aws marketplace

SECURITY WHITEPAPER

How to protect cloud workloads with modern endpoint detection and response

A SANS Whitepaper by
SANS senior instructor
Dave Shackleford

Sponsored by AWS Marketplace

aws marketplace

Introduction

In the *Build and implement an effective endpoint detection and response strategy* webinar, SANS senior instructor Dave Shackleford and Amazon Web Services (AWS) senior security solution architect, threat detection and incident response, Kyle Dickinson discuss how you can implement endpoint detection and response (EDR) tools into your security strategy. You can watch it here on demand.

In this whitepaper, Shackleford takes a more in depth look at EDR tools on the cloud, including detection, automation capabilities, and scalability requirements.

AWS will explain how you can build and deploy EDR solutions from independent software vendors in AWS Marketplace.

The featured solutions for this use case can be found in AWS Marketplace:

CROWDSTRIKE

SentinelOne®

Browse AWS Marketplace to discover these and other products that can enhance your overall cloud security posture.

Learn more by visiting AWS Marketplace ›

SANS

Analyst Program

Whitepaper

How to Protect Cloud Workloads with Modern Endpoint Detection and Response

Written by **Dave Shackleford**

April 2023

aws marketplace

Introduction

As more organizations build cloud infrastructures, unauthorized users are taking notice and going on the offensive. Data can be exposed in transit, in storage within the cloud, while applications and systems are operating in infrastructure-as-a-service (IaaS) and platform-as-a-service (PaaS) environments, and so on. Accordingly, security investigations and response teams have looked for indicators of compromise (IoCs) throughout their environments. For most mature organizations, endpoint detection and response (EDR) tools have been a mainstay of workload protection, detection, and incident response within their traditional network environments. Now, this technology has evolved to cover cloud environments as well. Although some aspects of EDR are similar for any environment, there are several differences in the cloud, including detection, automation capabilities, and scalability requirements.

What Is EDR?

The EDR space is relatively mature and continues to grow and evolve as endpoint security becomes more interwoven with network security, SIEM, and other market spaces. For many enterprises, EDR functionality (whether standalone or within another endpoint security product suite like extended detection and response, or XDR) is used for threat hunting, response, and some degree of prevention too. Many EDR tools are endpoint security products that offer a blend of prevention capabilities along with signature-based detection and heuristic analysis of endpoint behavior to prevent and detect malware and other cyber threats. Key features to look for in an EDR product include fast querying, risk-scoring/prioritization, attack visualization, forensic evidence collection, a deep analysis engine, clear reporting/dashboards, and hunting/investigation tools. The newest features that are still emerging/evolving are largely focused on automation, integration with APIs and other tools like security, orchestration, automation, and response (SOAR), improvements in hunt queries, and better-integrated threat intelligence.

Modern EDR tools ideally will offer the following capabilities:

- **Query capabilities—**Fast, real-time, natural language query tools produce rapid answers to questions about IoC-type objects against the centralized data store or, optionally, against live systems.
- **Risk-scoring and prioritization—**Provides risk-prioritized views based on the confidence and severity of the incident, as well as the business value of the assets affected. Also, “tagging” devices based on Active Directory (AD), process, machine, and network information is a very useful function for dynamically assigning business value. (Although not a must-have, it’s definitely a nice-to-have.)

- **Attack visualization—**Click-down attack chain visualization tools enable investigators to easily pivot on interesting data elements or drill down for more information.
- **Forensic evidence collection—**Enables organizations to collect suspect files or memory and disk captures.
- **Deep analysis engine—**Provides an automatic integrated analysis of suspect processes or files in a cloud or on-premises sandbox, with clearly visible metadata, combined with global information. Not all EDR solutions provide a malware sandbox/detonation platform, but most have integration with popular third-party solutions.
- **Reporting and dashboards—**Supplies severity and confidence indicators on threat alerts.
- **Hunting and investigation tools—**An alert management workflow enables incidents to be assigned, transferred, annotated, and easily resolved.

