Hardening AWS Environments and Automating Incident Response for AWS Compromises

Abstract

Incident response in the cloud is performed differently than when performed in on-premise systems. Specifically, in a cloud environment, a responder can not walk up to the physical asset, clone the drive with a write-blocker, or perform any action that requires hands on time with the system in question. Incident response best practices advise following predefined practiced procedures when dealing with a security incident, but organizations moving infrastructure to the cloud may fail to realize the procedural differences in obtaining forensic evidence. Furthermore, while cloud providers produce documents on handling incident response in the cloud, these documents may fail to address the newly released features or services that can aid incident response or help harden cloud infrastructure.

This paper covers AWS APIs and services that can be leveraged to increase an incident response procedure. Additionally, we introduce a suite of four, newly released, and open source tools we wrote to demonstrate how programmatic use of the AWS API can be used to augment many areas of the incident response process. Finally, we discuss other open source tools and illustrate techniques to use several tools in order to augment each area of the incident response process.

Authors & Contributors

Andrew Krug is a Senior Software Engineer at a large cyber security company. Krug has been Consultant, Network Architect, Systems Administrator, Operations Manager, Technical Trainer, and Software Engineer. Currently Krug works to develop gamified security education through security simulation scenarios.

Alex McCormack is a software developer who assists in the design and implementation of capture the flag (CTF) competitions and training events. Alex has designed CTF challenges since 2013 and given training since 2012. Prior to developing CTFs, Alex worked in Incident Response and Malware Analysis.

Joel Ferrier is the creator of Margarita Shotgun. Joel currently works as a Systems Administrator. Previously Joel worked as a Systems Engineer with a focus in Security Operations.

Jeff Parr is the Frontend Guru for the project. He has been developing web technologies for over 15 years; primarily Rails for the last 5. Parr claims, "I love seeing people use what I build." Parr has been a subcontractor through engineering/design firms, innovation consultant, and more; specializing in pairing, reviewing code, and contributing to open source.

Incident Response in AWS

At first glance, incident response within AWS may appear more challenging than incident response in an on-premise environment. Not only are servers not physically available, but techniques associated with virtual machines such as taking a full disk and memory snapshot are not available.

However, by leveraging the APIs provided by AWS, organizations can prepare themselves to automatically collect evidence and mitigate compromises of AWS instances. Before implementing such a solution, the organization should understand what preparations need to be made, mitigations need to be performed, and what evidence needs to be collected.

Preparing for an Incident within AWS

Before an incident ever occurs, an organization should ensure it is in the best possible place to deal with an incident. Two areas an organization should address include strengthening defenses to reduce the attack surface, and increasing visibility to detect, understand, and prevent future attacks. AWS has several services to aid in these objectives including CloudWatch, CloudTrail, Config and IAM.

CloudWatch is a monitoring service that can be used to monitor metrics of AWS resources or even custom metrics supplied by an organization's own applications. Alarms can be placed on these metrics so that if a threshold is exceeded, an action is taken such as sending a notification or even terminating a resource. One of the most common alarms is one that watches the estimated charges of an AWS account. An estimated charge alarm is useful for detecting scenarios where an IAM key is compromised, and used to spin up many resources. Default metrics include attributes of EC2 instances, S3 buckets, and several other AWS resources.

"The forensic value of CloudTrail Logs can not be understated."

CloudTrail is an AWS service that records data about AWS API calls. The API calls may come from the AWS Management console, AWS CLI or AWS SDK. For each call made, the record will contain the time of the call, IAM identity of the user making the call, the source IP address of the call, the request parameters and the response elements returned. The forensic value of these logs can not be understated. In the event of an IAM compromise, CloudTrail could be the only source indicating what actions an attacker took. You can even use CloudWatch to monitor the number of incoming events logged to a trail. This is especially useful in environments where AWS changes are relatively minimal, and an alert should be sent if several API calls are made in a short period of time.

AWS Config provides logging configuration details, called a “configuration item” for supported AWS resources whenever that resource is created, deleted, or changed. The configuration item represents state information about the resource. For example, a configuration item would include things like the resource identifier, the contents of key-value tags, the availability zone of the object and information about related objects such as attached volumes for an EC2 instance.

