Microsoft

GRAYHAT

```
GrayHat2020
| where TalkTitle == "Blue Teaming with Kusto Query Language (KQL)“
| project SpeakerName, CompanyName, TwitterHandle, ContentLink
```

SpeakerName	CompanyName	TwitterHandle	ContentLink
Ashwin Patil	Microsoft	https://www.twitter.com/ashwinpatil	https://github.com/ashwin-patil/blue-teaming-with-kql

About me

- Senior Program Manager @
Microsoft Threat Intelligence Center (MSTIC) RnD Team

- Over 10 years in Security Monitoring, DFIR.

- KQL user for around 3 years.

- Azure Techcommunity Blogs :
(Threat Hunting and Jupyter Notebooks)
www.aka.ms/AzureSentinelBlog

Agenda

- Why learn query language ?
- Intro to KQL
- Structure of Basic KQL Query
- KQL Basic Searches
- Exploring Tables, Schemas
- Asset/Device Details
- Query Parameterization
- Dynamic datatypes
- Datetime
- Regex Extraction
- Functions (User-defined ,Built-in)
- Externaldata
- Time Series Analysis
- Threat Hunting Use-Cases
  - Time Series Anomaly
  - Network Beaconing
- KQL Programmatic Interfaces
  - Msticpy query provider
- KQL Playground, Trainings, Resources
- Conclusion

Why learn Query Language ?

Data Cleaning and Preprocessing is often required prior to analysis

Faster and Efficient data analysis and investigation.

Find actionable insights from broad hunting hypothesis results

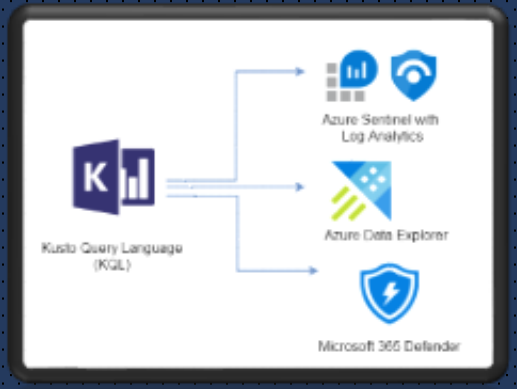
Leverage the full power of your SIEM.

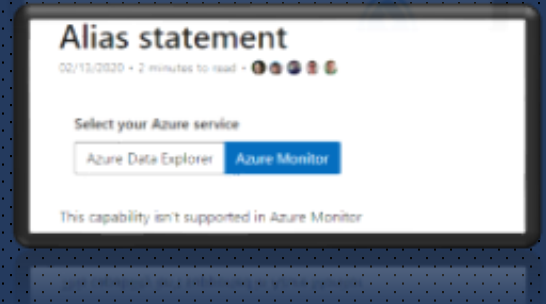
Intro to KQL

- What is KQL ?

- Applicable Products
  - Azure Data Explorer
  - Azure Sentinel/Log Analytics
  - Microsoft 365 Defender

- For some KQL operators – varying support between products (Validate on KQL Docs Page) – www.aka.ms/kqldocs
- KQL elements not supported in Azure Monitor





Structure of Basic KQL Query

1. Variable Declaration
2. Table Name
3. Datetime Filtering
4. Event type Filtering
5. Output Formatting/Display selected fields
6. Limit Results

```
let timeframe = 1d;  1
OfficeActivity  2
| where TimeGenerated >= ago(timeframe)  3
| where Operation == "MailboxLogin" and Logon_Type != "Owner"  4
| project Operation, OrganizationName, UserType, UserId, MailboxOwnerUPN, Logon_Type  5
| limit 100  6
```

KQL Basic Searches

- Search –
  - not recommended / CPU Intensive
- Sort/Order
- Filter
- Aggregation

```
// Search for presence of keyword and output tables where it is present
search "badaccount"
| where TimeGenerated > ago(4h)
| summarize count() by $tableName

// search for IP in multiple tables - irrespective of field names
search "8.8.8.8" in ("AzureNetworkAnalytics_CL", "CommonSecurityLog")
| where TimeGenerated > ago(1h)
| limit 100
```

```
// Sort by time
AzureActivity
| where TimeGenerated > ago(1h)
| sort by TimeGenerated desc
```

```
// aggregation by Field name
OfficeActivity
| where TimeGenerated > ago(1h)
| summarize count() by OperationName
```

```
// Filter by value
SecurityEvent
| where TimeGenerated > ago(1h)
| where EventID == 4688
| limit 100
```