There are several common use cases organizations rely on EDR to fulfill. Some of the more prevalent include:

- **Rapid detection of endpoint threats—**Both signature-based and behavioral detection with vendor-supplied threat intelligence should be a mainstay of EDR.
- **Automated or manual response capabilities at scale—**With agents deployed, clients can easily configure response policies that provide quarantine, alerting, and many more response capabilities, depending on the issue.
- **Deep threat intelligence and behavioral analysis—**Leading EDR solutions employ threat intelligence teams that track cyber threat campaigns and incident trends, IoCs, and more. These should be automatically ingested into the solution for cutting-edge threat detection.
- **Threat hunting and remote querying of endpoints—**With agents deployed, security analysts can perform targeted queries to endpoints, looking for a wide range of threat indicators.
- **Deep reporting of endpoint security posture—**A variety of default compliance and risk-centric reports should be available, and ideally custom report types can be generated.

There are many examples of incidents EDR could potentially help mitigate, both on premises and in the cloud. Once EDR tools are deployed, they often perform a variety of prevention and detection capabilities, while also allowing organizations to choose different models of response (automated, semi-automated, or manual interaction from analysts). For security analysts and cloud engineering teams, understanding events and the tactics employed is useful to better architect sound defense-in-depth strategies. Several examples of events against workloads follow.

Stealthy APT Compromise (On Premises or in the Cloud)

Although EDR tools often can prevent well-known threat incidents and campaigns, the most advanced adversaries sometimes still can manage to gain access to systems. In this case, EDR tools can record and collect artifacts and evidence on systems to aid in investigations and remediation, including network connections, user accounts, process execution, file creation and modification, and behaviors like privilege escalation. Alerts can tell security analysts that suspicious files or activity has been detected, perform automatic quarantine if desired, and also allow for remote connectivity to the systems in question for manual analysis.

Cryptomining Malware Targeting Cloud Workloads

A cryptomining incident in the cloud usually consists of compromise to a workload environment like Amazon Elastic Compute Cloud (EC2, for traditional instances), Elastic Container Service (ECS), or Elastic Kubernetes Service (EKS, for containers and Kubernetes orchestration). With EDR, there are many detection methods for cryptomining and malware activity, including:

- EDR malicious software prevention or detection
- Attempted privilege escalation or other local user activity
- Unusual access patterns to workloads or workload images
- Unusual privilege use or identity and access management (IAM) policy access
- Unusual network activity, potentially to known malicious domains

This workload could then be isolated automatically to prohibit outbound network access if forensic artifacts or investigation are desired/required, or the instance can be shut down.

Varied Attack Phases (Cloud-Centric)

This scenario follows several phases that align with MITRE ATT&CK® for Cloud:

- **Initial access—**A DevOps engineer is successfully phished during a campaign, which could be detected by EDR tools or email security platforms/services.
- **Execution—**Malware is executed on the compromised engineer's laptop, which can be detected by EDR, tuned, and updated properly.
- **Persistence—**The phisher discovers AWS CLI credentials stored locally on the engineer's laptop, granting access to the cloud environment, where they access a stored EC2 image. They install a backdoor on the image, which can likely be prevented or detected with the EDR agent installed.
- **Privilege escalation—**This EC2 image is used to spin up a new EC2 instance that has assigned credentials through an instance profile. The cybercriminal uses this metadata access to discover new services that the EC2 instance has access to, like S3 and Lambda. The EDR agent on the EC2 instance can detect access to the metadata service.

Most of these phases can be readily detected, and likely prevented, with a mature EDR platform widely deployed across systems in both on-premises and cloud environments.

There are several known challenges with EDR that many organizations may experience. Fortunately, there are ways to address the more common concerns and, in some case, method for avoidance. A few of the more common issues security teams face with EDR include:

