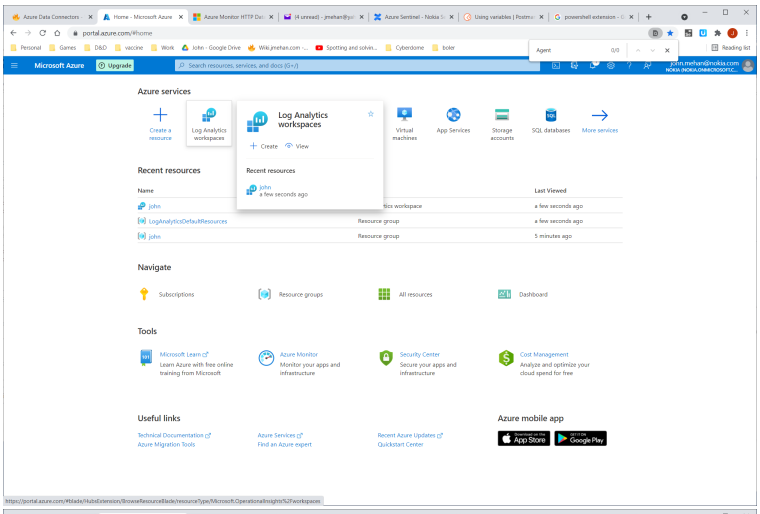
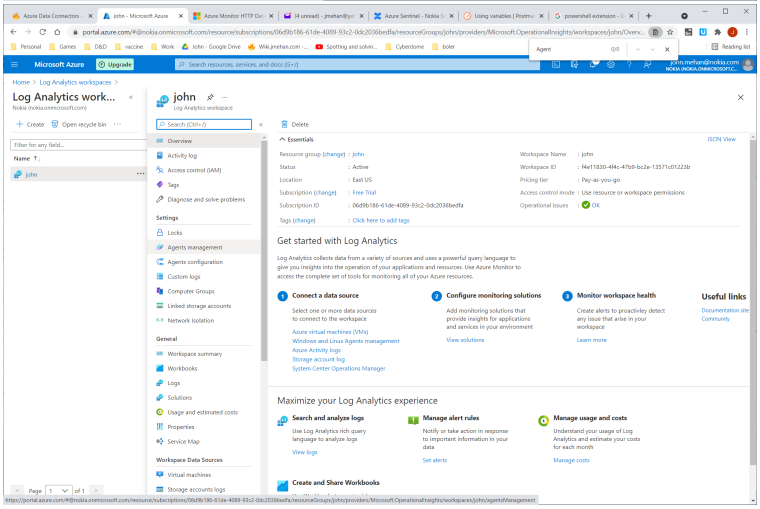
Azure Data Connectors

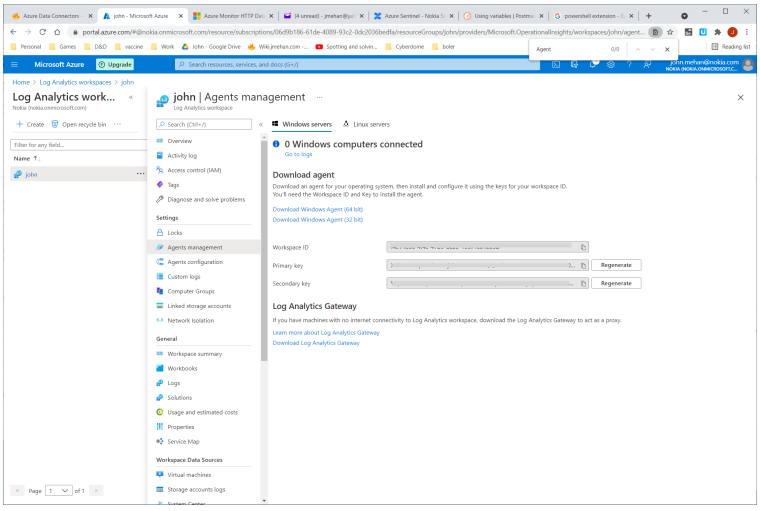
Your Credentials

To determine your credentials in Azure:

- locate your Log Analytics workspace.
- Select **Agents management**.
- To the right of **Workspace ID**, select the copy icon, and then paste the ID as the value of the **Customer ID** variable.
- To the right of **Primary Key**, select the copy icon, and then paste the ID as the value of the **Shared Key** variable.

