Defending Cloud Data with Infrastructure Encryption

Version 1.0

Released: July 12, 2013

Securosis, L.L.C. http://securosis.com

Author’s Note

The content in this report was developed independently of any sponsors. It is based on material originally posted on the Securosis blog but has been enhanced and professionally edited.

Special thanks to Chris Pepper for editing and content support.

Licensed by SafeNet

SafeNet is a leading global provider of data protection. For 30 years, Fortune 500 global corporations and government agencies have turned to SafeNet to secure and protect their most valuable data assets and intellectual property. SafeNet's data-centric approach focuses on the protection of high value information throughout its lifecycle, from the data center to the cloud-enabling customers to adapt to the escalating internal and external threats, and rapidly evolve to address new business requirements and compliance mandates. More than 25,000 customers across commercial enterprises and government agencies trust SafeNet to protect and control access to sensitive data, manage risk, ensure compliance, and secure virtual and cloud environments.

Licensed by Thales e-Security

Thales e-Security is a leading global provider of data protection solutions – delivering high assurance data encryption and key management solutions to the financial services, manufacturing, government, retail, healthcare, and technology sectors. The company has a 40-year track record of protecting sensitive corporate and government information across a wide range of technology areas including PKI, credential management, payment processing, network encryption, and many more. Thales e-Security solutions reduce the cost and complexity associated with the use of cryptography in today’s traditional, virtualized, and cloud-based infrastructures, helping organizations reduce risk, demonstrate compliance, enhance agility, and pursue strategic goals with greater confidence. The company is represented in over 90 countries around the world. For more information, visit www.thales-esecurity.com.

Copyright

Table of Contents

Introduction

Infrastructure as a Service (IaaS) is often thought of as merely a more efficient (outsourced) version of traditional infrastructure. On the surface we still manage things that look like traditional virtualized networks, computers, and storage. We ‘boot’ computers (launch instances), assign IP addresses, and connect (virtual) hard drives. But while the *presentation* of IaaS resembles traditional infrastructure, the reality underneath is decidedly not business as usual.

For both public and private clouds, the architecture of the physical infrastructure that comprises the cloud — as well as the connectivity and abstraction components used to provide it — dramatically alter how we need to manage security. The cloud is not inherently *more* or *less* secure than traditional infrastructure, but it is very *different*.

Protecting data in the cloud is a top priority for most organizations as they adopt cloud computing. In some cases this is due to moving onto a public cloud, with the standard concerns any time you allow someone else to access or hold your data. But private clouds pose the same risks, even if they don’t trigger the same gut reaction as outsourcing.

This paper will dig into ways to protect data stored in and used with Infrastructure as a Service. There are a few options, but we will show why the answer almost always comes down to encryption in the end — with a few twists.

What Is IaaS Storage?

Infrastructure as a Service includes two primary storage models:

- **Object storage** is a file repository. This higher-latency storage has looser performance requirements and is used to store individual files (‘objects’). Examples include Amazon S3 and RackSpace Cloud Files for public clouds, as well as OpenStack Swift for private clouds. Object storage is accessed through an API rather than a network share, which opens up a wealth of new uses. To make things easier you can generally layer a familiar file browsing interface on top of the API.
- **Volume storage** is a virtual hard drive. These higher-performing volumes attach to virtual machines for use just like a physical hard drive or array. Examples include VMware VMFS, Amazon EBS, RackSpace RAID, and OpenStack Cinder.

To (over)simplify, object storage replaces file servers and volume storage substitutes for hard drives. In both cases the cloud takes a storage pool — which could be anything from a SAN to hard drives in individual servers — and adds abstraction and management layers. There are other kinds of cloud storage, such as cloud databases, but they fall under either Platform as a Service (PaaS) or Software as a Service (SaaS). For this IaaS paper we will stick to object and volume storage.

Due to the design of Infrastructure as a Service, data storage is very different than ‘regular’ file repositories and databases. There are substantial advantages such as resilience, elasticity, and flexibility — as well as new risks in areas such as management, transparency, segregation, and isolation.

How IaaS Is Different

We will cover the technical details in the next section, but at a high level:

In **private cloud** infrastructure our data is co-mingled extensively, and the physical locations of data are opaque. You cannot point to a single server and say, "there are my credit card numbers" any more. Often you *can* set things up that way — thereby sacrificing the benefits of cloud computing.

Any given piece of data may be located in multiple physical systems or even storage types. Part of the file might be on a server, some of it in a SAN, and the rest in a NAS, but it all behaves as if it's in a single place. Your sensitive customer data might be on the same hard drive that, through layers of abstraction, also supports an unsecured development system. Plan incorrectly and your entire infrastructure can land in your PCI assessment scope — all mixed together at a physical level.

To top it off, infrastructure is now managed by a web-based API that, if not properly secured, could allow someone on the other side of the planet unfettered access to virtual data center.

We are serious advocates of cloud computing, but we are also security guys. It is our job to help you identify and mitigate risks, and we generally let infrastructure experts tell you why you should use IaaS in the first place.

Public cloud infrastructure brings the same risks as private clouds, with additional complications because you no longer control 'your' infrastructure, your data might be mingled with everyone else's, and you lose most visibility into who (at your cloud provider) can access your data.

Whether private or public, you need to adjust security controls to handle full abstraction of resources. You cannot rely on knowing where network cables plug into boxes any more.