- **Choosing the wrong solution—**Some organizations make decisions based on preexisting relationships with vendors, cost, licensing, or lack of insight into how a mature EDR solution should function.
  - **How to address—**Ensure the evaluation of EDR solutions is based on true capabilities and compatibility with all environments that need coverage.
- **Lack of operational capacity—**EDR tools require significant tuning and configuration and are not as effective without dedicated operational oversight and ongoing maintenance and monitoring.
  - **How to address—**Security teams should plan to dedicate operational resources to installing, configuring, and maintaining EDR solutions to ensure the best day-to-day outcomes. The heaviest operational requirements are usually upfront during installation and tuning, but some dedicated capacity will be needed to maintain the solution moving forward.
- **Improper tuning—**Organizations sometimes struggle with tuning EDR tools for widely disparate environments and threat models. Tuning can be one of the most time-consuming activities for analysts, both upon initial setup and deployment as well as later as environments and systems change.
  - **How to address—**Although tuning EDR platforms can't be avoided entirely, the best approach is to set up a dedicated test environment where images of workloads can be rapidly deployed for evaluation after EDR is installed.
- **Overemphasizing prevention—**Most leading EDR platforms offer prevention capabilities, but these solutions are largely oriented toward detection and response activities. Ideally, malware and intrusion efforts are blocked outright, with logs and alerts to accompany these attempts, but the cyber threat landscape is changing constantly, making it difficult to guarantee prevention in all cases.
  - **How to address—**Security teams should absolutely enable prevention policies with EDR where feasible but be pragmatic about how the tools will perform in a dynamic threat landscape. Focus on highly tuned detection along with more automated response actions to minimize impacts of security incidents and events on your workloads and be sure to set expectations with any interested stakeholders.
- **Implementation and management limitations—**Deployment of EDR can be problematic in very large, dispersed environments that cover both on-premises and cloud workloads. Network latency can have an impact on updates and installation, and most EDR needs access to the provider cloud for new signatures and policies, making systems not connected to the internet somewhat problematic to update easily.
  - **How to address—**Consider localized installation with different teams, if possible, to facilitate more streamlined implementation. Most EDR platforms can provide offline updates to systems not on the internet, too, so try to inventory these and develop a plan to keep them up to date with the latest threat detection and capabilities.

EDR for the Cloud

In today's cloud environments, many enterprises struggle with dynamic deployment models, rapidly changing scale and scope of assets, and overall visibility into all types of assets and services. There are numerous diverse systems in the public cloud, including containers and traditional workloads, and as organizations make the shift to cloud, there is often a lack of adequate tooling and process focus to update workflows and security processes. To improve the state of security monitoring and response in the cloud, security operations teams should:

- Perform proactive threat hunting
- Investigate systems rapidly
- Assess workload, identity state, and attributes quickly for response and investigation
- Align numerous cloud events to detect tactics, techniques, and procedures (TTPs)

Security teams are also realizing that a vast number of changes can occur with security operations workflows, investigations playbooks, evidence collection, and tooling. Although it may seem daunting at first, the cloud affords security operations teams a number of distinct benefits that may improve security controls efficacy and efficiency in a number of ways. First, the cloud provider fabric opens a number of APIs that EDR solutions and tools can integrate with. This allows a more programmatic approach to accessing cloud assets and data and can also foster interoperability and integration between numerous security tools and services in a variety of categories beyond just workload/endpoint platforms, including network monitoring and prevention, event management, and many more. Second, deployment updates might be much simpler given the ease of creating, updating, and deploying workloads in general.

Speaking of workloads, one fundamental difference between traditional on-premises servers and cloud workloads is in the format and types of workloads themselves. IaaS cloud environments readily support full virtual machines, which EDR can be easily deployed onto. For most organizations, the simplest strategy for workload installation will be to ensure all instance virtual machine (VM) images (such as AMIs in Amazon EC2) have agents in place and are updated as needed to ensure the production images are using the appropriate EDR version(s). In most ways, this might mimic internal processes and practices for installing and updating EDR in VM image templates deployed in a data center—although many teams are able to deploy and update much more flexibly and quickly in the cloud due to native cloud provider tools and services like AWS OpsWorks and AWS Systems Manager.

Many progressive cloud engineering and DevOps teams are leveraging newer, cloud-native IaaS and PaaS functionality for deploying and orchestrating workloads today, such as containers and serverless. For containers and related orchestration services in PaaS deployments, there are various ways to run and deploy container images. If the organization operates the underlying container host platform (such as EC2 instances running Linux), these should be locked down and secured like traditional server workloads. Container images should be locked down, as well, although the operating system configuration items won't be present. Container image security configuration should include:

- Restricting container service and application packages to a minimum and ensuring they are patched and updated
- Minimizing service and application permissions in images
- Ensuring no secrets are present in build and configuration files

For serverless PaaS deployments, neither the underlying host OS nor the container is under the organization's control. Because the container host platform—OS, for example—is largely out of scope, organizations should focus on the inputs to serverless functions, the function code, and the execution runtime options available in the cloud environment.