AWS Config Rules is a distinct offering from Config. Config Rules is responsible for evaluating the configuration item against a set of predefined criteria and then alerting AWS users if that criteria is not met. AWS provides a set of default configurable rules users may use, as well as the ability to make custom rules and integrate with AWS Lambda. Lambda, a service that runs code as a service, can be used to take programmatic steps to remediate misconfigurations that have been identified with AWS Config. AWS Config provides a powerful historical view of the configuration state of AWS resources most organizations should enable even if they choose not to leverage the Rules capability.

The Identity and Access Management or IAM service allows delegating specific permissions to individual users. IAM users can also be created for specific services such as running on an EC2 instance or granting permissions to a Lambda function. With IAM, an organization can attach policies to an IAM user that allow specific access for particular AWS resources. Organizations should monitor these accounts using the IAM Access Advisor. Access Advisor will show the services for which a particular user has access, and will report the last time that user accessed that service. The organization can then decide to remove access to services that the user has not used.

IAM also includes a role mechanism that allows an instance to assume the privileges of an IAM user at runtime. Once a role is attached to an instance, the instance will be granted the permissions of an IAM user by obtaining credentials from the AWS Security Token Service. The tokens are automatically rotated every 15 or so minutes.

Mitigations to Perform

In March of 2016, Toni de la Fuente wrote a blog post detailing steps to perform using the AWS command line utility in order to collect evidence and mitigate a compromise. The mitigations included isolation, tagging, and shutting down an instance.

Isolation consists of creating a security group with exceptionally prohibitive access rules. Outbound traffic should be blocked completely, and inbound traffic should only be allowed by the specific IP address of the examiner.

The instance should then be tagged with a case number for record keeping and to alert other users that this instance should be treated with care. Evidence is then collected and the instance is shut down.

Evidence to Collect

Two of the most informative pieces of evidence to collect during an incident are disk and memory images. Disk image preservation is important because the disk of a compromised host may contain host specific logs detailing what happened, copies of malware, or other artifacts left by an attacker. Some attackers will alter the code of web applications to insert back doors, or collect sensitive data. Without forensic disk data, it may be impossible to determine what an attacker did after gaining access to the system.

Memory analysis is increasingly becoming a critical technique for forensic investigations. Memory analysis can be used to collect malware that may have been deleted from the disk, or never written to the disk in the first place. Memory analysis can be used to collect commands typed into a shell, discover programs hidden by rootkits, and much more.

In addition to disk and memory evidence, there are many AWS specific data points that may aid in an investigation. Metadata of an instance will reveal the public and private IP addresses of an instance, and the associated security groups. Console output and console screen shots may provide debugging messages from crashed services. VPC Flow logs may illustrate where an attack came from, and the destination of exfiltrated data. Finally, logs from CloudTrail may provide insights into actions performed by IAM users.

Automating IR with ThreatResponse Tools

Nothing can make handling an incident more frustrating than not having a plan to follow. Without a plan, responders may accidentally delete or fail to collect important evidence and inadequately remove access from the attacker. This could lead the responder to fail to understand what happened, and therefore prevent the attacker from getting back in.

> “An incident response plan can reduce the stress of an incident. ”

Having an incident response plan can greatly reduce the stress of responding to an incident. Responders can walk through the plan and know they are doing the right thing. However, following the plan still introduces the risk of human error, as a responder may skip a step or perform sensitive data acquisitions out of order.

By automating the collection of evidence and compromise mitigations, organizations can be fully prepared should a compromise occur. To help organizations with this automation, we are releasing four distinct tools that demonstrate how to automatically prepare, collect evidence and mitigate compromises. These tools are available at www.threatresponse.cloud.

Margarita Shotgun: Capturing Memory from AWS Instances

Margarita Shotgun is a python module and a standalone command line tool that automates the process of acquiring memory from remote systems, both on premise and in Amazon Web Services. Margarita Shotgun makes heavy use of paramiko to securely connect to remote systems and secure memory in transit between the compromised server and the incident responder workstation.