Here are a few examples of how life changes:

- In private clouds, any virtual system that connects to any physical system holding credit card account numbers is within the scope of PCI assessment. If you run an application that collects credit cards in the same cloud as one that holds unsecured internal business systems, both are within assessment scope — unless you take precautions which we will discuss later.
- In public clouds an administrator at your cloud provider *could* access your virtual hard drives. This would violate all sorts of policies and contracts, but it is still technically possible.
- In most IaaS clouds a single command or API call can make an instant copy (snapshot) of an entire virtual hard drive, and then move it around your environment or make it public on the Internet.
- If your data is on the same hard drive as a criminal organization using the same cloud provider, and 'their' hardware is seized as part of an investigation, your data could be exposed. Yes, this has happened.

It comes down to less visibility below the abstraction layer, and data from multiple tenants mixed on the same physical infrastructure. This is all manageable — it's just different.

Most of what we need to do for security is to use encryption and other techniques to either restore this visibility, or eliminate the need for it entirely.

How IaaS Storage Works

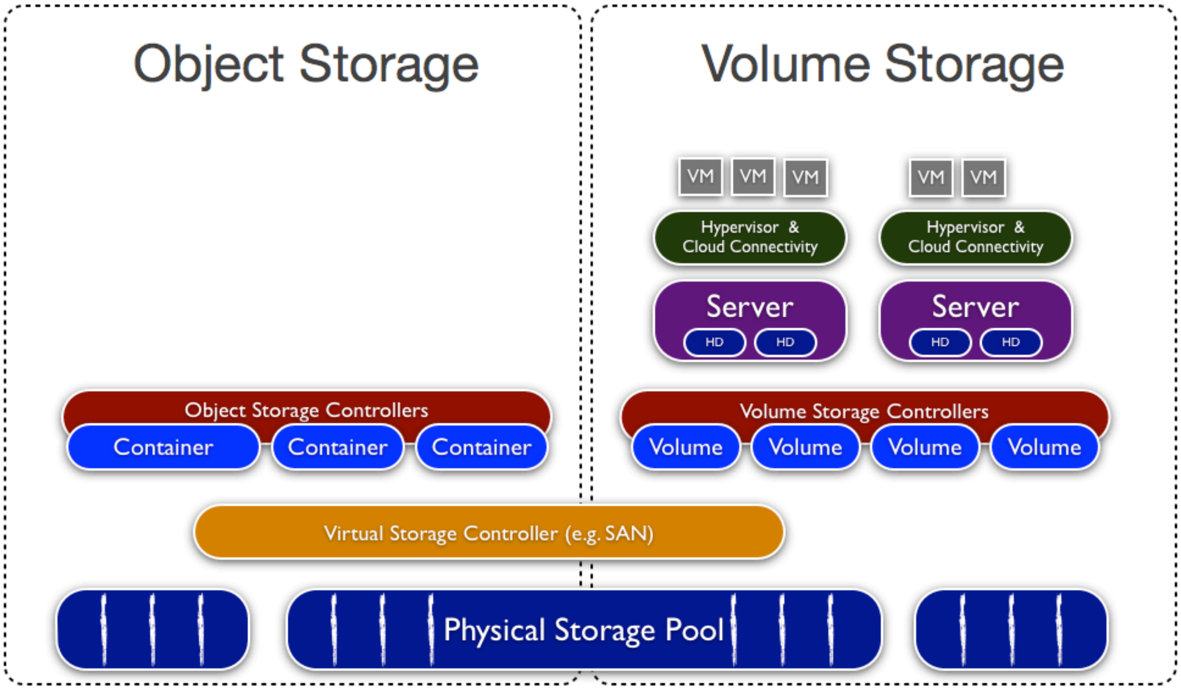
Infrastructure as a Service storage can be ridiculously complex when you include operational and performance requirements. First you need to create a resource pool, which might itself be a pool of virtualized and abstracted storage, and then you need to tie it all together with orchestration to support the dynamic requirements of the cloud — such as moving running virtual machines between servers, instantly snapshotting multi-terabyte virtual drives, and other things which were often simply impossible pre-cloud.

For security we don't need to know all the ins and outs of cloud storage, but we do need to understand the high-level architecture and how it affects our security controls. And keep in mind that implementations of our generic architecture vary widely between different public and private cloud platforms.

Public clouds are roughly equivalent to provider-class private clouds, except that they are designed to support multiple external tenants. We will focus on private cloud storage architectures with the understanding that public clouds are essentially the same except that customers have less control.

IaaS Storage Overview

Here is a diagram for us to walk through:


Object Storage
Volume Storage
VM
VM
VM
VM
VM
Hypervisor & Cloud Connectivity
Hypervisor & Cloud Connectivity
Server
HD
HD
Server
HD
HD
Object Storage Controllers
Container
Container
Container
Volume Storage Controllers
Volume
Volume
Volume
Volume
Virtual Storage Controller (e.g. SAN)
Physical Storage Pool

- At the lowest level is *physical storage*. This can be nearly anything that satisfies the cloud's performance and storage requirements. It might be commodity hard drives in commodity rack servers or high-performance SSD drives in high-end datacenter servers. But ultimately it could be nearly any storage appliance/system you can think of.
- Physical storage is generally pooled by a *virtual storage controller,* like a SAN. This is extremely common in production clouds but isn't limited to traditional SAN. Basically, as long as you can connect it to the cloud storage manager, you can use it. You could even dedicate certain LUNs from a larger shared SAN to cloud storage, while allocating other LUNs to non-cloud applications. If you aren't a storage person just remember there might be some sort of controller/server above the hard drives, outside your cloud servers, which needs to be secured.

On top of the physical storage we build object and volume storage.

