Impact of AWS Account Identifiers

Asked 16 years, 3 months ago Modified 11 years, 6 months ago

Viewed 2k times Part of AWS Collective

17

I'm using Amazon's tools to build a web app. I'm very happy with them, but I have a security concern.

Right now, I'm using multiple EC2 instances, S3, SimpleDB and SQS. In order to authenticate requests to the different services, you include your Access Identifiers (login required).

For example, to upload a file to S3 from an EC2 instance, your EC2 instance needs to have your *Access Key ID* and your *Secret Access Key*.

That basically means your username and password need to be in your instances.

If one of my instances were to be compromised, all of my Amazon assets would be compromised. The keys can be used upload/replace S3 and SimpleDB data, start and stop EC2 instances, etc.

How can I minimize the damage of a single compromised host?

My first thought is to get multiple identifiers per account so I can track changes made and quickly revoke the 'hacked' account. Amazon doesn't support more than one set of credentials per account.

My second thought was to create multiple accounts and use ACL's to control access. Unfortunately, not all the services support granting other accounts access to your data. Plus bandwidth is cheaper the more that you use, so having it all go through one account is ideal.

Has anyone dealt with, or at least thought about this problem?

aws

security amazon-web-services amazon-s3 amazon-ec2 amazon-simpledb

Share

Improve this question

Follow

edited Jan 6, 2013 at 13:53

Steffen Opel

64.6k ● 11 ● 195 ● 212

asked Sep 24, 2008 at 22:09

Gary Richardson

16.4k ● 10 ● 54 ● 48

4 Answers

Sorted by: Highest score (default)

What you can do is have a single, super-locked down 'authentication server'. The secret key only exists on this one server, and all the other servers will need to ask it for

5

permission. You can assign your own keys to the various servers, and lock it down by IP address as well. That way if a server gets compromised, you simply revoke its key from the 'authentication server'.

This is possible, because of the way the AWS authentication works. Say your webserver needs to upload a file to S3. First, it will generate the AWS request, and send that request along with your custom server key to the 'authentication server'. The authentication server will authenticate the request, doing the crypto magic stuff, and return the authenticated string back to the webserver. The webserver can then use this to actually submit the request along with the file to upload to S3.

Share Improve this answer Follow

answered Sep 25, 2008 at 17:20

davr
19.1k ● 17 ● 80 ● 99

"Rolling your own" authentication server of this kind is likely to result in unexpected, and unnoticed, security flaws. AWS now (but not back when this was first asked) has facilities to manage the risks from the question. In particular, IAM roles (as described by @cudds's answer) is a good fit for this problem. – Charles Engelke Jun 22, 2013 at 13:27

5

AWS allows you to create multiple users with [Identity and Access Management](). This will allow you to implement either of your scenarios.

I would suggest defining an IAM user per EC2 instance, this allows you to revoke access to a specific user (or just their access keys) if the corresponding EC2 instance is compromised and also use fine-grained permissions to restrict what APIs the user can call and what resources they can access (e.g. only permit the user to upload to a specific bucket).

Share Improve this answer Follow

edited Aug 9, 2011 at 17:04

answered Jun 23, 2011 at 6:51

BenM
4,143 ● 3 ● 26 ● 28

2 Yes, IAM is the way to go. IAM permissions can be further refined to only what the exact instance needs to do (upload or read data from a very specific bucket, etc.) – Daniel Lopez Aug 9, 2011 at 10:40

Thanks for the comments Daniel, I've updated the answer to incorporate your suggestion. – BenM Aug 9, 2011 at 17:05

2 Now that roles are available, they are a better choice than IAM users for this problem. – Charles Engelke Jun 22, 2013 at 13:20

3

Furthermore, AWS IAM roles allow you to assign permissions to an EC2 instance rather than having to place keys on the instance. See the blog post at http://aws.typepad.com/aws/2012/06/iam-roles-for-ec2-instances-simplified-secure-access-to-aws-service-apis-

from-ec2.html Most of the SDKs utilize the temporary keys created by the roles.

Share Improve this answer Follow

answered Jun 22, 2013 at 5:16

cudds
1,618 ● 1 ● 11 ● 12

1

AWS offers "Consolidated Billing" which addresses your concern in the second thought.

https://aws-portal.amazon.com/gp/aws/developer/account/index.html?ie=UTF8&action=consolidated-billing

"Consolidated Billing enables you to consolidate payment for multiple Amazon Web Services (AWS) accounts within your company by designating a single paying account. You can see a combined view of AWS costs incurred by all accounts, as well as obtain a detailed cost report for each of the individual AWS accounts associated with your paying account. Consolidated Billing may also lower your overall costs since the rolled up usage across all of your accounts could help you reach lower-priced volume tiers more quickly."

Share Improve this answer Follow

edited Sep 14, 2010 at 16:50

answered Sep 7, 2010 at 16:40

Erik Osterman
11 ● 2