Exploring Tables, Schemas

- Usage Table

KQL operators

- getschema
- workspace

```
// DataTypes ingested along with the Sizes
Usage
| where TimeGenerated > ago(1d)
| summarize DataSizeinMB = sum(Quantity) by DataType
| sort by DataSizeinMB desc
```

```
// Schema and datatypes for each field of Table
AzureActivity
| getschema
```

```
// Tables across Workspace Queries
union workspace('WorkSpace01').Heartbeat, workspace('WorkSpace02').Heartbeat
| where TimeGenerated > ago(1d)
| where Computer == "CH-UBNTVM"
| limit 100
```

Asset/Device Details

- Azure Sentinel Tables
  - Heartbeat (OMS Agent onboarded)

Microsoft 365 Defender
- DeviceInfo
- DeviceNetworkInfo

```
// Asset Details
Heartbeat
| where ComputerIP == "40.71.227.249"
| summarize LastReported = max(TimeGenerated) by Computer, ComputerIP, RemoteIPCountry,
ComputerEnvironment, OSType, OSMajorVersion, OSMinorVersion, SubscriptionId, TenantId
```

```
// Microsoft 365 Defender - Device Information
DeviceInfo
| where DeviceName == "contosohost" and isnotempty(OSPlatform)
| project TenantId, DeviceName, PublicIP, IsAzureADJoined, OSPlatform, OSBuild,
OSArchitecture, LoggedOnUsers
```

```
// Microsoft 365 Defender - Hostname based on Private IP addresses
DeviceNetworkInfo
| mv-expand IPAddresses
| extend IPAddress = tostring(parse_json(IPAddresses).IPAddress)
| where IPAddress== '10.0.0.100'
| project DeviceName, NetworkAdapterType, TunnelType, MacAddress
```