To cover all types of workloads (server instances, containers, and serverless), any mature EDR platform should offer both agent-based and agentless deployment options. Although IaaS instances and some container deployments with full host OS control will support agents, many container-based and all serverless PaaS deployments won't be able to. Many agentless models will rely on a combination of log monitoring and passive runtime detection using a specially configured container or serverless function to provide workload protection capabilities. There are benefits and drawbacks to agents and agentless models, including:

	Benefits	Drawbacks
Agent-based EDR	• More in-depth analysis of local workload incidents and behaviors • Improved EDR capabilities due to deeper integration	• Additional workload overhead • Agent deployment requirements and potential local conflicts with OS and applications
Agentless EDR	• Ability to monitor containers and serverless configuration and vulnerabilities • Minimal overhead for workloads	• May lack certain preventive capabilities for runtime issues • May require a separate container or serverless function to operate

For most enterprises, any deployment of EDR platforms will likely include a combination of both agents and agentless approaches in a mixed IaaS/PaaS infrastructure.

There are several threat issues EDR must focus on in cloud workloads environments. Many attacks against standard instances in the cloud will work just as they have for on-premises VMs if they're related to exposed OS vulnerabilities or applications installed within the instances. However, cloud instance workloads still offer a new realm of identity role assignment and integration with the cloud fabric, as well as access to the instance metadata services that could lead to privilege escalation attempts and many others that EDR will need to recognize.

Containers definitely introduce new opportunities for cloud-centric threats today. Many native container services offer a host of APIs that adversaries may infiltrate and attempt to exploit, along with traditional identity-centric issues such as privilege escalation and hijacking IAM keys to gain access to container workloads and services. When orchestration services like Kubernetes are in the mix (such as Amazon EKS), new configuration options and APIs are exposed, as well, which could lead to a variety of incidents, like cryptomining, malware implant attempts, and more. Many strikes against container environments will focus on container images in registries like Amazon Elastic Container Registry (ECR). If these services aren't well protected, images could be infected with malicious packages that EDR tools would need to detect once they began exhibiting malicious behaviors.

Common threat incidents in serverless environments may include:

- **Event injection—**Injection flaws still can affect your code and how it handles the input. Better input validation and predefined database layer logic (stored procedures, for example) can help solve these issues.
- **Broken authentication—**Organizations should enforce strong authentication and role definitions for users of serverless apps.
- **Insecure deployment settings—**Limiting memory usage, input vectors, and output data and formats can help minimize configuration issues and vulnerabilities.
- **Malicious serverless code access—**Limit permissions to the serverless services in cloud accounts and monitor for all unusual access events.
- **Insufficient logging—**Serverless logging can be captured through cloud control plane services like AWS CloudTrail.
- **Insecure storing of app secrets—**Secrets management and key store services like KMS should be used to minimize these configuration risks.
- **DoS threats and financial exhaustion—**Timeouts and threshold/throttling can help mitigate these threats to some extent.

EDR platforms should ideally be able to integrate with cloud provider APIs and services to monitor serverless functions, detect many of these types of incidents and behaviors, and initiate actions that can quarantine functions or isolate the threats to mitigate risk.

When organizations begin implementing EDR in the cloud, they should prioritize several core considerations. These include:

- **Scale—**EDR solutions need to be able to scale across numerous cloud accounts and regions. Many cloud environments are large and include many thousands of workloads. A mature EDR platform will be able to accommodate all workload types with no issue, incorporating all covered workloads in a single management console.
- **Automation—**For an EDR platform to function well in the cloud, it will need to be capable of integrating with cloud service provider APIs to facilitate response capabilities. For many types of cloud workloads, quarantine and isolation will be accomplished with restrictions in IAM role assignment or accessibility to other services, workloads, and app components like storage nodes.

