SPL_Query	XQL_Query
| tstats fillnull_value=N/A summariesonly=t values(Authentication.user_bunit) as user_bunit values (Authentication.signature) as signature values(Authentication.signature_id) as event_code values (Authentication.src) as src values(Authentication.dest) as dest values(Authentication.Logon_Type) as logon_type count from datamodel=Authentication where Authentication.action IN (failure) NOT sourcetype IN ("amp:*") Authentication.app IN (win:remote win:local) NOT Authentication.src IN (unknown) NOT Authentication.user IN ("*$") by index sourcetype Authentication.action Authentication.app Authentication.src_user Authentication.user \| `drop_dm_object_name("Authentication")` \| lookup access_amp_domain_excessive_failed_logins_ntt_exclusions user OUTPUT user as excluded_user \| lookup access_amp_domain_excessive_failed_logins_ntt_exclusions src_user OUTPUT src_user as excluded_src_user \| search NOT excluded_user IN (*) \| search NOT excluded_src_user IN (*) \| fields - excluded_user excluded_src_user \| where 'count'>=6 \| eval urgency="high" \|search src_user!=*$ AND src_user!=@@* AND src_user!="Account Domain:" ```| eval src_user = mvfilter(src_user!="Account Domain:") | eval user = mvfilter(user!="account domain:") | eval user=coalesce(user, src_user) | fields - src_user``` Drill Down Query index=* sourcetype="wineventlog" user IN ($user$) result=lockout | rename src_nt_host as src | table _time dest src user user_nick user_bunit EventCode name result | sort - _time	datamodel dataset = microsoft_windows_raw //| filter microsoft_windows_raw._collector_type ="XDR Collector". /Removed as per discussion on 23-Aug with hiep | filter xdm.event.id = "4625" | filter xdm.event.original_event_type = "Account Lockout" | fields _time as time_created, xdm.event.id as event_id, xdm.source.user.username as user_name, xdm.source.host.hostname as host_name, xdm.source.user.domain as domain, xdm.event.original_event_type as event_type, xdm.event.description as message | comp count(host_name) as locked_host, min(time_created) as Start_time, max(time_created) as End_time by event_id, event_type, host_name, user_name, domain | alter time_diff = timestamp_diff(End_time, Start_time,"MINUTE") | filter locked_host > 5 // Checking Failed Host Count is greater than 5 | filter time_diff < 20 // Checking time difference in less than 20 min | fields event_id, event_type, host_name, user_name, domain, locked_host, Start_time, End_time, time_diff | join (dataset = access_amp_domain_excessive_failed_logins_ntt_exclusions | fields *) as output output.user != user_name | dedup host_name Drill Down Query datamodel dataset = microsoft_windows_raw | filter xdm.source.user.username in ($user_name) | filter xdm.event.id = "4625" | filter xdm.event.original_event_type = "Account Lockout" //| rename src_nt_host as src //src_nt_host not found | fields _time , xdm.target.host.hostname as dest, xdm.source.host.hostname as src, xdm.source.user.username as user, xdm.event.id as EventCode, xdm.event.original_event_type as result // fields not found //name user_nick user_bunit | sort desc _time

| tstats fillnull_value=N/A summariesonly=t values(Authentication.user_bunit) as user_bunit values(Authentication.signature) as signature values(Authentication.signature_id) as event_code values(Authentication.src) as src values(Authentication.dest) as dest values(Authentication.Logon_Type) as logon_type count from datamodel=Authentication where Authentication.action IN (failure) NOT sourcetype IN ("amp:*") Authentication.app IN (win:remote win:local) NOT Authentication.src IN (unknown) NOT Authentication.user IN ("*$") by index sourcetype Authentication.action Authentication.app Authentication.src_user Authentication.user
| `drop_dm_object_name("Authentication")`
| lookup access_amp_domain_excessive_failed_logins_ntt_exclusions user OUTPUT user as excluded_user
| lookup access_amp_domain_excessive_failed_logins_ntt_exclusions src_user OUTPUT src_user as excluded_src_user
| search NOT excluded_user IN (*)
| search NOT excluded_src_user IN (*)
| fields - excluded_user excluded_src_user
| where 'count'>=6
| eval urgency="high"
|search src_user!=*$ AND src_user!=@@* AND src_user!="Account Domain:"
```| eval src_user = mvfilter(src_user!="Account Domain:")
| eval user = mvfilter(user!="account domain:")
| eval user=coalesce(user, src_user)
| fields - src_user```

Drill Down Query

config case_sensitive = false
| datamodel dataset = microsoft_windows_raw
| filter xdm.event.id = "4625"   // window event id for failed logons
| filter xdm.target.user.username not in ("*$")
| filter xdm.source.user.username not in ("*$", "@@*", "Account Domain:*")
| alter collector_type = microsoft_windows_raw._collector_type
//| filter collector_type = "XDR Collector"
//| filter xdm.target.user.username contains "tapin_centre" or xdm.target.user.username contains "administrator" or xdm.target.user.username contains "prince"

//| filter xdm.source.ipv4 not in ("unknown", "", null) // does unknown here refers to null values
| fields xdm.target.user.username , xdm.source.user.username , xdm.observer.action , xdm.event.outcome , xdm.event.outcome_reason , xdm.alert.description , xdm.event.description , xdm.event.id , xdm.source.host.hostname , xdm.target.host.hostname , xdm.logon.type , *

// filtering out the excluded source and target users from failed login attempts
| join type = left (
    dataset = access_amp_domain_excessive_failed_logins_ntt_exclusions
    | alter user = replace(user, "*", "")
    | fields user as excluded_user, src_user as excluded_src_user
) as excluded_users xdm.target.user.username contains excluded_users.excluded_user or xdm.source.user.username contains excluded_users.excluded_user
| filter excluded_user in ("", null)
//| filter excluded_src_user not contains "~$"
| fields xdm.source.user.username ,xdm.target.user.username , excluded_user , excluded_src_user , xdm.event.id , xdm.source.host.hostname , xdm.target.host.hostname , xdm.logon.type , xdm.event.outcome , *

| comp count() as total_events, min(_time) as firstEventTime, max(_time) as lastEventTime, values(xdm.source.host.hostname) as host, values(xdm.source.ipv4) as src_ip, values(xdm.logon.type) as logon_type, values(xdm.event.original_event_type) as event_type by xdm.source.user.username, xdm.target.user.username, excluded_src_user , excluded_user

| filter total_events >= 6

****
Old Query
config case_sensitive = false
| datamodel dataset = microsoft_windows_raw
| filter xdm.event.id = "4625"   // window event id for failed logons
| filter xdm.target.user.username not in ("*$")
| filter xdm.source.user.username not in ("*$", "@@*", "Account Domain:*")

//| filter microsoft_windows_raw._collector_type = "XDR Collector"

| filter xdm.source.ipv4 not in ("unknown", "", null) and xdm.target.user.username != null // does unknown here refers to null values

// filtering out the excluded source and target users from failed login attempts
| join type = left (
    dataset = access_amp_domain_excessive_failed_logins_ntt_exclusions
    | alter user = replace(user, "*", "")
    | fields user as excluded_user
) as excluded_users xdm.target.user.username contains excluded_users.excluded_user or xdm.source.user.username contains excluded_users.excluded_user
| filter excluded_user in ("", null)

| comp count() as total_events, min(_time) as firstEventTime, max(_time) as lastEventTime, values(xdm.source.host.hostname) as host, values(xdm.event.outcome) as action, values(xdm.event.id) as event_id,values(xdm.source.ipv4) as src_ip, values(xdm.logon.type) as logon_type, values(xdm.event.original_event_type) as event_type by xdm.source.user.username, xdm.target.user.username
| alter urgency = "High"
| filter total_events > 6
| alter host = arraystring(host, ",")

Drill Down

config case_sensitive = false
| datamodel dataset = microsoft_windows_raw
```

```
| tstats fillnull_value=N/A summariesonly=t allow_old_summaries=t
values(Authentication.src_user) as src_user, values(Authentication.user_bunit) as user_bunit, values
(Authentication.signature) as signature,
values(Authentication.signature_id) as event_code, values(Authentication.src) as src, values
(Authentication.dest) as dest, count as hit_count from datamodel=Authentication
where Authentication.action IN (failure), sourcetype IN ("amp:north:audit:csv"),
Authentication.app IN ("north"), NOT Authentication.src IN (unknown), NOT Authentication.user IN
("*$")
by index, sourcetype, Authentication.action, Authentication.app, Authentication.user
| `drop_dm_object_name("Authentication")`
| rename index as src_idx, sourcetype as src_st
| lookup access_amp_siem_excessive_failed_logins_exclusions app, dest, signature, user OUTPUT
user as excluded_user
| lookup access_amp_siem_excessive_failed_logins_exclusions app, dest, signature, user_bunit
OUTPUT user_bunit as excluded_bunit
| lookup access_amp_siem_excessive_failed_logins_exclusions app, dest, signature, src_user
OUTPUT src_user as excluded_src_user
| lookup access_amp_siem_excessive_failed_logins_exclusions app, dest, signature, src OUTPUT
src as excluded_src_cidr
| where isNull(excluded_user) AND isNull(excluded_src_user) AND isNull(excluded_src_cidr) AND
isNull(excluded_bunit)
| eval threshold_val = `amp_siem_standard_Authentication_thresholds`
| where hit_count >= threshold_val
| fields - excluded_user, excluded_src_user, excluded_src_cidr, excluded_bunit, threshold_val
| eval urgency="high"

Drill Down

[{"name":"View failures by user $user$ for the application $app$","search":"index=*
sourcetype=$src_st$ tag=authentication action=failure src IN ($src$) dest IN ($dest$) app=\"$app$\"
user IN (\"$user$\")\n| fillnull value=N/A\n| table _time action user user_bunit user_agent src_user app
src dest signature signature_id reason\n| sort + _time","earliest_offset":"$info_min_time$","
latest_offset":"$info_max_time$"}]
```

```
// Title: Access - [AMP NORTH] Excessive Failed Logins [Orro] - Rule
// Description: Detect failed logins from a SIEM log source. Aggregate by user.
// Author: Sahil Sharma, ssharma7@paloaltonetworks.com
// Datasets: amp_north_raw
// Date: 30/July/2024

dataset = amp_north_raw
| filter action = "failure"
| filter app = "north"
| filter user !~= "\$$"
| filter src not in ("", null, "unknown")

| join type = left (
    dataset = access_amp_siem_excessive_failed_logins_exclusions_csv
    | fields user as excluded_user, src_user as excluded_src_user, src as excluded_cidr
) as exclusion_data incidr(src, exclusion_data.excluded_cidr) = true and exclusion_data.excluded_user = user

| filter excluded_cidr = null and excluded_user = null

| comp count() as hit_count, max(_time) as lastEventTime, min(_time) as firstEventTime, values(_raw_log) as raw_log,
values(src) as src, values(result) as result, values(UserAgent) as UserAgent, values(UserAuditId) as userAuditId, values
(dest) as dest, values(signature) as signature, values(audittype) as autdittype by action, app, user

| alter dest = arraystring(dest , ","),
       src = arraystring(src , ",")
| alter threshold_val = if(app = "U2", 8, app = "peoplesoft-financials", 10, app = "north", 30, 6),
       urgency = "high"
| filter hit_count >= threshold_val

Drill down

dataset = amp_north_raw
| alter name = format_string("View failures by user $user$ for the application %s", $app)
| filter action = "failure"
| filter $src contains src and $dest contains dest
| filter app = $app and user in ($user)
| fields _time, action, user, UserAgent, src_user, app, src, dest, signature, reason, AuditType, UserAuditId, name
| sort asc _time
```

```
| tstats fillnull_value=N/A summariesonly=t allow_old_summaries=t
values(Authentication.src_user) as src_user, values(Authentication.user_bunit) as user_bunit, values
(Authentication.signature) as signature,
values(Authentication.signature_id) as event_code, values(Authentication.user) as user, values
(Authentication.dest) as dest, count as hit_count from datamodel=Authentication
where Authentication.action IN (failure), sourcetype IN ("amp:north:audit:csv"),
Authentication.app IN ("north"), NOT Authentication.src IN (unknown), NOT Authentication.user IN
("*$")
by index, sourcetype, Authentication.action, Authentication.app, Authentication.src
| `drop_dm_object_name("Authentication")`
| rename index as src_idx, sourcetype as src_st
| lookup access_amp_siem_excessive_failed_logins_exclusions app, dest, signature, user OUTPUT
user as excluded_user
| lookup access_amp_siem_excessive_failed_logins_exclusions app, dest, signature, user_bunit
OUTPUT user_bunit as excluded_bunit
| lookup access_amp_siem_excessive_failed_logins_exclusions app, dest, signature, src_user
OUTPUT src_user as excluded_src_user
| lookup access_amp_siem_excessive_failed_logins_exclusions app, dest, signature, src OUTPUT
src as excluded_src_cidr
| where isNull(excluded_user) AND isNull(excluded_src_user) AND isNull(excluded_src_cidr) AND
isNull(excluded_bunit)
| eval threshold_val = `amp_siem_standard_Authentication_thresholds`
| where hit_count >= ( threshold_val * 3 )
| fields - excluded_user, excluded_src_user, excluded_src_cidr, excluded_bunit, threshold_val
| eval urgency="high"

Drill Down Query

[{"name":"View failures by host $src$ for the application $app$","search":"index=*
sourcetype=$src_st$ tag=authentication action=failure src IN ($src$) dest IN ($dest$) app=\"$app$\"
user IN (\"$user$\")\n| fillnull value=N/A \n| table _time action user user_bunit user_agent src_user
app src dest signature signature_id reason\n| sort + _time","earliest_offset":"$info_min_time$","
latest_offset":"$info_max_time$"}]
```

```
// Title: Access - [AMP NORTH] Excessive Failed Logins from a single IP [Orro] - Rule
// Description: Detect failed logins from a SIEM log source. Aggregate by source IP (src).
// Author: Sahil Sharma, ssharma7@paloaltonetworks.com
// Datasets: amp_north_raw
// Date: 30/July/2024

dataset = amp_north_raw
| filter action = "failure"
| filter app = "north"
| filter user !~= "\$$"
| filter src not in ("", null, "unknown")

| join type = left (
    dataset = access_amp_siem_excessive_failed_logins_exclusions_csv
    | fields user as excluded_user, src_user as excluded_src_user, src as excluded_cidr
) as exclusion_data incidr(src, exclusion_data.excluded_cidr) = true and exclusion_data.excluded_user = user

| filter excluded_cidr = null and excluded_user = null

| comp count() as hit_count, max(_time) as lastEventTime, min(_time) as firstEventTime, values(_raw_log) as raw_log,
values(user) as user, values(result) as result, values(UserAgent) as UserAgent, values(UserAuditId) as userAuditId,
values(dest) as dest, values(signature) as signature, values(audittype) as autdittype by action, app, src

| alter dest = arraystring(dest , ","),
       user = arraystring(user , ","),
       threshold_val = if(app = "U2", 8, app = "peoplesoft-financials", 10, app = "north", 30, 6),
       urgency = "high"

| filter hit_count >= multiply(threshold_val, 3)

Drill Down Query
dataset = amp_north_raw
| alter name = format_string("View failures by host %s for the application %s", $src, $app)
| filter action = "failure"
| filter src in ($src)
| filter $dest contains dest
| filter app = $app
| filter $user contains user
| fields _time, action, src_user, UserAgent, user, app, src, dest, signature, AuditType, UserAuditId, name, reason
| sort asc _time
```

| tstats fillnull_value=N/A summariesonly=t allow_old_summaries=t values(Authentication.src_user) as src_user, values(Authentication.user_bunit) as user_bunit, values(Authentication.signature) as signature, values(Authentication.signature_id) as event_code, values(Authentication.user) as user, values(Authentication.dest) as dest, dc(Authentication.user) as hit_count from datamodel=Authentication where sourcetype IN ("amp:north:audit:csv"), Authentication.app IN ("north"), NOT Authentication.src IN (unknown), NOT Authentication.user IN ("*$") by index, sourcetype, Authentication.action, Authentication.app, Authentication.src | `drop_dm_object_name("Authentication")` | rename index as src_idx, sourcetype as src_st | lookup access_amp_siem_excessive_accounts_exclusions app, dest, signature, user OUTPUT user as excluded_user | lookup access_amp_siem_excessive_accounts_exclusions app, dest, signature, user_bunit OUTPUT user_bunit as excluded_bunit | lookup access_amp_siem_excessive_accounts_exclusions app, dest, signature, src_user OUTPUT src_user as excluded_src_user | lookup access_amp_siem_excessive_accounts_exclusions app, dest, signature, src OUTPUT src as excluded_src_cidr | where isNull(excluded_user) AND isNull(excluded_src_user) AND isNull(excluded_src_cidr) AND isNull(excluded_bunit) | eval threshold_val = `amp_siem_standard_Authentication_thresholds` | where hit_count >= threshold_val | fields - excluded_user, excluded_src_user, excluded_src_cidr, excluded_bunit, threshold_val | eval urgency="high" Drill Down [{"name":"View attempts by host src for the application app","search":"index=* sourcetype=src_st tag=authentication src IN (src) dest IN ($dest$) app=\"app\" user IN (\"$user$\")\n| fillnull value=N/A \n| table _time action user user_bunit user_agent src_user app src dest signature signature_id reason\n| sort + _time","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	// Title: Access - [AMP NORTH] Excessive Number of Accounts from a single IP [Orro] - Rule // Description: Detect excessive accounts from a single IP from a SIEM log source. Aggregate by source IP (src) with high user counts. // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: amp_north_raw // Date: 30/July/2024 dataset = amp_north_raw | filter app = "north" | filter src !~= "\$$" | filter user not in ("", null, "unknown") | join type = left (dataset = access_amp_siem_excessive_accounts_exclusions_csv | alter signature = replace(signature, "*", "") | fields user as excluded_user, src_user as excluded_src_user, src as excluded_cidr, app as excluded_app, signature as excluded_signature) as exclusion_data incidr(src, exclusion_data.excluded_cidr) = true and exclusion_data.excluded_user = user and exclusion_data.excluded_app = app and signature contains exclusion_data.excluded_signature | filter excluded_cidr = null and excluded_user = null and excluded_app = null and excluded_signature = null | comp count() as hit_count, max(_time) as lastEventTime, min(_time) as firstEventTime, values(_raw_log) as raw_log, valueS(user) as user, values(result) as result, values(UserAgent) as UserAgent, values(UserAuditId) as userAuditId, values(dest) as dest, values(signature) as signature, values(AuditType) as AuditType by action, app, src | alter user = arraystring(user, ","), dest = arraystring(dest, ","), threshold_val = if(app = "U2", 8, app = "peoplesoft-financials", 10, app = "north", 30, 6), urgency = "high" | filter hit_count >= threshold_val Drill Down dataset = amp_north_raw | alter name = format_string("View attempts by host %s for the application %s", $src, $app) \| filter src in ($src) and $dest contains dest | filter app = $app and $user contains user | fields _time, action, user, UserAgent, src_user, app, src, dest, signature, reason, name, AuditType, UserAuditId | sort asc _time
index=amp_peoplesoft_prod user="PSBATCH" user_agent!="-" | transaction SRID maxspan=30m | table _time index sourcetype action app user src_user user_agent src dest reason Drill Down [{"name":"View actions by user $user$ for the application app","search":"index=* sourcetype=$sourcetype$ src IN (src) dest IN ($dest$) app=\"app\" user IN (\"$user$\") reason IN (\"$reason$\")\n| fillnull value=N/A\n| table _time index sourcetype action app user src_user user_agent src dest reason\n| sort + _time","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	datamodel dataset = xdr_data /* index=amp_peoplesoft_prod user="PSBATCH" user_agent!="-" | transaction SRID maxspan=30m | table _time index sourcetype action app user src_user user_agent src dest reason */ // Sample log file name "Peoplesoft_Financials_SIEM_Integration" | filter xdm.source.user.username = "PSBATCH" and xdm.source.user_agent != "-" | fields _time , xdm.observer.type , xdm.event.outcome , xdm.event.outcome_reason , xdm.source.application.name , xdm.source.user.username , xdm.target.user.username , xdm.source.user_agent , xdm.source.ipv4 ,xdm.target.ipv4 New search added in XSIAM on 22-Aug dataset = peoplesoft_financials_raw | filter user="PSBATCH" | filter user_agent!="-" | fields _time,action, app,user,src_user,user_agent,src,dest,reason

| tstats fillnull_value=N/A summariesonly=t allow_old_summaries=t
values(Authentication.src_user) as src_user, values(Authentication.user_bunit) as user_bunit, values
(Authentication.signature) as signature,
values(Authentication.signature_id) as event_code, values(Authentication.src) as src, values
(Authentication.dest) as dest, count as hit_count from datamodel=Authentication
where Authentication.action IN (failure), sourcetype IN
`amp_siem_standard_Authentication_sourcetypes`,
Authentication.app IN `amp_siem_standard_Authentication_apps`, NOT Authentication.src IN
(unknown), NOT Authentication.user IN ("*$")
by index, sourcetype, Authentication.action, Authentication.app, Authentication.user
| `drop_dm_object_name("Authentication")`
| rename index as src_idx, sourcetype as src_st
| lookup access_amp_siem_excessive_failed_logins_exclusions app, dest, signature, user OUTPUT
user as excluded_user
| lookup access_amp_siem_excessive_failed_logins_exclusions app, dest, signature, user_bunit
OUTPUT user_bunit as excluded_bunit
| lookup access_amp_siem_excessive_failed_logins_exclusions app, dest, signature, src_user
OUTPUT src_user as excluded_src_user
| lookup access_amp_siem_excessive_failed_logins_exclusions app, dest, signature, src OUTPUT
src as excluded_src_cidr
| where isNull(excluded_user) AND isNull(excluded_src_user) AND isNull(excluded_src_cidr) AND
isNull(excluded_bunit)
| eval threshold_val = `amp_siem_standard_Authentication_thresholds`
| where hit_count >= threshold_val
| fields - excluded_user, excluded_src_user, excluded_src_cidr, excluded_bunit, threshold_val
| eval urgency="high"

Drill Down Query:
[{"name":"View failures by user $user$ for the application app","search":"index=*
sourcetype=src_st tag=authentication action=failure src IN (src) dest IN ($dest$) app=\"app\"
user IN (\"$user$\")\n| fillnull value=N/A\n| table _time action user user_bunit user_agent src_user app
src dest signature signature_id reason\n| sort + _time","earliest_offset":"$info_min_time$","
latest_offset":"$info_max_time$"}]

config case_sensitive = false
| datamodel dataset in (peoplesoft_financials_raw, amp_u2_ultimaas_raw)
| filter xdm.event.type = "authentication"
| filter xdm.event.outcome = "failure"

| fields xdm.source.ipv4 as src, xdm.source.application.name as app, xdm.event.operation as signature, xdm.target.ipv4
as dest, xdm.source.user.username as user, xdm.target.user.username as duser,xdm.intermediate.user.username as
suser, xdm.target.host.hostname as host, xdm.event.outcome as outcome , xdm.event.outcome_reason as reason, xdm.
event.id as signature_id, _time,
xdm.observer.vendor, xdm.observer.product, peoplesoft_financials_raw._raw_log as p_raw_log, amp_u2_ultimaas_raw.
_raw_log as u_raw_log

// Using join to find user from access_amp_siem_excessive_failed_logins_exclusions_csv database
| join type=left(
    dataset = access_amp_siem_excessive_failed_logins_exclusions_csv
    | fields app as app_l, dest as dest_l, signature as signature_l , user_bunit as user_bunit_l, src_user as src_user_l, src
as src_l, user as user_l
) as excessive_failed_logins (excessive_failed_logins.user_l = user and incidr(src, excessive_failed_logins.src_l) = true
and suser = excessive_failed_logins.src_user_l)

// Renaming fields
| fields user_l as excluded_user ,src_user_l as excluded_src_user, src_l as excluded_src_cidr, *

// filtering field for null values
| filter excluded_user = null and excluded_src_cidr = null and excluded_src_user = null

// filtering field for authentication for app values for macro `amp_siem_standard_Authentication_apps`
| filter app in ("U2", "peoplesoft-financials")

| alter raw_log = coalesce(p_raw_log, u_raw_log)
| comp min(_time) as firstEventTime, max(_time) as lastEventTime, values(suser) as src_user, values(signature) as
signature, values(signature_id) as event_code, values(host) as host, values(reason) as reason, values(src) as src,
values(dest) as dest, values(duser) as duser, count() as hit_count, values(raw_log) as raw_log, last(xdm.observer.
vendor) as vendor, last(xdm.observer.product) as product by outcome, app, user

| alter threshold_val = If(app   = "U2", 8,
                          app   = "peoplesoft-financials", 10,
                          app = "north", 30, 6)

| filter hit_count >= threshold_val

| alter src = arraystring(src, ","),
       dest = arraystring(dest, ",")

| fields raw_log, outcome, app, user, src_user, duser, signature, event_code, src, dest, host, hit_count, reason, vendor,
product, firstEventTime, lastEventTime, threshold_val

Drill Down Query
datamodel dataset in (amp_u2_ultimaas_raw, peoplesoft_financials_raw)
| alter name = format_string("View failures by user %s for the application %s",$user, $app)
| filter xdm.event.outcome = "failure"
| filter $src contains xdm.source.ipv4
| filter $dest contains xdm.target.ipv4
| filter xdm.source.application.name = $app
| filter xdm.source.user.username = $user

| fields _time, name, xdm.event.outcome as action, xdm.source.user.username as user, xdm.intermediate.user.
username as src_user, xdm.source.application.name as app, xdm.target.ipv4 as dest, xdm.event.outcome_reason as
reason, xdm.event.operation as signature, xdm.event.id as signature_id
| alter start_offset_time = timestamp_seconds($firstEventTime)
| alter end_offset_time = timestamp_seconds($lastEventTime)
| filter _time >= start_offset_time and _time <= end_offset_time
| sort asc _time

```
| tstats fillnull_value=N/A summariesonly=t allow_old_summaries=t
values(Authentication.src_user) as src_user, values(Authentication.user_bunit) as user_bunit, values
(Authentication.signature) as signature,
values(Authentication.signature_id) as event_code, values(Authentication.user) as user, values
(Authentication.dest) as dest, count as hit_count from datamodel=Authentication
where Authentication.action IN (failure), sourcetype IN
`amp_siem_standard_Authentication_srcip_sourcetypes`,
Authentication.app IN `amp_siem_standard_Authentication_srcip_apps`, NOT Authentication.src IN
(unknown), NOT Authentication.user IN ("*$")
by index, sourcetype, Authentication.action, Authentication.app, Authentication.src
| `drop_dm_object_name("Authentication")`
| rename index as src_idx, sourcetype as src_st

| lookup access_amp_siem_excessive_failed_logins_exclusions app, dest, signature, user OUTPUT
user as excluded_user

| lookup access_amp_siem_excessive_failed_logins_exclusions app, dest, signature, user_bunit
OUTPUT user_bunit as excluded_bunit
| lookup access_amp_siem_excessive_failed_logins_exclusions app, dest, signature, src_user
OUTPUT src_user as excluded_src_user
| lookup access_amp_siem_excessive_failed_logins_exclusions app, dest, signature, src OUTPUT
src as excluded_src_cidr
| where isNull(excluded_user) AND isNull(excluded_src_user) AND isNull(excluded_src_cidr) AND
isNull(excluded_bunit)
| eval threshold_val = `amp_siem_standard_Authentication_thresholds`
| where hit_count >= ( threshold_val * 3 )
| fields - excluded_user, excluded_src_user, excluded_src_cidr, excluded_bunit, threshold_val
| eval urgency="high"

Drill Down

[{"name":"View failures by host $src$ for the application $app$","search":"index=*
sourcetype=$src_st$ tag=authentication action=failure src IN ($src$) dest IN ($dest$) app=\"$app$\"
user IN (\"$user$\")\n| fillnull value=N/A \n| table _time action user user_bunit user_agent src_user
app src dest signature signature_id reason\n| sort + _time","earliest_offset":"$info_min_time$","
latest_offset":"$info_max_time$"}]
```

```
config case_sensitive = false
| dataset = amp_u2_ultimaas_raw
| alter src = coalesce(src,_reporting_device_ip)
// filter failed logons
| filter outcome = "failure"
| alter signature_id = arrayindex(regextract(reason,"(^[^\s]+)"), 0)  // to be removed (testing)
| fields src, app, cefName, dest, user, duser, suser, host, outcome, reason, _reporting_device_ip, signature_id
// Using join to find user from access_amp_siem_excessive_failed_logins_exclusions_csv database
| join type=left(
    dataset = access_amp_siem_excessive_failed_logins_exclusions_csv
    | fields app as app_l, dest as dest_l, signature as signature , user_bunit as user_bunit_l, src_user as src_user_l, src
as src_l, user as user_l
) as excessive_failed_logins (excessive_failed_logins.user_l = user and incidr(src ,excessive_failed_logins.src_l)= true
and suser = excessive_failed_logins.src_user_l)

// Renaming fields
| fields user_l as excluded_user ,src_user_l as excluded_src_user, src_l as excluded_src_cidr, *

// filtering field for null values
| filter excluded_user = null and excluded_src_cidr = null and excluded_src_user = null //and excluded_src_user = null
and excluded_bunit = null

// filtering field for authentication for app values for macro `amp_siem_standard_Authentication_apps`
| filter app in ("U2", "peoplesoft-financials")

| comp min(_time) as firstEventTime, max(_time) as lastEventTime, values(suser) as src_user, values(cefName ) as
signature, values(reason) as reason, values(signature_id) as event_code, values(user) as user, values
(_reporting_device_ip) as dest, count() as hit_count, values(host) as host by outcome, app, src   //values(user_bunit) as
user_bunit

| alter threshold_val = if( app="U2",8,app ="peoplesoft-financials",10,app="north", 30, 6)
| filter hit_count >= multiply(threshold_val ,3)


| filter hit_count >= threshold_val

| alter dest = arraystring(dest, ","),
       user = arraystring(user, ",")


Drill Down Query
dataset = amp_u2_ultimaas_raw
| alter name = format_string("View failures by host %s for the application %s", $src, $app)
| alter src = coalesce(src,_reporting_device_ip)
| filter outcome = "failure"
| filter src = $src
| filter $dest contains _reporting_device_ip
| filter app = $app
| filter $user contains user
| alter signature_id = arrayindex(regextract(reason,"(^[^\s]+)"), 0)  // to be removed (testing)
| fields name, outcome as action, user, suser as src_user, app, _reporting_device_ip as dest, reason, cefName as
signature, signature_id
| alter start_offset_time = timestamp_seconds($firstEventTime)
| alter end_offset_time = timestamp_seconds($lastEventTime)
| filter _time >= start_offset_time and _time <= end_offset_time
| sort asc _time
```

```
| tstats fillnull_value=N/A summariesonly=t allow_old_summaries=t
values(Authentication.src_user) as src_user, values(Authentication.user_bunit) as user_bunit, values
(Authentication.signature) as signature,
values(Authentication.signature_id) as event_code, values(Authentication.user) as user, values
(Authentication.dest) as dest, dc(Authentication.user) as hit_count from datamodel=Authentication
where sourcetype IN `amp_siem_standard_Authentication_srcip_sourcetypes`,
Authentication.app IN `amp_siem_standard_Authentication_srcip_apps`, NOT Authentication.src IN
(unknown), NOT Authentication.user IN ("*$")
by index, sourcetype, Authentication.action, Authentication.app, Authentication.src
| `drop_dm_object_name("Authentication")`
| rename index as src_idx, sourcetype as src_st
| lookup access_amp_siem_excessive_accounts_exclusions app, dest, signature, user OUTPUT user
as excluded_user
| lookup access_amp_siem_excessive_accounts_exclusions app, dest, signature, user_bunit
OUTPUT user_bunit as excluded_bunit
| lookup access_amp_siem_excessive_accounts_exclusions app, dest, signature, src_user OUTPUT
src_user as excluded_src_user
| lookup access_amp_siem_excessive_accounts_exclusions app, dest, signature, src OUTPUT src as
excluded_src_cidr
| where isNull(excluded_user) AND isNull(excluded_src_user) AND isNull(excluded_src_cidr) AND
isNull(excluded_bunit)
| eval threshold_val = `amp_siem_standard_Authentication_thresholds`
| where hit_count >= threshold_val
| fields - excluded_user, excluded_src_user, excluded_src_cidr, excluded_bunit, threshold_val
| eval urgency="high"
```

Drill Down

```
[{"name":"View attempts by host $src$ for the application $app$","search":"index=*
sourcetype=$src_st$ tag=authentication src IN ($src$) dest IN ($dest$) app=\"$app$\" user IN
(\"$user$\")\n| fillnull value=N/A \n| table _time action user user_bunit user_agent src_user app src
dest signature signature_id reason\n| sort + _time","earliest_offset":"$info_min_time$","
latest_offset":"$info_max_time$"}]
```

```
config case_sensitive = false
| dataset = amp_u2_ultimaas_raw
| alter src = coalesce(src,_reporting_device_ip)
//| alter src = if(src in (null,""), "unknown", src)
| filter app = "U2" and src not in ("unknown") and user not in ("*$")
| alter signature_id = arrayindex(regextract(reason,"(^[^\s]+)"), 0)  // to be removed (testing)
| fields src, app, cefName, dest, user, duser, suser, host, outcome, reason , _reporting_device_ip, signature_id
// Using join to find user from access_amp_siem_excessive_failed_logins_exclusions_csv database
| join type=left(
    dataset = access_amp_siem_excessive_accounts_exclusions_csv
    | fields app as app_l, dest as dest_l, signature as signature , user_bunit as user_bunit_l, src_user as src_user_l, src
as src_l, user as user_l
) as excessive_accounts_exclusions  (excessive_accounts_exclusions.user_l = user and incidr(src ,
excessive_accounts_exclusions.src_l)= true and suser = excessive_accounts_exclusions.src_user_l)

// Renaming fields
| fields user_l as excluded_user ,src_user_l as excluded_src_user, src  ,src_l as excluded_src_cidr, * //hit_count, *

// filtering field for null values
| filter excluded_user = null and excluded_src_cidr = null and excluded_src_user = null //and excluded_src_user = null
and excluded_bunit = null

| comp min(_time) as firstEventTime, max(_time) as lastEventTime, values(suser) as src_user, values(user) as user,
values(cefName) as signature, values(signature_id) as event_code, values(reason) as reason, values
(_reporting_device_ip) as dest, count_distinct(user) as hit_count, values(host) as host by outcome, app, src    //values
(user_bunit) as user_bunit

| alter threshold_val = if( app ="U2",8,app ="peoplesoft-financials",10,app ="north", 30, 6)
| filter hit_count >= threshold_val


| alter dest = arraystring(dest, ","),
       user = arraystring(user, ",")

| fields outcome as action, app, src, src_user,signature , event_code , user ,dest , hit_count ,*
```

Drill Down Query

```
dataset = amp_u2_ultimaas_raw
| alter name = format_string("View attempts by host %s for the application %s", $src, $app)
| alter src = coalesce(src,_reporting_device_ip)
| filter src = $src
| filter $dest contains _reporting_device_ip
| filter app = $app
| filter $user contains user
| alter signature_id = arrayindex(regextract(reason,"(^[^\s]+)"), 0)  // to be removed (testing)
| fields name, outcome as action, user, suser as src_user, app, _reporting_device_ip as dest, reason, cefName as
signature, signature_id
| alter start_offset_time = timestamp_seconds($firstEventTime)
| alter end_offset_time = timestamp_seconds($lastEventTime)
| filter _time >= start_offset_time and _time <= end_offset_time
| sort asc _time
```

```
| tstats fillnull_value=N/A summariesonly=t allow_old_summaries=t
values(Authentication.src_user) as src_user, values(Authentication.user_bunit) as user_bunit, values
(Authentication.signature) as signature,
values(Authentication.signature_id) as event_code, values(Authentication.user) as user, values
(Authentication.dest) as dest, count as hit_count from datamodel=Authentication
where sourcetype IN `amp_siem_standard_Authentication_srcip_sourcetypes`,
Authentication.app IN `amp_siem_standard_Authentication_srcip_apps`, NOT Authentication.src IN
(unknown), NOT Authentication.user IN ("*$")
by index, sourcetype, Authentication.action, Authentication.app, Authentication.src
| `drop_dm_object_name("Authentication")`
| rename index as src_idx, sourcetype as src_st
| eval ip_type = case(match('src',"172.(1[6-9].|2[0-9].|3[0-1].)[0-9]{1,3}.[0-9]{1,3}"),"1_private",match
('src',"(10.[0-9]{1,3}.[0-9]{1,3}.[0-9]{1,3}"),"1_private",match('src',"(192.168.[0-9]{1,3}.[0-9]{1,3})"),"
1_private",match('src',"(127.[0-9]{1,3}.[0-9]{1,3}.[0-9]{1,3})"),"3_loopback",match('src',"(169.254.[0-9]
{1,3}.[0-9]{1,3})"),"2_apipa",1=1,"0_public")
| where ip_type!="1_private"
| lookup access_amp_siem_excessive_failed_logins_exclusions app, dest, signature, user OUTPUT
user as excluded_user
| lookup access_amp_siem_excessive_failed_logins_exclusions app, dest, signature, user_bunit
OUTPUT user_bunit as excluded_bunit
| lookup access_amp_siem_excessive_failed_logins_exclusions app, dest, signature, src_user
OUTPUT src_user as excluded_src_user
| lookup access_amp_siem_excessive_failed_logins_exclusions app, dest, signature, src OUTPUT
src as excluded_src_cidr
| where isNull(excluded_user) AND isNull(excluded_src_user) AND isNull(excluded_src_cidr) AND
isNull(excluded_bunit)
| fields - excluded_user, excluded_src_user, excluded_src_cidr, excluded_bunit
| eval urgency="high"
```

Drill Down Query:

```
[{"name":"View attempts by external host $src$ for the application $app$","search":"index=*
sourcetype=$src_st$ tag=authentication src IN ($src$) dest IN ($dest$) app=\"$app$\" user IN
(\"$user$\")\n| fillnull value=N/A \n| table _time action user user_bunit user_agent src_user app src
dest signature signature_id reason\n| sort + _time","earliest_offset":"$info_min_time$","
latest_offset":"$info_max_time$"}]
```

```
config case_sensitive = false
| datamodel dataset = amp_u2_ultimaas_raw
| filter xdm.event.type = "authentication"
| filter xdm.source.application.name in ("U2")  // filter for "lookup amp_siem_standard_Authentication_srcip_apps"
| filter xdm.source.user.username  not contains "*$"
| filter xdm.source.ipv4 != null

| alter ip_type = if(incidr(xdm.source.ipv4, "172.16.0.0/16, 172.31.0.0/16, 10.0.0.0/8, 192.168.0.0/16"), "1_private", incidr
(xdm.source.ipv4, "127.0.0.0/8"), "3_loopback", incidr(xdm.source.ipv4, "169.254.0.0/16"), "2_apipa", "0_public")
| filter ip_type != "1_private"

// Using join to find user from access_amp_siem_excessive_failed_logins_exclusions_csv database
| join type=left(
    dataset = access_amp_siem_excessive_failed_logins_exclusions_csv
    | fields app as app_l, dest as dest_l, signature as signature , user_bunit as user_bunit_l, src_user as src_user_l, src
as src_l, user as user_l
) as excessive_failed_logins_exclusions (excessive_failed_logins_exclusions.user_l = xdm.source.user.username and
incidr(xdm.source.ipv4 ,excessive_failed_logins_exclusions.src_l) = true and
xdm.intermediate.user.username = excessive_failed_logins_exclusions.src_user_l)

// Renaming fields
| fields _time, xdm.source.ipv4 as src, xdm.source.application.name as app, xdm.event.operation as signature, xdm.
target.ipv4 as dest, xdm.source.user.username as user, xdm.target.user.username as duser, xdm.intermediate.user.
username as suser, xdm.target.host.hostname  as host, xdm.event.outcome as outcome, xdm.event.outcome_reason as
reason, xdm.event.id as event_code, user_l as excluded_user, src_user_l as excluded_src_user, src_l as
excluded_src_cidr

// filtering field for null values
| filter excluded_user = null and excluded_src_cidr = null and excluded_src_user = null

| comp count() as hit_count, min(_time) as firstEventTime, max(_time) as lastEventTime, values(suser) as src_user,
values(user) as user, values(signature) as signature, values(event_code) as event_code, values(reason) as reason,
values(dest) as dest, values(host) as host by outcome, app, src

| alter dest = arraystring(dest, ","),
       user = arraystring(user, ",")
```

Drill Down

```
config case_sensitive = false
| datamodel dataset = amp_u2_ultimaas_raw
| filter xdm.event.type = "authentication"
| alter name = format_string("View attempts by external host %s for the application %s", $src, $app)
| filter xdm.source.ipv4 IN ($src)
| filter xdm.target.ipv4 IN ($dest)
| filter xdm.source.application.name = $app
| filter xdm.source.user.username IN ($user)

| fields _time, xdm.source.ipv4 as src, xdm.source.application.name as app, xdm.event.operation as signature, xdm.
target.ipv4 as dest, xdm.source.user.username as user, xdm.target.user.username as duser, xdm.intermediate.user.
username as suser, xdm.target.host.hostname  as host, xdm.event.outcome as outcome, xdm.event.outcome_reason as
reason, xdm.event.id as event_code
| sort asc _time
```

|from datamodel "Authentication"."Authentication" |search NOT app IN (ClearPass,"amp:was:ivr:services") |search signature_id!=4771 | stats count(eval(action="success")) as successes count(eval(action="failure")) as failures values(app) as app values(signature_id) as signature_id by dest user | where successes>0 AND failures>100 |search NOT user IN ("", "svc_infra_ad_dc", "svc_infra_ad", "svc_aws_ad", "*$")	// Title: Access - [AMP] Brute Force Attack Detected [NTT] - Rule config case_sensitive = false | datamodel dataset in(linux_linux_raw, amp_edw_raw,msft_o365_azure_ad_raw,msft_o365_general_raw, versa_gateway_raw,was_*,ibm_tim_raw, microsoft_windows_raw ,cyber_ark_vault_raw, amp*, salesforce_login_raw, amazon_aws_raw , cyber_ark_vault_raw) | filter xdm.event.type = "authentication" or xdm.observer.product in ("windows","linux") | alter dest= if(xdm.observer.product = "windows", xdm.source.host.hostname , xdm.observer.product !="windows", coalesce(xdm.target.host.hostname , xdm.target.ipv4 , xdm.target.host.fqdn)) | alter dest = if((dest not in (null, "")) , dest, "unknown") | filter xdm.event.outcome in (XDM_CONST.OUTCOME_FAILED , XDM_CONST.OUTCOME_SUCCESS) | fields xdm.event.id, xdm.observer.product,xdm.event.outcome,xdm.source.user.username , xdm.source.host.hostname ,xdm.target.user.username,xdm.target.host.hostname, xdm.target.ipv4, xdm.target.host.fqdn , dest,_time //_collector_hostname, _reporting_device_ip | filter xdm.event.id != "4771" // signature_id | alter authevent= if(xdm.observer.product = "windows" and xdm.event.id in("4625","4776","4672","4624"), "auth_event", xdm.observer.product !="windows","auth_event") | filter (authevent not in (null, "")) | alter user= if(xdm.observer.product = "windows", xdm.target.user.username , xdm.observer.product !="windows",xdm.source.user.username) | filter user not in(null, "", "svc_infra_ad_dc", "svc_infra_ad", "svc_aws_ad", "*$") // For other datasets. | comp count(if(xdm.event.outcome = "success", true)) as successes, count(if(xdm.event.outcome ="failed", true)) as failures, values(xdm.observer.product) as app, values(xdm.event.id) as signature_id , values(authevent) as authevent, values(xdm.source.user.username) as `xdm.source.user.username`, values(xdm.target.user.username) as `xdm.target.user.username`, min(_time) as start_time, max(_time) as end_time by dest, user | fields dest, user, successes , failures ,app, signature_id, xdm.source.user.username , xdm.target.user.username, start_time , end_time | filter successes > 0 and failures > 100

```
index=os_event_prod sourcetype=WinEventLog host IN (AMPSYDADCP002*, AMPSYDADCP003*, AMPMELADCP002*, AMPMELADCP003*, AMPAWSZ2ADCP001*, AMPAWSZ1ADCP001*) EventCode IN (4728, 4729, 4732, 4733) src_user!=svc-tim-prd
| rename Member_Account_Name as user
| stats count min(_time) as firstTime max(_time) as lastTime values(EventCode) as EventCode values(name) as name values(user) as user by dest, src_user, user_group
| `security_content_ctime(firstTime)`
| `security_content_ctime(lastTime)`
| `microsoft_AD_groups_change_outside_change_process_filter`
```