Object Storage

- **Object storage controllers** (also called managers) connect to assigned physical or virtual storage and manage orchestration and connectivity. Managers are accessed through APIs. *Some* deployments include object storage connectivity software running on distributed commodity servers, to tie servers' hard drives into the storage pool.
- Object storage controllers create virtual *containers* (also called *buckets*) which are assigned to cloud users. A container is a pool of storage in which you can place objects (files). Every container stores each bit in multiple locations. This is called *data dispersion,* and we will discuss it in a moment.

Object storage is a cross between a database and a file share. You move files into and out of it, but instead of access through a file system you manage it with APIs — at an abstracted layer *above* whatever file systems actually store the data. Object storage APIs are almost always RESTful HTTP APIs, rather than classic network file protocols, which provides tremendous flexibility for integration into different applications and services. Object storage logic below the user-accessible layer manages features such as quotas, access control, and redundancy.

Volume Storage

- **Volume storage controllers** (also called managers) connect to assigned physical (or virtual) storage to manage orchestration and connectivity. Managers are accessed through APIs. The controller creates volumes on request and assigns them to specific cloud instances. In traditional virtualization terms it creates a virtual hard drive and connects it to a virtual machine, usually over the network. *Data dispersion* is often used to provide redundancy and robustness.
- A **volume** is essentially a virtual hard drive. It can be any size supported by the cloud platform and underlying resources, and a volume assigned to a virtual machine exists until it is destroyed — but note that terminating an instance often automatically deallocates its volumes and returns their capacity to the free storage pool.
- Physical servers run *hypervisors* and *cloud connectivity* software to tie them into the compute resource pool. This is where instances (virtual machines) run. They typically have local hard drives which can be assigned to the volume controller to expand the storage pool, or *used locally for non-persistent storage*. We call this 'ephemeral' storage, and it is great for swap files and other higher-performance data that doesn't require the resiliency of a full storage volume. If your cloud uses this model its cloud management software places swap on these local drives. When you move or shut down your instance this data is always lost, although it might be recoverable until overwritten.

We like to discuss volumes in terms of virtual hard drives but they are a bit more complex. Volumes may be distributed, with data dispersed across multiple physical drives. They are connected to virtual machines over the network (except for ephemeral storage). Next we will consider the implications for volumes and how they interact with object storage and things like snapshots and live migrations.

How Object and Volume Storage Interact

Most clouds include both object and volume storage, even if object storage isn't available directly to users. Here are the key uses:

- A *snapshot* is an effectively instant backup of a volume, which is moved into object storage. The underlying technology varies widely and is too complex for my feeble analyst brain, but a snapshot effectively copies a complete set of the storage blocks in your volume into a file in an object container which has been allocated for snapshots. Generally every block in your volumes is stored in multiple physical locations, typically 3 or more times, so taking a snapshot tells the volume controller to copy a complete set of blocks over to object storage. The operation may take a while to complete but *looks* instantaneous because the snapshot accurately reflects the state of the volume at the *instant* it was triggered, and the volume remains fully usable — running from another set of blocks while the snapshot is moved over (this is a **gross** oversimplification of something that makes my head hurt).
- *Images* are pre-defined storage volumes in object storage, which contain operating systems and other virtual hard drives used to launch instances. An image could be a base version of Windows or a completely configured server in an n-tier application stack. When you launch an instance the volume controller creates a volume of the required size, then copies the requested image from the object controller into the virtual machine.
- Because snapshots and images are objects just like normal files in object storage, they are very portable and (in public clouds) can be made available to the Internet with a single API call or mouse click.
- You can quickly create images from running instances. These images contain everything stored "on disk" unless you deliberately exclude particular locations such as swap files.

Understanding these components is essential for securing cloud resources. A snapshot is a near-instant backup of a (virtual) hard drive that is incredibly portable and easily made public. A few years ago I co-wrote a script that, if run on a cloud administrator's computer, would snapshot every single volume that administrator could access and make the snapshots public. With a nice metadata tag to make them easy to find. A few API calls from an unprotected developer or administrator system could expose all the data in your cloud.

Also, if you allow instances to store data in local ephemeral storage, sensitive data such as encryption keys may be left behind when you move or terminate an instance.

- **Data dispersion** is comparable to RAID protection but implemented differently. Any storage block is replicated in multiple physical locations across your cloud. In private clouds you configure this yourself but in public clouds it is generally an opaque feature. Dispersion is great for resiliency and valuable for security — any given file might be broken up and stored on multiple hard drives. On the bright side losing one drive might not matter much, but on the other hand you can rarely figure out exactly what data is stored on which drives.

Cloud Storage Networks

All this runs on multiple networks at least on cloud built for performance and reliability. Common networks include:

- If you use virtual storage (*e.g.,* SAN) this likely runs over its own storage network.
- A management network ties together the cloud controller components — particularly object and volume managers and agents.
- A data/storage network for connecting volumes to instances, to improve performance. This may also connect object and volume storage.
- The external public network for managing cloud controllers via API.
- A service network for communicating outside clients to instances, as well as between instances.

You will likely have at least one network to the outside world, a private network for storage (between volumes and instances), and another for management.

Some or all of these logical networks might run over the same physical wiring, segregated with VLANs, but consider how much you can trust VLANs shared with unknown parties running their own operating systems (you shouldn't). Lastly, these networks might violate your expectations for networks due to new physical platforms for cloud hosting, which may run storage and communications traffic over the same physical links.

We aren't trying to scare you — the ins and outs of designing and securing these networks are fodder for another day — but you need to be aware of what is under the surface.

