AWS Security Best Practices

January 2011

This paper is excerpt from
Architecting for the Cloud: Best Practices Whitepaper
(http://media.amazonwebservices.com/AWS_Cloud_Best_Practices.pdf)

Security Best Practices

In a multi-tenant environment, cloud architects often express concerns about security. *Security should be implemented in every layer of the cloud application architecture.* Physical security is typically handled by your service provider (Security Whitepaper [1]), which is an additional benefit of using the cloud. Network and application-level security is your responsibility and you should implement the best practices as applicable to your business. In this section, you will learn about some specific tools, features and guidelines on how to secure your cloud application in the AWS environment. It is recommended to take advantage of these tools and features mentioned to implement basic security and then implement additional security best practices using standard methods as appropriate or as they see fit.

Protect your data in transit

If you need to exchange sensitive or confidential information between a browser and a web server, configure SSL on your server instance. You'll need a certificate from an external certification authority like VeriSign[1] or Entrust[2]. The public key included in the certificate authenticates your server to the browser and serves as the basis for creating the shared session key used to encrypt the data in both directions.

Create a Virtual Private Cloud by making a few command line calls (using Amazon VPC). This will enable you to use your own logically isolated resources within the AWS cloud, and then connect those resources directly to your own datacenter using industry-standard encrypted IPSec VPN connections.

Protect your data at rest

If you are concerned about storing sensitive and confidential data in the cloud, you should encrypt the data (individual files) before uploading it to the cloud. For example, encrypt the data using any open source[3] or commercial[4] PGP-based tools before storing it as Amazon S3 objects and decrypt it after download. This is often a good practice when building HIPPA-Compliant applications [2] that need to store Protected Health Information (PHI).

On Amazon EC2, file encryption depends on the operating system. Amazon EC2 instances running Windows can use the built-in Windows Encrypting File System (EFS) feature. This feature will handle the encryption and decryption of files and folders automatically and make the process transparent to the users [6]. However, despite its name, EFS doesn't encrypt the entire file system; instead, it encrypts individual files. If you need a full encrypted volume, consider using the open-source TrueCrypt[5] product; this will integrate very well with NTFS-formatted EBS volumes. Amazon EC2 instances running Linux can mount EBS volumes using encrypted file systems using variety of approaches (EncFS[6], Loop-AES[7], dm-crypt[8], TrueCrypt[9]). Likewise, Amazon EC2 instances running OpenSolaris can take advantage of ZFS[10] Encryption Support [7]. Regardless of which approach you choose, encrypting files and volumes in Amazon EC2 helps protect files and log data so that only the users and processes on the server can see the data in clear text, but anything or anyone outside the server see only encrypted data.

[1] http://www.verisign.com/ssl/
[2] http://www.entrust.net/ssl-products.htm
[3] http://www.gnupg.org
[4] http://www.pgp.com/
[5] http://www.truecrypt.org/
[6] http://www.arg0.net/encfs
[7] http://loop-aes.sourceforge.net/loop-AES.README
[8] http://www.saout.de/misc/dm-crypt/
[9] http://www.truecrypt.org/
[10] http://www.opensolaris.org/os/community/zfs/

No matter which operating system or technology you choose, encrypting data at rest presents a challenge: managing the keys used to encrypt the data. If you lose the keys, you will lose your data forever and if your keys become compromised, the data may be at risk. Therefore, be sure to study the key management capabilities of any products you choose and establish a procedure that minimizes the risk of losing keys.

Besides protecting your data from eavesdropping, also consider how to protect it from disaster. Take periodic snapshots of Amazon EBS volumes to ensure it is highly durable and available. Snapshots are incremental in nature and stored on Amazon S3 (separate geo-location) and can be restored back with a few clicks or command line calls.

Protect your AWS credentials

AWS supplies two types of security credentials: AWS access keys and X.509 certificates. Your AWS access key has two parts: your *access key ID* and your *secret access key*. When using the REST or Query API, you have to use your secret access key to calculate a signature to include in your request for authentication. To prevent in-flight tampering, all requests should be sent over HTTPS.

If your Amazon Machine Image (AMI) is running processes that need to communicate with other AWS web services (for polling the Amazon SQS queue or for reading objects from Amazon S3, for example), one common design mistake is embedding the AWS credentials in the AMI. Instead of embedding the credentials, they should be passed in as arguments during launch and encrypted before being sent over the wire [5].

If your secret access key becomes compromised, you should obtain a new one by rotating[11] to a new access key ID. As a good practice, it is recommended that you incorporate a key rotation mechanism into your application architecture so that you can use it on a regular basis or occasionally (when disgruntled employee leaves the company) to ensure compromised keys can't last forever.

Alternately, you can use X.509 certificates for authentication to certain AWS services. The certificate file contains your public key in a base64-encoded DER certificate body. A separate file contains the corresponding base64-encoded PKCS#8 private key.

AWS supports multi-factor authentication[12] as an additional protector for working with your account information on aws.amazon.com and AWS Management Console[13].

Manage multiple Users and their permissions with IAM

AWS Identity and Access Management (IAM)[14] enables you to create multiple Users and manage the permissions for each of these Users within your AWS Account. A User is an identity (within your AWS Account) with unique security credentials that can be used to access AWS Services. IAM eliminates the need to share passwords or access keys, and makes it easy to enable or disable a User's access as appropriate.

IAM enables you to implement security best practices, such as least privilege, by granting unique credentials to every User within your AWS account and only grant permission to access the AWS Services and resources required for the Users to perform their job. IAM is secure by default; new Users have no access to AWS until permissions are explicitly granted.