Margarita Shotgun makes use of LiME to capture memory. A configurable repository of prebuilt LiME kernel modules is available to streamline the memory acquisition process.

After determining the remote system's kernel version and architecture the appropriate LiME kernel module is loaded and system memory is streamed over an ssh tunnel to the incident responder's workstation. Memory can be saved to disk or streamed directly to an s3 bucket.

Memory acquisitions are performed in parallel with the help of Python's multiprocessing library, decreasing the period that compromised instances must be left running.

AWS-IR: Automatic Evidence Collection and Mitigation

AWS-IR is a python module and standalone command line tool that can be used to collect evidence, mitigate compromises, or start an AWS specific incident response workstation. The command line tool has three subcommands: `host_compromise`, `key_compromise` and `create_workstation`.

AWS-IR is built on top of boto3, an AWS SDK for the python language. In order to use AWS-IR, a user should set up their environment for use with the AWS SDK or have the appropriate credentials in place.

When using one of the compromise commands, AWS-IR collects evidence and tags instances according to a case number. The case number can be provided with the `-n` flag, and if one isn't provided, a case number will be randomly assigned. The case is the basic organization of evidence and logging for a particular incident. All evidence collected by AWS-IR is stored in an S3 bucket for that specific case. Additionally, every action performed by AWS-IR is logged and stored with the case.

AWS-IR: Automatic Response to a Host Compromise

When the `host_compromise` subcommand is used, AWS-IR starts collecting evidence and then mitigates the affected host. To use the `host_compromise` command, the user only needs to provide an IP address. An SSH username with root privileges and SSH key should also be provided if that information is available as this information is used to facilitate memory collection.

Once the host compromise command is initialized, AWS-IR will start mitigating the compromised host by attaching a new, highly restrictive security group to the instance. This is designed to sever any existing session an attacker may have and stop the exfiltration of data. AWS-IR will then snapshot all attached volumes, attempt to capture memory, collect metadata about the instance, grab console output and save a console screenshot. After the evidence has been collected, AWS-IR will shutdown the instance and provide the case number.

AWS-IR: Automatic Response to a Key Compromise

When the `key_compromise` subcommand is used, the user must provide the AWS access key id of the compromised key. AWS-IR will locate the IAM account associated with the key and disable the key. The `key_compromise` subcommand should be used with the `-c` flag to automatically create a workstation so the user can perform more analysis to see what actions may have been taken with the compromised key.

Users can load the evidence of a case into an incident response workstation by using the `create_workstation` subcommand. Additionally, by specifying the `-c` flag, with either of the compromise subcommands, a workstation will automatically be created for the responder.

ThreatResponse-Web: An Incident Response Workstation for AWS

ThreatResponse-Web is an incident response workstation, packaged as an AMI, that demonstrates pipelining multiple open source tools to both analyze the collected evidence, or collect additional evidence from other instances. After connecting to the workstation, a dashboard is displayed. The dashboard will illustrate the AWS regions instances are running in, recently launched instances, and the different types of AMIs those instances are running.

From the workstation a responder can take advantage of a full-text search to find other instances by AMI-ID, IP address or availability zone. If the responder determines another instance needs to be processed, that instance can be added to the case right from the workstation. When an instance is added to a case, either from inside the workstation or within AWS-IR, the workstation provides memory, disk, and configuration analysis capabilities.

Analysis with ThreatResponse-Web

The memory view allows the responder to analyze a memory dump using the volatility memory forensics framework. By leveraging a Javascript terminal, a volatility shell is provided inside the workstation. This allows the full feature set of volatility, and results can be copied and pasted out of the shell.

The disk analysis view leverages a full data analysis pipeline to automatically process the disk images collected. When a disk is analyzed, an EC2 instance is created and the snapshot of that disk is mounted as a volume to that instance. Plaso's Log2Timeline is then run on the attached volume to collect well formatted log files into a single `.plaso` file. Finally, TimeSketch reads the file to present a web view to the responder for further inspection. By leveraging AWS, multiple disks can be processed in parallel.

Finally, a view is available for an interactive configuration checking tool called ThreatPrep.

ThreatPrep: Preparing Your Environment for Optimal Evidence Collection.

