

VAST Data Encryption

Table of Contents

Introduction	3
Building on a Secure Foundation	4
Operating System (OS)	4
Hardening the OS	4
FIPS Cryptography	5
Entropy	5
Cryptographic Library Linkage	5
External Key Management	6
In Memory Key Management	6
At-Rest Data Encryption	7
Tenant Segmentation	7
Encrypted Paths	7
Creating Encrypted Paths	7
Encrypted Path Key Management	7
Encryption Cipher	7
Data Encryption Process	8
1st Event	9
2nd Event	9
Crypto Erase	9
Conclusion	10

Introduction

The importance of robust security mechanisms cannot be overstated in today's interconnected digital landscape. Data protection becomes paramount as cyber threats grow in sophistication, especially for platforms tasked with large-scale data storage and management. This document details how VAST Data meets these ever-evolving security challenges and presents a deep dive into VAST's multi-faceted security approach that spans its foundational operating system to its sophisticated encryption methods, and specifically, the implementation of data encryption utilizing external key management.

Building on a Secure Foundation

Operating System (OS)

VAST partners with CIQ to provide and maintain a secure version of the Rocky Linux operating system (OS) for all nodes in the platform. This partnership ensures that the operating system is updated and patched regularly, addressing security vulnerabilities and maintaining compliance with regulatory standards. CIQ, as the founding sponsor of Rocky Linux, offers comprehensive support and services for the operating system, including security enhancements, performance optimization, and compliance with industry standards such as FIPS 140-3 and DISA STIG.

Hardening the OS

VAST applies the DISA STIG for Red Hat 8, leveraging the MAC 1 - Mission Critical Classified profile to all nodes in the platform. The STIG addresses various aspects, such as configuration settings, patch management, and access control that contribute to securing the VAST platform. By adhering to the DISA STIG guidelines, the VAST Data Platform establishes a more robust security posture that meets or exceeds regulatory compliance requirements such as PCI DSS, HIPAA, and SOX.

FIPS Cryptography

VAST utilizes the Federal Information Processing Standard (FIPS) 140-3 Level 1 accredited library Certificate #4675. VAST nodes operate with the kernel in FIPS mode, which signifies adherence to specific cryptographic standards. This mode ensures that all cryptographic operations performed by the system rely solely on algorithms that have been certified and validated against the FIPS 140 standard. VAST also leverages several NIST 140-3 level 1 capable libraries from CIQ, including GnuTLS, Kernel Crypto API, and Libgcrypt, which are in the testing phase with the National Institute of Standards and Technology's Cryptographic Module Validation Program.

Entropy

VAST's nodes utilize the Jitter RNG random number generator and CSPRNG, the kernel random number generator. The Jitter RNG library, which is based on CPU time jitter, has undergone extensive testing and validation to ensure it meets the stringent requirements of FIPS 140-2 and FIPS 140-3 standards. This validation process includes continuous self-tests and entropy source validation to guarantee the high quality and security of the generated random numbers. Jitter RNG is recognized for its ability to provide high-quality entropy across a wide range of CPUs and operating systems, making it a reliable source for cryptographic operations. Additionally, the CSPRNG, as part of the kernel's random number generation system, complements the Jitter RNG by providing a robust and secure pool of entropy.

Cryptographic Library Linkage

VAST links all in-flight and at-rest cryptographic operations to the FIPS validated OpenSSL 1.1.1 library. This library provides secure encryption capabilities for VAST cluster management (over OpenSSH and HTTPS) as well as in-flight encryption of data being transferred to and from the VAST system. VAST implements TLS 1.3 for protocols such as NFSv4 over TLS (File), S3 (Object), KMIP, and mTLS Replication between VAST deployments.

External Key Management

VAST Data Platform supports external key management (EKM) with multiple providers, including:

- Thales CipherTrust Manager (via the Thales REST API)
- Fernetix VaultCore (via the KMIP protocol)

In Memory Key Management

When a Leader Node is elected, it retrieves the Master Key (MK), Key Encryption Key (KEK), and Data Encryption Key (DEK) from the Enterprise Key Manager (EKM). The platform has the following controls implemented to secure access to the keys in memory:

- VAST does not allow any RPC, API, or CLI method to produce the keys in any way.
- Full Linux process core dumps are disabled via the kernel.
- Kernel kdump is disabled via one-way kexec.
- Memory locations with keys are locked with mlock and are never paged to non-volatile storage.
- The platform securely purges keys from memory utilizing `OPENSSL_cleanse()` when a key is revoked, a container is shut down, a pause command is issued, or during a system shutdown.
- Functions which can inspect running memory, such as `gdb`, `strace`, and `ptrace` attachment to running processes are disabled in the running kernel, preventing unauthorized access to keys.
- The ability to load unapproved kernel modules is disabled during the boot process.