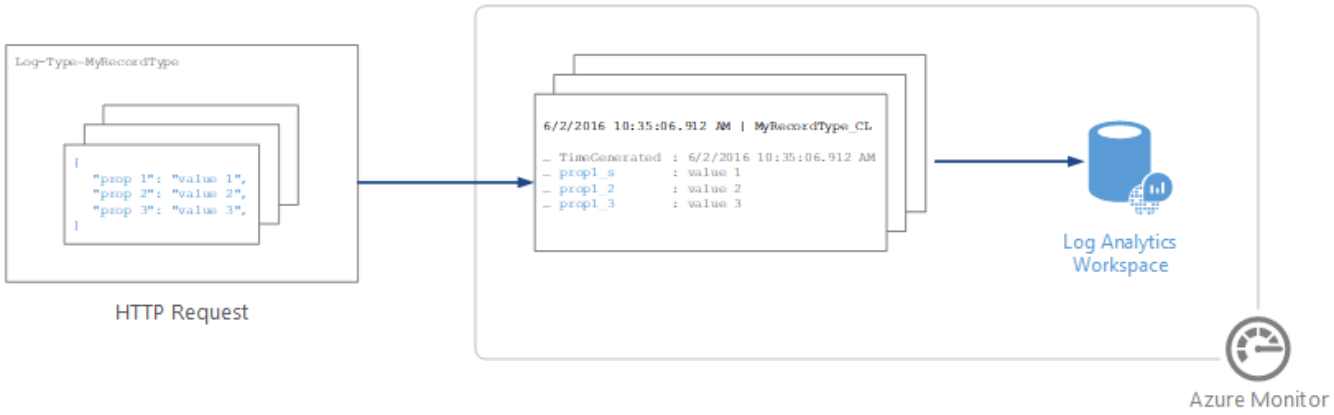






Azure HTTP Data Collector API

https://docs.microsoft.com/en-us/azure/azure-monitor/logs/data-collector-api



Authorization Header

Any request to the Azure Monitor HTTP Data Collector API must include an authorization header. To authenticate a request, you must sign the request with either the primary or the secondary key for the workspace that is making the request. Then, pass that signature as part of the request.

Here's the format for the authorization header:

```
Authorization: SharedKey <WorkspaceID>:<Signature>
```

WorkspaceID is the unique identifier for the Log Analytics workspace. *Signature* is a Hash-based Message Authentication Code (HMAC) that is constructed from the request and then computed by using the SHA256 algorithm. Then, you encode it by using Base64 encoding.

Use this format to encode the **SharedKey** signature string:

```
StringToSign = VERB + "\n" +
                  Content-Length + "\n" +
                  Content-Type + "\n" +
                  "x-ms-date:" + x-ms-date + "\n" +
                  "/api/logs";
```

Here's an example of a signature string:

```
POST\n1024\napplication/json\nx-ms-date:Mon, 04 Apr 2016 08:00:00 GMT\n/api/logs
```

When you have the signature string, encode it by using the HMAC-SHA256 algorithm on the UTF-8-encoded string, and then encode the result as Base64. Use this format:

```
Signature=Base64(HMAC-SHA256(UTF8(StringToSign)))
```

Request Body

The body of the message must be in JSON.

It must include one or more records with the property name and value pairs in the following format. The property name can only contain letters, numbers, and underscore (_).

```
JSON
[
    {
        "property 1": "value1",
        "property 2": "value2",
        "property 3": "value3",
        "property 4": "value4"
    }
]
```

Sample Script/Program

Sample powershell script to push data to your workspace.