Drill Down Query

```
[{"name":"Members Added or Removed from Groups","search":"index=os_event_prod sourcetype=WinEventLog host IN (AMPSYDADCP002*, AMPSYDADCP003*, AMPMELADCP002*, AMPMELADCP003*, AMPAWSZ2ADCP001*, AMPAWSZ1ADCP001*) EventCode IN (4728, 4729, 4732, 4733) src_user!=svc-tim-prd\n| rename Member_Account_Name as user\n| stats count min(_time) as firstTime max(_time) as lastTime values(EventCode) as EventCode values(name) as name values(user_nick) as Member by dest, src_user, user_group\n| `security_content_ctime(firstTime)`\n| `security_content_ctime(lastTime)`\n| `microsoft_AD_groups_change_outside_change_process_filter`\n| table firstTime, lastTime, EventCode, dest, name, src_user, user_group, Member\n| rename dest as AD_Server, name as Name, user_group as User_Group","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]
```

```
// Title: Access - [AMP] Microsoft AD Group Change Outside Change Process [SplunkPS] - Rule

config case_sensitive = false
| datamodel dataset = microsoft_windows_raw
| filter xdm.event.id in ("4728", "4729", "4732", "4733")

| alter match_host = arrayindex(split(xdm.source.host.hostname, "."), 0)  // logic to remove domain for matching if present
| join ( dataset = endpoints
    | fields endpoint_name, endpoint_id
    | getrole endpoint_id as endpoint_role
    | filter  endpoint_role contains "Domain Controllers"
) as endpoint endpoint.endpoint_name contains match_host

| filter xdm.source.user.username != "svc-tim-prd"

| alter user_dn = json_extract_scalar(microsoft_windows_raw.event_data, "$.MemberName"),
        userSid = json_extract_scalar(microsoft_windows_raw.event_data, "$.MemberSid"),
        user_group = json_extract_scalar(microsoft_windows_raw.event_data, "$.TargetUserName"),
        userGroupSid = json_extract_scalar(microsoft_windows_raw.event_data, "$.TargetSid"),
        subjectLogonId = json_extract_scalar(microsoft_windows_raw.event_data, "$.SubjectLogonId")
| alter user = coalesce(arrayindex(regextract(user_dn, "CN=([^,]+)"), 0), user_dn)

| comp count() as total_event, min(_time) as firstTime, max(_time) as lastTime, values(user) as user, values(user_dn) as user_dn, values(userSid) as userSid, values(xdm.event.id) as eventCode, values(xdm.event.original_event_type) as signature, values(xdm.event.description) as message, values(xdm.source.user.domain) as subjectDomainName, values(xdm.target.user.domain) as targetDomainName, values(subjectLogonId) as subjectLogonId, values(xdm.source.user.identifier) as subjectUserSid, values(userGroupSid) as userGroupSid, values(xdm.event.type) as logName, values(xdm.event.outcome) as outcome,  values(xdm.event.operation_sub_type ) as name by xdm.source.user.username, user_group, xdm.source.host.hostname

| fields xdm.source.host.hostname as dest, xdm.source.user.username as src_user,user_group, total_event, firstTime, lastTime, eventCode , name ,user, user_dn, userSid,  signature,  message, subjectDomainName, subjectUserSid, subjectLogonId, targetDomainName, logName, outcome

DRill Down Query
config case_sensitive = false
| datamodel dataset = microsoft_windows_raw
| alter name = format_string("Members Added or Removed from Groups")

| filter xdm.event.id in ("4728", "4729", "4732", "4733")

| alter match_host = arrayindex(split(xdm.source.host.hostname, "."), 0)  // logic to remove domain for matching if present
| join ( dataset = endpoints
    | fields endpoint_name, endpoint_id
    | getrole endpoint_id as endpoint_role
    | filter  endpoint_role contains "Domain Controllers"
) as endpoint endpoint.endpoint_name contains match_host

| filter xdm.source.user.username != "svc-tim-prd"

| alter user_dn = json_extract_scalar(microsoft_windows_raw.event_data, "$.MemberName"),
        userSid = json_extract_scalar(microsoft_windows_raw.event_data, "$.MemberSid"),
        user_group = json_extract_scalar(microsoft_windows_raw.event_data, "$.TargetUserName"),
        userGroupSid = json_extract_scalar(microsoft_windows_raw.event_data, "$.TargetSid"),
        subjectLogonId = json_extract_scalar(microsoft_windows_raw.event_data, "$.SubjectLogonId")
| alter user = coalesce(arrayindex(regextract(user_dn, "CN=([^,]+)"), 0), user_dn)

| comp count() as total_event, min(_time) as firstTime, max(_time) as lastTime, values(user) as user, values(user_dn) as user_dn, values(userSid) as userSid, values(xdm.event.id) as eventCode, values(xdm.event.original_event_type) as signature, values(xdm.event.description) as message, values(xdm.source.user.domain) as subjectDomainName, values(xdm.target.user.domain) as targetDomainName, values(subjectLogonId) as subjectLogonId, values(xdm.source.user.identifier) as subjectUserSid, values(userGroupSid) as userGroupSid, values(xdm.event.type) as logName, values(xdm.event.outcome) as outcome by xdm.source.user.username, user_group, xdm.source.host.hostname, name

| fields total_event, firstTime, lastTime, user, user_dn, userSid, eventCode, signature, xdm.source.host.hostname as AD_Server, user_group, userGroupSid, xdm.source.user.username as username, message, subjectDomainName, subjectUserSid, subjectLogonId, targetDomainName, logName, outcome, name
```

```
| from datamodel:"Authentication"."Successful_Authentication"
| search Logon_Type=2 user_category="service_account"
| stats count min(_time) as firstTime max(_time) as lastTime values(action) as action values(signature) as signature values(signature_id) as signature_id by src, user, dest
| `security_content_ctime(firstTime)`
| `security_content_ctime(lastTime)`
| `microsoft_ad_service_account_interactive_login`
| search NOT user IN(RDMAPPService, SVC-FLEXERA-PRD, PAM_*_RDP_*, GWDYHQ)
```

Drill Down Query

[{"name":"List of service accounts logging in","search":"| from datamodel:\"Authentication\".\"Successful_Authentication\" \n| search Logon_Type=2 user_category=\"service_account\" \n| stats count min(_time) as firstTime max(_time) as lastTime values(action) as action values(signature) as signature values(signature_id) as signature_id by src, user, dest\n| `security_content_ctime(firstTime)`\n| `security_content_ctime(lastTime)`\n| `microsoft_ad_service_account_interactive_login`\n| table firstTime, lastTime, user, signature, signature_id, dest","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]

```
datamodel dataset = microsoft_windows_raw
| filter xdm.event.id = "4624"
| filter xdm.logon.type = "INTERACTIVE"
| filter xdm.target.user.username not in ("RDMAPPService", "SVC-FLEXERA-PRD", "PAM_*_RDP_*", "GWDYHQ")
| filter xdm.event.original_event_type != "Logon"
| alter action = if(xdm.event.id = "4624", "success", "Failure" )
| fields xdm.observer.action , xdm.event.outcome , xdm.source.host.hostname , xdm.target.host.hostname , *
| comp count() as total_events, min(_time) as firstTime, max(_time) as lastTime, values(action ) as action ,values(xdm.source.host.hostname) as dest, values(xdm.source.user.username) as source_username, values(xdm.source.user.domain) as source_userdomain, values(xdm.target.user.domain) as target_userdomain by xdm.target.user.username, xdm.source.ipv4, xdm.event.id, xdm.event.original_event_type, xdm.logon.type
// for dest computer name field is blank


| join (
    dataset = pan_dss_raw
    | filter ou contains "Service accounts"
    | fields display_name, email, ou
) as pan_dss_raw xdm.target.user.username = pan_dss_raw.display_name or xdm.target.user.username = pan_dss_raw.email

| fields firstTime, lastTime, total_events, source_username, source_userdomain, xdm.target.user.username as username, dest ,action ,target_userdomain, xdm.source.ipv4 as src,xdm.logon.type as logon_type,  xdm.event.original_event_type as signature, xdm.event.id as event_id, ou
```

Drill Down Query

```
datamodel dataset = microsoft_windows_raw
| alter name = format_string("List of service accounts logging in")
| filter xdm.event.id = "4624"
| filter xdm.logon.type = "INTERACTIVE"
| filter xdm.event.original_event_type != "Logon"
| alter action = if(xdm.event.id = "4624", "success", "Failure" )
| fields xdm.observer.action , xdm.event.outcome , xdm.source.host.hostname , xdm.target.host.hostname , *
| comp count() as total_events, min(_time) as firstTime, max(_time) as lastTime, values(action ) as action ,values(xdm.source.host.hostname) as dest, values(xdm.source.user.username) as source_username, values(xdm.source.user.domain) as source_userdomain, values(xdm.target.user.domain) as target_userdomain by xdm.target.user.username, xdm.source.ipv4, xdm.event.id, xdm.event.original_event_type, xdm.logon.type
// for dest computer name field is blank
| join (
    dataset = pan_dss_raw
    | filter ou contains "Service accounts"
    | fields display_name, email, ou
) as pan_dss_raw xdm.target.user.username = pan_dss_raw.display_name or xdm.target.user.username = pan_dss_raw.email
| fields firstTime, lastTime, xdm.target.user.username as username, dest , xdm.event.original_event_type as signature
```

```
sourcetype="amp:oracle:exadata:audit" tag=authentication
| eval ip_type = case(match('src_ip',"172.(1[6-9].|2[0-9].|3[0-1].)[0-9]{1,3}.[0-9]{1,3}"),"1_private",match
('src_ip',"(10.[0-9]{1,3}.[0-9]{1,3}.[0-9]{1,3})"),"1_private",match('src_ip',"(192.168.[0-9]{1,3}.[0-9]
{1,3})"),"1_private",match('src_ip',"(127.[0-9]{1,3}.[0-9]{1,3}.[0-9]{1,3})"),"3_loopback",match('src_ip',"
(169.254.[0-9]{1,3}.[0-9]{1,3})"),"2_apipa",1=1,"0_public")
| where ip_type!="1_private"
| stats count by src_ip, ip_type

Drill Down Query
[{"name":"[EDW - Prod] Unexpected login from external IP [eSecure]","search":"sourcetype=\"amp:
oracle:exadata:audit\" tag=authentication | eval ip_type = case(match('src_ip',\"172.(1[6-9].|2[0-9].|3[0-
1].)[0-9]{1,3}.[0-9]{1,3}\"),\"1_private\",match('src_ip',\"(10.[0-9]{1,3}.[0-9]{1,3}.[0-9]{1,3})\"),\"
1_private\",match('src_ip',\"(192.168.[0-9]{1,3}.[0-9]{1,3})\"),\"1_private\",match('src_ip',\"(127.[0-9]
{1,3}.[0-9]{1,3}.[0-9]{1,3})\"),\"3_loopback\",match('src_ip',\"(169.254.[0-9]{1,3}.[0-9]{1,3})\"),\"
2_apipa\",1=1,\"0_public\") | where ip_type!=\"1_private\" | stats count, values(user) as user by src_ip,
src_host, dest, ip_type","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]
```

```
// Title: Access - [EDW - Prod] Unexpected login from external IP [eSecure] - Rule
// Description: Detect unexpected logins for the EDW application from a external IP addresses.
// KB:
// https://teamtools.amp.com.au/confluence/pages/viewpage.action?spaceKey=IS&title=Enterprise+Data+Warehouse+%
28EDW%29+SIEM+Integration
// Author: Sahil Sharma, ssharma7@paloaltonetworks.com
// Datasets: amp_ibm_isam_raw
// Date: 04/Sep/2024

config case_sensitive = false
| datamodel dataset = amp_edw_raw
| filter xdm.event.operation_sub_type = "LOGON"

| alter ip_type = if(xdm.source.ipv4 ~= "172.(1[6-9].|2[0-9].|3[0-1].)[0-9]{1,3}.[0-9]{1,3}", "1_private",
                    xdm.source.ipv4 ~= "(10.[0-9]{1,3}.[0-9]{1,3}.[0-9]{1,3})", "1_private",
                    xdm.source.ipv4 ~= "(192.168.[0-9]{1,3}.[0-9]{1,3})", "1_private",
                    xdm.source.ipv4 ~= "(127.[0-9]{1,3}.[0-9]{1,3}.[0-9]{1,3})", "3_loopback",
                    xdm.source.ipv4 ~= "(169.254.[0-9]{1,3}.[0-9]{1,3})", "2_apipa", "0_public")

| filter ip_type != "1_private"
| comp count() as total_events, values(xdm.source.host.hostname) as USERHOST , values(xdm.source.user.username)
as USERNAME, min(_time ) as firstTime, max(_time) as lastTime, values(xdm.event.description) as COMMENT_TEXT,
values(xdm.auth.privilege_level) as PRIV_USED, values(xdm.event.outcome) as ACTION, values(xdm.auth.
auth_method) as DBLINK, values(xdm.network.ip_protocol) as PROTOCOL, values(xdm.source.port) as PORT by xdm.
source.ipv4 ,ip_type, xdm.event.operation_sub_type

| fields total_events, USERHOST, USERNAME, firstTime, lastTime, COMMENT_TEXT, xdm.source.ipv4 as SRC_IP,
ip_type, xdm.event.operation_sub_type as ACTION_NAME, PRIV_USED, ACTION, DBLINK, PROTOCOL, PORT

Drill Down Query
config case_sensitive = false
| datamodel dataset = amp_edw_raw
| alter name = format_string("[EDW - Prod] Unexpected login from external IP [eSecure]")
| filter xdm.event.operation_sub_type = "LOGON"
| alter ip_type = if(xdm.source.ipv4 ~= "172.(1[6-9].|2[0-9].|3[0-1].)[0-9]{1,3}.[0-9]{1,3}", "1_private",
                    xdm.source.ipv4 ~= "(10.[0-9]{1,3}.[0-9]{1,3}.[0-9]{1,3})", "1_private",
                    xdm.source.ipv4 ~= "(192.168.[0-9]{1,3}.[0-9]{1,3})", "1_private",
                    xdm.source.ipv4 ~= "(127.[0-9]{1,3}.[0-9]{1,3}.[0-9]{1,3})", "3_loopback",
                    xdm.source.ipv4 ~= "(169.254.[0-9]{1,3}.[0-9]{1,3})", "2_apipa", "0_public")
| filter ip_type != "1_private"
| comp count() as total_events, values(xdm.source.user.username) as USERNAME, min(_time ) as firstTime, max
(_time) as lastTime, values(xdm.event.description) as COMMENT_TEXT, values(xdm.auth.privilege_level) as
PRIV_USED, values(xdm.event.outcome) as ACTION, values(xdm.auth.auth_method) as DBLINK, values(xdm.network.
ip_protocol) as PROTOCOL, values(xdm.source.port) as PORT  by xdm.source.ipv4, ip_type, xdm.event.
operation_sub_type, xdm.source.host.hostname // dest field mapping not found

| fields total_events, xdm.source.host.hostname as USERHOST, USERNAME, firstTime, lastTime, COMMENT_TEXT,
xdm.source.ipv4 as SRC_IP, ip_type, xdm.event.operation_sub_type as ACTION_NAME, PRIV_USED, ACTION,
DBLINK, PROTOCOL, PORT
```

sourcetype="amp:oracle:exadata:audit" authentication=default action=failure | stats count by user Drill down Query of Splunk: sourcetype="amp:oracle:exadata:audit" authentication=default action=failure | stats count by user | where count>=10 | sort -count	// Title: Access - [EDW] Excessive Failed Logins [eSecure] - Rule config case_sensitive = false | datamodel dataset = amp_edw_raw | filter xdm.event.operation_sub_type = "LOGON" | filter xdm.event.outcome = XDM_CONST.OUTCOME_FAILED | filter xdm.auth.auth_method != "DBLINK" // authentication=default | comp count() as total_failed_logins, values(xdm.network.ip_protocol) as protocol, values(xdm.source.ipv4) as SRC_IP, values(xdm.source.port) as port, values(xdm.network.session_id) as SESSIONID, values(xdm.source.host.hostname) as USERHOST, max(_time) as lastEventTime, min(_time) as firstEventTime, values(xdm.auth.privilege_level) as PRIV_USED, values(xdm.event.description) as COMMENT_TEXT, values(xdm.intermediate.user.username) as OS_USERNAME by xdm.source.user.username, xdm.event.operation_sub_type, xdm.event.outcome | filter total_failed_logins >= 10 // filtering for high number of failed logins | fields total_failed_logins, firstEventTime, lastEventTime, protocol, port, SRC_IP, SESSIONID, USERHOST, PRIV_USED, xdm.source.user.username as USERNAME, OS_USERNAME, xdm.event.operation_sub_type as ACTION_NAME, COMMENT_TEXT, xdm.event.outcome as action Drill Down Query datamodel dataset = amp_edw_raw | filter xdm.event.operation_sub_type = "LOGON" | filter xdm.event.outcome = XDM_CONST.OUTCOME_FAILED | filter xdm.auth.auth_method != "DBLINK" // authentication=default | comp count() as total_failed_logins, values(xdm.network.ip_protocol) as protocol, values(xdm.source.ipv4) as SRC_IP, values(xdm.source.port) as port, values(xdm.network.session_id) as SESSIONID, values(xdm.source.host.hostname) as USERHOST, max(_time) as lastEventTime, min(_time) as firstEventTime, values(xdm.auth.privilege_level) as PRIV_USED, values(xdm.event.description) as COMMENT_TEXT, values(xdm.intermediate.user.username) as OS_USERNAME by xdm.source.user.username, xdm.event.operation_sub_type, xdm.event.outcome | filter total_failed_logins >= 10 // filtering for high number of failed logins | sort desc total_failed_logins | fields total_failed_logins, firstEventTime, lastEventTime, protocol, port, SRC_IP, SESSIONID, USERHOST, PRIV_USED, xdm.source.user.username as USERNAME, OS_USERNAME, xdm.event.operation_sub_type as ACTION_NAME, COMMENT_TEXT, xdm.event.outcome as action

	// Title: Access - [EDW] Excessive Failed Logins from a single IP [eSecure] - Rule // Description: Detect failed logins from the EDW application for a Single IP. Aggregate by user. // KB: // https://teamtools.amp.com.au/confluence/pages/viewpage.action?spaceKey=IS&title=Enterprise+Data+Warehouse+%28EDW%29+SIEM+Integration // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: amp_edw_raw // Date: 04/Sep/2024 config case_sensitive = false | datamodel dataset = amp_edw_raw | filter xdm.event.operation_sub_type = "LOGON" | filter xdm.event.outcome = XDM_CONST.OUTCOME_FAILED | filter xdm.auth.auth_method != "DBLINK" // authentication=default | comp count() as total_failed_logins, values(xdm.network.ip_protocol) as PROTOCOL, values(xdm.source.port) as PORT, values(xdm.network.session_id) as SESSIONID, values(xdm.source.host.hostname) as USERHOST, max(_time) as lastEventTime, min(_time) as firstEventTime, values(xdm.auth.privilege_level) as PRIV_USED, values(xdm.event.description) as COMMENT_TEXT, values(xdm.intermediate.user.username) as OS_USERNAME, values(xdm.source.user.username) as USERNAME, values(xdm.event.outcome) as ACTION by xdm.source.ipv4, xdm.event.operation_sub_type | filter total_failed_logins >= 10 // filtering for high number of failed logins | fields total_failed_logins, firstEventTime, lastEventTime, protocol, port, xdm.source.ipv4 as SRC_IP, SESSIONID, USERHOST, PRIV_USED, USERNAME, OS_USERNAME, xdm.event.operation_sub_type as ACTION_NAME, COMMENT_TEXT, ACTION
sourcetype="amp:oracle:exadata:audit" authentication=default action=failure | stats count by src_ip Drill down Query of Splunk: sourcetype="amp:oracle:exadata:audit" authentication=default action=failure | stats count by src_ip | where count>=10 | sort -count	Drill Down Query datamodel dataset = amp_edw_raw | filter xdm.event.operation_sub_type = "LOGON" | filter xdm.event.outcome = XDM_CONST.OUTCOME_FAILED | filter xdm.auth.auth_method != "DBLINK" // authentication=default | comp count() as total_failed_logins, values(xdm.network.ip_protocol) as PROTOCOL, values(xdm.source.port) as PORT, values(xdm.network.session_id) as SESSIONID, values(xdm.source.host.hostname) as USERHOST, max(_time) as lastEventTime, min(_time) as firstEventTime, values(xdm.auth.privilege_level) as PRIV_USED, values(xdm.event.description) as COMMENT_TEXT, values(xdm.intermediate.user.username) as OS_USERNAME, values(xdm.source.user.username) as USERNAME, values(xdm.event.outcome) as ACTION by xdm.source.ipv4, xdm.event.operation_sub_type | filter total_failed_logins >= 10 // filtering for high number of failed logins | fields total_failed_logins, firstEventTime, lastEventTime, protocol, port, xdm.source.ipv4 as SRC_IP, SESSIONID, USERHOST, PRIV_USED, USERNAME, OS_USERNAME, xdm.event.operation_sub_type as ACTION_NAME, COMMENT_TEXT, ACTION

sourcetype="amp:oracle:exadata:audit" tag=authentication | stats dc(user) as count by src_ip Drill Down Query: sourcetype="amp:oracle:exadata:audit" tag=authentication | stats values(user), dc(user) as count by src_ip | where count>=10 | sort -count	// Title: Access - [EDW] Excessive Number of Accounts from a single IP [eSecure] - Rule // Title: Access - [EDW] Excessive Number of Accounts from a single IP [eSecure] - Rule config case_sensitive = false | datamodel dataset = amp_edw_raw | filter xdm.event.type = "authentication" | comp count_distinct(xdm.source.user.username) as distinct_user_count, values(xdm.source.user.username) as USERNAME, values(xdm.intermediate.user.username) as OS_USERNAME, values(xdm.network.ip_protocol) as protocol, values(xdm.source.port) as port, values(xdm.network.session_id) as SESSIONID, values(xdm.source.host.hostname) as USERHOST, max(_time) as lastEventTime, min(_time) as firstEventTime, values(xdm.auth.privilege_level) as PRIV_USED, values(xdm.event.description) as COMMENT_TEXT, values(xdm.event.outcome) as action by xdm.source.ipv4 //, xdm.event.operation_sub_type | filter distinct_user_count >= 10 // filtering for more number of account loging from same src ip | fields distinct_user_count, firstEventTime, lastEventTime, protocol, port, xdm.source.ipv4 as SRC_IP, SESSIONID, USERHOST, PRIV_USED, USERNAME, OS_USERNAME, COMMENT_TEXT, action Drill Down Query datamodel dataset = amp_edw_raw | filter xdm.event.type = "authentication" | comp count_distinct(xdm.source.user.username) as distinct_user_count, values(xdm.source.user.username) as USERNAME, values(xdm.intermediate.user.username) as OS_USERNAME, values(xdm.network.ip_protocol) as protocol, values(xdm.source.port) as port, values(xdm.network.session_id) as SESSIONID, values(xdm.source.host.hostname) as USERHOST, max(_time) as lastEventTime, min(_time) as firstEventTime, values(xdm.auth.privilege_level) as PRIV_USED, values(xdm.event.description) as COMMENT_TEXT, values(xdm.event.outcome) as action by xdm.source.ipv4 | filter distinct_user_count >= 10 // filtering for more number of account loging from same src ip | sort desc distinct_user_count | fields distinct_user_count, firstEventTime, lastEventTime, protocol, port, xdm.source.ipv4 as SRC_IP, SESSIONID, USERHOST, PRIV_USED, USERNAME, OS_USERNAME, xdm.event.operation_sub_type as ACTION_NAME, COMMENT_TEXT, action

index=amp_pam_prod cef_signature IN (300, 19, 7, 411, 99, 8, 5, 1, 106, 62, 51) | transaction endswith=(cef_signature=7) maxspan=40s | search cef_signature!=300 AND cef_signature!=411 AND cef_signature!=99 AND cef_signature!=8 AND cef_signature!=5 AND cef_signature!=1 AND cef_signature!=106 AND cef_signature!=62 AND cef_signature!=51 | lookup pam_monitored_acc.csv "user" as "suser" OUTPUT "monitored" | where isnotnull(monitored) | stats count by _time, suser, shost, cs1 | rename cs1 as "Affected User Name"	dataset = cyber_ark_vault_raw | filter cefDeviceEventClassId in ("300", "19", "7", "411", "99", "8", "5", "1", "106", "62", "51") | fields cefDeviceEventClassId, act, _time, cefDeviceVersion, cefName, suser, shost, cs1 as affected_username | transaction cefDeviceEventClassId, act, cefDeviceVersion, cefName, suser, shost, affected_username startswith = "300" startswith = "19" startswith = "7" startswith = "411" startswith = "99" startswith = "8" startswith = "5" startswith = "1" startswith = "106" startswith = "62" startswith = "51" endswith = "7" | filter cefDeviceEventClassId != "300" and cefDeviceEventClassId != "411" and cefDeviceEventClassId != "99" and cefDeviceEventClassId != "8" and cefDeviceEventClassId != "5" and cefDeviceEventClassId != "1" and cefDeviceEventClassId != "106" and cefDeviceEventClassId != "62" and cefDeviceEventClassId != "51" | filter _duration <= 40 | join type = left (dataset = pam_monitored_acc_csv | fields user, monitored) as monitored_account monitored_account.user = suser | filter monitored not in ("", null) | comp count() as total_events, values(act) as act, values(cefDeviceVersion) as cefDeviceVersion, values(cefDeviceEventClassId) as cefDeviceEventClassId, values(cefName) as cefName by suser, shost, affected_username // New version dataset = cyber_ark_vault_raw | filter cefDeviceEventClassId in ("300", "19", "7", "411", "99", "8", "5", "1", "106", "62", "51") | fields cefDeviceEventClassId, act, _time, cefDeviceVersion, cefName, suser, cef_signature ,shost, cs1 as affected_username | replacenull affected_username ="NA" , cef_signature = "NA" | transaction suser, shost startswith = cefDeviceEventClassId contains "19" endswith = cefDeviceEventClassId contains "7" | filter _duration <= 40 | join conflict_strategy = both (dataset = pam_monitored_acc_csv | fields user, monitored) as monitored_account monitored_account.user = suser | filter monitored not in ("", null) | alter affected_username = arrayindex(regextract(arrayindex(arrayfilter(_raw,"@element" contains "Full Gateway Connection"),0) -> _raw_log,"cs1=(.+) cs2Label"),0) | comp count() as total_events, values(_start_time) as start_time, values(_end_time) as end_time by suser, shost, affected_username

index="os_event_prod" source="*WinEventLog:Security" user=* EventCode=* action=failure Logon_Type=* Failure Reason Logon Type Status=0xC000015B	// Title: Access - Detect Many Unauthorized Access Attempts [AMP] - Rule // Description: We're bringing in our Windows security logs, looking specifically for the status code 0xC000015B which indicates that the user hasn't been granted the requested logon type (aka, logon right). // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: msft_graph_security_alerts_raw // Date: 07/May/2024 /* index="os_event_prod" source="*WinEventLog:Security" user=* EventCode=* action=failure Logon_Type=* Failure Reason Logon Type Status=0xC000015B */ config case_sensitive = false | datamodel dataset = microsoft_windows_raw | filter xdm.observer.type = "Microsoft-Windows-Security-Auditing" | filter xdm.event.id = "4625" | alter status = json_extract_scalar(microsoft_windows_raw.event_data, "$.Status") | filter status = "0xC000015B" | filter xdm.event.outcome = "FAILED" | filter xdm.target.user.username not in ("", null) | fields microsoft_windows_raw.event_data, xdm.event.id, xdm.event.original_event_type, xdm.target.user.username, xdm.source.user.username, xdm.target.host.hostname, xdm.source.host.hostname, status, xdm.logon.type, xdm.event.description, * | comp count() as total_failed_logins, min(_time) as firstTime, max(_time) as lastTime, values(xdm.source.user.username) as subjectUser, values(xdm.source.host.hostname) as host by xdm.target.user.username, xdm.event.id, status, xdm.event.outcome, xdm.event.original_event_type // | filter total_failed_logins >= 5 //

	//Title: Access - Insecure Or Cleartext Authentication - Rule config case_sensitive = false | datamodel dataset in (linux_linux_raw,microsoft_windows_raw ,amp_edw_raw,msft_o365_azure_ad_raw, msft_o365_general_raw,versa_gateway_raw,was_*,ibm_tim_raw ,cyber_ark_vault_raw, amp*, salesforce_login_raw, amazon_aws_raw , cyber_ark_vault_raw) //| filter xdm.event.outcome in (XDM_CONST.OUTCOME_FAILED , XDM_CONST.OUTCOME_SUCCESS) | filter xdm.event.type = "authentication" or xdm.observer.product in ("windows","linux") | alter authevent= if(xdm.observer.product = "windows" and xdm.event.id in("4625","4776","4672","4624"), "auth_event", xdm.observer.product !="windows","auth_event") | filter (authevent not in (null, "")) | alter dest= if(xdm.observer.product = "windows", xdm.source.host.hostname , xdm.observer.product !="windows", coalesce(xdm.target.host.hostname , xdm.target.ipv4 , xdm.target.host.fqdn)) | alter dest = if((dest not in (null, "")) , dest, "unknown") | alter insecure= if(xdm.observer.product = "windows" and xdm.logon.type = XDM_CONST. LOGON_TYPE_NETWORK_CLEARTEXT , "insecure", (xdm.observer.product !="windows" and (xdm.target.port in(21, 80, 23, 25, 110, 143, 161, 2049, 514, 389, 513))),"insecure") | filter (insecure not in (null, "")) | alter user= if(xdm.observer.product = "windows", xdm.target.user.username , xdm.observer.product !="windows",xdm. source.user.username) | fields _time ,xdm.event.id, xdm.event.type,authevent ,insecure, xdm.source.port ,xdm.target.port , xdm.observer. product as app,xdm.network.application_protocol, xdm.network.application_protocol_category , xdm.network. application_protocol_subcategory , xdm.event.outcome ,xdm.logon.type, xdm.source.user.username , xdm.source.host. hostname ,xdm.target.user.username,xdm.target.host.hostname, xdm.target.ipv4, xdm.target.host.fqdn , dest,xdm. event.description , xdm.event.operation ,xdm.event.operation_sub_type , xdm.auth.service , xdm.network. protocol_layers , xdm.intermediate.port , xdm.network.ip_protocol,* | filter user not contains "PBIEgwService" or user not contains "IUSR_" or user not contains "HealthMailbox" or user not contains "$" | comp count() as count, values(xdm.event.id) as signature_id , values(xdm.event.description) as raw_log, values(xdm. source.user.username) as `xdm.source.user.username`, values(xdm.target.user.username) as `xdm.target.user. username`, values(user) as user, min(_time) as start_time, max(_time) as end_time by dest, app //values(xdm. observer.product) as app,values(authevent) as authevent, | fields app,dest , user , xdm.source.user.username , xdm.target.user.username , signature_id , start_time , end_time , count , raw_log , *
| from datamodel:"Authentication"."Insecure_Authentication" |search NOT src_user IN (PBIEgwService, IUSR_*, HealthMailbox*, *$) \| stats max("_time") as "lastTime",latest("_raw") as "orig_raw",values("tag") as "tag" values(src_user) as "src_user",count by "app","dest" Drill Down \| from datamodel:"Authentication"."Insecure_Authentication" \| search app=$app|s$ dest=$dest\|s$	Drill Down config case_sensitive = false | datamodel dataset in (linux_linux_raw,microsoft_windows_raw ,amp_edw_raw,msft_o365_azure_ad_raw, msft_o365_general_raw,versa_gateway_raw,was_*,ibm_tim_raw ,cyber_ark_vault_raw, amp*, salesforce_login_raw, amazon_aws_raw , cyber_ark_vault_raw) | alter dest= if(xdm.observer.product = "windows", xdm.source.host.hostname , xdm.observer.product !="windows", coalesce(xdm.target.host.hostname , xdm.target.ipv4 , xdm.target.host.fqdn)) | alter dest = if((dest not in (null, "")) , dest, "unknown") | filter xdm.observer.product = $app | filter dest = $dest

```
| tstats `summariesonly` dc(All_Changes.action) as action_count values(All_Changes.action) as action
from datamodel=Change.All_Changes where nodename="All_Changes.Account_Management"
(All_Changes.action="created" OR All_Changes.action="deleted") by _time,All_Changes.dest,
All_Changes.user span=1s
| `drop_dm_object_name("All_Changes")`
| lookup short-lived_accounts_expected.csv Account as user OUTPUT Confirmed
| where isnull(Confirmed)
| streamstats range(_time) as delta,sum(count) as count by user,dest window=2 global=f
| where action_count>1 | `uptime2string(delta,timestr)` |where delta>1 |search  NOT ((user=unknown
AND dest="iam.amazonaws.com") OR (user=svc-conbastion AND dest="AMPAZ1ADCP01.au.amp.
local"))
| table user, dest, action_count,action,delta, timestr
```

Drill Down Query

```
| from datamodel:"Change"."Account_Management" | search user=$user|s$ (action="created" OR
action="deleted")
```

```
config case_sensitive = false
| datamodel dataset = microsoft_windows_raw
| filter microsoft_windows_raw._collector_type = "XDR Collector"
| filter xdm.event.id = "4720" or xdm.event.id = "4726"

| alter search1 = if(xdm.target.user.username in ("", null, "unknown") and xdm.source.host.hostname = "iam.amazonaws.
com", true, false),
       search2 = if(xdm.target.user.username = "svc-conbastion" and xdm.source.host.hostname = "AMPAZ1ADCP01.au.
amp.local", true, false)
| filter search1 = false or search2 = false

| alter UserCreationTime = if(xdm.event.id = "4720", _time),
        UserDeletionTime = if(xdm.event.id = "4726", _time),
        userPrincipalName = json_extract_scalar(microsoft_windows_raw.event_data, "$.UserPrincipalName")
| fields xdm.event.description, xdm.event.operation_sub_type, xdm.source.user.username, xdm.target.user.username,
xdm.event.outcome, UserCreationTime, UserDeletionTime, xdm.event.description, userPrincipalName, *

| join type = left (
    dataset = short_lived_accounts_expected_csv
    | fields Account as user, Confirmed
) as expected_account expected_account.user = xdm.target.user.username
| filter Confirmed = null

| comp count() as total_events, count_distinct(xdm.event.id) as action_count, values(xdm.event.id) as event_id, values
(xdm.event.original_event_type) as signature, values(xdm.source.user.username) as subject_user, earliest
(UserCreationTime) as UserCreatedTime, latest(UserDeletionTime) as UserDeletedTime, values(userPrincipalName) as
userPrincipalName, values(xdm.event.outcome) as outcome by xdm.target.user.username, xdm.source.host.hostname
| filter action_count > 1
| alter delta = timestamp_diff(UserCreatedTime , UserDeletedTime, "SECOND")
| filter delta > 1 // In seconds
| fields subject_user, xdm.target.user.username as user, UserCreatedTime, UserDeletedTime, signature, outcome, xdm.
source.host.hostname as dest, event_id, total_events, userPrincipalName
```

|inputlookup Applications_lastdata_info.csv |dedup index sourcetype | eval received_in_12h = if(lastTime > relative_time(now(),"-12h"),1,0), received_in_24h = if(lastTime > relative_time(now(),"-24h"),1,0), last_received = strftime(lastTime,"%c"), diff_in_hours=round((now()-lastTime)/3600, 2) |eval diff_in_days=round(diff_in_hours/24, 2) |lookup sourcetypes_exclusions.csv index sourcetype OUTPUTNEW Logs_Frequency |where diff_in_hours>24 |search NOT Logs_Frequency=0 |fillnull value=0|where diff_in_days>Logs_Frequency | rename sourcetype as Sourcetype index as Index |search NOT Index IN (main,_audit,_internal,lastchanceindex,ingest_testing,os_perf_prod,*nonprod, os_event_prod,ampc_realestate_prod,ampc_international_prod,_telemetry) |fields Index Sourcetype last_received received_in_12h received_in_24h diff_in_hours |eval received_in_12h=if(received_in_12h == 1,"Logs Received", "Logs not Received") |eval received_in_24h=if(received_in_24h == 1,"Logs Received", "Logs not Received") | eval urgency="high" Drill Down [{"name":"Drilldown to contributing events","search":"| tstats count where index=$Index$ sourcetype=$Sourcetype$ by host index sourcetype _time span=1s | stats count latest(_time) as Latest_Time by index sourcetype host | fieldformat Latest_Time=strftime(Latest_Time,\"%Y-%m-%d %H:%M\")","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	// Title: Audit - Critical Applications not reporting to Splunk - Rule // Description: There is a list of critical applications in AMP that needs to continuously send logs to Splunk Cloud. This usecase will trigger notable alerts whenever any expected sourcetype/relevant logs are not reporting to Splunk in the last 24 hrs // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: metrics_source // Date: 07/May/2024 config case_sensitive = false | preset = metrics_view | comp max(last_seen) as latest_time, latest(total_event_count) as total_event, latest(total_event_rate) as total_event_rate, values(_collector_hostname) as hostname, values(_collector_type) as host_type, values(_collector_ip) as host_ip, values(_collector_id) as host_id, values(_collector_name) as collector_name by _product, _vendor | alter received_in_12h = if(divide(timestamp_diff(current_time(), latest_time, "minute"), 60) < 12, 1, 0), received_in_24h = if(divide(timestamp_diff(current_time(), latest_time, "minute"), 60) < 24, 1, 0), diff_in_hrs = divide(timestamp_diff(current_time(), latest_time, "minute"), 60), diff_in_days = divide(timestamp_diff(current_time(), latest_time, "hour"), 24), last_received = latest_time // exclude product and vendor as per the lookup table | join type = left (dataset = sourcetypes_exclusions_csv | alter Logs_Frequency = to_number(Logs_Frequency) | fields Logs_Frequency, vendor, product, index, `xsiam dataset` as dataset_name) as exclusion_list exclusion_list.product = _product and exclusion_list.vendor = _vendor | filter diff_in_hrs > 24 // filter out events having ingestion less than 24 hours | filter Logs_Frequency != 0 | replacenull Logs_Frequency = 0 | filter diff_in_days > Logs_Frequency // | filter index not in ("main", "audit", "_internal", "lastchanceindex", "ingest_testing", "os_perf_prod", "*nonprod", "os_event_prod", "ampc_realestate_prod", "ampc_international_prod", "_telemetry") | alter received_in_12h = if(received_in_12h = 1, "Logs Received", "Logs Not Received"), // this is not relevant as we are filtering for > 24 hrs so it will always be Logs Not received received_in_24h = if(received_in_24h = 1, "Logs Received", "Logs Not Received") | fields received_in_12h, received_in_24h, diff_in_hrs, diff_in_days, latest_time, _vendor, _product, Logs_Frequency, last_received, host_ip, hostname, host_type, host_id, collector_name, dataset_name Drill Down config case_sensitive = false | preset = metrics_view | alter name = "Drilldown to contributing events" | filter _product = $_product | filter _vendor = $_vendor | comp sum(total_event_count) as total_event_count_sum, max(last_seen) as latest_time by _vendor, _product, _collector_hostname, _collector_ip | fields total_event_count_sum, latest_time, _vendor, _product, _collector_hostname as hostname, _collector_ip as host_ip

```
| metadata type=hosts index=*
| eval "host"=lower('host')
| stats min(firstTime) as firstTime,max(recentTime) as recentTime,max(lastTime) as lastTime,sum
(totalCount) as totalCount by host
| eval _time=lastTime
| lookup update=true asset_lookup_by_str asset as "host" OUTPUTNEW _key as host_asset_id,app
as host_app,asset as host_asset,asset_id as host_asset_id,asset_tag as host_asset_tag,bunit as
host_bunit,category as host_category,city as host_city,country as host_country,description as
host_description,dns as host_dns,expected_hours as host_expected_hours,image_id as
host_image_id,image_name as host_image_name,instance_type as host_instance_type,ip as
host_ip,is_expected as host_is_expected,lat as host_lat,long as host_long,mac as host_mac,
network_interface_id as host_network_interface_id,nt_host as host_nt_host,owner as host_owner,
pci_domain as host_pci_domain,priority as host_priority,procedure as host_procedure,requires_av as
host_requires_av,should_timesync as host_should_timesync,should_update as host_should_update,
subnet_id as host_subnet_id,vendor_account as host_vendor_account,vendor_region as
host_vendor_region
| lookup update=true asset_lookup_by_cidr asset as "host" OUTPUTNEW _key as host_asset_id,app
as host_app,asset as host_asset,asset_id as host_asset_id,asset_tag as host_asset_tag,bunit as
host_bunit,category as host_category,city as host_city,country as host_country,description as
host_description,dns as host_dns,expected_hours as host_expected_hours,image_id as
host_image_id,image_name as host_image_name,instance_type as host_instance_type,ip as
host_ip,is_expected as host_is_expected,lat as host_lat,long as host_long,mac as host_mac,
network_interface_id as host_network_interface_id,nt_host as host_nt_host,owner as host_owner,
pci_domain as host_pci_domain,priority as host_priority,procedure as host_procedure,requires_av as
host_requires_av,should_timesync as host_should_timesync,should_update as host_should_update,
subnet_id as host_subnet_id,vendor_account as host_vendor_account,vendor_region as
host_vendor_region
| lookup asset_lookup_default_fields key as host OUTPUTNEW pci_domain as host_pci_domain,
is_expected as host_is_expected, requires_av as host_requires_av, should_timesync as
host_should_timesync, should_update as host_should_update
| eval "host_ip"=case(match('host_ip', "^\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}$"),'host_ip',match('host', "^\d
{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}$"),host,1=1,null()),"tag"=mvdedup(mvappend('tag',
NULL,'host_asset_tag'))
| lookup update=true identity_lookup_expanded identity as host_owner OUTPUTNEW _key as
host_owner_identity_id,bunit as host_owner_bunit,category as host_owner_category,email as
host_owner_email,endDate as host_owner_endDate,first as host_owner_first,identity as
host_owner_identity,identity_tag as host_owner_identity_tag,last as host_owner_last,managedBy as
host_owner_managedBy,nick as host_owner_nick,phone as host_owner_phone,prefix as
host_owner_prefix,priority as host_owner_priority,startDate as host_owner_startDate,suffix as
host_owner_suffix,uac as host_owner_uac,watchlist as host_owner_watchlist,work_city as
host_owner_work_city,work_country as host_owner_work_country,work_lat as host_owner_work_lat,
work_long as host_owner_work_long
| lookup identity_lookup_default_fields key as host_owner OUTPUTNEW watchlist as
host_owner_watchlist
| eval "tag"=mvdedup(mvappend('tag',NULL,'host_owner_identity_tag')),"host_owner_startDate"=case
(isnum('host_owner_startDate'),'host_owner_startDate',isnum(strptime('host_owner_startDate',"%m/%
d/%Y %H:%M")),strptime('host_owner_startDate',"%m/%d/%Y %H:%M"),isnum(strptime
('host_owner_startDate',"%m/%d/%y %H:%M")),strptime('host_owner_startDate',"%m/%d/%y %H:%
M"),1=1,'host_owner_startDate'),"host_owner_endDate"=case(isnum
```

```
// Title: Audit - Expected Host Not Reporting v2 [NTT] - Rule
// Description: Discovers hosts that are longer reporting events but should be submitting log events. This rule is used to
monitor hosts that you know should be providing a constant stream of logs
    // in order to determine why the host has failed to provide log data.
    // [ NTT SVR23297736 | AMP RITM01781098 ] - expected host not reporting --- BlueCoat decommissioned - adding
host_exclusion kv lookup table to exclude decommissioned hosts. These hosts are to review once per quarter by NTT.
Once the hostname is flagged as decommissioned on ServiceNOW cmdb asset inventory, the hostname will be
removed from the host_exclusion list.
// Author: Sahil Sharma, ssharma7@paloaltonetworks.com
// Preset: metrics_view
// Date: 12/Sep/2024

config case_sensitive = false
| preset = metrics_view
| alter _vendor = lowercase(_vendor),
        _product = lowercase(_product),
        _collector_hostname = lowercase(_collector_hostname)

| comp sum(total_event_count) as total_event_count_sum, max(last_seen) as latestTime, values(_vendor) as vendor,
values(_product) as product, last(_collector_id) as host_id by _collector_hostname, _collector_ip
| filter total_event_count_sum = 0   // to find hosts with no data ingestion
| alter match_hostname = arrayindex(split(_collector_hostname, "."), 0)  // removing the domain from hostname to
compare with cmdb dataset

// filtering hostname from host exclusion lookup
| join type = left( dataset = host_exclusions
    | fields host as hostname
) as host_exclusion_list _collector_hostname contains host_exclusion_list.hostname
| filter hostname in (null, "")

// filtering ip adderess from host exclusion lookup
| join type = left( dataset = host_exclusions
    | fields ip_address
) as host_exclusion_list host_exclusion_list.ip_address = _collector_ip
| filter ip_address in ("", null)

// calculating day and hour difference
| alter curTime = current_time()
| alter dayDiff = timestamp_diff(curTime, latestTime, "DAY"),
        hourDiff = timestamp_diff(curTime, latestTime, "HOUR"),
        hourDiffAbsolute = divide(timestamp_diff(curTime, latestTime, "MINUTE"), 60)

| fields match_hostname, _collector_hostname, _collector_ip, vendor, product, latestTime, hourDiffAbsolute, hourDiff,
dayDiff, host_id

// Static Assets
| join type = left (dataset = static_assets
    // | filter category contains "prod" and category not contains "non_prod"
    | alter static_asset = true
    | fields category, country, dns, priority, static_asset, owner  as owned_by
) as static_asset static_asset.dns contains match_hostname

| join type = left (dataset = servicenow_cmdb_cmdb_ci_service_raw
    // | filter used_for contains "production"
    | alter cmdb_service = true
    | fields sys_created_by, sys_created_on, sys_domain, sys_id, sys_updated_by, sys_updated_on, serial_number,
manufacturer, asset, category, short_description, owned_by, location, u_environment, cost_center, dns_domain,
ip_address, cmdb_service
    | comp values(*) as * by ip_address
) as cmdb_service cmdb_service.ip_address = _collector_ip

// Host enrichment
| join type = left (dataset = servicenow_cmdb_cmdb_ci_server_raw
    // | filter used_for contains "production"
    | alter cmdb_server = true
    | fields host_name, sys_created_by, sys_created_on, sys_domain, sys_id, sys_updated_by, sys_updated_on,
serial_number, manufacturer, asset, category, short_description, owned_by, location, os, u_environment, cost_center,
disk_space, cpu_core_count, cpu_core_thread, cpu_count, cpu_name, cpu_speed, cpu_type, dns_domain, ip_address,
cmdb_server
    | comp values(*) as * by ip_address, host_name
```