ThreatPrep is a tool to examine an AWS environment with two main objectives. Firstly, identify areas where the security posture could be increased and secondly, identify areas where the amount of forensic evidence could be increased.

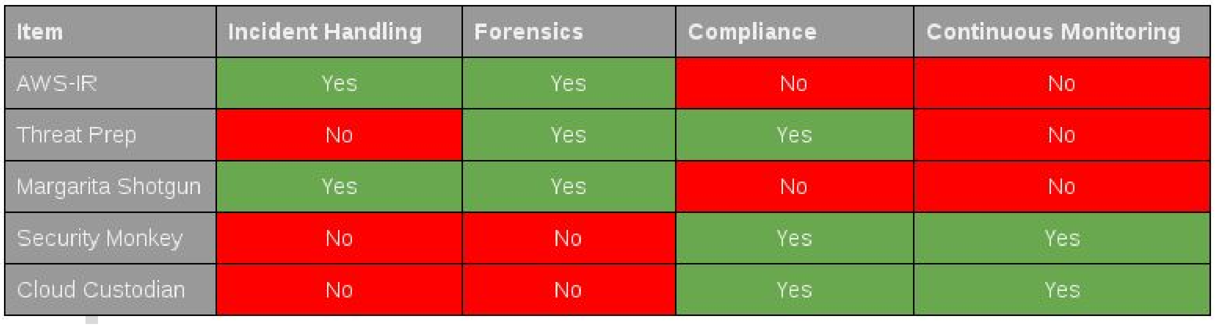
ThretPrep has many built in checks, including ensuring each S3 bucket has versioning and logging enabled and public reading or writing has been disabled. It checks each IAM user to ensure Multi-Factor Authentication has been enabled and that credentials have recently been rotated. It also checks each IAM user to ensure it is not attached to the AdministratorAccess policy. It will check each VPC to ensure flow logs have been enabled, ensure a CloudWatch alarm is set for Estimated Cost, and ensure roles and a multi-regional CloudTrail is enabled.

ThreatPrep can be used either from a command line, or used in a python project. In fact, the ThreatResponse-Web project includes output from ThreatPrep in the advice section.

ThreatPrep offers similar advice as AWS Trusted Advisor. Trusted Advisor is only available with a costly service plan. Further, Trusted Advisor can not be accessed programmatically. Because ThreatPrep can be used programmatically, it is easy to extend it to add additional checks or whitelist resources. ThreatPrep also shares similarities to AWS Config Rules, discussed above. However AWS Config Rules are not yet available in every AWS region and each rule costs two dollars per month per rule, and possibly more depending on how many times the rule is evaluated. The cost of the rules may make it prohibitive to smaller organizations who are price sensitive.

Additional Open Source Tools

In addition to the ThreatResponse tools discussed above and the built in AWS services like CloudTrail or Config and, organizations should also consider increasing their incident response process with other open source tools. Two of the larger open source projects in this area are Netflix's Security Monkey and Capital One's Cloud Custodian. Organizations can get a better idea of which tools they should use by considering where these tools can be used to augment the incident response procedure. The following graphic explains where each tool can augment a particular area of the incident response workflow.



Item	Incident Handling	Forensics	Compliance	Continuous Monitoring
AWS-IR	Yes	Yes	No	No
Threat Prep	No	Yes	Yes	No
Margarita Shotgun	Yes	Yes	No	No
Security Monkey	No	No	Yes	Yes
Cloud Custodian	No	No	Yes	Yes

Netflix is continuously releasing updates to Simian Army, which is a set of tools focused on many performance and compliance areas within cloud environments. One member of Simian Army is Security Monkey, which is described as a tool that "monitors policy changes and alerts on insecure configurations in an AWS account". For organizations wanting to try Security Monkey, Netflix recommends using their quick start docker container, and while it is not considered currently ready for production use, it will help organizations get started more quickly.

Cloud Custodian is an open source rules engine for managing an AWS environment. It describes itself as "[allowing] users to define policies to enable a well managed cloud infrastructure, that's both secure, and cost optimized". The policies are written in YAML configuration files for specific AWS resource types. Cloud Custodian integrates with lambda and cloudwatch events to validate and verify policies as changes are made to an AWS account. Cloud Custodian is a relatively new tool, being open sourced in April of 2016, but does appear to help address areas in compliance and continuous monitoring.