Query Parameterization

- Scalar Datatype

Dynamic – array/list

String Operators

in~ - Case Insensitive Match

has_any – check for whole terms (not substrings) in values

e.g. VmSize : Standard_m416

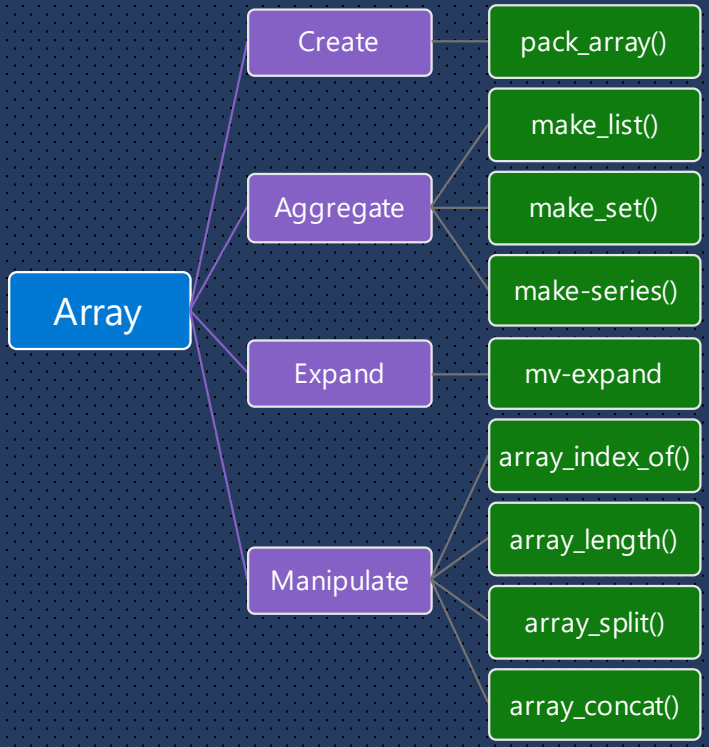
contains – search for substring

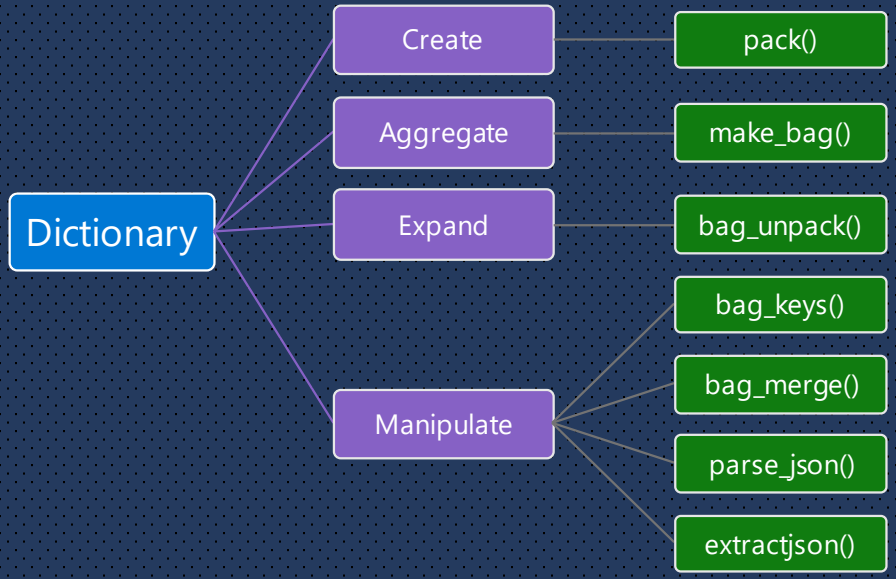
e.g. contains "admin" – abcadmin123

```
// Query Parameterization - Dynamic List - in~ operator - AWS CloudTrail Log Cleared
let timeframe = 1d;
let EventNameList = dynamic(["UpdateTrail","DeleteTrail","StopLogging","DeleteFlowLogs","DeleteEventBus"]);
AWSCloudTrail
| where TimeGenerated > ago(timeframe)
| where EventName in~ (EventNameList)
| limit 100
```

```
// Query Parameterization - Dynamic List - has_any operator - Azure Expensive Computes
let timeframe = 1d;
let tokens = dynamic(["416","208","128","120","96","80","72","64","48","44","40","g5","gs5","g4","gs4","nc12","nc24","nv12"]);
let operationList = dynamic(["Create or Update Virtual Machine", "Create Deployment"]);
AzureActivity
| where TimeGenerated >= ago(timeframe)
| where OperationName in (operationList)
| where ActivityStatus == "Accepted"
| where isnotempty(Properties)
| extend vmSize = tolower(tostring(parse_json(tostring(parse_json(tostring(parse_json(tostring
(parse_json(Properties).responseBody)).properties)).hardwareProfile)).vmSize))
| where isnotempty(vmSize)
| where vmSize has_any (tokens)
| limit 100
```

Dynamic Datatypes





Datetime

- Convert to datetime datatypes
  - todatetime

```
// todatetime demo
let CustomLogs = datatable(TimeGenerated:string)
[
    "2020-10-23 01:00:00",
    "2020-10-24 02:00:00"
];
CustomLogs
| extend TimeGenerated1 = todatetime(TimeGenerated)
| getschema
```

Datetime Operators

- format_datetime – change the datetime value format
- Bin - Aggregate into bins per time window
- datetime_diff – difference between two datetime fields (into- days, min, hours, sec)

```
// datetime conversion demo
let CustomLogs = datatable(TimeGenerated:string)
[
    "2020-10-23 01:00:00",
    "2020-10-24 02:00:00"
];
CustomLogs
| extend TimeGenerated1 = todatetime(TimeGenerated)
| extend Day = format_datetime(TimeGenerated1, "yyyy-MM-dd")
```

