Sumo Logic - Log Searches

rjury-sumo.github.io/sumologic-query-examples/logs-search-viewer.html

/Active Threats: AWS APIs/Threats by Actor

```
_sourceCategory={{CloudTrailLogsdatasource}}  sourceIPAddress !("s3.amazonaws.com")
| json "eventTime", "eventName", "eventSource", "awsRegion", "sourceIPAddress",
"errorCode", "userAgent" as event_time, event.action, eventSource, cloud.region,
client.ip, event.outcome, user_agent nodrop
| json "userIdentity.accountId", "userIdentity.type" as cloud.account.id,
user_identity_type nodrop
| json field=_raw "requestParameters.dBInstanceIdentifier",
"requestParameters.instancesSet.items[0].instanceId" as db_instance_id, cloud_instance_id
nodrop

| parse regex "\"(?i)userName\":\"(?<user_name>.*?)\"" nodrop
| parse "\"userId\":\"*\"" as user_id nodrop
| if (user_name="", user_id, user_name) as user.name
| parse field=eventSource "*." as cloud.service.name
| parse "\"accessKeyId\":\"*\"" as accessKeyId nodrop

| lookup type, actor, raw, threatlevel as malicious_confidence, threat from
sumo://threat/cs on threat=client.ip
| where type="ip_address"

| if (malicious_confidence = "low", 1, 0) as risk.static_level
| if (malicious_confidence = "medium", 2, risk.static_level) as risk.static_level
| if (malicious_confidence = "high", 3, risk.static_level) as risk.static_level

// handle empty values
| if(isEmpty(db_instance_id),cloud_instance_id,db_instance_id) as cloud.instance.id
| if(isEmpty(cloud.instance.id),"NA", cloud.instance.id) as cloud.instance.id
| if (isEmpty(event.outcome), "Success", event.outcome) as event.outcome
| if (isEmpty(user.name), "NA", user.name) as user.name
| if (isEmpty(actor), "Unassigned", actor) as actor
| if(isEmpty(cloud.account.id), "NA", cloud.account.id) as cloud.account.id

// global filters
| where if ("{{cloud.account.id}}" = "*", true, cloud.account.id matches "
{{cloud.account.id}}") AND if ("{{cloud.region}}" = "*", true, cloud.region matches "
{{cloud.region}}") AND if ("{{event.action}}" = "*", true, event.action matches "
{{event.action}}") AND if ("{{event.outcome}}" = "*", true, event.outcome matches "
{{event.outcome}}") AND if ("{{user.name}}" = "*", true, user.name matches "
{{user.name}}") AND if ("{{client.ip}}" = "*", true, client.ip matches "{{client.ip}}")
AND if ("{{cloud.service.name}}" = "*", true, cloud.service.name matches "
{{cloud.service.name}}") AND if ("{{cloud.instance.id}}" = "*", true, cloud.instance.id
matches "{{cloud.instance.id}}") AND if ("{{threat.group.name}}" = "*", true, actor
matches "{{threat.group.name}}") AND if ("{{risk.static_level}}" = "*", true,
risk.static_level >= toInt("{{risk.static_level}}"))

| json field=raw "malware_families[*]" as threat_malware_families nodrop
| json field=raw "last_updated" as last_updated nodrop
| formatDate(fromseconds(last_updated), "MM-dd-yyyy") as threat_last_updated
| json field=raw "labels[*].name" as label_name nodrop
| replace(label_name, "\\/","->") as label_name
| replace(label_name, "\""," ") as label_name
| count by actor, label_name, risk.static_level | sort by _count, actor, label_name,
risk.static_level asc
```

/Active Threats: AWS APIs/Threats by Events and Result

```
_sourceCategory={{CloudTrailLogsdatasource}}  sourceIPAddress !("s3.amazonaws.com")
| json "eventTime", "eventName", "eventSource", "awsRegion", "sourceIPAddress",
"errorCode", "userAgent" as event_time, event.action, eventSource, cloud.region,
client.ip, event.outcome, user_agent nodrop
| json "userIdentity.accountId", "userIdentity.type" as cloud.account.id,
user_identity_type nodrop
| json field=_raw "requestParameters.dBInstanceIdentifier",
"requestParameters.instancesSet.items[0].instanceId" as db_instance_id, cloud_instance_id
nodrop

| parse regex "\"(?i)userName\":\"(?<user_name>.*?)\"" nodrop
| parse "\"userId\":\"*\"" as user_id nodrop
| if (user_name="", user_id, user_name) as user.name
| parse field=eventSource "*." as cloud.service.name
| parse "\"accessKeyId\":\"*\"" as accessKeyId nodrop

| lookup type, actor, raw, threatlevel as malicious_confidence, threat from
sumo://threat/cs on threat=client.ip
| where type="ip_address"

| if (malicious_confidence = "low", 1, 0) as risk.static_level
| if (malicious_confidence = "medium", 2, risk.static_level) as risk.static_level
| if (malicious_confidence = "high", 3, risk.static_level) as risk.static_level

// handle empty values
| if(isEmpty(db_instance_id),cloud_instance_id,db_instance_id) as cloud.instance.id
| if(isEmpty(cloud.instance.id),"NA", cloud.instance.id) as cloud.instance.id
| if (isEmpty(event.outcome), "Success", event.outcome) as event.outcome
| if (isEmpty(user.name), "NA", user.name) as user.name
| if (isEmpty(actor), "Unassigned", actor) as actor
| if(isEmpty(cloud.account.id), "NA", cloud.account.id) as cloud.account.id

// global filters
| where if ("{{cloud.account.id}}" = "*", true, cloud.account.id matches "
{{cloud.account.id}}") AND if ("{{cloud.region}}" = "*", true, cloud.region matches "
{{cloud.region}}") AND if ("{{event.action}}" = "*", true, event.action matches "
{{event.action}}") AND if ("{{event.outcome}}" = "*", true, event.outcome matches "
{{event.outcome}}") AND if ("{{user.name}}" = "*", true, user.name matches "
{{user.name}}") AND if ("{{client.ip}}" = "*", true, client.ip matches "{{client.ip}}")
AND if ("{{cloud.service.name}}" = "*", true, cloud.service.name matches "
{{cloud.service.name}}") AND if ("{{cloud.instance.id}}" = "*", true, cloud.instance.id
matches "{{cloud.instance.id}}") AND if ("{{threat.group.name}}" = "*", true, actor
matches "{{threat.group.name}}") AND if ("{{risk.static_level}}" = "*", true,
risk.static_level >= toInt("{{risk.static_level}}"))

| json field=raw "malware_families[*]" as threat_malware_families nodrop
| json field=raw "last_updated" as last_updated nodrop
| formatDate(fromseconds(last_updated), "MM-dd-yyyy") as threat_last_updated
| json field=raw "labels[*].name" as label_name nodrop
| replace(label_name, "\\/","->") as label_name
| replace(label_name, "\""," ") as label_name
| count by event.action, event.outcome
| transpose row event.action column event.outcome
```

/Active Threats: AWS APIs/Threats by Geo Location

```
_sourceCategory={{CloudTrailLogsdatasource}}  sourceIPAddress !("s3.amazonaws.com")
| json "eventTime", "eventName", "eventSource", "awsRegion", "sourceIPAddress",
"errorCode", "userAgent" as event_time, event.action, eventSource, cloud.region,
client.ip, event.outcome, user_agent nodrop
| json "userIdentity.accountId", "userIdentity.type" as cloud.account.id,
user_identity_type nodrop
| json field=_raw "requestParameters.dBInstanceIdentifier",
"requestParameters.instancesSet.items[0].instanceId" as db_instance_id, cloud_instance_id
nodrop

| parse regex "\"(?i)userName\":\"(?<user_name>.*?)\"" nodrop
| parse "\"userId\":\"*\"" as user_id nodrop
| if (user_name="", user_id, user_name) as user.name
| parse field=eventSource "*." as cloud.service.name
| parse "\"accessKeyId\":\"*\"" as accessKeyId nodrop

| lookup type, actor, raw, threatlevel as malicious_confidence, threat from
sumo://threat/cs on threat=client.ip
| where type="ip_address"

| if (malicious_confidence = "low", 1, 0) as risk.static_level
| if (malicious_confidence = "medium", 2, risk.static_level) as risk.static_level
| if (malicious_confidence = "high", 3, risk.static_level) as risk.static_level

// handle empty values
| if(isEmpty(db_instance_id),cloud_instance_id,db_instance_id) as cloud.instance.id
| if(isEmpty(cloud.instance.id),"NA", cloud.instance.id) as cloud.instance.id
| if (isEmpty(event.outcome), "Success", event.outcome) as event.outcome
| if (isEmpty(user.name), "NA", user.name) as user.name
| if (isEmpty(actor), "Unassigned", actor) as actor
| if(isEmpty(cloud.account.id), "NA", cloud.account.id) as cloud.account.id

// global filters
| where if ("{{cloud.account.id}}" = "*", true, cloud.account.id matches "
{{cloud.account.id}}") AND if ("{{cloud.region}}" = "*", true, cloud.region matches "
{{cloud.region}}") AND if ("{{event.action}}" = "*", true, event.action matches "
{{event.action}}") AND if ("{{event.outcome}}" = "*", true, event.outcome matches "
{{event.outcome}}") AND if ("{{user.name}}" = "*", true, user.name matches "
{{user.name}}") AND if ("{{client.ip}}" = "*", true, client.ip matches "{{client.ip}}")
AND if ("{{cloud.service.name}}" = "*", true, cloud.service.name matches "
{{cloud.service.name}}") AND if ("{{cloud.instance.id}}" = "*", true, cloud.instance.id
matches "{{cloud.instance.id}}") AND if ("{{threat.group.name}}" = "*", true, actor
matches "{{threat.group.name}}") AND if ("{{risk.static_level}}" = "*", true,
risk.static_level >= toInt("{{risk.static_level}}"))

| json field=raw "malware_families[*]" as threat_malware_families nodrop
| json field=raw "last_updated" as last_updated nodrop
| formatDate(fromseconds(last_updated), "MM-dd-yyyy") as threat_last_updated
| json field=raw "labels[*].name" as label_name nodrop
| replace(label_name, "\\/","->") as label_name
| replace(label_name, "\""," ") as label_name
| lookup latitude, longitude from geo://location on ip = client.ip
| count by latitude, longitude, label_name, threat_last_updated
```

/Active Threats: AWS APIs/Threats by Resource

```
_sourceCategory={{CloudTrailLogsdatasource}}  sourceIPAddress !("s3.amazonaws.com")
| json "eventTime", "eventName", "eventSource", "awsRegion", "sourceIPAddress",
"errorCode", "userAgent" as event_time, event.action, eventSource, cloud.region,
client.ip, event.outcome, user_agent nodrop
| json "userIdentity.accountId", "userIdentity.type" as cloud.account.id,
user_identity_type nodrop
| json field=_raw "requestParameters.dBInstanceIdentifier",
"requestParameters.instancesSet.items[0].instanceId" as db_instance_id, cloud_instance_id
nodrop

| parse regex "\"(?i)userName\":\"(?<user_name>.*?)\"" nodrop
| parse "\"userId\":\"*\"" as user_id nodrop
| if (user_name="", user_id, user_name) as user.name
| parse field=eventSource "*." as cloud.service.name
| parse "\"accessKeyId\":\"*\"" as accessKeyId nodrop

| lookup type, actor, raw, threatlevel as malicious_confidence, threat from
sumo://threat/cs on threat=client.ip
| where type="ip_address"

| if (malicious_confidence = "low", 1, 0) as risk.static_level
| if (malicious_confidence = "medium", 2, risk.static_level) as risk.static_level
| if (malicious_confidence = "high", 3, risk.static_level) as risk.static_level

// handle empty values
| if(isEmpty(db_instance_id),cloud_instance_id,db_instance_id) as cloud.instance.id
| if(isEmpty(cloud.instance.id),"NA", cloud.instance.id) as cloud.instance.id
| if (isEmpty(event.outcome), "Success", event.outcome) as event.outcome
| if (isEmpty(user.name), "NA", user.name) as user.name
| if (isEmpty(actor), "Unassigned", actor) as actor
| if(isEmpty(cloud.account.id), "NA", cloud.account.id) as cloud.account.id

// global filters
| where if ("{{cloud.account.id}}" = "*", true, cloud.account.id matches "
{{cloud.account.id}}") AND if ("{{cloud.region}}" = "*", true, cloud.region matches "
{{cloud.region}}") AND if ("{{event.action}}" = "*", true, event.action matches "
{{event.action}}") AND if ("{{event.outcome}}" = "*", true, event.outcome matches "
{{event.outcome}}") AND if ("{{user.name}}" = "*", true, user.name matches "
{{user.name}}") AND if ("{{client.ip}}" = "*", true, client.ip matches "{{client.ip}}")
AND if ("{{cloud.service.name}}" = "*", true, cloud.service.name matches "
{{cloud.service.name}}") AND if ("{{cloud.instance.id}}" = "*", true, cloud.instance.id
matches "{{cloud.instance.id}}") AND if ("{{threat.group.name}}" = "*", true, actor
matches "{{threat.group.name}}") AND if ("{{risk.static_level}}" = "*", true,
risk.static_level >= toInt("{{risk.static_level}}"))

| json field=raw "malware_families[*]" as threat_malware_families nodrop
| json field=raw "last_updated" as last_updated nodrop
| formatDate(fromseconds(last_updated), "MM-dd-yyyy") as threat_last_updated
| json field=raw "labels[*].name" as label_name nodrop
| replace(label_name, "\\/","->") as label_name
| replace(label_name, "\"","  ") as label_name
| count by client.ip, event.action, user.name, cloud.account.id, cloud.region,
cloud.instance.id, cloud.service.name, user_identity_type, accessKeyId
| sum(_count) as total_threats by client.ip,
event.action,user_identity_type,user.name,accessKeyId , cloud.account.id, cloud.region,
cloud.instance.id, cloud.service.name
| sort by total_threats
```

/Active Threats: AWS APIs/Threats Count

```
_sourceCategory={{CloudTrailLogsdatasource}}  sourceIPAddress !("s3.amazonaws.com")
| json "eventTime", "eventName", "eventSource", "awsRegion", "sourceIPAddress",
"errorCode", "userAgent" as event_time, event.action, eventSource, cloud.region,
client.ip, event.outcome, user_agent nodrop
| json "userIdentity.accountId", "userIdentity.type" as cloud.account.id,
user_identity_type nodrop
| json field=_raw "requestParameters.dBInstanceIdentifier",
"requestParameters.instancesSet.items[0].instanceId" as db_instance_id, cloud_instance_id
nodrop

| parse regex "\"(?i)userName\":\"(?<user_name>.*?)\"" nodrop
| parse "\"userId\":\"*\"" as user_id nodrop
| if (user_name="", user_id, user_name) as user.name
| parse field=eventSource "*." as cloud.service.name
| parse "\"accessKeyId\":\"*\"" as accessKeyId nodrop

| lookup type, actor, raw, threatlevel as malicious_confidence, threat from
sumo://threat/cs on threat=client.ip
| where type="ip_address"

| if (malicious_confidence = "low", 1, 0) as risk.static_level
| if (malicious_confidence = "medium", 2, risk.static_level) as risk.static_level
| if (malicious_confidence = "high", 3, risk.static_level) as risk.static_level

// handle empty values
| if(isEmpty(db_instance_id),cloud_instance_id,db_instance_id) as cloud.instance.id
| if(isEmpty(cloud.instance.id),"NA", cloud.instance.id) as cloud.instance.id
| if (isEmpty(event.outcome), "Success", event.outcome) as event.outcome
| if (isEmpty(user.name), "NA", user.name) as user.name
| if (isEmpty(actor), "Unassigned", actor) as actor
| if(isEmpty(cloud.account.id), "NA", cloud.account.id) as cloud.account.id

// global filters
| where if ("{{cloud.account.id}}" = "*", true, cloud.account.id matches "
{{cloud.account.id}}") AND if ("{{cloud.region}}" = "*", true, cloud.region matches "
{{cloud.region}}") AND if ("{{event.action}}" = "*", true, event.action matches "
{{event.action}}") AND if ("{{event.outcome}}" = "*", true, event.outcome matches "
{{event.outcome}}") AND if ("{{user.name}}" = "*", true, user.name matches "
{{user.name}}") AND if ("{{client.ip}}" = "*", true, client.ip matches "{{client.ip}}")
AND if ("{{cloud.service.name}}" = "*", true, cloud.service.name matches "
{{cloud.service.name}}") AND if ("{{cloud.instance.id}}" = "*", true, cloud.instance.id
matches "{{cloud.instance.id}}") AND if ("{{threat.group.name}}" = "*", true, actor
matches "{{threat.group.name}}") AND if ("{{risk.static_level}}" = "*", true,
risk.static_level >= toInt("{{risk.static_level}}"))

| json field=raw "malware_families[*]" as threat_malware_families nodrop
| json field=raw "last_updated" as last_updated nodrop
| formatDate(fromseconds(last_updated), "MM-dd-yyyy") as threat_last_updated
| json field=raw "labels[*].name" as label_name nodrop
| replace(label_name, "\\/","->") as label_name
| replace(label_name, "\""," ") as label_name
| count
```

/Active Threats: AWS APIs/Threats Trend

```
_sourceCategory={{CloudTrailLogsdatasource}}  sourceIPAddress !("s3.amazonaws.com")
| json "eventTime", "eventName", "eventSource", "awsRegion", "sourceIPAddress",
"errorCode", "userAgent" as event_time, event.action, eventSource, cloud.region,
client.ip, event.outcome, user_agent nodrop
| json "userIdentity.accountId", "userIdentity.type" as cloud.account.id,
user_identity_type nodrop
| json field=_raw "requestParameters.dBInstanceIdentifier",
"requestParameters.instancesSet.items[0].instanceId" as db_instance_id, cloud_instance_id
nodrop

| parse regex "\"(?i)userName\":\"(?<user_name>.*?)\"" nodrop
| parse "\"userId\":\"*\"" as user_id nodrop
| if (user_name="", user_id, user_name) as user.name
| parse field=eventSource "*." as cloud.service.name
| parse "\"accessKeyId\":\"*\"" as accessKeyId nodrop

| lookup type, actor, raw, threatlevel as malicious_confidence, threat from
sumo://threat/cs on threat=client.ip
| where type="ip_address"

| if (malicious_confidence = "low", 1, 0) as risk.static_level
| if (malicious_confidence = "medium", 2, risk.static_level) as risk.static_level
| if (malicious_confidence = "high", 3, risk.static_level) as risk.static_level

// handle empty values
| if(isEmpty(db_instance_id),cloud_instance_id,db_instance_id) as cloud.instance.id
| if(isEmpty(cloud.instance.id),"NA", cloud.instance.id) as cloud.instance.id
| if (isEmpty(event.outcome), "Success", event.outcome) as event.outcome
| if (isEmpty(user.name), "NA", user.name) as user.name
| if (isEmpty(actor), "Unassigned", actor) as actor
| if(isEmpty(cloud.account.id), "NA", cloud.account.id) as cloud.account.id

// global filters
| where if ("{{cloud.account.id}}" = "*", true, cloud.account.id matches "
{{cloud.account.id}}") AND if ("{{cloud.region}}" = "*", true, cloud.region matches "
{{cloud.region}}") AND if ("{{event.action}}" = "*", true, event.action matches "
{{event.action}}") AND if ("{{event.outcome}}" = "*", true, event.outcome matches "
{{event.outcome}}") AND if ("{{user.name}}" = "*", true, user.name matches "
{{user.name}}") AND if ("{{client.ip}}" = "*", true, client.ip matches "{{client.ip}}")
AND if ("{{cloud.service.name}}" = "*", true, cloud.service.name matches "
{{cloud.service.name}}") AND if ("{{cloud.instance.id}}" = "*", true, cloud.instance.id
matches "{{cloud.instance.id}}") AND if ("{{threat.group.name}}" = "*", true, actor
matches "{{threat.group.name}}") AND if ("{{risk.static_level}}" = "*", true,
risk.static_level >= toInt("{{risk.static_level}}"))

| json field=raw "malware_families[*]" as threat_malware_families nodrop
| json field=raw "last_updated" as last_updated nodrop
| formatDate(fromseconds(last_updated), "MM-dd-yyyy") as threat_last_updated
| json field=raw "labels[*].name" as label_name nodrop
| replace(label_name, "\\/","->") as label_name
| replace(label_name, "\""," ") as label_name
| timeslice 1d
| count by _timeslice
| fillmissing timeslice
```