Conclusion

The architecture and resiliency of cloud storage models create new and interesting risks:

- Cloud administrators, in either your environment or your cloud provider's, can access any data stored in the cloud over the network. This is very different than traditional infrastructure where storage access typically requires physical connectivity.
- Snapshots become ubiquitous because they are effectively instantaneous, highly portable, and accessible over the network. They pose a significantly increased risk of exposure compared to traditional infrastructure — where snapshots are less common, less portable, and less exposed.
- Images of instances may contain and expose sensitive data.
- All this is managed with networks and APIs, which remove some of our traditional security controls and expectations. Someone accessing a cloud administrator's or developer's system could, depending on how things are set up, access literally an entire (virtual) datacenter.
- Cloud data can be incredibly resilient, with any given bit stored in multiple places across the cloud.
- You may have 3 or more networks to secure (for storage) and segregate. Don't trust VLANs as a security control, since there are techniques to hop VLANs if you are on the physical network.
- There is very little visibility into where things are actually stored, although some cloud platforms are beginning to offer more transparency — this is an evolving area.
- You still have physical and virtual storage to keep secure, underneath everything else.

Due to all this complexity and portability, *encryption* is the best tool available for most cloud data security. Implemented properly, encryption protects data as it moves through your environment. It doesn't matter if there are 3 versions of a particular block exposed on multiple hard drives because without the key they are *all* meaningless. It doesn't matter if someone makes a snapshot of an encrypted volume public. Only exposure of data *and* its associated keys is problematic.

Of course encryption cannot wipe all security issues away. As we will discuss, not all products or approaches work in all situations (especially for boot volumes), and data on unencrypted volumes is still exposed. But in combination with our other recommendations, encryption enables you to store and process even sensitive data in the cloud.

Understanding Encryption Systems

Now that we have covered the basics of how IaaS platforms store data, we need to review the relevant parts of an encryption system for protecting cloud data. Encryption isn't our only security tool, as mentioned in the previous section, but it is one of the only *practical* data-specific tools available for cloud computing.

The Three Components of a Data Encryption System

Cryptographic algorithms and implementation specifics are important at the micro level, but when designing encryption for cloud computing or anything else, the overall structure of the cryptographic system is just as important. There are many resources on which algorithms to select and how to use them, but far less guidance on how to piece together an entire system.

When encrypting data in the cloud knowing how and where to place these pieces is incredibly important, and one of the most common causes of failure. In a multi-tenant environment — even in a private cloud — with almost zero barriers to portability, we need to pay particular attention to where we manage keys.

Three major components define the overall structure of an encryption system:

- **The data:** The object or objects to encrypt. It may seem silly to break this out but the security and complexity of the system are dependent on the payload's nature, as well as where it is located or collected.
- **The encryption engine:** This component handles actual encryption and decryption operations.
- **The key manager:** This handles keys and passes them to the encryption engine as authorized.

Data
Encryption Engine
Key Management

In a basic encryption system all three components are likely to be located on the same system. As an example consider personal full disk encryption (the built-in tools you might use on your home PC or Mac): the encryption key, data, and engine are all stored and used on the same hardware. Lose that hardware and you lose the key, data, and engine. But the engine is simply a standard component of the operating system, and the key is protected with another key that is not stored on the system — but if the system is lost while running with the key in memory, that is a problem.

In a traditional application we normally break the components out — with the encryption engine in an application server, the data in a database, and key management in an external service or appliance.

Some limitations and facilities of cloud computing drive encryption into certain architectural models:

- One risk to protect against is a rogue cloud administrator, or anyone with administrative access to the infrastructure, seeing your data. So we have fewer options for *where to securely manage keys* since the keys must be unavailable to those administrators.
- Data is *much more portable* than in traditional infrastructure, thanks to native storage redundancy and data management tools such as snapshots.
- Encryption engines may run on shared resources with other tenants, so they may need special techniques to protect keys in RAM, or you may need to alter your architecture to reduce risk.
- Automation dramatically impacts your architecture — you might have 20 instances of a server spin up at the same time, then go away. Provisioning of storage and keys must be as dynamic and elastic as the underlying cloud application itself.
- Automation also means you may manage many more keys than in a traditional (more static) application environment.

As you will see when we work through the details, we leverage the separation of these components in a few different ways to compensate for security risks in the cloud. Fortunately the end result is likely to be *more* secure than traditional infrastructure and application architectures.

Protecting Instances and Volume Storage

Now that we have gotten through all the pesky background, we can start delving into the best ways to protect data.

Securing the Storage Management Plane and Infrastructure

Your first step is to lock down the management plane and infrastructure of your cloud storage. Encryption can compensate for many configuration errors and defend against many management plane attacks but you cannot afford to skip the basics. Also, depending on which encryption architecture you select, a poorly-secured cloud deployment could bypass all those nice crypto benefits by giving away too much access to portions of your encryption implementation.

We are focusing on data protection so we don't have space to cover all the ins and outs of management plane security, but here are some areas to be aware of:

- **Limit administrative access:** Even if you trust all your developers and administrators completely, all it takes is one vulnerability on one workstation to compromise everything you have in the cloud. Use access controls and tiered accounts to limit administrative access as you do for other systems. For example, restrict snapshot privileges to a few designated accounts, and then restrict them from otherwise managing instances. Integrate all this into your privileged user management.
- **Compartmentalize:** We all know where flat networks get you, and the same goes for flat clouds. But cloud segregation occurs at the management plane level. Group systems and servers and limit management access to those resources. An administrative account for development systems shouldn't also be able to spin up or terminate instances in production accounting systems.
- **Lock down the storage architecture:** All clouds still run on physical systems. If you are running a private cloud make sure you keep everything up to date and configured securely.
- **Audit:** Keep audit logs (if your platform or provider supports them) of management-plane activities including starting instances, creating snapshots, and altering security groups.
- **Secure snapshot repositories:** Snapshots normally end up in object storage so follow all your object storage rules (below) to keep them safe. In private clouds snapshot storage should be separate from the object storage used to support users and applications.
- **Alerts:** For highly sensitive applications — depending on your cloud platform — you may be able to generate alerts when snapshots are created, new instances are launched from particular instances, etc. This isn't typically available out of the box but shouldn't be hard to script, and may be provided by an intermediary cloud broker service or platform if you use one.