(host=*.*splunk*.* NOT host=sh*.*splunk*.* index=_telemetry source=*license_usage_summary.log* type="RolloverSummary") | bin _time span=1d | stats latest(b) AS b by slave, pool, _time | timechart span=1d sum(b) AS "volume" fixedrange=true | eval GB=round((((volume / 1024) / 1024) / 1024),3), Volume=GB | fields - GB, volume | search Volume > 135 | eval urgency = "high" Drill Down (host=*.*splunk*.* NOT host=sh*.*splunk*.* index=_telemetry source=*license_usage_summary.log* type="RolloverSummary") | bin _time span=1d | stats latest(b) AS b by slave, pool, _time | timechart span=1d sum(b) AS "volume" fixedrange=true | eval GB=round((((volume / 1024) / 1024) / 1024),3), Volume=GB | fields - GB, volume	config case_sensitive = false | dataset = metrics_source | fields _vendor , _product , total_size_bytes , total_size_rate | comp sum(total_size_bytes) as ingestion | alter Ingestion_by_GB = divide(round(multiply(divide(ingestion , pow(2,30)),1000)),1000) //Rounding to 3 Decimal Places | alter Percentage_Ingestion_Done = divide(round(multiply(divide(Ingestion_by_GB, 650),10000)),100) // Ingested in GB Divided by Total Ingestible Limit 650GB | filter Percentage_Ingestion_Done >= 90 | fields Ingestion_by_GB as total_GB_used, Percentage_Ingestion_Done as percentage_license_utilise, ingestion as number_of_logs_ingested , * Drill Down config case_sensitive = false | dataset = metrics_source | fields _vendor , _product , total_size_bytes , total_size_rate | comp sum(total_size_bytes) as ingestion | alter Ingestion_by_GB = divide(round(multiply(divide(ingestion , pow(2,30)),1000)),1000) //Rounding to 3 Decimal Places | alter Percentage_Ingestion_Done = divide(round(multiply(divide(Ingestion_by_GB, 650),10000)),100) // Ingested in GB Divided by Total Ingestible Limit 650GB

```
index=os_event_prod sourcetype=WinEventLog host IN (AMPSYDADCP002*, AMPSYDADCP003*, AMPMELADCP002*, AMPMELADCP003*, AMPAWSZ2ADCP001*, AMPAWSZ1ADCP001*) EventCode IN (4728, 4729, 4732, 4733)
| lookup au_amp_local_ad_group.csv sAMAccountName as Group_Name OUTPUT desktopProfile
| search desktopProfile=HIGH_PRIVILEGED_AD_GROUP
| rename Member_Account_Name as user
| stats count min(_time) as firstTime max(_time) as lastTime values(EventCode) as EventCode values(name) as name values(user) as user by dest, src_user, user_group
| `security_content_ctime(firstTime)`
| `security_content_ctime(lastTime)`
| `microsoft_ad_highly_privileged_groups_change_filter`
```

Drill Down Query

```
[{"name":"Members Added or Removed from Groups","search":"index=os_event_prod sourcetype=WinEventLog host IN (AMPSYDADCP002*, AMPSYDADCP003*, AMPMELADCP002*, AMPMELADCP003*, AMPAWSZ2ADCP001*, AMPAWSZ1ADCP001*) EventCode IN (4728,4729,4732,4733)\n| lookup au_amp_local_ad_group.csv sAMAccountName as Group_Name OUTPUT desktopProfile\n| search desktopProfile=HIGH_PRIVILEGED_AD_GROUP\n| rename Member_Account_Name as user\n| stats count min(_time) as firstTime max(_time) as lastTime values(EventCode) as EventCode values(name) as name values(user_nick) as Member by dest, src_user, user_group\n| `security_content_ctime(firstTime)`\n| `security_content_ctime(lastTime)`\n| `microsoft_ad_highly_privileged_groups_change_filter`\n| table firstTime, lastTime, EventCode, dest, name, src_user, user_group, Member\n| rename dest as AD_Server, name as Name, user_group as User_Group","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]
```

Lookup for XSIAM

dataset = ad_high_desktop_profile By HIEP

```
// Title: Endpoint - [AMP] Microsoft AD High Privileged Group Change [SplunkPS] - Rule
config case_sensitive = false
| datamodel dataset = microsoft_windows_raw
| filter xdm.event.id in ("4728", "4729", "4732", "4733")
| filter xdm.source.host.hostname in ("AMPSYDADCP002*", "AMPSYDADCP003*", "AMPMELADCP002*", "AMPMELADCP003*", "AMPAWSZ2ADCP001*", "AMPAWSZ1ADCP001*")

| join (
    dataset = ad_high_desktop_profile
    | filter desktopProfile = "HIGH_PRIVILEGED_AD_GROUP"
    | fields desktopProfile, sAMAccountName
) as amp_local_ad_group amp_local_ad_group.sAMAccountName = xdm.target.user.username

| alter user = json_extract_scalar(microsoft_windows_raw.event_data, "$.MemberName")

| comp count() as total_events, min(_time) as firstTime, max(_time) as lastTime, values(xdm.event.id) as eventCode, values(xdm.event.original_event_type) as event_type, values(user) as user by xdm.source.user.username, xdm.target.user.username, xdm.source.host.hostname, desktopProfile
| fields xdm.source.host.hostname ,xdm.source.user.username , xdm.target.user.username , total_events , firstTime , lastTime , eventCode , event_type , *
```

Drill Down Query

```
config case_sensitive = false
| datamodel dataset = microsoft_windows_raw
| alter name = format_string("Members Added or Removed from Groups")
| filter xdm.event.id in ("4728", "4729", "4732", "4733")
| filter xdm.source.host.hostname in ("AMPSYDADCP002*", "AMPSYDADCP003*", "AMPMELADCP002*", "AMPMELADCP003*", "AMPAWSZ2ADCP001*", "AMPAWSZ1ADCP001*")

| join (
    dataset = ad_high_desktop_profile
    | filter desktopProfile = "HIGH_PRIVILEGED_AD_GROUP"
    | fields desktopProfile, sAMAccountName
) as amp_local_ad_group amp_local_ad_group.sAMAccountName = xdm.target.user.username

| alter user = json_extract_scalar(microsoft_windows_raw.event_data, "$.MemberName"),
       user_group = json_extract_scalar(microsoft_windows_raw.event_data, "$.TargetUserName")
| comp count() as total_events, min(_time) as firstTime, max(_time) as lastTime, values(xdm.event.id) as eventCode, values(xdm.event.original_event_type) as event_type, values(user) as user, values(name) as name ,values(user_group ) as user_group by xdm.source.user.username, xdm.target.user.username, xdm.source.host.hostname, desktopProfile
| fields firstTime ,lastTime ,eventCode ,xdm.target.user.username ,name,xdm.source.user.username ,user_group ,user
```

```
index=amp_msdefender_prod sourcetype=ms:defender:atp:alerts productName="Microsoft Defender for Identity" severity != informational
| fillnull value=""
| stats count min(_time) as firstTime max(_time) as lastTime values(mitre_technique_id) as mitre_technique_id values(category) as category last(subject) as subject last(productName) as productName last(severity) as severity last(description) as description last(dest) as dest  last(incidentWebUrl) as incidentWebUrl last(status) as incidentStatus by incidentId
| eval urgency='severity'
``` Although the intention was not to create a risk rule here, usefull for the future```
| rename mitre_technique_id as annotations.mitre_attack
``` lookup the the mitre attack tatic id based on the category value casue most of the time technique_ids are missing```
| rename category as mitre_tactic_label
| join mitre_tactic_label [| inputlookup mitre_attack_lookup | stats count by mitre_tactic_label, mitre_tactic_id | eval mitre_tactic_label = replace(mitre_tactic_label,"\s","")]
| rename mitre_tactic_label as annotations.mitre_attack.mitre_tactic
| rename mitre_tactic_id as annotations.mitre_attack.mitre_tactic_id
| eval annotations._frameworks="mitre_attack"
| eval annotations._all='annotations.mitre_attack'
| `security_content_ctime(firstTime)`
| `security_content_ctime(lastTime)`
| eval impact = case(severity="informational", 20, severity="low", 40, severity="medium", 60, severity="high", 80, severity="critical", 100, true(), 0)
| eval risk_score=impact*80/100
| fields - impact, count
```

```
// Title:[AMP] Microsoft Defender Incidents - Ms Defender for Identity [SplunkPS]
// Description: Alerts generated from the Microsoft 365 Defender portal for Product Name "Microsoft Defender for Identity"
// Author: Sahil Sharma, ssharma7@paloaltonetworks.com
// Datasets: msft_graph_security_alerts_raw
// Date: 7/May/2024


config case_sensitive = false
| dataset = msft_graph_security_alerts_raw
| filter detectionSource = "MicrosoftDefenderforIdentity"
| filter severity != "informational"

| alter mitreTechniques = mitreTechniques -> []
| arrayexpand mitreTechniques
| alter mitreTechniques = replace(mitreTechniques, "\"", "")

| join (
    dataset = mitre_attack_lookup
    | alter mitre_tactic_label = replace(mitre_tactic_label, " ", "")
    | comp count() as mitre_stats by mitre_tactic_id, mitre_tactic_label
    | fields mitre_tactic_id, mitre_tactic_label as tactic_label
) as mitre_attack mitre_attack.tactic_label = category

| comp min(_time) as firstTime, max(_time) as lastTime, values(mitreTechniques) as mitre_technique_id, values(category) as mitre_tactic_label, values(mitre_tactic_id) as mitre_tactic_id, last(title) as title, last(productName) as productName, last(severity) as severity, last(description) as description, last(incidentWebUrl) as incidentWebUrl, last(status) as incidentStatus by incidentId, detectionSource

| fields firstTime, lastTime, mitre_technique_id, mitre_tactic_id, mitre_tactic_label, title, productName, description, incidentWebUrl, incidentStatus, incidentId, detectionSource
```

index=amp_msdefender_prod sourcetype=ms365:defender:incident severity != informational | dedup id | table _time, id, incident, incidentWebUrl, severity, status | rename status as incidentStatus, id as incidentId | join incidentId [search index=amp_msdefender_prod sourcetype=ms:defender:atp:alerts | table incidentId, productName] | eval urgency='severity' | `microsoft_defender_incident_filter`	// Title: [AMP] Microsoft Defender Incidents [SplunkPS] // Description: [AMP] Microsoft Defender Incidents [SplunkPS] // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: msft_graph_security_alerts_raw // Date: 07/May/2024 config case_sensitive = false | dataset = msdefendercollector_generic_alert_raw | filter severity != "informational" | dedup id | alter incidentStatus = status, incidentId = id | fields _time, incidentId, incidentStatus, incidentWebUrl, severity, displayName | join (dataset = msft_graph_security_alerts_raw | fields incidentId, productName) as alerts alerts.incidentId = incidentId // logic for macro `microsoft_defender_incident_filter` | filter productName not in ("Microsoft Defender for Endpoint", "Microsoft Data Loss Prevention", "Microsoft 365 Defender", "Microsoft Cloud App Security", "Microsoft Defender for Office 365", "Microsoft Defender for Identity") | fields incidentId, severity, incidentWebUrl, incidentStatus, productName, displayName, * /* // Title: [AMP] Microsoft Defender Incidents [SplunkPS] // Description: [AMP] Microsoft Defender Incidents [SplunkPS] // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: msft_graph_security_alerts_raw // Date: 07/May/2024 config case_sensitive = false | dataset = msft_graph_security_alerts_raw | fields incidentId, productName | join (dataset = msdefendercollector_generic_alert_raw | filter severity != "informational" | dedup id by asc _time | alter incidentStatus = status, incidentId = id | fields _time, incidentId, incidentStatus, incidentWebUrl, severity, displayName) as alerts alerts.incidentId = incidentId // logic for macro `microsoft_defender_incident_filter` | filter productName not in ("Microsoft Defender for Endpoint", "Microsoft Data Loss Prevention", "Microsoft 365 Defender", "Microsoft Cloud App Security", "Microsoft Defender for Office 365", "Microsoft Defender for Identity") | fields _time, incidentId, displayName,incidentWebUrl, severity, incidentStatus, productName, * */

	// Title: Endpoint - AMP - Malware Outbreak Detected - Rule // Description: This usecase will trigger the alerts whenever a similar signature malware attack on a multiple hosts // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: fireeye_hx_raw // Date: 01/Jul/2024 config case_sensitive = false | datamodel dataset = fireeye_hx_raw | replacenull xdm.alert.original_threat_name = "unknown" | filter xdm.alert.original_threat_name != "unknown"
| tstats summariesonly=true values(host) as host values(Malware_Attacks.file_hash) as file_hash values(Malware_Attacks.file_path) as file_path values(Malware_Attacks.act) as device_action values(Malware_Attacks.file_name) as file_name values(Malware_Attacks.category) as category values(Malware_Attacks.vendor_product) as vendor values(Malware_Attacks.action) as action values(Malware_Attacks.dest) as Malware_Attacks.dest dc(Malware_Attacks.dest) as Malware_Attacks.distinct_hosts values(Malware_Attacks.user) as Malware_Attacks.user from datamodel=Malware.Malware_Attacks where NOT Malware_Attacks.vendor_product IN (versa_fileFilterLog, "unknown*") by Malware_Attacks.signature |fillnull value="N/A" vendor |search vendor!="N/A" AND Malware_Attacks.signature!=unknown |rename Malware_Attacks.* as * |where distinct_hosts > 5 | eval urgency="high"	| comp count() as total_events, min(_time) as firstEventTime, max(_time) as lastEventTime, values(xdm.source.host.hostname) as host, count_distinct(xdm.source.host.hostname) as distinct_host, values(xdm.source.process.executable.md5) as file_hash, values(xdm.target.file.filename) as file_name, values(xdm.target.file.path) as file_path, values(xdm.observer.action) as device_action, values(xdm.alert.subcategory) as category, values(xdm.observer.vendor) as vendor, values(xdm.observer.product) as product, values(xdm.source.user.username) as user, values(xdm.source.ipv4) as src, values(xdm.event.description) as action by xdm.alert.original_threat_name | filter distinct_host > 5 | alter file_hash = arraystring(file_hash,", ")
Drill Down Query index=* tag=malware file_hash IN ($file_hash$) |stats values(file_hash) as file_hash values(file_path) as file_path values(file_name) as file_name values(act) as device_action values(action) as action values(category) as category values(vendor) as vendor by dhost	Drill Down Query config case_sensitive = false | datamodel dataset = fireeye_hx_raw | filter $file_hash contains xdm.source.process.executable.md5 | comp values(xdm.source.process.executable.md5) as file_hash, values(xdm.target.file.path) as file_path, values(xdm.target.file.filename) as file_name,values(xdm.observer.action) as device_action,values(xdm.event.description) as action, values(xdm.alert.subcategory) as category, values(xdm.observer.vendor) as vendor by xdm.target.host.hostname

```
config case_sensitive = false
| dataset = microsoft_windows_raw
| filter event_id in (104, 1102)

| filter message in ("*log file was cleared*", "*log was cleared*")
| filter computer_name not in ("IP-0A9C62*", "IP-0A9C63*", "IP-0AA21C*", "IP-0AA21D*", "IP-0AA21E*") // taking
computer name as dest
| filter host_name not in ("ip-10-156-98-*", "ip-10-156-99-*" , "ip-10-162-28-*", "ip-10-162-29-*", "ip-10-162-30-*")

// Filter for host names starting with IP-
| alter host_name_hex = if(host_name contains "IP-", replace(host_name,"IP-",""))

// Extract each hexadecimal octet
| alter hex1 = if(host_name contains "IP-", regextract(host_name_hex, "(\w\w)\w\w\w\w\w\w"))
| alter hex2 = if(host_name contains "IP-", regextract(host_name_hex, "\w\w(\w\w)\w\w\w\w"))
| alter hex3 = if(host_name contains "IP-", regextract(host_name_hex, "\w\w\w\w(\w\w)\w\w"))
| alter hex4 = if(host_name contains "IP-", regextract(host_name_hex, "\w\w\w\w\w\w(\w\w)"))

// Convert hex1 to decimal
| alter hex1_digit1 = arrayindex(regextract(arrayindex(hex1,0), "(\w)\w"),0)
| alter hex1_digit2 = arrayindex(regextract(arrayindex(hex1,0), "\w(\w)"),0)

| alter hex1_digit1 = if(hex1_digit1 = "A", replace(hex1_digit1, "A", "10"),
                    hex1_digit1 = "B", replace(hex1_digit1, "B", "11"),
                    hex1_digit1 = "C", replace(hex1_digit1, "C", "12"),
                    hex1_digit1 = "D", replace(hex1_digit1, "D", "13"),
                    hex1_digit1 = "E", replace(hex1_digit1, "E", "14"),
                    hex1_digit1 = "F", replace(hex1_digit1, "F", "15"), hex1_digit1),
      hex1_digit2 = if(hex1_digit2 = "A", replace(hex1_digit2, "A", "10"),
                    hex1_digit2 = "B", replace(hex1_digit2, "B", "11"),
                    hex1_digit2 = "C", replace(hex1_digit2, "C", "12"),
                    hex1_digit2 = "D", replace(hex1_digit2, "D", "13"),
                    hex1_digit2 = "E", replace(hex1_digit2, "E", "14"),
                    hex1_digit2 = "F", replace(hex1_digit2, "F", "15"), hex1_digit2)

| alter hex1_to_dec = add(multiply(to_number(hex1_digit1), 16), to_number(hex1_digit2))

//Convert hex2 to decimal
| alter hex2_digit1 = arrayindex(regextract(arrayindex(hex2,0), "(\w)\w"),0)
| alter hex2_digit2 = arrayindex(regextract(arrayindex(hex2,0), "\w(\w)"),0)

| alter hex2_digit1 = if(hex2_digit1 = "A", replace(hex2_digit1, "A", "10"),
                    hex2_digit1 = "B", replace(hex2_digit1, "B", "11"),
                    hex2_digit1 = "C", replace(hex2_digit1, "C", "12"),
                    hex2_digit1 = "D", replace(hex2_digit1, "D", "13"),
                    hex2_digit1 = "E", replace(hex2_digit1, "E", "14"),
                    hex2_digit1 = "F", replace(hex2_digit1, "F", "15"), hex2_digit1),
      hex2_digit2 = if(hex2_digit2 = "A", replace(hex2_digit2, "A", "10"),
                    hex2_digit2 = "B", replace(hex2_digit2, "B", "11"),
                    hex2_digit2 = "C", replace(hex2_digit2, "C", "12"),
                    hex2_digit2 = "D", replace(hex2_digit2, "D", "13"),
                    hex2_digit2 = "E", replace(hex2_digit2, "E", "14"),
                    hex2_digit2 = "F", replace(hex2_digit2, "F", "15"), hex2_digit2)

| alter hex2_to_dec = add(multiply(to_number(hex2_digit1), 16), to_number(hex2_digit2))

// //Convert hex3 to decimal
| alter hex3_digit1 = arrayindex(regextract(arrayindex(hex3,0), "(\w)\w"),0)
| alter hex3_digit2 = arrayindex(regextract(arrayindex(hex3,0), "\w(\w)"),0)

| alter hex3_digit1 = if(hex3_digit1 = "A", replace(hex3_digit1, "A", "10"),
                    hex3_digit1 = "B", replace(hex3_digit1, "B", "11"),
                    hex3_digit1 = "C", replace(hex3_digit1, "C", "12"),
                    hex3_digit1 = "D", replace(hex3_digit1, "D", "13"),
                    hex3_digit1 = "E", replace(hex3_digit1, "E", "14"),
                    hex3_digit1 = "F", replace(hex3_digit1, "F", "15"), hex3_digit1),
      hex3_digit2 = if(hex3_digit2 = "A", replace(hex3_digit2, "A", "10"),
                    hex3_digit2 = "B", replace(hex3_digit2, "B", "11"),
                    hex3_digit2 = "C", replace(hex3_digit2, "C", "12"),
                    hex3_digit2 = "D", replace(hex3_digit2, "D", "13"),
```

| tstats `security_content_summariesonly` count min(_time) as firstTime max(_time) as lastTime from datamodel=Endpoint.Processes where (Processes.process_name ="7z.exe" OR Processes.process_name = "7za.exe" OR Processes.original_file_name = "7z.exe" OR Processes.original_file_name = "7za.exe") AND (Processes.process="*\\C$*" OR Processes.process="*\\Admin$*" OR Processes.process="*\\IPC$*") by Processes.original_file_name Processes.parent_process_name Processes.parent_process Processes.process_name Processes.process Processes.parent_process_id Processes.process_id Processes.dest Processes.user | `drop_dm_object_name(Processes)` | `security_content_ctime(firstTime)` | `security_content_ctime(lastTime)` | `7zip_commandline_to_smb_share_path_filter`	// Title : ESCU - 7zip CommandLine To SMB Share Path - Rule // Description : This search is to detect a suspicious 7z process with commandline pointing to SMB network share. This technique was seen in CONTI LEAK tools where it use 7z to archive a sensitive files and place it in network share tmp folder. This search is a good hunting query that may give analyst a hint why specific user try to archive a file pointing to SMB user which is un usual. // Developed by : Aditya gour, agour@paloaltonetworks.com // datamodel/dataset : datamodel dataset = xdr_data // Tag : MSFT config case_sensitive = false |datamodel dataset = xdr_data | filter ((xdm.target.process.name in ("7z.exe", "7za.exe") or Xdm.source.process.executable.filename in ("7za.exe","7z.exe")) and (xdm.source.process.executable.path contains "*\C$*" or xdm.source.process.executable.path contains "*\Admin$*" or xdm.source.process.executable.path contains "*\IPC$*")) //or xdm.source.process.executable.path contains "$" // Uncomment to test the query logic | fields xdm.source.process.name, xdm.target.process.name, Xdm.source.process.executable.filename,xdm.target.process.executable.filename , xdm.target.process.executable.file_type , xdm.target.file.path , xdm.target.file.filename , xdm.target.file.file_type , xdm.target.file.extension , xdm.source.user.username , xdm.source.process.pid, xdm.source.process.thread_id , xdm.source.process.executable.path , xdm.source.process.command_line, xdm.source.host.hostname, xdm.source.host.ipv4_addresses , xdm.observer.type , xdm.observer.name , xdm.event.type ,xdm.event.operation_sub_type , xdm.event.operation ,* | comp min(_time) as firsttime, max(_time) as lastTime by xdm.target.process.name, Xdm.source.process.executable.filename, xdm.source.user.username, xdm.source.process.pid , xdm.source.process.parent_id

`cloudtrail` eventName=CreatePolicyVersion eventSource = iam.amazonaws.com errorCode = success "detail.service.additionalInfo.sample"!=true | spath input=requestParameters.policyDocument output=key_policy_statements path=Statement{} | mvexpand key_policy_statements | spath input=key_policy_statements output=key_policy_action_1 path=Action | where key_policy_action_1 = "*" | stats count min(_time) as firstTime max(_time) as lastTime values(key_policy_statements) as policy_added by eventName eventSource aws_account_id errorCode userAgent eventID awsRegion user user_arn | `security_content_ctime(firstTime)` | `security_content_ctime(lastTime)` |`aws_create_policy_version_to_allow_all_resources_filter`	// Title: ESCU - AWS Create Policy Version to allow all resources - Rule // Description: This search looks for AWS CloudTrail events where a user created a policy version that allows them to access any resource in their account. // Author: Anjali Verma, anjverma@paloaltonetworks.com // Reviewer: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: amazon_aws_raw // Date: 27/Jun/2024 config case_sensitive = false | dataset = amazon_aws_raw | filter _collector_name contains "Cloudtrail" // filtering for cloudtrail logs | filter eventName = "CreatePolicyVersion" | filter eventSource = "iam.amazonaws.com" | alter console_login = responseElements -> ConsoleLogin | alter errorCode = coalesce(errorCode, if(console_login = "Failure", "Failure", "Success"), "Success") | filter errorCode = "Success" //| filter "detail.service.additionalInfo.sample"!=true // field detail not found in field mapping document and sample data as well | alter key_policy_document = json_extract_scalar(requestParameters, "$.policyDocument") \| alter key_policy_statements = json_extract_array(key_policy_document, "$.Statement") | arrayexpand key_policy_statements | alter key_policy_action = json_extract_array(key_policy_statements, "$.Action") \| alter allow_all_resources = arraymap(key_policy_action, if("@element" ~= "^\"*\"$", true, false)) // filtering to check if all resources are allowed | filter allow_all_resources contains true | alter user_arn = userIdentity -> arn, aws_account_id = userIdentity -> accountId, type = userIdentity -> type, principalId = userIdentity -> principalId, accessKeyId = userIdentity -> accessKeyId | alter user = arrayindex(split(principalId, ":"), -1) | comp count() as total_events, min(_time) as firstTime, max(_time) as lastTime, values(key_policy_statements) as policy_added, values(principalId) as principalId, values(accessKeyId) as accessKeyId, values(_collector_name) as _collector_name, values(eventType) as eventType, values(requestID) as requestId, values(sourceIPAddress) as sourceIpAddress by eventName, eventSource, userAgent, eventID, awsRegion, user_arn, aws_account_id, errorCode, user | fields eventName, eventID, eventSource, policy_added, userAgent, awsRegion, user_arn, aws_account_id, _collector_name, firstTime, lastTime, total_events, accessKeyId, eventType, requestId, user, sourceIpAddress, principalId, errorCode

`cloudtrail` eventName = CreateAccessKey userAgent !=console.amazonaws.com errorCode = success NOT "detail.service.additionalInfo.sample"=true | eval match=if(match(userIdentity.userName,requestParameters.userName),1,0) | search match=0 | stats count min(_time) as firstTime max(_time) as lastTime by requestParameters.userName src eventName eventSource aws_account_id errorCode userAgent eventID awsRegion userIdentity.principalId user_arn | `security_content_ctime(firstTime)` | `security_content_ctime(lastTime)` |`aws_createaccesskey_filter`	// Title: ESCU - AWS CreateAccessKey - Rule // Description: This detection rule monitors for the creation of AWS Identity and Access Management (IAM) access keys. An IAM access key consists of an access key ID and secret access key, which are used to sign programmatic requests to AWS services. While IAM access keys can be legitimately used by developers and administrators for API access, their creation can also be indicative of malicious activity. Attackers who have gained unauthorized access to an AWS environment might create access keys as a means to establish persistence or to exfiltrate data through the APIs. Moreover, because access keys can be used to authenticate with AWS services without the need for further interaction, they can be particularly appealing for bad actors looking to operate under the radar. Consequently, it's important to vigilantly monitor and scrutinize access key creation events, especially if they are associated with unusual activity or are created by users who don't typically perform these actions. This hunting query identifies when a potentially compromised user creates a IAM access key for another user who may have higher privilleges, which can be a sign for privilege escalation. Hunting queries are designed to be executed manual during threat hunting. // Author: Anjali Verma, anjverma@paloaltonetworks.com // reviewer: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: amazon_aws_raw // Date: 09/July/2024 config case_sensitive = false | dataset = amazon_aws_raw | filter _collector_name contains "Cloudtrail" // filtering for cloudtrail logs | filter eventName = "CreateAccessKey" | filter userAgent != "console.amazonaws.com" | alter console_login = responseElements -> ConsoleLogin | alter status = coalesce(errorCode, if(console_login = "Failure", "Failure", "Success"), "Success") | filter status = "Success" //| filter "detail.service.additionalInfo.sample"!=true // field detail not found in field mapping document and sample data as well | alter session_issuer_username = json_extract_scalar(userIdentity ,"$.sessionContext.sessionIssuer.userName"), user_arn = json_extract_scalar(userIdentity ,"$.arn"), aws_account_id = json_extract_scalar(userIdentity ,"$.accountId"), request_userName = json_extract_scalar(requestParameters,"$.userName"), principalId = json_extract_scalar(userIdentity ,"$.principalId") | alter username = arrayindex(split(principalId, ":"), -1) | filter username != request_userName // filtering for username not equals to request_username | comp count() as event_count, min(_time) as firstTime, max(_time) as lastTime, values(username) as username, values(eventType) as eventType, values(requestID) as requestId by request_userName, sourceIPAddress, eventName, eventSource, aws_account_id, userAgent, eventID, awsRegion, principalId, user_arn, status | fields firstTime, lastTime, username, request_userName, principalId, user_arn, sourceIPAddress, eventName, eventSource, aws_account_id, status as errorCode, userAgent, eventID, awsRegion, eventType, requestId

requestParameters.roleName responseElements.role.roleName responseElements.role.createDate | `aws_detect_sts_assume_role_abuse_filter`	// Title: ESCU - aws detect sts assume role abuse - Rule config case_sensitive = false | dataset = amazon_aws_raw | filter _collector_name contains "Cloudtrail" // filtering for cloudtrail logs | alter user_type = json_extract_scalar(userIdentity ,"$.type") \| alter user_type = "AssumedRole" \| alter type = json_extract_scalar(userIdentity ,"$.sessionContext.sessionIssuer.type") | filter type = "Role" | alter user_arn = json_extract_scalar(userIdentity ,"$.arn") \| alter accessKeyId = json_extract_scalar(userIdentity ,"$.accessKeyId") | alter status = json_extract_scalar(requestParameters ,"$.executionResult.status") \| alter request_roleName = json_extract_scalar(requestParameters ,"$.roleName") // responseElements.roleName field not found in dataset | alter response_roleName = json_extract_scalar(responseElements ,"$.role.roleName") // responseElements.role.roleName field not found in dataset \| alter response_createDate = json_extract_scalar(responseElements ,"$.responseElements.role.createDate") // responseElements.role.createDate field not found in dataset | fields sourceIPAddress,user_arn,userAgent,accessKeyId as user_access_key, action ,status,request_roleName, response_roleName ,response_createDate , *
`cloudtrail` eventName=CreateKey OR eventName=PutKeyPolicy "detail.service.additionalInfo.sample"!=true | spath input=requestParameters.policy output=key_policy_statements path=Statement{} | mvexpand key_policy_statements | spath input=key_policy_statements output=key_policy_action_1 path=Action | spath input=key_policy_statements output=key_policy_action_2 path=Action{} | eval key_policy_action=mvappend(key_policy_action_1, key_policy_action_2) | spath input=key_policy_statements output=key_policy_principal path=Principal.AWS | search key_policy_action="kms:Encrypt" AND key_policy_principal="*" | stats count min(_time) as firstTime max(_time) as lastTime by eventName eventSource eventID awsRegion userIdentity.principalId | `security_content_ctime(firstTime)`| `security_content_ctime(lastTime)` |`aws_detect_users_creating_keys_with_encrypt_policy_without_mfa_filter`	config case_sensitive = false | dataset = amazon_aws_raw | filter _collector_name contains "Cloudtrail" | filter eventName = "CreateKey" or eventName = "PutKeyPolicy" //| filter "detail.service.additionalInfo.sample"!=true | alter key_policy = json_extract_scalar(requestParameters, "$.policy") \| alter key_policy_statements = json_extract_array(key_policy, "$.Statement") | arrayexpand key_policy_statements | alter key_policy_action = json_extract(key_policy_statements, "$.Action"), policy_principal_aws = json_extract(key_policy_statements, "$.Principal.AWS"), principalId = json_extract_scalar(userIdentity ,"$.principalId"), accountId = json_extract_scalar(userIdentity ,"$.accountId"), arn = json_extract_scalar(userIdentity ,"$.arn"), accessKeyId = userIdentity -> accessKeyId | alter key_policy_action = replace(key_policy_action, "\"", ""), policy_principal_aws = replace(policy_principal_aws, "\"", "") | filter key_policy_action contains "kms:Encrypt" | filter policy_principal_aws = "*" | comp count() as total_events, min(_time) as firstTime, max(_time) as lastTime, values(accountId) as accountId, values(arn) as arn, values(accessKeyId) as accessKeyId, values(_collector_name) as collector_name, values(eventType) as eventType, values(requestID) as requestId, values(sourceIPAddress) as sourceIpAddress, values(userAgent) as userAgent by eventName, eventSource, eventID, awsRegion, principalId, policy_principal_aws,key_policy_action | fields eventName, eventSource, eventID, awsRegion, principalId,total_events as count,firstTime, lastTime, policy_principal_aws, key_policy_action, accountId, arn, eventType, requestId, sourceIpAddress, userAgent

`cloudtrail` (errorCode=MalformedPolicyDocumentException) status=failure (userAgent!=*.amazonaws.com) "detail.service.additionalInfo.sample"!=true | stats count min(_time) as firstTime max(_time) as lastTime values(requestParameters.policyName) as policy_name by src eventName eventSource aws_account_id errorCode requestParameters.policyDocument userAgent eventID awsRegion userIdentity.principalId user_arn | where count >= 2 | `security_content_ctime(firstTime)` | `security_content_ctime(lastTime)` | `aws_iam_assume_role_policy_brute_force_filter`	// Title: ESCU - AWS IAM Assume Role Policy Brute Force - Rule dataset = amazon_aws_raw | filter _collector_name contains "Cloudtrail" // filtering for cloudtrail logs | filter errorCode = "MalformedPolicyDocumentException" // "MalformedPolicyDocumentException" errocode not found | alter status = json_extract_scalar(requestParameters ,"$.executionResult.status") \| filter status = "failure" \| filter userAgent != "*.amazonaws.com" \| alter policyName = json_extract_scalar(requestParameters,"$.policyName") | alter policyDocument = json_extract_scalar(requestParameters, "$.policyDocument") \| alter username = json_extract_scalar(userIdentity ,"$.principalId") | filter policyDocument not in (null,"") | filter policyName not in (null,"") | alter user_arn = json_extract_scalar(userIdentity ,"$.arn") \| alter aws_account_id = json_extract_scalar(userIdentity ,"$.accountId") | comp count() as total_events , min(_time) as firstTime, max(_time) as lastTime,values(policyName) as policy_name , values(errorMessage) as errorMessage by sourceIPAddress , eventName ,eventSource,aws_account_id,errorCode, policyDocument,userAgent,eventID, awsRegion ,username,user_arn //,errorMessage | filter total_events >= 2 | fields sourceIPAddress as src ,eventName ,eventSource,aws_account_id,errorCode ,policyDocument,userAgent, eventID, awsRegion,username ,user_arn, total_events as count, firstTime,lastTime,policy_name,errorMessage
`cloudtrail` eventName=DeletePolicy (userAgent!=*.amazonaws.com) "detail.service.additionalInfo.sample"!=true | stats count min(_time) as firstTime max(_time) as lastTime values(requestParameters.policyArn) as policyArn by src user_arn eventName eventSource aws_account_id errorCode errorMessage userAgent eventID awsRegion userIdentity.principalId | `security_content_ctime(firstTime)` | `security_content_ctime(lastTime)` | `aws_iam_delete_policy_filter`	// Title: ESCU - AWS IAM Delete Policy - Rule // Description: The following detection identifies when a policy is deleted on AWS. This does not identify whether successful or failed, but the error messages tell a story of suspicious attempts. There is a specific process to follow when deleting a policy. First, detach the policy from all users, groups, and roles that the policy is attached to, using DetachUserPolicy , DetachGroupPolicy , or DetachRolePolicy. // Author: Anjali Verma, anjverma@paloaltonetworks.com // Datasets: amazon_aws_raw // Date: 09/July/2024 dataset = amazon_aws_raw | filter _collector_name contains "Cloudtrail" // filtering for cloudtrail logs | filter eventName = "DeletePolicy" | filter userAgent != "*.amazonaws.com" | alter console_login = responseElements -> ConsoleLogin | alter errorCode = coalesce(errorCode, if(console_login = "Failure", "Failure", "Success"), "Success") | alter policyArn = json_extract_scalar(requestParameters,"$.policyArn") \| alter username = json_extract_scalar(userIdentity ,"$.principalId") | alter user_arn = json_extract_scalar(userIdentity ,"$.arn") \| alter aws_account_id = json_extract_scalar(userIdentity ,"$.accountId") | comp count() as total_events , min(_time) as firstTime, max(_time) as lastTime,values(policyArn) as policyArn by sourceIPAddress,user_arn , eventName ,eventSource,aws_account_id,errorCode,errorMessage,userAgent,eventID, awsRegion ,username | fields firstTime,lastTime, sourceIPAddress , eventName ,eventSource,aws_account_id,errorCode , userAgent, eventID, awsRegion ,username,user_arn

`cloudtrail` eventSource=iam.amazonaws.com eventName=DeleteGroup errorCode=success (userAgent!=*.amazonaws.com) "detail.service.additionalInfo.sample"!=true | stats count min(_time) as firstTime max(_time) as lastTime values(requestParameters.groupName) as group_deleted by src eventName eventSource errorCode user_agent awsRegion userIdentity.principalId user_arn | `security_content_ctime(firstTime)` | `security_content_ctime(lastTime)` | `aws_iam_successful_group_deletion_filter`	// Title: ESCU - AWS IAM Successful Group Deletion - Rule // Description: The following query uses IAM events to track the success of a group being deleted on AWS. This is typically not indicative of malicious behavior, but a precurser to additional events thay may unfold. Review parallel IAM events - recently added users, new groups and so forth. Inversely, review failed attempts in a similar manner. // Author: Anjali Verma, anjverma@paloaltonetworks.com // Datasets: amazon_aws_raw // Date: 09/July/2024 dataset = amazon_aws_raw | filter _collector_name contains "Cloudtrail" // filtering for cloudtrail logs | filter eventSource = "iam.amazonaws.com" | filter eventName = "DeleteGroup" // "DeleteGroup" not found in eventName | alter console_login = responseElements -> ConsoleLogin | alter errorCode = coalesce(errorCode, if(console_login = "Failure", "Failure", "Success"), "Success") | filter errorCode = "Success" | filter userAgent != "*.amazonaws.com" | alter username = json_extract_scalar(userIdentity ,"$.principalId") \| alter user_arn = json_extract_scalar(userIdentity ,"$.arn") | alter aws_account_id = json_extract_scalar(userIdentity ,"$.accountId") \| alter groupName = json_extract_scalar(requestParameters,"$.groupName") | comp count() as total_events, min(_time) as firstTime, max(_time) as lastTime,values(groupName) as group_deleted by sourceIPAddress,eventName,eventSource,errorCode,userAgent,awsRegion,username,eventType,user_arn, aws_account_id, eventID | fields firstTime,lastTime,sourceIPAddress,eventName,eventSource,errorCode,userAgent,awsRegion,username, eventType,user_arn,aws_account_id, eventID,group_deleted
`cloudtrail` eventName=UpdateSAMLProvider "detail.service.additionalInfo.sample"!=true | stats count min(_time) as firstTime max(_time) as lastTime by eventType eventName requestParameters. sAMLProviderArn userIdentity.sessionContext.sessionIssuer.arn sourceIPAddress userIdentity. accessKeyId userIdentity.principalId | `security_content_ctime(firstTime)`| `security_content_ctime (lastTime)` |`aws_saml_update_identity_provider_filter`	// Title: ESCU - AWS SAML Update identity provider - Rule // Description: This search provides detection of updates to SAML provider in AWS. Updates to SAML provider need to be monitored closely as they may indicate possible perimeter compromise of federated credentials, or backdoor access from another cloud provider set by attacker. // Author: Anjali Verma, anjverma@paloaltonetworks.com // Datasets: amazon_aws_raw // Date: 09/July/2024 dataset = amazon_aws_raw | filter _collector_name contains "Cloudtrail" // filtering for cloudtrail logs | filter eventName = "UpdateSAMLProvider" | alter policyArn = json_extract_scalar(requestParameters,"$.policyArn") \| alter sAMLProviderArn = json_extract_scalar(requestParameters,"$.sAMLProviderArn") | alter username = json_extract_scalar(userIdentity ,"$.principalId") \| alter aws_account_id = json_extract_scalar(userIdentity ,"$.accountId") | alter accessKeyId = json_extract_scalar(userIdentity ,"$.accessKeyId") \| alter user_arn = json_extract_scalar(userIdentity ,"$.sessionContext.sessionIssuer.arn") | comp count() as total_events, min(_time) as firstTime, max(_time) as lastTime by eventType,eventName, sAMLProviderArn,user_arn ,sourceIPAddress,accessKeyId,username | fields firstTime,lastTime,eventType,eventName, sAMLProviderArn,user_arn ,sourceIPAddress,accessKeyId,username, user_arn