/Active Threats: AWS Resources/Action Plan

```
_sourceCategory={{GuardDutyLogsdatasource}}
| json "accountId", "region", "partition", "id", "arn",
"type","service.serviceName","service.detectorId","service.action","severity","title","desc
"vpcId", "subnetId", "groupId" , "tags", "groupName", "resource.instanceDetails",
"resource.accessKeyDetails.userName" as cloud.account.id, cloud.region, partition, id,
arn, type, service_name, detector_id, action, severity_level, title, description, vpcId,
subnetId , securityGroupId, tags, securityGroupName, instanceDetails, user.name nodrop

| json field=instanceDetails "instanceId", "instanceType","networkInterfaces[0].publicIp"
as instanceid, cloud.machine.type, server.ip
| json field=_raw "resource.resourceType" as resourceType
| json field=_raw "resource.s3BucketDetails[0].name" as bucketName nodrop
| if (resourceType = "S3Bucket", bucketName, instanceid) as cloud.instance.id

| json field=action "awsApiCallAction.remoteIpDetails.ipAddressV4",
"networkConnectionAction.remoteIpDetails.ipAddressV4","networkConnectionAction.localPortDet
"networkConnectionAction.remoteIpDetails.geoLocation.lat",
"networkConnectionAction.remoteIpDetails.organization.asnOrg",
"networkConnectionAction.remoteIpDetails.organization.org",
"networkConnectionAction.remoteIpDetails.organization.isp" as awsCallActionIp,
networkActionIp, localPort,longitude, latitude, asnOrg, organization, isp nodrop

| parse field=type "*:*/*" as threat.tactic.name,cloud.service.name,threat.technique.name
| if (severity_level = 0, 0, 0) as risk.static_level
| if (severity_level > 0 and severity_level <= 2, 1, risk.static_level) as
risk.static_level
| if (severity_level > 2 and severity_level <= 5, 2, risk.static_level) as
risk.static_level
| if (severity_level > 5 and severity_level <= 9, 3, risk.static_level) as
risk.static_level
| if (severity_level > 9, 3, risk.static_level) as risk.static_level

// handle empty values
| if(isEmpty(user.name), "NA", user.name) as user.name
| if(isEmpty(awsCallActionIp), if (isEmpty(networkActionIp), "NA",networkActionIp) ,
awsCallActionIp) as client.ip
| if(isEmpty(cloud.account.id), "NA", cloud.account.id) as cloud.account.id

// global filters
| where if ("{{cloud.account.id}}" = "*", true, cloud.account.id matches "
{{cloud.account.id}}") AND if ("{{cloud.region}}" = "*", true, cloud.region matches "
{{cloud.region}}") AND if ("{{cloud.instance.id}}" = "*", true, cloud.instance.id matches
"{{cloud.instance.id}}") AND if ("{{server.ip}}" = "*", true, server.ip matches "
{{server.ip}}") AND if ("{{risk.static_level}}" = "*", true, risk.static_level >= toInt("
{{risk.static_level}}")) AND if ("{{threat.tactic.name}}" = "*", true, threat.tactic.name
matches "{{threat.tactic.name}}") AND if ("{{cloud.service.name}}" = "*", true,
cloud.service.name matches "{{cloud.service.name}}") AND if ("{{user.name}}" = "*", true,
user.name matches "{{user.name}}") AND if ("{{client.ip}}" = "*", true, client.ip matches
"{{client.ip}}")

| if(!isNull(cloud.instance.id),concat
("https://",region,".console.aws.amazon.com/ec2/v2/home?
region=",cloud.region,"#Instances:search=",cloud.instance.id),"") as link
| tourl (link, cloud.account.id) as cloud.account.id
| count as frequency by title, cloud.account.id, cloud.service.name, organization, isp,
client.ip
| sort by frequency
```

/Active Threats: AWS Resources/Findings by Category

```
_sourceCategory={{GuardDutyLogsdatasource}}
| json "accountId", "region", "partition", "id", "arn",
"type","service.serviceName","service.detectorId","service.action","severity","title","desc
"vpcId", "subnetId", "groupId" , "tags", "groupName", "resource.instanceDetails",
"resource.accessKeyDetails.userName" as cloud.account.id, cloud.region, partition, id,
arn, type, service_name, detector_id, action, severity_level, title, description, vpcId,
subnetId , securityGroupId, tags, securityGroupName, instanceDetails, user.name nodrop

| json field=instanceDetails "instanceId", "instanceType","networkInterfaces[0].publicIp"
as instanceid, cloud.machine.type, server.ip
| json field=_raw "resource.resourceType" as resourceType
| json field=_raw "resource.s3BucketDetails[0].name" as bucketName nodrop
| if (resourceType = "S3Bucket", bucketName, instanceid) as cloud.instance.id

| json field=action "awsApiCallAction.remoteIpDetails.ipAddressV4",
"networkConnectionAction.remoteIpDetails.ipAddressV4","networkConnectionAction.localPortDet
as awsCallActionIp, networkActionIp, localPort nodrop

| parse field=type "*:*/*" as threat.tactic.name,cloud.service.name,threat.technique.name
| if (severity_level = 0, 0, 0) as risk.static_level
| if (severity_level > 0 and severity_level <= 2, 1, risk.static_level) as
risk.static_level
| if (severity_level > 2 and severity_level <= 5, 2, risk.static_level) as
risk.static_level
| if (severity_level > 5 and severity_level <= 9, 3, risk.static_level) as
risk.static_level
| if (severity_level > 9, 3, risk.static_level) as risk.static_level

// handle empty values
| if(isEmpty(user.name), "NA", user.name) as user.name
| if(isEmpty(awsCallActionIp), if (isEmpty(networkActionIp), "NA",networkActionIp) ,
awsCallActionIp) as client.ip
| if(isEmpty(cloud.account.id), "NA", cloud.account.id) as cloud.account.id

// global filters
| where if ("{{cloud.account.id}}" = "*", true, cloud.account.id matches "
{{cloud.account.id}}") AND if ("{{cloud.region}}" = "*", true, cloud.region matches "
{{cloud.region}}") AND if ("{{cloud.instance.id}}" = "*", true, cloud.instance.id matches
"{{cloud.instance.id}}") AND if ("{{server.ip}}" = "*", true, server.ip matches "
{{server.ip}}") AND if ("{{risk.static_level}}" = "*", true, risk.static_level >= toInt("
{{risk.static_level}}")) AND if ("{{threat.tactic.name}}" = "*", true, threat.tactic.name
matches "{{threat.tactic.name}}") AND if ("{{cloud.service.name}}" = "*", true,
cloud.service.name matches "{{cloud.service.name}}") AND if ("{{user.name}}" = "*", true,
user.name matches "{{user.name}}") AND if ("{{client.ip}}" = "*", true, client.ip matches
"{{client.ip}}")

| count by threat.tactic.name
| sort by _count
| limit 10
```

/Active Threats: AWS Resources/Findings by Resource

```
_sourceCategory={{GuardDutyLogsdatasource}}
| json "accountId", "region", "partition", "id", "arn",
"type","service.serviceName","service.detectorId","service.action","severity","title","desc
"vpcId", "subnetId", "groupId" , "tags", "groupName", "resource.instanceDetails",
"resource.accessKeyDetails.userName" as cloud.account.id, cloud.region, partition, id,
arn, type, service_name, detector_id, action, severity_level, title, description, vpcId,
subnetId , securityGroupId, tags, securityGroupName, instanceDetails, user.name nodrop

| json field=instanceDetails "instanceId", "instanceType","networkInterfaces[0].publicIp"
as instanceid, cloud.machine.type, server.ip
| json field=_raw "resource.resourceType" as resourceType
| json field=_raw "resource.s3BucketDetails[0].name" as bucketName nodrop
| if (resourceType = "S3Bucket", bucketName, instanceid) as cloud.instance.id

| json field=action "awsApiCallAction.remoteIpDetails.ipAddressV4",
"networkConnectionAction.remoteIpDetails.ipAddressV4","networkConnectionAction.localPortDet
as awsCallActionIp, networkActionIp, localPort nodrop

| parse field=type "*:*/*" as threat.tactic.name,cloud.service.name,threat.technique.name
| if (severity_level = 0, 0, 0) as risk.static_level
| if (severity_level > 0 and severity_level <= 2, 1, risk.static_level) as
risk.static_level
| if (severity_level > 2 and severity_level <= 5, 2, risk.static_level) as
risk.static_level
| if (severity_level > 5 and severity_level <= 9, 3, risk.static_level) as
risk.static_level
| if (severity_level > 9, 3, risk.static_level) as risk.static_level

// handle empty values
| if(isEmpty(user.name), "NA", user.name) as user.name
| if(isEmpty(awsCallActionIp), if (isEmpty(networkActionIp), "NA",networkActionIp) ,
awsCallActionIp) as client.ip
| if(isEmpty(cloud.account.id), "NA", cloud.account.id) as cloud.account.id

// global filters
| where if ("{{cloud.account.id}}" = "*", true, cloud.account.id matches "
{{cloud.account.id}}") AND if ("{{cloud.region}}" = "*", true, cloud.region matches "
{{cloud.region}}") AND if ("{{cloud.instance.id}}" = "*", true, cloud.instance.id matches
"{{cloud.instance.id}}") AND if ("{{server.ip}}" = "*", true, server.ip matches "
{{server.ip}}") AND if ("{{risk.static_level}}" = "*", true, risk.static_level >= toInt("
{{risk.static_level}}")) AND if ("{{threat.tactic.name}}" = "*", true, threat.tactic.name
matches "{{threat.tactic.name}}") AND if ("{{cloud.service.name}}" = "*", true,
cloud.service.name matches "{{cloud.service.name}}") AND if ("{{user.name}}" = "*", true,
user.name matches "{{user.name}}") AND if ("{{client.ip}}" = "*", true, client.ip matches
"{{client.ip}}")

| count by cloud.instance.id, cloud.account.id, cloud.region,
risk.static_level,threat.tactic.name,cloud.service.name,threat.technique.name, user.name,
client.ip
| sort by _count
| limit 10
```

/Active Threats: AWS Resources/Findings by Resource Type

```
_sourceCategory={{GuardDutyLogsdatasource}}
| json "accountId", "region", "partition", "id", "arn",
"type","service.serviceName","service.detectorId","service.action","severity","title","desc
"vpcId", "subnetId", "groupId" , "tags", "groupName", "resource.instanceDetails",
"resource.accessKeyDetails.userName" as cloud.account.id, cloud.region, partition, id,
arn, type, service_name, detector_id, action, severity_level, title, description, vpcId,
subnetId , securityGroupId, tags, securityGroupName, instanceDetails, user.name nodrop

| json field=instanceDetails "instanceId", "instanceType","networkInterfaces[0].publicIp"
as instanceid, cloud.machine.type, server.ip
| json field=_raw "resource.resourceType" as resourceType
| json field=_raw "resource.s3BucketDetails[0].name" as bucketName nodrop
| if (resourceType = "S3Bucket", bucketName, instanceid) as cloud.instance.id

| json field=action "awsApiCallAction.remoteIpDetails.ipAddressV4",
"networkConnectionAction.remoteIpDetails.ipAddressV4","networkConnectionAction.localPortDet
as awsCallActionIp, networkActionIp, localPort nodrop

| parse field=type "*:*/*" as threat.tactic.name,cloud.service.name,threat.technique.name
| if (severity_level = 0, 0, 0) as risk.static_level
| if (severity_level > 0 and severity_level <= 2, 1, risk.static_level) as
risk.static_level
| if (severity_level > 2 and severity_level <= 5, 2, risk.static_level) as
risk.static_level
| if (severity_level > 5 and severity_level <= 9, 3, risk.static_level) as
risk.static_level
| if (severity_level > 9, 3, risk.static_level) as risk.static_level

// handle empty values
| if(isEmpty(user.name), "NA", user.name) as user.name
| if(isEmpty(awsCallActionIp), if (isEmpty(networkActionIp), "NA",networkActionIp) ,
awsCallActionIp) as client.ip
| if(isEmpty(cloud.account.id), "NA", cloud.account.id) as cloud.account.id

// global filters
| where if ("{{cloud.account.id}}" = "*", true, cloud.account.id matches "
{{cloud.account.id}}") AND if ("{{cloud.region}}" = "*", true, cloud.region matches "
{{cloud.region}}") AND if ("{{cloud.instance.id}}" = "*", true, cloud.instance.id matches
"{{cloud.instance.id}}") AND if ("{{server.ip}}" = "*", true, server.ip matches "
{{server.ip}}") AND if ("{{risk.static_level}}" = "*", true, risk.static_level >= toInt("
{{risk.static_level}}")) AND if ("{{threat.tactic.name}}" = "*", true, threat.tactic.name
matches "{{threat.tactic.name}}") AND if ("{{cloud.service.name}}" = "*", true,
cloud.service.name matches "{{cloud.service.name}}") AND if ("{{user.name}}" = "*", true,
user.name matches "{{user.name}}") AND if ("{{client.ip}}" = "*", true, client.ip matches
"{{client.ip}}")

| count by cloud.service.name
| sort by _count
| limit 10
```

/Active Threats: AWS Resources/Findings Trend

```
_sourceCategory={{GuardDutyLogsdatasource}}
| json "accountId", "region", "partition", "id", "arn",
"type","service.serviceName","service.detectorId","service.action","severity","title","desc
"vpcId", "subnetId", "groupId" , "tags", "groupName", "resource.instanceDetails",
"resource.accessKeyDetails.userName" as cloud.account.id, cloud.region, partition, id,
arn, type, service_name, detector_id, action, severity_level, title, description, vpcId,
subnetId , securityGroupId, tags, securityGroupName, instanceDetails, user.name nodrop

| json field=instanceDetails "instanceId", "instanceType","networkInterfaces[0].publicIp"
as instanceid, cloud.machine.type, server.ip
| json field=_raw "resource.resourceType" as resourceType
| json field=_raw "resource.s3BucketDetails[0].name" as bucketName nodrop
| if (resourceType = "S3Bucket", bucketName, instanceid) as cloud.instance.id

| json field=action "awsApiCallAction.remoteIpDetails.ipAddressV4",
"networkConnectionAction.remoteIpDetails.ipAddressV4","networkConnectionAction.localPortDet
as awsCallActionIp, networkActionIp, localPort nodrop

| parse field=type "*:*/*" as threat.tactic.name,cloud.service.name,threat.technique.name
| if (severity_level = 0, 0, 0) as risk.static_level
| if (severity_level > 0 and severity_level <= 2, 1, risk.static_level) as
risk.static_level
| if (severity_level > 2 and severity_level <= 5, 2, risk.static_level) as
risk.static_level
| if (severity_level > 5 and severity_level <= 9, 3, risk.static_level) as
risk.static_level
| if (severity_level > 9, 3, risk.static_level) as risk.static_level

// handle empty values
| if(isEmpty(user.name), "NA", user.name) as user.name
| if(isEmpty(awsCallActionIp), if (isEmpty(networkActionIp), "NA",networkActionIp) ,
awsCallActionIp) as client.ip
| if(isEmpty(cloud.account.id), "NA", cloud.account.id) as cloud.account.id

// global filters
| where if ("{{cloud.account.id}}" = "*", true, cloud.account.id matches "
{{cloud.account.id}}") AND if ("{{cloud.region}}" = "*", true, cloud.region matches "
{{cloud.region}}") AND if ("{{cloud.instance.id}}" = "*", true, cloud.instance.id matches
"{{cloud.instance.id}}") AND if ("{{server.ip}}" = "*", true, server.ip matches "
{{server.ip}}") AND if ("{{risk.static_level}}" = "*", true, risk.static_level >= toInt("
{{risk.static_level}}")) AND if ("{{threat.tactic.name}}" = "*", true, threat.tactic.name
matches "{{threat.tactic.name}}") AND if ("{{cloud.service.name}}" = "*", true,
cloud.service.name matches "{{cloud.service.name}}") AND if ("{{user.name}}" = "*", true,
user.name matches "{{user.name}}") AND if ("{{client.ip}}" = "*", true, client.ip matches
"{{client.ip}}")

| timeslice 1d
| count by _timeslice
```
/Active Threats: AWS Resources/Threats by Country

```
_sourceCategory={{GuardDutyLogsdatasource}}  (geoLocation and lat and lon)
| json "accountId", "region", "partition", "id", "arn",
"type","service.serviceName","service.detectorId","service.action","severity","title","desc
"vpcId", "subnetId", "groupId" , "tags", "groupName", "resource.instanceDetails",
"resource.accessKeyDetails.userName" as cloud.account.id, cloud.region, partition, id,
arn, type, service_name, detector_id, action, severity_level, title, description, vpcId,
subnetId , securityGroupId, tags, securityGroupName, instanceDetails, user.name nodrop

| json field=instanceDetails "instanceId", "instanceType","networkInterfaces[0].publicIp"
as instanceid, cloud.machine.type, server.ip
| json field=_raw "resource.resourceType" as resourceType
| json field=_raw "resource.s3BucketDetails[0].name" as bucketName nodrop
| if (resourceType = "S3Bucket", bucketName, instanceid) as cloud.instance.id

| json field=action "awsApiCallAction.remoteIpDetails.ipAddressV4",
"networkConnectionAction.remoteIpDetails.ipAddressV4","networkConnectionAction.localPortDet
"networkConnectionAction.remoteIpDetails.geoLocation.lat",
"networkConnectionAction.remoteIpDetails.organization.asnOrg",
"networkConnectionAction.remoteIpDetails.organization.org",
"networkConnectionAction.remoteIpDetails.organization.isp" as awsCallActionIp,
networkActionIp, localPort,longitude, latitude, asnOrg, organization, isp nodrop

| parse field=type "*:*/*" as threat.tactic.name,cloud.service.name,threat.technique.name
| if (severity_level = 0, 0, 0) as risk.static_level
| if (severity_level > 0 and severity_level <= 2, 1, risk.static_level) as
risk.static_level
| if (severity_level > 2 and severity_level <= 5, 2, risk.static_level) as
risk.static_level
| if (severity_level > 5 and severity_level <= 9, 3, risk.static_level) as
risk.static_level
| if (severity_level > 9, 3, risk.static_level) as risk.static_level

// handle empty values
| if(isEmpty(user.name), "NA", user.name) as user.name
| if(isEmpty(awsCallActionIp), networkActionIp, awsCallActionIp) as client.ip
| if(isEmpty(cloud.account.id), "NA", cloud.account.id) as cloud.account.id

// global filters
| where if ("{{cloud.account.id}}" = "*", true, cloud.account.id matches "
{{cloud.account.id}}") AND if ("{{cloud.region}}" = "*", true, cloud.region matches "
{{cloud.region}}") AND if ("{{cloud.instance.id}}" = "*", true, cloud.instance.id matches
"{{cloud.instance.id}}") AND if ("{{server.ip}}" = "*", true, server.ip matches "
{{server.ip}}") AND if ("{{risk.static_level}}" = "*", true, risk.static_level >= toInt("
{{risk.static_level}}")) AND if ("{{threat.tactic.name}}" = "*", true, threat.tactic.name
matches "{{threat.tactic.name}}") AND if ("{{cloud.service.name}}" = "*", true,
cloud.service.name matches "{{cloud.service.name}}") AND if ("{{user.name}}" = "*", true,
user.name matches "{{user.name}}") AND if ("{{client.ip}}" = "*", true, client.ip matches
"{{client.ip}}")

| lookup latitude, longitude from geo://location on ip = client.ip
| count by latitude, longitude
```

/Active Threats: AWS Resources/Total Findings

```
_sourceCategory={{GuardDutyLogsdatasource}}
| json "accountId", "region", "partition", "id", "arn",
"type","service.serviceName","service.detectorId","service.action","severity","title","desc
"vpcId", "subnetId", "groupId" , "tags", "groupName", "resource.instanceDetails",
"resource.accessKeyDetails.userName" as cloud.account.id, cloud.region, partition, id,
arn, type, service_name, detector_id, action, severity_level, title, description, vpcId,
subnetId , securityGroupId, tags, securityGroupName, instanceDetails, user.name nodrop

| json field=instanceDetails "instanceId", "instanceType","networkInterfaces[0].publicIp"
as instanceid, cloud.machine.type, server.ip
| json field=_raw "resource.resourceType" as resourceType
| json field=_raw "resource.s3BucketDetails[0].name" as bucketName nodrop
| if (resourceType = "S3Bucket", bucketName, instanceid) as cloud.instance.id

| json field=action "awsApiCallAction.remoteIpDetails.ipAddressV4",
"networkConnectionAction.remoteIpDetails.ipAddressV4","networkConnectionAction.localPortDet
as awsCallActionIp, networkActionIp, localPort nodrop

| parse field=type "*:*/*" as threat.tactic.name,cloud.service.name,threat.technique.name
| if (severity_level = 0, 0, 0) as risk.static_level
| if (severity_level > 0 and severity_level <= 2, 1, risk.static_level) as
risk.static_level
| if (severity_level > 2 and severity_level <= 5, 2, risk.static_level) as
risk.static_level
| if (severity_level > 5 and severity_level <= 9, 3, risk.static_level) as
risk.static_level
| if (severity_level > 9, 3, risk.static_level) as risk.static_level

// handle empty values
| if(isEmpty(user.name), "NA", user.name) as user.name
| if(isEmpty(awsCallActionIp), if (isEmpty(networkActionIp), "NA",networkActionIp) ,
awsCallActionIp) as client.ip
| if(isEmpty(cloud.account.id), "NA", cloud.account.id) as cloud.account.id

// global filters
| where if ("{{cloud.account.id}}" = "*", true, cloud.account.id matches "
{{cloud.account.id}}") AND if ("{{cloud.region}}" = "*", true, cloud.region matches "
{{cloud.region}}") AND if ("{{cloud.instance.id}}" = "*", true, cloud.instance.id matches
"{{cloud.instance.id}}") AND if ("{{server.ip}}" = "*", true, server.ip matches "
{{server.ip}}") AND if ("{{risk.static_level}}" = "*", true, risk.static_level >= toInt("
{{risk.static_level}}")) AND if ("{{threat.tactic.name}}" = "*", true, threat.tactic.name
matches "{{threat.tactic.name}}") AND if ("{{cloud.service.name}}" = "*", true,
cloud.service.name matches "{{cloud.service.name}}") AND if ("{{user.name}}" = "*", true,
user.name matches "{{user.name}}") AND if ("{{client.ip}}" = "*", true, client.ip matches
"{{client.ip}}")

| count
```