Regex Extraction

KQL Operators

- matches regex
- extract_all()
- mv-apply

```
// matches regex demo - Cisco - firewall block but success logon to Azure AD
let PrivateIPregex = @'^127\.|^10\.|^172\.1[6-9]\.|^172\.2[0-9]\.|^172\.3[0-1]\.|^192\.168\.';
let endtime = 1d;
CommonSecurityLog
| where TimeGenerated >= ago(endtime)
| where DeviceVendor =~ "Cisco"
| where DeviceAction =~ "denied"
| extend SourceIPType = iff(SourceIP matches regex PrivateIPregex,"private" ,"public" )
| where SourceIPType == "public"
| summarize count() by SourceIP
| join (
  // Successful signins from IPs blocked by the firewall solution are suspect
  // Include fully successful sign-ins, but also ones that failed only at MFA stage
  // as that supposes the password was sucessfully guessed.
    SigninLogs
    | where ResultType in ("0", "50074", "50076")
    ) on $left.SourceIP == $right.IPAddress
| limit 100
```

```
// Extract key value pair from AdditionalExtension field in CommonSecurityLog
let CommonSecurityLog = datatable (DeviceVendor: string, AdditionalExtensions: string)
[
"ZScaler", "country=United States;sourceAddress=10.10.10.10;sourcehostname=http://abc.ac.com;deviceTranslatedPort=60095;tunnelType=IPSEC;dnat=N
"Fortinet", "FortinetFortiGatelogid=1059028704;cat=ute:app-ctrl;FortinetFortiGatesubtype=app-ctrl;FortinetFortiGateeventtype=signature;Fortinet
"Palo Alto Networks", "cat=general;PanOSDGl1=0;PanOSDGl2=0;PanOSDGl3=0;PanOSDGl4=0;PanOSVsysName=;PanOSActionFlags=0x0"
];
CommonSecurityLog
| extend AdditionalExtensions = extract_all(@"(?P<key>\w+)=(?P<value>[a-zA-Z0-9-_:/@. ]+)", dynamic(["key","value"]), AdditionalExtensions)
| mv-apply AdditionalExtensions on (
summarize AdditionalExtensionsParsed = make_bag(pack(tostring(AdditionalExtensions[0]), AdditionalExtensions[1]))
)
```

AdditionalExtensionsParsed	{"country":"United States","sourceAddres
avgduration	143
country	United States
cs6label	threatname
destCountry	Italy
deviceTranslatedPort	60095
dnat	No
reason	Allow DNS
sourceAddress	10.10.10.10
sourcehostname	http://abc.ac.com
stateful	Yes
tunnelType	IPSEC

Functions

· To use log query in another query

Documentation:

Using functions in Azure Monitor log queries

Blog – Using KQL functions to speed up analysis

```
// Function Name - GetAllAlertsOnHost
// Source - https://github.com/Azure/Azure-Sentinel/blob/master/Exploration%20Queries/InputEntity_Host/AlertsOnHost.txt
let GetAllAlertsOnHost = (suspiciousEventTime:datetime, v_Host:string){
//-3d and +6h as some alerts fire after accumulation of events
let v_StartTime = suspiciousEventTime-3d;
let v_EndTime = suspiciousEventTime+6h;
SecurityAlert
| where TimeGenerated between (v_StartTime .. v_EndTime)
// expand JSON properties
| extend Extprop = parsejson(ExtendedProperties)
| extend Computer = toupper(tostring(Extprop["Compromised Host"]))
| where Computer contains v_Host
| project TimeGenerated, AlertName, Computer, ExtendedProperties
};
// change datetime value and hostname value below
GetAllAlertsOnHost(datetime('2020-10-23T06:00:00.000'), toupper("VICTIM00"))
```

Is there a built-in function already ?