VAST creates all keys within the EKM and leverages the EKM to store keys persistently. The platform supports the following key management capabilities:

- Master key rotation (14 Days)
- Key rotation of the KEK
- Key evocation of a KEK and DEK
- Key renewal (Rekey) of the DEK
- All communications with the EKM are over TLS 1.3, leveraging ES-256-GCM encryption
- Every tenant has a unique KEK and DEK per tenant or shares a DEK
- Key renewal (rekey) by the EKM for the KEK and DEK

At-Rest Data Encryption

Tenant Segmentation

Tenant-level encryption in the VAST Data Platform is managed with the use of Encryption Groups, which are collections of encryption keys that secure data for a specific tenant. When a tenant is created, the administrator can either generate a new Encryption Group with a unique Key Encryption Key (KEK) and Data Encryption Key (DEK) or select an existing Encryption Group.

Encrypted Paths

When creating a tenant in the VAST Data Platform configured with an Enterprise Key Manager (EKM), the administrator has the option to create a new “Encryption Group” with a unique Key Encryption Key (KEK) and Data Encryption Key (DEK) or select an existing Encryption Group. The platform also supports creation of “Encrypted Paths” within a given tenant/Encryption Group. An Encrypted Path defines a new Encryption Group which can be applied to a specific sub-directory within a tenant.

Creating Encrypted Paths

Administrators can create Encrypted Paths by creating a new View and selecting the option to create an Encrypted Path alongside the directory represented by that View. Encrypted Paths may also be created explicitly by Administrators separately from Views to ensure that data within a location in the filesystem is always encrypted using a specified Encryption Group.

Encrypted Path Key Management

Encryption Groups used for Encrypted Paths within a Tenant are wrapped using the Encryption Group’s KEK for secure retrieval from the EKM. Encrypted Paths can be shared out as Views, allowing view-level segmentation of data within a tenant.

Encryption Cipher

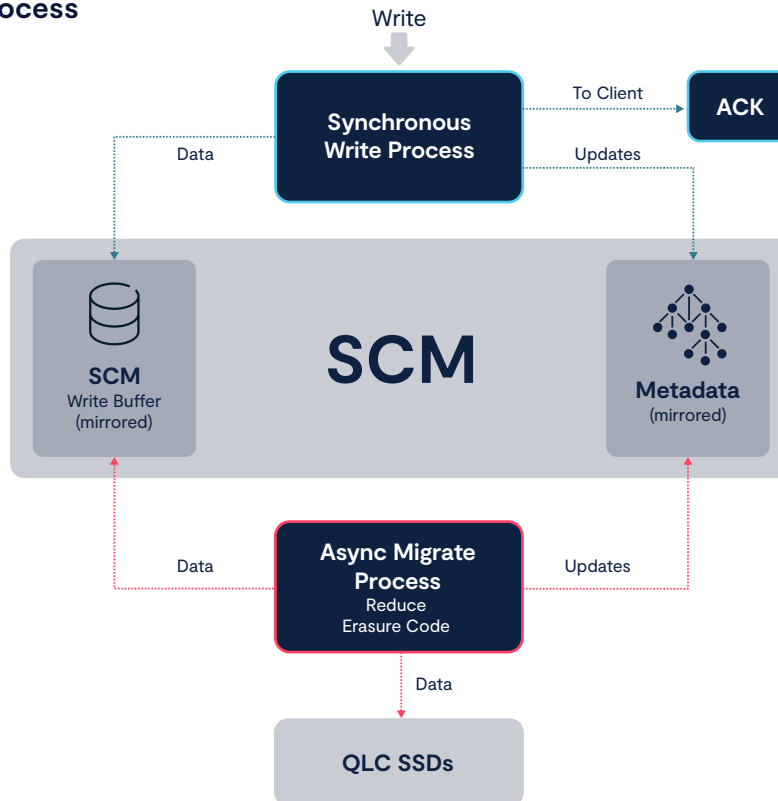
VAST utilizes AES-XTS-256 encryption, which is a specific implementation of the Advanced Encryption Standard (AES) algorithm in XTS mode with a 256-bit key size. This encryption method is commonly employed to safeguard data at rest, particularly on storage devices such as solid-state drives (SSD). Below is an overview of AES-XTS-256 encryption for securing data within the VAST Data Platform:

- **AES Encryption:** AES is a symmetric encryption algorithm renowned for its robust security and operational efficiency. AES functions on fixed-size data blocks.
- **256-Bit Key Size:** AES-256 employs a 256-bit encryption key that yields an astronomical number of possible keys. This immense key space renders brute-force attacks exceedingly impractical, as it would require an unfeasible amount of computational power to decipher the encrypted data without the correct key.

- **XTS Mode:** XTS (XEX-based Tweak Codebook mode with ciphertext Stealing) represents a specialized mode of operation tailored for block-level encryption. VAST uses XTS with a unique “Tweak Value” per encrypted block regardless of where the block is stored within the VAST system. This guarantees that identical plaintext blocks do not result in identical ciphertext blocks, even when identical data is written to distinct locations in the VAST platform.

The VAST platform has two encryption events for all data written or inserted into the platform via NFS, S3, SMB, and SQL. The encryption process is outlined below:

Data Encryption Process



1st event:

Synchronous Write

- Data is received from Buffer and encrypted in 512-byte chunks and a header is added with the encryption key ID
- Mirrored or RAIDed the encrypted chunk to storage class memory
- Updates metadata
- Sends ACK

2nd event:

Asynchronous Migrate

- System reads encryption ID from header then locates the specified key from DRAM
- Decrypts and Reads SCM buffer
- Calls Similarity pipeline and reduces block size
- Encrypt reduced block with same key use for the initial write
- Erasure codes
- Writes to QLC
- Updates metadata
- Pointers from SCM to QLC
- Releases write buffers

1st Event

Initially, prior to platform acknowledgment of an I/O, data is segmented into 512-byte units, each garnished with a header. The respective tenant's DEK key ID is written into the chunk's header, then mirrored or RAIDed on multiple storage class memory (SCM) devices. Following this, the system metadata is updated, and an acknowledgment is sent to the client.

2nd Event

The migration function which pulls data from SCM and writes it to QLC NAND, segments the data into 256-kilobyte shards and casts them broadly across the drives. Data retrieval from QLC necessitates chunk location details, the key ID, a reference block from SCM, and decryption for client requests.

Crypto Erase

VAST's platform allows for cryptographic erasure of data at the tenant or encrypted path level by enabling administrators to revoke specific encryption keys within the system. By revoking a Data Encryption Key (DEK) associated with a tenant or path, all data encrypted by that DEK is marked as unavailable and inaccessible. Once the keys are revoked, the administrator can proceed to delete the tenant and/or views associated with those keys. During this process, the revoked keys are permanently removed from the Encryption Key Manager (EKM). Simultaneously, the system marks all data encrypted with the revoked keys for immediate deletion by the garbage collection process, ensuring a thorough and secure removal of the data. This approach aligns with the NIST SP 800-88 Rev. 1 Guidelines for Media Sanitization, specifically the Cryptographic Erase Device Guidelines outlined in Appendix D. The guidelines provide considerations for key generation, media encryption, and key wrapping techniques to ensure the cryptographic erasure process offers sufficient assurance against future recovery of the data.

Conclusion

In the digital transformation era, the safety of data is at the forefront of concerns for organizations worldwide. The multifaceted security approach adopted by VAST, as discussed in this document, ensures the platform's resilience against a plethora of cyber threats. From the robust foundation of the operating system to the intricate processes of encryption and key management, each layer of protection fortifies data against breaches, unauthorized access, and potential vulnerabilities. As cyber threats continue to evolve, platforms like VAST serve as a beacon, showcasing the importance of comprehensive, multi-layered security in safeguarding valuable data assets.



For more information on the VAST Data Platform and how it can help you solve your application problems, reach out to us at hello@vastdata.com.

Disclaimer: View-level encryption becomes GA in the 5.2 release of the Vast Data Platform.