/Active Threats: AWS Storage/Threats by Actor

```
_sourceCategory={{CloudTrailLogsdatasource}}  "s3.amazonaws.com"
| json "userIdentity", "eventTime", "eventSource", "eventName", "awsRegion",
"sourceIPAddress", "userAgent", "errorCode", "requestParameters", "eventType" as
userIdentity, event_time, eventSource, event.action, cloud.region, client.ip, user_agent,
event.outcome, requestParameters, eventType nodrop
| where eventSource = "s3.amazonaws.com"
| json field=userIdentity "type", "accountId" as userType, cloud.account.id
| json field=requestParameters "bucketName" as cloud.instance.id
| parse field=eventSource "*." as cloud.service.name
| parse regex "\"(?i)userName\":\"(?<user_name>.*?)\"" nodrop
| parse "\"userId\":\"*\"" as user_id nodrop
| if (user_name="", user_id, user_name) as user.name

| if (userType matches("IAMUser"), "user", "machine") as event.agent

| lookup type, actor, raw, threatlevel as malicious_confidence, threat from
sumo://threat/cs on threat=client.ip
| where type="ip_address"

| if (malicious_confidence = "low", 1, 0) as risk.static_level
| if (malicious_confidence = "medium", 2, risk.static_level) as risk.static_level
| if (malicious_confidence = "high", 3, risk.static_level) as risk.static_level

// handle empty values
| if (isEmpty(event.outcome), "Success", event.outcome) as event.outcome
| if (isEmpty(user.name), "NA", user.name) as user.name
| if (isEmpty(actor), "Unassigned", actor) as actor
| if(isEmpty(cloud.account.id), "NA", cloud.account.id) as cloud.account.id

// global filters
| where if ("{{cloud.instance.id}}" = "*", true, cloud.instance.id matches "
{{cloud.instance.id}}") AND if ("{{event.action}}" = "*", true, event.action matches "
{{event.action}}") AND if ("{{user.name}}" = "*", true, user.name matches "{{user.name}}")
AND if ("{{event.action}}" = "*", true, event.action matches "{{event.action}}") AND if ("
{{client.ip}}" = "*", true, client.ip matches "{{client.ip}}") AND if ("
{{risk.static_level}}" = "*", true, risk.static_level >= toInt("{{risk.static_level}}"))
AND if ("{{threat.group.name}}" = "*", true, actor matches "{{threat.group.name}}") AND if
("{{source.user}}" = "*", true, source.user matches "{{source.user}}}")

| count by actor
| sort by _count
```

/Active Threats: AWS Storage/Threats by Events and Result

```
_sourceCategory={{CloudTrailLogsdatasource}}  "s3.amazonaws.com"
| json "userIdentity", "eventTime", "eventSource", "eventName", "awsRegion",
"sourceIPAddress", "userAgent", "errorCode", "requestParameters", "eventType" as
userIdentity, event_time, eventSource, event.action, cloud.region, client.ip, user_agent,
event.outcome, requestParameters, eventType nodrop
| where eventSource = "s3.amazonaws.com"
| json field=userIdentity "type", "accountId" as userType, cloud.account.id
| json field=requestParameters "bucketName" as cloud.instance.id
| parse field=eventSource "*." as cloud.service.name
| parse regex "\"(?i)userName\":\"(?<user_name>.*?)\"" nodrop
| parse "\"userId\":\"*\"" as user_id nodrop
| if (user_name="", user_id, user_name) as user.name

| if (userType matches("IAMUser"), "user", "machine") as event.agent

| lookup type, actor, raw, threatlevel as malicious_confidence, threat from
sumo://threat/cs on threat=client.ip
| where type="ip_address"

| if (malicious_confidence = "low", 1, 0) as risk.static_level
| if (malicious_confidence = "medium", 2, risk.static_level) as risk.static_level
| if (malicious_confidence = "high", 3, risk.static_level) as risk.static_level

// handle empty values
| if (isEmpty(event.outcome), "Success", event.outcome) as event.outcome
| if (isEmpty(user.name), "NA", user.name) as user.name
| if (isEmpty(actor), "Unassigned", actor) as actor
| if(isEmpty(cloud.account.id), "NA", cloud.account.id) as cloud.account.id

// global filters
| where if ("{{cloud.instance.id}}" = "*", true, cloud.instance.id matches "
{{cloud.instance.id}}") AND if ("{{event.action}}" = "*", true, event.action matches "
{{event.action}}") AND if ("{{user.name}}" = "*", true, user.name matches "{{user.name}}")
AND if ("{{event.action}}" = "*", true, event.action matches "{{event.action}}") AND if ("
{{client.ip}}" = "*", true, client.ip matches "{{client.ip}}") AND if ("
{{risk.static_level}}" = "*", true, risk.static_level >= toInt("{{risk.static_level}}"))
AND if ("{{threat.group.name}}" = "*", true, actor matches "{{threat.group.name}}") AND if
("{{source.user}}" = "*", true, source.user matches "{{source.user}}}")

| count by event.action, event.outcome
| transpose row event.action column event.outcome
```

/Active Threats: AWS Storage/Threats by Geo Location

```
_sourceCategory={{CloudTrailLogsdatasource}}  "s3.amazonaws.com"
| json "userIdentity", "eventTime", "eventSource", "eventName", "awsRegion",
"sourceIPAddress", "userAgent", "errorCode", "requestParameters", "eventType" as
userIdentity, event_time, eventSource, event.action, cloud.region, client.ip, user_agent,
event.outcome, requestParameters, eventType nodrop
| where eventSource = "s3.amazonaws.com"
| json field=userIdentity "type", "accountId" as userType, cloud.account.id
| json field=requestParameters "bucketName" as cloud.instance.id
| parse field=eventSource "*." as cloud.service.name
| parse regex "\"(?i)userName\":\"(?<user_name>.*?)\"" nodrop
| parse "\"userId\":\"*\"" as user_id nodrop
| if (user_name="", user_id, user_name) as user.name

| if (userType matches("IAMUser"), "user", "machine") as event.agent

| lookup type, actor, raw, threatlevel as malicious_confidence, threat from
sumo://threat/cs on threat=client.ip
| where type="ip_address"

| if (malicious_confidence = "low", 1, 0) as risk.static_level
| if (malicious_confidence = "medium", 2, risk.static_level) as risk.static_level
| if (malicious_confidence = "high", 3, risk.static_level) as risk.static_level

// handle empty values
| if (isEmpty(event.outcome), "Success", event.outcome) as event.outcome
| if (isEmpty(user.name), "NA", user.name) as user.name
| if (isEmpty(actor), "Unassigned", actor) as actor
| if(isEmpty(cloud.account.id), "NA", cloud.account.id) as cloud.account.id

// global filters
| where if ("{{cloud.instance.id}}" = "*", true, cloud.instance.id matches "
{{cloud.instance.id}}") AND if ("{{event.action}}" = "*", true, event.action matches "
{{event.action}}") AND if ("{{user.name}}" = "*", true, user.name matches "{{user.name}}")
AND if ("{{event.action}}" = "*", true, event.action matches "{{event.action}}") AND if ("
{{client.ip}}" = "*", true, client.ip matches "{{client.ip}}") AND if ("
{{risk.static_level}}" = "*", true, risk.static_level >= toInt("{{risk.static_level}}"))
AND if ("{{threat.group.name}}" = "*", true, actor matches "{{threat.group.name}}") AND if
("{{source.user}}" = "*", true, source.user matches "{{source.user}}}")

| lookup latitude, longitude from geo://location on ip = client.ip
| count by latitude, longitude
```

/Active Threats: AWS Storage/Threats by Resource

```
_sourceCategory={{CloudTrailLogsdatasource}}  "s3.amazonaws.com"
| json "userIdentity", "eventTime", "eventSource", "eventName", "awsRegion",
"sourceIPAddress", "userAgent", "errorCode", "requestParameters", "eventType" as
userIdentity, event_time, eventSource, event.action, cloud.region, client.ip, user_agent,
event.outcome, requestParameters, eventType nodrop
| where eventSource = "s3.amazonaws.com"
| json field=userIdentity "type", "accountId" as userType, cloud.account.id
| json field=requestParameters "bucketName" as cloud.instance.id
| parse field=eventSource "*." as cloud.service.name
| if (userType matches("IAMUser"), "user", "machine") as event.agent
| parse regex "\"(?i)userName\":\"(?<user_name>.*?)\"" nodrop
| parse "\"userId\":\"*\"" as user_id nodrop
| if (user_name="", user_id, user_name) as user.name

| lookup type, actor, raw, threatlevel as malicious_confidence, threat from
sumo://threat/cs on threat=client.ip
| where type="ip_address"

| if (malicious_confidence = "low", 1, 0) as risk.static_level
| if (malicious_confidence = "medium", 2, risk.static_level) as risk.static_level
| if (malicious_confidence = "high", 3, risk.static_level) as risk.static_level

// handle empty values
| if (isEmpty(event.outcome), "Success", event.outcome) as event.outcome
| if (isEmpty(user.name), "NA", user.name) as user.name
| if (isEmpty(actor), "Unassigned", actor) as actor
| if(isEmpty(cloud.account.id), "NA", cloud.account.id) as cloud.account.id

// global filters
| where if ("{{cloud.instance.id}}" = "*", true, cloud.instance.id matches "
{{cloud.instance.id}}") AND if ("{{event.action}}" = "*", true, event.action matches "
{{event.action}}") AND if ("{{user.name}}" = "*", true, user.name matches "{{user.name}}")
AND if ("{{event.action}}" = "*", true, event.action matches "{{event.action}}") AND if ("
{{client.ip}}" = "*", true, client.ip matches "{{client.ip}}") AND if ("
{{risk.static_level}}" = "*", true, risk.static_level >= toInt("{{risk.static_level}}"))
AND if ("{{threat.group.name}}" = "*", true, actor matches "{{threat.group.name}}") AND if
("{{source.user}}" = "*", true, source.user matches "{{source.user}}}")

| count by cloud.instance.id, client.ip, user.name, risk.static_level
| sort by _count
```

/Active Threats: AWS Storage/Threats Count

```
_sourceCategory={{CloudTrailLogsdatasource}}  "s3.amazonaws.com"
| json "userIdentity", "eventTime", "eventSource", "eventName", "awsRegion",
"sourceIPAddress", "userAgent", "errorCode", "requestParameters", "eventType" as
userIdentity, event_time, eventSource, event.action, cloud.region, client.ip, user_agent,
event.outcome, requestParameters, eventType nodrop
| where eventSource = "s3.amazonaws.com"
| json field=userIdentity "type", "accountId" as userType, cloud.account.id
| json field=requestParameters "bucketName" as cloud.instance.id
| parse field=eventSource "*." as cloud.service.name
| if (userType matches("IAMUser"), "user", "machine") as event.agent
| parse regex "\"(?i)userName\":\"(?<user_name>.*?)\"" nodrop
| parse "\"userId\":\"*\"" as user_id nodrop
| if (user_name="", user_id, user_name) as user.name

| lookup type, actor, raw, threatlevel as malicious_confidence, threat from
sumo://threat/cs on threat=client.ip
| where type="ip_address"

| if (malicious_confidence = "low", 1, 0) as risk.static_level
| if (malicious_confidence = "medium", 2, risk.static_level) as risk.static_level
| if (malicious_confidence = "high", 3, risk.static_level) as risk.static_level

// handle empty values
| if (isEmpty(event.outcome), "Success", event.outcome) as event.outcome
| if (isEmpty(user.name), "NA", user.name) as user.name
| if (isEmpty(actor), "Unassigned", actor) as actor
| if(isEmpty(cloud.account.id), "NA", cloud.account.id) as cloud.account.id

// global filters
| where if ("{{cloud.instance.id}}" = "*", true, cloud.instance.id matches "
{{cloud.instance.id}}") AND if ("{{event.action}}" = "*", true, event.action matches "
{{event.action}}") AND if ("{{user.name}}" = "*", true, user.name matches "{{user.name}}")
AND if ("{{event.action}}" = "*", true, event.action matches "{{event.action}}") AND if ("
{{client.ip}}" = "*", true, client.ip matches "{{client.ip}}") AND if ("
{{risk.static_level}}" = "*", true, risk.static_level >= toInt("{{risk.static_level}}"))
AND if ("{{threat.group.name}}" = "*", true, actor matches "{{threat.group.name}}") AND if
("{{source.user}}" = "*", true, source.user matches "{{source.user}}}")

| count
```
/Active Threats: AWS Storage/Threats Trend

```
_sourceCategory={{CloudTrailLogsdatasource}}  "s3.amazonaws.com"
| json "userIdentity", "eventTime", "eventSource", "eventName", "awsRegion",
"sourceIPAddress", "userAgent", "errorCode", "requestParameters", "eventType" as
userIdentity, event_time, eventSource, event.action, cloud.region, client.ip, user_agent,
event.outcome, requestParameters, eventType nodrop
| where eventSource = "s3.amazonaws.com"
| json field=userIdentity "type", "accountId" as userType, cloud.account.id
| json field=requestParameters "bucketName" as cloud.instance.id
| parse field=eventSource "*." as cloud.service.name
| parse regex "\"(?i)userName\":\"(?<user_name>.*?)\"" nodrop
| parse "\"userId\":\"*\"" as user_id nodrop
| if (user_name="", user_id, user_name) as user.name

| if (userType matches("IAMUser"), "user", "machine") as event.agent

| lookup type, actor, raw, threatlevel as malicious_confidence, threat from
sumo://threat/cs on threat=client.ip
| where type="ip_address"

| if (malicious_confidence = "low", 1, 0) as risk.static_level
| if (malicious_confidence = "medium", 2, risk.static_level) as risk.static_level
| if (malicious_confidence = "high", 3, risk.static_level) as risk.static_level

// handle empty values
| if (isEmpty(event.outcome), "Success", event.outcome) as event.outcome
| if (isEmpty(user.name), "NA", user.name) as user.name
| if (isEmpty(actor), "Unassigned", actor) as actor
| if(isEmpty(cloud.account.id), "NA", cloud.account.id) as cloud.account.id

// global filters
| where if ("{{cloud.instance.id}}" = "*", true, cloud.instance.id matches "
{{cloud.instance.id}}") AND if ("{{event.action}}" = "*", true, event.action matches "
{{event.action}}") AND if ("{{user.name}}" = "*", true, user.name matches "{{user.name}}")
AND if ("{{event.action}}" = "*", true, event.action matches "{{event.action}}") AND if ("
{{client.ip}}" = "*", true, client.ip matches "{{client.ip}}") AND if ("
{{risk.static_level}}" = "*", true, risk.static_level >= toInt("{{risk.static_level}}"))
AND if ("{{threat.group.name}}" = "*", true, actor matches "{{threat.group.name}}") AND if
("{{source.user}}" = "*", true, source.user matches "{{source.user}}}")

| timeslice 1d
| count by _timeslice
| fillmissing timeslice
```

/Attack Surface: Create,Delete,Update/Attack Surface: Create,Delete,Update

```
_sourceCategory = Labs/AWS/CloudTrail eventName
| json "eventName", "errorCode" nodrop
| where isBlank(errorCode) and (
    (eventName matches "Modify*" or eventName matches "Disassociate*" or eventName matches
"Disable*" or eventName matches "Enable*" or eventName matches "Detach*" or eventName
matches "Unassign*" or eventName matches "Revoke*" or eventName matches "Reset*" or
eventName matches "Update*" or eventName matches "Put*" or eventName matches "Add*" or
eventName matches "Remove*" or eventName matches "Change*")
or (eventName matches "Create*")
or (eventName matches "Delete*"))
| "" as classification
| if (eventName matches
"Create*",concat("ResourcesCount_Create,",classification),classification) as
classification
| if (eventName matches
"Delete*",concat("ResourcesCount_Delete,",classification),classification) as
classification
| if (eventName matches "Modify*" or eventName matches "Disassociate*" or eventName
matches "Disable*" or eventName matches "Enable*" or eventName matches "Detach*" or
eventName matches "Unassign*" or eventName matches "Revoke*" or eventName matches "Reset*"
or eventName matches "Update*" or eventName matches "Put*" or eventName matches "Add*" or
eventName matches "Remove*" or eventName matches
"Change*",concat("ResourcesCount_Update,",classification),classification) as
classification
| parse regex field=classification "(?<benchmarkname>[^,]+)," multi
| "ActionEventCount" as type
| count as count by benchmarkname, type
| fillmissing
values("ResourcesCount_Create","ResourcesCount_Update","ResourcesCount_Delete") in
benchmarkname, values("ActionEventCount") in type
| toInt(count) as count
```

/Attack Surface: EC2,Redshift,S3/Attack Surface: EC2,Redshift,S3

```
_sourceCategory = Labs/AWS/CloudTrail (instanceId or clusterIdentifier or bucketName)
| json "eventSource", "errorCode" nodrop
| where isBlank(errorCode) and (eventSource="ec2.amazonaws.com"  or
eventSource="redshift.amazonaws.com" or eventSource="s3.amazonaws.com")
| parse regex "\"(?<objectType>instanceId|clusterIdentifier|bucketName)\".*?:.*?(?
<objectId>[^\"]+)\"" multi
| "" as classification
| if (objectType="instanceId" and
eventSource="ec2.amazonaws.com",concat("ResourcesCount_EC2,",classification),classificatior
as classification
| if (objectType="clusterIdentifier" and
eventSource="redshift.amazonaws.com",concat("ResourcesCount_Redshift,",classification),clas
as classification
| if (objectType="bucketName" and
eventSource="s3.amazonaws.com",concat("ResourcesCount_S3,",classification),classification)
as classification
| parse regex field=classification "(?<benchmarkname>[^,]+)," multi
| "ResourceCount" as type
| count_distinct(objectId) as count by benchmarkName, type
| fillmissing values("ResourcesCount_EC2","ResourcesCount_Redshift","ResourcesCount_S3")
in benchmarkname, values("ResourceCount") in type
| toInt(count) as count
```

/Attack Surface: IAM,KMS,Lambda,RDS/Attack Surface: IAM,KMS,Lambda,RDS

```
_sourceCategory = Labs/AWS/CloudTrail (arn:aws:iam or arn:aws:kms  or arn:aws:lambda  or
arn:aws:rds)
| json "errorCode" nodrop
| where isBlank(errorCode)
| parse regex "\"(?<arn>arn:[^\"]+)\"" multi
| parse regex field=arn ".*?:.*?:(?<service>[^:]+):.*?:.*?:(?<resource>[^:]+$)"
| where service in ("iam","rds","lambda","kms")
| "" as classification
| if (service="iam",concat("ResourcesCount_IAM,",classification),classification) as
classification
| if (service="rds",concat("ResourcesCount_RDS,",classification),classification) as
classification
| if (service="lambda",concat("ResourcesCount_Lambda,",classification),classification) as
classification
| if (service="kms",concat("ResourcesCount_KMS,",classification),classification) as
classification
| parse regex field=classification "(?<benchmarkname>[^,]+)," multi
| "ResourceCount" as type
| count_distinct(arn) as count by benchmarkname,type
| fillmissing
values("ResourcesCount_IAM","ResourcesCount_KMS","ResourcesCount_Lambda","ResourcesCount_RD
in benchmarkname, values("ResourceCount") in type
| toInt(count) as count
```

/Attack Surface: Service/Attack Surface: Service

```
_sourceCategory = Labs/AWS/CloudTrail
| json "eventSource", "errorCode" nodrop
| where isBlank(errorCode)
| count_distinct(eventSource) as count
| "ResourcesCount_Service" as benchmarkname
| "ResourceCount" as type
| fillmissing values("ResourcesCount_Service") in benchmarkname, values("ResourceCount")
in type
| toInt(count) as count
```

1Password/Failed Sign-ins/Breakdown by Client App

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| where !(category matches "*succ*")
| count as count by client_app_name
| sort by count
```

1Password/Failed Sign-ins/Breakdown by Client OS

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| where !(category matches "*succ*")
| count as count by client_os
| sort by count
```

1Password/Failed Sign-ins/Breakdown by Country, Region, City

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| where !(category matches "*succ*")
| count as count by country, region, city
| sort by count
```

1Password/Failed Sign-ins/Geolocation of Clients

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| where !(category matches "*succ*")
| lookup latitude, longitude, country_code, country_name, region, city, postal_code from
geo://location on ip = client_ip
| count by latitude, longitude, country_code, country_name, region, city, postal_code
| sort _count
```

