<http://meship.com/Blog/2011/04/01/security-in-the-cloud-top-issues-in-building-users-trust/>

**Encryption is already used**

For instance you can have rotating keys and my favorite is private VPNs. If you have a good working security structure in place you can now use a private VPN from within your existing system to scale cloud resources without opening your system to the outside.

**Improved cloud encryption techniques are being researched**

The basic, and very powerful, idea is to apply encryption agents to every virtual computing instance. Thus, every VM would have its own resident manager to ensure the proper application of encryption security resources.

The big win here is you'd have, in essence, automated application of security policies everywhere. Thus, you'd have cryptographic key management built into the process and also no worry about unprotected VM instances among your computing resources.

**The key issue**

Management of cryptographic keys is by no means a trivial thing. When you think about it, all of your cloud security rests on being able to generate and hand out those keys, while keeping them out of the hands of bad guys. Hackers aren't going to be able to break your keys; what they'll do to breach your security is to steal them instead.

homomorphic encryption-it would allow you to send encrypted data throughout the cloud, manipulate it any way you want, and then at the end of the day, you'd still be able to decrypt it.

Currently, there are severe limitations on the operations you can perform on encrypted data, because some of the manipulations will muck it up so that it's no longer decryptable.

*Cloud Security tip #1:* START FROM THE DATA when considering the cloud provider. You should ask your cloud provider what mechanisms he has in place to secure your data, ask about data security best practice and which security vendors are supporting his cloud and can provide security and encryption services for you out of the gate.

[](http://www.porticor.com/wp-content/uploads/2011/09/dart_166x320.jpg)Data in a public cloud is more vulnerable to some attacks compared with data stored in a data center. New risk vectors exploiting the shared infrastructure and resources, the insider threat, or cloud-account hijacking exist. These are either new threats, or old threats that are qualitatively different for data in the cloud. Data encryption in the cloud mitigates many of these, and is therefore top priority. But encryption doesn’t come easy in a cloud infrastructure. Encryption in the cloud could be cost intensive and timely if not done effectively.

*Cloud Security tip #2:* Choose the data encryption strategy wisely. Understand how you would like to approach data encryption: Will you implement encryption using the tools provided by your cloud provider, or will you use a third party vendor? In order to do so you need to get to the details of your cloud technology, and your technical team capabilities. If you’re implementing a small server project in the cloud and have great security skills in-house, you might want to secure your virtual server yourself; but if the plan is to aggressively grow in the cloud, adding many servers and applications, a security vendor approach would prove to be more effective. (And if you’re a software vendor developing your application in the cloud – you’ll find my previous blog regarding [*cloud encryption and key management for software vendors*](http://www.porticor.com/2011/07/cloud-encryption-key-management-software-vendors/) *relevant)*

Now that you’ve chosen your cloud provider, and have a concrete encryption strategy in place, let’s talk about the encryption keys. How will you keep your encryption keys private in a public cloud? The common approach recommended by most security vendors is to keep your keys away from the cloud for security reasons. This approach is a result of migrating the traditional PKI infrastructure to the cloud as-is, and will force you to use a third party SaaS vendor to manage your keys, or to install a key management server back in your datacenter. But there’s a third option for key-management in the cloud:

*Cloud Security tip #3:* Be the master of your (cloud) domain. As mentioned above, the current approach is to store the encryption keys as far as possible from your cloud, meaning you will need to invest additional resources and manage yet another system in or outside your datacenter. You should know that there are alternatives. Companies such as [Porticor](http://www.porticor.com/) have developed a unique key management platform tailored for the cloud, enabling you to store your encryption keys in your cloud, without compromising the security of your keys.

**Encryption is not enough**

Encryption alone is not enough in an encryption solution when it comes to cloud environments. When dealing the multi-tenant nature of the public cloud, or even the inter-departmental shared resources of a private cloud, how encryption keys are stored and accessed is at least equally as important to securing data.

Policy-based key management can limit where and when data can be accessed.