Optimized performance to run at scale

- parse_csv()
- parse_json()
- parse_xml()
- parse_ipv4()
- ip4_is_match()
- parse_ipv6()
- parse_path()
- parse_url()
- parse_useragent()

```
// parse_path demo
let SecurityEvent = datatable (EventID: string, ShareLocalPath: string)
[
"5145",@"\\shared\users\temp\file.txt.gz",
"5145",@"\\shared\users\temp\bad.exe",
"5145",@"\\shared\users\temp\script.ps1"
];
SecurityEvent
| where EventID == 5145
| extend ShareLocalPathParsed = parse_path(ShareLocalPath)
| extend extension = tostring(parse_json(ShareLocalPathParsed).Extension),
 FileName = tostring(parse_json(ShareLocalPathParsed).Filename),
 DirName = tostring(parse_json(ShareLocalPathParsed).DirectoryName)
```

```
// ipv4_is_match with lookup demo
let lookup = dynamic (["13.66.60.119/32","13.66.143.220/30","13.66.202.14/32"]);
let AzureSubnetMatchedIPs=materialize(
CommonSecurityLog
| where TimeGenerated > ago(4h)
| mv-apply l=lookup to typeof(string) on
(
where ipv4_is_match (DestinationIP, l)
)
| project-away l);
AzureSubnetMatchedIPs
| limit 100
```

```
// Windows XML Parsing of Dynamic Field - EventData
SecurityEvent
| where TimeGenerated > ago(4h)
| extend EventData = parse_xml(EventData).EventData.Data
| mv-expand bagexpansion=array EventData
| extend EventName=tostring(EventData['@Name']), EventValue=EventData['#text']
| evaluate pivot(EventName, any(EventValue), TimeGenerated, EventID)
| limit 100
```

Externaldata

- Externaldata

```
// External data - Github Feed
let covidIndicators = (externaldata(TimeGenerated:datetime, FileHashValue:string, FileHashType: string )
[@"https://raw.githubusercontent.com/Azure/Azure-Sentinel/master/Sample%20Data/Feeds/Microsoft.Covid19.Indicators.csv"]
with (format="csv"));
covidIndicators
```

Blog with use cases:

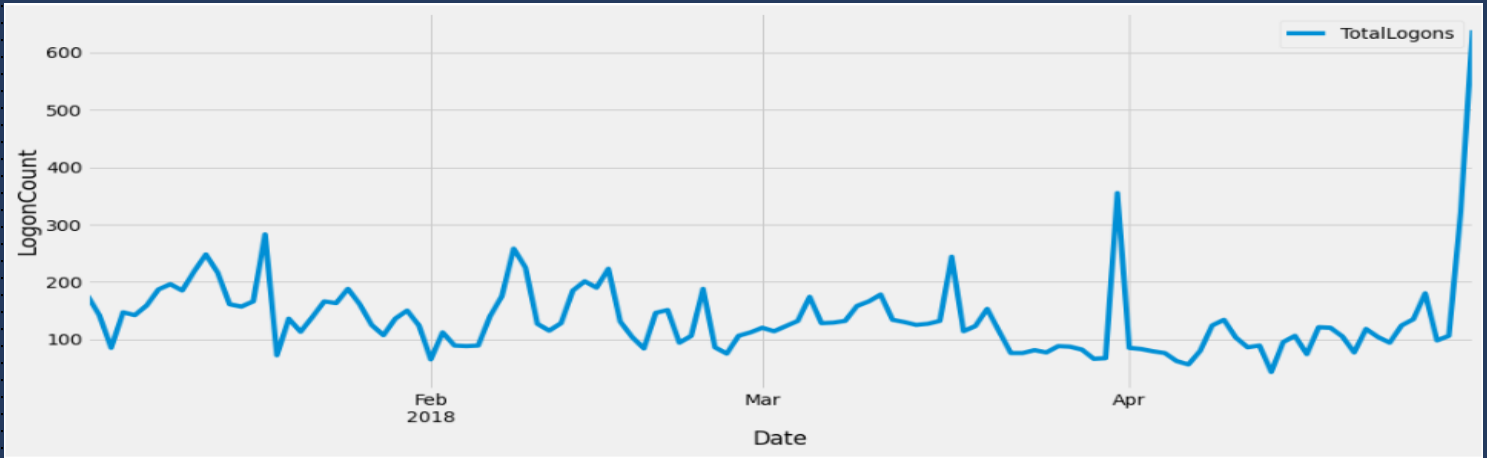
Using External data sources to enrich network logs using Azure Storage and KQL

```
// Externaldata - Azure IP ranges feed. Link is not static and gets expired as new content arrives
let AzureIPRangesPublicCloud = (externaldata(changeNumber:string, cloud:string, values: dynamic)
[@"https://download.microsoft.com/download/7/1/D/71D86715-5596-4529-9B13-DA13A5DE5B63/ServiceTags_Public_20201019.json"]
with (format="multijson"));
let AzureSubnetRangeAllowlist = AzureIPRangesPublicCloud
| mv-expand values
| extend addressPrefixes = parse_json(parse_json(values).properties).addressPrefixes;
AzureSubnetRangeAllowlist
```