1Password/Failed Sign-ins/Outlier - Failed Sign-ins

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| where !(category matches "*succ*")
| timeslice 1d
| count by _timeslice
| outlier _count window=5,threshold=2,consecutive=2,direction=+
```

1Password/Failed Sign-ins/Sign-in Events - One Day Time Comparison

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_version",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| where !(category matches "*succ*")
| count as count by target_user_name
| compare with timeshift 1d
| sort by count
```

1Password/Failed Sign-ins/Sign-in Events Over Time

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| where !(category matches "*succ*")
| timeslice 1d
| count as count by _timeslice, target_user_name
| transpose row _timeslice column target_user_name
```

1Password/Failed Sign-ins/Sign-in Summary

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| where !(category matches "*succ*")
| count by timestamp, target_user_name, type, category, details,client_app_name,
client_app_version, client_platform, client_os, client_os_version, client_ip, country,
region, city
| fields - _count
```

1Password/Failed Sign-ins/Top 10 Active Users

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| where !(category matches "*succ*")
| count as count by target_user_name
| sort by count
| limit 10
```

1Password/Item Usage/Breakdown by Action

```
_sourceCategory={{_sourceCategory}} action
| json "timestamp", "user.name", "client.app_name", "client.platform_name",
"client.platform_version", "client.os_name", "client.os_version", "client.ip_address",
"location.country", "location.region", "location.city", "action", "vault_uuid",
"item_uuid" as timestamp, user_name, client_app_name, client_platform,
client_platform_version, client_os, client_os_version, client_ip, country, region, city,
action, vault_uuid, item_uuid
| where action matches  "{{action}}"
| count by action
| sort by _count
```

1Password/Item Usage/Geolocation of Clients

```
_sourceCategory={{_sourceCategory}} action
| json "timestamp", "user.name", "client.app_name", "client.platform_name",
"client.platform_version", "client.os_name", "client.os_version", "client.ip_address",
"location.country", "location.region", "location.city", "action", "vault_uuid",
"item_uuid" as timestamp, user_name, client_app_name, client_platform,
client_platform_version, client_os, client_os_version, client_ip, country, region, city,
action, vault_uuid, item_uuid
| where action matches  "{{action}}"
| lookup latitude, longitude, country_code, country_name, region, city, postal_code from
geo://location on ip = client_ip
| count by latitude, longitude, country_code, country_name, region, city, postal_code
| sort _count
```

1Password/Item Usage/Item Usage Summary

```
_sourceCategory={{_sourceCategory}} action
| json "timestamp", "user.name", "client.app_name", "client.platform_name",
"client.platform_version", "client.os_name", "client.os_version", "client.ip_address",
"location.country", "location.region", "location.city", "action", "vault_uuid",
"item_uuid" as timestamp, user_name, client_app_name, client_platform,
client_platform_version, client_os, client_os_version, client_ip, country, region, city,
action, vault_uuid, item_uuid
| where action matches  "{{action}}"
| count by timestamp, action, user_name, client_app_name, client_platform,
client_platform_version, client_os, client_os_version, client_ip, country, region, city,
vault_uuid, item_uuid
```

1Password/Item Usage/Threat Intel - Item Usage Clients

```
_sourceCategory={{_sourceCategory}} action
| json "timestamp", "user.name", "client.app_name", "client.platform_name",
"client.platform_version", "client.os_name", "client.os_version", "client.ip_address",
"location.country", "location.region", "location.city", "action", "vault_uuid",
"item_uuid" as timestamp, user_name, client_app_name, client_platform,
client_platform_version, client_os, client_os_version, client_ip, country, region, city,
action, vault_uuid, item_uuid
| where action matches  "{{action}}"
| where !isBlank(client_ip)
| lookup type, actor, raw, threatlevel as malicious_confidence from sumo://threat/cs on
threat=client_ip
| where malicious_confidence matches "*"
| json field=raw "labels[*].name" as label_name
| replace(label_name, "\\/","->") as label_name
| replace(label_name, "\""," ") as label_name
| where type="ip_address" and !isNull(malicious_confidence)
| if (isEmpty(actor), "Unassigned", actor) as Actor
| count by timestamp, action, user_name, client_ip, malicious_confidence,actor,
client_app_name, client_platform, client_os, client_os_version, country, region, city,
vault_uuid, item_uuid
| sort by _count
```

1Password/Overview/Failed Sign In Events

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"|
where category in ("credentials_failed", "mfa_failed", "modern_version_failed",
"firewall_failed")
| count by target_user_name, category, type, details, country, city, client_app_name,
client_platform
| sort by _count
```

1Password/Overview/Geolocation of Clients

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "
{{client_os}}"|lookup latitude, longitude, country_code, country_name, region, city,
postal_code from geo://location on ip = client_ip
| count by latitude, longitude, country_code, country_name, region, city, postal_code
| sort _count
```

1Password/Overview/One Day Time Comparison

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| count as count by  category, type | compare with timeshift 1d
| sort by count
```

1Password/Overview/Sign In Events by Category

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| timeslice 1d
| count by category, _timeslice
| transpose row _timeslice column category
```

1Password/Overview/Sign In Events by Type

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| timeslice 1d
| count by type, _timeslice
| transpose row _timeslice column type
```

1Password/Successful Sign-ins/Breakdown by Client App

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| where category matches "*succ*"
| count as count by client_app_name
| sort by count
```

1Password/Successful Sign-ins/Breakdown by Client OS

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| where category matches "*succ*"
| count as count by client_os
| sort by count
```

1Password/Successful Sign-ins/Breakdown by Country, Region, City

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| where category matches "*succ*"
| count as count by country, region, city
| sort by count
```

1Password/Successful Sign-ins/Geolocation of Clients

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| where category matches "*succ*"
| lookup latitude, longitude, country_code, country_name, region, city, postal_code from
geo://location on ip = client_ip
| count by latitude, longitude, country_code, country_name, region, city, postal_code
| sort _count
```

1Password/Successful Sign-ins/Outlier - Sign-in Events

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| where category matches "*succ*"
| timeslice 1d
| count by _timeslice
| outlier _count window=5,threshold=2,consecutive=2,direction=+
```

1Password/Successful Sign-ins/Sign-in Events - One Day Time Comparison

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| where category matches "*succ*"
| count as count by target_user_name
| compare with timeshift 1d
| sort by count
```

1Password/Successful Sign-ins/Sign-in Events Over Time

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| where category matches "*succ*"
| timeslice 1d
| count as count by _timeslice, target_user_name
| transpose row _timeslice column target_user_name
```

1Password/Successful Sign-ins/Sign-in Summary

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| where category matches "*succ*"
| count by timestamp, target_user_name, type, category, details,client_app_name,
client_app_version, client_platform, client_os, client_os_version, client_ip, country,
region, city
| fields - _count
```

1Password/Successful Sign-ins/Top 10 Active Users

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| where category matches "*succ*"
| count as count by target_user_name
| sort by count
| limit 10
```

1Password/Threat Intel/Highly Malicious Threat Table

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| where !isBlank(client_ip)
| lookup type, actor, raw, threatlevel as malicious_confidence from sumo://threat/cs on
threat=client_ip
| where malicious_confidence matches "high"
| json field=raw "labels[*].name" as label_name
| replace(label_name, "\\/","->") as label_name
| replace(label_name, "\""," ") as label_name
| where type="ip_address" and !isNull(malicious_confidence)
| if (isEmpty(actor), "Unassigned", actor) as Actor
| count by timestamp, target_user_name, malicious_confidence, client_ip, actor, type,
category, client_app_name, client_platform, country, region, city
| sort by _count
```

1Password/Threat Intel/Threat by Actors

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| where !isBlank(client_ip)
| lookup type, actor, raw, threatlevel as malicious_confidence from sumo://threat/cs on
threat=client_ip
| where malicious_confidence matches "{{malicious_confidence}}"
| json field=raw "labels[*].name" as label_name
| replace(label_name, "\\/","->") as label_name
| replace(label_name, "\""," ") as label_name
| where type="ip_address" and !isNull(malicious_confidence)
| if (isEmpty(actor), "Unassigned", actor) as Actor
| count as threatCount by Actor
| sort by threatCount, Actor asc
```

1Password/Threat Intel/Threat by Malicious Confidence

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| where !isBlank(client_ip)
| lookup type, actor, raw, threatlevel as malicious_confidence from sumo://threat/cs on
threat=client_ip
| where malicious_confidence matches "*"
| json field=raw "labels[*].name" as label_name
| replace(label_name, "\\/","->") as label_name
| replace(label_name, "\""," ") as label_name
| where type="ip_address" and !isNull(malicious_confidence)
| if (isEmpty(actor), "Unassigned", actor) as Actor
| count as threatCount by malicious_confidence
| sort by threatCount, malicious_confidence asc
```

1Password/Threat Intel/Threat Count

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| where !isBlank(client_ip)
| lookup type, actor, raw, threatlevel as malicious_confidence from sumo://threat/cs on
threat=client_ip
| where malicious_confidence matches "{{malicious_confidence}}"
| where !isNull(malicious_confidence)
| lookup latitude, longitude, country_code, country_name, city from geo://location on ip =
client_ip
| count
```

1Password/Threat Intel/Threat Locations

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| where !isBlank(client_ip)
| lookup type, actor, raw, threatlevel as malicious_confidence from sumo://threat/cs on
threat=client_ip
| where malicious_confidence matches "{{malicious_confidence}}"
| where !isNull(malicious_confidence)
| lookup latitude, longitude, country_code, country_name, city from geo://location on ip =
client_ip
| count by client_ip, country_coude, country_name, city, type, actor,
malicious_confidence, user_agent, latitude, longitude, target_user_name, client_app_name,
categhory
| sort by _count
```

1Password/Threat Intel/Threat Table

```
_sourceCategory={{_sourceCategory}}
| json "type", "category", "timestamp",  "details", "target_user.name",
"target_user.email", "client.app_name", "client.app_version", "client.platform_name",
"client.os_name", "client.os_version", "client.ip_address", "location.country",
"location.region", "location.city" as type, category, timestamp, details,
target_user_name, target_user_email, client_app_name, client_app_version, client_platform,
client_os, client_os_version, client_ip, country, region, city
| where category matches  "{{category}}" AND type matches  "{{type}}" AND country matches
"{{country}}" AND city matches  "{{city}}" AND target_user_name matches  "
{{target_user_name}}" AND client_app_name matches  "{{client_app_name}}" AND
client_platform matches  "{{client_platform}}" AND client_os matches  "{{client_os}}"
| where !isBlank(client_ip)
| lookup type, actor, raw, threatlevel as malicious_confidence from sumo://threat/cs on
threat=client_ip
| where malicious_confidence matches "{{malicious_confidence}}"
| json field=raw "labels[*].name" as label_name
| replace(label_name, "\\/","->") as label_name
| replace(label_name, "\""," ") as label_name
| where type="ip_address" and !isNull(malicious_confidence)
| if (isEmpty(actor), "Unassigned", actor) as Actor
| count by timestamp, target_user_name, malicious_confidence, client_ip, actor, type,
category, client_app_name, client_platform, country, region, city
| sort by _count
```

Abnormal Security/Cases/Cases by Severity Level

```
_sourceCategory={{Logsdatasource}}  sourcetype case_log
| json "event.severity_level", "event.caseId", "event.description", "sourcetype" as
severity, case_id, description, source_type nodrop

// global filters
| where severity matches "{{severity}}"

| where source_type matches ("case_log")
| count as frequency by severity
| sort by frequency, severity
```

Abnormal Security/Cases/Cases Over Time

```
_sourceCategory={{Logsdatasource}}  sourcetype case_log
| json "event.severity_level", "event.caseId", "event.description", "sourcetype" as
severity, case_id, description, source_type nodrop

// global filters
| where severity matches "{{severity}}"

| where source_type matches ("case_log")
| timeslice 1d
| count as frequency by _timeslice
```

Abnormal Security/Cases/Latest Cases

```
_sourceCategory={{Logsdatasource}}  sourcetype case_log
| json field=_raw "event.severity_level", "event.caseId", "event.description",
"event.last_modified", "sourcetype" as severity, case_id, description, case_time,
source_type nodrop

// global filters
| where severity matches "{{severity}}"

| where source_type matches ("case_log")
| count as frequency by _messageTime, case_id, severity, description
| sort by _messageTime
| formatDate(_messageTime, "dd-MM-yyyy HH:mm:ss") as time
| fields time, case_id, severity, description
| fields - _messageTime
| limit 100
```

Abnormal Security/Emails/Auto Remediated Emails

```
_sourceCategory={{Logsdatasource}}  sourcetype threat_log
| json "event.from_address", "event.recipient_address", "event.post_remediated",
"event.auto_remediated",  "event.attack_strategy", "event.attacked_party",
"event.attack_vector", "event.attack_type", "sourcetype" as sender, receiver,
post_remediated, auto_remediated, strategy, party, vector, type, source_type nodrop

// global filters
| where type matches "{{attack_type}}"
| where party matches "{{attack_party}}"
| where strategy matches "{{attack_strategy}}"
| where vector matches "{{attack_vector}}"
| where auto_remediated matches "{{auto_remediated}}"
| where post_remediated matches "{{post_remediated}}"

| where source_type matches ("threat_log") and auto_remediated matches ("true")
| count
```

Abnormal Security/Emails/Geo Location of Senders

```
_sourceCategory={{Logsdatasource}}  sourcetype threat_log
| json "event.from_address", "event.to_addresses", "event.post_remediated",
"event.auto_remediated",  "event.attack_strategy", "event.attacked_party",
"event.attack_vector", "event.attack_type",
"sourcetype","event.is_read","event.sender_ip_address" as sender, receiver,
post_remediated, auto_remediated, strategy, party, vector, type, source_type, is_open, ip
nodrop

// global filters
| where type matches "{{attack_type}}"
| where party matches "{{attack_party}}"
| where strategy matches "{{attack_strategy}}"
| where vector matches "{{attack_vector}}"
| where auto_remediated matches "{{auto_remediated}}"
| where post_remediated matches "{{post_remediated}}"

| where source_type matches("threat_log")
| where isValidIPv4(ip) or isValidIPv6(ip)
| where !isNull(ip)
| if(isValidIPv4(ip), if(!isPrivateIP(ip),true,false),true) as is_public
| where is_public
| count as frequency by ip
| lookup latitude, longitude, country_code from geo://location on ip = ip
```

Abnormal Security/Emails/Opened Threat Emails

```
_sourceCategory={{Logsdatasource}}  sourcetype threat_log
| json "event.from_address", "event.recipient_address", "event.post_remediated",
"event.auto_remediated",  "event.attack_strategy", "event.attacked_party",
"event.attack_vector", "event.attack_type", "sourcetype", "event.is_read" as sender,
receiver, post_remediated, auto_remediated, strategy, party, vector, type, source_type,
is_open nodrop

// global filters
| where type matches "{{attack_type}}"
| where party matches "{{attack_party}}"
| where strategy matches "{{attack_strategy}}"
| where vector matches "{{attack_vector}}"
| where auto_remediated matches "{{auto_remediated}}"
| where post_remediated matches "{{post_remediated}}"

| where source_type matches ("threat_log") and is_open matches ("true")
| count
```

Abnormal Security/Emails/Opened Threat Emails Over Time

```
_sourceCategory={{Logsdatasource}}  sourcetype threat_log
| json "event.from_address", "event.recipient_address", "event.post_remediated",
"event.auto_remediated",  "event.attack_strategy", "event.attacked_party",
"event.attack_vector", "event.attack_type", "sourcetype", "event.is_read" as sender,
receiver, post_remediated, auto_remediated, strategy, party, vector, type, source_type,
is_open nodrop

// global filters
| where type matches "{{attack_type}}"
| where party matches "{{attack_party}}"
| where strategy matches "{{attack_strategy}}"
| where vector matches "{{attack_vector}}"
| where auto_remediated matches "{{auto_remediated}}"
| where post_remediated matches "{{post_remediated}}"

| where source_type matches ("threat_log") and is_open matches ("true")
| timeslice 1d
| count as frequency by _timeslice
```

Abnormal Security/Emails/Post Remediated Emails

```
_sourceCategory={{Logsdatasource}}  sourcetype threat_log
| json "event.from_address", "event.recipient_address", "event.post_remediated",
"event.auto_remediated",  "event.attack_strategy", "event.attacked_party",
"event.attack_vector", "event.attack_type", "sourcetype" as sender, receiver,
post_remediated, auto_remediated, strategy, party, vector, type, source_type nodrop

// global filters
| where type matches "{{attack_type}}"
| where party matches "{{attack_party}}"
| where strategy matches "{{attack_strategy}}"
| where vector matches "{{attack_vector}}"
| where auto_remediated matches "{{auto_remediated}}"
| where post_remediated matches "{{post_remediated}}"

| where source_type matches ("threat_log")
| where post_remediated matches ("true")
| count
```

Abnormal Security/Emails/Remediation Triggered Emails

```
_sourceCategory={{Logsdatasource}}  sourcetype threat_log
| json "event.from_address", "event.recipient_address", "event.post_remediated",
"event.auto_remediated",  "event.attack_strategy", "event.attacked_party",
"event.attack_vector", "event.attack_type", "sourcetype" as sender, receiver,
post_remediated, auto_remediated, strategy, party, vector, type, source_type nodrop

// global filters
| where type matches "{{attack_type}}"
| where party matches "{{attack_party}}"
| where strategy matches "{{attack_strategy}}"
| where vector matches "{{attack_vector}}"
| where auto_remediated matches "{{auto_remediated}}"
| where post_remediated matches "{{post_remediated}}"

| where source_type matches ("threat_log")
| where auto_remediated matches ("false") and post_remediated matches ("false")
| count
```

Abnormal Security/Emails/Senders from Risky Geo Locations

```
_sourceCategory={{Logsdatasource}}  sourcetype threat_log
| json "event.from_address", "event.to_addresses", "event.sent_time",
"event.post_remediated", "event.auto_remediated",  "event.attack_strategy",
"event.attacked_party", "event.attack_vector", "event.attack_type",
"sourcetype","event.is_read","event.sender_ip_address" as sender, receiver, send_time,
post_remediated, auto_remediated, strategy, party, vector, type, source_type, is_open, ip
nodrop

// global filters
| where type matches "{{attack_type}}"
| where party matches "{{attack_party}}"
| where strategy matches "{{attack_strategy}}"
| where vector matches "{{attack_vector}}"
| where auto_remediated matches "{{auto_remediated}}"
| where post_remediated matches "{{post_remediated}}"

| where sourceType matches("threat_log")
| where isValidIPv4(ip) or isValidIPv6(ip)
| where !isNull(ip)
| if(isValidIPv4(ip), if(!isPrivateIP(ip),true,false),true) as is_public
| where is_public
| count as frequency by ip
| lookup latitude, longitude, country_code from geo://location on ip = ip
| lookup country_code from https://sumologic-app-data.s3.amazonaws.com/riskycountries.csv
on country_code=country_code
| where !isBlank(country_code)
```

Abnormal Security/Emails/Top 10 Senders

```
_sourceCategory={{Logsdatasource}}  sourcetype threat_log
| json "event.from_address", "event.recipient_address", "event.post_remediated",
"event.auto_remediated",  "event.attack_strategy", "event.attacked_party",
"event.attack_vector", "event.attack_type", "sourcetype" as sender, receiver,
post_remediated, auto_remediated, strategy, party, vector, type, source_type nodrop

// global filters
| where type matches "{{attack_type}}"
| where party matches "{{attack_party}}"
| where strategy matches "{{attack_strategy}}"
| where vector matches "{{attack_vector}}"
| where auto_remediated matches "{{auto_remediated}}"
| where post_remediated matches "{{post_remediated}}"

| where source_type matches ("threat_log")
| count as threat_messages by sender
| sort by threat_messages, sender
| limit 10
```

Abnormal Security/Emails/Top 10 Threat Receivers

```
_sourceCategory={{Logsdatasource}}  sourcetype threat_log
| json "event.from_address", "event.recipient_address", "event.post_remediated",
"event.auto_remediated",  "event.attack_strategy", "event.attacked_party",
"event.attack_vector", "event.attack_type", "sourcetype", "event.is_read" as sender,
receiver, post_remediated, auto_remediated, strategy, party, vector, type, source_type,
is_open nodrop

// global filters
| where type matches "{{attack_type}}"
| where party matches "{{attack_party}}"
| where strategy matches "{{attack_strategy}}"
| where vector matches "{{attack_vector}}"
| where auto_remediated matches "{{auto_remediated}}"
| where post_remediated matches "{{post_remediated}}"

| where source_type matches ("threat_log")
| count as threat_messages by receiver
| sort by threat_messages, receiver
| limit 10
```