- **Zones:** For compliance use zones to cluster similarly-regulated data on the same physical hardware. Most cloud platforms support creating zones: groups of hardware and cloud resources. These enables both physical and virtual segregation, managed by policy.

There is much more to locking down a management plane but start with a focus on limiting administrative access, segregating your environment at the cloud level with groups and good account privileges, and locking down the back-end storage architecture.

Encrypting Entire Volumes

As a reminder, volume encryption protects from the following risks:

- Protects volumes from snapshot cloning/exposure
- Protects volumes from examination by the cloud provider, including cloud administrators
- Protects volumes from being exposed by physical drive loss (more for compliance than a real-world security issue)

IaaS volumes can be encrypted three ways:

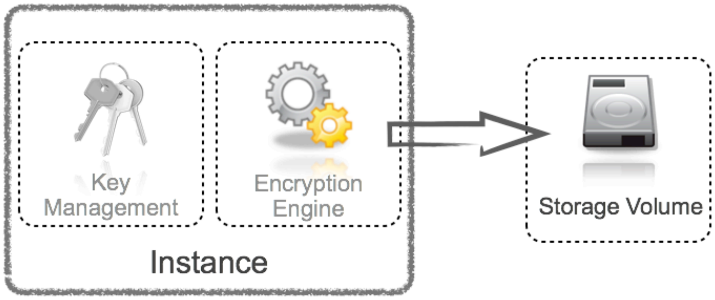
- **Instance-managed encryption:** The encryption engine runs within the instance and keys are stored in the volume protected by a passphrase or keypair.
- **Externally-managed encryption:** The encryption engine runs in the instance but keys are managed externally and issued to instances on request.
- **Proxy encryption:** You connect the volume to a special instance or appliance/software to perform the encryption operations; then you connect the application instance to the proxy for the unencrypted data. The proxy handles all cryptographic operations and may keep keys either onboard or externally.

Instance-managed Encryption

This is the least secure and manageable option; it is generally only suitable for development environments, test instances, and other situations where long-term manageability isn't a concern.

- The encryption engine runs inside the instance. Examples include TrueCrypt and the Linux `dm-crypt` tool.
- You connect a second *new* storage volume.
- You log into your instance, and using the encryption engine you encrypt the new storage volume. Everything is inside the instance except the raw storage, so you use a passphrase, file-based key, or digital certificate for the key.
- You can also use this technique with a tool like TrueCrypt and create and mount a storage volume that's really just a large encrypted file on your boot volume.

Any data stored on the encrypted volume is protected from being read directly (for instance if a physical drive is lost or a cloud administrator tries to access your files using their native API), but is accessible from the logged-in instance while the encrypted volume is mounted. This protects you from many cloud administrators, because only someone with actual *access to log into your instance* can see its data, and most cloud administrators lack the credentials to do this.



This option also protects data in snapshots. Better yet, you can snapshot a volume and then connect it to a different instance with the key or passphrase. Instance-managed encryption works well for public and private clouds.

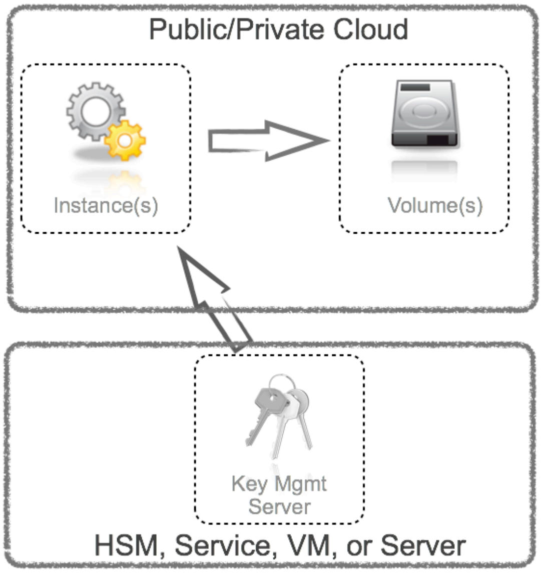
Unfortunately this approach is completely unmanageable. The only moderately secure option is to use a passphrase when you mount the encrypted volume, which requires manual intervention every time you reboot an instance or connect it (or a snapshot) to a different instance. For security reasons you can't store the key (or passphrase) in a file in the instance or use a stored digital certificate, because *anything stored on the unencrypted boot volume of the instance is exposed*.

This is fine for test and development, or to exchange data volumes with someone else, but should otherwise be avoided.

Externally-managed Encryption

Externally-managed encryption is similar to instance-managed except that the keys are handled outside the instance in a key management server, service, or Hardware Security Module (HSM). *This is the preferred option for most cloud deployments.*

With this option the encryption engine (typically a client/agent for whatever key management tool you are using) connects to an external key manager or HSM. The key is provided subject to the key manager's security checks, and then used by the engine or client to access the storage volume. The key is never stored on disk in the instance, so its only exposure is in RAM (including memory snapshots). Many products reduce this exposure by overwriting keys in memory when they aren't in active use, or using other memory protection techniques.