`cloudtrail` eventName=SetDefaultPolicyVersion eventSource = iam.amazonaws.com "detail.service.additionalInfo.sample"!=true | stats count min(_time) as firstTime max(_time) as lastTime values (requestParameters.policyArn) as policy_arn by src requestParameters.versionId eventName eventSource aws_account_id errorCode userAgent eventID awsRegion userIdentity.principalId user_arn | `security_content_ctime(firstTime)` | `security_content_ctime(lastTime)` | `aws_setdefaultpolicyversion_filter`	// Title: ESCU - AWS SetDefaultPolicyVersion - Rule // Description: This search looks for AWS CloudTrail events where a user has set a default policy versions. Attackers have been know to use this technique for Privilege Escalation in case the previous versions of the policy had permissions to access more resources than the current version of the policy. // Author: Anjali Verma, anjverma@paloaltonetworks.com // Datasets: amazon_aws_raw // Date: 09/July/2024 dataset = amazon_aws_raw | filter _collector_name contains "Cloudtrail" // filtering for cloudtrail logs | filter eventName = "SetDefaultPolicyVersion" // "SetDefaultPolicyVersion" event not found | filter eventSource = "iam.amazonaws.com" | alter console_login = responseElements -> ConsoleLogin | alter errorCode = coalesce(errorCode, if(console_login = "Failure", "Failure", "Success"), "Success") | alter policyArn = json_extract_scalar(requestParameters,"$.policyArn") \| alter versionId = json_extract_scalar(requestParameters,"$.versionId") | alter username = json_extract_scalar(userIdentity ,"$.principalId") \| alter user_arn = json_extract_scalar(userIdentity ,"$.arn") | alter aws_account_id = json_extract_scalar(userIdentity ,"$.accountId") \| comp count() as total_events, min(_time) as firstTime, max(_time) as lastTime,values(policyArn) as policy_arn by sourceIPAddress,versionId, eventName, eventSource,aws_account_id, errorCode, userAgent, eventID, awsRegion, username, user_arn \| fields firstTime,lastTime,sourceIPAddress,userAgent,eventName,eventSource,errorCode,eventID,awsRegion, username,user_arn,versionId,aws_account_id Udpate query: Waiting for Demo event dataset = amazon_aws_raw \| filter _collector_name contains "Cloudtrail" // filtering for cloudtrail logs //\| filter eventName = "SetDefaultPolicyVersion" // "SetDefaultPolicyVersion" event not found \| filter eventSource = "iam.amazonaws.com" \| alter principalID = json_extract_scalar(userIdentity ,"$.principalId") | alter user_arn = json_extract_scalar(userIdentity ,"$.arn") \| alter policyArn = json_extract_scalar(requestParameters,"$.policyArn") | fields * /* | alter console_login = responseElements -> ConsoleLogin | alter errorCode = coalesce(errorCode, if(console_login = "Failure", "Failure", "Success"), "Success") | alter policyArn = json_extract_scalar(requestParameters,"$.policyArn") \| alter versionId = json_extract_scalar(requestParameters,"$.versionId") | alter username = json_extract_scalar(userIdentity ,"$.principalId") \| alter user_arn = json_extract_scalar(userIdentity ,"$.arn") | alter aws_account_id = json_extract_scalar(userIdentity ,"$.accountId")*/ | comp count() as total_events, min(_time) as firstTime, max(_time) as lastTime,values(policyArn) as policy_arn by sourceIPAddress, srcaddr ,eventName, eventSource,account_id , errorCode, userAgent, eventID, awsRegion, principalID , user_arn //versionId | fields sourceIPAddress,srcaddr,eventName,eventSource,account_id , errorCode,userAgent , eventID,awsRegion, principalID , user_arn //,version , apiVersion , eventVersion , //,username,user_arn,versionId,aws_account_id,firstTime, lastTime,account_id as aws_account_id,

	// Title: ESCU - Detect AWS Console Login by New User - Rule // Description: **WARNING**, this detection is marked **EXPERIMENTAL** by the Splunk Threat Research Team. This means that the detection has been manually tested but we do not have the associated attack data to perform automated testing or cannot share this attack dataset due to its sensitive nature. If you have any questions feel free to email us at: research@splunk.com. This search looks for AWS CloudTrail events wherein a console login event by a user was recorded within the last hour, then compares the event to a lookup file of previously seen users (by ARN values) who have logged into the console. The alert is fired if the user has logged into the console for the first time within the last hour // Author: Anjali Verma, anjverma@paloaltonetworks.com // Reviewer: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: microsoft_windows_raw // Date: 08/July/2024 config case_sensitive = false | dataset = amazon_aws_raw | filter eventName = "ConsoleLogin" // Extracting username | alter principalID = json_extract_scalar(userIdentity ,"$.principalId"), user = json_extract_scalar(userIdentity ,"$.userName"), user_arn = json_extract_scalar(userIdentity, "$.arn"), aws_account_id = json_extract_scalar(userIdentity , "$.accountId") | alter principalID = arrayindex(split(principalID, ":"), 1) | alter username = coalesce(principalID, user) | comp min(_time) as firstTime, max(_time) as lastTime, values(aws_account_id) as aws_account_id, values(user_arn) as user_arn, values(sourceIPAddress) as sourceIPAddress, values(userAgent) as userAgent, values(awsRegion) as awsRegion by username, eventCategory, eventName, eventSource, eventType | join type = left (dataset = previously_seen_users_console_logins | alter timestamp = to_timestamp(to_integer(firstTime), "SECONDS") | comp min(timestamp) as earliestSeen by user | fields earliestSeen, user) as prev_logins prev_logins.user = username or to_string(user_arn) contains prev_logins.user | alter currentTime = current_time() | alter timeDiffHour = divide(timestamp_diff(currentTime, earliestSeen, "MINUTE"), 60) // calculate time difference in Hours | alter userStatus = if(timeDiffHour <= 24 or earliestSeen = null, "First Time Logging into AWS Console", "Previously Seen User") | filter userStatus = "First Time Logging into AWS Console" | fields firstTime, lastTime, eventCategory, eventType, eventName, eventSource, sourceIPAddress, userAgent, user_arn, username, earliestseen, userStatus, aws_account_id, awsRegion
| tstats earliest(_time) as firstTime latest(_time) as lastTime from datamodel=Authentication where Authentication.signature=ConsoleLogin by Authentication.user | `drop_dm_object_name (Authentication)` | join user type=outer [| inputlookup previously_seen_users_console_logins | stats min(firstTime) as earliestseen by user] | eval userStatus=if(earliestseen >= relative_time(now(), "-24 h@h") OR isnull(earliestseen), "First Time Logging into AWS Console", "Previously Seen User") | where userStatus="First Time Logging into AWS Console" | `security_content_ctime(firstTime)` | `security_content_ctime(lastTime)` | `detect_aws_console_login_by_new_user_filter`	

	// Title: ESCU - Detect AWS Console Login by User from New City - Rule // Description: This search looks for AWS CloudTrail events wherein a console login event by a user was recorded within the last hour, then compares the event to a lookup file of previously seen users (by ARN values) who have logged into the console. The alert is fired if the user has logged into the console for the first time within the last hour // Author: Anjali Verma, anjverma@paloaltonetworks.com // Reviewer: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: microsoft_windows_raw // Date: 08/July/2024 config case_sensitive = false | dataset = amazon_aws_raw | filter eventName = "ConsoleLogin" // Extracting username | alter principalID = json_extract_scalar(userIdentity ,"$.principalId"), user = json_extract_scalar(userIdentity ,"$.userName"), user_arn = json_extract_scalar(userIdentity, "$.arn"), aws_account_id = json_extract_scalar(userIdentity , "$.accountId") | alter principalID = arrayindex(split(principalID, ":"), 1) | alter username = coalesce(principalID, user) | comp min(_time) as firstTime, max(_time) as lastTime, values(aws_account_id) as aws_account_id, values(user_arn) as user_arn, values(userAgent) as userAgent, values(awsRegion) as awsRegion by username, sourceIPAddress, eventCategory, eventName, eventSource, eventType | iploc sourceIPAddress loc_city as justSeenCity | join type = left (dataset = previously_seen_users_console_logins | alter timestamp = to_timestamp(to_integer(firstTime), "SECONDS") | comp min(timestamp) as earliestSeen by user, City | fields earliestSeen, user, City as previouslySeenCity) as prev_logins prev_logins.user = username or to_string(user_arn) contains prev_logins.user | alter currentTime = current_time() | alter timeDiffHour = divide(timestamp_diff(currentTime, earliestSeen, "MINUTE"), 60) // calculate time difference in Hours | alter userCity = if(timeDiffHour <= 24, "New City", "Previously Seen City") | filter userCity = "New City" | fields firstTime, lastTime, eventCategory, eventType, eventName, eventSource, sourceIPAddress, userAgent, user_arn, username, earliestseen, aws_account_id, awsRegion, sourceIPAddress, justSeenCity, previouslySeenCity, userCity
| tstats earliest(_time) as firstTime latest(_time) as lastTime from datamodel=Authentication where Authentication.signature=ConsoleLogin by Authentication.user Authentication.src | iplocation Authentication.src | `drop_dm_object_name(Authentication)` | rename City as justSeenCity | table firstTime lastTime user justSeenCity | join user type=outer [| inputlookup previously_seen_users_console_logins | rename City as previouslySeenCity | stats min(firstTime) AS earliestseen by user previouslySeenCity | fields earliestseen user previouslySeenCity] | eval userCity=if(firstTime >= relative_time(now(), "-24h@h"), "New City","Previously Seen City") | where userCity = "New City" | `security_content_ctime(firstTime)` | `security_content_ctime(lastTime)` | table firstTime lastTime user previouslySeenCity justSeenCity userCity | `detect_aws_console_login_by_user_from_new_city_filter`	

	// Title: ESCU - Detect AWS Console Login by User from New Country - Rule // Description: This search looks for AWS CloudTrail events wherein a console login event by a user was recorded within the last hour, then compares the event to a lookup file of previously seen users (by ARN values) who have logged into the console. The alert is fired if the user has logged into the console for the first time within the last hour // Author: Anjali Verma, anjverma@paloaltonetworks.com // Reviewer: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: microsoft_windows_raw // Date: 08/July/2024 config case_sensitive = false | dataset = amazon_aws_raw | filter eventName = "ConsoleLogin" // Extracting username | alter principalID = json_extract_scalar(userIdentity ,"$.principalId"), user = json_extract_scalar(userIdentity ,"$.userName"), user_arn = json_extract_scalar(userIdentity, "$.arn"), aws_account_id = json_extract_scalar(userIdentity , "$.accountId") | alter principalID = arrayindex(split(principalID, ":"), 1) | alter username = coalesce(principalID, user) | comp min(_time) as firstTime, max(_time) as lastTime, values(aws_account_id) as aws_account_id, values(user_arn) as user_arn, values(userAgent) as userAgent, values(awsRegion) as awsRegion by username, sourceIPAddress, eventCategory, eventName, eventSource, eventType | iploc sourceIPAddress loc_country as justSeenCountry | join type = left (dataset = previously_seen_users_console_logins | alter timestamp = to_timestamp(to_integer(firstTime), "SECONDS") | comp min(timestamp) as earliestSeen by user, Country | fields earliestSeen, user, Country as previouslySeenCountry) as prev_logins prev_logins.user = username or to_string(user_arn) contains prev_logins.user | alter currentTime = current_time() | alter timeDiffHour = divide(timestamp_diff(currentTime, earliestSeen, "MINUTE"), 60) // calculate time difference in Hours | alter userCountry = if(timeDiffHour <= 24, "New Country", "Previously Seen Country") | filter userCountry = "New Country" | fields firstTime, lastTime, eventCategory, eventType, eventName, eventSource, sourceIPAddress, userAgent, user_arn, username, earliestseen, aws_account_id, awsRegion, sourceIPAddress, justSeenCountry, previouslySeenCountry, userCountry
| tstats earliest(_time) as firstTime latest(_time) as lastTime from datamodel=Authentication where Authentication.signature=ConsoleLogin by Authentication.user Authentication.src | iplocation Authentication.src | `drop_dm_object_name(Authentication)` | rename Country as justSeenCountry | table firstTime lastTime user justSeenCountry | join user type=outer [| inputlookup previously_seen_users_console_logins | rename Country as previouslySeenCountry | stats min(firstTime) AS earliestseen by user previouslySeenCountry | fields earliestseen user previouslySeenCountry] | eval userCountry=if(firstTime >= relative_time(now(), "-24h@h"), "New Country","Previously Seen Country") | where userCountry = "New Country" | `security_content_ctime(firstTime)` | `security_content_ctime(lastTime)` | table firstTime lastTime user previouslySeenCountry justSeenCountry userCountry | `detect_aws_console_login_by_user_from_new_country_filter`	

| tstats earliest(_time) as firstTime latest(_time) as lastTime from datamodel=Authentication where Authentication.signature=ConsoleLogin by Authentication.user Authentication.src | iplocation Authentication.src | `drop_dm_object_name(Authentication)` | rename Region as justSeenRegion | table firstTime lastTime user justSeenRegion | join user type=outer [| inputlookup previously_seen_users_console_logins | rename Region as previouslySeenRegion | stats min(firstTime) AS earliestseen by user previouslySeenRegion | fields earliestseen user previouslySeenRegion] | eval userRegion=if(firstTime >= relative_time(now(), "-24h@h"), "New Region","Previously Seen Region") | where userRegion= "New Region" | `security_content_ctime(firstTime)` | `security_content_ctime(lastTime)` | table firstTime lastTime user previouslySeenRegion justSeenRegion userRegion | `detect_aws_console_login_by_user_from_new_region_filter`	// Title: ESCU - Detect AWS Console Login by User from New Region - Rule // Description: This search looks for AWS CloudTrail events wherein a console login event by a user was recorded within the last hour, then compares the event to a lookup file of previously seen users (by ARN values) who have logged into the console. The alert is fired if the user has logged into the console for the first time within the last hour // Author: Anjali Verma, anjverma@paloaltonetworks.com // Reviewer: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: microsoft_windows_raw // Date: 08/July/2024 config case_sensitive = false | dataset = amazon_aws_raw | filter eventName = "ConsoleLogin" // Extracting username | alter principalID = json_extract_scalar(userIdentity ,"$.principalId"), user = json_extract_scalar(userIdentity ,"$.userName"), user_arn = json_extract_scalar(userIdentity, "$.arn"), aws_account_id = json_extract_scalar(userIdentity , "$.accountId") | alter principalID = arrayindex(split(principalID, ":"), 1) | alter username = coalesce(principalID, user) | comp min(_time) as firstTime, max(_time) as lastTime, values(aws_account_id) as aws_account_id, values(user_arn) as user_arn, values(userAgent) as userAgent, values(awsRegion) as awsRegion by username, sourceIPAddress, eventCategory, eventName, eventSource, eventType | iploc sourceIPAddress loc_region as justSeenRegion | join type = left (dataset = previously_seen_users_console_logins | alter timestamp = to_timestamp(to_integer(firstTime), "SECONDS") | comp min(timestamp) as earliestSeen by user, Region | fields earliestSeen, user, Region as previouslySeenRegion) as prev_logins prev_logins.user = username or to_string(user_arn) contains prev_logins.user | alter currentTime = current_time() | alter timeDiffHour = divide(timestamp_diff(currentTime, earliestSeen, "MINUTE"), 60) // calculate time difference in Hours | alter userRegion = if(timeDiffHour <= 24, "New Region", "Previously Seen Region") | filter userRegion = "New Region" | fields firstTime, lastTime, eventCategory, eventType, eventName, eventSource, sourceIPAddress, userAgent, user_arn, username, earliestseen, aws_account_id, awsRegion, sourceIPAddress, justSeenRegion, previouslySeenRegion, userRegion
`wineventlog_security` EventCode=4624 OR EventCode=4742 TargetUserName="ANONYMOUS LOGON" LogonType=3 | stats count values(host) as host, values(TargetDomainName) as Domain, values(user) as user | `detect_computer_changed_with_anonymous_account_filter`	// Title : ESCU - Detect Computer Changed with Anonymous Account - Rule // Description : This search looks for Event Code 4742 (Computer Change) or EventCode 4624 (An account was successfully logged on) with an anonymous account. // Developed by : Aditya gour, agour@paloaltonetworks.com // dataset = microsoft_windows_raw config case_sensitive = false | datamodel dataset = microsoft_windows_raw | filter xdm.observer.type = "Microsoft-Windows-Security-Auditing" | filter xdm.event.id in ("4624", "4742") | filter xdm.logon.type = "NETWORK" | filter xdm.target.user.username contains "ANONYMOUS LOGON" | fields xdm.event.id as event_code, xdm.event.original_event_type as signature, xdm.event.description as message, xdm.logon.type as logon_type, xdm.source.host.hostname as host, xdm.source.ipv4 as src, xdm.target.user.username as target_username, xdm.source.user.username as source_username, xdm.target.user.domain as targetUserDomain | comp count() as total_count, values(host) as host, values(targetUserDomain) as targetUserDomain, values (source_username) as user, values(src) as src, values(signature) as signature by target_username

`wineventlog_security` EventCode=4769 Service_Name!="*$" (Ticket_Options=0x40810000 OR Ticket_Options=0x40800000 OR Ticket_Options=0x40810010) Ticket_Encryption_Type=0x17 | stats count min(_time) as firstTime max(_time) as lastTime by dest, service, service_id, Ticket_Encryption_Type, Ticket_Options | `security_content_ctime(lastTime)` | `security_content_ctime(firstTime)` | `kerberoasting_spn_request_with_rc4_encryption_filter`	// Title: ESCU - Kerberoasting spn request with RC4 encryption - Rule dataset = microsoft_windows_raw | filter event_id = 4769 | alter Service_Name = json_extract_scalar(event_data ,"$.ServiceName"), service_id = json_extract_scalar(event_data ,"$.ServiceSid"), Ticket_Options = json_extract_scalar(event_data ,"$.TicketOptions"), Ticket_Encryption_Type = json_extract_scalar(event_data ,"$.Status"), IpAddress = json_extract_scalar(event_data ,"$.IpAddress"), Account_Name = json_extract_scalar(event_data, "$.TargetUserName"), Account_Domain = json_extract_scalar(event_data, "$.TargetDomainName") \| alter src = arrayindex(regextract(IpAddress, "\b(?:[0-9]{1,3}\.){3}[0-9]{1,3}\b"), 0) \| filter Service_Name != "*$" | filter Ticket_Options in ("0x40810000","0x40800000","0x40810010") | filter Ticket_Encryption_Type = "0x17" | fields event_id, event_data, message, service_name, service_id, src, Ticket_Options, Ticket_Encryption_Type, computer_name as dest, Account_Domain, Account_Name, * | comp count() as total_events, min(_time) as firstEventTime, max(_time) as lastEventtime, values(Account_Domain) as Account_Domain, values(Account_Name) as Account_Name, values(src) as src, values(event_action) as event_action, values(event_result) as event_result by dest, service_id, Service_Name, Ticket_Encryption_Type, Ticket_Options, event_id
| tstats `security_content_summariesonly` count min(_time) as firstTime max(_time) as lastTime values(Filesystem.user) as user values(Filesystem.dest) as dest values(Filesystem.file_path) as file_path from datamodel=Endpoint.Filesystem where (Filesystem.file_path="*\\Windows\\System32\\sethc.exe* OR Filesystem.file_path="*\\Windows\\System32\\utilman.exe* OR Filesystem.file_path="*\\Windows\\System32\\osk.exe* OR Filesystem.file_path="*\\Windows\\System32\\Magnify.exe* OR Filesystem.file_path="*\\Windows\\System32\\Narrator.exe* OR Filesystem.file_path="*\\Windows\\System32\\DisplaySwitch.exe* OR Filesystem.file_path="*\\Windows\\System32\\AtBroker.exe*) by Filesystem.file_name Filesystem.dest | `drop_dm_object_name(Filesystem)` | `security_content_ctime(lastTime)` | `security_content_ctime(firstTime)` | `overwriting_accessibility_binaries_filter`	config case_sensitive = false |datamodel dataset = xdr_data | filter xdm.source.host.os_family = XDM_CONST.OS_FAMILY_WINDOWS | filter xdm.target.file.path in ("*\Windows\System32\sethc.exe*", "*\Windows\System32\utilman.exe*", "*\Windows\System32\osk.exe*", "*\Windows\System32\Magnify.exe*", "*\Windows\System32\Narrator.exe*", "*\Windows\System32\DisplaySwitch.exe*", "*\Windows\System32\AtBroker.exe*") | fields xdm.target.file.path, _time, xdm.target.file.filename, xdm.source.user.username , xdm.source.ipv4, xdm.source.host.hostname, xdm.target.ipv4, xdm.target.host.hostname | comp min(_time) as firsttime, max(_time) as lastTime, values(xdm.source.user.username) as users, values(xdm.source.host.hostname) as source_hostname, values(xdm.target.file.path) as file_path by xdm.target.file.filename
| tstats `security_content_summariesonly` dc(All_Changes.result_id) as result_count values(All_Changes.result_id) as result_id count min(_time) as firstTime max(_time) as lastTime from datamodel=Change where All_Changes.result_id=4720 OR All_Changes.result_id=4726 by _time span=4h All_Changes.user All_Changes.dest | `security_content_ctime(lastTime)` | `security_content_ctime(firstTime)` |where result_count>1 | `drop_dm_object_name("All_Changes")` | search result_id = 4720 result_id=4726 | transaction user connected=false maxspan=240m | table firstTime lastTime result_count user dest result_id | `short_lived_windows_accounts_filter`	config case_sensitive = false | datamodel dataset = microsoft_windows_raw | filter xdm.event.id in ("4720", "4726") | bin _time span = 4h | comp count_distinct(xdm.event.id) as results_count, values(xdm.event.id) as results_list, min(_time) as firstTime, max(_time) as lastTime, values(xdm.source.user.username) as subject_user, values(xdm.event.original_event_type) as signature by xdm.target.user.username, xdm.source.host.hostname, _time | filter results_count > 1 // | filter timestamp_diff(lastTime, firstTime, "MINUTE") >= 0 and timestamp_diff(lastTime, firstTime, "MINUTE") <= 240 | arrayexpand results_list | transaction xdm.target.user.username span = 240m

	// Title: Identity - [AD} Detect Change in USER_NOT_DELEGATED flag [AMP] - Rule
	config case_sensitive = false | datamodel dataset = microsoft_windows_raw
	| alter collector_type = microsoft_windows_raw._collector_type | filter collector_type = "XDR Collector" | filter xdm.source.host.hostname in ("AMPSYDADCP002*", "AMPSYDADCP003*", "AMPMELADCP002*", "AMPMELADCP003*", "AMPAWSZ2ADCP001*", "AMPAWSZ1ADCP001*") | filter xdm.event.id = "4738"
	| alter OldUacValue = json_extract_scalar(microsoft_windows_raw.event_data ,"$.OldUacValue"), NewUacValue = json_extract_scalar(microsoft_windows_raw.event_data ,"$.NewUacValue"), status = json_extract_scalar(microsoft_windows_raw.event_data, "$.Status")
	| filter OldUacValue != ""
	| alter newUAC = to_integer(arrayindex(regextract(NewUacValue,"^0x([0-9]{1,8})"),0)), oldUAC = to_integer(arrayindex(regextract(OldUacValue,"^0x([0-9]{1,8})"),0)), target_user_sid = json_extract_scalar(microsoft_windows_raw.event_data, "$.TargetSid") | alter result = subtract(oldUAC ,newUAC)
	| filter result = 4000
index=os_event_prod host IN (AMPSYDADCP002*, AMPSYDADCP003*, AMPMELADCP002*, AMPMELADCP003*, AMPAWSZ2ADCP001*, AMPAWSZ1ADCP001*) EventCode=4738 Old_UAC_Value!="" | rex field=New_UAC_Value "^0x(?<newUAC>[0-9]{1,8})" | rex field=Old_UAC_Value "^0x(?<oldUAC>[0-9]{1,8})" | eval result = tonumber(oldUAC) - tonumber(newUAC) | search result=4000 | rename Target_Account_Name as target_user, Subject_Account_Name as src_user | stats count min(_time) as firstTime max(_time) as lastTime values(EventCode) as eventCode by target_user, src_user	| comp count() as event_count, min(_time) as firstTime, max(_time) as lastTime, values(xdm.event.id) as eventCode, values(xdm.event.original_event_type) as signature, values(xdm.source.host.hostname) as dest, values(xdm.source.ipv4) as src, values(NewUacValue) as newUaValue, values(OldUacValue) as oldUacValue, values(xdm.event.description) as message, values(xdm.source.user.domain) as src_user_domain, values(xdm.target.user.domain) as target_user_domain, values(xdm.source.user.identifier) as src_user_sid, values(target_user_sid) as target_user_sid by xdm.target.user.username, xdm.source.user.username, result | fields event_count, firstTime, lastTime, eventCode, signature, dest, src, newUaValue, oldUacValue, xdm.target.user.username as target_user, xdm.source.user.username as src_user, result, message, src_user_domain, target_user_domain, src_user_sid, target_user_sid

	// Title: Network - [AWS] Detect AWS Usage of root Account [eSecure] - Rule // Description: Detect usage of AWS "root" account. CIS Amazon Web Services Foundations - 3.3 // Author: Mandeep Singh, msingh8@paloaltonetworks.com // Datasets: amazon_aws_raw // Date: 10/July/2024 config case_sensitive = false | dataset = amazon_aws_raw | filter _collector_name contains "Cloudtrail" | alter userIdentity_type = userIdentity -> type, userName = userIdentity -> userName, arn = userIdentity -> arn, accessKeyId = userIdentity -> accessKeyId, accountId = userIdentity -> accountId, principalId = userIdentity -> principalId | filter userIdentity_type = "root" | filter eventType != "AwsServiceEvent" // "detail.service.additionalInfo.sample"!=true // field not found
sourcetype=aws:cloudtrail userIdentity.type=root eventType!=AwsServiceEvent "detail.service.additionalInfo.sample"!=true | stats count min(_time) as firstTime max(_time) as lastTime values(errorCode) values(userAgent) values(userIdentity.*) by aws_account_id src userName userIdentity.arn eventName | convert timeformat="%m/%d/%Y %H:%M:%S" ctime(firstTime) | convert timeformat="%m/%d/%Y %H:%M:%S" ctime(lastTime)	| comp count() as total_events, min(_time) as firstTime, max(_time) as lastTime, values(errorCode) as errorCode, values(userAgent) as userAgent, values(principalId) as userIdentity_principalId, values(accessKeyId) as accessKeyId, values(awsRegion) as awsRegion by accountId, sourceIPAddress, userName, arn, eventName, userIdentity_type
Drill Down	Drill Down
sourcetype=aws:cloudtrail userIdentity.type!=root eventType!=AwsServiceEvent src=src	config case_sensitive = false | dataset = amazon_aws_raw | alter userIdentity_type = userIdentity -> type | filter userIdentity_type != "root" and eventType != "AwsServiceEvent" and sourceIPAddress = $sourceIPAddress

```
index=* sourcetype=aws:cloudtrail eventName=ConsoleLogin OR eventName=CreateImage OR
eventName=AssociateAddress OR eventName=AttachInternetGateway OR
eventName=AttachVolume OR eventName=StartInstances OR eventName=StopInstances OR
eventName=UpdateService OR eventName=UpdateLoginProfile "detail.service.additionalInfo.
sample"!=true
| bucket _time span=1d
| stats count by user _time| eval maxtime=now()

| stats count as num_data_samples

max(eval(if(_time >= relative_time(maxtime, "-1d@d"), 'count',null))) as count

avg(eval(if(_time<relative_time(maxtime,"-1d@d"),'count',null))) as avg

stdev(eval(if(_time<relative_time(maxtime,"-1d@d"),'count',null))) as stdev by user

| eval lowerBound=(avg-stdev*2), upperBound=(avg+stdev*2)| eval isOutlier=if(('count' < lowerBound
OR 'count' > upperBound) AND num_data_samples >=7, 1, 0) | search isOutlier=1
```

```
config case_sensitive = false
| dataset = amazon_aws_raw
| filter _collector_name contains "Cloudtrail"
| filter eventName in ("ConsoleLogin", "CreateImage", "AssociateAddress", "AttachInternetGateway", "AttachVolume",
"StartInstances", "StopInstances", "UpdateService", "UpdateLoginProfile")
//| filter "detail.service.additionalInfo.sample"!=true // field detail not found in field mapping document and sample data as
well

| alter console_login = responseElements -> ConsoleLogin
| alter errorCode = coalesce(errorCode, if(console_login = "Failure", "Failure", "Success"), "Success")

| alter session_issuer_username = json_extract_scalar(userIdentity ,"$.sessionContext.sessionIssuer.userName"),
      user_arn = json_extract_scalar(userIdentity ,"$.arn"),
      aws_account_id = json_extract_scalar(userIdentity ,"$.accountId"),
      request_userName = json_extract_scalar(requestParameters,"$.userName"),
      principalId = json_extract_scalar(userIdentity ,"$.principalId")
| alter username = arrayindex(split(principalId, ":"), -1)

| bin _time span = 1d

| comp count() as count_ by _time, username
| alter maxtime = current_time()
| alter time_diff = divide(timestamp_diff(maxtime, _time, "MINUTE"), 60)
// | filter time_diff <= 24
| comp count() as num_data_samples, max(if(time_diff <=  24, count_)) as total_count, avg(if(time_diff > 24, count_)) as
average, min(_time) as firstTime, max(_time) as lastTime, stddev_population(if(time_diff > 24, count_)) as stdev by
username
| alter lowerbound = subtract(average, multiply(stdev, 2)),
      upperbound = add(average, multiply(stdev, 2))
| alter isoutlier= if((total_count < lowerbound or total_count > upperbound) and num_data_samples >= 7, 1,0)
| filter isoutlier = 1


// | fields eventID, eventName, eventSource, awsRegion, aws_account_id, username, principalId, user_arn,
request_userName, userAgent, errorCode, recipientAccountId, requestID, session_issuer_username, sourceIPAddress,
```

index=* sourcetype=aws:cloudtrail eventName=Create* OR eventName=Run* OR eventName=Attach* "detail.service.additionalInfo.sample"!=true |stats count by src eventName | iplocation src| stats earliest(_time) as earliest latest(_time) as latest by Country, sourcetype| eval maxlatest=now() | eval isOutlier=if(earliest >= relative_time(maxlatest, "-1d@d"), 1, 0) | search isOutlier=1	// Title: SSE - AWS Cloud Provisioning Activity from Unusual Country - Rule // Description: AWS Cloud Provisioning Activity from Unusual Country // Datasets: amazon_aws_raw // Date: 01/Jul/2024 /* index=* sourcetype=aws:cloudtrail eventName=Create* OR eventName=Run* OR eventName=Attach* "detail.service.additionalInfo.sample"!=true |stats count by src eventName | iplocation src| stats earliest(_time) as earliest latest(_time) as latest by Country, sourcetype| eval maxlatest=now() | eval isOutlier=if(earliest >= relative_time(maxlatest, "-1d@d"), 1, 0) | search isOutlier=1 */ config case_sensitive = false | dataset = amazon_aws_raw | filter _collector_name contains "Cloudtrail" | filter eventName in ("Create*", "Run*", "Attach*") // | filter detail.service.additionalInfo.sample"!=true // | comp count() as cnt by eventName, sourceIPAddress | alter userIdentity_type = userIdentity -> type, sourceType = requestParameters -> sourceType, arn = userIdentity -> arn, principalId = userIdentity -> principalId, accountId = userIdentity -> accountId | iploc sourceIPAddress loc_country as country, loc_city as city | comp count() as total_events, min(_time) as earliest_time, max(_time) as latest_time, values(awsRegion) as awsRegion, values(arn) as arn, values(principalId) as principalId, values(sourceIPAddress) as src, values(accountId) as accountId, values(eventName) as eventName, values(eventSource) as eventSource, values(city) as city, values(userAgent) as userAgent by country, sourceType | alter cur_time = current_time() | alter time_diff = timestamp_diff(cur_time, earliest_time, "HOUR") | filter time_diff <= 24
index=amp_cloudflare_prod ActionType=login OR ActionType=zone_init OR ActionType=zone_delete |table When ActionType Metadata.actual_user.user_email ActorEmail Metadata.zone_name	// Title: Threat - [AMP Network] CloudFlare High Risk Activities [CEng] - Rule // Description: This search detects high risk admin activities in CloudFlare such as: // ActionType=cloudflare_login (local / emergency account) // ActionType=zone_init (creation of new zones) // ActionType=zone_delete (deletion of zones) // Author: Mandeep Singh, msingh8@paloaltonetworks.com // Datasets: cloudflare_dns_raw // Date: 23/Sep/2024 dataset = cloudflare_dns_raw | filter ActionType in ("login", "zone_init", "zone_delete") | alter ZoneName = Metadata -> zone_name, AccountId = Metadata -> account_id, ActorEmail = Metadata -> actor_email, ActorId = Metadata -> actor_id, ActualUserEmail = json_extract_scalar(Metadata, "$.actual_user.user_email") | fields When, ActionType, Metadata, ZoneName, AccountId, ActorEmail, ActualUserEmail, ActorID, ActorIP, ActorType, *

index=o365_prod sourcetype=o365:management:activity tag=authentication | lookup amp_egress_ip_cidr cidr as src_ip OUTPUT egress | where isnull(egress) | iplocation src_ip | stats values(Operation) as operation, values(user) as user dc(user) as countUser by src, City, Country | where countUser > 10	// Title: Threat - [AMP] Azure Multiple Authentication from Single IP [SplunkPS] - Rule config case_sensitive = true | datamodel dataset = msft_o365_* | filter xdm.event.type = "authentication" | filter xdm.event.original_event_type in ("UserLoggedIn", "UserLoginFailed","TeamsSessionStarted") | filter xdm.source.ipv4 not in ("", null) //| filter xdm.source.ipv4 = "103.225.105.29" | join type = left (dataset = amp_egress_ip_cidr | fields cidr, egress) as egress_ip (incidr(xdm.source.ipv4, egress_ip.cidr) = true) | filter egress in ("", null) | comp count(xdm.event.original_event_type) as total_operations_count, values(xdm.event.original_event_type) as operations, min(_time) as min_time, max(_time) as max_time, values(xdm.source.user.username) as user_ids, count_distinct(xdm.source.user.username) as distinct_user_count by xdm.source.ipv4, xdm.source.location.city, xdm.source.location.country //, xdm.source.location.continent | filter distinct_user_count > 10 | fields xdm.source.ipv4 ,xdm.source.location.city , xdm.source.location.country , operations , user_ids , distinct_user_count , min_time ,max_time //,xdm.source.location.continent
index=amp_msdefender_prod sourcetype="ms:defender:atp:alerts" | fillnull value="" | stats count min(_time) as firstTime max(_time) as lastTime values(mitre_technique_id) as mitre_technique_id values(category) as category by user, dest, subject, incidentId, productName, severity, description | rename mitre_technique_id as annotations.mitre_attack ``` lookup the the mitre attack tatic_id based on the category value since technique_ids are empty most of the time``` | rename category as mitre_tactic_label | join mitre_tactic_label [| inputlookup mitre_attack_lookup | stats count by mitre_tactic_label, mitre_tactic_id | eval mitre_tactic_label = replace(mitre_tactic_label,"\\s","")] | rename mitre_tactic_label as annotations.mitre_attack.mitre_tactic | rename mitre_tactic_id as annotations.mitre_attack.mitre_tactic_id | eval annotations._frameworks="mitre_attack" | eval annotations._all='annotations.mitre_attack' | `security_content_ctime(firstTime)` | `security_content_ctime(lastTime)` | eval impact = case(severity="informational", 20, severity="low", 40, severity="medium", 60, severity="high", 80, severity="critical", 100, true(), 0) | eval risk_score=impact*80/100 | fields - impact, count | `microsoft_defender_alert_filter`	config case_sensitive = false | dataset = msft_graph_security_alerts_raw | filter productName in ("Microsoft Defender for Office 365", "Microsoft Data Loss Prevention") | alter evidence = evidence -> [], userPrincipalName = json_extract_scalar(evidence_microsoft_graph_security_userEvidence, "$.0.userAccount.userPrincipalName"), userDisplayName = json_extract_scalar(evidence_microsoft_graph_security_userEvidence, "$.userAccount.displayName"), userPrincipalName_evidence = json_extract_scalar(evidence ,"$.0.userAccount.userPrincipalName") \| alter userPrincipalName= coalesce(userPrincipalName,userPrincipalName_evidence) \| alter subject = to_string(arraymap(evidence, "@element"->subject)), mitreTechniques = json_extract_scalar(mitreTechniques, "$.0") | join type = left (dataset = mitre_attack_lookup | alter mitre_tactic_label = replace(mitre_tactic_label, " ", "") | comp count() as mitre_stats by mitre_tactic_id, mitre_tactic_label | fields mitre_tactic_id, mitre_tactic_label as tactic_label) as mitre_attack mitre_attack.tactic_label = category | comp count() as total_events, min(_time) as firstTime, max(_time) as lastTime, values(mitreTechniques) as mitre_technique_id, values(category) as mitre_tactic_label, values(mitre_tactic_id) as mitre_tactic_id by userPrincipalName, evidence_deviceDnsName, title , incidentId, productName, severity, description // // // // risk_scrore calculation //| alter score = if(severity = "informational", 20, severity = "low", 40, severity = "medium", 60, severity = "high", 80, severity = "critical", 100, 0) //| alter risk_score = divide(multiply(score, 80), 100) | alter `annotations._frameworks`="mitre_attack" | fields userPrincipalName as user, evidence_deviceDnsName as dest,title as subject , incidentId,productName, severity , description , firstTime , lastTime , mitre_technique_id as `annotations.mitre_attack`, mitre_tactic_label as `annotations.mitre_attack.mitre_tactic`, `annotations._frameworks` ,mitre_tactic_id as `annotations.mitre_attack.mitre_tactic_id`// , risk_score

sourcetype=aws:cloudtrail (eventName=PutBucketAcl OR eventName=PutBucketPolicy OR eventName=PutBucketCors OR eventName=PutBucketLifecycle OR eventName=PutBucketReplication OR eventName=DeleteBucketPolicy OR eventName=DeleteBucketCors OR eventName=DeleteBucketLifecycle OR eventName=DeleteBucketReplication) NOT "detail.service.additionalInfo.sample"=true | table userName, aws_account_id, eventName, eventType, eventCategory, awsRegion, eventSource Drill Down sourcetype=aws:cloudtrail (eventName=PutBucketAcl OR eventName=PutBucketPolicy OR eventName=PutBucketCors OR eventName=PutBucketLifecycle OR eventName=PutBucketReplication OR eventName=DeleteBucketPolicy OR eventName=DeleteBucketCors OR eventName=DeleteBucketLifecycle OR eventName=DeleteBucketReplication)	// Title: Threat - [AWS] Detect AWS S3 Policy Changes [eSecure] - Rule // Description: Monitoring these changes might reduce time to detect and correct permissive policies on sensitive S3 buckets. // Author: Mandeep Singh, msingh8@paloaltonetworks.com // Reviewer: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: amazon_aws_raw // Date: 09/July/2024 config case_sensitive = false | dataset = amazon_aws_raw | filter _collector_name contains "Cloudtrail" // filtering for cloudtrail logs | filter eventName in ("PutBucketAcl", "PutBucketPolicy", "PutBucketCors", "PutBucketLifecycle", "PutBucketReplication", "DeleteBucketPolicy", "DeleteBucketCors", "DeleteBucketLifecycle", "DeleteBucketReplication") //"detail.service.additionalInfo.sample"!=true | alter username = json_extract_scalar(userIdentity , "$.sessionContext.sessionIssuer.userName"), user_arn = json_extract_scalar(userIdentity , "$.sessionContext.sessionIssuer.arn"), aws_account_id = json_extract_scalar(userIdentity , "$.accountId") | fields _time, username, eventName, eventID, userAgent, awsRegion, eventCategory, eventSource, requestParameters, eventType, user_arn, aws_account_id, _collector_name // | comp count() as total_events, min(_time) as startTime, max(_time) as lasttime, values(eventID) as eventID, values(requestParameters) as requestParameters, values(userAgent) as userAgent by username, user_arn, awsRegion, eventCategory, eventSource, _collector_name, aws_account_id, eventName Drill Down config case_sensitive = false | dataset = amazon_aws_raw | filter eventName in ("PutBucketAcl", "PutBucketPolicy", "PutBucketCors", "PutBucketLifecycle", "PutBucketReplication", "DeleteBucketPolicy", "DeleteBucketCors", "DeleteBucketLifecycle", "DeleteBucketReplication")
sourcetype="amp:was:ivr:services" eventtype=amp_ivr_login_failure action=failure | eventstats count | where count>=20 Drill Down sourcetype="amp:was:ivr:services" eventtype=amp_ivr_login_failure	dataset = was_ivr_service_raw |filter action="failure" |filter signature = "Authentication with Bank not successful" |comp count() as count |filter count >= 20 Drill Down: dataset = was_ivr_service_raw |filter action="failure" |filter signature = "Authentication with Bank not successful"
sourcetype=amp:was:ivr:services eventtype=amp_ivr_password_reset_failure message="*Error happened in change password (pin) operation*" | eventstats count | where count>=5 Drill Down sourcetype=amp:was:ivr:services eventtype=amp_ivr_password_reset_failure message="*Error happened in change password (pin) operation*"	dataset = was_ivr_service_raw |filter action="modified" |filter status = "failure" |windowcomp count() as count |filter count >= 5 Drill Down: dataset = was_ivr_service_raw |filter action="modified" |filter status = "failure" |filter message="*Error happened in change password (pin) operation*"
| tstats count where sourcetype=amp:was:ivr:services | where count>=2000 Drill Down sourcetype=amp:was:ivr:services	dataset = was_ivr_service_raw |comp count() as count |filter count >= 2000 Drill Down: dataset = was_ivr_service_raw

	// Title: [DDC] Account Logins From AU/NZ and International [eSecure] // Description: This search is to detect account logins from outside of expected countries. This could be indicative of an attacker using harvested credentials to sign-in to the account. // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: amp_ibm_isam_raw // Date: 29/May/2024 config case_sensitive = false | datamodel dataset in (ibm_isam_rp_prod_raw) | filter xdm.target.application.name = "/ddc" | filter xdm.source.user.username != "unauthenticated" // //| filter src_asset_tag!="amp_trusted" //field does not exist on XSIAM dataset // as per last | iploc xdm.source.ipv4 loc_country as Country, loc_city as City, loc_region as Region, loc_latlon as lat_lon | join type = left (dataset = expected_countries_csv | fields Country as cont, Expected) as expected_countries expected_countries.cont = Country | filter Expected = null | comp count() as total_events, count_distinct(Country) as country_count, values(Country) as Country, values(Region) as Region, values(City) as City, values(lat_lon) as Latitude_Longitude, values(xdm.source.ipv4) as source_ip, values (xdm.target.ipv4) as client_addr, values(xdm.network.http.method) as method, values(xdm.target.host.fqdn) as host, values(xdm.network.http.url) as url, values(xdm.network.http.response_code) as response_code, values(xdm.event.id) as x_request_id, values(xdm.source.application.name) as c2app, values(xdm.intermediate.application.name) as c2subapp, last(xdm.observer.vendor) as vendor, last(xdm.observer.product) as product, values(xdm.source.application.version) as c2env, min(_time) as firstEventTime, max(_time) as lastEventTime by xdm.source.user.username | filter country_count > 2 // filter for logn from more than two countries for same user | fields total_events, xdm.source.user.username as iv_user, source_ip, Latitude_Longitude, Region, City, Country, country_count, client_addr, method, host, url, response_code, x_request_id, c2app, c2subapp, vendor, product, c2env, firstEventTime, lastEventTime Drill Down config case_sensitive = false | datamodel dataset = ibm_isam_rp_prod_raw | filter xdm.source.user.username != "unauthenticated" // //| filter src_asset_tag!="amp_trusted" //field does not exist on XSIAM dataset | iploc xdm.source.ipv4 loc_country as Country, loc_city as City, loc_region as Region, loc_latlon as lat_lon | comp count() as total_events, count_distinct(Country) as country_count, values(Country) as Country, values(Region) as Region, values(City) as City, values(lat_lon) as Latitude_Longitude, values(xdm.source.ipv4) as source_ip, values (xdm.target.ipv4) as client_addr, values(xdm.network.http.method) as method, values(xdm.target.host.fqdn) as host, values(xdm.network.http.url) as url, values(xdm.network.http.response_code) as response_code, values(xdm.event.id) as x_request_id, values(xdm.source.application.name) as c2app, values(xdm.intermediate.application.name) as c2subapp, last(xdm.observer.vendor) as vendor, last(xdm.observer.product) as product, values(xdm.source.application.version) as c2env, min(_time) as firstEventTime, max(_time) as lastEventTime by xdm.source.user.username | filter country_count > 2 // filter for logn from more than two countries for same user | fields total_events, xdm.source.user.username as iv_user, source_ip, Latitude_Longitude, Region, City, Country, country_count, client_addr, method, host, url, response_code, x_request_id, c2app, c2subapp, vendor, product, c2env, firstEventTime, lastEventTime
sourcetype="amp:tam" app=ddc user!=unauthenticated src_asset_tag!=amp_trusted | stats count by src, user | iplocation src allfields=true | where Country!="" | eventstats dc(Country) by user | rename dc(Country) as count | where count>2 | lookup expected_countries.csv Country as Country OUTPUT Expected | where isnull(Expected) | table user, src, Continent, Country, City, Region, lat, lon Drill Down Query sourcetype="amp:tam" app=ddc user!=unauthenticated src_asset_tag!=amp_trusted | stats count by src, user | iplocation src allfields=true | where Country!="" | eventstats dc(Country) by user | rename dc(Country) as count | where count>2	