- **Performance—**In cloud environments, workload processing increases can lead to higher billing costs, so minimizing performance impacts is critical. Leading solutions usually have lightweight agents, and any additional serverless functions or container images also should be as small and nimble as possible.
- **Licensing—**Due to the nature of cloud workload deployment life cycles, it's common for any given workload or orchestrated set of workloads to run for short periods of time before new, updated ones replace them. As every new workload has a unique identifier associated with it, EDR tools may view this dynamic, fast-paced cycling of workloads as a continuous stream of "new" agent licenses that may exceed traditional licensing plans and models. Accordingly, security teams should review licensing plans with any prospective EDR provider to make sure they understand the nature of cloud workload operation and don't continually flag customers for exceeding licensing thresholds as workloads are cycled in cloud service environments.
- **Cloud-specific threat signatures—**Cloud EDR needs to recognize threat signatures and behaviors that are specific to the cloud, in addition to the breadth of standard incidents you might encounter in any traditional environment (malware, exploit attempts, brute-force attacks against services, etc.). These would include strikes against IAM policies, attempts to access workload APIs, attempts to access workloads or storage services that aren't normally associated with the workload in question, and many more.

Building and Implementing a Cloud EDR Solution

For deployment of EDR into a cloud environment, there are options available for all possible types of workloads. As most leading EDR uses a cloud-based management console, the primary concern is agents and workload monitoring/response. The following are things to consider during deployment and for ongoing operational management and oversight:

- **EDR on traditional instances (such as Amazon EC2)—**For traditional instances, it likely makes sense to deploy EDR agents as you would on servers in a data center. Many EDR solutions can ingest IaaS API keys to discover instances across accounts and regions if provisioned properly, and can then perform automatic deployments to instances as desired. An alternative is to deploy an agent into instance image templates (like EC2 Amazon Machine Images, or AMIs) that is then instantiated when the image is used to deploy a running instance in production.

- **EDR on containers and container services (like Amazon ECS, EKS, or AWS Fargate)—** For containers and container services, things are a bit more complicated. Consider the following container options for cloud EDR:
  - **Agents—**Deploying agents into containers is not necessarily recommended. Although likely effective in some container runtime models, agents can lead to significant added overhead for containers, usually much more lightweight than instance workloads.
  - **Sidecar containers—**The "sidecar" model deploys a separate container for EDR monitoring and response that is associated with the containers it protects. This is a good balance between capabilities/functionality and cost management, but requires more operational oversight to align the sidecar with each container it needs to protect. This can prove unwieldy for highly dynamic container workload deployments, unfortunately.
  - **A centralized agent—**Some EDR solutions support deployment of a single instance with an agent that can then communicate with container workloads via API. This can save organizations significant financial and operational resources if supported, and also can function as efficiently as sidecar deployments in most cases.
- **EDR for serverless (like AWS Lambda)—**Some EDR solutions can readily monitor and respond to threats against serverless workloads, usually through deployment of a dedicated EDR function that has permissions to any protected workloads. This function is usually more adept at rapid detection than prevention, but can trigger automated API calls for response to isolate or shut down functions exhibiting malicious behaviors.

Today, beyond just deployment models and integration options, many AWS Marketplace EDR solutions also can integrate with a wide variety of other tools and services, such as network detection and response (NDR), extended detection and response (XDR), and native security options like AWS Security Hub, Amazon GuardDuty, Amazon Detective, Amazon CloudWatch, and more.

Architecture and Operations Design and Considerations

For EDR in the cloud, there are some best practices we've learned after doing this for several years. Some of the most important operational and architectural considerations include:

- **Leverage automation and APIs.** To fully realize the power of EDR in a cloud environment, security teams will need to heavily leverage cloud-provider APIs and services, especially for remediation and response actions. Services like AWS Config, AWS CloudTrail, and Amazon CloudWatch can play definitive roles in modifying workload configurations or role assignments for systems under investigation or assign a more isolated Security Group to restrict network traffic. All of these services are easily accessible and controllable through direct API access, and EDR tools can act as the initiator of automation workflows that tie into numerous distinct services depending on the circumstance.

