stored data which is why enterprise applications are still reluctant to move valuable data to public clouds. Securing peta-bytes of data using encryption is a natural solution but it comes at a very cost to performance and demands hefty investment in more computing power to make encryption a feasible solution. One possible solution is Intel's newly introduced AES-NI (AES New Instructions) in Xeon 5600 series processors but it also has the downside of (still) putting additional workload on an already occupied CPU. Commercial as well as (Free) Open Source solutions have adopted this for their FDE offerings. With gKrypt, the need for purchasing expensive Xeon series processors is eliminated since the bulk of the encryption work is done on the GPU, sparing the CPU to continue processing business logic of the application. As already explained, gKrypt implements the AES standard and therefore, since it is backward compatible, ISVs as well as end-user scan easily put gKrypt's AES encryption/decryption functions in their libraries/applications.

2. Data in Motion

As already mentioned, data in motion is further categorized into network data and data in processing. We will briefly touch both these types and where gKrypt fits in the stack.

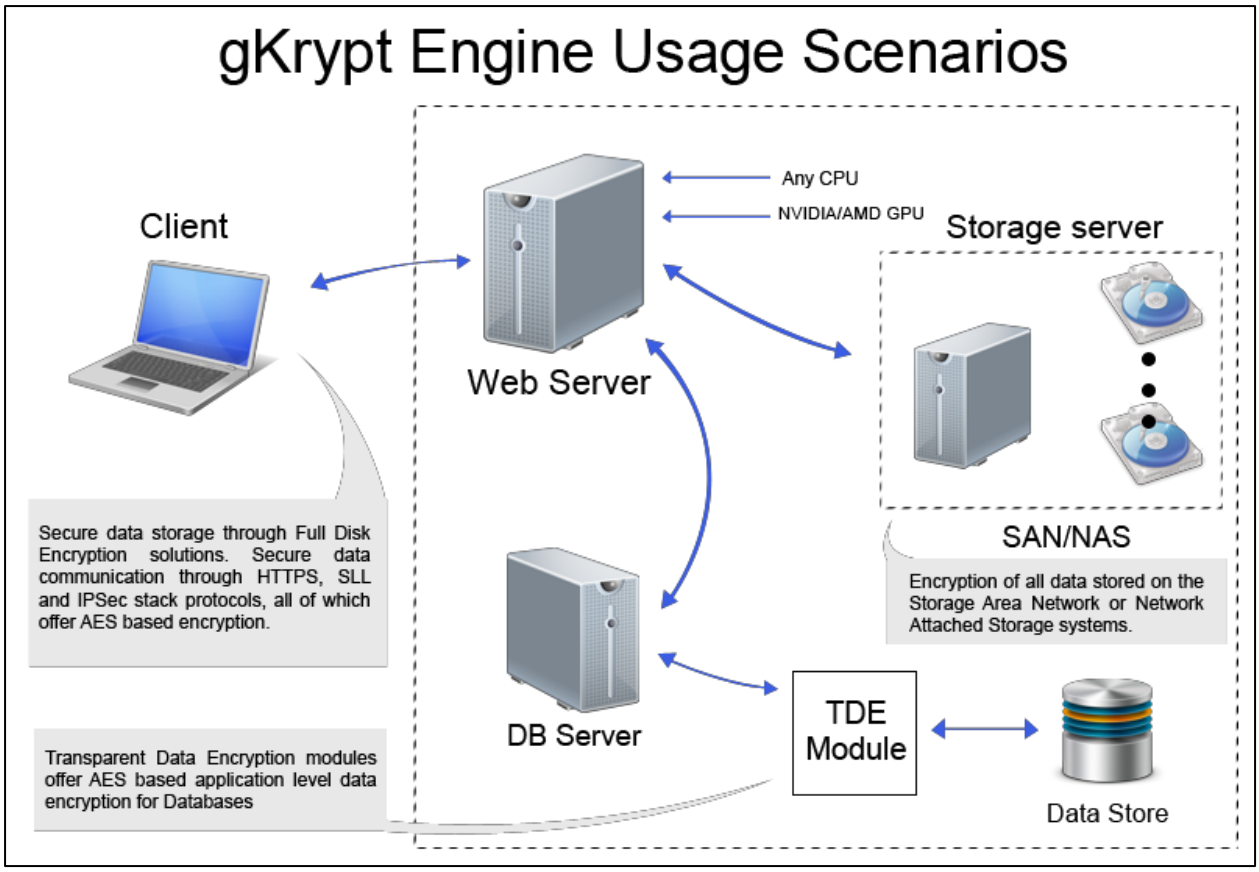
2.1 Network Data

Millions if not billions of transactions conclude every hour over the internet and intranet (local LAN). The secure set of protocols like HTTPS, SSL, TLS, IPSec etc all use some form of data encryption for secure communication over the internet. These protocols also support AES-128 encryption. A typical transaction would begin with a handshake between the client and the server. Once that's established and keys