Abnormal Security/Overview/Latest Threats

```
_sourceCategory={{Logsdatasource}}  sourcetype threat_log
| json "event.from_address", "event.to_addresses", "event.post_remediated",
"event.auto_remediated",  "event.attack_strategy", "event.attacked_party",
"event.attack_vector", "event.attack_type", "sourcetype" as sender, receiver,
post_remediated, auto_remediated, strategy, party, vector, type, source_type nodrop

// global filters
| where type matches "{{attack_type}}"
| where party matches "{{attack_party}}"
| where strategy matches "{{attack_strategy}}"
| where vector matches "{{attack_vector}}"
| where auto_remediated matches "{{auto_remediated}}"
| where post_remediated matches "{{post_remediated}}"

| where source_type matches ("threat_log")
| count as frequency by _messageTime, sender, receiver, post_remediated, auto_remediated,
strategy, party, vector, type
| sort by _messageTime
| formatDate(_messageTime, "dd-MM-yyyy HH:mm:ss") as time
| fields time, sender, receiver, type, strategy, party, vector, auto_remediated,
post_remediated
| fields - _messageTime
| limit 100
```

Abnormal Security/Overview/Malware Attacks

```
_sourceCategory={{Logsdatasource}}  sourcetype threat_log
| json "event.attack_strategy", "event.attacked_party", "event.attack_vector",
"event.attack_type", "sourcetype", "event.auto_remediated", "event.post_remediated" as
strategy, party, vector, type, source_type, auto_remediated, post_remediated nodrop

// global filters
| where type matches "{{attack_type}}"
| where party matches "{{attack_party}}"
| where strategy matches "{{attack_strategy}}"
| where vector matches "{{attack_vector}}"
| where auto_remediated matches "{{auto_remediated}}"
| where post_remediated matches "{{post_remediated}}"

| where source_type matches ("threat_log") and type matches ("Malware")
| count
```

Abnormal Security/Overview/Malware Attacks Over Time

```
_sourceCategory={{Logsdatasource}}  sourcetype threat_log
| json "event.attack_strategy", "event.attacked_party", "event.attack_vector",
"event.attack_type", "sourcetype", "event.auto_remediated", "event.post_remediated" as
strategy, party, vector, type, source_type, auto_remediated, post_remediated nodrop

// global filters
| where type matches "{{attack_type}}"
| where party matches "{{attack_party}}"
| where strategy matches "{{attack_strategy}}"
| where vector matches "{{attack_vector}}"
| where auto_remediated matches "{{auto_remediated}}"
| where post_remediated matches "{{post_remediated}}"

| where source_type matches ("threat_log") and type matches ("*Malware*")
| timeslice 1d
| count as frequency by _timeslice
```

Abnormal Security/Overview/Phishing Attacks

```
_sourceCategory={{Logsdatasource}}  sourcetype threat_log
| json "event.attack_strategy", "event.attacked_party", "event.attack_vector",
"event.attack_type", "sourcetype", "event.auto_remediated", "event.post_remediated" as
strategy, party, vector, type, source_type, auto_remediated, post_remediated nodrop

// global filters
| where type matches "{{attack_type}}"
| where party matches "{{attack_party}}"
| where strategy matches "{{attack_strategy}}"
| where vector matches "{{attack_vector}}"
| where auto_remediated matches "{{auto_remediated}}"
| where post_remediated matches "{{post_remediated}}"

| where source_type matches ("threat_log")
| where type matches ("*Phishing*")
| count
```

Abnormal Security/Overview/Phishing Attacks Over Time

```
_sourceCategory={{Logsdatasource}}  sourcetype threat_log
| json "event.attack_strategy", "event.attacked_party", "event.attack_vector",
"event.attack_type", "sourcetype", "event.auto_remediated", "event.post_remediated" as
strategy, party, vector, type, source_type, auto_remediated, post_remediated nodrop

// global filters
| where type matches "{{attack_type}}"
| where party matches "{{attack_party}}"
| where strategy matches "{{attack_strategy}}"
| where vector matches "{{attack_vector}}"
| where auto_remediated matches "{{auto_remediated}}"
| where post_remediated matches "{{post_remediated}}"

| where source_type matches ("threat_log") and type matches ("*Phishing*")
| timeslice 1d
| count as frequency by _timeslice
```

Abnormal Security/Overview/Threats by Attack Party

```
_sourceCategory={{Logsdatasource}}  sourcetype threat_log
| json "event.attack_strategy", "event.attacked_party", "event.attack_vector",
"event.attack_type", "sourcetype", "event.auto_remediated", "event.post_remediated" as
strategy, party, vector, type, source_type, auto_remediated, post_remediated nodrop

// global filters
| where type matches "{{attack_type}}"
| where party matches "{{attack_party}}"
| where strategy matches "{{attack_strategy}}"
| where vector matches "{{attack_vector}}"
| where auto_remediated matches "{{auto_remediated}}"
| where post_remediated matches "{{post_remediated}}"

| where source_type matches ("threat_log")
| count as frequency by party
| sort by frequency, party
```

Abnormal Security/Overview/Threats by Attack Strategy

```
_sourceCategory={{Logsdatasource}}  sourcetype threat_log
| json "event.attack_strategy", "event.attacked_party", "event.attack_vector",
"event.attack_type", "sourcetype", "event.auto_remediated", "event.post_remediated" as
strategy, party, vector, type, source_type, auto_remediated, post_remediated nodrop

// global filters
| where type matches "{{attack_type}}"
| where party matches "{{attack_party}}"
| where strategy matches "{{attack_strategy}}"
| where vector matches "{{attack_vector}}"
| where auto_remediated matches "{{auto_remediated}}"
| where post_remediated matches "{{post_remediated}}"

| where source_type matches ("threat_log")
| count as frequency by strategy
| sort by frequency, strategy
```

Abnormal Security/Overview/Threats by Attack Type

```
_sourceCategory={{Logsdatasource}}  sourcetype threat_log
| json "event.attack_strategy", "event.attacked_party", "event.attack_vector",
"event.attack_type", "sourcetype", "event.auto_remediated", "event.post_remediated" as
strategy, party, vector, type, source_type, auto_remediated, post_remediated nodrop

// global filters
| where type matches "{{attack_type}}"
| where party matches "{{attack_party}}"
| where strategy matches "{{attack_strategy}}"
| where vector matches "{{attack_vector}}"
| where auto_remediated matches "{{auto_remediated}}"
| where post_remediated matches "{{post_remediated}}"

| where source_type matches ("threat_log")
| count as frequency by type
| sort by frequency, type
```

Abnormal Security/Overview/Threats by Attack Vector

```
_sourceCategory={{Logsdatasource}}  sourcetype threat_log
| json "event.attack_strategy", "event.attacked_party", "event.attack_vector",
"event.attack_type", "sourcetype", "event.auto_remediated", "event.post_remediated" as
strategy, party, vector, type, source_type, auto_remediated, post_remediated nodrop

// global filters
| where type matches "{{attack_type}}"
| where party matches "{{attack_party}}"
| where strategy matches "{{attack_strategy}}"
| where vector matches "{{attack_vector}}"
| where auto_remediated matches "{{auto_remediated}}"
| where post_remediated matches "{{post_remediated}}"

| where source_type matches ("threat_log")
| count as frequency by vector
| sort by frequency, vector
```

Abnormal Security/Overview/Threats Over Time

```
_sourceCategory={{Logsdatasource}}  sourcetype threat_log
| json "event.attack_strategy", "event.attacked_party", "event.attack_vector",
"event.attack_type", "sourcetype", "event.auto_remediated", "event.post_remediated" as
strategy, party, vector, type, source_type, auto_remediated, post_remediated nodrop

// global filters
| where type matches "{{attack_type}}"
| where party matches "{{attack_party}}"
| where strategy matches "{{attack_strategy}}"
| where vector matches "{{attack_vector}}"
| where auto_remediated matches "{{auto_remediated}}"
| where post_remediated matches "{{post_remediated}}"

| where source_type matches ("threat_log")
| timeslice 1d
| count as frequency by _timeslice
```

Abnormal Security/Overview/Total Threats

```
_sourceCategory={{Logsdatasource}}  sourcetype threat_log
| json "event.attack_strategy", "event.attacked_party", "event.attack_vector",
"event.attack_type", "sourcetype", "event.auto_remediated", "event.post_remediated" as
strategy, party, vector, type, source_type, auto_remediated, post_remediated nodrop

// global filters
| where type matches "{{attack_type}}"
| where party matches "{{attack_party}}"
| where strategy matches "{{attack_strategy}}"
| where vector matches "{{attack_vector}}"
| where auto_remediated matches "{{auto_remediated}}"
| where post_remediated matches "{{post_remediated}}"

| where source_type matches ("threat_log")
| count
```

Acquia/Apache Overview/Number of Hits by Server - One Day Time Comparison

```
_sourceCategory={{_sourceCategory}} apache-access
| parse " - - - * - - [*] \"* * HTTP/1.1\" * * \"*\" \"*\" vhost=* host=* hosting_site=*
pid=* request_time=* forwarded_for=\"*\" request_id=\"*\" location=\"*\"" as
src_ip,timestamp,method,url,status_code,size,referrer,user_agent,vhost,host,hosting_site,pi
| count by host
| compare with timeshift 1d
```

Acquia/Apache Overview/Responses Over Time

```
_sourceCategory={{_sourceCategory}} apache-access
| parse " - - - * - - [*] \"* * HTTP/1.1\" * * \"*\" \"*\" vhost=* host=* hosting_site=*
pid=* request_time=* forwarded_for=\"*\" request_id=\"*\" location=\"*\"" as
src_ip,timestamp,method,url,status_code,size,referrer,user_agent,vhost,host,hosting_site,pi
| if(status_code matches "2*", 1, 0) as successes
| if(status_code matches "3*", 1, 0) as redirects
| if(status_code matches "4*", 1, 0) as client_errors
| if(status_code matches "5*", 1, 0) as server_errors
| timeslice by 1h
| sum(successes) as successes, sum(client_errors) as client_errors,  sum(redirects) as
redirects, sum(server_errors) as server_errors by _timeslice
| sort by _timeslice asc
```

Acquia/Apache Overview/Traffic Distribution by Server

```
_sourceCategory={{_sourceCategory}} apache-access
| parse " - - - * - - [*] \"* * HTTP/1.1\" * * \"*\" \"*\" vhost=* host=* hosting_site=*
pid=* request_time=* forwarded_for=\"*\" request_id=\"*\" location=\"*\"" as
src_ip,timestamp,method,url,status_code,size,referrer,user_agent,vhost,host,hosting_site,pi
| timeslice 1h
| count by _timeslice, host
| transpose row _timeslice column host
```

Acquia/Apache Overview/Traffic Volume and Bytes Served

```
_sourceCategory={{_sourceCategory}} apache-access
| parse " - - - * - - [*] \"* * HTTP/1.1\" * * \"*\" \"*\" vhost=* host=* hosting_site=*
pid=* request_time=* forwarded_for=\"*\" request_id=\"*\" location=\"*\"" as
src_ip,timestamp,method,url,status_code,size,referrer,user_agent,vhost,host,hosting_site,pi
| where size != "-"
| timeslice 1h
| (size/(1048576*1000)) as gbytes
| count as hits, sum(gbytes) as gbytes by _timeslice
| sort _timeslice asc
```

Acquia/Apache Overview/Visitor Locations

```
_sourceCategory={{_sourceCategory}} apache-access
| parse " - - - * - - [*] \"* * HTTP/1.1\" * * \"*\" \"*\" vhost=* host=* hosting_site=*
pid=* request_time=* forwarded_for=\"*\" request_id=\"*\" location=\"*\"" as
src_ip,timestamp,method,url,status_code,size,referrer,user_agent,vhost,host,hosting_site,pi
| lookup latitude, longitude, country_code, country_name, region, city, postal_code from
geo://location on ip = src_ip
| count by latitude, longitude, country_code, country_name, region, city, postal_code
| sort _count
```

Acquia/Apache Overview/Visitor Locations - One Day Time Comparison

```
_sourceCategory={{_sourceCategory}} apache-access
| parse " - - - * - - [*] \"* * HTTP/1.1\" * * \"*\" \"*\" vhost=* host=* hosting_site=*
pid=* request_time=* forwarded_for=\"*\" request_id=\"*\" location=\"*\"" as
src_ip,timestamp,method,url,status_code,size,referrer,user_agent,vhost,host,hosting_site,pi
| where isPublicIP(src_ip)
| lookup latitude, longitude, country_code, country_name, region, city, postal_code from
geo://location on ip = src_ip
| count by country_name, city
| sort _count
| compare with timeshift 1d
```

Acquia/Drupal Requests Overview/Average Response Time Outlier

```
_sourceCategory={{_sourceCategory}} drupal-requests
| parse "<133>1 * * *.* - - - [*] * * * http_code=* query=* uid=* php_pid=* php_time=*
queue_wait=* request_id=\"*\"" as
timestamp,lb,host,logtype,time,appurl,method,url,http_code,query,uid,php_id,php_time,queue_
| where query != "" or url != ""
| timeslice 1m
| avg(php_time) as seconds by _timeslice
| outlier seconds window=5,threshold=3,direction=+
```

Acquia/Drupal Requests Overview/Queue Time Outlier

```
_sourceCategory={{_sourceCategory}} drupal-requests
| parse "<133>1 * * *.* - - - [*] * * * http_code=* query=* uid=* php_pid=* php_time=*
queue_wait=* request_id=\"*\"" as
timestamp,lb,host,logtype,time,appurl,method,url,http_code,query,uid,php_id,php_time,queue_
| where queue_wait != "0"
| timeslice 1m
| count(queue_wait) as queue_wait by _timeslice
| outlier queue_wait window=5,threshold=3,consecutive=1,direction=+
```

Acquia/Drupal Requests Overview/Response Codes Overtime

```
_sourceCategory={{_sourceCategory}} drupal-requests
| parse "<133>1 * * *.* - - - [*] * * * http_code=* query=* uid=* php_pid=* php_time=*
queue_wait=* request_id=\"*\"" as
timestamp,lb,host,logtype,time,appurl,method,url,http_code,query,uid,php_id,php_time,queue_
| timeslice 1m
| count by _timeslice, http_code
| transpose row _timeslice column http_code as *
```

Acquia/Drupal Requests Overview/Top 10 Slowest Queries

```
_sourceCategory={{_sourceCategory}} drupal-requests
| parse "<133>1 * * *.* - - - [*] * * * http_code=* query=* uid=* php_pid=* php_time=*
queue_wait=* request_id=\"*\"" as
timestamp,lb,host,logtype,time,appurl,method,url,status_code,query,uid,php_id,php_time,queu
| split url delim='?' extract 1 as url, 2 as qstring
| parse "http_code=* query=* uid=* php_pid=* php_time=* queue_wait=* request_id=\"*\"" as
http_code,query,uid,php_pid,php_time,queue_wait,request_id
| where query != "" and url != ""
| avg(php_time) by query
| _avg/1000 as seconds
| fields - _avg
| sort seconds desc
| limit 10
```

Acquia/Drupal Requests Overview/Top 10 Slowest URLs

```
_sourceCategory={{_sourceCategory}} drupal-requests
| parse "<133>1 * * *.* - - - [*] * * * http_code=* query=* uid=* php_pid=* php_time=*
queue_wait=* request_id=\"*\"" as
timestamp,lb,host,logtype,time,appurl,method,url,http_code,query,uid,php_id,php_time,queue_
| split url delim='?' extract 1 as url, 2 as qstring
| where query != "" and url != ""
| avg(php_time) by url
| _avg/1000 as seconds
| fields - _avg
| sort seconds desc
| limit 10
```

Acquia/Errors Overview/Activity Over Time

```
_sourceCategory={{_sourceCategory}} drupal-watchdog
| parse ": *|*|*|*|*|*|*|*|* request_id=\"*\"" as
domain,timestamp,module,src_ip,url,referrer,uid,link,message,request_id
| timeslice 1m
| count as count by _timeslice
```

Acquia/Errors Overview/Apache Non 2XX Status Codes

```
_sourceCategory={{_sourceCategory}} apache-access
| parse " - - - * - - [*] \"* * HTTP/1.1\" * * \"*\" \"*\" vhost=* host=* hosting_site=*
pid=* request_time=* forwarded_for=\"*\" request_id=\"*\" location=\"*\"" as
src_ip,timestamp,method,url,status_code,size,referrer,user_agent,vhost,host,hosting_site,pi
| where !(status_code matches "2*")
| count
```

Acquia/Errors Overview/Cache Miss Rate Outlier

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as msg
| json field=msg "hitmiss"
| json field=msg "request_id"
| if (hitmiss="hit", 1, 0) as hit
| if (hitmiss="miss", 1, 0) as miss
| timeslice 1m
| sum(hit) as hit_cnt, sum(miss) as miss_cnt by _timeslice
| miss_cnt/hit_cnt as miss_rate
| sort _timeslice desc
| outlier miss_rate window=10, threshold=3, consecutive=1, direction=+
```

Acquia/Errors Overview/Cache Performance

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as msg
| json field=msg "handling"
| json field=msg "request_id"
| count by handling
```

Acquia/Errors Overview/Client Errors Over Time

```
_sourceCategory={{_sourceCategory}} apache-access
| parse " - - - * - - [*] \"* * HTTP/1.1\" * * \"*\" \"*\" vhost=* host=* hosting_site=*
pid=* request_time=* forwarded_for=\"*\" request_id=\"*\" location=\"*\"" as
src_ip,timestamp,method,url,status_code,size,referrer,user_agent,vhost,host,hosting_site,pi
| where status_code matches "4*"
| timeslice 5m
| count as count by _timeslice,status_code
| transpose row _timeslice column status_code as *
```

Acquia/Errors Overview/Client Location - 4xx Errors

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "request_id"
| json field=message "status"
| json field=message "client_ip"
| where status matches "4*"
| lookup latitude, longitude, country_code, country_name, region, city, postal_code from
geo://location on ip = client_ip
| count by latitude, longitude, country_code, country_name, region, city, postal_code
| sort _count
```

Acquia/Errors Overview/Drupal Requests Non 2xx Status Codes

```
_sourceCategory={{_sourceCategory}} drupal-requests //v-0a1773c0-9147-11e9-bfae-
f3b30d819671
| parse "<133>1 * * *.* - - - [*] * * * http_code=* query=* uid=* php_pid=* php_time=*
queue_wait=* request_id=\"*\"" as
timestamp,lb,host,logtype,time,appurl,method,url,http_code,query,uid,php_id,php_time,queue_
| split url delim='?' extract 1 as url, 2 as qstring
| where !(http_code matches "2*")
| count
```

Acquia/Errors Overview/Errors by Domain

```
_sourceCategory={{_sourceCategory}} drupal-watchdog
| parse ": *|*|*|*|*|*|*|*|* request_id=\"*\"" as
domain,timestamp,module,src_ip,url,referrer,uid,link,message,request_id
| where message matches "*error*"
| count as count by domain
```

Acquia/Errors Overview/Errors by Module

```
_sourceCategory={{_sourceCategory}} drupal-watchdog error
| parse ": *|*|*|*|*|*|*|*|* request_id=\"*\"" as
domain,timestamp,module,src_ip,url,referrer,uid,link,message,request_id
| where message matches "*error*"
| count as count by module
```

Acquia/Errors Overview/Errors by URL

```
_sourceCategory={{_sourceCategory}} drupal-watchdog
| parse ": *|*|*|*|*|*|*|*|* request_id=\"*\"" as
domain,timestamp,module,src_ip,url,referrer,uid,link,message,request_id
| where message matches "*error*"
| split url delim='?' extract 1 as url, 2 as qstring
| count as count by url
```

Acquia/Errors Overview/Errors Outlier Over Time

```
_sourceCategory={{_sourceCategory}} drupal-watchdog
| parse ": *|*|*|*|*|*|*|*|* request_id=\"*\"" as
domain,timestamp,module,src_ip,url,referrer,uid,link,message,request_id
| where message matches "*error*"
| timeslice 1m
| count as count by _timeslice
| outlier count window=5,direction=+,threshold=3
```

Acquia/Errors Overview/Non 2xx Status Codes by Location

```
_sourceCategory={{_sourceCategory}} apache-access
| parse " - - - * - - [*] \"* * HTTP/1.1\" * * \"*\" \"*\" vhost=* host=* hosting_site=*
pid=* request_time=* forwarded_for=\"*\" request_id=\"*\" location=\"*\"" as
src_ip,timestamp,method,url,status_code,size,referrer,user_agent,vhost,host,hosting_site,pi
| where status_code matches "4*" or status_code matches "5*"
| lookup latitude, longitude, country_code, country_name, region, city, postal_code from
geo://location on ip = src_ip
| count by latitude, longitude, country_code, country_name, region, city, postal_code
| sort _count
```

Acquia/Errors Overview/Non 2xx Status Codes Over Time

```
_sourceCategory={{_sourceCategory}} apache-access
| parse " - - - * - - [*] \"* * HTTP/1.1\" * * \"*\" \"*\" vhost=* host=* hosting_site=*
pid=* request_time=* forwarded_for=\"*\" request_id=\"*\" location=\"*\"" as
src_ip,timestamp,method,url,status_code,size,referrer,user_agent,vhost,host,hosting_site,pi
| where !(status_code matches "2*")
| timeslice 1m
| count(status_code) as status_code by _timeslice
| outlier status_code window=5,threshold=3,consecutive=1,direction=+
```

Acquia/Errors Overview/PHP Error Details

```
_sourceCategory={{_sourceCategory}} php-errors
| parse "* * * * - - - [*] *: * request_id=\"*\"" as
head,systime,env,host,time,type,message,request_id
| count as count by Type, message
| sort by count
```

Acquia/Errors Overview/PHP Error Outlier