As with instance-managed encryption, storage volumes and snapshots are protected from cloud administrators. But an external key manager offers a variety of new benefits, including:

- This option supports reboots, autoscaling, and other cloud operations that instance-managed encryption cannot. The key manager can perform additional security checks — which may be quite deep — to ensure only approved instances access keys. It can then provide keys automatically or alert a security administrator for quick approval.
- Auditing and reporting are centralized, which is essential for security and compliance.
- Keys are centrally managed and stored, which dramatically improves manageability and resiliency at enterprise scale.
- Externally-managed encryption supports a wide range of deployment options, such as hybrid clouds and even managing keys for multiple clouds.
- This approach works well for both public and private clouds.
- Some products support encryption of boot volumes, similar to laptop Full Disk Encryption (FDE). This isn't currently possible with other volume encryption options.

Of course there are downsides as well, including:

- The capital investment is greater — you need a key management server, subscription service, or HSM, and a compatible encryption engine.

- You must install and maintain a key management server or HSM that is accessible to your cloud infrastructure, or accept lock-in to a service provider.
- You need to ensure your key manager/HSM will scale with your cloud usage. This is not a question of how many keys it can store, but of how well it performs in or when connected to a cloud — likely with high network latency.

This is often the best option for encrypting volume storage, but there are many deployment and feature options to consider.

Deployment and feature options

The first question is how to deploy external key management. There are four options:

- *An HSM or other hardware key management appliance*. This provides the greatest physical security, but the appliance must be deployed outside the cloud. When using a public cloud this means running the key manager internally, relying on a virtual private cloud, and connecting the two with a VPN. In private clouds you run it somewhere on the network near your cloud, which is much easier.
- *A key management virtual appliance*. Your vendor provides a pre-configured virtual appliance (instance) for you to run in your private cloud. We do not recommend you run this in a public cloud because of the much greater exposure to live memory exploitation and loss of keys — even if the instance is encrypted. If you decide to go this route anyway, use a vendor who takes exceptional memory protection precautions. A virtual appliance doesn't offer the same physical security as a physical server, but they come hardened and support more flexible deployment options so they can run within your cloud.
- *Key management software*, which can run either on a dedicated server or on an instance within the cloud. The difference between software and a virtual appliance is that you install the software yourself rather than receiving a configured and hardened image. Otherwise the risks and benefits are the same, assuming you harden the server (instance) as well as the virtual appliance.
- *Key management Software as a Service (SaaS)*. Multiple vendors now offer key management as a service, specifically to support public cloud encryption. This also works for other kinds of encryption, including private clouds, but most usage is for public clouds. There are a few different deployment topologies, which we will discuss in a moment.

When deploying a key manager in a cloud there are a few wrinkles to consider. The first is that if you have hardware security requirements, your only option is to deploy a HSM or encryption/key management appliance compatible with your cloud computing requirements — for instance, your cloud is likely to have many more dynamic network connections than a traditional network (note that raw key operations per second is rarely the limiting factor). This can be on-premise with your private cloud or remote with a VPN connection to the virtual private cloud. It could also be offered as a service by your cloud provider in their data center, with native cloud management APIs. Another option is to store the root key on your own hardware but deploy a bastion provisioning and management server as a cloud instance. This server handles communication with encryption clients/agents and orchestrates key exchanges, but does not require the actual keys to be stored within the cloud.

If you don't have hardware security requirements a number of additional options open up. Hardware is often required for compliance but isn't always necessary.

Virtual appliances and software servers are fairly self-explanatory. The key issue (no pun intended) is that you are likely to need additional synchronization and orchestration to handle multiple virtual appliances in different zones and clouds. We will talk about this more under features.

As with hardware appliances, some key management service providers also deploy a local instance to assist with key provisioning (this is provider dependent and not always necessary). In other cases the agents communicate directly with the cloud provider over the Internet. A final option is for the security provider to partner with the cloud provider and install some components within their cloud to improve performance, enhance resilience, and/or reduce Internet traffic — which cloud providers charge for.

To choose an appropriate topology answer the following questions:

- Do you need hardware-level key security?
- How many instances and key operations will you need to support?
- What is the topology of your cloud deployment? Public or private? Zones?
- What degree of separation of duties and keys do you need?
- Are you willing to work with a key management service provider?

Cloud Features

For a full overview of key management servers, see our paper Understanding and Selecting a Key Management Solution. Rather than regurgitate an 18-page paper we will focus on a few cloud-specific requirements we haven't otherwise covered yet.

- If you use any kind of key management service, *pay particular attention to how keys are segregated and isolated between cloud consumers and from service administrators*. Different providers have different architectures and technologies to manage this, and you should map your security requirements agains how they manage keys. In some cases you might be okay with a provider being technically able to get your keys, while in other cases that is completely unacceptable. Ask for technical details of how they manage key isolation and their root of trust.
- Even if you deploy your own encryption system *you need granular isolation and segregation of keys to support cloud automation*. For example if a business unit or development team is spinning up and shutting down instances dynamically, you will likely want to enable them to manage some of their own keys without exposing the rest of the organization.
- Cloud infrastructure is more dynamic than traditional infrastructure, and relies more on APIs and network connectivity — so you are likely to have more network connections from a greater number of instances. *Any cloud encryption tool should support APIs and a high number of concurrent network connections for key provisioning*.
- *For volume encryption, look for native clients/agents designed to work with your specific cloud platform*. These are often able to provide information above and beyond standard encryption agents to ensure only acceptable instances access keys. For example they might provide instance identifiers, location information, and other indicators which do not exist on physical hardware. When available, you might use these identifiers to only allow access to encrypted storage from instances located in the correct availability zone, to verify that an authorized user launched the instance, etc. They can also accommodate the peculiarities of IaaS storage more smoothly. For boot volume encryption this is mandatory.
- Cloud-specific management and reporting enables you to better manage keys for the cloud and manually provision keys as needed. *The encryption tool should report instance-level details of key provisioning*, such as instance and zone identifiers. This information is critical for manual provisioning or approval of key releases to ensure someone doesn't just clone an instance, modify it, and then use it to steal keys.
- *Cloud encryption agents should pay particular attention to minimizing key exposure in volatile memory (RAM)*. This is essential to reduce exposure of keys to cloud administrators and other tenants on the same physical server, depending on the memory protection features of the cloud platform.