	// Title: Threat - [DDC] Excessive Failed Logins [eSecure] - Rule config case_sensitive = false | datamodel dataset in (ibm_isam_rp_prod_raw,ibm_isam_rp_nonprod_raw) | filter xdm.target.application.name = "/ddc" | filter xdm.network.http.response_code in ("404") // filtering for action = failure | filter xdm.network.http.url not contains "*/ddc/public/ui/*" | filter xdm.source.user.username != "unauthenticated" | iploc xdm.source.ipv4 loc_country as country, loc_city as city, loc_continent as continent | alter location = format_string("%s|%s|%s", city, country, continent) | fields _time ,xdm.source.user.username , xdm.target.application.name , xdm.network.http.response_code , xdm.network.http.url , xdm.source.ipv4 , location , * | comp count() as total_events, values(location) as location, values(xdm.source.ipv4) as source_ip, values(xdm.target.ipv4) as client_addr, values(xdm.network.http.method) as method, values(xdm.target.host.fqdn) as host, values(xdm.network.http.url) as url, values(xdm.network.http.response_code) as response_code, values(xdm.event.id) as x_request_id, values(xdm.source.application.name) as c2app, values(xdm.intermediate.application.name) as c2subapp, values(xdm.target.application.name) as junction, last(xdm.observer.vendor) as vendor, last(xdm.observer.product) as product, values(xdm.source.application.version) as c2env, min(_time) as firstEventTime, max(_time) as lastEventTime by xdm.source.user.username // values(ibm_isam_rp_nonprod_raw._raw_log) as raw_log, | filter total_events >= 75 | alter assignment_group = "AMP_ES&I_IDAM Technical Support_SNow", configuration_item = "MY AMP" | fields total_events, xdm.source.user.username as user, source_ip, location, client_addr, method, host, url, response_code, x_request_id, c2app, c2subapp, junction, vendor, product, c2env, firstEventTime, lastEventTime, assignment_group , configuration_item Drill Down Query config case_sensitive = false | datamodel dataset in (ibm_isam_rp_prod_raw, ibm_isam_rp_nonprod_raw) | filter xdm.target.application.name = "/ddc" | filter xdm.network.http.response_code not in ("404") | filter xdm.network.http.url not contains "*/ddc/public/ui/*" | filter xdm.source.user.username != "unauthenticated" | iploc xdm.source.ipv4 loc_country as country, loc_city as city, loc_continent as continent | alter location = format_string("%s|%s|%s", city, country, continent)
junction="/ddc" sourcetype=amp:tam action=failure request!=/ddc/public/ui/* user!=unauthenticated | stats count by user | where count>=75 Drill Down Query junction="/ddc" sourcetype=amp:tam action=failure request!=/ddc/public/ui/* user!=unauthenticated	| fields _time, location, xdm.source.ipv4 as source_ip, xdm.target.ipv4 as client_addr, xdm.network.http.method as method, xdm.target.host.fqdn as host, xdm.network.http.url as url, xdm.network.http.response_code as response_code, xdm.event.id as x_request_id, xdm.source.application.name as c2app, xdm.intermediate.application.name as c2subapp, ibm_isam_rp_nonprod_raw.junction as junction, ibm_isam_rp_nonprod_raw._raw_log as raw_log, xdm.observer.vendor as vendor, xdm.observer.product as product, xdm.source.application.version as c2env, xdm.source.user.username as iv_user

sourcetype="amp:tam" app=ddc c2env=prd NOT src_asset_tag=amp_trusted. //NTT to provide asset tag field data set name | stats dc(user) by src | rename dc(user) as count | where count>=50 Drill Down Query sourcetype="amp:tam" app=ddc c2env=prd NOT src_asset_tag=amp_trusted	// Title : [DDC] Excessive Number of Accounts from a single IP [eSecure] config case_sensitive = false | datamodel dataset in (ibm_isam_rp_prod_raw) | filter xdm.source.application.version in ("prd") | filter xdm.target.application.name = "/ddc" //| filter xdm.network.http.url contains "*/ddc/public/api/forms/retrieve?*" | fields _time ,xdm.source.user.username , xdm.source.ipv4 , xdm.source.application.version,xdm.source.application.name ,xdm.target.application.name , xdm.network.http.url , * | fields _time , xdm.source.user.username , xdm.source.ipv4 , xdm.source.application.version, xdm.source.application.name ,xdm.target.application.name ,xdm.target.application.name , xdm.network.http.url ,* //| comp count_distinct(xdm.source.user.username) as user by xdm.source.ipv4 | comp count_distinct(xdm.source.user.username)as dc_user_count, values(xdm.source.user.username) as iv_user, values(xdm.target.ipv4) as client_addr, values(xdm.network.http.method) as method, values(xdm.target.host.fqdn) as host, values(xdm.network.http.url) as url, values(xdm.network.http.response_code) as response_code, values(xdm.event.id) as x_request_id, values(xdm.source.application.name) as c2app, values(xdm.intermediate.application.name) as c2subapp, last(xdm.observer.vendor) as vendor, last(xdm.observer.product) as product by xdm.source.ipv4, xdm.source.application.version | filter dc_user_count >= 50 | fields dc_user_count, xdm.source.application.version as c2env, iv_user, client_addr, method, host, url, response_code, x_request_id, c2app, c2subapp, xdm.source.ipv4 as source_ip,vendor, product Drill Down Query config case_sensitive = false | datamodel dataset = ibm_isam_rp_prod_raw | filter xdm.source.application.version = "prd" // | filter src_asset_tag != amp_trusted // Field "src_asset_tag" not found | filter ibm_isam_rp_prod_raw.junction contains "ddc" | fields xdm.source.application.version as c2env, xdm.source.user.username as iv_user, xdm.target.ipv4 as client_addr, xdm.network.http.method as method, xdm.target.host.fqdn as host, xdm.network.http.url as url, xdm.network.http.response_code as response_code, xdm.event.id as x_request_id, xdm.source.application.name as c2app, xdm.intermediate.application.name as c2subapp, ibm_isam_rp_prod_raw.junction as junction, xdm.source.ipv4 as source_ip, ibm_isam_rp_prod_raw._raw_log as raw_log

	config case_sensitive = false | datamodel dataset in (ibm_isam_rp_prod_raw, ibm_isam_rp_nonprod_raw) | filter xdm.source.application.version in ("prd", "uat") | filter xdm.target.application.name = "/ddc" | filter xdm.network.http.url contains "*/ddc/public/api/forms/retrieve?*" | alter action = if(xdm.network.http.response_code = "404","failure") | filter xdm.network.http.response_code = "404" //Need to disable for nonProd | fields _time , xdm.source.user.username , xdm.source.ipv4 , xdm.source.application.version,xdm.source.application.name ,xdm.target.application.name ,action , xdm.network.http.url , * | fields _time , xdm.source.user.username , xdm.source.ipv4 , xdm.source.application.version, xdm.source.application.name ,xdm.target.application.name ,xdm.target.application.name ,action , xdm.network.http.url ,* | join (datamodel dataset in(ibm_isam_rp_prod_raw, ibm_isam_rp_nonprod_raw) | filter xdm.source.application.version in ("prd", "uat") | filter xdm.target.application.name = "/ddc" | filter xdm.network.http.url contains "*/ddc/public/api/forms/retrieve?*" | alter action = if(xdm.network.http.response_code = "404","failure") //| filter xdm.network.http.response_code = "404" | limit 1000000 | comp count() as total_events by action) as all_events all_events.action = action | comp count() as events_count, min(_time) as firstEventTime, max(_time) as lastEventTime, values(xdm.target.ipv4) as client_addr, values(xdm.network.http.method) as method, values(xdm.target.host.fqdn) as host, values(xdm.network.http.url) as url, values(xdm.network.http.response_code) as response_code, values(xdm.event.id) as x_request_id, values(xdm.source.application.name) as c2app, values(xdm.intermediate.application.name) as c2subapp, last(xdm.observer.vendor) as vendor, last(xdm.observer.product) as product, values(xdm.source.ipv4) as src by xdm.source.application.version, xdm.source.user.username, xdm.target.application.name , total_events | filter total_events >= 50 | fields xdm.source.user.username as user, src,client_addr, method, host, url, response_code, x_request_id, c2app, xdm.source.application.version as c2env,c2subapp, xdm.target.application.name as app, vendor, product, events_count ,firstEventTime, lastEventTime, total_events
sourcetype="amp:tam" c2env=prd app="ddc" uri_path="/ddc/public/api/forms/retrieve?*" action=failure | eventstats count | where count>=50 | table user, src | dedup user, src Drill Down sourcetype="amp:tam" c2env=prd junction="/ddc" request="/ddc/public/api/forms/retrieve?*"	DrillDown Query: config case_sensitive = false | datamodel dataset = ibm_isam_rp_prod_raw | filter xdm.source.application.version = "prd" | filter ibm_isam_rp_prod_raw.junction = "/ddc" | filter xdm.network.http.url contains "*/ddc/public/api/forms/retrieve?*" | alter action = if(xdm.network.http.response_code = "404","failure") | filter xdm.network.http.response_code = "404" | fields _time, src, xdm.target.ipv4 as client_addr, xdm.network.http.method as method, xdm.target.host.fqdn as host, xdm.network.http.url as url, xdm.network.http.response_code as response_code, xdm.event.id as x_request_id, xdm.source.application.name as c2app, xdm.intermediate.application.name as c2subapp, ibm_isam_rp_prod_raw.junction as junction, ibm_isam_rp_prod_raw._raw_log as raw_log, xdm.observer.vendor as vendor, xdm.observer.product as product, xdm.source.application.version as e2env, xdm.source.user.username as iv_user, xdm.source.ipv4 as source_ip

sourcetype="amp:tam" c2env=prd junction="/ddc" request="/ddc/public/api/forms/save?*" | eventstats count | where count>=1000 | table user, src | dedup user, src Drill Down sourcetype="amp:tam" c2env=prd junction="/ddc" request="/ddc/public/api/forms/save?*"	// Title : Threat - [DDC] Excessive Public Form Saves [eSecure] - Rule config case_sensitive = false | datamodel dataset in (ibm_isam_rp_prod_raw, ibm_isam_rp_nonprod_raw) | filter xdm.source.application.version in ("prd", "uat") | filter xdm.target.application.name = "/ddc" | filter xdm.network.http.url contains "*/ddc/public/api/forms/save?*" | join (datamodel dataset in (ibm_isam_rp_prod_raw, ibm_isam_rp_nonprod_raw) | filter xdm.source.application.version in ("prd", "uat") | filter xdm.target.application.name = "/ddc" | filter xdm.network.http.url contains "*/ddc/public/api/forms/save?*" | limit 1000000 | comp count() as total_events by xdm.source.application.version) as all_events all_events.xdm.source.application.version = xdm.source.application.version | comp count() as events_count, min(_time) as firstEventTime, max(_time) as lastEventTime, values(xdm.target.ipv4) as client_addr, values(xdm.network.http.method) as method, values(xdm.target.host.fqdn) as host, values(xdm.network.http.url) as url, values(xdm.network.http.response_code) as response_code, values(xdm.event.id) as x_request_id, values(xdm.source.application.name) as c2app, values(xdm.intermediate.application.name) as c2subapp, last(xdm.observer.vendor) as vendor, last(xdm.observer.product) as product, values(xdm.source.ipv4) as src by xdm.source.application.version, xdm.source.user.username, xdm.target.application.name , total_events | filter total_events >= 1000 | fields xdm.source.user.username as user, src,client_addr, method, host, url, response_code, x_request_id, c2app, xdm.source.application.version as c2env,c2subapp, xdm.target.application.name as app, vendor, product, events_count ,firstEventTime, lastEventTime, total_events DrillDown Query: datamodel dataset in (ibm_isam_rp_prod_raw, ibm_isam_rp_nonprod_raw) | filter xdm.source.application.version in ("prd", "uat") | filter xdm.target.application.name = "/ddc" | filter xdm.network.http.url contains "*/ddc/public/api/forms/save?*" | fields xdm.source.user.username as user, xdm.source.ipv4 as src,*
| tstats count where ((sourcetype="amp:tam" app=ddc) OR sourcetype="amp:was:ddc:access") | where count>=5000 Drill Down ((sourcetype="amp:tam" app=ddc) OR sourcetype="amp:was:ddc:access")	// Title: Threat - [DDC] Possible Application DoS Attempt [eSecure] - Rule // Description: This search detects when the number of requests to the DDC application exceed the threshold. // This could be indicative of a malicious actor looking to overload the application. Using the source IP, users, and a timechart you can investigate for anomalous spikes. // See: https://teamtools.amp.com.au/confluence/display/CYS/IDAM+Alerts+-+TAM+WAS+-+MyAMP // Author: Anjali Verma, anjverma@paloaltonetworks.com // Datasets: microsoft_windows_raw // Date: 08/July/2024 config case_sensitive = false | datamodel dataset = amp_ibm_isam_raw //| filter sourcetype="amp:tam" // Field does not exist on XSIAM dataset | filter amp_ibm_isam_raw.junction = "/ddc" //|filter sourcetype="amp:was:ddc:access" //Field does not exist on XSIAM dataset | comp values(xdm.source.application.version) as version,values(amp_ibm_isam_raw.junction) as application_name, count() as request_count by xdm.source.user.username ,xdm.source.ipv4 ,_time | filter request_count >= 5000 | fields _time ,application_name,version,xdm.source.user.username as username,xdm.source.ipv4 as src_ip, request_count

	//Threat - [GuardDuty] AWS Anomalous Behaviour - HIGH Priority Alerts [NTT] config case_sensitive = false | dataset = aws_guardduty_raw | filter _collector_name = "AWS Cloud2 Guardduty" //| filter accountId =591041037789 | filter type IN ("*AnomalousBehavior*") | filter severity > 6.9 | alter severity = if(severity >= 7, "High", severity >= 4 and severity <= 6.9 , "Medium", severity >= 1 and severity < 3.9 , "Low", to_string(severity)) //NOT detail.service.additionalInfo.sample=true. | filter not (incidr(src_ip, "10.162.0.0/16") or incidr(src_ip, "10.156.0.0/15")) | fields _time , severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner, Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host , API_Call ,API_Call_Remote_Ip , ScannedPort , Effective_Permission , AssumeRole , Request_Domain , DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name , Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets , Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets , _device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole , _raw_log , service , resource , userIdentity_principalId
index=aws_main_prod sourcetype="aws:cloudwatch:guardduty" account=591041037789 detail.type IN ("*AnomalousBehavior*") NOT detail.service.additionalInfo.sample=true | `guardduty_alert_result_table` | `aws_guardduty_search_ip_range` | search Severity > 6.9	Drill Down config case_sensitive = false | dataset = aws_guardduty_raw | alter name = "drill down: [GuardDuty] AWS uardDuty] AWS Anomalous Behaviour - HIGH Priority Alerts [NTT]" | filter accountId = $AccountId | filter Resource_Name = $Resource_Name | fields _time, severity, DetectorID, type, accountId, region, Resource_Name, Username, owner, Private_DNS_Name, PrivateIpAddress, Bucket_Name, description, Traffic, src_host, src_ip, dest_ip, dest_host, API_Call, API_Call_Remote_Ip, ScannedPort, Effective_Permission, AssumeRole, Request_Domain, DNS_Action_Blocked, Instance_Details_Value, Server, ImageDescription, eks_cluster_name, Acct_Block_Public_ACLs, Acct_Block_Public_Policy, Acct_Ignore_Public_ACLs, Acct_Restrict_Public_Buckets, Buck_Block_Public_ACLs, Buck_Block_Public_Policy, Buck_Ignore_Public_ACLs, Buck_Restrict_Public_Buckets, _device_id, src_user, service_count, service_eventFirstSeen, service_eventLastSeen, service_resourceRole, _raw_log, service, resource, userIdentity_principalId
Drill-down [{"name":"drill down: [GuardDuty] AWS uardDuty] AWS Anomalous Behaviour - HIGH Priority Alerts [NTT]","search":"index=aws_main_prod $Impacted_Account_Id$ $Resource_Name$","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	

index=aws_main_prod sourcetype="aws:cloudwatch:guardduty" account=591041037789 detail.type IN ("*AnomalousBehavior*", "Exfiltration:S3/ObjectRead.Unusual") NOT detail.service.additionalInfo.sample=true | `guardduty_alert_result_table` | `aws_guardduty_search_ip_range` | search Severity >= 4.0 AND Severity < 7.0 Drill Down Query [{"name":"drill down: [GuardDuty] $Resource_Name$ AWS Anomalous Behaviour - MEDIUM Priority Alerts [NTT]","search":"index=aws_main_prod $Impacted_Account_Id$ $Resource_Name$","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	config case_sensitive = false | dataset = aws_guardduty_raw | filter _collector_name = "AWS Cloud2 Guardduty" //| filter accountId =591041037789 | filter type IN ("*AnomalousBehavior*", "Exfiltration:S3/ObjectRead.Unusual") | filter severity >= 4.0 AND Severity < 7.0 | alter severity = if(severity >= 7, "High", severity >= 4 and severity <= 6.9 , "Medium", severity >= 1 and severity < 3.9 , "Low", to_string(severity)), sample = service -> additionalInfo.sample | filter sample != "true" // filtering out sample events | filter not (incidr(src_ip, "10.162.0.0/16") or incidr(src_ip, "10.156.0.0/15")) | fields _time, sample, severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner, Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host , API_Call ,API_Call_Remote_Ip , ScannedPort , Effective_Permission , AssumeRole , Request_Domain , DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name , Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets , Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets , _device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole , _raw_log , service , resource , userIdentity_principalId Drill Down Query config case_sensitive = false | dataset = aws_guardduty_raw | alter name = format_string("drill down: [GuardDuty] %s AWS Anomalous Behaviour - MEDIUM Priority Alerts [NTT]",$Resource_Name) | filter accountId = $AccountId | filter Resource_Name = $Resource_Name | fields _time, severity, DetectorID, type, accountId, region, Resource_Name, Username, owner, Private_DNS_Name, PrivateIpAddress, Bucket_Name, description, Traffic, src_host, src_ip, dest_ip, dest_host, API_Call, API_Call_Remote_Ip, ScannedPort, Effective_Permission, AssumeRole, Request_Domain, DNS_Action_Blocked, Instance_Details_Value, Server, ImageDescription, eks_cluster_name, Acct_Block_Public_ACLs, Acct_Block_Public_Policy, Acct_Ignore_Public_ACLs, Acct_Restrict_Public_Buckets, Buck_Block_Public_ACLs, Buck_Block_Public_Policy, Buck_Ignore_Public_ACLs, Buck_Restrict_Public_Buckets, _device_id, src_user, service_count, service_eventFirstSeen, service_eventLastSeen, service_resourceRole, _raw_log, service, resource, userIdentity_principalId

index=aws_main_prod sourcetype="aws:cloudwatch:guardduty" account=591041037789 detail.type IN ("Behavior:EC2/NetworkPortUnusual", "Behavior:EC2/TrafficVolumeUnusual") NOT detail.service.additionalInfo.sample=true | `guardduty_alert_result_table` | `aws_guardduty_search_ip_range` | search Severity >= 4.0 AND Severity < 7.0 Drill Down Query [{"name":"drill down: [GuardDuty] $Resource_Name$ AWS Baseline Deviation - MEDIUM Priority Alerts [NTT]","search":"index=aws_main_prod $Impacted_Account_Id$ $Resource_Name$","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	// Title: Threat - [GuardDuty] AWS Baseline Deviation - MEDIUM Priority Alerts [NTT] - Rule config case_sensitive = false | dataset = aws_guardduty_raw | filter _collector_name = "AWS Cloud2 Guardduty" //| filter accountId =591041037789 | filter type IN ("Behavior:EC2/NetworkPortUnusual", "Behavior:EC2/TrafficVolumeUnusual") | filter severity >= 4.0 AND Severity < 7.0 | alter severity = if(severity >= 7, "High", severity >= 4 and severity <= 6.9 , "Medium", severity >= 1 and severity < 3.9 , "Low", to_string(severity)) //NOT detail.service.additionalInfo.sample=true. | filter not (incidr(src_ip, "10.162.0.0/16") or incidr(src_ip, "10.156.0.0/15")) | fields _time , severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner, Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host , API_Call ,API_Call_Remote_Ip , ScannedPort , Effective_Permission , AssumeRole , Request_Domain , DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name , Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets , Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets , _device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole , _raw_log , service , resource , userIdentity_principalId Drill Down Query config case_sensitive = false | dataset = aws_guardduty_raw | alter name = format_string("drill down: [GuardDuty] %s AWS Baseline Deviation - MEDIUM Priority Alerts [NTT]",$Resource_Name) | filter accountId = $AccountId | filter Resource_Name = $Resource_Name | fields _time, severity, DetectorID, type, accountId, region, Resource_Name, Username, owner, Private_DNS_Name, PrivateIpAddress, Bucket_Name, description, Traffic, src_host, src_ip, dest_ip, dest_host, API_Call, API_Call_Remote_Ip, ScannedPort, Effective_Permission, AssumeRole, Request_Domain, DNS_Action_Blocked, Instance_Details_Value, Server, ImageDescription, eks_cluster_name, Acct_Block_Public_ACLs, Acct_Block_Public_Policy, Acct_Ignore_Public_ACLs, Acct_Restrict_Public_Buckets, Buck_Block_Public_ACLs, Buck_Block_Public_Policy, Buck_Ignore_Public_ACLs, Buck_Restrict_Public_Buckets, _device_id, src_user, service_count, service_eventFirstSeen, service_eventLastSeen, service_resourceRole, _raw_log, service, resource, userIdentity_principalId

index=aws_main_prod sourcetype="aws:cloudwatch:guardduty" account=591041037789 detail.type IN ("*BruteForce*") NOT "detail.service.additionalInfo.sample"=true | `guardduty_alert_result_table` | `aws_guardduty_search_ip_range` | search Severity > 6.9 Drill Down Query [{"name":"drill down:[GuardDuty] AWS Brute Force - HIGH Priority Alerts [NTT]","search":" index=aws_main_prod $Impacted_Account_Id$ $Resource_Name$"," earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	// Title: TThreat - [GuardDuty] AWS Brute Force - HIGH Priority Alerts [NTT] - Rule config case_sensitive = false | dataset = aws_guardduty_raw | filter _collector_name = "AWS Cloud2 Guardduty" //| filter accountId =591041037789 | filter type IN ("*BruteForce*") | filter severity > 6.9 | alter severity = if(severity >= 7, "High", severity >= 4 and severity <= 6.9 , "Medium", severity >= 1 and severity < 3.9 , "Low", to_string(severity)) //NOT detail.service.additionalInfo.sample=true. | filter not (incidr(src_ip, "10.162.0.0/16") or incidr(src_ip, "10.156.0.0/15")) | fields _time , severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner, Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host , API_Call ,API_Call_Remote_Ip , ScannedPort , Effective_Permission , AssumeRole , Request_Domain , DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name , Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets , Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets , _device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole , _raw_log , service , resource , userIdentity_principalId Drill Down Query config case_sensitive = false | dataset = aws_guardduty_raw | alter name = format_string("drill down:[GuardDuty] AWS Brute Force - HIGH Priority Alerts [NTT]") | filter accountId = $AccountId | filter Resource_Name = $Resource_Name | fields _time, severity, DetectorID, type, accountId, region, Resource_Name, Username, owner, Private_DNS_Name, PrivateIpAddress, Bucket_Name, description, Traffic, src_host, src_ip, dest_ip, dest_host, API_Call, API_Call_Remote_Ip, ScannedPort, Effective_Permission, AssumeRole, Request_Domain, DNS_Action_Blocked, Instance_Details_Value, Server, ImageDescription, eks_cluster_name, Acct_Block_Public_ACLs, Acct_Block_Public_Policy, Acct_Ignore_Public_ACLs, Acct_Restrict_Public_Buckets, Buck_Block_Public_ACLs, Buck_Block_Public_Policy, Buck_Ignore_Public_ACLs, Buck_Restrict_Public_Buckets, _device_id, src_user, service_count, service_eventFirstSeen, service_eventLastSeen, service_resourceRole, _raw_log, service, resource, userIdentity_principalId

	// Title: Threat - [GuardDuty] AWS GuardDuty Catch All HIGH Priority Alerts [NTT] - Rule // Description: All alerts with severity rating larger than 6.9. It can reveal brand-new alerts that haven't been prepared with event name and playbook. // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: aws_guardduty_raw // Date: 26/Aug/2024 config case_sensitive = false | dataset = aws_guardduty_raw | filter _collector_name = "AWS Cloud2 Guardduty" | filter accountId = "591041037789" // | filter accountid = "235008511430" | filter severity > 6.9 | alter severity = if(severity >= 7, "High", severity >= 4 and severity <= 6.9 , "Medium", severity >= 1 and severity < 3.9 , "Low", to_string(severity)) | join type = left (dataset = GuardDuty_Detail_Type_csv | filter severity = "High" | alter service_detail_type = `Detail Type` | alter `Detail Type` = replace(`Detail Type`, "*", "") | fields `Detail Type` as detail_type, service_detail_type) as self type contains self.detail_type | filter detail_type in (null, "") //excluding cloud-2 pilot range | filter not (incidr(src_ip, "10.162.0.0/16") or incidr(src_ip, "10.156.0.0/15")) | fields _time, severity, DetectorID, service_detail_type, type, accountId, region, Resource_Name, Username, owner, Private_DNS_Name, PrivateIpAddress, Bucket_Name, description, Traffic, src_host, src_ip, dest_ip, dest_host, API_Call, API_Call_Remote_Ip, ScannedPort, Effective_Permission, AssumeRole, Request_Domain, DNS_Action_Blocked, Instance_Details_Value, Server, ImageDescription, eks_cluster_name, Acct_Block_Public_ACLs, Acct_Block_Public_Policy, Acct_Ignore_Public_ACLs, Acct_Restrict_Public_Buckets, Buck_Block_Public_ACLs, Buck_Block_Public_Policy, Buck_Ignore_Public_ACLs, Buck_Restrict_Public_Buckets, _device_id, src_user, service_count, service_eventFirstSeen, service_eventLastSeen, service_resourceRole, _raw_log, service, resource, userIdentity_principalId Drill Down Query dataset = aws_guardduty_raw | alter name = format_string("Drill-down: [GuardDuty] %s AWS GuardDuty Catch All HIGH Priority Alerts [NTT]", $Resource_Name) | filter accountId = $accountId | filter Resource_Name = $Resource_Name | fields _time, severity, DetectorID, type, accountId, region, Resource_Name, Username, owner, Private_DNS_Name, PrivateIpAddress, Bucket_Name, description, Traffic, src_host, src_ip, dest_ip, dest_host, API_Call, API_Call_Remote_Ip, ScannedPort, Effective_Permission, AssumeRole, Request_Domain, DNS_Action_Blocked, Instance_Details_Value, Server, ImageDescription, eks_cluster_name, Acct_Block_Public_ACLs, Acct_Block_Public_Policy, Acct_Ignore_Public_ACLs, Acct_Restrict_Public_Buckets, Buck_Block_Public_ACLs, Buck_Block_Public_Policy, Buck_Ignore_Public_ACLs, Buck_Restrict_Public_Buckets, _device_id, src_user, service_count, service_eventFirstSeen, service_eventLastSeen, service_resourceRole, _raw_log, service, resource, userIdentity_principalId
index=aws_main_prod sourcetype="aws:cloudwatch:guardduty" account=591041037789 NOT detail.type IN ([|inputlookup GuardDuty_Detail_Type.csv |search severity=High|fields "Detail Type"|rename "Detail Type" as search| mvcombine delim="," search | mvcombine delim=", " search]) NOT detail.service.additionalInfo.sample=true | `guardduty_alert_result_table` | `aws_guardduty_search_ip_range` | search Severity > 6.9 Drill Down Query [{"name":"Drill-down: [GuardDuty] $Resource_Name$ AWS GuardDuty Catch All HIGH Priority Alerts [NTT]","search":"index=aws_main_prod $Impacted_Account_Id$ $Resource_Name$","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	

index=aws_main_prod sourcetype="aws:cloudwatch:guardduty" account=591041037789 NOT detail.type IN ([|inputlookup GuardDuty_Detail_Type.csv |search severity=Medium|fields "Detail Type" |rename "Detail Type" as search| mvcombine delim="," search | mvcombine delim=", " search]) NOT detail.service.additionalInfo.sample=true | `guardduty_alert_result_table` | `aws_guardduty_search_ip_range` | search Severity > 3.9 AND Severity < 7 Drill Down Query [{"name":"Drill-down: $Resource_Name$ AWS GuardDuty Catch All MEDIUM Priority Alerts [NTT]","search":"index=aws_main_prod $Impacted_Account_Id$ $Resource_Name$","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	// Title: Threat - [GuardDuty] AWS GuardDuty Catch All MEDIUM Priority Alerts [NTT] - Rule // Description: All AWS GuardDuty alerts with severity ratings between 4.0 to 6.9 // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: aws_guardduty_raw // Date: 26/Aug/2024 config case_sensitive = false | dataset = aws_guardduty_raw | filter _collector_name = "AWS Cloud2 Guardduty" | filter accountId = "591041037789" // | filter accountid = "235008511430" | filter severity > 3.9 and severity < 7 | alter severity = if(severity >= 7, "High", severity >= 4 and severity <= 6.9 , "Medium", severity >= 1 and severity < 3.9 , "Low", to_string(severity)) | join type = left (dataset = GuardDuty_Detail_Type_csv | filter severity = "Medium" | alter search_detail_type = `Detail Type` | alter `Detail Type` = replace(`Detail Type`, "*", "") | fields `Detail Type` as detail_type, search_detail_type) as self type contains self.detail_type | filter detail_type in (null, "") //excluding cloud-2 pilot range | filter not (incidr(src_ip, "10.162.0.0/16") or incidr(src_ip, "10.156.0.0/15")) | fields _time, severity, DetectorID, search_detail_type, type, accountId, region, Resource_Name, Username, owner, Private_DNS_Name, PrivateIpAddress, Bucket_Name, description, Traffic, src_host, src_ip, dest_ip, dest_host, API_Call, API_Call_Remote_Ip, ScannedPort, Effective_Permission, AssumeRole, Request_Domain, DNS_Action_Blocked, Instance_Details_Value, Server, ImageDescription, eks_cluster_name, Acct_Block_Public_ACLs, Acct_Block_Public_Policy, Acct_Ignore_Public_ACLs, Acct_Restrict_Public_Buckets, Buck_Block_Public_ACLs, Buck_Block_Public_Policy, Buck_Ignore_Public_ACLs, Buck_Restrict_Public_Buckets, _device_id, src_user, service_count, service_eventFirstSeen, service_eventLastSeen, service_resourceRole, _raw_log, service, resource, userIdentity_principalId Drill Down Query dataset = aws_guardduty_raw | alter name = format_string("Drill-down: [GuardDuty] %s AWS GuardDuty Catch All MEDIUM Priority Alerts [NTT]", $Resource_Name) | filter accountId = $accountId | filter Resource_Name = $Resource_Name | fields _time, severity, DetectorID, type, accountId, region, Resource_Name, Username, owner, Private_DNS_Name, PrivateIpAddress, Bucket_Name, description, Traffic, src_host, src_ip, dest_ip, dest_host, API_Call, API_Call_Remote_Ip, ScannedPort, Effective_Permission, AssumeRole, Request_Domain, DNS_Action_Blocked, Instance_Details_Value, Server, ImageDescription, eks_cluster_name, Acct_Block_Public_ACLs, Acct_Block_Public_Policy, Acct_Ignore_Public_ACLs, Acct_Restrict_Public_Buckets, Buck_Block_Public_ACLs, Buck_Block_Public_Policy, Buck_Ignore_Public_ACLs, Buck_Restrict_Public_Buckets, _device_id, src_user, service_count, service_eventFirstSeen, service_eventLastSeen, service_resourceRole, _raw_log, service, resource, userIdentity_principalId

	// Title: Threat - [GuardDuty] AWS GuardDuty HIGH Instance Credential Exfiltration [NTT] - Rule // Description: High alerts with severity rating larger than 6.9 and FindingType is “UnauthorizedAccess: IAMUser/InstanceCredentialExfiltration.InsideAWS” OR “UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration. OutsideAWS” // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: aws_guardduty_raw // Date: 26/Aug/2024 config case_sensitive = false | dataset = aws_guardduty_raw | filter _collector_name = "AWS Cloud2 Guardduty" | filter accountId = "591041037789" // | filter accountid = "235008511430" | filter type IN ("*UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.InsideAWS*", "*UnauthorizedAccess: IAMUser/InstanceCredentialExfiltration.OutsideAWS*") | filter severity > 6.9 | alter severity = if(severity >= 7, "High", severity >= 4 and severity <= 6.9 , "Medium", severity >= 1 and severity < 3.9 , "Low", to_string(severity)) | filter not (incidr(src_ip, "10.162.0.0/16") or incidr(src_ip, "10.156.0.0/15")) | fields _time , severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner, Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host , API_Call ,API_Call_Remote_Ip , ScannedPort , Effective_Permission , AssumeRole , Request_Domain , DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name , Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets , Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets , _device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole , _raw_log , service , resource , userIdentity_principalId
index=aws_main_prod sourcetype="aws:cloudwatch:guardduty" account=591041037789 detail.type=" UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.InsideAWS*" OR detail.type=" UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.OutsideAWS*" NOT detail.service. additionalInfo.sample!=true | `guardduty_alert_result_table` | `aws_guardduty_search_ip_range` | search Severity > 6.9 Drill Down Query [{"name":"Drill-down:[GuardDuty] AWS GuardDuty HIGH Instance Credential Exfiltration [NTT]"," search":"index=aws_main_prod $Impacted_Account_Id$ $Resource_Name$"," earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	Drill Down Query dataset = aws_guardduty_raw | alter name = "Drill-down:[GuardDuty] AWS GuardDuty HIGH Instance Credential Exfiltration [NTT]" | filter accountId = $accountId | filter Resource_Name = $Resource_Name and | fields _time, severity, DetectorID, type, accountId, region, Resource_Name, Username, owner, Private_DNS_Name, PrivateIpAddress, Bucket_Name, description, Traffic, src_host, src_ip, dest_ip, dest_host, API_Call, API_Call_Remote_Ip, ScannedPort, Effective_Permission, AssumeRole, Request_Domain, DNS_Action_Blocked, Instance_Details_Value, Server, ImageDescription, eks_cluster_name, Acct_Block_Public_ACLs, Acct_Block_Public_Policy, Acct_Ignore_Public_ACLs, Acct_Restrict_Public_Buckets, Buck_Block_Public_ACLs, Buck_Block_Public_Policy, Buck_Ignore_Public_ACLs, Buck_Restrict_Public_Buckets, _device_id, src_user, service_count, service_eventFirstSeen, service_eventLastSeen, service_resourceRole, _raw_log, service, resource, userIdentity_principalId

index=aws_main_prod sourcetype="aws:cloudwatch:guardduty" account=591041037789 detail.type="*" NOT detail.service.additionalInfo.sample=true | `guardduty_alert_result_table` | `aws_guardduty_search_ip_range` | search Severity > 0.9 AND Severity < 4 Drill Down Query [{"name":"Drill-down: [GuardDuty] AWS GuardDuty LOW Priority Alerts [NTT]","search":"index=aws_main_prod $Impacted_Account_Id$ $Resource_Name$","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	// Title: Threat - [GuardDuty] AWS GuardDuty LOW Priority Alerts [NTT] - Rule // Description: All AWS GuardDuty alerts with severity ratings between 1.0 to 3.9 // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: aws_guardduty_raw // Date: 26/Aug/2024 config case_sensitive = false | dataset = aws_guardduty_raw | filter _collector_name = "AWS Cloud2 Guardduty" | filter accountId = "591041037789" // | filter accountid = "235008511430" | filter type = "*" | filter severity > 0.9 and severity < 4 | alter severity = if(severity >= 7, "High", severity >= 4 and severity <= 6.9 , "Medium", severity >= 1 and severity < 3.9 , "Low", to_string(severity)) | filter not (incidr(src_ip, "10.162.0.0/16") or incidr(src_ip, "10.156.0.0/15")) | fields _time , severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner, Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host , API_Call ,API_Call_Remote_Ip , ScannedPort , Effective_Permission , AssumeRole , Request_Domain , DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name , Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets , Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets , _device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole , _raw_log , service , resource , userIdentity_principalId Drill Down Query dataset = aws_guardduty_raw | alter name = "Drill-down: [GuardDuty] AWS GuardDuty LOW Priority Alerts [NTT]" | filter accountId = $accountId | filter Resource_Name = $Resource_Name and | fields _time, severity, DetectorID, type, accountId, region, Resource_Name, Username, owner, Private_DNS_Name, PrivateIpAddress, Bucket_Name, description, Traffic, src_host, src_ip, dest_ip, dest_host, API_Call, API_Call_Remote_Ip, ScannedPort, Effective_Permission, AssumeRole, Request_Domain, DNS_Action_Blocked, Instance_Details_Value, Server, ImageDescription, eks_cluster_name, Acct_Block_Public_ACLs, Acct_Block_Public_Policy, Acct_Ignore_Public_ACLs, Acct_Restrict_Public_Buckets, Buck_Block_Public_ACLs, Buck_Block_Public_Policy, Buck_Ignore_Public_ACLs, Buck_Restrict_Public_Buckets, _device_id, src_user, service_count, service_eventFirstSeen, service_eventLastSeen, service_resourceRole, _raw_log, service, resource, userIdentity_principalId

index=aws_main_prod sourcetype="aws:cloudwatch:guardduty" account=591041037789 detail.type IN ("UnauthorizedAccess:IAMUser/ConsoleLoginSuccess*") NOT detail.service.additionalInfo.sample=true | `guardduty_alert_result_table` | `aws_guardduty_search_ip_range` Drill Down Query [{"name":"drill down: [GuardDuty] $Resource_Name$ AWS Impossible Travel Activity - MEDIUM Priority Alert [NTT]","search":"index=aws_main_prod $Impacted_Account_Id$ $Resource_Name$","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	// Title: Threat - [GuardDuty] AWS Impossible Travel Activity - MEDIUM Priority Alert [NTT] - Rule // Title: Threat - [GuardDuty] AWS Impossible Travel Activity - MEDIUM Priority Alert [NTT] - Rule config case_sensitive = false | dataset = aws_guardduty_raw | filter _collector_name = "AWS Cloud2 Guardduty" //| filter accountId =591041037789 | filter type IN ("UnauthorizedAccess:IAMUser/ConsoleLoginSuccess*") | alter severity = if(severity >= 7, "High", severity >= 4 and severity <= 6.9 , "Medium", severity >= 1 and severity < 3.9 , "Low", to_string(severity)), sampleEvent = service -> additionalInfo.sample //| filter sampleEvent != "true" // filtering out sample events | filter not (incidr(src_ip, "10.162.0.0/16") or incidr(src_ip, "10.156.0.0/15")) | fields _time, sampleEvent, severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner, Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host , API_Call ,API_Call_Remote_Ip , ScannedPort , Effective_Permission , AssumeRole , Request_Domain , DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name , Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets , Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets , _device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole , _raw_log , service , resource , userIdentity_principalId Drill Down Query dataset = aws_guardduty_raw | alter name = format_string("drill down: [GuardDuty] %s AWS Impossible Travel Activity - MEDIUM Priority Alert [NTT]",$Resource_Name) | filter Resource_Name = $Resource_Name | filter accountId = $accountId | fields _time, severity, DetectorID, type, accountId, region, Resource_Name, Username, owner, Private_DNS_Name, PrivateIpAddress, Bucket_Name, description, Traffic, src_host, src_ip, dest_ip, dest_host, API_Call, API_Call_Remote_Ip, ScannedPort, Effective_Permission, AssumeRole, Request_Domain, DNS_Action_Blocked, Instance_Details_Value, Server, ImageDescription, eks_cluster_name, Acct_Block_Public_ACLs, Acct_Block_Public_Policy, Acct_Ignore_Public_ACLs, Acct_Restrict_Public_Buckets, Buck_Block_Public_ACLs, Buck_Block_Public_Policy, Buck_Ignore_Public_ACLs, Buck_Restrict_Public_Buckets, _device_id, src_user, service_count, service_eventFirstSeen, service_eventLastSeen, service_resourceRole, _raw_log, service, resource, userIdentity_principalId | filter Resource_Name = $Resource_Name

index=aws_main_prod sourcetype="aws:cloudwatch:guardduty" account=591041037789 detail.type IN ("UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.InsideAWS") NOT detail.service.additionalInfo.sample=true | `guardduty_alert_result_table` | `aws_guardduty_search_ip_range` | search Severity >= 4.0 AND Severity < 7.0 Drill Down Query [{"name":"drill down: [GuardDuty] $Resource_Name$ AWS Instance Credential Exfiltration - MEDIUM Priority Alerts [NTT]","search":"index=aws_main_prod $Impacted_Account_Id$ $Resource_Name$ | `guardduty_alert_result_table`","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	// Title: Threat - [GuardDuty] AWS Instance Credential Exfiltration - MEDIUM Priority Alerts [NTT] - Rule // Title: Threat - [GuardDuty] AWS Instance Credential Exfiltration - MEDIUM Priority Alerts [NTT] - Rule config case_sensitive = false | dataset = aws_guardduty_raw | filter _collector_name = "AWS Cloud2 Guardduty" //| filter accountId =591041037789 | filter type IN ("UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.InsideAWS") | filter severity >= 4.0 AND Severity < 7.0 | alter severity = if(severity >= 7, "High", severity >= 4 and severity <= 6.9 , "Medium", severity >= 1 and severity < 3.9 , "Low", to_string(severity)), sampleEvent = service -> additionalInfo.sample | filter sampleEvent != "true" // filtering out sample events | filter not (incidr(src_ip, "10.162.0.0/16") or incidr(src_ip, "10.156.0.0/15")) | fields _time, sampleEvent, severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner, Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host , API_Call ,API_Call_Remote_Ip , ScannedPort , Effective_Permission , AssumeRole , Request_Domain , DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name , Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets , Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets , _device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole , _raw_log , service , resource , userIdentity_principalId Drill Down dataset = aws_guardduty_raw | alter name = format_string("drill down: [GuardDuty] %s AWS Instance Credential Exfiltration - MEDIUM Priority Alerts [NTT]", $Resource_Name) | filter AccountId = $accountId | filter Resource_Name = $Resource_Name | fields _time, severity, DetectorID, type, accountId, region, Resource_Name, Username, owner, Private_DNS_Name, PrivateIpAddress, Bucket_Name, description, Traffic, src_host, src_ip, dest_ip, dest_host, API_Call, API_Call_Remote_Ip, ScannedPort, Effective_Permission, AssumeRole, Request_Domain, DNS_Action_Blocked, Instance_Details_Value, Server, ImageDescription, eks_cluster_name, Acct_Block_Public_ACLs, Acct_Block_Public_Policy, Acct_Ignore_Public_ACLs, Acct_Restrict_Public_Buckets, Buck_Block_Public_ACLs, Buck_Block_Public_Policy, Buck_Ignore_Public_ACLs, Buck_Restrict_Public_Buckets, _device_id, src_user, service_count, service_eventFirstSeen, service_eventLastSeen, service_resourceRole, _raw_log, service, resource, userIdentity_principalId