- **Integrate with SIEM and network security where possible.** It's well known that many security teams are evolving standalone security controls like EDR into a more converged detection and response model known as extended detection and response (XDR). With XDR, network security and event collection and analysis are aligned and integrated to provide more complete visibility into activity in the environment and end-to-end capabilities for defense. For cloud EDR, both cloud-native and third-party network security and SIEM can help improve the security architecture for detection and response.
- **Build API-driven playbooks.** Playbooks for cloud-based detection and response scenarios should be developed with APIs and cloud services in mind. When developing EDR playbooks, security teams usually map out specific detection events that are categorized and prioritized (usually by severity), followed by a series of automated alerting and response actions mixed with investigation and analysis steps. SOC teams that are managing EDR in the cloud should become acquainted with all the types of cloud services they can take advantage of in AWS, such as:
  - **CloudTrail—**Logs of API events can easily be analyzed for context and analysis of activity affecting workloads.
  - **Config—**AWS Config can report workload status and perform configuration changes as desired.
  - **S3—**Automated artifact and evidence collection can be sent to S3 buckets for protection and analysis.
  - **Security Hub—**Leading EDR tools should natively integrate into Security Hub for reporting and development of metrics.
- **Develop cloud-specific queries for IOCs and TTPs.** One of the most valuable functions of EDR is the ability to query systems for specific IoCs and TTPs used by cybercriminals. Security operations teams need to develop queries that look for cloud-specific behaviors and artifacts that align with the workloads in use. For example, Amazon Linux is a very specific Linux kernel that is heavily used in EC2 instances, and has AWS-specific configuration files and service changes suited to the AWS environment. Attackers also will look for AWS API keys exposed within instances and seek to access and manipulate workload-specific APIs that may be exposed. Solutions available in the AWS Marketplace ideally should have some queries available "out of the box" that align with AWS native workloads.

Cloud EDR should be updated frequently with new intelligence from the provider's cloud-based analytics platform. Leading EDR solutions have highly capable machine learning models that can provide cutting-edge detection signatures and behavioral analysis to keep pace with the constantly changing cloud threat landscape. EDR agents and monitoring capabilities in the cloud will need continuous access to the provider's cloud to stay up to date. To add an additional layer of security for workloads and agents, cloud-native firewall services like AWS Network Firewall or even basic Network Access Control Lists (NACLs) at the subnet level should be applied to ensure access is carefully controlled.

Conclusion

EDR for the cloud is a growing and maturing area that is quickly becoming the prevalent deployment model for many organizations. Both EDR providers and cloud service providers are expanding and enhancing their capabilities and integration options, all with an eye toward efficiency and automation. One of the biggest challenges for security teams is learning all the different types of services and APIs available within the cloud fabric with which EDR can integrate and operate. Developing a truly efficient and effective EDR strategy requires deep alignment across services that EDR has exposure to, and playbooks from the EDR providers themselves can help SOC teams learn and build automated workflows that arguably provide faster and more effective detection and response than traditional on-premises models ever could.

Sponsor

SANS would like to thank this paper's sponsor:

aws marketplace

Why use AWS Marketplace?

AWS Marketplace is a digital software catalog that makes it easy to find, try, buy, deploy, and manage software that runs on AWS. AWS Marketplace has a broad and deep selection of security solutions offered by hundreds of independent software vendors, spanning infrastructure security, logging and monitoring, identity and access control, data protection, and more.

Customers can launch pre-configured solutions in just a few clicks in both Amazon Machine Image (AMI) formats and SaaS subscriptions, with entitlement options such as hourly, monthly, annual, and multi-year contracts.

AWS Marketplace is supported by a global team of solutions architects, product specialists, and other experts to help IT teams connect with the tools and resources needed to streamline migration journeys to AWS.

How to get started with EDR security solutions in AWS Marketplace

Security investigations and response teams use AWS native services and seller solutions in AWS Marketplace to help build automated, innovative, and secure solutions to address relevant use cases and further harden their cloud security footprint.

Here are some EDR solution resources to get you started:

Watch the webinar:
Build and implement an effective endpoint detection and response strategy

Watch on demand ›

Discover solutions:
Find EDR security tools available to protect your AWS architecture.

Visit AWS Marketplace ›

Talk to an expert:
Speak with a solution architect who can help solve your business challenges.

Get Connected ›