```
_sourceCategory={{_sourceCategory}} php-errors
| timeslice 1m
| count by _timeslice
| outlier _count window=5,threshold=3,consecutive=2,direction=+
```

Acquia/Errors Overview/PHP Errors Over Time

```
_sourceCategory={{_sourceCategory}} php-errors
| parse "* * * * - - - [*] *: * request_id=\"*\"" as
head,systime,env,host,time,type,message,request_id
| timeslice 5m
| count as count by _timeslice,type
| transpose row _timeslice column type as *
```

Acquia/Errors Overview/Server Errors Over Time

```
_sourceCategory={{_sourceCategory}} apache-access
| parse " - - - * - - [*] \"* * HTTP/1.1\" * * \"*\" \"*\" vhost=* host=* hosting_site=*
pid=* request_time=* forwarded_for=\"*\" request_id=\"*\" location=\"*\"" as
src_ip,timestamp,method,url,status_code,size,referrer,user_agent,vhost,host,hosting_site,pi
| where status_code matches "5*"
| timeslice 5m
| count as count by _timeslice,status_code
| transpose row _timeslice column status_code as *
```

Acquia/Errors Overview/Top 10 Error Codes

```
_sourceCategory={{_sourceCategory}} apache-error
| parse " vhost=* forwarded_for=\"*\" request_id=\"*\" hosting_site=* *: *" as
vhost,forwarded_for,request_id,hosting_site,error_code,error
| count by error_code
| sort by _count
| limit 10
```

Acquia/Errors Overview/Varnish Non 2xx Status Codes

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "status"
| json field=message "request_id"
| where !(status matches "2*")
| count
```

Acquia/Errors Overview/Watchdog Errors Over time

```
_sourceCategory={{_sourceCategory}} drupal-watchdog
| parse ": *|*|*|*|*|*|*|*|* request_id=\"*\"" as
domain,timestamp,module,src_ip,url,referrer,uid,link,message,request_id
| where message matches "*error*"
| timeslice 1m
| count as count by _timeslice
```

Acquia/FPM Overview/Average CPU Consumed by Timeslice

```
_sourceCategory={{_sourceCategory}} fpm-access !AH_FPM_STATUS
| parse "\"* /*\" * memory_kb=* %cpu=* duration_ms=* request_id=\"*\"" as
method,url,status_code,memory_kb,cpu,duration_ms,request_id
| memory_kb / 1000 as memory_mb
| timeslice 1m
| avg(cpu) as cpu_percent by _timeslice
```

Acquia/FPM Overview/Average Memory Consumed by Timeslice

```
_sourceCategory={{_sourceCategory}} fpm-access !AH_FPM_STATUS
| parse "\"* /*\" * memory_kb=* %cpu=* duration_ms=* request_id=\"*\"" as
method,url,status_code,memory_kb,cpu,duration_ms,request_id
| memory_kb / 1000 as memory_mb
| timeslice 1m
| avg(memory_mb) as memory by _timeslice
```

Acquia/FPM Overview/Non 2** Response Codes

```
_sourceCategory={{_sourceCategory}} fpm-access
| parse "\"* /*\" * memory_kb=* %cpu=* duration_ms=* request_id=\"*\"" as
method,url,status_code,memory_kb,cpu,duration_ms,request_id
| where !(status_code matches "2*")
| count as count by status_code
| sort by count
```

Acquia/FPM Overview/Response Time Outlier

```
_sourceCategory={{_sourceCategory}} fpm-access !AH_FPM_STATUS
| parse "\"* /*\" * memory_kb=* %cpu=* duration_ms=* request_id=\"*\"" as
method,url,status_code,memory_kb,cpu,duration_ms,request_id
| timeslice 1m
| avg(duration_ms) as duration by _timeslice
| outlier duration window=5,threshold=3,consecutive=1,direction=+
```

Acquia/FPM Overview/Status Codes Over Time

```
_sourceCategory={{_sourceCategory}} fpm-access
| parse "\"* /*\" * memory_kb=* %cpu=* duration_ms=* request_id=\"*\"" as
method,url,status_code,memory_kb,cpu,duration_ms,request_id
| timeslice 1m
| count as count by status_code, _timeslice
| transpose row _timeslice column status_code
```

Acquia/Overview/Apache Request Volume

```
_sourceCategory={{_sourceCategory}} apache-access
| parse regex "(?<src_ip>\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})" nodrop
| parse regex "(?<method>[A-Z]+)\s(?<url>\S+)\sHTTP/[\d\.]+\"\s(?<status_code>\d+)\s(?
<size>[\d-]+)" nodrop
| parse regex "(?<method>[A-Z]+)\s(?<url>\S+)\sHTTP/[\d\.]+\"\s(?<status_code>\d+)\s(?
<size>[\d-]+)\s\"(?<referrer>.*?)\"\s\"(?<user_agent>.+?)\".*" nodrop
| timeslice 5m
| count by _timeslice
```

Acquia/Overview/Apache Response Size Over Time

```
_sourceCategory={{_sourceCategory}} apache-access
| parse regex "(?<src_ip>\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})" nodrop
| parse regex "(?<method>[A-Z]+)\s(?<url>\S+)\sHTTP/[\d\.]+\"\s(?<status_code>\d+)\s(?
<size>[\d-]+)" nodrop
| parse regex "(?<method>[A-Z]+)\s(?<url>\S+)\sHTTP/[\d\.]+\"\s(?<status_code>\d+)\s(?
<size>[\d-]+)\s\"(?<referrer>.*?)\"\s\"(?<user_agent>.+?)\".*" nodrop
| timeslice 1h
| where _timeslice > queryStartTime() and __timeslice_end < queryEndTime()
| count as totalCount, sum(size) as size by _timeslice
```

Acquia/Overview/Apache Response Time Outlier

```
_sourceCategory={{_sourceCategory}} apache-access
| parse "vhost=* host=* hosting_site=* pid=* request_time=* forwarded_for=\"*\"
request_id=\"*\" location=\"*\"" as
vhost,host,hosting_site,pid,request_time,forwarded_for,request_id,location
| timeslice 1m
| toLong(request_time/(1000000)) as request_time
| avg(request_time) as seconds by _timeslice
| outlier seconds window=5,threshold=3,consecutive=1, direction=+
```

Acquia/Overview/Bot Traffic Over Time

```
_sourceCategory={{_sourceCategory}} (apache-access or varnishncsa)
| parse " - - - * - - [*] \"* * HTTP/1.1\" * * \"*\" \"*\" vhost=* host=* hosting_site=*
pid=* request_time=* forwarded_for=\"*\" request_id=\"*\" location=\"*\"" as
src_ip,timestamp,method,url,status_code,size,referrer,user_agent,vhost,host,hosting_site,pi
nodrop
| parse "\"user_agent\":\"*\"," as user_agent
| timeslice 1m
| where user_agent matches "*bot*" or user_agent matches "*index*"
| count by user_agent, _timeslice
| transpose row _timeslice column user_agent
```

Acquia/Overview/PHP Errors Over Time

```
_sourceCategory={{_sourceCategory}} php-errors
| parse "* * * * - - - [*] *: * request_id=\"*\"" as
head,systime,env,host,time,type,message,request_id
| timeslice 5m
| count by type, _timeslice
| transpose row _timeslice column type as *
```

Acquia/Overview/Requests by Domain

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "\"host\":\"*\"," as host nodrop
| count by host
```

Acquia/Overview/Top 10 Referrers

```
_sourceCategory={{_sourceCategory}} (apache-access OR varnishncsa)
| parse " - - - * - - [*] \"* * HTTP/1.1\" * * \"*\" \"*\" vhost=* host=* hosting_site=*
pid=* request_time=* forwarded_for=\"*\" request_id=\"*\" location=\"*\"" as
src_ip,timestamp,method,url,status_code,size,referrer,user_agent,vhost,host,hosting_site,pi
nodrop
| parse "\"referrer\":\"*\"," as referrer
| where referrer != "-"
| count by referrer | top 10 referrer by _count
```

Acquia/Overview/Top 5 User Agents

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "\"user_agent\":\"*\"," as user_agent
| where user_agent != "-"
| urldecode(user_agent) as user_agent
| count by user_agent
| sort _count
| limit 5
```

Acquia/Overview/Top Requests

```
_sourceCategory={{_sourceCategory}} (apache-access OR varnishncsa)
| parse " - - - * - - [*] \"* * HTTP/1.1\" * * \"*\" \"*\" vhost=* host=* hosting_site=*
pid=* request_time=* forwarded_for=\"*\" request_id=\"*\" location=\"*\"" as
src_ip,timestamp,method,url,status_code,size,referrer,user_agent,vhost,host,hosting_site,pi
nodrop
| parse "\"url\":\"*\"," as url
| urldecode(url) as url
| count by url | top 10 url by _count
```

Acquia/Overview/Traffic by Host Over Time

```
_sourceCategory={{_sourceCategory}} varnishncsa
|parse "<133>1 * * *.* - - - *" as timestamp,host,site,reportType,msg
| timeslice 1m
| count by host, _timeslice
| transpose row _timeslice column host as *
```

Acquia/Overview/Traffic by Site Over Time

```
_sourceCategory={{_sourceCategory}} varnishncsa
|parse "<133>1 * * *.* - - - *" as timestamp,host,site,reportType,msg
| timeslice 1m
| count by site, _timeslice
| transpose row _timeslice column site as *
```

Acquia/Overview/Varnish Cache Performance

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "handling"
| count by handling
```

Acquia/Overview/Varnish Requests by Country

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse regex "(?<src_ip>\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})"
| lookup latitude, longitude, country_code from geo://location on ip = src_ip
| count by country_code
```

Acquia/Overview/Varnish Response Time Outlier

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as msg
| json field=msg "time_firstbyte"
| timeslice 1m
| avg(time_firstbyte) as seconds by _timeslice
| outlier seconds window=5,threshold=3,consecutive=1, direction=+
```

Acquia/Overview/Visitor Geolocations

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse regex "(?<src_ip>\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})" nodrop
| lookup latitude, longitude, country_code, country_name, region, city, postal_code from
geo://location on ip = src_ip
| count by latitude, longitude, country_code, country_name, region, city, postal_code
| sort _count
```

Acquia/Threat Analysis/Threat Breakdown by Sources

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "client_ip" as src_ip
|count as ip_count by src_ip, _source
| lookup type, actor, raw, threatlevel as malicious_confidence from sumo://threat/cs on
threat=src_ip
| json field=raw "labels[*].name" as label_name
| replace(label_name, "\\/","->") as label_name
| replace(label_name, "\""," ") as label_name
| where  type="ip_address" and !isNull(malicious_confidence)
| if (isEmpty(actor), "Unassigned", actor) as Actor
|sum (ip_count) as threat_count by _source
```

Acquia/Threat Analysis/Threat by Actors

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "client_ip" as src_ip
|count as ip_count by src_ip
| lookup type, actor, raw, threatlevel as malicious_confidence from sumo://threat/cs on
threat=src_ip
| json field=raw "labels[*].name" as label_name
| replace(label_name, "\\/","->") as label_name
| replace(label_name, "\""," ") as label_name
| where  type="ip_address" and !isNull(malicious_confidence)
| if (isEmpty(actor), "Unassigned", actor) as Actor
|sum (ip_count) as threat_count by Actor
```

Acquia/Threat Analysis/Threat by Geo Location

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "client_ip" as src_ip
|count as ip_count by src_ip
| lookup type, actor, raw, threatlevel as malicious_confidence from sumo://threat/cs on
threat=src_ip
| json field=raw "labels[*].name" as label_name
| replace(label_name, "\\/","->") as label_name
| replace(label_name, "\""," ") as label_name
| where  type="ip_address" and !isNull(malicious_confidence)
| lookup latitude, longitude, country_code, country_name, region, city, postal_code from
geo://location on ip = src_ip
| count by latitude, longitude, country_code, country_name, region, city, postal_code,
malicious_confidence
| sort _count
```

Acquia/Threat Analysis/Threat Count

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "client_ip" as src_ip
|count as ip_count by src_ip
| lookup type, actor, raw, threatlevel as malicious_confidence from sumo://threat/cs on
threat=src_ip
| json field=raw "labels[*].name" as label_name
| replace(label_name, "\\/","->") as label_name
| replace(label_name, "\""," ") as label_name
| where  type="ip_address" and !isNull(malicious_confidence)
| if (isEmpty(actor), "Unassigned", actor) as Actor
|sum (ip_count) as threat_count
```

Acquia/Threat Analysis/Threat Table

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "client_ip" as src_ip
| lookup type, actor, raw, threatlevel as malicious_confidence from sumo://threat/cs on
threat=src_ip
| where  type="ip_address" and !isNull(malicious_confidence)
| json field=raw "labels[*].name" as label_name
| replace(label_name, "\\/","->") as label_name
| replace(label_name, "\""," ") as label_name
| if (isEmpty(actor), "Unassigned", actor) as Actor
|count by src_ip,method, status_code, size, referrer, user_agent
| sort by _count
```

Acquia/Threat Analysis/Threats by Malicious Confidence

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "client_ip" as src_ip
|count as ip_count by src_ip
| lookup type, actor, raw, threatlevel as malicious_confidence from sumo://threat/cs on
threat=src_ip
| json field=raw "labels[*].name" as label_name
| replace(label_name, "\\/","->") as label_name
| replace(label_name, "\""," ") as label_name
| where  type="ip_address" and !isNull(malicious_confidence)
| if (isEmpty(actor), "Unassigned", actor) as Actor
|sum (ip_count) as threat_count by malicious_confidence
```

Acquia/Varnish Overview/Cache Performance Over Time

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "handling"
| json field=message "host"
| timeslice 1m
|where host matches "{{host}}"
|count by handling, _timeslice
| transpose row _timeslice column handling
```

Acquia/Varnish Overview/Number of Hits by Host - One Day Time Comparison

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "host"
|where host matches "{{host}}"
|count by host
| compare with timeshift 1d
```

Acquia/Varnish Overview/Responses Over Time

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "host"
| json field=message "status" as status_code
| if(status_code matches "2*", 1, 0) as successes
| if(status_code matches "3*", 1, 0) as redirects
| if(status_code matches "4*", 1, 0) as client_errors
| if(status_code matches "5*", 1, 0) as server_errors
| timeslice by 1h
|where host matches "{{host}}"
|sum(successes) as successes, sum(client_errors) as client_errors,  sum(redirects) as
redirects, sum(server_errors) as server_errors by _timeslice
| sort by _timeslice asc
```

Acquia/Varnish Overview/Traffic Distribution by Host

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "host"
| timeslice 1h
|where host matches "{{host}}"
|count by _timeslice,host
| transpose row _timeslice column host
```

Acquia/Varnish Overview/Traffic Volume and Bytes Served

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "bytes" as size
| json field=message "host"
| where size != "-"
| timeslice 1h
| (size/1024/1024) as mbytes
|where host matches "{{host}}"
|count as hits, sum(mbytes) as mbytes by _timeslice
| sort _timeslice asc
```

Acquia/Varnish Overview/Visitor Locations

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "client_ip"
| json field=message "host"
| lookup latitude, longitude, country_code, country_name, region, city, postal_code from
geo://location on ip = client_ip
|where host matches "{{host}}"
|count by latitude, longitude, country_code, country_name, region, city, postal_code
| sort _count
```

Acquia/Varnish Overview/Visitor Locations - One Day Time Comparison

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "client_ip"
| json field=message "host"
| where isPublicIP(client_ip)
| lookup latitude, longitude, country_code, country_name, region, city, postal_code from
geo://location on ip = client_ip
|where host matches "{{host}}"
|count by country_name, city
| sort _count
| compare with timeshift 1d
```

Acquia/Varnish Visitor Access Types/Browsers and Operating Systems

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "host"
| json field=message "user_agent" as agent
| if (agent matches "*Windows *" OR agent matches "*Win32*" OR agent matches
"*Win64*","Windows","") as OS
| if (agent matches "*Macintosh*" OR agent matches "*Darwin/*" OR agent matches "*Mac
OS*","MacOS",OS) as OS
| if (agent matches "* CrOS *","Chrome OS",OS) as OS
| if (agent matches "*Linux*","Linux",OS) as OS
| if (agent matches "*iPad*","iPad",OS) as OS
| if (agent matches "*iPhone*","iPhone",OS) as OS
|  if (agent matches "*Android*","Android",OS) as OS
| if (agent matches "*Windows Phone*","Windows Phone",OS) as OS
| if (OS == "","Other",OS) as OS
| if (agent matches "Mozilla/*; rv:*)*","Mozilla","") as Browser
| if (agent matches "*MSIE*","Internet Explorer",Browser) as Browser
| if (agent matches "*Firefox*","Firefox",Browser) as Browser
| if (agent matches "*Safari*","Safari",Browser) as Browser
| if (OS=="Android" AND agent matches "*WebKit*","WebKit",Browser) as Browser
| if ((OS=="iPhone" OR OS=="iPad") AND (agent matches "*Mobile/*" OR agent matches
"*AppleWebKit*(KHTML*Gecko)*"),"Mobile Safari",Browser) as Browser
| if (browser ="" AND OS=="MacOS" AND agent matches "Mozilla/* (Macintosh;*(KHTML, like
Gecko)*","Safari",Browser) as Browser
| if (agent matches "*MobileSafari/*","Mobile Safari",Browser) as Browser
| if (agent matches "*Chrome*","Chrome",Browser) as Browser
| if (agent matches "Opera*","Opera",Browser) as Browser
| if (agent matches "Dolphin*","Dolphin",Browser) as Browser
|  if (Browser == "","Other",Browser) as Browser
|where host matches "{{host}}"
|count by OS,browser
| transpose row os column browser as *
```

Acquia/Varnish Visitor Access Types/Popular Mobile Device Versions

```
_sourceCategory={{_sourceCategory}} varnishncsa ( iphone or ipad or android or samsung)
| parse "- - - *" as message
| json field=message "host"
| parse regex "\((?<device>iPhone).+? CPU iPhone OS (?<version>.+?) like Mac"  nodrop
| parse regex "\((?<device>iPad).+? CPU OS (?<version>.+?) like Mac"  nodrop
| parse regex " (?<device>Android) (?<version>[\d\.]+)" nodrop
| parse regex "(?<device>SAMSUNG).+?(?<version>(?:GT-\w+|SGH-\w+|SPH-\w+|SCH-\w+))"
|where host matches "{{host}}"
|count as count by device,version
| sort count
| limit 10
```

Acquia/Varnish Visitor Access Types/Top 10 PC and Mac Versions

```
_sourceCategory={{_sourceCategory}} varnishncsa   ("macintosh" OR "mac os" OR "windows" OR
"i686" or "PC" or ("Linux" AND !android))
| parse "- - - *" as message
| json field=message "host"
| json field=message "user_agent" as agent
| parse regex field=agent "(?<os>Mac OS) (?<version>[^;\)]+?)(?:;|\))" nodrop
| parse regex field=agent "(?<os>Windows)(?: NT | )(?<version>[\d.]+)" nodrop
| parse regex field=agent "(?<os>Linux) (?<version>\S+?)(?:\)|;)" nodrop
| where os!=""
|where host matches "{{host}}"
|count as count by os,version
| sort count
| limit 10
```

Acquia/Varnish Visitor Access Types/Vistor Platforms

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "user_agent" as agent
| json field=message "host"
| if (agent matches "*iPad*" OR agent matches "*iPhone*" OR agent matches "*Android*" OR
agent matches "*BlackBerry*" OR agent matches "*Galaxy*" OR agent matches "LGE-*" OR agent
matches "SCH-*" OR agent matches "*HUAWEI-*" OR agent matches "*SAMSUNG*" OR agent matches
"*samsung*" OR agent matches "Dalvik/*" OR agent matches "Nokia*" OR agent matches "LG-*"
OR agent matches "*CFNetwork*"  OR agent matches "*UP.Browser/*" OR agent matches
"*QQ*rowser*" OR agent matches "*NetFront/*" OR agent matches "*Windows Phone*" OR agent
matches "*Mobile Safari*" OR agent matches "*iemobile*","Mobile","") as type
| if (agent matches "*Windows*" OR agent matches "*Win64*" OR agent matches "*Win32*" OR
agent matches "*i686*" OR agent matches "*x86*","PC",type) as type
| if (agent matches "*Macintosh*" OR agent matches "*Mac OS*","Mac",type) as type
| if (type = "","bots/unknown",type) as type
|where host matches "{{host}}"
|count type
```

Acquia/Varnish Visitor Locations/United States

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "client_ip" as client_ip
| json field=message "host" as host
| lookup latitude, longitude, country_code, country_name, region, city, postal_code from
geo://location on ip = client_ip
| where country_name="United States"
|where host matches "{{host}}"
|count by latitude, longitude, country_code, country_name, region, city, postal_code
| sort _count
```