index=aws_main_prod sourcetype="aws:cloudwatch:guardduty" account=591041037789 detail.type IN ("*MaliciousIPCaller*" "*Tor*") NOT detail.service.additionalInfo.sample=true | `guardduty_alert_result_table` | `aws_guardduty_search_ip_range` | search Severity > 6.9 Drill Down Query [{"name":"drill down: [GuardDuty] AWS Suspected Malicious Communication - HIGH Priority Alerts [NTT]","search":"index=aws_main_prod $Impacted_Account_Id$ $Resource_Name$","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	// Title: Threat - [GuardDuty] AWS Malicious Communication - HIGH Priority Alerts [NTT] - Rule // Title: Threat - [GuardDuty] AWS Malicious Communication - HIGH Priority Alerts [NTT] - Rule config case_sensitive = false | dataset = aws_guardduty_raw | filter _collector_name = "AWS Cloud2 Guardduty" //| filter accountId =591041037789 | filter type contains "*MaliciousIPCaller*" or type contains "*Tor*" | filter severity > 6.9 | alter severity = if(severity >= 7, "High", severity >= 4 and severity <= 6.9 , "Medium", severity >= 1 and severity < 3.9 , "Low", to_string(severity)), sampleEvent = service -> additionalInfo.sample //| filter sampleEvent != "true" // filtering out sample events | filter not (incidr(src_ip, "10.162.0.0/16") or incidr(src_ip, "10.156.0.0/15")) | fields _time, sampleEvent, severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner, Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host , API_Call ,API_Call_Remote_Ip , ScannedPort , Effective_Permission , AssumeRole , Request_Domain , DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name , Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets , Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets , _device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole , _raw_log , service , resource , userIdentity_principalId Drill Down Query dataset = aws_guardduty_raw | alter name = format_string("drill down: [GuardDuty] AWS Suspected Malicious Communication - HIGH Priority Alerts [NTT]") | filter accountId = $AccountId | filter Resource_Name = $Resource_Name | fields _time, severity, DetectorID, type, accountId, region, Resource_Name, Username, owner, Private_DNS_Name, PrivateIpAddress, Bucket_Name, description, Traffic, src_host, src_ip, dest_ip, dest_host, API_Call, API_Call_Remote_Ip, ScannedPort, Effective_Permission, AssumeRole, Request_Domain, DNS_Action_Blocked, Instance_Details_Value, Server, ImageDescription, eks_cluster_name, Acct_Block_Public_ACLs, Acct_Block_Public_Policy, Acct_Ignore_Public_ACLs, Acct_Restrict_Public_Buckets, Buck_Block_Public_ACLs, Buck_Block_Public_Policy, Buck_Ignore_Public_ACLs, Buck_Restrict_Public_Buckets, _device_id, src_user, service_count, service_eventFirstSeen, service_eventLastSeen, service_resourceRole, _raw_log, service, resource, userIdentity_principalId

index=aws_main_prod sourcetype="aws:cloudwatch:guardduty" account=591041037789 detail.type IN ("Discovery:Kubernetes/MaliciousIPCaller", "Discovery:Kubernetes/MaliciousIPCaller.Custom", "Discovery:Kubernetes/TorIPCaller", "Discovery:S3/TorIPCaller", "Impact:EC2/AbusedDomainRequest.Reputation", "Persistence:Kubernetes/MaliciousIPCaller", "Persistence:Kubernetes/MaliciousIPCaller.Custom", "Persistence:Kubernetes/TorIPCaller", "Recon:IAMUser/MaliciousIPCaller", "Recon:IAMUser/MaliciousIPCaller.Custom", "Recon:IAMUser/TorIPCaller", "UnauthorizedAccess:EC2/MaliciousIPCaller.Custom", "UnauthorizedAccess:IAMUser/MaliciousIPCaller", "UnauthorizedAccess:IAMUser/MaliciousIPCaller.Custom", "UnauthorizedAccess:IAMUser/TorIPCaller") NOT detail.service.additionalInfo.sample=true | `guardduty_alert_result_table` | `aws_guardduty_search_ip_range` | search Severity >= 4.0 AND Severity < 7.0	config case_sensitive = false | dataset = aws_guardduty_raw | filter _collector_name = "AWS Cloud2 Guardduty" //| filter accountId =591041037789 | filter type in ("Discovery:Kubernetes/MaliciousIPCaller", "Discovery:Kubernetes/MaliciousIPCaller.Custom", "Discovery:Kubernetes/TorIPCaller", "Discovery:S3/TorIPCaller", "Impact:EC2/AbusedDomainRequest.Reputation", "Persistence:Kubernetes/MaliciousIPCaller", "Persistence:Kubernetes/MaliciousIPCaller.Custom", "Persistence:Kubernetes/TorIPCaller", "Recon:IAMUser/MaliciousIPCaller", "Recon:IAMUser/MaliciousIPCaller.Custom", "Recon:IAMUser/TorIPCaller", "UnauthorizedAccess:EC2/MaliciousIPCaller.Custom", "UnauthorizedAccess:IAMUser/MaliciousIPCaller", "UnauthorizedAccess:IAMUser/MaliciousIPCaller.Custom", "UnauthorizedAccess:IAMUser/TorIPCaller") | filter severity >= 4.0 AND Severity < 7.0 | alter severity = if(severity >= 7, "High", severity >= 4 and severity <= 6.9 , "Medium", severity >= 1 and severity < 3.9 , "Low", to_string(severity)), sampleEvent = service -> additionalInfo.sample //| filter sampleEvent != "true" // filtering out sample events | filter not (incidr(src_ip, "10.162.0.0/16") or incidr(src_ip, "10.156.0.0/15")) | fields _time, sampleEvent, severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner, Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host , API_Call ,API_Call_Remote_Ip , ScannedPort , Effective_Permission , AssumeRole , Request_Domain , DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name , Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets , Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets , _device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole , _raw_log , service , resource , userIdentity_principalId
Drill down [{"name":"drill down: [GuardDuty] $Resource_Name$ AWS Malicious Communication - MEDIUM Priority Alert","search":"index=aws_main_prod $Impacted_Account_Id$ $Resource_Name$"," earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	Drill Down config case_sensitive = false | dataset = aws_guardduty_raw | alter name = format_string("drill down: [GuardDuty] %s AWS Malicious Communication - MEDIUM Priority Alert", $Resource_Name) | filter accountId = $AccountId | filter Resource_Name = $Resource_Name | fields _time, severity, DetectorID, type, accountId, region, Resource_Name, Username, owner, Private_DNS_Name, PrivateIpAddress, Bucket_Name, description, Traffic, src_host, src_ip, dest_ip, dest_host, API_Call, API_Call_Remote_Ip, ScannedPort, Effective_Permission, AssumeRole, Request_Domain, DNS_Action_Blocked, Instance_Details_Value, Server, ImageDescription, eks_cluster_name, Acct_Block_Public_ACLs, Acct_Block_Public_Policy, Acct_Ignore_Public_ACLs, Acct_Restrict_Public_Buckets, Buck_Block_Public_ACLs, Buck_Block_Public_Policy, Buck_Ignore_Public_ACLs, Buck_Restrict_Public_Buckets, _device_id, src_user, service_count, service_eventFirstSeen, service_eventLastSeen, service_resourceRole, _raw_log, service, resource, userIdentity_principalId

index=aws_main_prod sourcetype="aws:cloudwatch:guardduty" account=591041037789 detail.type IN ("Backdoor:*" "Trojan:*") AND detail.type!="*DenialOfService*" NOT detail.service.additionalInfo.sample=true | `guardduty_alert_result_table` | `aws_guardduty_search_ip_range` | search Severity > 6.9 Drill down [{"name":"drill down: [GuardDuty] AWS Malware - HIGH Priority Alerts [NTT]","search":"index=aws_main_prod $Impacted_Account_Id$ $Resource_Name$ | `guardduty_alert_result_table`","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	// title = Threat - [GuardDuty] AWS Malware - HIGH Priority Alerts [NTT] - Rule config case_sensitive = false | dataset = aws_guardduty_raw | filter _collector_name = "AWS Cloud2 Guardduty" //| filter accountId =591041037789 | filter ((type contains "Backdoor:*" or type contains "Trojan:*") AND (type not contains "*DenialOfService*")) | filter severity > 6.9 | alter severity = if(severity >= 7, "High", severity >= 4 and severity <= 6.9 , "Medium", severity >= 1 and severity < 3.9 , "Low", to_string(severity)), sampleEvent = service -> additionalInfo.sample | filter sampleEvent != "true" // filtering out sample events | filter not (incidr(src_ip, "10.162.0.0/16") or incidr(src_ip, "10.156.0.0/15")) | fields _time, sampleEvent, severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner, Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host , API_Call ,API_Call_Remote_Ip , ScannedPort , Effective_Permission , AssumeRole , Request_Domain , DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name , Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets , Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets , _device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole , _raw_log , service , resource , userIdentity_principalId Drill Down config case_sensitive = false | dataset = aws_guardduty_raw | alter name = "drill down: [GuardDuty] AWS Malware - HIGH Priority Alerts [NTT]" | filter accountId = $accountId | filter Resource_Name = $Resource_Name | fields _time, severity, DetectorID, type, accountId, region, Resource_Name, Username, owner, Private_DNS_Name, PrivateIpAddress, Bucket_Name, description, Traffic, src_host, src_ip, dest_ip, dest_host, API_Call, API_Call_Remote_Ip, ScannedPort, Effective_Permission, AssumeRole, Request_Domain, DNS_Action_Blocked, Instance_Details_Value, Server, ImageDescription, eks_cluster_name, Acct_Block_Public_ACLs, Acct_Block_Public_Policy, Acct_Ignore_Public_ACLs, Acct_Restrict_Public_Buckets, Buck_Block_Public_ACLs, Buck_Block_Public_Policy, Buck_Ignore_Public_ACLs, Buck_Restrict_Public_Buckets, _device_id, src_user, service_count, service_eventFirstSeen, service_eventLastSeen, service_resourceRole, _raw_log, service, resource, userIdentity_principalId

	// title = Threat - [GuardDuty] AWS Malware - MEDIUM Priority Alerts [NTT] - Rule config case_sensitive = false | dataset = aws_guardduty_raw | filter _collector_name = "AWS Cloud2 Guardduty" //| filter accountId =591041037789 | filter type in ("Backdoor:EC2/Spambot", "Trojan:EC2/BlackholeTraffic", "Trojan:EC2/BlackholeTraffic!DNS", "Trojan:EC2/DropPoint", "Trojan:EC2/DropPoint!DNS") and (type not contains "*DenialOfService*") | filter Severity >= 4.0 AND Severity < 7.0 | alter severity = if(severity >= 7, "High", severity >= 4 and severity <= 6.9 , "Medium", severity >= 1 and severity < 3.9 , "Low", to_string(severity)) |alter sampleEvent = service -> additionalInfo.sample | filter sampleEvent != "true" // filtering out sample events | filter not (incidr(src_ip, "10.162.0.0/16") or incidr(src_ip, "10.156.0.0/15")) | fields _time, sampleEvent, severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner, Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host , API_Call ,API_Call_Remote_Ip , ScannedPort , Effective_Permission , AssumeRole , Request_Domain , DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name , Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets , Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets , _device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole , _raw_log , service , resource , userIdentity_principalId Drill Down
index=aws_main_prod sourcetype="aws:cloudwatch:guardduty" account=591041037789 detail.type IN ("Backdoor:EC2/Spambot", "Trojan:EC2/BlackholeTraffic", "Trojan:EC2/BlackholeTraffic!DNS", "Trojan:EC2/DropPoint", "Trojan:EC2/DropPoint!DNS") AND detail.type!="*DenialOfService*" NOT detail.service.additionalInfo.sample=true | `guardduty_alert_result_table` | `aws_guardduty_search_ip_range` | search Severity >= 4.0 AND Severity < 7.0 Drill down [{"name":"drill down: [GuardDuty] $Resource_Name$ AWS Malware - HIGH Priority Alerts [NTT]","search":"index=aws_main_prod $Impacted_Account_Id$ $Resource_Name$ | `guardduty_alert_result_table`","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	config case_sensitive = false | datamodel dataset = aws_guardduty_raw | alter name = "drill down: [GuardDuty] AWS Malware - MEDIUM Priority Alerts [NTT]" | filter accountId = $accountId | filter Resource_Name = $Resource_Name | fields _time, severity, DetectorID, type, accountId, region, Resource_Name, Username, owner, Private_DNS_Name, PrivateIpAddress, Bucket_Name, description, Traffic, src_host, src_ip, dest_ip, dest_host, API_Call, API_Call_Remote_Ip, ScannedPort, Effective_Permission, AssumeRole, Request_Domain, DNS_Action_Blocked, Instance_Details_Value, Server, ImageDescription, eks_cluster_name, Acct_Block_Public_ACLs, Acct_Block_Public_Policy, Acct_Ignore_Public_ACLs, Acct_Restrict_Public_Buckets, Buck_Block_Public_ACLs, Buck_Block_Public_Policy, Buck_Ignore_Public_ACLs, Buck_Restrict_Public_Buckets, _device_id, src_user, service_count, service_eventFirstSeen, service_eventLastSeen, service_resourceRole, _raw_log, service, resource, userIdentity_principalId

index=aws_main_prod sourcetype="aws:cloudwatch:guardduty" account=591041037789 detail.type IN ("PenTest*") "detail.service.additionalInfo.sample"!=true | `guardduty_alert_result_table` | `aws_guardduty_search_ip_range` Drill down [{"name":"drill down: [GuardDuty] AWS Pentest [NTT]","search":"index=aws_main_prod $Impacted_Account_Id$ $Resource_Name$","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	// title :Threat - [GuardDuty] AWS Pentest - MEDIUM Priority Alert [NTT] - Rule config case_sensitive = false | dataset = aws_guardduty_raw | filter _collector_name = "AWS Cloud2 Guardduty" //| filter accountId =591041037789 | filter type in ("PenTest*") | alter severity = if(severity >= 7, "High", severity >= 4 and severity <= 6.9 , "Medium", severity >= 1 and severity < 3.9 , "Low", to_string(severity)) |alter sampleEvent = service -> additionalInfo.sample | filter sampleEvent != "true" // filtering out sample events | filter not (incidr(src_ip, "10.162.0.0/16") or incidr(src_ip, "10.156.0.0/15")) | fields _time, sampleEvent, severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner, Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host , API_Call ,API_Call_Remote_Ip , ScannedPort , Effective_Permission , AssumeRole , Request_Domain , DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name , Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets , Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets , _device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole , _raw_log , service , resource , userIdentity_principalId Drill Down config case_sensitive = false | datamodel dataset = aws_guardduty_raw | alter name = "drill down: [GuardDuty] AWS Pentest - MEDIUM Priority Alert [NTT] " | filter accountId = $accountId | filter Resource_Name = $Resource_Name | fields _time, severity, DetectorID, type, accountId, region, Resource_Name, Username, owner, Private_DNS_Name, PrivateIpAddress, Bucket_Name, description, Traffic, src_host, src_ip, dest_ip, dest_host, API_Call, API_Call_Remote_Ip, ScannedPort, Effective_Permission, AssumeRole, Request_Domain, DNS_Action_Blocked, Instance_Details_Value, Server, ImageDescription, eks_cluster_name, Acct_Block_Public_ACLs, Acct_Block_Public_Policy, Acct_Ignore_Public_ACLs, Acct_Restrict_Public_Buckets, Buck_Block_Public_ACLs, Buck_Block_Public_Policy, Buck_Ignore_Public_ACLs, Buck_Restrict_Public_Buckets, _device_id, src_user, service_count, service_eventFirstSeen, service_eventLastSeen, service_resourceRole, _raw_log, service, resource, userIdentity_principalId

	// title :Threat - [GuardDuty] AWS Pentest - MEDIUM Priority Alert [NTT] - Rule config case_sensitive = false | dataset = aws_guardduty_raw | filter _collector_name = "AWS Cloud2 Guardduty" //| filter accountId =591041037789 | filter type contains "Recon*" | filter Severity >= 6.9 | alter severity = if(severity >= 7, "High", severity >= 4 and severity <= 6.9 , "Medium", severity >= 1 and severity < 3.9 , "Low", to_string(severity)) |alter sampleEvent = service -> additionalInfo.sample | filter sampleEvent != "true" // filtering out sample events | filter not (incidr(src_ip, "10.162.0.0/16") or incidr(src_ip, "10.156.0.0/15")) | fields _time, sampleEvent, severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner, Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host , API_Call ,API_Call_Remote_Ip , ScannedPort , Effective_Permission , AssumeRole , Request_Domain , DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name , Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets , Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets , _device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole , _raw_log , service , resource , userIdentity_principalId Drill Down
index=aws_main_prod sourcetype="aws:cloudwatch:guardduty" account=591041037789 detail.type IN ("Recon*") NOT detail.service.additionalInfo.sample=true | `guardduty_alert_result_table` | `aws_guardduty_search_ip_range` | search Severity > 6.9 Drill Down [{"name":"drill down: [GuardDuty] AWS Reconnaissance - HIGH Priority Alerts [NTT]","search":"index=aws_main_prod $Impacted_Account_Id$ $Resource_Name$","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	config case_sensitive = false | dataset = aws_guardduty_raw | alter name = "drill down: [GuardDuty] AWS Reconnaissance - HIGH Priority Alerts [NTT] " | filter accountId = $accountId | filter Resource_Name = $Resource_Name | fields _time, severity, DetectorID, type, accountId, region, Resource_Name, Username, owner, Private_DNS_Name, PrivateIpAddress, Bucket_Name, description, Traffic, src_host, src_ip, dest_ip, dest_host, API_Call, API_Call_Remote_Ip, ScannedPort, Effective_Permission, AssumeRole, Request_Domain, DNS_Action_Blocked, Instance_Details_Value, Server, ImageDescription, eks_cluster_name, Acct_Block_Public_ACLs, Acct_Block_Public_Policy, Acct_Ignore_Public_ACLs, Acct_Restrict_Public_Buckets, Buck_Block_Public_ACLs, Buck_Block_Public_Policy, Buck_Ignore_Public_ACLs, Buck_Restrict_Public_Buckets, _device_id, src_user, service_count, service_eventFirstSeen, service_eventLastSeen, service_resourceRole, _raw_log, service, resource, userIdentity_principalId

	// title :Threat - [GuardDuty] AWS Reconnaissance - MEDIUM Priority Alerts [NTT] - Rule config case_sensitive = false | dataset = aws_guardduty_raw | filter _collector_name = "AWS Cloud2 Guardduty" //| filter accountId =591041037789 | filter type contains "Recon*" | filter Severity >= 4.0 AND Severity < 7.0 | alter severity = if(severity >= 7, "High", severity >= 4 and severity <= 6.9 , "Medium", severity >= 1 and severity < 3.9 , "Low", to_string(severity)) |alter sampleEvent = service -> additionalInfo.sample | filter sampleEvent != "true" // filtering out sample events | filter not (incidr(src_ip, "10.162.0.0/16") or incidr(src_ip, "10.156.0.0/15")) | fields _time, sampleEvent, severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner, Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host , API_Call ,API_Call_Remote_Ip , ScannedPort , Effective_Permission , AssumeRole , Request_Domain , DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name , Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets , Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets , _device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole , _raw_log , service , resource , userIdentity_principalId Drill Down
index=aws_main_prod sourcetype="aws:cloudwatch:guardduty" account=591041037789 detail.type IN ("Recon*") NOT detail.service.additionalInfo.sample=true | `guardduty_alert_result_table` | `aws_guardduty_search_ip_range` | search Severity >= 4.0 AND Severity < 7.0 Drill Down [{"name":"drill down: [GuardDuty] $Resource_Name$ AWS Reconnaissance - MEDIUM Priority Alerts [NTT]","search":"index=aws_main_prod $Impacted_Account_Id$ $Resource_Name$","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	config case_sensitive = false | dataset = aws_guardduty_raw | alter name = "drill down: [GuardDuty] AWS Reconnaissance - MEDIUM Priority Alerts [NTT] " | filter accountId = $accountId | filter Resource_Name = $Resource_Name | fields _time, severity, DetectorID, type, accountId, region, Resource_Name, Username, owner, Private_DNS_Name, PrivateIpAddress, Bucket_Name, description, Traffic, src_host, src_ip, dest_ip, dest_host, API_Call, API_Call_Remote_Ip, ScannedPort, Effective_Permission, AssumeRole, Request_Domain, DNS_Action_Blocked, Instance_Details_Value, Server, ImageDescription, eks_cluster_name, Acct_Block_Public_ACLs, Acct_Block_Public_Policy, Acct_Ignore_Public_ACLs, Acct_Restrict_Public_Buckets, Buck_Block_Public_ACLs, Buck_Block_Public_Policy, Buck_Ignore_Public_ACLs, Buck_Restrict_Public_Buckets, _device_id, src_user, service_count, service_eventFirstSeen, service_eventLastSeen, service_resourceRole, _raw_log, service, resource, userIdentity_principalId

```
index=aws_main_prod sourcetype="aws:cloudwatch:guardduty" account=591041037789 detail.type
IN ("Stealth*") NOT detail.service.additionalInfo.sample=true
| `guardduty_alert_result_table`
| `aws_guardduty_search_ip_range`
```

Drill down

```
[{"name":"drill down: [GuardDuty] AWS Stealth [NTT]","search":"index=aws_main_prod
$Impacted_Account_Id$ $Resource_Name$","earliest_offset":"$info_min_time$","
latest_offset":"$info_max_time$"}]
```

```
// title = Threat - [GuardDuty] AWS Stealth [NTT] - Rule
// Description = AWS GuardDuty Findings for pickup up the Type starts with "Stealth*". Severity includes all.

/*"index=aws_main_prod sourcetype=""aws:cloudwatch:guardduty"" account=591041037789 detail.type IN (""Stealth*"")
NOT detail.service.additionalInfo.sample=true
| `guardduty_alert_result_table`
| `aws_guardduty_search_ip_range`"*/

datamodel dataset = aws_guardduty_raw
//| filter xdm.target.resource.id  = "591041037789"
| filter aws_guardduty_raw._collector_name = "AWS Cloud2 Guardduty"
| filter xdm.alert.subcategory in ("Stealth*")

// Fetching additional Information and filter it
| alter additional_info_sample = json_extract_scalar(aws_guardduty_raw.service, "$.additionalInfo.sample")
// | filter additional_info_sample != "true"

// fetching data from dataset
| alter actionType = json_extract_scalar(aws_guardduty_raw.service, "$.action.actionType"),
      eventLastSeen = json_extract_scalar(aws_guardduty_raw.service, "$.eventLastSeen"),
      eventFirstSeen = json_extract_scalar(aws_guardduty_raw.service, "$.eventFirstSeen"),
      traffic = json_extract_scalar(aws_guardduty_raw.service, "$.action.networkConnectionAction.connectionDirection"),
      destHost = json_extract_scalar(aws_guardduty_raw.resource, "$.instanceDetails.instanceId"),
      assumeRole = json_extract_scalar(aws_guardduty_raw.service, "$.additionalInfo.profiledBehavior.
frequentProfiledUserNamesAccountProfiling"),
      DNS_Action_Blocked = json_extract_scalar(aws_guardduty_raw.service, "$.action.dnsRequestAction.blocked"),
      imageDescription = json_extract_scalar(aws_guardduty_raw.resource, "$.instanceDetails.imageDescription"),
      serviceCount = json_extract_scalar(aws_guardduty_raw.service, "$.count"),
      resourceRole = json_extract_scalar(aws_guardduty_raw.service, "$.resourceRole"),
      API_Call_Remote_Ip_Country = json_extract_scalar(aws_guardduty_raw.service, "$.action.awsApiCallAction.
remoteIpDetails.country.countryName"),
      API_Call_Remote_Ip_City = json_extract_scalar(aws_guardduty_raw.service, "$.action.awsApiCallAction.
remoteIpDetails.city.cityName")

| alter network_interfaces = json_extract_array(aws_guardduty_raw.resource, "$.instanceDetails.networkInterfaces"),
      bucket_details = json_extract_array(aws_guardduty_raw.resource, "$.s3BucketDetails")

| alter bucketName = arraystring(arraymap(bucket_details, "@element" -> arn), ",")
| alter privateIpAddress = arraystring(arraymap(network_interfaces, json_extract_scalar("@element", "$.
privateIpAddress")), ",")
| alter Private_DNS_Name = arraystring(arraymap(network_interfaces, json_extract_scalar("@element", "$.
privateDnsName")), ",")

// For macro`aws_guardduty_search_ip_range`
| filter not (incidr(xdm.source.ipv4, "10.162.0.0/16") or incidr(xdm.source.ipv4, "10.156.0.0/15"))

| fields _time, additional_info_sample, xdm.alert.severity, xdm.alert.category , xdm.alert.subcategory as type, xdm.target.
process.name, xdm.target.cloud.region, xdm.alert.original_threat_name as detectorId, xdm.target.resource.id as
impactecAccountId, xdm.target.resource.name as resourceName, xdm.source.user.username as userName,
bucketName, privateIpAddress, Private_DNS_Name, destHost, xdm.alert.description, traffic, xdm.source.ipv4, xdm.
target.ipv4, API_Call_Remote_Ip_Country, API_Call_Remote_Ip_City, assumeRole, DNS_Action_Blocked,
imageDescription, xdm.target.host.hostname as eksClusterName, eventFirstSeen, eventLastSeen, serviceCount,
resourceRole, xdm.alert.description ,xdm.event.outcome_reason , xdm.target.cloud.region
```

Drill Down

```
config case_sensitive = false
| datamodel dataset = aws_guardduty_raw
| alter name = "drill down: [GuardDuty] AWS Stealth [NTT]"
| filter xdm.target.resource.id = $impactecAccountId
| filter xdm.target.resource.name = $resourceName
```

	// title = Threat - [GuardDuty] AWS Successful Anonymous Access - HIGH Priority Alerts [NTT] - Rule // Description = AWS GuardDuty Findings for pickup up the Type starts with "*AnonymousAccess*". Severity > 6.9 // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: aws_guardduty_raw // Date: 29/Aug/2024 config case_sensitive = false | dataset = aws_guardduty_raw | filter _collector_name = "AWS Cloud2 Guardduty" // | filter accountId = "591041037789" // | filter accountid = "235008511430" | filter type IN ("*AnonymousAccess*") | filter severity > 6.9 | alter severity = if(severity >= 7, "High", severity >= 4 and severity <= 6.9 , "Medium", severity >= 1 and severity < 3.9 , "Low", to_string(severity)), sampleEvent = service -> additionalInfo.sample | filter sampleEvent != "true" // filtering out sample events | filter not (incidr(src_ip, "10.162.0.0/16") or incidr(src_ip, "10.156.0.0/15")) | fields _time , severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner, Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host , API_Call ,API_Call_Remote_Ip , ScannedPort , Effective_Permission , AssumeRole , Request_Domain , DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name , Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets , Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets , _device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole , _raw_log , service , resource , userIdentity_principalId
	Drill Down
index=aws_main_prod sourcetype="aws:cloudwatch:guardduty" account=591041037789 detail.type IN ("*AnonymousAccess*") NOT "detail.service.additionalInfo.sample"=true | `guardduty_alert_result_table` | `aws_guardduty_search_ip_range` | search Severity > 6.9	dataset = aws_guardduty_raw | alter name = "drill down: [GuardDuty] AWS Successful Anonymous Access - HIGH Priority Alerts [NTT]" | filter accountId = $accountId | filter Resource_Name = $Resource_Name | fields _time , severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner, Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host , API_Call ,API_Call_Remote_Ip , ScannedPort , Effective_Permission , AssumeRole , Request_Domain , DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name , Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets , Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets , _device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole , _raw_log , service , resource , userIdentity_principalId
Drill down [{"name":"drill down: [GuardDuty] AWS Successful Anonymous Access - HIGH Priority Alerts [NTT]","search":"index=aws_main_prod $Impacted_Account_Id$ $Resource_Name$","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	

	// title = Threat - [GuardDuty] AWS Successful Anonymous Access - MEDIUM Priority Alerts [NTT] - Rule // Description = AWS GuardDuty Findings for pickup up the Type starts with "Discovery: Kubernetes/SuccessfulAnonymousAccess". Severity >= 4.0 AND Severity < 7.0 // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: aws_guardduty_raw // Date: 29/Aug/2024 config case_sensitive = false | dataset = aws_guardduty_raw | filter _collector_name = "AWS Cloud2 Guardduty" // | filter accountId = "591041037789" // | filter accountid = "235008511430" | filter type IN ("Discovery:Kubernetes/SuccessfulAnonymousAccess") | filter severity >= 4 and severity < 7 | alter severity = if(severity >= 7, "High", severity >= 4 and severity <= 6.9 , "Medium", severity >= 1 and severity < 3.9 , "Low", to_string(severity)), sampleEvent = service -> additionalInfo.sample | filter sampleEvent != "true" // filtering out sample events | filter not (incidr(src_ip, "10.162.0.0/16") or incidr(src_ip, "10.156.0.0/15")) | fields _time , severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner, Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host , API_Call ,API_Call_Remote_Ip , ScannedPort , Effective_Permission , AssumeRole , Request_Domain , DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name , Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets , Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets , _device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole , _raw_log , service , resource , userIdentity_principalId Drill Down
index=aws_main_prod sourcetype="aws:cloudwatch:guardduty" account=591041037789 detail.type IN ("Discovery:Kubernetes/SuccessfulAnonymousAccess") "detail.service.additionalInfo.sample"!=true | `guardduty_alert_result_table` | `aws_guardduty_search_ip_range` | search Severity >= 4.0 AND Severity < 7.0 Drill down [{"name":"drill down: [GuardDuty] $Resource_Name$ AWS Successful Anonymous Access - MEDIUM Priority Alerts [NTT]","search":"index=aws_main_prod $Impacted_Account_Id$ $Resource_Name$","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	dataset = aws_guardduty_raw | alter name = format_string("drill down: [GuardDuty] %s AWS Successful Anonymous Access - MEDIUM Priority Alerts [NTT]", $Resource_Name) | filter accountId = $accountId | filter Resource_Name = $Resource_Name | fields _time , severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner, Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host , API_Call ,API_Call_Remote_Ip , ScannedPort , Effective_Permission , AssumeRole , Request_Domain , DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name , Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets , Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets , _device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole , _raw_log , service , resource , userIdentity_principalId

```
index=aws_main_prod sourcetype="aws:cloudwatch:guardduty" account=591041037789 detail.type
IN (Policy:Kubernetes/AdminAccessToDefaultServiceAccount, Policy:
Kubernetes/AnonymousAccessGranted, Policy:S3/BucketAnonymousAccessGranted, Policy:
S3/BucketPublicAccessGranted) "detail.service.additionalInfo.sample"!=true
| `guardduty_alert_result_table`
| `aws_guardduty_search_ip_range`
| search Severity > 6.9
```

Drill Down

```
[{"name":"drill-down:  [GuardDuty] AWS Suspicious Configuration Changes - HIGH Priority Alerts
[NTT]","search":"index=aws_main_prod sourcetype=\"aws:cloudwatch:guardduty\" $Resource_Name$
| eval Traffic=if(src_ip=PrivateIpAddress, \"outbound\", \"inbound\")  | table DetectorID Type Traffic
src_ip src_host dest_ip dest_host Owner Server Instance_Details","
earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]
```

```
// Title: [GuardDuty] AWS Suspicious Configuration Changes - HIGH Priority Alerts [NTT]
// Description: GuardDuty Finding is triggered as notables in Splunk:
//    a. AdminAccessToDefaultServiceAccount
//    b. AnonymousAccessGranted
//    c. BucketAnonymousAccessGranted
//    d. BucketPublicAccessGranted
//    Severity > 6.9
// Author: Sahil Sharma, ssharma7@paloaltonetworks.com
// Datasets: aws_guardduty_raw
// Date: 29/Aug/2024

config case_sensitive = false
| dataset = aws_guardduty_raw
| filter _collector_name = "AWS Cloud2 Guardduty"
// | filter accountId = "591041037789"
// | filter accountid = "235008511430"
| filter type IN ("Policy:Kubernetes/AdminAccessToDefaultServiceAccount", "Policy:
Kubernetes/AnonymousAccessGranted", "Policy:S3/BucketAnonymousAccessGranted", "Policy:
S3/BucketPublicAccessGranted")
| filter severity > 6.9

| alter severity = if( severity >= 7, "High", severity >= 4 and severity <= 6.9 , "Medium", severity >= 1 and severity < 3.9 ,
"Low", to_string(severity)),
      sampleEvent = service -> additionalInfo.sample

| filter sampleEvent != "true" // filtering out sample events
| filter not (incidr(src_ip, "10.162.0.0/16") or incidr(src_ip, "10.156.0.0/15"))

| fields _time , severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner,
Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host ,
API_Call ,API_Call_Remote_Ip , ScannedPort  , Effective_Permission , AssumeRole , Request_Domain ,
DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name ,
Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets ,
Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets ,
_device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole ,
_raw_log , service , resource , userIdentity_principalId
```

Drill Down

```
dataset = aws_guardduty_raw
| alter name = "drill-down:  [GuardDuty] AWS Suspicious Configuration Changes - HIGH Priority Alerts [NTT]"
| filter Resource_Name = $Resource_Name
| alter Traffic=if(src_ip = PrivateIpAddress, "outbound", "inbound")
| fields _time , severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner,
Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host ,
API_Call ,API_Call_Remote_Ip , ScannedPort  , Effective_Permission , AssumeRole , Request_Domain ,
DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name ,
Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets ,
Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets ,
_device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole ,
_raw_log , service , resource , userIdentity_principalId
```

	// Title: Threat - [GuardDuty] AWS Suspicious Configuration Changes - MEDIUM Priority Alerts [NTT] - Rule // Description: GuardDuty Finding is triggered as notables in Splunk: // a. Execution:Kubernetes/ExecInKubeSystemPod // b. Persistence:Kubernetes/ContainerWithSensitiveMount // c. Policy:Kubernetes/ExposedDashboard // d. Policy:Kubernetes/KubeflowDashboardExposed // e. PrivilegeEscalation:Kubernetes/PrivilegedContainer // Severity >= 4.0 AND Severity < 7.0 // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: aws_guardduty_raw // Date: 29/Aug/2024 config case_sensitive = false | dataset = aws_guardduty_raw | filter _collector_name = "AWS Cloud2 Guardduty" // | filter accountId = "591041037789" // | filter accountid = "235008511430" | filter type IN ("Execution:Kubernetes/ExecInKubeSystemPod", "Persistence:Kubernetes/ContainerWithSensitiveMount", "Policy:Kubernetes/ExposedDashboard", "Policy:Kubernetes/KubeflowDashboardExposed", "PrivilegeEscalation:Kubernetes/PrivilegedContainer") | filter severity >= 4 and severity < 7 | alter severity = if(severity >= 7, "High", severity >= 4 and severity <= 6.9 , "Medium", severity >= 1 and severity < 3.9 , "Low", to_string(severity)), sampleEvent = service -> additionalInfo.sample | filter sampleEvent != "true" // filtering out sample events | filter not (incidr(src_ip, "10.162.0.0/16") or incidr(src_ip, "10.156.0.0/15")) | fields _time , severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner, Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host , API_Call ,API_Call_Remote_Ip , ScannedPort , Effective_Permission , AssumeRole , Request_Domain , DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name , Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets , Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets , _device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole , _raw_log , service , resource , userIdentity_principalId
index=aws_main_prod sourcetype="aws:cloudwatch:guardduty" account=591041037789 detail.type IN (Execution:Kubernetes/ExecInKubeSystemPod, Persistence:Kubernetes/ContainerWithSensitiveMount, Policy:Kubernetes/ExposedDashboard, Policy:Kubernetes/KubeflowDashboardExposed PrivilegeEscalation:Kubernetes/PrivilegedContainer) "detail.service.additionalInfo.sample"!=true | `guardduty_alert_result_table` | `aws_guardduty_search_ip_range` | search Severity >= 4.0 AND Severity < 7.0 Drill Down Query [{"name":"drill-down: [GuardDuty] $Resource_Name$AWS Suspicious Configuration Changes - MEDIUM Priority Alerts [NTT]","search":"index=aws_main_prod sourcetype=\"aws:cloudwatch:guardduty\" $Resource_Name$ | eval Traffic=if(src_ip=PrivateIpAddress, \"outbound\", \"inbound\") | table DetectorID Type Traffic src_ip src_host dest_ip dest_host Owner Server Instance_Details","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	Drill Down Query dataset = aws_guardduty_raw | alter name = format_string("drill-down: [GuardDuty] %s AWS Suspicious Configuration Changes - MEDIUM Priority Alerts [NTT]",$Resource_Name) | filter Resource_Name = $Resource_Name | alter Traffic=if(src_ip = PrivateIpAddress, "outbound", "inbound") | fields _time , severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner, Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host , API_Call ,API_Call_Remote_Ip , ScannedPort , Effective_Permission , AssumeRole , Request_Domain , DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name , Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets , Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets , _device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole , _raw_log , service , resource , userIdentity_principalId

index=aws_main_prod sourcetype="aws:cloudwatch:guardduty" detail.type IN ("*DenialOfService*") "detail.service.additionalInfo.sample"!=true | `guardduty_alert_result_table` | `aws_guardduty_search_ip_range` | search Severity > 6.9 Drill Down Query [{"name":"drill down: [GuardDuty] AWS Denial Of Service - HIGH Priority Alerts [NTT]","search":" index=aws_main_prod $Impacted_Account_Id$ $Resource_Name$"," earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	// Title: Threat - [GuardDuty] AWS Suspicious Outbound Denial Of Service - HIGH Priority Alerts [NTT] - Rule // Description: AWS GuardDuty Findings for pickup up the Type starts with "*DenialOfService*". Severity > 6.9 // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: aws_guardduty_raw // Date: 29/Aug/2024 config case_sensitive = false | dataset = aws_guardduty_raw | filter _collector_name = "AWS Cloud2 Guardduty" // | filter accountId = "591041037789" // | filter accountid = "235008511430" | filter type IN ("*DenialOfService*") | filter severity > 6.9 | alter severity = if(severity >= 7, "High", severity >= 4 and severity <= 6.9 , "Medium", severity >= 1 and severity < 3.9 , "Low", to_string(severity)), sampleEvent = service -> additionalInfo.sample | filter sampleEvent != "true" // filtering out sample events | filter not (incidr(src_ip, "10.162.0.0/16") or incidr(src_ip, "10.156.0.0/15")) | fields _time , severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner, Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host , API_Call ,API_Call_Remote_Ip , ScannedPort , Effective_Permission , AssumeRole , Request_Domain , DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name , Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets , Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets , _device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole , _raw_log , service , resource , userIdentity_principalId Drill Down Query dataset = aws_guardduty_raw | alter name = format_string("Drill Down: [GuardDuty] AWS Denial Of Service - HIGH Priority Alerts [NTT]") | filter accountId = $AccountId | filter Resource_Name = $Resource_Name | fields _time , severity ,DetectorID , type , accountId , region , Resource_Name , Username , owner, Private_DNS_Name , PrivateIpAddress , Bucket_Name , description , Traffic , src_host , src_ip ,dest_ip , dest_host , API_Call ,API_Call_Remote_Ip , ScannedPort , Effective_Permission , AssumeRole , Request_Domain , DNS_Action_Blocked ,Instance_Details_Value , Server , ImageDescription , eks_cluster_name , Acct_Block_Public_ACLs , Acct_Block_Public_Policy , Acct_Ignore_Public_ACLs ,Acct_Restrict_Public_Buckets , Buck_Block_Public_ACLs , Buck_Block_Public_Policy , Buck_Ignore_Public_ACLs , Buck_Restrict_Public_Buckets , _device_id ,src_user, service_count , service_eventFirstSeen , service_eventLastSeen ,service_resourceRole , _raw_log , service , resource , userIdentity_principalId

sourcetype=amp:was:myamp:audit eventtype=amp_myamp_audit_account_lockout dest_asset_tag=prod | stats count by user | where count>=30 Drill Down Query sourcetype=amp:was:myamp:audit eventtype=amp_myamp_audit_account_lockout dest!=secure-uat.amp.com.au	// title :Threat - [IDAM WAS] Excessive Account Lockouts [eSecure] - Rule config case_sensitive = false |dataset = was_myamp_audit_raw |alter x_forwarded_host = trim(x_forwarded_host) |filter x_forwarded_host != "" | join (dataset = servicenow_cmdb_cmdb_ci_server_raw |filter host_name != ""| filter u_environment contains "production" and u_environment not contains "Non-production") as servicenow_cmdb_cmdb_ci_server_raw servicenow_cmdb_cmdb_ci_server_raw.host_name contains x_forwarded_host or x_forwarded_host contains servicenow_cmdb_cmdb_ci_server_raw.host_name |filter event_type = "FailedLogin" |filter reason = "The account is locked by user entering too many incorrect passwords." |comp count() as count by user_id | alter assignment_group = "AMP_ES&I_IDAM Technical Support_Snow", configuration_item = "IDAM TAM" | filter count>=30 Drill Down Query config case_sensitive = false |dataset = was_myamp_audit_raw |filter event_type = "FailedLogin" and reason = "The account is locked by user entering too many incorrect passwords." and x_forwarded_host !="secure-uat.amp.com.au"
sourcetype=amp:was:myamp:audit dest_asset_tag=prod NOT src_asset_tag=amp_trusted | stats dc(user) by src_ip | rename dc(user) as count | where count>=50 Drill Down Query sourcetype=amp:was:myamp:audit dest!=secure-uat.amp.com.au NOT src_asset_tag=amp_trusted	// title : Threat - [IDAM WAS] Excessive Number of Accounts from a single IP [eSecure] - Rule config case_sensitive = false |dataset = was_myamp_audit_raw |alter x_forwarded_host = trim(x_forwarded_host) | join (dataset = static_assets | fields dns, category |filter category contains "prod" and category not in ("non_prod","amp_trusted")) as static_assets static_assets.dns = x_forwarded_host |comp count_distinct(user_id) as count by client_ip |filter count >=50 Drill Down Query config case_sensitive = false |dataset = was_myamp_audit_raw x_forwarded_host!=”secure-uat.amp.com.au”
index=amp_digital_prod sourcetype=amp:was:mobility:services dest_asset_tag=prod eventtype=amp_mobility_services_logon_failure | stats count by dvc_id | where count>=75 Drill Down Query sourcetype=amp:was:mobility:services url!=https://esb-uat1.amp.com.au/* eventtype=amp_mobility_services_logon_failure	config case_sensitive = false | dataset = was_mobility_services_raw | alter dest = arrayindex(regextract(_raw_log, "https?:\/\/([^:\/]+)"),0) | alter device_id = arrayindex(regextract(_raw_log, "\\"DeviceId\\":\\"([a-zA-z0-9-]+)"),0) | filter action="failure" |comp count() as count by device_id | alter assignment_group = "AMP_ES&I_IDAM Technical Support_SNow", configuration_item = "MY AMP" |filter count >=75 Drill Down Query: config case_sensitive = false | dataset = was_mobility_services_raw |filter action="failure" |filter url!="https://esb-uat1.amp.com.au/*"