These are merely the cloud-specific features to look for in addition to standard key management and encryption features.

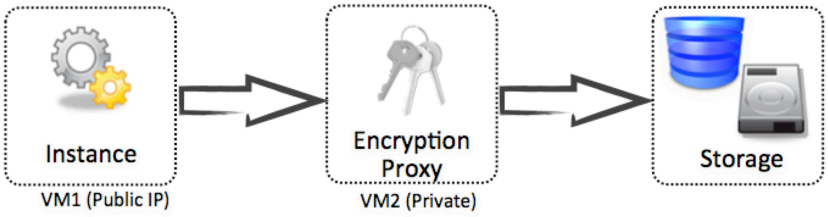
Proxy Encryption

The last encryption option uses an inline software encryption proxy to encrypt and decrypt data. This doesn’t work for boot volumes but may enable you to encrypt a wider range of storage types, and offers an alternate technical architecture for connections to external volumes.

The proxy is a virtual appliance running in the same zone as the instance accessing the data and the storage volume. We are talking about IaaS volumes in this section, so they will be our focus.

The storage volume attaches to the proxy, which performs all cryptographic operations. Keys can be managed in the proxy or stored in an external key manager. The proxy uses memory protection techniques to resist memory parsing attacks, and never stores unencrypted keys in its own persistent storage.

The instance accessing the data connects to the proxy using a network file system/sharing protocol such as iSCSI. Depending on the pieces used this could, for example, allow multiple instances to connect to a single encrypted storage volume.



Protecting Object Storage

Object storage — such as Amazon S3, OpenStack Swift, and Rackspace Cloud Files — is fairly straightforward to encrypt, with three options:

- Server-side encryption
- Client/agent encryption
- Proxy encryption

As with our earlier examples overall security is dependent on where you place the encryption agent, key management, and data. But first we need to address the two types of object storage. Plain object storage, as in our examples above, is accessed and managed only via APIs and forms the foundation of cloud data storage — although it may be built on top of a traditional SAN/NAS layer.

A number of popular cloud storage services such as Dropbox, Box.com, and Copy.com — as well as tools for building private internal systems — include basic object storage but layer on PaaS and SaaS features. Some of these even rely on Amazon, Rackspace, or other 'root' services to provide their underlying storage. The main difference is that these services add their own APIs and web interfaces, as well as native clients for various mobile and desktop operating systems.

Server-side Encryption

With this option all data is encrypted in storage by the cloud platform itself. The encryption engine, keys, and data are all within the cloud platform and managed by cloud administrators. This option is extremely common at many public cloud object storage providers, sometimes without additional cost.

Server-side encryption really only protects against a single threat: lost media. It is more a compliance tool than an actual security tool because the cloud administrators have the keys anyway. Server-side encryption can offer minimal additional security in private clouds but it fails to disrupt most of the real risks to data.

Server-side encryption offers no protection against cloud administrators, and depending on configuration it may provide little protection in case of management plane compromise.

Client/Agent Encryption

If you don't trust the storage environment your best option is to encrypt the data before sending it up. We call this *Virtual Private Storage* because, as with a Virtual Private Network, we turn a shared public resource into a private one by encrypting it while retaining the keys. The first way to do this is with an encryption agent on the host connecting to the cloud service.

This is architecturally equivalent to *externally-managed encryption* for storage volumes. You install a local agent to encrypt/decrypt the data before it moves into the cloud and manage the keys in an external appliance, service, or server. It is *possible* to manage keys locally, as with *instance-managed encryption*, but even less useful because object storage is normally accessed by multiple systems, so we *always* need access to keys from multiple locations.

The minimum architecture is comprised of encryption agents and a key management server. Agents implement the cloud's native object storage API and provide logical volumes or directories with decrypted access to the encrypted volume so applications are insulated from cloud storage and encryption APIs. This option is most often used with cloud storage and backup services rather than for direct access to raw object storage.

Some agents are advances on file/folder encryption, especially tools for services like Dropbox or Box.com which are accessed as normal directories on client systems. But stock agents need to be tuned to work with the specific platform in question… which is outside our scope.

Proxy Encryption

One of the best options for business-scale use of object storage, especially public object storage, is an inline or cloud-hosted proxy.

There are two main topologies:

- The proxy resides on your network, and all data access runs through it for encryption and decryption. The proxy uses the cloud's native object storage APIs.
- The proxy runs as a virtual appliance in either a public or private cloud.

There are two key management options: internal to the proxy or external. The usual deployment options are available: hardware/appliance, virtual appliance, or software.

Proxies are especially useful for object storage because they make it very easy to implement Virtual Private Storage. You route all approved connections through the proxy, which encrypts the data and then passes it on to the object storage service.