Acquia/Varnish Visitor Locations/Visits by Country Over Time

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "client_ip" as client_ip
| json field=message "host" as host
| lookup latitude, longitude, country_code, country_name, region, city, postal_code from
geo://location on ip = client_ip
| where !isNull(country_name) AND country_name !=""
| timeslice 5m
|where host matches "{{host}}"
|count by _timeslice,country_name
| transpose row _timeslice column country_name as *
```

Acquia/Varnish Visitor Locations/Visits by US State Over Time

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "client_ip" as client_ip
| json field=message "host" as host
| lookup latitude, longitude, country_code, country_name, region, city, postal_code from
geo://location on ip = client_ip
| timeslice 5m
| where country_name="United States" AND !isNull(region) AND region !=""
|where host matches "{{host}}"
|count by _timeslice,region
| transpose row _timeslice column region as *
```

Acquia/Varnish Visitor Locations/Worldwide

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "client_ip" as client_ip
| json field=message "host" as host
| lookup latitude, longitude, country_code, country_name, region, city, postal_code from
geo://location on ip = client_ip
|where host matches "{{host}}"
|count by latitude, longitude, country_code, country_name, region, city, postal_code
| sort _count
```

Acquia/Varnish Visitor Traffic Insight/Media Types Served

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "url" as url
| json field=message "host" as host
| parse regex field=url "^[^\?]+?\.(?<type>[a-zA-Z]{2,4})$" nodrop
| parse regex field=url "/\S+?(?<email_prefix>(?:%40|@)[^.]+?)\.\w+"
| where email_prefix=""
|where host matches "{{host}}"
|count as count by type
```

Acquia/Varnish Visitor Traffic Insight/Top 10 Search Terms from Popular Search Engines

```
_sourceCategory={{_sourceCategory}} varnishncsa  (google OR bing OR aol OR ask OR yahoo)
("p=" OR "q=" OR "wd=" OR "searchfor=")
| parse "- - - *" as message
| json field=message "referrer" as referrer
| json field=message "host" as host
| parse regex field=referrer "(?:\?|&)(?:p|q|wd|searchfor)=(?<search_term>[^=]+?)(?:&|$)"
nodrop
| where search_term!=""
| urldecode(search_term)
| if(referrer matches "*google.*","Google","") as engine
| if(referrer matches "*bing.*","Bing",engine) as engine
| if(referrer matches "*ask.com*","Ask.com",engine) as engine
| if(referrer matches "*aol.*","AOL",engine) as engine
| if(referrer matches "*yahoo.*","Yahoo",engine) as engine
| where engine !=""
|where host matches "{{host}}"
|count as count  by search_term
| sort count
| limit 10
```

Acquia/Varnish Visitor Traffic Insight/Top Documents

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "url" as url
| json field=message "host" as host
|where host matches "{{host}}"
|count as count by url
| sort by count
| limit 10
```

Acquia/Varnish Visitor Traffic Insight/Top Referrers

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "referrer" as referrer
| json field=message "host" as host
| where referrer != "-"
|where host matches "{{host}}"
|count by referrer
| sort by _count
| limit 5
```

Acquia/Varnish Web Server Operations/Client Locations - 4xx Errors

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "status" as status_code
| json field=message "client_ip" as client_ip
| json field=message "host" as host
| where status_code matches "4*"
| lookup latitude, longitude, country_code, country_name, region, city, postal_code from
geo://location on ip = client_ip
|where host matches "{{host}}"
|count by latitude, longitude, country_code, country_name, region, city, postal_code
| sort _count
```

Acquia/Varnish Web Server Operations/Error Responses by Server

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "status" as status_code
| json field=message "host" as host
| num(status_code)
| where status_code >= 400
| timeslice 5m
|where host matches "{{host}}"
|count as count by _timeslice, host
| transpose row _timeslice column host
```

Acquia/Varnish Web Server Operations/Errors by Environment

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "status" as status_code
| json field=message "ah_environment" as env
| json field=message "host" as host
| where status_code matches "5*"
| timeslice 5m
|where host matches "{{host}}"
|count as count by _timeslice, env
| transpose row _timeslice column env as *
```

Acquia/Varnish Web Server Operations/Non 200 Response Status Codes

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "status" as status_code
| json field=message "host" as host
| where !(status_code matches "2*")
|where host matches "{{host}}"
|count as count by status_code
| sort by count
```

Acquia/Varnish Web Server Operations/Server Errors Over Time

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "status" as status_code
| json field=message "host" as host
| where status_code matches "5*"
| timeslice 5m
|where host matches "{{host}}"
|count as count by _timeslice,status_code
| transpose row _timeslice column status_code as *
```

Acquia/Varnish Web Server Operations/Top 10 Bots Observed

```
_sourceCategory={{_sourceCategory}} varnishncsa ("Googlebot" OR "AskJeeves" OR "Digger" OR
"Lycos"
OR "msnbot" OR "Inktomi Slurp" OR "Yahoo" OR "Nutch" OR "bingbot" OR
"BingPreview" OR "Mediapartners-Google" OR "proximic" OR "AhrefsBot" OR
"AdsBot-Google" OR "Ezooms" OR "AddThis.com" OR "facebookexternalhit" OR
"MetaURI" OR "Feedfetcher-Google" OR "PaperLiBot" OR "TweetmemeBot" OR
"Sogou web spider" OR "GoogleProducer" OR "RockmeltEmbedder" OR
"ShareThisFetcher" OR "YandexBot" OR "rogerbot-crawler" OR "ShowyouBot" OR "Baiduspider"
OR "Sosospider" OR "Exabot")
| parse "- - - *" as message
| json field=message "user_agent" as agent
| json field=message "host" as host
| parse regex field=agent "(?<bot_name>facebook)externalhit?\W+" nodrop
| parse regex field=agent "Feedfetcher-(?<bot_name>Google?)\S+" nodrop
| parse regex field=agent "(?<bot_name>PaperLiBot?)/.+" nodrop
| parse regex field=agent "(?<bot_name>TweetmemeBot?)/.+" nodrop
| parse regex field=agent "(?<bot_name>msn?)bot\W" nodrop
| parse regex field=agent "(?<bot_name>Nutch?)-.+" nodrop
| parse regex field=agent "(?<bot_name>Google?)bot\W" nodrop
| parse regex field=agent "Feedfetcher-(?<bot_name>Google?)\W" nodrop
| parse regex field=agent "(?<bot_name>Yahoo?)!\s+Slurp[;/].+" nodrop
| parse regex field=agent "(?<bot_name>bing?)bot\W" nodrop
| parse regex field=agent "(?<bot_name>Bing?)Preview\W" nodrop
| parse regex field=agent "(?<bot_name>Sogou?)\s+web\s" nodrop
| parse regex field=agent "(?<bot_name>Yandex?)Bot\W" nodrop
| parse regex field=agent "(?<bot_name>rogerbot?)\W" nodrop
| parse regex field=agent "(?<bot_name>AddThis\.com?)\s+robot\s+" nodrop
| parse regex field=agent "(?<bot_name>ShareThis?)Fetcher/.+" nodrop
| parse regex field=agent "(?<bot_name>Ahrefs?)Bot/.+" nodrop
| parse regex field=agent "(?<bot_name>MetaURI?)\s+API/.+" nodrop
| parse regex field=agent "(?<bot_name>Showyou?)Bot\s+" nodrop
| parse regex field=agent "(?<bot_name>Google?)Producer;" nodrop
| parse regex field=agent "(?<bot_name>Ezooms?)\W" nodrop
| parse regex field=agent "(?<bot_name>Rockmelt?)Embedder\s+" nodrop
| parse regex field=agent "(?<bot_name>Sosospider?)\W" nodrop
| parse regex field=agent "(?<bot_name>Baidu?)spider" nodrop
| parse regex field=agent "(?<bot_name>Exabot?)\W"
| where bot_name != ""
| if (bot_name="bing","Bing",bot_name) as bot_name
|where host matches "{{host}}"
|count as count by bot_name
| sort by count
| limit 10
```

Acquia/Varnish Web Server Operations/Top 5 Clients Causing 4xx Errors

```
_sourceCategory={{_sourceCategory}} varnishncsa (400 OR 401 OR 402 OR 403 OR 404)
| parse "- - - *" as message
| json field=message "client_ip" as client_ip
| json field=message "status" as status_code
| json field=message "host" as host
| where status_code matches "4*"
|where host matches "{{host}}"
|count as count by client_ip
| sort count
| limit 5
```

Acquia/Varnish Web Server Operations/Top 5 URIs causing 404 Responses

```
_sourceCategory={{_sourceCategory}} varnishncsa 404
| parse "- - - *" as message
| json field=message "status" as status_code
| json field=message "url" as url
| json field=message "host" as host
| where status_code="404"
|where host matches "{{host}}"
|count as count by url
| sort count
| limit 5
```

Acquia/Varnish Web Server Operations/Top Clients

```
_sourceCategory={{_sourceCategory}} varnishncsa
| parse "- - - *" as message
| json field=message "host" as host
| json field=message "client_ip" as client_ip
| lookup country_name from geo://location on ip = client_ip
|where host matches "{{host}}"
|count by client_ip, country_name
| sort _count
| limit 10
```

Active Directory 2012+ - OpenTelemetry/Active Directory Service Activity/Category Over Time

```
sumo.datasource=windows
| json "event_id", "computer", "message", "task" as event_id, host.name, msg_summary, task
nodrop
| parse regex field=msg_summary "(?<msg_summary>.*\.*)" nodrop
| where host.name matches "{{host.name}}"
| timeslice 1h
| count by _timeslice, task
| transpose row _timeslice column task
```

Active Directory 2012+ - OpenTelemetry/Active Directory Service Activity/Object Creation

```
sumo.datasource=windows  "\"Channel\":\"Security\"" (4720 or 4741 or 4727 or 4731 or 4744
or 4749 or 4754 or 4759 or 4783 or 5137 or 4902 or 4790)
| json "event_id", "computer", "message" as event_id, host.name, msg_summary nodrop
| json field=event_id "id" as event_id
| if ((event_id in ("4720")), "user", if ((event_id in ("4720", "4741")), "computer", if
((event_id in ("4727", "4731", "4744", "4749", "4754", "4759", "4783")), "group", if
((event_id in ("5137", "4902", "4790")), "object", "")))) as object_created
| where object_created != "" and host.name matches "{{host.name}}"
| timeslice 1h
| count by object_created, _timeslice
| transpose row _timeslice column object_created
| fillmissing timeslice(1h)
```

Active Directory 2012+ - OpenTelemetry/Active Directory Service Activity/Object Deletion

```
sumo.datasource=windows (4726 or 4743 or 4730 or 4734 or 4748 or 4753 or 4758 or 4763 or
4789 or 4659 or 4660 or 5141)
| json "event_id", "computer", "message", "channel" as event_id, host, msg_summary,
channel nodrop
| json field=event_id "id" as event_id
| if ((event_id in ("4726")), "user", if ((event_id in ("4743")), "computer", if
((event_id in ("4730", "4734", "4748", "4753", "4758", "4763", "4789")), "group", if
((event_id in ("4659", "4660", "5141")), "object", "") ))) as object_deleted
| where object_deleted != "" and host matches "{{host.name}}"
| timeslice 1h
| count by object_deleted, _timeslice
| transpose row _timeslice column object_deleted
| fillmissing timeslice(1h)
```

Active Directory 2012+ - OpenTelemetry/Active Directory Service Activity/off Activity

```
sumo.datasource=windows
| json "event_id", "computer", "event_data", "channel", "message"  as event_id, host,
event_data,  channel, msg_summary nodrop
| json field=event_id "id" as event_id
| parse field=event_data "\"LogonType\":\"*\"" as logon_type
| where event_id in ("4624", "4634") and channel = "Security" and host matches "
{{host.name}}"
| if (logon_type = "2", "interactive", if (logon_type = "3", "network", if (logon_type =
"4", "batch", if (logon_type = "5", "service", if (logon_type = "7", "unlock", if
(logon_type = "8", "networkcleartext", if (logon_type = "9", "newcredentials", if
(logon_type = "10", "remoteinteractive", if (logon_type = "11", "cachedinteractive", ""
))))))))) as type
| if (event_id = "4624", "logon", "logged off") as addonstr
| concat (type, " ", addonstr) as type
| where type != ""
| timeslice 1h
| count by _timeslice, type
| transpose row _timeslice column type
```

Active Directory 2012+ - OpenTelemetry/Active Directory Service Activity/Rights Management

```
sumo.datasource=windows
| json "event_id", "computer" as event_id, host nodrop
| json field=event_id "id" as event_id
| where event_id in ("4704", "4705") and host matches "{{host.name}}"
| if ((event_id = "4704"), "assigned", if ((event_id = "4705"), "removed", "")) as right
| timeslice 1h
| count by right, _timeslice
| transpose row _timeslice column right
| fillmissing timeslice(1h)
```

Active Directory 2012+ - OpenTelemetry/Active Directory Service Activity/Top 10 Messages

```
sumo.datasource=windows
| json "event_id", "computer", "message" as event_id, host.name, msg_summary nodrop
| parse regex field=msg_summary "(?<msg_summary>.*\.*)" nodrop
| where !isEmpty(msg_summary)
| where host.name matches "{{host.name}}"
| count by msg_summary
| top 10 msg_summary by _count
```

Active Directory 2012+ - OpenTelemetry/Active Directory Service Failures/Admin Activity by Category

```
sumo.datasource=windows  Task
| json "computer", "message", "keywords", "event_data", "channel", "task" as  host,
msg_summary, Keywords, event_data, channel, task nodrop
| parse regex field=msg_summary "(?<msg_summary>.*\.*)" nodrop
| parse field=event_data "\"SubjectUserName\":\"*\"" as src_user
| parse field=event_data "\"TargetUserName\":\"*\"" as dest_user
| where (src_user matches "Administrator*" or dest_user matches "Administrator*") and host
matches "{{host.name}}"
| count by task, Keywords, msg_summary
| topk(10, _count) by task | fields -_rank
```

Active Directory 2012+ - OpenTelemetry/Active Directory Service Failures/All Failures by IP

```
sumo.datasource=windows  ("fatal" or "failure" or "error" or "Audit Failure")
| json "event_id", "computer", "keywords", "event_data" as event_id, host, Keywords,
event_data nodrop
| parse field=event_data "\"IpAddress\":\"*\"" as src_ip
| where !jsonArrayContains(Keywords, "Audit Success") and !isEmpty(src_ip) and src_ip !=
"-" and host matches "{{host.name}}"
| count by src_ip
| top 10 src_ip by _count
```

Active Directory 2012+ - OpenTelemetry/Active Directory Service Failures/Audit Failures Over Time

```
sumo.datasource=windows  "Audit Failure"
| json "event_id", "computer", "keywords" as event_id, host.name, keywords nodrop
| json field=event_id "id" as event_id
| where jsonArrayContains(keywords, "Audit Failure") and host.name matches "{{host.name}}"
| timeslice 1h
| count as failures by _timeslice, event_id
| transpose row _timeslice column event_id
```

Active Directory 2012+ - OpenTelemetry/Active Directory Service Failures/Successes vs Failures

```
sumo.datasource=windows  keywords
| json  "computer", "keywords" as host.name, keywords nodrop
| if (jsonArrayContains(keywords, "Audit Failure"), "Failure", "Success") as status
| where host.name matches "{{host.name}}"
| timeslice 1h
| count as EventCount by  status, _timeslice
| transpose row _timeslice column status
```

Active Directory 2012+ (JSON)/Account Lock Out Events/Account Lock Out Events

```
_sourceCategory={{_sourceCategory}}  "\"Channel\":\"Security\"" "\"EventID\":\"4740\""
_sourceName=Security
| json "EventID", "Computer", "Message", "EventData.SubjectUserName",
"EventData.SubjectDomainName", "EventData.TargetUserName", "EventData.TargetDomainName",
"Channel", "keywords" as event_id, host, msg_summary, src_user, src_domain, dest_user,
dest_domain, channel, keywords nodrop
| parse regex field=msg_summary "(?<msg_summary>.*\.*)" nodrop
| where event_id = "4740"
| fields host, event_id, msg_summary, src_user, src_domain, dest_user, dest_domain
```

Active Directory 2012+ (JSON)/Active Directory Service Activity/Category Over Time

```
_sourceCategory={{_sourceCategory}}
| json "EventID", "Computer", "Message", "Task" as event_id, host, msg_summary, task
nodrop
| parse regex field=msg_summary "(?<msg_summary>.*\.*)" nodrop
| where host matches "{{host}}"
| timeslice 1h
| count by _timeslice, task
| transpose row _timeslice column task
```

Active Directory 2012+ (JSON)/Active Directory Service Activity/Object Creation

```
_sourceCategory={{_sourceCategory}}  "\"Channel\":\"Security\"" (4720 or 4741 or 4727 or
4731 or 4744 or 4749 or 4754 or 4759 or 4783 or 5137 or 4902 or 4790)
_sourceName=Security
| json "EventID", "Computer", "Message" as event_id, host, msg_summary nodrop
| if ((event_id in ("4720")), "user", "") as object_created
| if ((event_id in ("4720", "4741")), "computer", "") as object_created
| if ((event_id in ("4727", "4731", "4744", "4749", "4754", "4759", "4783")), "group",
object_created) as object_created
| if ((event_id in ("5137", "4902", "4790")), "object", object_created) as object_created
| where object_created != "" and host matches "{{host}}"
| timeslice 1h
| count by object_created, _timeslice
| transpose row _timeslice column object_created
| fillmissing timeslice(1h)
```

Active Directory 2012+ (JSON)/Active Directory Service Activity/Object Deletion

```
_sourceCategory={{_sourceCategory}}  "\"Channel\":\"Security\"" (4726 or 4743 or 4730 or
4734 or 4748 or 4753 or 4758 or 4763 or 4789 or 4659 or 4660 or 5141)
_sourceName=Security
| json "EventID", "Computer", "Message", "EventData.LogonType", "Channel" as event_id,
host, msg_summary, logon_type, channel nodrop
| if ((event_id in ("4726")), "user", "") as object_deleted
| if ((event_id in ("4743")), "computer", "") as object_deleted
| if ((event_id in ("4730", "4734", "4748", "4753", "4758", "4763", "4789")), "group",
object_deleted) as object_deleted
| if ((event_id in ("4659", "4660", "5141")), "object", object_deleted) as object_deleted
| where object_deleted != "" and host matches "{{host}}"
| timeslice 1h
| count by object_deleted, _timeslice
| transpose row _timeslice column object_deleted
| fillmissing timeslice(1h)
```

Active Directory 2012+ (JSON)/Active Directory Service Activity/off Activity

```
_sourceCategory={{_sourceCategory}}  "\"Channel\":\"Security\"" ("\"EventID\":\"4624\"" or
"\"EventID\":\"4634\"")  _sourceName=Security
| json "EventID", "Computer", "EventData.SubjectUserName",  "EventData.SubjectDomainName",
"EventData.TargetUserName", "EventData.TargetDomainName", "EventData.LogonType",
"EventData.LogonGuid", "EventData.SubjectLogonId", "EventData.TargetLogonId", "Channel",
"Message"  as event_id, host, src_user, src_domain, dest_user, dest_domain, logon_type,
LogonGuid, SubjectLogonId, TargetLogonId, channel, msg_summary nodrop
| where event_id in ("4624", "4634") and channel = "Security" and host matches "{{host}}"
| if (logon_type = "2", "interactive", if (logon_type = "3", "network", if (logon_type =
"4", "batch", if (logon_type = "5", "service", if (logon_type = "7", "unlock", if
(logon_type = "8", "networkcleartext", if (logon_type = "9", "newcredentials", if
(logon_type = "10", "remoteinteractive", if (logon_type = "11", "cachedinteractive", ""
))))))))) as type
| if (event_id = "4624", "logon", "logged off") as addonstr
| concat (type, " ", addonstr) as type
| where type != ""
| timeslice 1h
| count by _timeslice, type
| transpose row _timeslice column type
```

Active Directory 2012+ (JSON)/Active Directory Service Activity/Rights Management

```
_sourceCategory={{_sourceCategory}}  "\"Channel\":\"Security\"" ("\"EventID\":\"4704\"" or
"\"EventID\":\"4705\"")  _sourceName=Security
| json "EventID", "Computer", "Message", "Keywords", "Channel", "Level",
"EventData.SubjectUserName", "EventData.SubjectDomainName", "EventData.SubjectLogonId",
"EventData.PrivilegeList" as event_id, host, msg_summary, Keywords, channel, Level,
src_user, src_domain, src_logonid, PrivilegeList nodrop
| where event_id in ("4704", "4705") and host matches "{{host}}"
| if ((event_id = "4704"), "assigned", if ((event_id = "4705"), "removed", "")) as right
| timeslice 1h
| count by right, _timeslice
| transpose row _timeslice column right
| fillmissing timeslice(1h)
```

Active Directory 2012+ (JSON)/Active Directory Service Activity/Top 10 Messages

```
_sourceCategory={{_sourceCategory}}
| json "EventID", "Computer", "Message" as event_id, host, msg_summary nodrop
| parse regex field=msg_summary "(?<msg_summary>.*\.*)" nodrop
| where host matches "{{host}}"
| count by msg_summary
| top 10 msg_summary by _count
```

Active Directory 2012+ (JSON)/Active Directory Service Failures/Admin Activity by Category