sourcetype=amp:was:mobility:services dest_asset_tag=prod dvc_id=* | stats dc(user) as count by dvc_id | where count>=10 Drill Down Query sourcetype=amp:was:mobility:services url!=https://esb-uat1.amp.com.au/* dvc_id=*	config case_sensitive = false | dataset = was_mobility_services_raw | alter dest = arrayindex(regextract(_raw_log, "https?:\/\/([^:\/]+)"),0) | alter device_id = arrayindex(regextract(_raw_log, "\\"DeviceId\\":\\"([a-zA-z0-9-]+)"),0) | alter user = coalesce(arrayindex(regextract(_raw_log,"UserID\":\"(.+?)\""),0),arrayindex(regextract(_raw_log,"User\:\sOptional\[(.+?)\]"),0)) | filter device_id = "*" | comp count_distinct(user) as user_count by device_id | alter assignment_group = "AMP_ES&I_IDAM Technical Support_SNow", configuration_item = "MY AMP" | filter user_count >= 10 Drill Down: config case_sensitive = false | dataset = was_mobility_services_raw | alter device_id = arrayindex(regextract(_raw_log, "\\"DeviceId\\":\\"([a-zA-z0-9-]+)"),0) | alter url = coalesce(arrayindex(regextract(_raw_log, "endPointUrl \:(.*?)\s"),0), arrayindex(regextract(_raw_log, "url: (.*?)\s"),0)) |filter device_id="*" | filter url!="https://esb-uat1.amp.com.au/*"
| tstats count where sourcetype=amp:was:mobility:* | where count>=9000000 Drill Down sourcetype=amp:was:mobility:*	dataset IN (was_mobility_audit_raw, was_mobility_device_raw,was_mobility_services_raw) |comp count() as count |filter count>=9000000 Drill Down dataset IN (was_mobility_audit_raw, was_mobility_device_raw,was_mobility_services_raw)
sourcetype=amp:was:mobility:services dest_asset_tag=prod session_id=* tam_id!=NOT_FOUND | stats dc(session_id) as count by tam_id | where count>=150 Drill Down sourcetype=amp:was:mobility:services url!=https://esb-uat1.amp.com.au/* session_id=* tam_id!=NOT_FOUND tam_id!=385f8b55-e8ce-4eed-ad93-c227943bb4f0 tam_id!=4f4de397-0976-45f6-a6eb-0c50821e6174	config case_sensitive = false | dataset = was_mobility_services_raw |filter dest IN ("secure.amp.com.au", "secure.amp.co.nz", "eam.isam.ampaws.com.au", "secure.amp.com.au", "ip-10-164-*.isam.ampaws.com.au", "ip-192-168-16-*.isam.ampaws.com.au", "ip-192-168-17-*.isam.ampaws.com.au", "ip-192-168-18-*.isam.ampaws.com.au") |filter action="failure" and tam_id!="NOT_FOUND" |comp count(session_id) as count by tam_id |filter count >=150 Drill Down config case_sensitive = false | dataset = was_mobility_services_raw |filter url!="https://esb-uat1.amp.com.au/*" AND session_id="*" |filter tam_id!="NOT_FOUND" OR tam_id!="385f8b55-e8ce-4eed-ad93-c227943bb4f0" OR tam_id!="4f4de397-0976-45f6-a6eb-0c50821e6174"
`o365_management_activity` user_agent="aadconnect/*"	datamodel dataset = msft_o365_azure_ad_raw | filter xdm.source.user_agent = "aadconnect/*" // value "aadconnect/*" not found | fields xdm.source.user_agent as user_agent, xdm.event.type as event_type, xdm.event.id as Id, xdm.source.ipv4 as IP, xdm.source.user.upn as UserId, xdm.source.user.username as UserName, xdm.event.original_event_type as Operation, xdm.event.outcome as ResultStatus, xdm.event.operation as properties, xdm.target.host.hostname as hostname, xdm.event.description as description, xdm.event.outcome_reason as reason

index=o365_prod sourcetype=o365:management:activity Workload=Threatintelligence Verdict=" Phish" NOT PolicyAction IN (MoveToJmf, Quarantine, Delete) NOT SenderIp IN ("104.40.0.0/13" "13.55.65.8" "13.55.54.143") Directionality=Inbound "Recipients{}"!="azuresecops@amp.com.au" | stats values(MessageTime) as MessageTime values(AttachmentData{}.FileName) as FileNames values(AttachmentData{}.FileType) as FileType values(AttachmentData{}.SHA256) as Hash_Value values(DetectionType) as DetectionType values(P1Sender) as P1Sender values(SenderIp) as SenderIp values(Subject) as Subject values(signature) as signature values(EventDeepLink) as Link values(DeliveryAction) as DeliveryAction count by Recipients{} | rename Recipients{} as user	// Title: [O365] Allowed Phish [Helix] // Description: This rule triggers on unblocked email that Office 365 Threat Intelligence has flagged as phishing (Sourcetype need to be replaced with the Proofpoint in XSIAM) // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: msft_o365_general_raw // Date: 26/June/2024 config case_sensitive = false | dataset = msft_o365_general_raw | filter Workload = "Threatintelligence" | filter Verdict = "Phish" | filter PolicyAction not in ("MoveToJmf", "Quarantine", "Delete") | filter SenderIp not in ("13.55.65.8", "13.55.54.143") and not incidr(SenderIp , "104.40.0.0/13") | filter Directionality = "Inbound" | alter Recipients = Recipients -> [] | alter Recipients = replace(arrayindex(Recipients, 0), "\"", "") | filter Recipients not contains "azuresecops@amp.com.au" | alter AttachmentData = AttachmentData -> [] | alter FileNames = to_string(arraymap(AttachmentData, if("@element"->FileName = "*", "@element"->FileName))) | alter FileTypes = to_string(arraymap(AttachmentData, if("@element"->FileType = "*", "@element"->FileType))) | alter HashValues = to_string(arraymap(AttachmentData, if("@element"->SHA256 = "*", "@element"->SHA256))) | alter policyName = PolicyDetails -> PolicyName, policyRule = json_extract_scalar(PolicyDetails, "$.0.Rules.0.RuleName") | alter signature1 = if(policyName != null and Operation in ("DLPRuleMatch") and Workload in ("SharePoint", "OneDrive"), policyName, null), signature2 = if(policyRule != null and Operation in ("DLPRuleUndo") and Workload in ("OneDrive"), policyRule, Operation) | alter signature = coalesce(signature1, signature2) | join type = left (dataset = splunk_ta_o365_cim_authentication_RecordType | fields record_type, RecordType as RecordTypeNum, RecordTypeDescription) as auth_record_type auth_record_type.RecordTypeNum = to_string(recordType) | comp count() as total_event, min(_time) as first_event_time, max(_time) as last_event_time, values(MessageTime) as MessageTime, values(FileNames) as FileNames, values(FileTypes) as FileType, values(HashValues) as HashValues, values(DetectionType) as DetectionType, values(P1Sender) as P1Sender, values(SenderIp) as SenderIp, values(Subject) as Subject, values(EventDeepLink) as Link, values(DeliveryAction) as DeliveryAction, values(Signature) as Signature, values(Verdict) as Verdict, values(DetectionMethod) as DetectionMethod, values(Id) as Id, values(Operation) as Operation, values(record_type) as RecordType, values(RecordTypeDescription) as RecordTypeDescription, values(Connector) as Connector by Recipients, Workload, Directionality | fields total_event, MessageTime, first_event_time, last_event_time, FileNames, FileType, HashValues, Signature, DetectionType, P1Sender, SenderIp, Subject, Link, DeliveryAction, Recipients as user, Verdict, DetectionMethod, Workload, Directionality, Id, Operation, RecordType, RecordTypeDescription, Connector
`o365_management_activity` Operation="consent to application." "Offline_Access"	// Title: Threat - [O365] Consent To Application With Offline Access [Helix] - Rule config case_sensitive = false | dataset = msft_o365_azure_ad_raw | filter Operation contains "consent to application" | filter ModifiedProperties contains "Offline_Access" | fields Operation, ModifiedProperties, CreationTime, Workload, ApplicationId, ExtendedProperties, ExtendedProperties_KeepMeSignedIn, `Target` , *

`o365_management_activity` Operation="Set-AdminAuditLogConfig" "{\"Name\": \"AdminAuditLogEnabled\", \"Value\": \"False\"}"	// Title: Threat - [O365] Disable Audit Log [Helix] - Rule // Description: This alert identifies Office 365 logs where the admin audit log is disabled. This may be indicative of an attacker attempting to remove visibility into the system activity. Verify user and source IPv4 and confirm that this was done intentionally and authorized. // Author: Mandeep Singh, msingh8@paloaltonetworks.com // Datasets: msft_o365_exchange_online_raw // Date: 08/July/2024 /* `o365_management_activity` Operation="Set-AdminAuditLogConfig" "{\"Name\": \"AdminAuditLogEnabled\", \"Value\": \"False\"}" */ config case_sensitive = false | dataset = msft_o365_exchange_online_raw | filter Operation = "Set-AdminAuditLogConfig" | alter Parameters = Parameters -> [] | alter AdminAuditLogEnabled = to_string(arraymap(Parameters, if("@element" -> Name = "AdminAuditLogEnabled", "@element" -> Value))) | filter AdminAuditLogEnabled contains "False" | fields Operation, AdminAuditLogEnabled, Parameters, Id, UserId, UserType, ResultStatus , OperationProperties, Sender , LogonUserDisplayName , LogonUserSid, IncidentId , ClientIP , ClientIPAddress

index=o365_prod sourcetype=o365:management:activity Workload="threatintelligence" Verdict=" malware" AND NOT PolicyAction IN(delete, quarantine, replaceattachment) DeliveryAction=Delivered|rename Recipients{} as Recipients P1Sender as Sender |stats values (AttachmentData{}.FileName) as FileNames values(AttachmentData{}.FileType) as FileType values (AttachmentData{}.SHA256) as Hash_Value values(DetectionType) as DetectionType values (SenderIp) as SenderIp values(signature) as signature values(EventDeepLink) as Link values (DeliveryAction) as DeliveryAction count by Recipients Sender

Drill Down query

[{"name":"[O365] Malware observed on the $Recipients$ mailbox","search":"index=o365_prod sourcetype=o365:management:activity Workload=\"threatintelligence\" Verdict=\"malware\" AND NOT PolicyAction IN(delete, quarantine, replaceattachment) DeliveryAction=Delivered $Recipients$ |rename Recipients{} as Recipients P1Sender as Sender |stats values(AttachmentData{}.FileName) as FileNames values(AttachmentData{}.FileType) as FileType values(AttachmentData{}.SHA256) as Hash_Value values(DetectionType) as DetectionType values(SenderIp) as SenderIp values (signature) as signature values(EventDeepLink) as Link values(DeliveryAction) as DeliveryAction count by Recipients Sender","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]

```
// Title: Threat - [O365] Email Malware [Helix] - Rule
// Description: This rule triggers on Office 365 Threat Intelligence malware alerts, specifically for email that has not been deleted, quarantined, or had the attachments replaced per the filtering policy.
// Author: Mandeep Singh, msingh8@paloaltonetworks.com
// Datasets: msft_o365_general_raw
// Date: 08/July/2024

/*
index=o365_prod sourcetype=o365:management:activity Workload="threatintelligence" Verdict="malware" AND NOT PolicyAction IN(delete, quarantine, replaceattachment) DeliveryAction=Delivered
|rename Recipients{} as Recipients P1Sender as Sender
|stats values(AttachmentData{}.FileName) as FileNames values(AttachmentData{}.FileType) as FileType values (AttachmentData{}.SHA256) as Hash_Value values(DetectionType) as DetectionType values(SenderIp) as SenderIp values(signature) as signature values(EventDeepLink) as Link values(DeliveryAction) as DeliveryAction count by Recipients Sender
*/
dataset = msft_o365_general_raw
| filter Workload = "threatintelligence"
| filter Verdict = "malware"
| filter PolicyAction not in ("delete", "quarantine", "replaceattachment")
| filter DeliveryAction="Delivered"
| alter policyName = PolicyDetails -> PolicyName,
       policyRule = json_extract_scalar(PolicyDetails, "$.0.Rules.0.RuleName")
| alter signature1 = if(policyName != null and Operation in ("DLPRuleMatch") and Workload in ("SharePoint", "OneDrive"), policyName, null),
       signature2 = if(policyRule != null and Operation in ("DLPRuleUndo") and Workload in ("OneDrive"), policyRule, Operation)
| alter signature = coalesce(signature1, signature2 )
| alter AttachmentData = AttachmentData->[]
| alter FileName = to_string(arraymap(AttachmentData , "@element"->FileName))
| alter FileType  = to_string(arraymap(AttachmentData , "@element"->FileType))
| alter SHA256   = to_string(arraymap(AttachmentData , "@element"->SHA256))
| fields P1Sender as Sender, *
| comp min(_time) as firstEventTime, max(_time) as lastEventTime, values(FileName) as FileNames, values(FileType) as FileType, values(SHA256) as Hash_Value, values(DetectionType) as DetectionType, values(SenderIp) as SenderIp, values(signature) as signature, values(EventDeepLink) as Link, values(DeliveryAction) as DeliveryAction, count() as count by Recipients, Sender
```

Drill Down query

```
dataset = msft_o365_general_raw
| alter name = format_string("[O365] Malware observed on the %s mailbox", $Recipients)
| filter Workload = "threatintelligence"
| filter Verdict = "malware"
| filter PolicyAction not in ("delete", "quarantine", "replaceattachment")
| filter DeliveryAction="Delivered" and Recipients = $Recipients
| alter policyName = PolicyDetails -> PolicyName,
       policyRule = json_extract_scalar(PolicyDetails, "$.0.Rules.0.RuleName")
| alter signature1 = if(policyName != null and Operation in ("DLPRuleMatch") and Workload in ("SharePoint", "OneDrive"), policyName, null),
       signature2 = if(policyRule != null and Operation in ("DLPRuleUndo") and Workload in ("OneDrive"), policyRule, Operation)
| alter signature = coalesce(signature1, signature2 )
| alter AttachmentData = AttachmentData->[]
| alter FileName = to_string(arraymap(AttachmentData , "@element"->FileName))
| alter FileType  = to_string(arraymap(AttachmentData , "@element"->FileType))
| alter SHA256   = to_string(arraymap(AttachmentData , "@element"->SHA256))
| fields P1Sender as Sender, *
| comp values(FileName) as FileNames, values(FileType) as FileType, values(SHA256) as Hash_Value, values (DetectionType) as DetectionType, values(SenderIp) as SenderIp, values(signature) as signature, values (EventDeepLink) as Link, values(DeliveryAction) as DeliveryAction, count() as count by Recipients, Sender
```

```
`o365_management_activity` Workload="exchange" AND user_priority="critical" AND Operation IN ( updateinboxrules, addfolderpermissions,removefolderpermissions, modifyfolderpermissions, add-mailboxfolderpermission, remove-mailboxfolderpermission, add-mailboxpermission, remove-mailboxpermission ) AND ResultStatus="Succeeded" |stats values(CreationTime) as CreationTime values(ClientIPAddress) as ClientIPAddress values(ClientProcessName) as ClientProcessName values(Item.ParentFolder.Name) as ParentFolder values(Operation) as Operation values (OrganizationName) as OrganizationName values(record_type) as Record_Type count by MailboxOwnerUPN
| rename MailboxOwnerUPN as user

Drill Down Query
[{"name":"Executive Mailbox and Folder Permission Change by $user$","search":"
`o365_management_activity` Workload=\"exchange\" AND user_priority=\"critical\" AND Operation IN ( updateinboxrules, addfolderpermissions,removefolderpermissions, modifyfolderpermissions, add-mailboxfolderpermission, remove-mailboxfolderpermission, add-mailboxpermission, remove-mailboxpermission ) AND ResultStatus=\"Succeeded\" $user$ |stats values(CreationTime) as CreationTime values(ClientIPAddress) as ClientIPAddress values(ClientProcessName) as ClientProcessName values(Item.ParentFolder.Name) as ParentFolder values(Operation) as
```

```
// Title:[AMP] Threat - [O365] Executive Mailbox and Folder Permission Change [HX] - Rule
// Description: This rule detects modifications of mailbox permissions or folder permissions inside a mailbox. This is limited to high-value executive user mailboxes. This may be benign activity. External attackers or a malicious insider might change permissions to allow access to view sensitive messages or for insider trading. Verify the source IP, the type of logon (Owner, Delegate or Admin) and hour of operation.
// Author: Sahil Sharma, ssharma7@paloaltonetworks.com
// Datasets: msft_o365_exchange_online_raw
// Date: 01/Jul/2024

config case_sensitive = false
| dataset = msft_o365_exchange_online_raw
| filter Workload = "Exchange"
| filter LogonType in (0,1,2) // 0 -> owner, 1 -> admin, 2 -> delegate)
| filter Operation in("UpdateInboxRules", "AddFolderPermissions", "RemoveFolderPermissions", "ModifyFolderPermissions", "Remove-MailboxPermission","Remove-MailboxFolderPermission", "Add-MailboxPermission", "add-mailboxfolderpermission")
| filter ResultStatus = "Succeeded"

| alter parentFolder = json_extract_scalar(Item, "$.ParentFolder.Name")

| alter OperationPropertiesArray = OperationProperties -> []
| alter OperationPropertiesData = arraystring(arraymap(OperationPropertiesArray, format_string("%s: %s", json_extract_scalar("@element", "$.Name"), json_extract_scalar("@element", "$.Value"))), ",")

| join type = left (dataset = splunk_ta_o365_cim_authentication_RecordType
    | fields record_type, RecordType as RecordTypeNum, RecordTypeDescription
) as auth_record_type auth_record_type.RecordTypeNum = to_string(recordType)

| comp min(_time) as startTime, max(_time) as endTime, count() as count, values(ClientIPAddress) as clientIpAddress, values(OperationPropertiesData) as operationProperties, values(ClientProcessName) as clientProcessName, values (Operation) as operation, values(parentFolder) as parentFolder, values(OrganizationName) as organizationName, values(record_type) as record_type, values(RecordTypeDescription) as RecordTypeDescription by MailboxOwnerUPN, workload, ResultStatus, userid

| join ( dataset = pan_dss_raw
    | filter member_of contains "VIP_Users_Snow"
    | fields upn, member_of, email
) as vip_users MailboxOwnerUPN = vip_users.upn

| fields startTime, endTime,  MailboxOwnerUPN as user,clientIpAddress, clientProcessName, parentFolder,operation, organizationName, record_type, RecordTypeDescription, parentFolder,count ,email, member_of, Workload, ResultStatus, operationProperties, userid

Drill Down Query
config case_sensitive = false
| dataset = msft_o365_exchange_online_raw
| alter name = format_string("Executive Mailbox and Folder Permission Change by %s",$user)
| filter Workload = "Exchange"
| filter LogonType in (0,1,2) // 0 -> owner, 1 -> admin, 2 -> delegate)
| filter Operation in("UpdateInboxRules", "AddFolderPermissions", "RemoveFolderPermissions", "ModifyFolderPermissions", "Remove-MailboxPermission","Remove-MailboxFolderPermission", "Add-MailboxPermission", "add-mailboxfolderpermission")
| filter ResultStatus = "Succeeded"
| filter MailboxOwnerUPN = $user
| alter parentFolder = json_extract_scalar(Item, "$.ParentFolder.Name")
| alter OperationPropertiesArray = OperationProperties -> []
| alter OperationPropertiesData = arraystring(arraymap(OperationPropertiesArray, format_string("%s: %s", json_extract_scalar("@element", "$.Name"), json_extract_scalar("@element", "$.Value"))), ",")

| join type = left (dataset = splunk_ta_o365_cim_authentication_RecordType
    | fields record_type, RecordType as RecordTypeNum, RecordTypeDescription
) as auth_record_type auth_record_type.RecordTypeNum = to_string(recordType)

| comp min(_time) as startTime, max(_time) as endTime, values(ClientIPAddress) as clientIpAddress, values (ClientProcessName) as clientProcessName, values(Operation) as operation, values(parentFolder) as parentFolder, values(OrganizationName) as organizationName, values(record_type) as record_type, values(RecordTypeDescription) as RecordTypeDescription by MailboxOwnerUPN

| join ( dataset = pan_dss_raw
    | filter member_of contains "VIP_Users_Snow"
```

`o365_management_activity` Workload="exchange" Operation="mailboxlogin" ClientInfoString IN ("microsoft.exchange.powershell", "microsoft winrm client")	// Title: Threat - [O365] External PowerShell Mailbox Access [Helix] - Rule config case_sensitive = false | dataset = msft_o365_exchange_online_raw | filter Workload = "exchange" | filter Operation contains "mailboxlogin" | filter ClientInfoString contains "microsoft.exchange.powershell" or ClientInfoString contains "microsoft winrm client" | fields _time, Workload, Operation, ClientIP, UserId, ResultStatus ,ClientInfoString, MailboxOwnerUPN, LogonType,*
`o365_management_activity` Workload="Exchange" (Operation="New-InboxRule" OR Operation="Set-InboxRule") ("{\"Name\": \"MoveToFolder\", \"Value\": \"Deleted Items\"}" OR "{\"Name\": \"DeleteMessage\", \"Value\": \"True\"}") ("Hack" OR "Hacked" OR "Virus" OR "Spam") Drill Down `o365_management_activity` Workload="Exchange" (Operation="New-InboxRule" OR Operation="Set-InboxRule") ("{\"Name\": \"MoveToFolder\", \"Value\": \"Deleted Items\"}" OR "{\"Name\": \"DeleteMessage\", \"Value\": \"True\"}") ("Hack" OR "Hacked" OR "Virus" OR "Spam")	// Title: Threat - [O365] Inbox Rule Contains Hack, Hacked, Virus, Spam [Helix] - Rule // Description: This rule alerts whenever a new inbox rule is created to automatically delete messages containing the words: Hack, Hacked, Virus, Spam // Author: Mandeep Singh, msingh8@paloaltonetworks.com // Reviewer: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: msft_o365_exchange_online_raw // Date: 08/July/2024 config case_sensitive = false | dataset = msft_o365_exchange_online_raw | filter Workload = "Exchange" | filter Operation = "New-InboxRule" OR Operation="Set-InboxRule" | alter Parameters = Parameters ->[] | alter ruledata=arraymap(Parameters,concat("@element"->Name,":","@element"->Value)) // creating key:value pairs of data within array | filter ruledata contains "MoveToFolder:Deleted Items" or ruledata contains "DeleteMessage:True" | alter bodyContainsWords=arraystring(arraymap(Parameters,if("@element"->Name = "BodyContainsWords", "@element"->Value)) , ",")// fetching value of bodyContainswords | filter bodyContainsWords in ("*Hack*", "*Hacked*", "*Virus*", "*Spam*") | join type = left (dataset = splunk_ta_o365_cim_authentication_RecordType | fields record_type, RecordType as RecordTypeNum, RecordTypeDescription) as auth_record_type auth_record_type.RecordTypeNum = to_string(recordType) | fields _time, CreationTime, UserId, UserType, AppId, AppPoolName, ClientIP, Workload, ruleData, Operation, parameters, ResultStatus, ObjectId, id, record_type, RecordTypeDescription, SessionId, RequestId Drill Down config case_sensitive = false | dataset = msft_o365_exchange_online_raw | filter Workload = "Exchange" | filter Operation = "New-InboxRule" OR Operation="Set-InboxRule" | alter Parameters = Parameters ->[] | alter ruledata=arraymap(Parameters,concat("@element"->Name,":","@element"->Value)) // creating key:value pairs of data within array | filter ruledata contains "MoveToFolder:Deleted Items" or ruledata contains "DeleteMessage:True" | alter bodyContainsWords=arraystring(arraymap(Parameters,if("@element"->Name = "BodyContainsWords", "@element"->Value)) , ",")// fetching value of bodyContainswords | filter bodyContainsWords in ("*Hack*", "*Hacked*", "*Virus*", "*Spam*") | join type = left (dataset = splunk_ta_o365_cim_authentication_RecordType | fields record_type, RecordType as RecordTypeNum, RecordTypeDescription) as auth_record_type auth_record_type.RecordTypeNum = to_string(recordType) | fields _time, CreationTime, UserId, UserType, AppId, AppPoolName, ClientIP, Workload, ruleData, Operation, parameters, ResultStatus, ObjectId, id, record_type, RecordTypeDescription, SessionId, RequestId

`o365_management_activity` Workload="Exchange" (Operation="New-InboxRule" OR Operation="Set-InboxRule") ("{\"Name\": \"MoveToFolder\", \"Value\": \"Deleted Items\"}" OR "{\"Name\": \"DeleteMessage\", \"Value\": \"True\"}") NOT Parameters{}.Name IN("From", "SubjectContainsWords", "FromAddressContainsWords") Drill Down: `o365_management_activity` Workload="Exchange" (Operation="New-InboxRule" OR Operation="Set-InboxRule") ("{\"Name\": \"MoveToFolder\", \"Value\": \"Deleted Items\"}" OR "{\"Name\": \"DeleteMessage\", \"Value\": \"True\"}") NOT Parameters{}.Name IN("From", "SubjectContainsWords", "FromAddressContainsWords")	// Title: Threat - [O365] Inbox Rule Delete [Helix] - Rule // Description: This rule detects any new inbox rule that's designed to automatically delete messages. This may indicate a user has been compromised, used the account to phish others in the organization, and is covering their tracks. Inspect the filtering terms used and source IP of the activity for signs of suspicious behavior. // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: msft_o365_exchange_online_raw // Date: 03/July/2024 config case_sensitive = false | dataset = msft_o365_exchange_online_raw | filter Workload = "Exchange" | filter Operation = "New-InboxRule" OR Operation="Set-InboxRule" | alter data_array = Parameters -> [], parentFolder = json_extract_scalar(Item, "$.ParentFolder.Name") | alter Name = arraymap(data_array, "@element"->Name) | alter ruledata = arraymap(data_array, concat("@element" -> Name,":", "@element" -> Value)) // creating key:value pairs of data within array | filter ruledata contains "MoveToFolder:Deleted Items" or ruledata contains "DeleteMessage:True" | filter Name not in ("From", "SubjectContainsWords", "FromAddressContainsWords") | fields UserId, ClientIP, Workload, Name, ruleData, Operation, ResultStatus, parameters, OperationProperties, ResultStatus, ClientProcessName, parentFolder, OrganizationName, OrganizationId, MailboxOwnerUPN, * Drill Down Query config case_sensitive = false | dataset = msft_o365_exchange_online_raw | filter Workload = "Exchange" | filter Operation = "New-InboxRule" OR Operation="Set-InboxRule" | alter data_array = Parameters -> [] | alter Name = arraymap(data_array, "@element"->Name) | alter ruledata = arraymap(data_array, concat("@element" -> Name,":", "@element" -> Value)) // creating key:value pairs of data within array | filter ruledata contains "MoveToFolder:Deleted Items" or ruledata contains "DeleteMessage:True" | filter Name not in ("From", "SubjectContainsWords", "FromAddressContainsWords") | fields UserId, ClientIP, Workload,Name, ruleData, Operation, ResultStatus, parameters

`o365_management_activity` Workload=exchange Operation="set-mailboxauditbypassassociation" |stats values(Identity) as Target_User values(ClientIP) as IP_Address values(command) as Command_Name values(OriginatingServer) as Orginisation_Server values(ResultStatus) as Action values(status) as Status count by user_id AuditBypassEnabled |rename user_id as Modified_user Drill Down [{"name":"Contriburing logs modified by the $Modified_user$","search":"`o365_management_activity` Workload=exchange Operation=\"set-mailboxauditbypassassociation\" $Modified_user$","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]	// Title: Threat - [O365] Mailbox Audit Bypass [Helix] - Rule // Description: This rule detects an Exchange mailbox that was configured to bypass mailbox audit logging. This can be used to reduce noise for accounts that generate numerous logs. This could also be used by an attacker to cover their tracks.(Sourcetype need to be replaced with the Proofpoint in XSIAM) // Datasets: msft_o365_exchange_online_raw // Date: 23/July/2024 config case_sensitive = false | dataset = msft_o365_exchange_online_raw | filter Workload = "exchange" | filter Operation = "set-mailboxauditbypassassociation" | alter Parameters_array = Parameters -> [] | alter Identity = arraystring(arraymap(Parameters_array, if("@element"->Name = "Identity", "@element"->Value)), ",") | fields Parameters, Identity, ClientIP, Id, * | comp values(Identity) as Target_User, values(Id) as id, values(ClientIP) as IP_Address, values(CommandType) as command_type, values(OriginatingServer) as Orginisation_Server, values(ResultStatus) as Action, count() as total_events, min(_time) as firstTime, max(_time) as lastTime by UserId //, AuditBypassEnabled | fields total_events, firstTime, lastTime, Target_User, UserId as Modified_user, IP_Address, Orginisation_Server, Action, id, command_type Drill Down config case_sensitive = false | dataset = msft_o365_exchange_online_raw | alter name = format_string("Contriburing logs modified by the %s", UserId) | filter Workload = "exchange" | filter Operation = "set-mailboxauditbypassassociation" | filter UserId = $Modified_user

```
index=o365_prod sourcetype=o365:management:activity Operation IN(modifyfolderpermissions,
addfolderpermissions) AND Item.ParentFolder.Name IN(Inbox, "Top of Information Store") AND NOT
Item.ParentFolder.MemberRights IN(ReadAny, Visible, FreeBusySimple, FreeBusyDetailed)
ClientInfoString!="Client=WebServices;Action=ConfigureGroupMailbox" LogonType!=2 "Item.
ParentFolder.MemberUpn"=Everyone
| rename Item.ParentFolder.MemberUpn as MemberUpn
| rename Item.ParentFolder.Name as MailboxFolder
| rename Item.ParentFolder.MemberRights as MemberRights
| `security_content_ctime(_time)`
| rename _time as Timestamp
| table Timestamp MailboxOwnerUPN Operation MemberUpn MemberRights LogonType
MailboxFolder ClientInfoString ClientIP

Drill Down Query
`o365_management_activity` Operation IN(modifyfolderpermissions, addfolderpermissions) AND
Item.ParentFolder.Name IN(Inbox, "Top of Information Store") AND NOT Item.ParentFolder.
MemberRights IN(ReadAny, Visible, FreeBusySimple, FreeBusyDetailed) ClientInfoString!="
Client=WebServices;Action=ConfigureGroupMailbox" LogonType!=2
```

```
// Title: [O365] Mailbox Permission Change - Everyone Allowed [Helix]
// Description: This rule detects when the "Everyone" user is given excessive permissions to a user's mailbox. This
technique has been used by attackers to gain access to a mailbox without needing to login interactively as that user.
Within a Microsoft Exchange mailbox, granular access can be granted to individual folders by modifying folder
permissions. These permissions can be granted directly by the mailbox owner, or on behalf of a mailbox owner by
either an account that has been delegated access to the mailbox or an Exchange administrator. For each mailbox, the
user-accessible folders (e.g., Inbox, Sent Items, Deleted Items) reside within a hidden folder root called the "Top of
Information Store". Just like user-visible folders, folder permissions can be granted to it.
// Author: Sahil Sharma, ssharma7@paloaltonetworks.com
// Datasets: msft_o365_exchange_online_raw
// Date: 29/Apr/2024

config case_sensitive = false
| dataset = msft_o365_exchange_online_raw
| filter Operation in ("modifyfolderpermissions", "addfolderpermissions")
| filter ResultStatus = "Succeeded"
| filter ClientInfoString != "Client=WebServices;Action=ConfigureGroupMailbox"
| filter LogonType != 2

// extract item parent folder information
| alter mailbox_folder = Item -> ParentFolder.Name
| alter member_rights = Item -> ParentFolder.MemberRights
| alter member_upn = Item -> ParentFolder.MemberUpn

| filter mailbox_folder in ("Inbox", "Top of Information Store")

| alter member_rights_exist = if(member_rights in ("ReadAny", "Visible", "FreeBusySimple", "FreeBusyDetailed"), true,
false)
| filter member_rights_exist = false
| filter member_upn = "Everyone"

| fields CreationTime, ClientIP, UserId, LogonType, ClientInfoString, Operation, ResultStatus, mailbox_folder,
member_rights, member_rights_exist, Item, member_upn, workload

Drill Down Query
config case_sensitive = false
| dataset = msft_o365_exchange_online_raw
| filter Operation in ("modifyfolderpermissions", "addfolderpermissions")
| filter ResultStatus = "Succeeded"
| filter ClientInfoString != "Client=WebServices;Action=ConfigureGroupMailbox"
| filter LogonType != 2
// extract item parent folder information
| alter mailbox_folder = Item -> ParentFolder.Name
| alter member_rights = Item -> ParentFolder.MemberRights
| alter member_upn = Item -> ParentFolder.MemberUpn
| filter mailbox_folder in ("Inbox", "Top of Information Store")
| alter member_rights_exist = if(member_rights in ("ReadAny", "Visible", "FreeBusySimple", "FreeBusyDetailed"), true,
false)
| filter member_rights_exist = false
```

`o365_management_activity` Workload="Exchange" (Operation="New-InboxRule" OR Operation="Set-InboxRule") "{\"Name\": \"MoveToFolder\", \"Value\": \"RSS Subscriptions\"}" NOT Parameters{}.Name IN("From", "SubjectContainsWords", "FromAddressContainsWords") Drill Down Query `o365_management_activity` Workload="Exchange" (Operation="New-InboxRule" OR Operation="Set-InboxRule") "{\"Name\": \"MoveToFolder\", \"Value\": \"RSS Subscriptions\"}" NOT Parameters{}.Name IN("From", "SubjectContainsWords", "FromAddressContainsWords")	// Title: [O365] New Inbox Rule - MoveToFolder RSS Subscriptions [Helix] config case_sensitive = false | dataset = msft_o365_exchange_online_raw | filter Workload = "Exchange" | filter Operation in ("New-InboxRule", "Set-InboxRule") | alter data_array = Parameters -> [] | alter Name = arraymap(data_array, "@element" -> Name) | alter ruledata = arraymap(data_array, concat("@element" -> Name, ":", "@element" -> Value)) // creating key:value pairs of data within array | filter ruledata contains "MoveToFolder:RSS Subscriptions" | filter Name not in ("From", "SubjectContainsWords", "FromAddressContainsWords") | fields UserId, ClientIP, Workload,Name, Operation, ResultStatus, name, Parameters, ruledata, * Drill Down Query config case_sensitive = false | dataset = msft_o365_exchange_online_raw | filter Workload = "Exchange" | filter Operation in ("New-InboxRule", "Set-InboxRule") | alter data_array = Parameters -> [] | alter Name = arraymap(data_array, "@element" -> Name) | alter ruledata = arraymap(data_array, concat("@element" -> Name, ":", "@element" -> Value)) // creating key:value pairs of data within array | filter ruledata contains "MoveToFolder:RSS Subscriptions" | filter Name not in ("From", "SubjectContainsWords", "FromAddressContainsWords") | fields UserId, ClientIP, Workload,Name, Operation, ResultStatus, name, Parameters, ruledata, *
`o365_management_activity` Operation IN("remove-antiphish", "disable-antiphish*", "disable-safelinks*", "remove-safelinks*", "disable-safeattachment*", "remove-safeattachment*", "remove-dlp*", "disable-dlp*")	// Title: Threat - [O365] Policy Tampered [Helix] - Rule config case_sensitive = false | dataset = msft_o365_general_raw | filter Operation contains "remove-antiphish" or operation contains "disable-antiphish*" or operation contains "disable-safelinks*" or operation contains "remove-safelinks*" or operation contains "disable-safeattachment*" or operation contains "remove-safeattachment*" or operation contains "remove-dlp*" or operation contains "disable-dlp*" | fields Operation, Parameters, Id, UserId, UserType, ResultStatus , OperationProperties, _reporting_device_name , _vendor ,_final_reporting_device_name , _id , _insert_time , _product ,_collector_name , _collector_type , DeviceId , * //, Sender , LogonUserDisplayName , LogonUserSid, IncidentId , ClientIP , ClientIPAddress ,*

```
// Title: Threat - [PAM] RDP Access to CyberArk Connector Server [eSecure] - Rule
// Description: Alert for direct RDP access to CyberArk connector servers. As per the design, BAU access to the connector servers should be coming from cross-cloud servers (Cloud-2 to Cloud-X & Cloud-X to Cloud-2). An alert should be raised for any connections coming to the servers with the below nominated accounts directly from the user PCs.
// Author: Mandeep Singh, msingh8@paloaltonetworks.com
// Reviewer: Sahil Sharma, ssharma7@paloaltonetworks.com
// Datasets: microsoft_windows_raw
// Date: 08/July/2024

config case_sensitive = false
| datamodel dataset = microsoft_windows_raw
| filter xdm.event.id = "4624"
| filter xdm.event.original_event_type = "Logon"
| filter xdm.logon.type = "REMOTE_INTERACTIVE"
| filter xdm.source.host.hostname  ~= "^CA.*"
| filter xdm.target.user.username !~= ".*\$$"

| alter matching_computer_name = arrayindex(split(xdm.source.host.hostname, "."), 0)

| join type = left (
    dataset = pam_excluded_ccoe_engineers
    | fields ID as Account_Name
) as pam_excluded_ccoe_engineers pam_excluded_ccoe_engineers.Account_Name = xdm.target.user.username
| filter Account_Name = null

| join type = left (
    dataset = pam_excluded_hosts
    | fields Excluded, Excluded_Hosts, Client_Name
) as pam_excluded_hosts pam_excluded_hosts.Client_Name contains matching_computer_name
| filter Excluded_Hosts = null and Excluded = null

| join type = left (
    dataset = pam_rdp_excluded_users
    | alter User = replace(User , "*", "")
    | fields User , Excluded_RDP
) as pam_rdp_excluded_users xdm.source.user.username contains pam_rdp_excluded_users.User
| filter Excluded_RDP = null

| comp count(xdm.source.user.username) as src_user_count, values(xdm.event.description) as message by _time, xdm.event.id, xdm.event.operation_sub_type, xdm.target.user.username, xdm.source.host.fqdn, xdm.source.host.hostname, xdm.logon.type, xdm.source.user.username, xdm.event.original_event_type, xdm.source.ipv4

| alter computer_name = xdm.source.host.hostname

| fields _time as time, src_user_count, xdm.event.id as event_code, xdm.event.operation_sub_type as signature, xdm.event.original_event_type as task_category, xdm.target.user.username as account_name, xdm.source.user.username as src_user, xdm.source.host.hostname as dest, xdm.source.host.fqdn as workstation_name, xdm.logon.type as logon_type, computer_name, xdm.source.ipv4 as src, message
| alter assignment_group = "AMP_PAM_PROD_Support_Snow", configuration_item = "PAM - Privileged Access Management"



Drill down query

/*config case_sensitive = false
| datamodel dataset = microsoft_windows_raw
| alter account_domain = regextract(xdm.event.description, "\tAccount Domain:\t\t([\w\s-]*)\r\n")
| alter SessionName = json_extract_scalar(microsoft_windows_raw.event_data, "$.SessionName")
| filter xdm.source.host.hostname = "CA-*" and SessionName = "RDP*" and account_domain != "CA-*"
// | filter src_category!="pam" // src_category field not found
| fields _time, xdm.source.user.username as src_user, xdm.source.host.hostname as computer_name, xdm.target.user.username as account_name, account_domain, SessionName, xdm.source.ipv4 as src, xdm.event.description as message*/

config case_sensitive = false
| datamodel dataset = microsoft_windows_raw

| alter AccountDomain = json_extract_scalar(microsoft_windows_raw.event_data, "$.AccountDomain"),
```

index=amp_pam_prod cef_signature=385 OR cef_signature=386 | stats count by _time, act, suser, shost, dvc, _raw | fields - count	// Title: [[PAM - Prod] Changes to CyberArk Master Policy [eSecure] config case_sensitive = false | datamodel dataset = cyber_ark_vault_raw | filter xdm.event.id in ("385", "386") | comp count() as total_event by _time, xdm.event.operation, xdm.source.user.username, xdm.source.ipv4, xdm.target.ipv4, xdm.alert.description , xdm.event.id | fields _time as time, xdm.event.operation as act, xdm.source.user.username as suser, xdm.source.ipv4 as shost, xdm.target.ipv4 as dvc,xdm.event.id as signature, total_event,xdm.alert.description as _raw_log
index=amp_pam_prod Safe_Name=AMPAU-*-BG act="Retrieve password" | lookup pam_excluded_users "suser" as "suser" OUTPUT Excluded | stats count by _time, suser, shost, duser, Safe_Name | fields - count	// Title: [PAM - Prod] Credential Retrieval from Break-Glass Safe [eSecure] // Title: [PAM - Prod] Credential Retrieval from Break-Glass Safe [eSecure] config case_sensitive = false | dataset = cyber_ark_vault_raw | filter cs2 contains "AMPAU-*-BG" | filter act= "Retrieve password" | fields suser , shost , duser , cs2 as Safe_name, act , * | join type = left (dataset = pam_excluded_users | alter suser1 = replace(suser, "*", "") | fields Excluded, suser1) as excluded_users suser contains excluded_users.suser1 | fields suser ,shost , duser , Safe_name , act ,* | comp count() as count by _time , suser ,duser ,Safe_name
index=amp_pam_prod act="CPM Change Password Failed" [| inputlookup pam_included_password_rotate_vaults | fields Safe_Name] | stats count by _time act suser shost duser Safe_Name | fields - count	// Title: [PAM - Prod] Password Rotation Failure in CyberArk [eSecure] // Title: [PAM - Prod] Password Rotation Failure in CyberArk [eSecure] config case_sensitive = false | dataset = cyber_ark_vault_raw | filter act= "CPM Change Password Failed" | fields suser , shost , duser , cs2, act , * | join (dataset = pam_included_password_rotate_vaults | fields Safe_Name) as password_rotate_vaults cs2 =password_rotate_vaults.Safe_Name | fields _time , act , suser ,shost ,duser ,Safe_name , cs2, * | comp count() as count by _time , act , suser ,shost ,duser, cs2 | fields _time , act , suser ,shost ,duser, cs2 as safe_name, count

index=amp_pam_prod act="Retrieve password" Safe_Name=* | lookup pam_included_application_vaults "Safe_Name" as "Safe_Name" OUTPUT Included_Vault | lookup pam_application_vaults_excluded_users "suser" as "suser" OUTPUT Excluded | where isnull(Excluded) | where isnotnull(Included_Vault) | stats count by _time, suser, duser, shost, Safe_Name | where count>0	// Title: Threat - [PAM - Prod] Sensitive Credential Retrieval from Application Safe [NTT] - Rule config case_sensitive = false | dataset = cyber_ark_vault_raw | filter act= "Retrieve password" | filter cs2 = "*" | fields _time , suser , duser , shost , cs2, * | join (dataset= pam_included_application_vaults | fields Safe_Name, Included_Vault) as pam_included_application_vaults pam_included_application_vaults.Safe_Name = cs2 | join (dataset = pam_application_vaults_excluded_users | alter suser = replace(suser, "*", "") | fields suser, Excluded) as pam_application_vaults_excluded_users suser contains pam_application_vaults_excluded_users.suser | fields _time , suser , duser , shost , cs2, * | filter Excluded in ("", null) | filter Included_Vault not in ("", null) | comp count() as count by _time , suser , duser , shost , cs2 | filter count >0 | fields _time , suser , duser , shost , cs2 as Safe_name, count
index=amp_pam_prod act="Retrieve password" Safe_Name=* | lookup pam_included_admin_vaults "Safe_Name" as "Safe_Name" OUTPUT Included_Vault | lookup pam_admin_vaults_excluded_users "suser" as "suser" OUTPUT Excluded | where isnull(Excluded) | where isnotnull(Included_Vault) | stats count by _time, suser, duser, shost, Safe_Name | where count>0	// Title: Threat - [PAM - Prod] Sensitive Credential Retrieval from Vault Admin Safe [NTT] - Rule config case_sensitive = false | dataset = cyber_ark_vault_raw | filter act= "Retrieve password" | filter cs2 = "*" | fields _time , suser , duser , shost , cs2, * | join (dataset = pam_included_admin_vaults | fields Safe_Name, Included_Vault) as admin_vaults admin_vaults.Safe_Name = cs2 | join type = left (dataset = pam_admin_vaults_excluded_users | alter suser1 = replace(suser, "*", "") | fields suser1, Excluded) as excluded_users suser contains excluded_users.suser1 | filter Excluded in (null,"") | filter Included_Vault not in (null,"") | comp count() as count by _time , suser , duser , shost , cs2 | filter count >0 | fields _time , suser , duser , shost , cs2 as Safe_name, count

```
source="WinEventLog:Security" Account_Name=pam*
| lookup pam_srv_acc.csv "sAMAccountName" as "Account_Name" OUTPUT Confirmed
| lookup pam_excluded_hosts "Client_Name" as "ComputerName" OUTPUT Excluded
| where isnotnull(Confirmed)
| where isnull(Excluded)
| stats count by _time Account_Name src_user ComputerName dest EventCode, signature

Drill Down Query
source="WinEventLog:Security" Account_Name=pam* ComputerName!=CA-* dest_asset_tag!
=svc_*_ad*
 | lookup pam_srv_acc.csv "sAMAccountName" as "Account_Name" | where isnotnull(Confirmed)
```

```
// Title: [PAM] CyberArk Account Session to Server not from CyberArk [eSecure]
// Description: Alert for usage of the reconciliation account (service account) for PAM. These accounts should be used
only from one of the CyberArk CPM servers, if the request is coming from outside of the CyberArk component servers
then an alert should be created.
// Author: Sahil Sharma, ssharma@paloaltonetworks.com
// Datasets: microsoft_windows_raw
// Date: 10/Sep/2024

config case_sensitive = false
| datamodel dataset = microsoft_windows_raw
| filter xdm.observer.type = "Microsoft-Windows-Security-Auditing"
| filter xdm.source.user.username ~= "^pam*"
| alter dest = xdm.source.host.hostname,
       matching_computer_name = arrayindex(split(xdm.source.host.hostname, "."), 0)

| join type = left (
    dataset = pam_srv_acc
    | fields sAMAccountName, Confirmed
) as pam_srv_acc pam_srv_acc.sAMAccountName = xdm.source.user.username

| join type = left (
    dataset = pam_excluded_hosts
    | alter matching_client_name = arrayindex(split(Client_Name, "."), 0)
    | fields Client_Name, Excluded, matching_client_name
) as pam_excluded_hosts pam_excluded_hosts.matching_client_name = matching_computer_name

| filter Excluded = null
| filter Confirmed != null

| fields _time as time, xdm.source.host.hostname as ComputerName, xdm.source.user.username as user_name, xdm.
source.user.sam_account_name as account_name, xdm.event.original_event_type as signature, xdm.event.description
as message, xdm.event.id as event_code, dest
| comp count() as total_event, values(message) as message by time, account_name, user_name, ComputerName,
event_code, signature, dest
| alter assignment_group = "AMP_PAM_PROD_Support_Snow", configuration_item = "PAM - Privileged Access
Management"


Drill Down Query
config case_sensitive = false
| datamodel dataset = microsoft_windows_raw
| filter xdm.observer.type = "Microsoft-Windows-Security-Auditing"
| filter xdm.source.user.username ~= "^pam*"
| alter ComputerName = xdm.source.host.hostname
| filter ComputerName != "CA-*"

| alter match_computer_name = arrayindex(split(ComputerName, "."), 0)
| join ( dataset = endpoints
    | fields endpoint_name, endpoint_id
    | getrole endpoint_id as endpoint_role
    | filter endpoint_role not contains "Domain Controllers"
) as endpoint endpoint.endpoint_name contains match_computer_name

| join type = left (
    dataset = pam_srv_acc
    | fields sAMAccountName, Confirmed
) as pam_srv_acc pam_srv_acc.sAMAccountName = xdm.source.user.username
| filter Confirmed != null

| fields xdm.source.user.username as user_name, xdm.source.user.sam_account_name as account_name,
ComputerName, endpoint_role, xdm.event.description as message, xdm.event.id as event_code, xdm.event.
original_event_type as signature
```

```
index=os_event_prod sourcetype=WinEventLog ComputerName=CA* TaskCategory=Logon app="
win:remote" EventCode=4624 NOT
    [| inputlookup pam_excluded_ccoe_engineers
    | rename ID as Account_Name
    | fields Account_Name]
| lookup pam_excluded_hosts "Client_Name" as "Workstation_Name" OUTPUT Excluded
```

```
// Title: Threat - [PAM] RDP Access to CyberArk Connector Server [eSecure] - Rule
// Description: Alert for direct RDP access to CyberArk connector servers. As per the design, BAU access to the
connector servers should be coming from cross-cloud servers (Cloud-2 to Cloud-X & Cloud-X to Cloud-2). An alert
should be raised for any connections coming to the servers with the below nominated accounts directly from the user
PCs.
// Author: Mandeep Singh, msingh8@paloaltonetworks.com
// Reviewer: Sahil Sharma, ssharma7@paloaltonetworks.com
// Datasets: microsoft_windows_raw
// Date: 08/July/2024

config case_sensitive = false
| datamodel dataset = microsoft_windows_raw
| filter xdm.event.id = "4624"
| filter xdm.event.original_event_type = "Logon"
| filter xdm.logon.type = "REMOTE_INTERACTIVE"
| filter xdm.source.host.hostname  ~= "^CA.*"
| filter xdm.target.user.username !~= ".*\$$"

| alter matching_computer_name = arrayindex(split(xdm.source.host.hostname, "."), 0)

| join type = left (
    dataset = pam_excluded_ccoe_engineers
    | fields ID as Account_Name
) as pam_excluded_ccoe_engineers pam_excluded_ccoe_engineers.Account_Name = xdm.target.user.username
| filter Account_Name = null

| join type = left (
    dataset = pam_excluded_hosts
    | fields Excluded, Excluded_Hosts, Client_Name
) as pam_excluded_hosts pam_excluded_hosts.Client_Name contains matching_computer_name
| filter Excluded_Hosts = null and Excluded = null

| join type = left (
    dataset = pam_rdp_excluded_users
    | alter User = replace(User , "*", "")
    | fields User , Excluded_RDP
) as pam_rdp_excluded_users xdm.source.user.username contains pam_rdp_excluded_users.User
| filter Excluded_RDP = null

| comp count(xdm.source.user.username) as src_user_count, values(xdm.event.description) as message by _time, xdm.
event.id, xdm.event.operation_sub_type, xdm.target.user.username, xdm.source.host.fqdn, xdm.source.host.hostname,
xdm.logon.type, xdm.source.user.username, xdm.event.original_event_type, xdm.source.ipv4

| alter computer_name = xdm.source.host.hostname

| fields _time as time, src_user_count, xdm.event.id as event_code, xdm.event.operation_sub_type as signature, xdm.
event.original_event_type as task_category, xdm.target.user.username as account_name, xdm.source.user.username
as src_user, xdm.source.host.hostname as dest, xdm.source.host.fqdn as workstation_name, xdm.logon.type as
logon_type, computer_name, xdm.source.ipv4 as src, message
| alter assignment_group = "AMP_PAM_PROD_Support_Snow", configuration_item = "PAM - Privileged Access
Management"
```