Object storage encryption proxies are evolving very quickly to meet user needs. For example, some tie into the Amazon Web Services Storage Gateway to keep some data local and some in the cloud for better performance. Others both proxy to the cloud storage service and offer a normal network file share for on-premise access.

How to Choose

There is no single way to pick the best encryption option. Which is 'best' depends on many factors — including the specifics of the cloud deployment, what is already available for key management and encryption, and the nature of the data. That said, here are some guidelines:

Volume Storage

Key Criteria

- *Always use external key management.* Instance-managed encryption is only acceptable for test/development systems you know will never go into production.
- For sensitive data in public clouds, choose a system with protection for keys in volatile memory (RAM). Don't use a cloud's native encryption capabilities if you have any concern about risk from cloud administrators.
- In private clouds you may also need a product that protects keys in memory if sensitive data is encrypted in instances which share physical hosts with untrusted instances that could attack memory.
- Pick a product designed to handle the greater dynamism of cloud computing environments. Specifically one with workflow for rapidly provisioning keys to cloud instances and API support for your cloud platform.
- The two key features to look for, after platform/topology support, are granular key management (role-based with good isolation/segregation) and good reporting.

Key Criteria
External key management
Keys protected in memory
Cloud and management API support
Granular key management

Situational Criteria

- If you need to encrypt boot volumes and not just attached storage volumes, select a product with a client that includes that capability for the operating systems you use in instances. On the other hand, don't assume you need boot volume support — that depends on how you architect cloud applications.
- Know your compliance requirements and use hardware (such as an HSM) if necessary for root key storage.
- Key management services may reduce the overhead of building your own key infrastructure if you are comfortable with how they handle key security. As cloud natives they may also offer other performance and management advantages, but this varies widely between products and cloud platforms/services.

If Needed
HSM for root key storage
Boot volume encryption support
Key Management as a Service

It is impossible to be more specific without details of your cloud deployment, but these questions should get you moving in the right direction. The main things to understand before you start looking for a product are:

1. What cloud platform(s) are we on?
2. Are we using public or private cloud, or both? Does our encryption need to be standardized between the two?
3. What operating systems will our instances run?
4. What are our compliance and reporting requirements?
5. Do we need boot volume encryption for instances? (Don't assume this — it isn't always a requirement).
6. Do root keys need to be stored in tamper-proof hardware? (Generally a compliance requirement — virtual appliances and software servers are actually quite secure).
7. What is our cloud and application topology? How often (and where) will we provision keys?

Object storage

- For server-based object storage, such as is used to back applications, *a cloud encryption gateway is likely your best option*. Use a system where you manage the keys rather than your cloud provider, and don't store those keys in the cloud.
- To support users on services like Dropbox, use a software client/agent with centralized key management. If you want to support mobile devices make sure your product supports your mobile platforms.

As you can see, object storage encryption is usually much simpler than volume storage.

Conclusion

Encryption is usually the best tool to protect cloud data. It allows us to separate security from cloud infrastructure without sacrificing the advantages of cloud computing. Splitting key management from data storage and encryption engines enables a wide variety of deployment options and use cases. We can now store data in multi-tenant systems and services without compromising security.

This paper focuses on protecting data in IaaS (Infrastructure as a Service) environments, but different encryption options — such as encrypting data when you collect it in an application — might be a better choice, or a complementary option for greater granularity.

Encrypting cloud data can be more complex than on traditional infrastructure, but once you understand the basics adapting your approach shouldn't be too difficult. The key is to reconsider how you encrypt and manage keys (assuming you even do) rather than simply attempting to replicate your traditional architecture. Understand how you use the cloud and adapt your approach so encryption becomes an enabler rather than an obstacle.

If you have any questions on this topic, or want to discuss your situation specifically, feel free to send us a note at info@securosis.com or ask via the Securosis Nexus (http://nexus.securosis.com/).

Who We Are

About the Author

Rich Mogull, Analyst and CEO

Rich has twenty years of experience in information security, physical security, and risk management. He specializes in data security, application security, emerging security technologies, and security management. Prior to founding Securosis, Rich was a Research Vice President at Gartner on the security team where he also served as research co-chair for the Gartner Security Summit. Prior to his seven years at Gartner, Rich worked as an independent consultant, web application developer, software development manager at the University of Colorado, and systems and network administrator. Rich is the Security Editor of TidBITS, a monthly columnist for Dark Reading, and a frequent contributor to publications ranging from Information Security Magazine to Macworld. He is a frequent industry speaker at events including the RSA Security Conference and DefCon, and has spoken on every continent except Antarctica (where he's happy to speak for free — assuming travel is covered).

About Securosis

Securosis, L.L.C. is an independent research and analysis firm dedicated to thought leadership, objectivity, and transparency. Our analysts have all held executive level positions and are dedicated to providing high-value, pragmatic advisory services.

We provide services in four main areas:

- Publishing and speaking: Including independent objective white papers, webcasts, and in-person presentations.
- Strategic consulting for end users: Including product selection assistance, technology and architecture strategy, education, security management evaluations, and risk assessments.
- Strategic consulting for vendors: Including market and product analysis and strategy, technology guidance, product evaluations, and merger and acquisition assessments.
- Investor consulting: Technical due diligence including product and market evaluations, available in conjunction with deep product assessments with our research partners.

Our clients range from stealth startups to some of the best known technology vendors and end users. Clients include large financial institutions, institutional investors, mid-sized enterprises, and major security vendors.

Securosis has partnered with security testing labs to provide unique product evaluations that combine in-depth technical analysis with high-level product, architecture, and market analysis.