When using shared computing environments, you also want to limit which servers can gain access to your data.  When a server makes an encryption key request, the encryption solution must be able to authenticate the server.  And server authentication can go beyond identity-based validations and include integrity checks as well, ensuring that the requesting server has up-to-date security in place before releasing the encryption keys.

Key ownership is another important element in encryption solutions.  If you only want to apply encryption to a particular public cloud service, accessing encryption through the service provider can be an easy add-on when available.  However, if you want an encryption solution that can be used with data stored in physical, virtual, and cloud servers, and even across cloud vendors, then you’ll want to retain key ownership in a solution that lets you manage encryption across all of those deployments.  This also maintains a separation of duties between you and your service provider.  And there are different key service options, such as on premise or through a SaaS solution, depending on what best meets your needs.

Twenty Rules for Amazon Cloud Security

1. Encrypt all network traffic.
2. Use only encrypted file systems for block devices and non-root local devices.
3. Encrypt everything you put in S3 using strong encryption.
4. Never allow decryption keys to enter the cloud—unless and only for the duration of an actual decryption activity.
5. Include NO authentication credentials in your AMIs except a key for decrypting the file system key.
6. Pass in your file system key encrypted at instance start-up.
7. Do not allow password-based authentication for shell access. Ever.
8. Do not require passwords for sudo access.
9. Design your systems so that you do not rely on a particular AMI structure for your application to function.
10. Regularly pull full backups out of Amazon and store them securely elsewhere.
11. Run only one service per EC2 instance.
12. Open only the minimum ports necessary to support the services on an instance.
13. Specify source addresses when setting up your instance; only allow global access for global services like HTTP/HTTPS.
14. Segment out sensitive data from non-sensitive data into separate databases in separate security groups when hosting an application with highly sensitive data.
15. Automate your security embarrassments\*.
16. Install a host-based intrusion detection system like [**OSSEC**](http://www.ossec.net).
17. Leverage system hardening tools like [**Bastille Linux**](http://bastille-linux.sourceforge.net/).
18. If you suspect a compromise, backup the root file system, snapshot your block volumes, and shut down the instance. You can perform forensics on an uncompromised system later.
19. Design things so you can roll out a security patch to an AMI and simply relaunch your instances.
20. **Above all else, write secure web applications**.

\* You know you have had them at one time or another. Things like that anonymous FTP site you have to have open for the batch file a client is sending you every night.

Homomorphic Encryption

Homomorphic encryption alludes to an encryption where plain texts and cipher texts both are treated with an equivalent algebraic function. Now, the plain text and the cipher texts might also be not related but the emphasis is on the algebraic operation that works on both of them. This encryption method is all the rage at the moment, more so because homomorphic encryptions can be specifically designed and altered to suit particular purposes.

**Secure Watermarking** - Watermarking is a new age method of securing content and data. However, over the years too many faulty systems of watermarking had made it very easy for people to encrypt important data without the owners’ knowledge. With homomorphic encryption however, this problem is as good as gone. The encryption method concentrates on altering the ciphering to suit the individual data, effectively individualizing the watermark for the data.

The limitation of homomorphic encryption so far has been the incredible amounts of processing power required to handle all that secure crunching.

While encrypted cloud data would present some difficulties – such as indexing and searching – it would also offer a new degree of security from hackers. Without the decryption key, even if a server was raided and data stolen, it would be considerably more difficult to make malicious use of it. The expectation is that partial-homomorphic encryption is likely to show up in federal and medical record keeping initially.

Searching databases is usually done in the clear. And even if the query is encrypted, it has to be decrypted (revealing its contents) before it can be used by a search engine. What's worse is that databases themselves are stored as plaintext, available to anyone gaining access. The smarter way to handle sensitive information would be to encrypt the queries, encrypt the database and search it in its encrypted form. Impossible until now, "homomorphic" encryption scheme that allows encrypted data to be searched, sorted and processed without decrypting it.