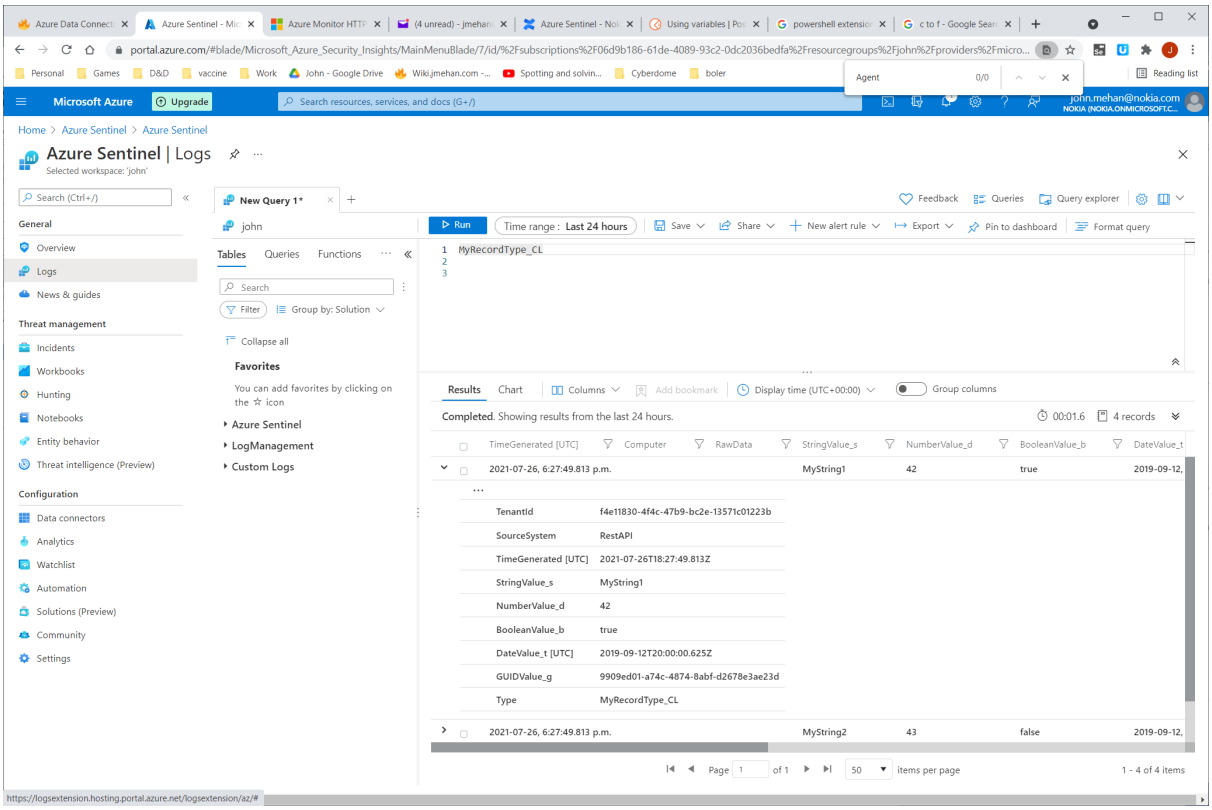
LogType: **MyRecordType_CL**

samplePush.ps1

```
# Replace with your Workspace ID
$CustomerId = "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"

# Replace with your Primary Key
$SharedKey = "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"

# Specify the name of the record type that you'll be creating
$LogType = "MyRecordType"

# You can use an optional field to specify the timestamp from the data. If the time field is not specified,
Azure Monitor assumes the time is the message ingestion time
$TimeStampField = ""


# Create two records with the same set of properties to create
$json = @"
[{  "StringValue": "MyString1",
    "NumberValue": 42,
    "BooleanValue": true,
    "DateValue": "2019-09-12T20:00:00.625Z",
    "GUIDValue": "9909ED01-A74C-4874-8ABF-D2678E3AE23D"
},
{   "StringValue": "MyString2",
    "NumberValue": 43,
    "BooleanValue": false,
    "DateValue": "2019-09-12T20:00:00.625Z",
    "GUIDValue": "8809ED01-A74C-4874-8ABF-D2678E3AE23D"
}]
"@

# Create the function to create the authorization signature
Function Build-Signature ($customerId, $sharedKey, $date, $contentLength, $method, $contentType, $resource)
{
    $xHeaders = "x-ms-date:" + $date
    $stringToHash = $method + "`n" + $contentLength + "`n" + $contentType + "`n" + $xHeaders + "`n" + $resource

    $bytesToHash = [Text.Encoding]::UTF8.GetBytes($stringToHash)
    $keyBytes = [Convert]::FromBase64String($sharedKey)

    $sha256 = New-Object System.Security.Cryptography.HMACSHA256
    $sha256.Key = $keyBytes
    $calculatedHash = $sha256.ComputeHash($bytesToHash)
    $encodedHash = [Convert]::ToBase64String($calculatedHash)
    $authorization = 'SharedKey {0}:{1}' -f $customerId,$encodedHash
    return $authorization
```

```
}


# Create the function to create and post the request
Function Post-LogAnalyticsData($customerId, $sharedKey, $body, $logType)
{
    $method = "POST"
    $contentType = "application/json"
    $resource = "/api/logs"
    $rfc1123date = [DateTime]::UtcNow.ToString("r")
    $contentLength = $body.Length
    $signature = Build-Signature `
        -customerId $customerId `
        -sharedKey $sharedKey `
        -date $rfc1123date `
        -contentLength $contentLength `
        -method $method `
        -contentType $contentType `
        -resource $resource
    $uri = "https://" + $customerId + ".ods.opinsights.azure.com" + $resource + "?api-version=2016-04-01"

    $headers = @{
        "Authorization" = $signature;
        "Log-Type" = $logType;
        "x-ms-date" = $rfc1123date;
        "time-generated-field" = $TimeStampField;
    }

    $response = Invoke-WebRequest -Uri $uri -Method $method -ContentType $contentType -Headers $headers -Body
$body -UseBasicParsing
    return $response.StatusCode

}

# Submit the data to the API endpoint
Post-LogAnalyticsData -customerId $customerId -sharedKey $sharedKey -body ([System.Text.Encoding]::UTF8.GetBytes
($json)) -logType $logType
```

