WHITE PAPER – **DECEMBER 2017**

VMWARE vSPHERE® VIRTUAL MACHINE ENCRYPTION

Virtual Machine Encryption Management

vmware®

Contents

Executive Summary

Virtual machine (VM) encryption has been around in different forms for many years and has met with various levels of success. The challenges of ensuring security versus running IT operations frequently led to solutions that, while secure, exponentially increased operational workload.

The attempt to force existing rules onto a newer platform caused difficulties. By its very nature, virtualization changes the game. Rather than looking to that layer to provide security services, existing encryption solutions try to take the same approach one would take with a laptop or bare metal server. This requires modifications to the VM operating system (OS) or disk layout.

VMware vSphere® 6.5 addresses the operational and security challenges by leveraging the hypervisor to perform the encryption with no modification to the VM. The security architecture of VMware ESXi™ achieves this goal at the hypervisor layer, which yields the following benefits:

- No modification to VM OSs – No changes to existing applications are required, providing a common method of encryption across any OS supported by vSphere.
- No specialized hardware or infrastructure required – The encryption works with existing storage devices and storage fabrics.
- Policy-based enforcement that is supported by the vSphere SDK and tools such as VMware vSphere PowerCLI™ – This provides easy integration into current and future provisioning solutions.

In this document, we will elaborate on how the security architecture and controls of vSphere VM encryption address the concerns of the security team while providing the IT operations team with the necessary tools to minimize impact on day-to-day operations.

This information is for both seasoned security specialists and experienced IT professionals. Some of the concepts herein might require a deeper understanding of hypervisor security. This is documented in the *Security of the VMware vSphere Hypervisor* technical white paper.

Traditional Encryption Solutions

Before describing the VMware solution for data encryption, we will discuss existing solutions by way of comparison. Generally speaking, there are two traditional approaches to encryption: in-guest and infrastructure-based.

In-Guest Encryption

In this scenario, encryption occurs within the guest VM. This is one of the more common methods of encryption outside of the virtualized data center. For example, many corporate laptops and desktops enforce the use of Microsoft Windows BitLocker, macOS FileVault, or Linux dm-crypt.

This type of solution typically uses a preboot partition to control access to the encrypted partition. This involves custom partitioning of disks. The system boots from the preboot partition. Keys are retrieved via a hardware device or software control to enable the encrypted partition to boot.

Each of these solutions requires additional setup and management. For example, it might be required to present a Trusted Platform Module (TPM) device to the VM. Multifactor authentication configuration might also be required. These solutions are OS specific in each case.

The following are among the significant challenges with some of these solutions:

- There is no common encryption policy across Windows, Linux, and other OSs.
  - Each is managed and configured separately.
- Encryption occurs in the context of the OS.
  - Encryption takes place in the same context as potential malware.
- Encryption might require disabling to apply updates to the OS or applications.
  - This adds to the operational burden and to the chances for error and misconfiguration.
- Changes in hardware configurations can lead to problems.
  - This includes changes that might cause specialized investigation of encryption failures.

All of these factors introduce large operational overhead costs. They all require individual configuration, management, and checking to ensure proper performance. Each environment has the capacity to run a unique number of VMs, beyond which operation becomes unwieldy and difficult to manage.

Infrastructure-Based Encryption

In this broad category, data is encrypted via the hardware deployed in the virtual environment. There are several points at which this encryption can occur.

Self-Encrypting Drives

Disk-based encryption is an approach by which the data is encrypted as it is written to disk. Self-encrypting drives (SEDs) have built-in hardware, which encrypts the stream of bits being written to an individual disk drive. Each drive is encrypted with a unique media encryption key (MEK), which is then encrypted with a key encryption key (KEK). If no KEK is used, no protection of the data is provided if the disk is moved to another system, even though the data is encrypted on the device via the MEK. From a hardware perspective, a SED without a KEK is essentially a “normal” disk.

KEKs for SEDs can be managed in two ways. The first is via local key management. In this scenario, the server RAID adapter is configured with individual KEKs for each server. At boot time, the adapter loads the KEK into the respective SED, unlocking the drive. This requires keeping track of which KEK is configured in each server. In case of adapter failure, adapter reconfiguration with the recorded KEKs is necessary for data retrieval. In the second method, the HBA on the server interfaces with an external key manager provider, retrieves the KEK, and loads it into the SED.