```
_sourceCategory={{_sourceCategory}}  Task
| json "EventID", "Computer", "Level", "Message", "Keywords", "EventData.SubjectUserName",
"EventData.SubjectDomainName", "EventData.TargetUserName", "EventData.TargetDomainName",
"Channel", "Task" as event_id, host, event_type, msg_summary, Keywords, src_user,
src_domain, dest_user, dest_domain, channel, task nodrop
| parse regex field=msg_summary "(?<msg_summary>.*\.*)" nodrop
| where (src_user matches "Administrator*" or dest_user matches "Administrator*") and host
matches "{{host}}"
| count by task, Keywords, msg_summary
| topk(10, _count) by task | fields -_rank
```

Active Directory 2012+ (JSON)/Active Directory Service Failures/All Failures by IP

```
_sourceCategory={{_sourceCategory}}  ("fatal" or "failure" or "error" or "Audit Failure")
| json "EventID", "Computer", "Keywords", "EventData.IpAddress" as event_id, host,
Keywords, src_ip nodrop
| where Keywords != "Audit Success" and !isEmpty(src_ip) and src_ip != "-" and host
matches "{{host}}"
| count by src_ip
| top 10 src_ip by _count
```

Active Directory 2012+ (JSON)/Active Directory Service Failures/Audit Failures Over Time

```
_sourceCategory={{_sourceCategory}}  "\"Keywords\":\"Audit Failure\""
| json "EventID", "Computer", "Keywords" as event_id, host, keywords nodrop
| where keywords = "Audit Failure" and host matches "{{host}}"
| timeslice 1h
| count as failures by _timeslice, event_id
| transpose row _timeslice column event_id
```

Active Directory 2012+ (JSON)/Active Directory Service Failures/Successes vs Failures

```
_sourceCategory={{_sourceCategory}}  Keywords
| json "EventID", "Computer", "Keywords" as event_id, host, keywords nodrop
| if (keywords = "Audit Failure", "Failure", "Success") as status
| where host matches "{{host}}"
| timeslice 1h
| count as EventCount by  status, _timeslice
| transpose row _timeslice column status
```

Active Directory 2012+ (JSON)/Directory Service Object Changes/Directory Service Object Changes

```
_sourceCategory={{_sourceCategory}}  "\"Channel\":\"Security\"" (5136 or 5137 or 5138 or
5139 or 5141)  _sourceName=Security
| json "EventID", "Computer", "Message", "EventData.SubjectUserName",
"EventData.SubjectDomainName", "EventData.SubjectLogonId", "EventData.DSName",
"EventData.DSType", "EventData.ObjectDN", "EventData.ObjectClass",
"EventData.AttributeValue", "Task", "Keywords", "Channel", "Level" as event_id, host,
msg_summary, src_user, src_domain, src_LogonId, directory_service_name,
directory_service_type, object_dn, object_class, AttributeValue, task, Keywords, channel,
level nodrop
| where event_id in ("5136", "5137", "5138", "5139", "5141") and channel = "Security"
| parse regex field=msg_summary "(?<msg_summary>.*\.*)" nodrop
| count by event_id, host, msg_summary, src_user, src_domain, src_LogonId,
directory_service_name, directory_service_type, object_dn, object_class, AttributeValue,
task, Keywords, channel, level
```

Active Directory 2012+ (JSON)/Failed User Logins/Failed User Logins

```
_sourceCategory={{_sourceCategory}}  "\"Channel\":\"Security\"" (4771 or 4776 or 4768 or
4769 or 4625) "Audit Failure"  _sourceName=Security
| json "EventID", "Computer", "Message", "EventData.LogonType", "EventData.FailureReason",
"EventData.IpAddress", "EventData.IpPort", "EventData.SubjectUserName",
"EventData.SubjectDomainName", "EventData.TargetUserName", "EventData.TargetDomainName",
"EventData.WorkstationName", "Channel", "EventData.Status", "EventData.SubStatus",
"EventData.Workstation", "Keywords" as event_id, host, msg_summary, logon_type,
fail_reason, src_ip, src_port, src_user, src_domain, dest_user, dest_domain, src_host,
channel, status, sub_status, work_station, Keywords nodrop
| parse regex field=msg_summary "Failure Information:\s+Failure Reason:\s+(?
<failure_reason>[^.\r]+?)[.\r]" nodrop
| parse regex field=msg_summary "Result Code:\s+(?<result_code>[^\r]+)\r" nodrop
| parse regex field=msg_summary "Failure Code:\s+(?<failure_code>[^\r]+)\r" nodrop
| parse regex field=msg_summary "(?<msg_summary>.*\.*)" nodrop
| where event_id in ("4625", "4771") or (event_id = "4768" and result_code != "0x0") or
(event_id = "4769" and failure_code != "0x0") or (event_id = "4776" and status != "0x0")
| fields host, event_id, msg_summary, src_user, src_ip, src_port, src_domain, dest_user,
dest_domain, fail_reason, logon_type, result_code, failure_code, src_host, status,
sub_status
```

Active Directory 2012+ (JSON)/Last Successful Login for a specific user/Last Successful Login for a specific user

```
_sourceCategory={{_sourceCategory}}  "\"Channel\":\"Security\"" "\"EventID\":\"4624\""
myuser  _sourceName=Security
// provide username you want to search for in place of myuser
| json "EventID", "Computer", "Message", "Keywords", "EventData.SubjectUserName",
"EventData.SubjectDomainName", "EventData.TargetUserName", "EventData.TargetDomainName",
"EventData.IpAddress", "EventData.IpPort", "Channel", "Level", "EventData.LogonType" as
event_id, host, msg_summary, Keywords, src_user, src_domain, dest_user, dest_domain,
src_ip, src_port, channel, level, logon_type nodrop
| parse regex field=msg_summary "Client Address:\s+(?<src_ip>[^\r]+?)\r[\s\S]+?Client
Port:\s+?(?<src_port>[\d-]+)" nodrop
| parse regex field=msg_summary "Source Network Address:\s+(?<src_ip>[^\r]+?)\r[\s\S]+?
Source Port:\s+?(?<src_port>[\d-]+)" nodrop
| parse regex field=msg_summary "Client Name:\s+(?<src_host>[^\r]+?)\r[\s\S]+?Client
Address:[\s\r]+(?<src_ip>[^\r]+)" nodrop
| where event_id = "4624" and logon_type in ("2","7","8","10","11") and dest_user matches
"myuser" // provide username you want to search for in place of myuser
| first(_messagetime) as last_login by host, src_user, src_domain, src_ip, src_port,
dest_user, dest_domain, logon_type, src_host
| sort by last_login
| formatDate(fromMillis(last_login), "MM-dd-yyyy HH:mm:ss z") as last_login
```

Active Directory 2012+ (JSON)/Last Successful Login Report/Last Successful Login Report

```
_sourceCategory={{_sourceCategory}}  "\"Channel\":\"Security\"" "\"EventID\":\"4624\""
_sourceName=Security
| json "EventID", "Computer", "Message", "Keywords", "EventData.SubjectUserName",
"EventData.SubjectDomainName", "EventData.TargetUserName", "EventData.TargetDomainName",
"EventData.IpAddress", "EventData.IpPort", "Channel", "Level", "EventData.LogonType" as
event_id, host, msg_summary, Keywords, src_user, src_domain, dest_user, dest_domain,
src_ip, src_port, channel, level, logon_type nodrop
| parse regex field=msg_summary "Client Address:\s+(?<src_ip>[^\r]+?)\r[\s\S]+?Client
Port:\s+?(?<src_port>[\d-]+)" nodrop
| parse regex field=msg_summary "Source Network Address:\s+(?<src_ip>[^\r]+?)\r[\s\S]+?
Source Port:\s+?(?<src_port>[\d-]+)" nodrop
| parse regex field=msg_summary "Client Name:\s+(?<src_host>[^\r]+?)\r[\s\S]+?Client
Address:[\s\r]+(?<src_ip>[^\r]+)" nodrop
| where event_id = "4624" and logon_type in ("2", "7", "8", "10", "11")
| first(_messagetime) as last_login by src_ip, dest_user, dest_domain, logon_type,
src_port, src_domain, src_user, src_host, host
| sort by last_login
| formatDate(fromMillis(last_login), "MM-dd-yyyy HH:mm:ss z") as last_login
```

Active Directory 2012+ (JSON)/Login attempts to disabled accounts/Login attempts to disabled accounts

```
_sourceCategory={{_sourceCategory}}  "\"Channel\":\"Security\"" "Audit Failure"
("\"EventID\":\"4625\"" or "\"EventID\":\"4768\"" or "\"EventID\":\"4771\"" or
"\"EventID\":\"4776\"" or "\"EventID\":\"4769\"") ("currently disabled" or 0xc0000072)
_sourceName=Security
| json "EventID", "Computer", "Message", "EventData.LogonType", "EventData.FailureReason",
"EventData.IpAddress", "EventData.IpPort", "EventData.SubjectUserName",
"EventData.SubjectDomainName", "EventData.TargetUserName", "EventData.TargetDomainName",
"EventData.WorkstationName", "Channel", "EventData.Status", "EventData.SubStatus",
"EventData.Workstation", "Keywords" as event_id, host, msg_summary, logon_type,
fail_reason, src_ip, src_port, src_user, src_domain, dest_user, dest_domain, src_host,
channel, status, sub_status, work_station, Keywords nodrop
| parse regex field=msg_summary "Failure Information:\s+Failure Reason:\s+(?
<failure_reason>[^.\r]+?)[.\r]" nodrop
| parse regex field=msg_summary "Failure Information:\s+(?<test>.*)" nodrop
| parse regex field=msg_summary "Result Code:\s+(?<result_code>[^\r]+)\r" nodrop
| parse regex field=msg_summary "Failure Code:\s+(?<failure_code>[^\r]+)\r" nodrop
| parse regex field=msg_summary "(?<msg_summary>.*\.*)" nodrop
| where event_id in ("4625", "4771") or (event_id = "4768" and result_code != "0x0") or
(event_id = "4769" and failure_code != "0x0") or (event_id = "4776" and status != "0x0")
and (failure_reason = "Account currently disabled" or sub_status = "0xc0000072")
| fields host, event_id, msg_summary, src_user, src_ip, src_port, src_domain, dest_user,
dest_domain, fail_reason, logon_type, result_code, failure_code, src_host, failure_reason,
status, sub_status
```

Active Directory 2012+ (JSON)/New Account Creation/New Account Creation

```
_sourceCategory={{_sourceCategory}}  "\"Channel\":\"Security\"" "\"EventID\":\"4720\""
_sourceName=Security
| json "EventID", "Computer", "Message", "EventData.SubjectUserName",
"EventData.SubjectDomainName", "EventData.TargetUserName", "EventData.TargetDomainName",
"Channel" as event_id, host, msg_summary, src_user, src_domain, dest_user, dest_domain,
channel nodrop
| parse regex field=msg_summary "(?<msg_summary>.*\.*)" nodrop
| where event_id = "4720" and channel = "Security"
| fields host, event_id, msg_summary, src_user, dest_user, src_domain, dest_domain
```

Active Directory 2012+ (JSON)/New Computer Account Creation/New Computer Account Creation

```
_sourceCategory={{_sourceCategory}}  "\"Channel\":\"Security\"" "\"EventID\":\"4741\""
"computer account was created"  _sourceName=Security
| json "EventID", "Computer", "Message", "EventData.SubjectUserName",
"EventData.SubjectDomainName", "EventData.TargetUserName", "EventData.TargetDomainName",
"EventData.SamAccountName", "EventData.HomeDirectory", "Channel", "Task", "Keywords" as
event_id, host, msg_summary, src_user, src_domain, dest_user, dest_domain, SamAccountName,
HomeDirectory, channel, task, Keywords nodrop
| parse regex field=msg_summary "(?<msg_summary>.*\.*)" nodrop
| where event_id = "4741"
| fields event_id, host, msg_summary, src_user, src_domain, dest_user, dest_domain,
SamAccountName, HomeDirectory, channel, task, Keywords
```

Active Directory 2012+ (JSON)/New Group Creation/New Group Creation

```
_sourceCategory={{_sourceCategory}}  "\"Channel\":\"Security\"" ("\"EventID\":\"4727\"" or
"\"EventID\":\"4731\"" or "\"EventID\":\"4754\"")  _sourceName=Security
| json "EventID", "Computer", "Message", "EventData.SubjectUserName",
"EventData.SubjectDomainName", "EventData.TargetUserName", "EventData.TargetDomainName",
"Channel" as event_id, host, msg_summary, src_user, src_domain, dest_user, dest_domain,
channel nodrop
| parse regex field=msg_summary "(?<msg_summary>.*\.*)" nodrop
| where event_id in ("4727", "4731", "4754")
| dest_user as group_name | dest_domain as group_domain
| fields host, event_id, msg_summary, src_user, src_domain, group_name, group_domain
```

Active Directory 2012+ (JSON)/OU Creation/OU Creation

```
_sourceCategory={{_sourceCategory}}  "\"Channel\":\"Security\"" "\"EventID\":\"5137\""
_sourceName=Security
| json "EventID", "Computer", "Message", "EventData.SubjectUserName",
"EventData.SubjectDomainName", "EventData.SubjectLogonId", "EventData.DSName",
"EventData.DSType", "EventData.ObjectDN", "EventData.ObjectClass", "Task", "Keywords",
"Channel", "Level" as event_id, host, msg_summary, src_user, src_domain, src_LogonId,
directory_service_name, directory_service_type, object_dn, object_class, task, Keywords,
channel, level nodrop
| where event_id = "5137" and channel = "Security"
| parse regex field=msg_summary "(?<msg_summary>.*\.*)" nodrop
| count by event_id, host, msg_summary, src_user, src_domain, src_LogonId,
directory_service_name, directory_service_type, object_dn, object_class, task, Keywords,
channel, level
```

Active Directory 2012+ (JSON)/OU Deletion/OU Deletion

```
_sourceCategory={{_sourceCategory}}  "\"Channel\":\"Security\"" "\"EventID\":\"5141\""
_sourceName=Security
| json "EventID", "Computer", "Message", "EventData.SubjectUserName",
"EventData.SubjectDomainName", "EventData.SubjectLogonId", "EventData.DSName",
"EventData.DSType", "EventData.ObjectDN", "EventData.ObjectClass", "Task", "Keywords",
"Channel", "Level" as event_id, host, msg_summary, src_user, src_domain, src_LogonId,
directory_service_name, directory_service_type, object_dn, object_class, task, Keywords,
channel, level nodrop
| where event_id = "5141" and channel = "Security"
| parse regex field=msg_summary "(?<msg_summary>.*\.*)" nodrop
| fields event_id, host, msg_summary, src_user, src_domain, src_LogonId,
directory_service_name, directory_service_type, object_dn, object_class, task, Keywords,
channel, level
```

Active Directory 2012+ (JSON)/Password Change Attempts/Password Change Attempts

```
_sourceCategory={{_sourceCategory}}  "\"Channel\":\"Security\"" "\"EventID\":\"4723\""
_sourceName=Security
| json "EventID", "Computer", "Message", "EventData.SubjectUserName",
"EventData.SubjectDomainName", "EventData.TargetUserName", "EventData.TargetDomainName",
"Channel" as event_id, host, msg_summary, src_user, src_domain, dest_user, dest_domain,
channel nodrop
| parse regex field=msg_summary "(?<msg_summary>.*\.*)" nodrop
| where event_id = "4723"
| fields host, event_id, msg_summary, src_user, dest_user, src_domain, dest_domain
```

Active Directory 2012+ (JSON)/Password Reset Attempts/Password Reset Attempts

```
_sourceCategory={{_sourceCategory}}  "\"Channel\":\"Security\"" "\"EventID\":\"4724\""
_sourceName=Security
| json "EventID", "Computer", "Message", "EventData.SubjectUserName",
"EventData.SubjectDomainName", "EventData.TargetUserName", "EventData.TargetDomainName",
"Channel" as event_id, host, msg_summary, src_user, src_domain, dest_user, dest_domain,
channel nodrop
| parse regex field=msg_summary "(?<msg_summary>.*\.*)" nodrop
| where event_id = "4724"
| fields host, event_id, msg_summary, src_user, src_domain, dest_user, dest_domain
```

Active Directory 2012+ (JSON)/Policy Changes/Policy Changes

```
_sourceCategory={{_sourceCategory}}  "\"Channel\":\"Security\"" (4902 or 4904 or 4905 or
4906 or 4907 or 4912 or 4715 or 4719 or 4739 or "Audit Policy Change" or "System audit
policy was changed" or *policy*change* or "Policy Change")  _sourceName=Security
| json "EventID", "Computer", "Message", "EventData.SubjectUserName",
"EventData.SubjectDomainName", "EventData.TargetUserName", "EventData.TargetDomainName",
"Channel", "Task" as event_id, host, msg_summary, src_user, src_domain, dest_user,
dest_domain, channel, task nodrop
| where event_id in ("4902", "4904", "4905", "4906", "4907", "4912", "4715", "4719",
"4739") or msg_summary matches "System audit policy was changed" or msg_summary matches
"*policy*change*"
| parse regex field=msg_summary "(?<msg_summary>.*\.*)" nodrop
| count as num_changes, first(_messageTime) as last_change by src_domain, host, task,
msg_summary
| sort by last_change
| formatDate(fromMillis(last_change), "MM-dd-yyyy HH:mm:ss z") as last_change
```

Active Directory 2012+ (JSON)/Successful User Logins/Successful User Logins

```
_sourceCategory={{_sourceCategory}}  "\"Channel\":\"Security\"" "\"EventID\":\"4624\""
_sourceName=Security
| json "EventID", "Computer", "Message", "Keywords", "EventData.SubjectUserName",
"EventData.SubjectDomainName", "EventData.TargetUserName", "EventData.TargetDomainName",
"EventData.IpAddress", "EventData.IpPort", "Channel", "Level", "EventData.LogonType" as
event_id, host, msg_summary, Keywords, src_user, src_domain, dest_user, dest_domain,
src_ip, src_port, channel, level, logon_type nodrop
| parse regex field=msg_summary "Client Address:\s+(?<src_ip>[^\r]+?)\r[\s\S]+?Client
Port:\s+?(?<src_port>[\d-]+)" nodrop
| parse regex field=msg_summary "Source Network Address:\s+(?<src_ip>[^\r]+?)\r[\s\S]+?
Source Port:\s+?(?<src_port>[\d-]+)" nodrop
| parse regex field=msg_summary "Client Name:\s+(?<src_host>[^\r]+?)\r[\s\S]+?Client
Address:[\s\r]+(?<src_ip>[^\r]+)" nodrop
| parse regex field=msg_summary "(?<msg_summary>.*\.*)" nodrop
| where event_id = "4624"
| fields event_id, msg_summary, src_host, src_ip, src_port, src_user, src_domain,
dest_user, dest_domain, logon_type
```

Active Directory 2012+ (JSON)/Top 10 Domain Controllers with Most Login Failures/Top 10 Domain Controllers with Most Login Failures

```
_sourceCategory={{_sourceCategory}}  "\"Channel\":\"Security\"" (4771 or 4776 or 4768 or
4769 or 4625) "Audit Failure"  _sourceName=Security
| json "EventID", "Computer", "Message", "EventData.LogonType", "EventData.FailureReason",
"EventData.IpAddress", "EventData.IpPort", "EventData.SubjectUserName",
"EventData.SubjectDomainName", "EventData.TargetUserName", "EventData.TargetDomainName",
"EventData.WorkstationName", "Channel", "EventData.Status", "EventData.SubStatus",
"EventData.Workstation", "Keywords" as event_id, host, msg_summary, logon_type,
fail_reason, src_ip, src_port, src_user, src_domain, dest_user, dest_domain, src_host,
channel, status, sub_status, work_station, Keywords nodrop
| parse regex field=msg_summary "Failure Information:\s+Failure Reason:\s+(?
<failure_reason>[^.\r]+?)[.\r]" nodrop
| parse regex field=msg_summary "Result Code:\s+(?<result_code>[^\r]+)\r" nodrop
| parse regex field=msg_summary "Failure Code:\s+(?<failure_code>[^\r]+)\r" nodrop
| parse regex field=msg_summary "Client Address:\s+(?<src_ip>[^\r]+?)\r[\s\S]+?Client
Port:\s+?(?<src_port>[\d-]+)" nodrop
| parse regex field=msg_summary "Source Network Address:\s+(?<src_ip>[^\r]+?)\r[\s\S]+?
Source Port:\s+?(?<src_port>[\d-]+)" nodrop
| parse regex field=msg_summary "Client Name:\s+(?<src_host>[^\r]+?)\r[\s\S]+?Client
Address:[\s\r]+(?<src_ip>[^\r]+)" nodrop
| where event_id in ("4625", "4771") or (event_id = "4768" and result_code != "0x0") or
(event_id = "4769" and failure_code != "0x0") or (event_id = "4776" and status != "0x0")
| fields host, event_id, msg_summary, src_user, src_ip, src_port, src_domain, dest_user,
dest_domain, fail_reason, logon_type, result_code, failure_code, src_host, status,
sub_status
| _sourceHost as ReportingDC | count as %"LoginFailureCount" by ReportingDC | top 10
ReportingDC by %"LoginFailureCount"
```