Other Logs

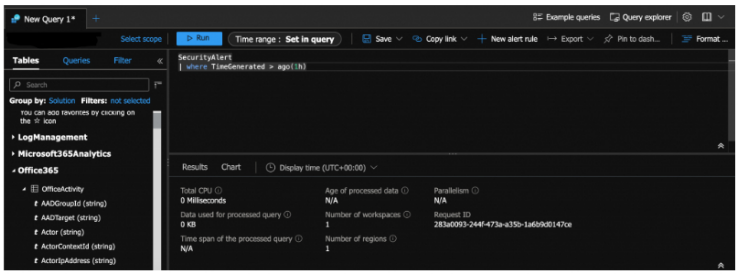
This lesson will discuss some logs that do not fall under Syslog, Sysmon or Windows Event logs. While these three encompass a lot of log sources we need to monitor, from desktop PCs to servers, network devices to applications, some systems use different methods to create and transport logs. Cloud providers such as Microsoft Azure and Amazon Web Services use their own methods for logging and monitoring, called an Application Programming Interface, or API. We will also cover OSQuery for endpoints, and Moloch for network traffic capture and indexing.

MICROSOFT AZURE

Microsoft Azure is one of the most commonly used cloud services in the world. Because of the large adoption of Azure, it is important to have general knowledge on how to navigate logging and monitoring in Azure. Logs in Azure, are primarily monitored through Azure Monitor and Log Analytic Workspaces.

Azure Monitor is able to pick up logs from a multitude of different Azure services such as, virtual machines, virtual networks, Azure Active Directory, and Azure Security Center, as well as on-premises services. Azure has three primary categories of logs: Control/Management logs, Data Plane logs, and Processed Events. These logs are fed to Azure through the Azure REST API, the Microsoft Graph API, JSON, and various other sources. Azure logs can also be connected to different kinds of SIEMs such as Splunk or even Microsoft's own Azure Sentinel.

When investigating logs in Azure, you will need to use the Kusto Query Language (KQL) to query logs. While the details of KQL are out of scope for this exam, it is something to be aware of, if you are tasked with monitoring logs inside of an Azure environment. The below screenshot is an example of a KQL query.



KQL Query for all Security Alerts within the past hour

AMAZON WEB SERVICES

AWS is huge. Like, seriously, freaking massive. While the GUI offers an easy way to access and manage resources, Amazon uses their own API for AWS, which is extremely extensive. Looking at the below example of an AWS API call, it resembles a (simple) REST API example:

```
(example provided by https://www.dummies.com/programming/cloud-computing/amazon-w

https://ec2.amazonaws.com/?Action=RunInstances
&ImageId=ami-60a54009
&MaxCount=3
&MinCount=1
&Placement.AvailabilityZone=us-east-1b
```

```
&Monitoring.Enabled=true
&AUTHPARAMS
```

The call, which is straightforward, instructs AWS to run between one and three instances based on an Amazon machine image of ami-60a54009 and to place them in the us-east-1b availability zone.

AWS provides monitoring capabilities, and this call instructs AWS to enable this monitoring. The AUTHPARAMS part is a stand-in for the information that AWS uses to implement security in its API. Know that this call has the appropriate security mechanisms in place to ensure its execution.

If you want to learn more about the AWS API, you can find guides on their documentation subdomain – https://docs.aws.amazon.com/index.html

OSQUERY

Osquery is a universal endpoint agent that was developed by Facebook in 2014. It is an active and growing open source project on GitHub, with 230 contributors and over 90 releases to-date.

According to the official osquery docs, osquery (os=operating system) is an operating system instrumentation framework that exposes an operating system as a high-performance relational database. Using SQL, you can write a single query to explore any given data, regardless of operating system.

This is a unique approach in the security landscape, creating one agent for many operating systems, leveraging a standard query language instead of creating a proprietary one, and collecting rich data sets which have broad applications. Osquery represents a fundamental rethinking of the fragmented, siloed approach plaguing the security industry today.

With that said, osquery is just an agent – "an instrumentation framework" for data collection. Security teams looking to put osquery into production and leverage the data for security protocols will need to consider:

1. How you'll configure, deploy, and manage the agent
2. How you'll manage query packs (more on these below) and schedules as the community adds more
3. Where you'll store osquery data (and how much it will cost)
4. How you'll analyze the data – i.e., what problems are you looking to solve? What questions do you need to ask?
5. How you'll handle suspicious activity that requires further investigation or remediation
6. Whether you need any integrations with existing tooling
7. How you'll troubleshoot production issues and develop any custom functionality you may need

MOLOCH

Moloch augments your current security infrastructure to store and index network traffic in standard PCAP format, providing fast, indexed access. An intuitive and simple web interface is provided for PCAP browsing, searching, and exporting. Moloch exposes APIs which allow for PCAP data and JSON formatted session data to be downloaded and consumed directly. Moloch stores and exports all packets in standard PCAP format, allowing you to also use your favorite PCAP ingesting tools, such as wireshark, during your analysis workflow.

Moloch is built to be deployed across many systems and can scale to handle tens of gigabits/sec of traffic. PCAP retention is based on available sensor disk space. Metadata retention is based on the Elasticsearch cluster scale. Both can be increased at anytime and are under your complete control.

You can find more information on the Moloch Github – https://github.com/aol/moloch

Previous Topic
Mark Complete
Back to Lesson