A variety of tools should be used in order to enhance the incident response workflow. But picking the right set of tools for a particular environment can be challenging. Organizations should arrange their environments to encourage experimentations and evaluations of the various tools to determine what works best for their environment.

Recommendations for Augmenting Incident Response

As organizations move towards an automated incident response process, they should consider separating their environments into multiple AWS accounts, building a continuous integration culture around their IR tooling, and utilizing Incident Response game days or security simulations.

Separate environments for testing, development, and production can be created by using multiple AWS accounts with consolidated billing. There are several high impact benefits to implementing environments in this manner. One advantage is that separating environments into multiple accounts allow the engineers or developers running that account to focus on the areas that most affects them. This separation of concerns can be empowering, as engineers understand that they can take risks and try new ideas without affecting the entire operation of the company. This allows and encourages testing and perfecting automatic responses to misconfigurations or instructions.

After separate environments are created, an organization should dedicate time to continuously improving their incident response tooling. These tools should be tested in the test and staging environment before being pushed into production. Even though many of the tools being utilized may be one-off scripts or not feel like “real” software projects, the authors should still maintain documentation, a bug tracker, and source code management. Encouraging security engineers to follow software development best practices will assist in avoiding technical debt around aging automated incident response tooling.

Finally, organizations should test their tooling with incident response game days or security simulations. Incident Response game days are exercises that allow the security team to practice how the organization would respond, should an incident occur. Generally speaking, a small set of employees are designated as the “red” team, and are supposed to see how far they can get into the organization’s systems before the blue team discovers and stops them. Some organizations let their blue team know about the testing, and others keep them in the dark. Organizations may also “skip ahead”, by disclosing an IAM access key to one of the red team members at the start of the exercise. This allows the security team to scope the assessment as understanding what would happen if a key were compromised. It should be noted that AWS does allow for incident response game days or security simulations but the terms of service require advanced notice of the event. For more information on how to host a successful security simulation, check out the AWS re:Invent talk “AWS re:Invent 2015 | (SEC316) Harden Your Architecture w/ Security Incident Response Simulations”.

Help Improve ThreatResponse

The ThreatResponse team encourages anyone interested in developing or providing feedback to connect with us on GitHub or send us an email at info@threatresponse.cloud.

References and Related Work

- AWS re:Invent 2015 | (SEC316) Harden Your Architecture w/ Security Incident Response Simulations
  - https://www.youtube.com/watch?v=u-mRU44Q5u4
- AWS Policy for Penetration Testing and Security Game Days
  - https://aws.amazon.com/security/penetration-testing/
- Boto3 python module for AWS SDK
  - http://boto3.readthedocs.io/en/latest/
- Cloud Custodian
  - https://github.com/capitalone/cloud-custodian
- Forensics in AWS, an introduction
  - http://blyx.com/2016/03/11/forensics-in-aws-an-introduction/
- Installing the AWS SDK
  - http://docs.aws.amazon.com/cli/latest/userguide/installing.html
- LiME Linux Memory Extractor
  - https://github.com/504ensicsLabs/LiME
- Log2timeline Project
  - https://github.com/log2timeline/plaso
- Paramiko Python SSH Module
  - https://github.com/paramiko/paramiko
- Remove Unnecessary Permissions in Your IAM Policies by Using Service Last Accessed
  - https://blogs.aws.amazon.com/security/post/Tx280RX2WH6WUD7
- Security Monkey
  - https://github.com/Netflix/security_monkey
- Security Monkey Docker Installation
  - https://github.com/Netflix-Skunkworks/zerotodocker/wiki/Security-Monkey
- Simian Army
  - https://github.com/Netflix/SimianArmy
- ThreatResponse
  - http://www.threatresponse.cloud
- ThreatResponse, GitHub
  - https://github.com/ThreatResponse
- TimeSketch
  - https://github.com/google/timesketch
- Volatility Memory Forensics
  - http://www.volatilityfoundation.org/