are shared, which in case of asymmetric cipher would be a pair of keys and in case of symmetric cipher technique, would be one key shared with both server and client, actual communication begins. Client encrypts outgoing bytes and decrypts incoming bytes. This is where gKrypt comes in. As in the case of data-at-rest, CPU based AES routines are replaced with gKrypt's AES routines. No change in code required and the result is improved encryption speed and drastically reduced CPU workload.

2.2 Data in Process

One of the primary examples of data in process is the database encryption at cell, column and database table level. Every enterprise database offers something called TDE – Transparent Data Encryption – where, as the name specifies, data is encrypted in real-time, transparent from the user/admin which saves additional complexity while inserting and fetching data and also gives database manufacturers the option to keep the "business logic" separate from data security functionality. There are 3rd party solutions for adding TDE in popular databases like MySQL, Oracle, MS SQL Server. These TDE solutions in turn use cryptography libraries to perform the encryption/decryption of data. Out of all other use cases for data encryption, this is the most demanding use case in terms of performance since database transactions need to happen in real-time and we cannot wait, for example, to insert rows in bulk while they're being encrypted. Again, Intel IPP Cryptography (which uses AES-NI on Xeon 5600 series processor) is a viable option but gKrypt adds even more value by eliminating the need to buy more expensive computing resources to add encryption while keeping up with the same level of performance.