Array-Based Encryption

With array-based encryption, the controller in a storage array encrypts the data as it is written to the disks. Encryption can be performed via custom application-specific integrated circuits (ASICs) "in hardware" or in software. In both cases, key management can be achieved via an onboard key manager or through the use of an external Key Management Interoperability Protocol (KMIP)–compliant key manager.

Disadvantages to Disk- and Array-Based Encryption

The main disadvantage of disk-based encryption is that data is not encrypted until just before it reaches the storage medium. This means that it travels in the clear from the application through the storage fabric or network. This might not be a concern when, for example, a SED is used for local storage on a server. But most data centers are architected with data generated or obtained by systems that are separate from the systems that store the data.

This solution also lacks context of the workloads that are running, so it is not possible to manage workload granularity and multitenancy. For example, there is no easy way to enable different workloads to have different encryption keys.

Although disk-based encryption might satisfy a strict definition for protecting data at rest, the physical stealing of a disk or array is the main threat it protects against. Moreover, in the case of array-based encryption, the entire disk array comes into scope for a compliance audit. The management plane of the array must be analyzed to identify the members of the storage team who have access to the data as well as their privileges to perform operations on it.

Also, disk-based encryption can incur a significant cost. SEDs generally cost more than their standard counterparts, and arrays with encryption capabilities are usually high end and expensive.

Fabric-Based Encryption

Fabric-based encryption is possible when servers in a data center send their data to storage devices through a network that is usually dedicated to storage. The data is encrypted in the storage fabric as it leaves the physical server. There are two primary ways through which this is done.

Host Bus Adapter (HBA)

With the first method, data is encrypted by the server's host bus adapter (HBA) as it leaves the host and enters the storage network. For many HBA devices, key management occurs via existing external KMIP-based key managers. Because the encryption is handled at the host, data does not travel in the clear at all.

Disadvantages to HBA Encryption

The main drawback to HBA encryption is that it must be managed at the per-host level. This is not easy to scale up in large environments. It also greatly reduces workload portability: VMs encrypted on one host are not easily moved to another host. For this reason, it is also difficult to provide for multitenancy.

Switch-Based Encryption

With the second method, the data leaves the host and travels in the clear until it reaches a switch, which then performs the encryption before sending the data on to the storage array. The switch might be a Fibre Channel switch or, in the case of NFS, a network switch. The switch typically also integrates with an external, KMIP-compliant key manager.

Disadvantages to Switch-Based Encryption

As mentioned, data is encrypted only after it has left the host and arrived at the switch. The switch presents array-based LUNs or NFS shares to each host, which in turn presents the LUN or share as a datastore. This limits the ability and flexibility to move a VM from datastore to datastore. Using a mix of Fibre Channel and NFS requires encryption at multiple fabric switches. There is significant overhead in setup, configuration, and orchestration of VM workloads and infrastructure. Multitenancy is limited to the datastore on which a VM resides, limiting operational efficiency.

vSphere VM Encryption

vSphere VM encryption enables creation of encrypted VMs and encrypts existing VMs. Because all VM files that contain sensitive information are encrypted, the entire VM is protected. Only administrators with encryption privileges can perform encryption and decryption tasks.

What Is Encrypted

vSphere VM encryption supports encryption of the following types of files:

- VM files
- Virtual disk files
- Host core dump files

What Is Not Encrypted

Some files associated with a VM are not encrypted or are partially encrypted because they don’t contain sensitive information:

- Log files
- VM configuration files
- Virtual disk descriptor files

How Encryption Is Performed

To explain how encryption is performed, we first discuss the three major components used to encrypt:

- Key management server (KMS)
- VMware vCenter Server®
- ESXi hosts

Key Management Server

The vCenter Server instance requests keys from an external KMS. The KMS generates and stores KEKs and passes them to the vCenter Server instance for distribution. As a KMIP client, the vCenter Server system uses that protocol to facilitate use of the chosen KMS.

vCenter Server

The vCenter Server instance obtains keys from the KMS and transfers them to the ESXi hosts. It does not store or persist the KMS keys, but it keeps a list of key IDs. The vCenter Server system checks the privileges of users who perform *cryptographic operations*. VMware vSphere Web Client assigns cryptographic operation privileges and limits the users who can perform these operations. The vCenter Server system adds cryptography events to the list of events that can be viewed and exported from the vSphere Web Client event console. Each event includes the user, time, key ID, and cryptographic operation.