Time Series Analysis

- make-series – Convert the data into Time Series format
- series_decompose() – Decompose the time series data into seasonal, trend and residual
- series_decompose_anomalies() – Apply Tuckey`s algo to find anomalies – Spikes and dips

Blog – Time Series analysis and its applications in Security



Anomaly detections for Applied Time Series Analysis

EVENT NAME	TYPE OF DATA SOURCE	EVENT TYPES	MITRE TACTICS
Logon Counts	Host	WEF- 4624, 4625 Syslog- SSH Auth, Auditd - Logons	Initial Access, Lateral Movement
Process Execution Frequency	Host	WEF – 4688 Auditd- execve events	Execution, Persistence
Outbound Data Transfer	Network	Firewall, Proxy Logs	Command and Control, Exfiltration
Logon Counts	Cloud	Azure AAD Signin Logs, AWS Console Logons	Initial Access, Lateral Movement
Secret Access Events from Vaults	Cloud	Keyvault, KMS	Credential Access
Data Transferred size from Storage Blobs	Cloud	Blob storage, S3 Object Access Logs	Collection, Exfiltration
API Event Execution frequency	Cloud	AWS	Initial Access, Persistence

Threat Hunting Use case – Time Series

```
let starttime = 14d;
let endtime = 1d;
let timeframe = 1h;
let TotalEventsThreshold = 5;
let ExeList = dynamic(["powershell.exe","cmd.exe","wmic.exe","psexec.exe","cacls.exe","rundll.exe"]);
```

1

```
let TimeSeriesData =
SecurityEvent
| where EventID == 4688 | extend Process = tolower(Process)
| where TimeGenerated between (startofday(ago(starttime))..startofday(ago(endtime)))
| where Process in (ExeList)
| project TimeGenerated, Computer, AccountType, Account, Process
| make-series Total=count() on TimeGenerated from ago(starttime) to ago(endtime) step timeframe by Process;
```

2

```
let TimeSeriesAlerts = TimeSeriesData
| extend (anomalies, score, baseline) = series_decompose_anomalies(Total, 1.5, -1, 'linefit')
| mv-expand Total to typeof(double), TimeGenerated to typeof(datetime), anomalies to typeof(double), score to typeof(double), baseline to typeof(long)
| where anomalies > 0
| project Process, TimeGenerated, Total, baseline, anomalies, score
| where Total > TotalEventsThreshold;
```

3

```
TimeSeriesAlerts
| join (
SecurityEvent
| where EventID == 4688 | extend Process = tolower(Process)
| summarize CommandlineCount = count() by bin(TimeGenerated, 1h), Process, CommandLine, Computer, Account
) on Process, TimeGenerated
| project AnomalyHour = TimeGenerated, Computer, Account, Process, CommandLine, CommandlineCount, Total, baseline, anomalies, score
| extend timestamp = AnomalyHour, AccountCustomEntity = Account, HostCustomEntity = Computer
```

4

1. Variable Declaration
2. Time Series Data Conversion
3. Anomaly Detection
4. Investigation of Anomalies

Threat Hunting Use Case: Network Beaconing

- Network Beaconing via Intra-request Time Deltas
- Reference/Previous Work :
  - Threat hunting Project
  - Flare by Austin Taylor