Querying Submitted Data

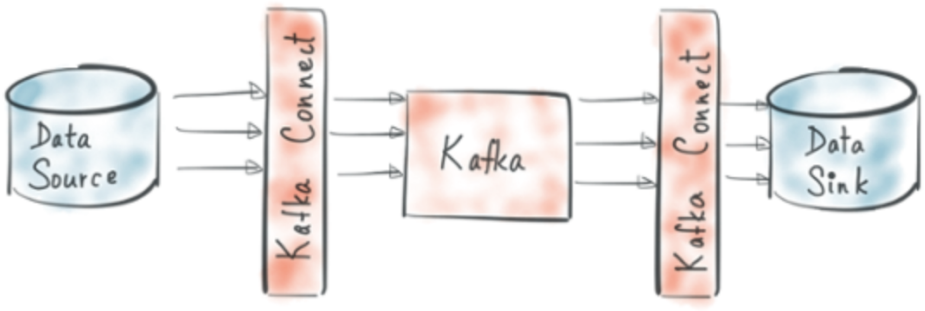


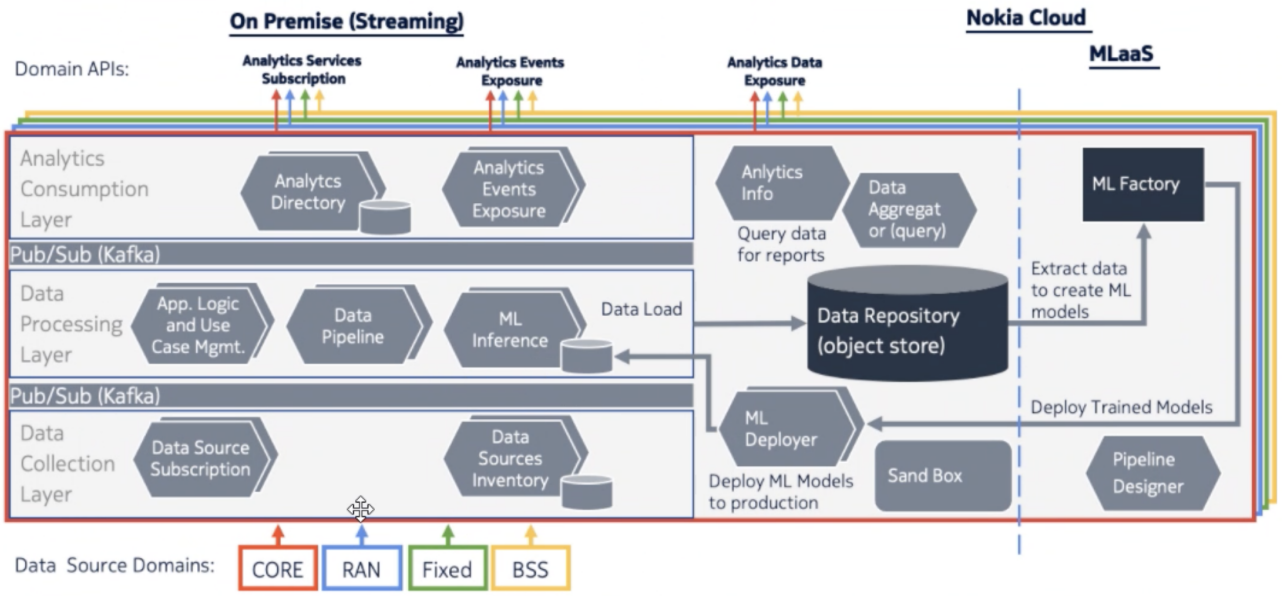
Azure Arc

https://docs.microsoft.com/en-us/azure/architecture/hybrid/arc-hybrid-kubernetes

Kafka Connect with Azure Log Analytics Sink Connector

https://www.confluent.de/hub/chaitalisagesh/kafka-connect-log-analytics





Log Analytics Agent for Linux

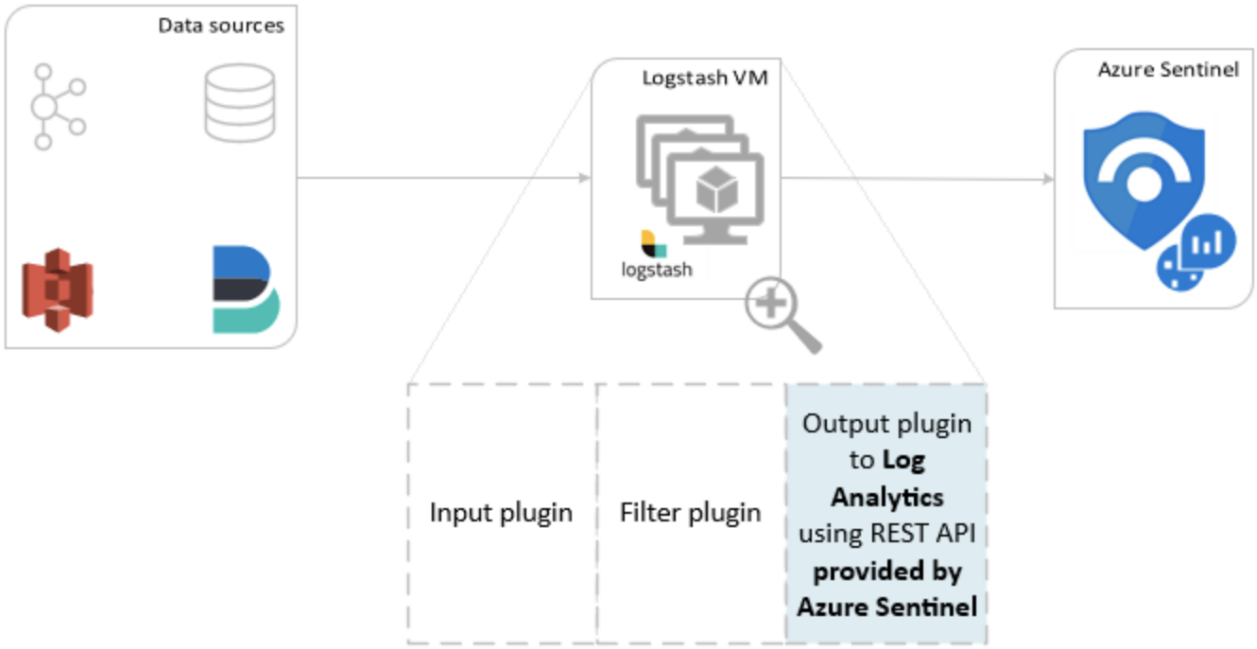
https://docs.microsoft.com/en-us/azure/azure-monitor/agents/agent-linux

Pushes data to Azure Data Collector API.

Logstash

https://docs.microsoft.com/en-us/azure/sentinel/connect-logstash

Pushes data to Azure Data Collector API.



"The components for log parsing are different per logging tool. Fluentd uses standard built-in parsers (JSON, regex, csv etc.) and **Logstash** uses plugins for this. **This makes Fluentd favorable over Logstash,** because it does not need extra plugins installed, making the architecture more complex and more prone to errors"

Fluent-bit

https://docs.fluentbit.io/manual/pipeline/outputs/azure

Pushes data to Azure Data Collector API.

References

Reference	URL
Azure HTTP Data Collector API	https://docs.microsoft.com/en-us/azure/azure-monitor/logs/data-collector-api
Azure Log Analytics Sink Connector	https://www.confluent.de/hub/chaitalisagesh/kafka-connect-log-analytics
Log Analytics Agent for Linux	https://docs.microsoft.com/en-us/azure/azure-monitor/agents/agent-linux
Logstash	https://docs.microsoft.com/en-us/azure/sentinel/connect-logstash
Fluent-bit	https://docs.fluentbit.io/manual/pipeline/outputs/azure
Kubernetes Logging: Comparing Fluentd vs. Logstash	https://platform9.com/blog/kubernetes-logging-comparing-fluentd-vs-logstash/#:~:text=The%20components%20for%20log%20parsing,and%20more%20prone%20to%20errors.