[11] http://aws.amazon.com/about-aws/whats-new/2009/08/31/seamlessly-rotate-your-access-credentials/
[12] More info about Multi-factor Authentication is available at http://aws.amazon.com/mfa/
[13] AWS Management Console http://aws.amazon.com/console/
[14] More Info at http://aws.amazon.com.com/iam

IAM is natively integrated into most AWS Services. No service APIs have changed to support IAM, and applications and tools built on top of the AWS service APIs will continue to work when using IAM. Applications only need to begin using the access keys generated for a new User.

You should minimize the use of your AWS Account credentials as much as possible when interacting with your AWS Services and take advantage of IAM User credentials to access AWS Services and resources.

Secure your Application

Every Amazon EC2 instance is protected by one or more *security groups*[15], named sets of rules that specify which ingress (i.e., incoming) network traffic should be delivered to your instance. You can specify TCP and UDP ports, ICMP types and codes, and source addresses. Security groups give you basic firewall-like protection for running instances. For example, instances that belong to a web application can have the following security group settings:

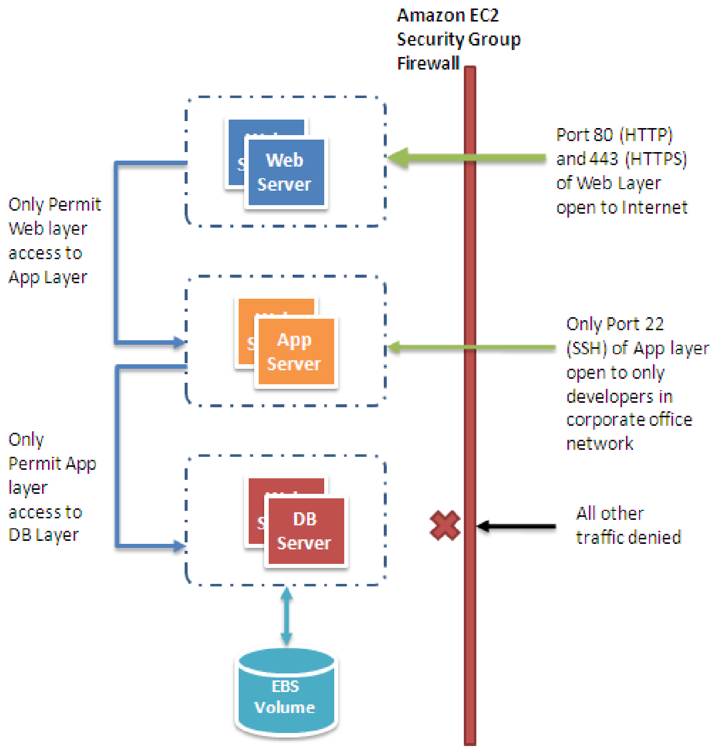
Amazon EC2 Security Group Firewall
Web Server
App Server
DB Server
EBS Volume
Only Permit Web layer access to App Layer
Only Permit App layer access to DB Layer
Port 80 (HTTP) and 443 (HTTPS) of Web Layer open to Internet
Only Port 22 (SSH) of App layer open to only developers in corporate office network
All other traffic denied

Figure 1: Securing your Web Application using Amazon EC2 Security Groups

Another way to restrict incoming traffic is to configure software-based firewalls on your instances. Windows instances can use the built-in firewall[16]. Linux instances can use *netfilter*[17] and *iptables*.

[15] More info about Security Group is available at http://docs.amazonwebservices.com/AWSEC2/2009-07-15/UserGuide/index.html?using-network-security.html
[16] http://technet.microsoft.com/en-us/library/cc779199(WS.10).aspx, March 2003
[17] http://www.netfilter.org/

Over time, errors in software are discovered and require patches to fix. You should ensure the following basic guidelines to maximize security of your application:

- Regularly download patches from the vendor's web site and update your AMIs
- Redeploy instances from the new AMIs and test your applications to ensure the patches don't break anything. Ensure that the latest AMI is deployed across *all* instances
- Invest in test scripts so that you can run security checks periodically and automate the process
- Ensure that the third-party software is configured to the most secure settings
- Never run your processes as *root* or *Administrator* login unless absolutely necessary

All the standard security practices pre-cloud era like adopting good coding practices, isolating sensitive data are still applicable and should be implemented.

In retrospect, the cloud abstracts the complexity of the physical security from you and gives you the control through tools and features so that you can secure your application.

This paper is excerpt from Architecting for the Cloud: Best Practices Whitepaper. [3]

References and Further Reading

1. **Amazon Security Team, Overview of Security Processes,** http://media.amazonaws.com/pdf/AWS_Security_Whitepaper.pdf, 2009-06-01
2. **Amazon Web Services Team, Creating HIPPA-Compliant Medical Data Applications With AWS,** http://media.amazonaws.com/AWS_HIPAA_Whitepaper_Final.pdf, 2009-04-01
3. **J.Varia, Architecting for the Cloud: Best Practices** http://media.amazonwebservices.com/AWS_Cloud_Best_Practices.pdf, 2009,-01-01
4. R. Bragg, The Encrypting File System, http://technet.microsoft.com/en-us/library/cc700811.aspx, 2009
5. S. Swidler, How to keep your AWS credentials on an EC2 instance securely, http://clouddevelopertips.blogspot.com/2009/08/how-to-keep-your-aws-credentials-on-ec2.html, 2009-08-31
6. Microsoft Support Team, Best Practices For Encrypting File System (Windows), http://support.microsoft.com/kb/223316, 2009
7. Solaris Security Team, ZFS Encryption Project (OpenSolaris), http://www.opensolaris.org/os/project/zfs-crypto/, 2009-05-01