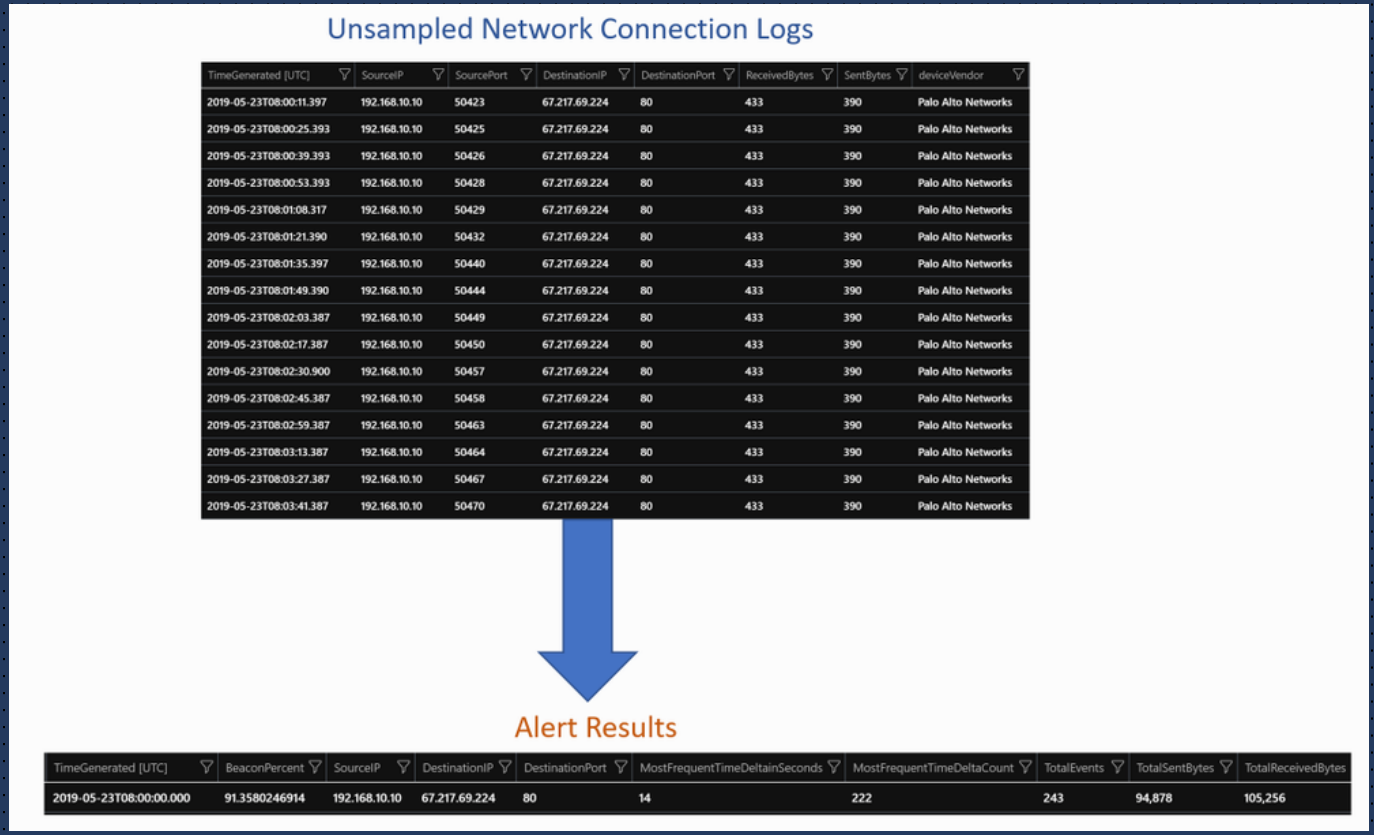
Tech community Blog :