ESXi Hosts

The ESXi host is responsible for several aspects of the encryption workflow:

- Performs the encryption of VM disks
- Ensures that guest data for encrypted VMs is not sent over the network without encryption

Encryption per se is performed by the industry-standard OpenSSL libraries and algorithms described in the following section. VM encryption does not impose any new hardware requirements, but using a processor that supports the AES-NI instruction set accelerates encryption and decryption operations.

Virtual Machine Keys

Two types of keys are used for VM encryption:

- Data encryption key (DEK): The ESXi host generates and uses internal keys to encrypt VMs and disks. These XTS-AES-256 keys are used as DEKs.
- Key encryption key (KEK): The vCenter Server instance requests AES-256 keys from the KMS. vCenter Server stores only the ID of each KEK, but not the key itself.

The vCenter Server system transfers VM KEKs to an ESXi host when the host requires a key. The ESXi host uses the KEK to encrypt the DEK, and it stores the encrypted internal key on disk. The ESXi host does not store the KEK on disk. If a host reboots, the vCenter Server instance requests the KEK with the corresponding ID from the KMS and makes it available to the ESXi host. The ESXi host can then decrypt the DEKs as needed. Figure 1 illustrates the roles of the KEKs and DEKs.

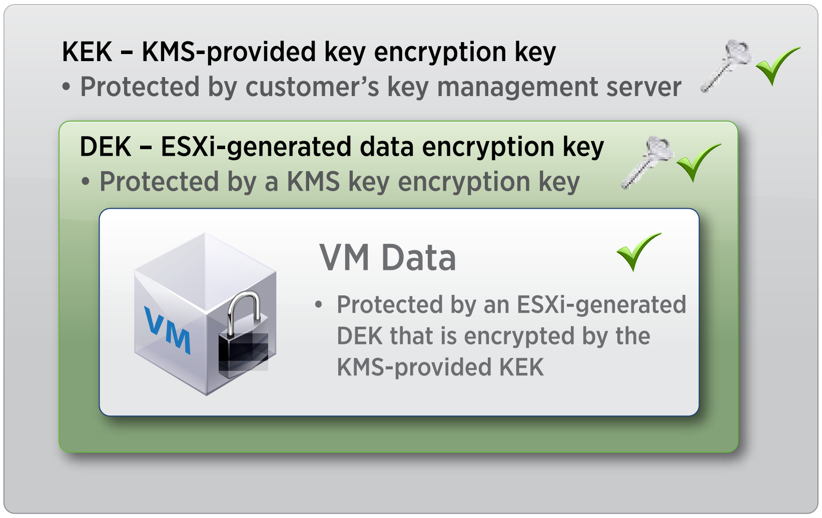
KEK – KMS-provided key encryption key
• Protected by customer's key management server
DEK – ESXi-generated data encryption key
• Protected by a KMS key encryption key
VM Data
• Protected by an ESXi-generated DEK that is encrypted by the KMS-provided KEK
VM

Figure 1. Roles of Data Encryption Key and Key Encryption Key

Roles and Permissions

A new set of permissions has been added to vCenter Server to enable fine-grained permissions regarding VM encryption. These permissions fall under the banner of cryptographic operations. They enable the creation of custom roles that meet the business and security needs of a data center. For example, a custom role can be created to enable an administrator to encrypt a VM but not decrypt it.

No Cryptography Administrator

A new predefined role, "no cryptography administrator," supports all administrator privileges except for the cryptographic operations privileges. This role enables standard administrator tasks such as powering on or migrating a VM via VMware vSphere vMotion®. The following operations are not enabled:

- Encrypting a VM
- Decrypting a VM
- Allowing console access to an encrypted VM
- Downloading an encrypted VM
- Configuring key managers

Least Privilege

These permissions and the new "no cryptography administrator" role align with the industry movement toward *least privilege management*, which gives users only the privileges required to perform their roles.

The standard vCenter Server “administrator” role has all privileges, **including encryption and decryption of VMs and configuration of key managers.** Those adopting VM encryption are strongly encouraged to review which individuals have elevated privileges and whether they can be assigned roles with few capabilities.

Host Encryption Mode

Encrypted VMs can be encrypted or run only if host encryption mode is enabled for the ESXi host. Host encryption mode is often enabled automatically, but it also can be enabled explicitly. Users can check and explicitly set the current host encryption mode from the vSphere Web Client instance or by using the VMware vSphere Storage APIs.

Automatic changes occur when encryption operations attempt to enable host encryption mode—for example, when adding an encrypted VM to a host on which host encryption mode is not enabled. Encryption mode changes automatically to “enabled” when a user has the required privileges to complete the operation on the host.