Usage Scenarios

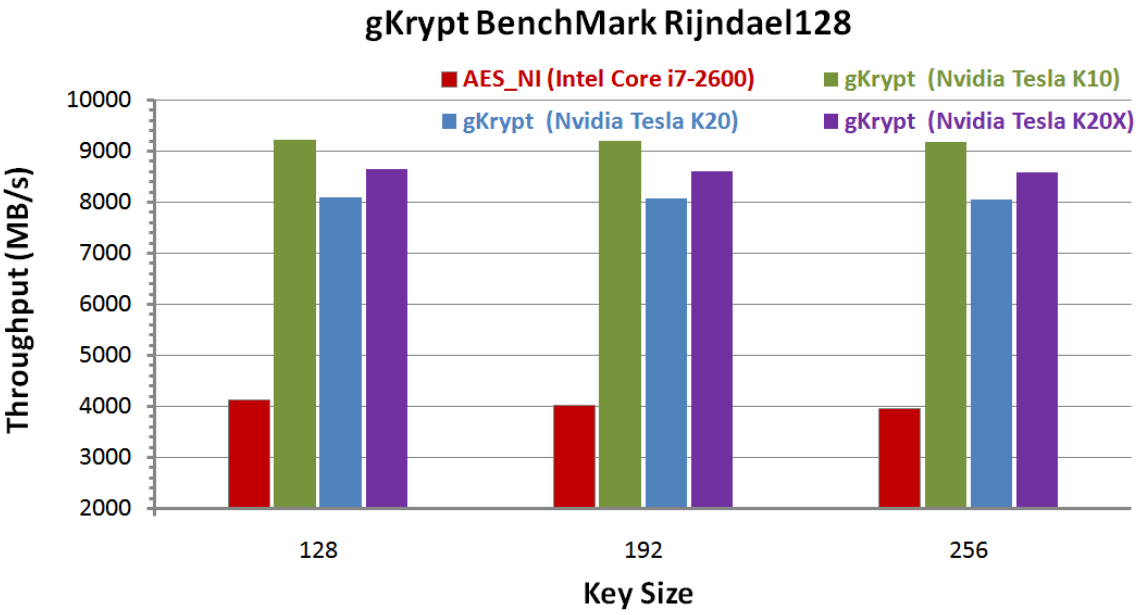


Enhanced Performance

This section compares gKrypt's performance with Intel® IPP™ (AES-NI™) which is a gold standard for high performance AES routines. Performance gains are two-fold, first it shows how fast gKrypt's AES routines are compared with Intel® IPP™ and second is the fact that this performance is achieved without putting any encryption related workload on the CPU, hence zero *performance-tax*.

Performance of a cipher implementation may be gauged by metrics like the function/kernel level throughput, application level throughput which is also an indicator of the specific implementation methodology used. We will use the application throughput metric to compare gKrypt against Intel® IPP™ with and without Intel AES-NI™.

The Benchmarking system is equipped with a second generation Intel Core i7 2600k CPU running at 3.4 GHz, 8GB DDR3 RAM and NVIDIA TESLA C2050 GPU.



Performance Implications for Consumer Applications

In consumer applications that require real-time and high throughput processing, the CPU is completely or partially paralyzed due to increased overload of cryptography and the application performance suffers. This causes stutters, reduced system responsiveness, resource failures and hang-ups. Such applications include video conferencing, file sharing/sending software, desktop virtualization, online collaboration tools, network gaming, web browsers, cloud applications, full disk encryption and many more. The gKrypt can offload the encryption functionality if a GPU is available in the system, sparing the CPU to focus on the application processing. This improves system responsiveness and speed while keeping your data equally secure.

Performance Implications for Enterprise Applications

The enterprise solution developers can easily cash upon the parallel processing potential of GPUs. gKrypt provides up to 10GB/sec per GPU application throughput for AES 256-bit Encryption/Decryption. This can be scaled to up to 8 GPUs with as much as 1536 processing cores per device to get up to 8X performance increment. This is ideal to tackle the huge workloads in database encryption (TDE), cloud based services, email backend servers, CRM and ERP applications, online payment systems, full disk encryption software, RSA and SSL handshaking, networking applications, online data storage and retrieval, operating systems and low level transport protocols among others. Due to higher performance per watt, ease of deployment and scaling as well as reduced costs, the gKrypt supporting GPU technologies for CUDA™ and OpenCL™ ensures most optimized systems for compute intensive cryptography workloads.

Conclusion

The need for securing sensitive information cannot be overemphasized. gKrypt boosts the encryption performance without requiring dedicated expensive processors, which effectively removes the performance-tax by reducing the cost and complexity from cryptography.

Download & Contacts

You can get started right away by contacting us on the website for evaluation. We have made a very nice, easy to use Windows file encryption app which uses gKrypt for encryption. You can download this app and try it on your PC. It has a “Speed” tab which tells you real-time encryption throughput for your CPU and GPU. Finally, you can contact us through email and/or phone.

Website	gkrypt.com
Evaluation Request	gkrypt.com/contact
Email	contact@tunacode.com

REFERENCES

[1] "Announcing the ADVANCED ENCRYPTION STANDARD (AES)" Federal Information Processing Standards Publications, November 26, 2001.

[2] "A Universal Algorithm for Sequential Data Compression", Jacob Ziv and Abraham Lempel; IEEE Transactions on Information Theory, Vol. 23, No. 3, pp. 337–343, May 1977.

Notice

ALL INFORMATION PROVIDED IN THIS WHITE PAPER, INCLUDING COMMENTARY, OPINION, GKRYPT ENGINE DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, "MATERIALS") ARE BEING PROVIDED "AS IS." GKRYPT DSS MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE.

Information furnished is believed to be accurate and reliable. However, TunaCode assumes no responsibility for the consequences of use of such information or for any infringement of patents or other rights of third parties that may result from its use. No license is granted by implication or otherwise under any patent or patent rights of TunaCode. Specifications mentioned in this publication are subject to change without notice. This publication supersedes and replaces all information previously supplied. TunaCode products are not authorized for use as critical components in life support devices or systems without express written approval of TunaCode Inc.

Copyright