Detect Network beaconing via Intra-Request time delta patterns in Azure Sentinel

Threat Hunting Use Case: Network Beaconing

```
let starttime = 2d;
let endtime = 1d;
let TimeDeltaThreshold = 10;
let TotalEventsThreshold = 15;
let PercentBeaconThreshold = 80;
let PrivateIPregex = @'^127\.|^10\.|^172\.1[6-9]\.|^172\.2[0-9]\.|^172\.3[0-1]\.|^192\.168\.';
let DestIPList = CommonSecurityLog
| where DeviceVendor == "Palo Alto Networks" and Activity == "TRAFFIC"
| where TimeGenerated between (ago(starttime)..ago(endtime))
| extend DestinationIPType = iff(DestinationIP matches regex PrivateIPregex,"private" ,"public" )
| where DestinationIPType == "public"
| summarize dcount(SourceIP) by DestinationIP
| where dcount_SourceIP < 5
| distinct DestinationIP;
CommonSecurityLog
| where DeviceVendor == "Palo Alto Networks" and Activity == "TRAFFIC"
| where TimeGenerated between (ago(starttime)..ago(endtime))
| where DestinationIP in ((DestIPList))
| project TimeGenerated, DeviceName, SourceUserID, SourceIP, SourcePort, DestinationIP, DestinationPort, ReceivedBytes, SentBytes
| sort by SourceIP asc,TimeGenerated asc, DestinationIP asc, DestinationPort asc
| serialize
| extend nextTimeGenerated = next(TimeGenerated, 1), nextSourceIP = next(SourceIP, 1)
| extend TimeDeltainSeconds = datetime_diff('second',nextTimeGenerated,TimeGenerated)
| where SourceIP == nextSourceIP
//Whitelisting criteria/ threshold criteria
| where TimeDeltainSeconds > TimeDeltaThreshold
| project TimeGenerated, TimeDeltainSeconds, DeviceName, SourceUserID, SourceIP, SourcePort, DestinationIP, DestinationPort, ReceivedBytes, SentBytes
| summarize count(), sum(ReceivedBytes), sum(SentBytes), make_list(TimeDeltainSeconds)
by TimeDeltainSeconds, bin(TimeGenerated, 1h), DeviceName, SourceUserID, SourceIP, DestinationIP, DestinationPort
| summarize (MostFrequentTimeDeltaCount, MostFrequentTimeDeltainSeconds) = arg_max(count_, TimeDeltainSeconds), TotalEvents=sum(count_)
, TotalSentBytes = sum(sum_SentBytes), TotalReceivedBytes = sum(sum_ReceivedBytes)
by bin(TimeGenerated, 1h), DeviceName, SourceUserID, SourceIP, DestinationIP, DestinationPort
| where TotalEvents > TotalEventsThreshold
| extend BeaconPercent = MostFrequentTimeDeltaCount/toreal(TotalEvents) * 100
| where BeaconPercent > PercentBeaconThreshold
| extend timestamp = TimeGenerated, IPCustomEntity = DestinationIP, AccountCustomEntity = SourceUserID, HostCustomEntity = DeviceName
```

Sample Output – Network Beaconing



KQL Programmatic Interfaces

- KqlMagic
- Msticpy (Ian Hellen, Pete Bryan)
  - Query Provider Interface for Connected Data source
  - List of Data Queries – Readthedocs
    - Azure Sentinel
    - Microsoft 365 Defender
    - SecurityGraph
    - Splunk
  - Functions:
    - list_queries()
    - Execute a Query
      - Built-in :: qry_prov.<DataSource>.<QueryName>
      - Ad-Hoc:: exec_query(KQLQuery)
    - Import_query_file – import_query_file()
    - Splitting Query Execution into Chunks

MSTICpy demo – Query provider

Notebook Demo – Data Queries

Recorded Gif :

https://github.com/ashwin-patil/blue-teaming-with-kql/blob/main/images/nbdemo.gif

KQL Playground

- Demo Instances :
  - Log Analytics : http://aka.ms/lademo
  - Azure Data Explorer- https://dataexplorer.azure.com/clusters/help

Azure Sentinel2Go: Cyb3rWard0g /Roberto Rodriguez

- Scenario Driven Azure Sentinel Labs – One click to deploy
  - Win10
  - Win10+DC
  - Win10 + PAN
  - Ubuntu
- https://github.com/OTRF/Azure-Sentinel2Go

KQL Trainings

- Pluralsight Free Trainings
  - Kusto Query Language (KQL) from Scratch
  - Microsoft Azure Data Explorer – Advanced KQL

- Ninja Trainings
  - Azure Sentinel Ninja Training – Module 7 : KQL
  - Microsoft Defender Ninja Training –
    - SecOps Intermediate Module 2 : Advanced Hunting
    - SecOps Expert Module 4 : Advanced Hunting

KQL Query Resources

- GitHub Repositories
  - Azure Sentinel
  - Microsoft 365 Defender Hunting Queries

- Community Guides/Resources
  - Stackoverflow – KQL tag
  - BlueTeamLabs- sentinel-attack by @netvert and contributors
  - KQL Internals by @DebugPrivilege
  - Kusto King by @castello_johnny

  and more queries shared by community on GitHub, Twitter, blogs etc.

Conclusion

- KQL provides powerful data analysis framework to analysts
- Mastering KQL skills supercharges Defender.
- Leverage the community-built resources, queries.
- Share queries with the community
- Contribute to Azure Sentinel GitHub.