Key Management

Setting up a trusted connection between the vCenter Server instance and a KMS is one element of initial configuration of vSphere VM encryption. The vCenter Server system can then retrieve keys from the KMS as needed.

Key Manager Availability

The biggest requirement of key management is availability. DNS provides a good comparison. Best practices discourage use of a single DNS server in an environment. A properly configured data center should have multiple replicating DNS servers in case something goes wrong. The same holds true for key management. Most modern KMS designs enable the use of replicating key managers.

KMS Cluster or Alias

In the context of vCenter Server, a KMS cluster or alias is essentially a list of replicating key manager FQDN or IP addresses grouped into a single namespace—“KMS Cluster,” for example. In this example, multiple tenants, each with their respective key management infrastructures, can be named “Tenant A KMS Cluster” and “Tenant B KMS Cluster.”

Policy-Based Enforcement

Despite the sophistication of vSphere VM encryption, the actual task of encrypting a VM is quite straightforward. Encryption is essentially a storage policy that is applied to a VM, as with any other storage policy. After the policy has been applied, the VM is automatically encrypted.

The encryption policy can be applied on the **Storage Policy** screens in the vSphere Web Client instance. It can also be applied programmatically, via the vSphere Storage APIs or vSphere PowerCLI for example. This latter method enables encryption operations that can be performed across many VMs simultaneously, regardless of their OS type. Because of this policy-based enforcement, automation of VM encryption is simple, and it is very easy to integrate it with any overall provisioning workflow.

Assurance and Attestation

Regarding encryption, two concepts apply. The first is *assurance*: knowing that only verified code is being run during encryption operations and that any hardware is configured correctly. The second is *attestation*: the process by which the assurance can be measured. Another key term is *remote attestation*: out-of-band assurance measurement. A remote system determines the level of trust.

With VM encryption, there is assurance of the device doing the encryption, in this case the ESXi hypervisor. This is accomplished in vSphere by enabling Secure Boot on the host that ensures that only digitally signed code is allowed to run. Remote attestation is another feature under consideration for the future. This will provide additional assurance that the hypervisor and underlying hardware have not been tampered with.

HBAs, in-guest, fabric, SEDs, and arrays do not normally have a common method that provides for this level of remote attestation and certainly not one that is easy to consume at scale.

Advantages of vSphere Virtual Machine Encryption

The key difference between VM encryption and all other traditional solutions is ease of management. When VMs are treated as objects that can have a policy applied to them, there is no need to manage them individually.

In addition, implementing encryption at the hypervisor eliminates the need for additional, specialized hardware to perform encryption. It also removes the need to bring into scope switches, drives, and all of the configuration requirements they introduce. This exponentially shrinks operational overhead and considerably reduces the chances of misconfiguration or abuse.

If a security solution is too difficult, methods will be discovered to circumvent that solution. Traditional encryption solutions are difficult and time consuming to set up and manage. They also don’t take into account the new rules that encryption at scale demands. These factors were the driving force behind the development of VM encryption in vSphere 6.5. It is easy to configure, easy to use, and also easy to report, so compliance objectives can be efficiently met.

Table 1 lists the major features that characterize encryption solutions and compares the abilities of the various methods.

Feature	VM Encryption	HBA	Switch	SED	In-Guest
Multitenancy	Yes	No	Maybe	No	Yes
Guest OS Agnostic	Yes	Yes	Yes	Yes	No
Software Policy-Based Enforcement	Yes	No	No	No	No
Automation	Yes	No	No	No	Maybe
Additional Hardware	No	Yes	Yes	Yes	Maybe
Datastore Independent	Yes	No	No	No	Yes
No Access to Encryption Keys by the Guest	Yes	Yes	Yes	Yes	No

Table 1. Encryption Comparison Table

Conclusion

New technologies bring new rules. Encryption of data has a long history of being difficult to manage. Previous technologies struggle to keep up with the ever-increasing scale and the unique challenges that scale brings. Applying old rules to new technologies negatively impacts the ROI of these technologies. The increased burden of configuration, management, and attestation of hardware-based solutions hinders IT operations, security, and compliance.

With VMware vSphere, VMware has built one of the most secure and robust virtualization platforms. Encryption is now available via software policies that are independent of the operating system and applications while still maintaining operational efficiencies inherent in the vSphere platform, all while preserving the security of the virtual machine.

VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com