Drill down query

```
/*config case_sensitive = false
| datamodel dataset = microsoft_windows_raw
| alter account_domain = regextract(xdm.event.description, "\tAccount Domain:\t\t([\w\s-]*)\r\n")
| alter SessionName = json_extract_scalar(microsoft_windows_raw.event_data, "$.SessionName")
| filter xdm.source.host.hostname = "CA-*" and SessionName = "RDP*" and account_domain != "CA-*"
// | filter src_category!="pam" // src_category field not found
| fields _time, xdm.source.user.username as src_user, xdm.source.host.hostname as computer_name, xdm.target.user.
username as account_name, account_domain, SessionName, xdm.source.ipv4 as src, xdm.event.description as
message*/

config case_sensitive = false
| datamodel dataset = microsoft_windows_raw

| alter AccountDomain = json_extract_scalar(microsoft_windows_raw.event_data, "$.AccountDomain"),
```

sourcetype="o365:management:activity" Workload=AzureActiveDirectory Operation=UserLoginFailed ObjectId=*amp.privilegecloud.cyberark.com* | stats count by _time, ClientIP, UserId	// Title: [PAM] SAML Authentication Failures [eSecure] // Description: Authentication failures to the PAM solution. The authentication for AMP users is via SAML and so the source of authentication failure logs would be via AAD for Enterprise Application 'amp.privilegecloud.cyberark.com'. // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: msft_o365_azure_ad_raw // Date: 26/Jun/2024 config case_sensitive = false | dataset = msft_o365_azure_ad_raw | filter Workload = "AzureActiveDirectory" | filter Operation contains "UserLoginFailed" | filter ObjectId contains "*amp.privilegecloud.cyberark.com*" | fields ActorIpAddress, ClientIP, DeviceProperties_BrowserType as BrowserType, DeviceProperties_OS as OS, ErrorNumber, ExtendedProperties_UserAgent as UserAgent, ExtendedProperties_RequestType as RequestType, ExtendedProperties_ResultStatusDetail as RequestStatusDetail, LogonError, Operation, RecordType, ResultStatus, UserId, Workload, DeviceProperties_DisplayName as DisplayName, ObjectId | join type = left (dataset = splunk_ta_o365_cim_authentication_RecordType | fields record_type, RecordType as record_type_num, RecordTypeDescription) as record_type_mapping record_type_mapping.record_type_num = to_string(RecordType) | comp count() as total_failed_login, values(ObjectId) as ObjectId, values(LogonError) as LogonError, values(Operation) as Operation, values(ResultStatus) as Outcome, values(UserAgent) as UserAgent, values(DisplayName) as DisplayName, values(OS) as OS, last(Workload) as Workload, values(BrowserType) as BrowserType, values(RequestType) as RequestType, values(record_type) as RecordType, values(RecordTypeDescription) as RecordTypeDescription, values(ActorIpAddress) as ActorIpAddress by _time, ClientIP, UserId
index="ecrm_prod" sourcetype="sfdc:setupaudittrail" congasign Section="Manage Users" | stats count by _time Action Display Section | fields - count	// Title: Threat - [Salesforce] Congasign Account Activity [eSecure] - Rule /* index="ecrm_prod" sourcetype="sfdc:setupaudittrail" congasign Section="Manage Users" | stats count by _time Action Display Section | fields - count */ config case_sensitive = false | dataset = salesforce_audit_raw // congasign field not found | filter Section = "Manage Users" | fields Section , action, Display , CreatedById ,CreatedByIssuer ,_product ,* | comp count() as count, values(CreatedById) as createdById, values(CreatedByIssuer) as createdbyissuer, values(_product) as _product, min(_time) as start_time, max(_time) as end_time by Action , Display , Section //| fields -count | fields Section ,Action ,Display , createdById , createdbyissuer ,_product , start_time , end_time
index=amp_trust1_prod action=failure | stats count, min(_time) as early_time, max(_time) as last_time by user action user_email | `security_content_ctime(early_time)` | `security_content_ctime(last_time)` | where count >= 6 Drill Down index=amp_trust1_prod $user$	dataset = amp_trustone_raw | filter action="failure" | fields _raw_log , _raw_json , * | comp count() as count, min(_time) as early_time, max(_time) as last_time by user, action, user_id //| `security_content_ctime(early_time)` //| `security_content_ctime(last_time)` | filter count >= 6 | fields user , action , user_id as user_email, count, early_time , last_time Drill Down dataset = amp_trustone_raw | filter "$user" = user

source="WinEventLog:Security" EventCode=4720 OR (EventCode=4732 Administrators) | transaction Security_ID maxspan=180m | search EventCode=4720 (EventCode=4732 Administrators) | table _time EventCode Account_Name Target_Account_Name Message Drill Down Query source="WinEventLog:Security" EventCode=4720 OR (EventCode=4732 Administrators)	// Title: [Threat - [Windows] New Local Admin Account [eSecure] - Rule // Description: Local admin accounts are used by legitimate technicians, but they're also used by attackers. This search looks for newly created accounts that are elevated to local admins. // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: microsoft_windows_raw // Date: 26/June/2024 datamodel dataset = microsoft_windows_raw | filter microsoft_windows_raw._collector_type = "Cortex Agent" | filter xdm.event.id = "4720" or (xdm.event.id = "4732" and xdm.target.user.username = "Administrators") | alter username = if(xdm.event.id = "4720", xdm.target.user.username, xdm.event.id = "4732", json_extract_scalar (microsoft_windows_raw.event_data, "$.MemberName")), user_sid = if(xdm.event.id = "4720", json_extract_scalar(microsoft_windows_raw.event_data, "$.TargetSid"), xdm. event.id = "4732", json_extract_scalar(microsoft_windows_raw.event_data, "$.MemberSid")), group_name = if(xdm.event.id = "4732", xdm.target.user.username), group_sid = if(xdm.event.id = "4732", json_extract_scalar(microsoft_windows_raw.event_data, "$.TargetSid")) | fields xdm.event.id as event_id, username, group_name, xdm.source.user.username as source_username, user_sid, group_sid, xdm.event.description as message, xdm.source.host.hostname as hostname, xdm.event.original_event_type as signature | transaction user_sid span = 180m | filter _raw contains "4720" and _raw contains "4732" and _raw contains "Administrators" Drill down Query datamodel dataset = microsoft_windows_raw | filter microsoft_windows_raw._collector_type = "Cortex Agent" | filter xdm.event.id = "4720" or (xdm.event.id = "4732" and xdm.target.user.username = "Administrators")
| tstats summariesonly=true allow_old_summaries=true count from datamodel="Authentication"."Authentication" where Authentication.Logon_Type="2" AND host!=AG625106 AND host!=AG625086 AND (Authentication.user=A01R12 OR Authentication.user=A02R2Q) AND Authentication.dest!="rpagdv4b7.au.amp.local" by "Authentication.src","Authentication.user","Authentication.dest","host","Authentication.signature","Authentication.action","Authentication.Logon_Type" | rename "Authentication.src" as "src","Authentication.user" as "user","Authentication.dest" as "dest","Authentication.signature" as "signature","Authentication.action" as "action","Authentication.Logon_Type" as "logon_type" Drill Down | tstats summariesonly=true allow_old_summaries=true count from datamodel="Authentication"."Authentication" where Authentication.src=src Authentication.user=$user by "Authentication.src","Authentication.user","Authentication.dest","host","Authentication.signature","Authentication.action" | rename "Authentication.src" as "src","Authentication.user" as "user","Authentication.dest" as "dest","Authentication.signature" as "signature","Authentication.action" as "action"	config case_sensitive = false | dataset = microsoft_windows_raw |filter _collector_type = "XDR Collector" |alter logonType = event_data -> LogonType |filter host_name != "AG625086*" and host_name != "AG625106*" |filter logonType = "2" |alter user = event_data -> SubjectUserName |filter user in ("A01R12","A02R2Q") | filter user != "*$" |alter action=lowercase(arrayindex(regextract(keywords,"Audit\s(.+)\"\]"),0)) |alter signature=arrayindex(regextract(message, "([^\.]+)"), 0) |alter src= event_data -> WorkstationName |alter dest = _reporting_device_name | bin _time span=1h |comp count(), max(_time) as _time by src,dest,action,user,signature,logonType,host_name Drill Down config case_sensitive = false | dataset = microsoft_windows_raw |filter _collector_type = "XDR Collector" | filter src = $src | filter user = $user |alter action=lowercase(arrayindex(regextract(keywords,"Audit\s(.+)\"\]"),0)) |alter signature=arrayindex(regextract(message, "([^\.]+)"), 0) |alter dest = _reporting_device_name | comp count(), by src, user, dest, host, signature, action

index=aws_main_prod sourcetype=aws:cloudtrail ConsoleLogin "additionalEventData.MFAUsed"!=Yes "userIdentity.type"=IAMUser user_type!="SAML" src_ip IN (10.152.0.0/13 10.152.0.0/15 10.154.0.0/15 10.156.0.0/15 10.159.0.0/17 10.162.0.0/16 10.164.0.0/16 10.165.0.0/16 10.168.0.0/13 10.168.0.0/14) "detail.service.additionalInfo.sample"!=true | dedup userIdentity.arn sourceIPAddress | table _time "userIdentity.accountId" "userIdentity.arn" sourceIPAddress "responseElements.ConsoleLogin" "additionalEventData.MFAUsed" userIdentity.principalId	// Title: Threat - AMP - 03006 - INI - Log in with AWS without MFA [NTT] - Rule // Description: P3 - It shouldn't log in AWS console without MFA because the best practice is to enable MFA by default for all IAM users. // Datasets: amazon_aws_raw // Date: 23/July/2024 config case_sensitive = false | dataset = amazon_aws_raw | filter _collector_name contains "Cloudtrail" | filter eventName = "ConsoleLogin" //| filter "detail.service.additionalInfo.sample"!=true | alter MFAUsed = additionalEventData -> MFAUsed, userIdentity_type = userIdentity -> type, arn = userIdentity -> arn, accountId = userIdentity -> accountId | alter ConsoleLogin = responseElements -> ConsoleLogin | alter userIdentity_principalId = userIdentity -> principalId // | filter user_type!="SAML" // user_type Field not found | filter MFAUsed != "Yes" | filter userIdentity_type = "IAMUser" | filter incidr(sourceIPAddress, "10.152.0.0/13, 10.152.0.0/15, 10.154.0.0/15, 10.156.0.0/15, 10.159.0.0/17, 10.162.0.0/16, 10.164.0.0/16, 10.165.0.0/16, 10.168.0.0/13, 10.168.0.0/14") = true | dedup arn, sourceIPAddress | fields _time, userIdentity_principalId, accountId, arn, sourceIPAddress, ConsoleLogin, MFAUsed, awsRegion, eventName, eventSource, eventType, *
| tstats `security_content_summariesonly` values(Processes.dest) as dest values(Processes.user) as user values(Processes.process_name) as file_path min(_time) as firstTime max(_time) as lastTime from datamodel=Endpoint.Processes by Processes.user Processes.dest | rex field=file_path "(?<process_name>[^\\\]+$)" \| mvexpand file_path \| mvexpand process_name \| `drop_dm_object_name("Processes")` \| `security_content_ctime(firstTime)`\| `security_content_ctime(lastTime)` \| rex field=file_path "(?<file_name>[^\\\]+$)" | fields - process_name systemFile | dedup file_path | `ut_shannon(file_name)` | where ut_shannon > 7.2 Drill Down | tstats `security_content_summariesonly` values(Processes.dest) as dest values(Processes.user) as user values(Processes.process_name) as file_path min(_time) as firstTime max(_time) as lastTime from datamodel=Endpoint.Processes by Processes.user Processes.dest | rex field=file_path "(?<process_name>[^\\\]+$)" \| mvexpand file_path \| mvexpand process_name \| `drop_dm_object_name("Processes")` \| `security_content_ctime(firstTime)`\| `security_content_ctime(lastTime)` \| rex field=file_path "(?<file_name>[^\\\]+$)" | fields - process_name systemFile | dedup file_path | `ut_shannon(file_name)` | where ut_shannon > 7.2	dataset = microsoft_windows_raw | alter DestAddress = json_extract_scalar(event_data ,"$.DestAddress") // | filter (user not in (null, "")) | comp values(process_name) as file_path, min(_time) as firstTime ,max(_time) as lastTime by user,DestAddress | arrayexpand file_path | dedup file_path

| tstats `summariesonly` values(Web.http_method) as http_method values(web.dest) as dest from datamodel=Web.Web by Web.src, Web.http_method _time Web.user Web.bytes_out Web.action Web.status | `drop_dm_object_name("Web")` | search action!=log | rename dest as domain | lookup ip_intel domain OUTPUT domain ip threat_key description | where isnotnull(threat_key) | `security_content_ctime(_time)` | rename _time as Timestamp Drill Down | tstats `summariesonly` values(Web.http_method) as http_method values(web.dest) as dest from datamodel=Web.Web by Web.src, Web.http_method _time Web.user Web.bytes_out Web.action Web.status | `drop_dm_object_name("Web")` | search action!=log | rename dest as domain | lookup ip_intel domain OUTPUT domain ip threat_key description | where isnotnull(threat_key) | `security_content_ctime(_time)` | rename _time as Timestamp	xdm.target.host | filter dataset="imperva_waf_raw"
`wineventlog_security` EventCode=4720 OR (EventCode=4732 Group_Name=Administrators)| where NOT isnull(src_user) | where subject!="A user account was created" | search NOT host IN (ip-10-162-* ip-10-161-* ip-10-165-* ip-10-172-*) | regex host!="(?i)AMP\w{7,9}(ST|PT|BT|PD)\d{2}" | transaction member_id connected=false maxspan=180m | rename member_id as user | stats count min(_time) as firstTime max(_time) as lastTime by src_user dest subject user_email host member_obj_id | join member_obj_id [| search index=os_event_prod sourcetype=WinEventLog | fields user user_email user_bunit | rename user as member_obj_id, user_email as member_email, user_bunit as member_bunit | table member_obj_id member_email member_bunit] | `security_content_ctime(firstTime)`| `security_content_ctime(lastTime)` | `detect_new_local_admin_account_filter` Drill Down Query `wineventlog_security` EventCode=4720 OR (EventCode=4732 Group_Name=Administrators)| where NOT isnull(src_user) | where subject!="A user account was created" | search NOT host IN (ip-10-162-* ip-10-161-* ip-10-165-* ip-10-172-*) | regex host!="(?i)AMP\w{7,9}(ST|PT|BT|PD)\d{2}" | transaction member_id connected=false maxspan=180m | rename member_id as user | stats count min(_time) as firstTime max(_time) as lastTime by src_user dest subject user_email host member_obj_id | `security_content_ctime(firstTime)`| `security_content_ctime(lastTime)` | `detect_new_local_admin_account_filter`	// Title: Threat - AMP - 06003 - PRI - Detect Windows New Local Admin account [NTT] - Rule / config case_sensitive = false | datamodel dataset = microsoft_windows_raw | filter xdm.observer.type = "Microsoft-Windows-Security-Auditing" // Filter source type | filter xdm.event.id = "4720" or (xdm.event.id = "4732" and xdm.source.user.groups = "Administrators") // filter event id | filter xdm.source.user.username != null // filter for null value | filter xdm.event.original_event_type != "A user account was created" // filter for message | filter incidr(xdm.source.ipv4, "10.162.0.0/16") = false and incidr(xdm.source.ipv4, "10.161.0.0/16") = false and incidr (xdm.source.ipv4, "10.165.0.0/16") = false and incidr(xdm.source.ipv4, "10.172.0.0/16") = false// filter for ip range | filter xdm.source.host.hostname !~= "(?i)AMP\w{7,9}(ST|PT|BT|PD)\d{2}" | comp count(xdm.event.id) as event_count, min(_time) as firstTime, max(_time) as lastTime by xdm.source.user.username, xdm.observer.type, xdm.event.original_event_type, xdm.source.ipv4, xdm.source.host.hostname, xdm.event.id | fields xdm.event.id, xdm.event.original_event_type, xdm.observer.type, xdm.source.host.hostname, xdm.source.user.username, event_count, firstTime, lastTime

	config case_sensitive = false | datamodel dataset = microsoft_windows_raw | filter xdm.observer.type = "Microsoft-Windows-Security-Auditing" | filter microsoft_windows_raw.message contains "*\Microsoft\Windows NT\CurrentVersion\Image File Execution*" | filter microsoft_windows_raw.message contains "GlobalFlag" or microsoft_windows_raw.message contains "Debugger" or microsoft_windows_raw.message contains "DEBUG_PROCESS" or microsoft_windows_raw.message contains "DEBUG_ONLY_THIS_PROCESS" or microsoft_windows_raw.message contains "RegCreateKeyEx" or microsoft_windows_raw.message contains "RegSetValueEx" | fields _time as Timestamp, xdm.observer.type, xdm.event.id, xdm.event.description,xdm.event.operation_sub_type , xdm.event.original_event_type ,xdm.event.outcome ,xdm.source.process.executable.path ,xdm.source.process.name , xdm.source.user.sam_account_name ,xdm.source.user.scope , xdm.source.user.username , xdm.source.host. hostname, xdm.target.user.username, xdm.target.process.name, xdm.target.process.command_line, xdm.target. process.executable.path, microsoft_windows_raw.message ,*
index=os_event_prod source="WinEventLog:Security" "*\\Microsoft\\Windows NT\\CurrentVersion\\Image File Execution*" AND (GlobalFlag OR Debugger OR DEBUG_PROCESS OR DEBUG_ONLY_THIS_PROCESS OR RegCreateKeyEx OR RegSetValueEx) | `security_content_ctime(_time)` | rename _time as Timestamp | table Timestamp Account_Name message ComputerName Object_Name Object_Type Creator_Process_Name Process_Command_Line Drill Down index=os_event_prod source="WinEventLog:Security" "*\\Microsoft\\Windows NT\\CurrentVersion\\Image File Execution*" AND (GlobalFlag OR Debugger OR DEBUG_PROCESS OR DEBUG_ONLY_THIS_PROCESS OR RegCreateKeyEx OR RegSetValueEx) | `security_content_ctime(_time)` | rename _time as Timestamp | table Timestamp Account_Name message ComputerName Object_Name Object_Type Creator_Process_Name Process_Command_Line	Drill Down config case_sensitive = false | datamodel dataset = microsoft_windows_raw | filter microsoft_windows_raw.message contains "*\Microsoft\Windows NT\CurrentVersion\Image File Execution*" | filter microsoft_windows_raw.message contains "GlobalFlag" or microsoft_windows_raw.message contains "Debugger" or microsoft_windows_raw.message contains "DEBUG_PROCESS" or microsoft_windows_raw.message contains "DEBUG_ONLY_THIS_PROCESS" or microsoft_windows_raw.message contains "RegCreateKeyEx" or microsoft_windows_raw.message contains "RegSetValueEx" | fields _time as Timestamp, xdm.observer.type, xdm.event.id, xdm.event.description,xdm.event.operation_sub_type , xdm.event.original_event_type ,xdm.event.outcome ,xdm.source.process.executable.path ,xdm.source.process.name , xdm.source.user.sam_account_name ,xdm.source.user.scope , xdm.source.user.username , xdm.source.host. hostname, xdm.target.user.username, xdm.target.process.name, xdm.target.process.command_line, xdm.target. process.executable.path, microsoft_windows_raw.message ,*

index=aws_main_prod sourcetype=aws:cloudtrail eventName IN (DeleteTrail, DeleteLogGroup, DeleteLogStream) NOT "detail.service.additionalInfo.sample"=true | fields _time, userName, eventName, userAgent, awsRegion | table _time, userName, eventName, userAgent, awsRegion | dedup userName Drill Down Query index=aws_main_prod sourcetype=aws:cloudtrail eventName IN (DeleteTrail, DeleteLogGroup, DeleteLogStream) | fields _time, userName, eventName, userAgent, awsRegion | table _time, userName, eventName, userAgent, awsRegion | dedup userName	// Title: Threat - AMP - 14009 - IMP - Detect Clearing AWS Cloudtrail Logs [NTT] - Rule // Description: P3 - Someone is deleting AWS logs, verify with user if that's a legitimate action // Author: Anjali Verma, anjverma@paloaltonetworks.com // Reviewer: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: amazon_aws_raw // Date: 09/Jul/2024 config case_sensitive = false | dataset = amazon_aws_raw | filter _collector_name contains "Cloudtrail" // filtering for cloudtrail logs | filter eventName in ("DeleteTrail", "DeleteLogGroup", "DeleteLogStream") //"detail.service.additionalInfo.sample"!=true // field not found | alter username = json_extract_scalar(userIdentity , "$.sessionContext.sessionIssuer.userName"), user_arn = json_extract_scalar(userIdentity , "$.sessionContext.sessionIssuer.arn"), aws_account_id = json_extract_scalar(userIdentity , "$.accountId") \| dedup username \| fields _time, username, eventName, eventID, userAgent, awsRegion, eventSource, requestParameters, user_arn, aws_account_id, _collector_name Drill Down Query config case_sensitive = false \| dataset = amazon_aws_raw \| filter eventName in ("DeleteTrail", "DeleteLogGroup", "DeleteLogStream") \| alter username = json_extract_scalar(userIdentity , "$.sessionContext.sessionIssuer.userName") | dedup username | fields _time, username, eventName,userAgent, awsRegion

```
| tstats summariesonly=true values(host) as Host values(Malware_Attacks.file_hash) as file_hash values(Malware_Attacks.file_path) as file_path values(Malware_Attacks.file_name) as file_name values(Malware_Attacks.act) as device_action values(Malware_Attacks.category) as category values(Malware_Attacks.vendor_product) as vendor values(Malware_Attacks.action) as action from datamodel=Malware.Malware_Attacks where NOT Malware_Attacks.vendor_product IN (versa_fileFilterLog, "unknown*") by  Malware_Attacks.signature Malware_Attacks.dest Malware_Attacks.user
|fillnull value="N/A" vendor
|search vendor!="N/A" AND action=allowed
|search device_action!=Acquisition*
|rename Malware_Attacks.* as *
| eval urgency="high"
| lookup trendmicro_malware_exclusions.csv File_Path as file_path output Hash File_Path
| search NOT File_Path=*
| eval hash_compare=if(match(Hash, file_hash), "matched", "not matched")
| search hash_compare="not matched"

Drill Down Query
[{"name":"Contributing endpoint logs based on file hash($file_hash$)","search":"index=* tag=malware tag=attack file_hash IN ($file_hash$) |stats values(file_hash) asH file_hash values(file_path) as file_path values(file_name) as file_name values(act) as device_action values(action) as action values(category) as category values(vendor) as vendor by dhost","earliest_offset":"$info_min_time$","latest_offset":"$info_max_time$"}]
```

```
// Title: Threat - AMP - Endpoint tool unable to block malware detection - Rule
// Description: Detect an end-system with malware detection that was not properly blocked by the Endpoint tool, as they carry a high risk of damage or disclosure of data.
// 18/03/2024 - NTT SVR23362316 | AMP RITM01834875 - Endpoint tool unable to block malware detection | Fine Tuning. Confluence page: https://teamtools.amp.com.au/confluence/pages/viewpage.action?pageId=592188536
// Author: Sahil Sharma, ssharma7@paloaltonetworks.com
// Datasets: fireeye_hx_raw
// Date: 11/Jul/2024

config case_sensitive = false
| datamodel dataset = fireeye_hx_raw

| replacenull xdm.alert.original_threat_name = "unknown"

| filter xdm.event.description = "allowed"
| filter xdm.observer.action != "Acquisition*"

| join type = left (
    dataset = trendmicro_malware_exclusions_csv
    | fields File_Path, Hash
) as malware_exclusion malware_exclusion.File_Path = xdm.target.file.path
| filter File_Path != "*"
| filter Hash != xdm.source.process.executable.md5


| comp count() as total_events, min(_time) as firstEventTime, max(_time) as lastEventTime, values(xdm.source.host.hostname) as host, values(xdm.source.process.executable.md5) as file_hash, values(xdm.observer.action) as device_action, values(xdm.target.file.filename) as file_name, values(xdm.target.file.path) as file_path, values(xdm.alert.subcategory) as category, values(xdm.observer.vendor) as vendor, values(xdm.observer.product) as product, values(xdm.source.ipv4) as src, values(xdm.event.description) as action by xdm.alert.original_threat_name, xdm.source.user.username

| alter file_hash = arraystring(file_hash, ",")

Drill Down Query:
config case_sensitive = false
| datamodel dataset = fireeye_hx_raw
| filter $file_hash contains xdm.source.process.executable.md5
| alter name = format_string("Contributing endpoint logs based on file hash %s", $file_hash)
| comp values(xdm.source.process.executable.md5) as file_hash, values(xdm.target.file.path) as file_path, values(xdm.target.file.filename) as file_name,values(xdm.observer.action) as device_action,values(xdm.event.description) as action, values(xdm.alert.subcategory) as category, values(xdm.observer.vendor) as vendor, values(name) as name by xdm.target.host.hostname
```

```
index=secops_waf sourcetype=imperva:aa:cef |stats values(cef_severity) as severity values(src) as Attacker_IP values(request) as URL_Extension values(cs8) as Site_ID values (requestClientApplication) as Browser_Used values(cs1) as "Logs_Count" values(cs2) as Percentage_Blocked values(cs7) as Signature_Category values(cs3) as src_country count by _time signature dhost
|eval WAF_action = if(Percentage_Blocked=100, "Blocked", "Not Blocked")
|search WAF_action!=Blocked
|eval Percentage_Blocked = Percentage_Blocked + "%"
|eval severity = case(severity == "CRITICAL" and WAF_action == "Not Blocked","Critical", severity == "MAJOR" and WAF_action == "Not Blocked", "High", severity == "MINOR" and WAF_action == "Not Blocked", "Medium")
|rename _time as Alert_Time |convert ctime(Alert_Time)
|fields - count
```

Drill down :

```
index IN (amp_north_prod amp_openbanking_np amp_openbanking_prod secops_waf) sourcetype IN (incapsula:cef imperva:aa:cef) src IN ($Attacker_IP$) action=allowed
|stats values(sourceServiceName) as Site_Application values(siteid) as Site_Application_ID values (sip) as Server_IP values(requestClientApplication) as Browser_Type values(ccode) as src_country values(customer) as customer values(xff) as xff values(request) as URL values(cn1) as HTTP_Response_Code values(app) as protocal values(act) as action count by src_ip | append [| search $Attacker_IP$ action=allowed requestMethod=POST| table src_ip _time, action, request, postbody]
```

```
// Title: Threat - Imperva WAF Attack Analytics Alerts - Rulea
// Description: This usecase will trigger the results based on the alerts triggered due to inbuilt rules/signatures in Imperva
// Author: Sahil Sharma, ssharma7@paloaltonetworks.com
// Datasets: imperva_waf_raw
// Date: 03/July/2024

config case_sensitive = false
| datamodel dataset = imperva_waf_raw
| filter xdm.event.type = "Analytics"

| alter percentage_blocked = to_number(imperva_waf_raw.ImpervaAAPercentBlocked),
      cef_severity = imperva_waf_raw.cefSeverity,
      dhost = imperva_waf_raw.dhost
| alter waf_action = if(percentage_blocked = 100, "blocked", "not blocked")
| alter severity = if(cef_severity = "CRITICAL" and xdm.observer.action  = "not blocked", "Critical",
                cef_severity = "MAJOR" and xdm.observer.action  = "not blocked", "High",
                cef_severity = "MINOR" and xdm.observer.action  = "not blocked", "Medium", "Low"),
      percentage_blocked = concat(to_string(percentage_blocked), "%")

| filter waf_action != "blocked"

| comp values(severity) as severity, values(xdm.source.ipv4) as Attacker_IP, values(xdm.intermediate.host.device_id) as Site_ID, values(xdm.source.user_agent) as Browser_Used, values(imperva_waf_raw.ImpervaAANumberOfEvents) as Logs_Count, values(percentage_blocked) as Percentage_Blocked, values(xdm.source.location.country) as Src_Country, min(_time) as firstEventtime, max(_time) as lastEventTime, values(imperva_waf_raw.msg) as msg, values (imperva_waf_raw.ImpervaAAAttackType) as attack_type, values(xdm.network.http.url) as URL_Extension, values (waf_action) as waf_action by dhost, xdm.alert.name, _time, xdm.event.type

| alter Attacker_IP = arraystring(Attacker_IP, ",")

| fields  _time  as Alert_time, xdm.alert.name as signature, dhost, severity, Attacker_IP, URL_Extension, Site_ID, Browser_Used, Logs_Count, Percentage_Blocked, attack_type, Src_Country, waf_action, msg, xdm.event.type as logsubtype
```

Drill Down Query

```
config case_sensitive = false
| datamodel dataset = imperva_waf_raw
| filter $Attacker_IP contains xdm.source.ipv4
| filter xdm.observer.action = "allowed"

| comp values(imperva_waf_raw.sourceservicename) as Site_Application, values(xdm.intermediate.host.device_id) as Site_ID, values(xdm.target.ipv4) as server_ip, values(xdm.source.user_agent) as Browser_Type, values(xdm.source.location.country) as src_country, values(imperva_waf_raw.customer) as customer, values(imperva_waf_raw.xff) as xff, values(xdm.network.http.url) as URL, values(imperva_waf_raw.cn1) as HTTP_Response_Code, values (imperva_waf_raw.app) as protocol, count() as total_event_count, values(imperva_waf_raw.requestMethod) as requestMethod, values(xdm.event.outcome) as action by xdm.source.ipv4, xdm.observer.action
| fields xdm.source.ipv4 as src_ip, xdm.observer.action as waf_action, *

| union (datamodel dataset = imperva_waf_raw
    | filter $Attacker_IP contains xdm.source.ipv4

    | filter xdm.observer.action = "allowed"
    | filter imperva_waf_raw.requestMethod = "POST"
    | fields xdm.source.ipv4 as src_ip, xdm.observer.action as waf_action, xdm.event.outcome as action, _time, xdm.network.http.url as request, xdm.target.ipv4 as server_ip)
```

	// Title: Threat - Imperva-DDOS Attack on a Application - Rule // Description: This usecase will help us to trigger the alert whenever we observed the DDOS based attacked coming from the Imperva logs // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: imperva_waf_raw // Date: 03/July/2024 config case_sensitive = false |datamodel dataset =imperva_waf_raw | filter xdm.event.type = "WAF" | filter xdm.event.outcome = "REQ_BAD_TIMEOUT_CONNECTION_TO_SERVER" | join (datamodel dataset =imperva_waf_raw | filter xdm.event.type = "WAF" | filter xdm.alert.name = "DDOS" | fields xdm.intermediate.host.device_id as siteid | comp count() as count by siteid) as ddos_data ddos_data.siteid = xdm.intermediate.host.device_id |comp count() as total_events, values(imperva_waf_raw.sourceservicename) as Site_Application, values(xdm.source.user.identifier) as suid, values(imperva_waf_raw.cefSeverity) as severity, values(xdm.alert.name) as signature, values(xdm.source.ipv4) as Attacker_IP, values(xdm.target.ipv4) as Server_IP, values(xdm.source.port) as Attacker_Port, values(xdm.target.port) as Server_Port, values(xdm.source.user_agent) as Browser_Type, values(xdm.source.location.country) as src_country, values(xdm.source.location.city) as src_city, values(imperva_waf_raw.customer) as customer, values(xdm.network.http.url) as URL, values(xdm.network.http.method) as requestMethod, values(imperva_waf_raw.xff) as xff, values(imperva_waf_raw.app) as protocol, min(_time) as firstTime, max(_time) as lastTime, values(imperva_waf_raw.dproc) as dproc, values(xdm.observer.action) as waf_action, values(xdm.event.outcome) as action by xdm.intermediate.host.device_id, xdm.event.type | alter Site_Application = arraystring(Site_Application, ",") | fields total_events, firstTime, lastTime, action, suid, Site_Application, Attacker_IP, severity, signature, Attacker_IP, Attacker_Port, Server_IP, Server_Port, Browser_Type, src_city, src_country, customer, URL, xff, protocol, requestMethod, dproc, xdm.intermediate.host.device_id as site_id, waf_action, xdm.event.type as logsubtype
index IN (amp_north_prod amp_openbanking_np amp_openbanking_prod secops_waf) sourcetype = incapsula:cef act = "REQ_BAD_TIMEOUT_CONNECTION_TO_SERVER" |search [|search index IN (amp_north_prod amp_openbanking_np amp_openbanking_prod secops_waf) sourcetype = incapsula:cef cef_name=DDOS |stats count by siteid |fields - count|format] |stats values(sourceServiceName) as Site_Application values(src_ip) as Attacker_IP values(sip) as Server_IP values(request) as request values(requestClientApplication) as Browser_Type values(ccode) as src_country values(Customer) as customer values(xff) as xff values(request) as URL values(app) as protocal values(act) as action count by siteid |sort-count	Drill Down Query config case_sensitive = false |datamodel dataset = imperva_waf_raw | filter xdm.event.type = "WAF" | filter xdm.event.outcome = "REQ_BAD_TIMEOUT_CONNECTION_TO_SERVER" | filter $Site_Application contains imperva_waf_raw.sourceservicename | fields imperva_waf_raw.sourceServiceName as Site_Application, xdm.source.user.identifier as suid, imperva_waf_raw.cefSeverity as severity, xdm.alert.name as signature, xdm.source.ipv4 as Attacker_IP, xdm.target.ipv4 as Server_IP, xdm.source.port as Attacker_Port, xdm.target.port as Server_Port, xdm.source.user_agent as Browser_Type, xdm.source.location.country as src_country, xdm.source.location.city as src_city, imperva_waf_raw.customer as customer, xdm.network.http.url as URL, xdm.network.http.method as requestMethod, imperva_waf_raw.xff as xff, imperva_waf_raw.app as protocol, imperva_waf_raw.dproc as dproc, xdm.observer.action as waf_action, xdm.event.outcome as action, xdm.intermediate.host.device_id as site_id, xdm.event.type as logsubtype
Drill Down: index IN (amp_north_prod amp_openbanking_np amp_openbanking_prod secops_waf) sourcetype=incapsula:cef act="REQ_BAD_TIMEOUT_CONNECTION_TO_SERVER" sourceServiceName="$Site_Application$"	

tstats allow_old_summaries=f summariesonly=t fillnull_value="missing" count min(_time) as first_event_time max(_time) as last_event_time values(processes.vendor) as vendor values (processes.vendor_product) as vendor_product values(sourcetype) as orig_sourcetype values(host) as orig_host values(processes.src) as src values(processes.process_id) as process_id values (processes.user) as user values(processes.action) as action from datamodel=endpoint.processes where ((processes.process_name in ("ipconfig.exe", "systeminfo.exe", "net.exe", "net1.exe", "arp.exe", "nslookup.exe", "route.exe", "netstat.exe", "whoami.exe") and processes.parent_process_name=* and not (processes.parent_process_name in ("cmd.exe", "powershell*", "pwsh.exe", "explorer.exe", "unknown") or (processes.parent_process_name="net.exe" and processes.process_name="net1.exe") or (processes.parent_process_name="monitoringhost.exe" and processes.process_name="whoami.exe") or (processes.parent_process_name in ("windowsazureguestagent.exe", "windowsazuretelemetryservice.exe") and processes.process_name in ("arp.exe", "route.exe", "ipconfig.exe"))))) and (`micro_search_global_filtering_list("mscap - cmdline tool not executed in cmd shell acc (ccx) - summary gen")`) by processes.parent_process_name processes.parent_process processes.process_name processes.original_file_name processes.process_id processes.process processes.dest processes.user index | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | rename processes.* as * | fillnull vendor vendor_product value="missing" | stats values(vendor) as vendor values(vendor_product) as vendor_product values (orig_sourcetype) as orig_sourcetype values(orig_host) as orig_host values(index) as orig_index values(process_id) as process_id values(parent_process) as parent_process values(process) as process values(src) as src values(user) as user min(first_event_time) as first_event_time max (last_event_time) as last_event_time values(action) as action by parent_process_name process_name dest ccx_customer_zone | eval metadata_cis20=null(), metadata_killchainstage=" exploitation", metadata_tactics="execution", metadata_techniques="t1059|t1059.007", metadata_attack_type=null(), metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft windows", metadata_index_macros=null(), metadata_cim_datamodels="endpoint.processes", metadata_event_codes=null()| `finalise_micro_search("mscap - cmdline tool not executed in cmd shell acc (ccx) - summary gen", "ccx_customer_zone,process_name,parent_process_name,dest")`| `ccx_kill_switch`	// Title: MSCAP - Cmdline Tool Not Executed In CMD Shell ACC (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: datamodel(xdr_data) // Date: 05/June/2024 config case_sensitive = false | datamodel dataset = xdr_data | filter xdm.source.host.os_family = XDM_CONST.OS_FAMILY_WINDOWS | filter xdm.event.type = "1" | filter xdm.target.process.name in ("ipconfig.exe", "systeminfo.exe", "net.exe", "net1.exe", "arp.exe", "nslookup.exe", "route.exe", "netstat.exe", "whoami.exe") | filter xdm.source.process.name not in (null, "") | alter p1 = if(xdm.source.process.name in ("cmd.exe", "powershell*", "pwsh.exe", "explorer.exe", "unknown"), true, false) | alter p2 = if(xdm.source.process.name = "net.exe" and xdm.target.process.name = "net1.exe", true, false) | alter p3 = if(xdm.source.process.name = "monitoringhost.exe" and xdm.target.process.name = "whoami.exe", true, false) | alter p4 = if(xdm.source.process.name in ("windowsazureguestagent.exe", "windowsazuretelemetryservice.exe") and xdm.target.process.name in ("arp.exe", "route.exe", "ipconfig.exe"), true, false) | alter result = if(p1 = true or p2 = true or p3 = true or p4 = true, true, false) | filter result = false | comp count() as total_events, min(_time) as first_event_time, max(_time) as last_event_time, values(_vendor) as vendor, values(_product) as product, values(xdm.event.outcome) as outcome, values(xdm.source.host.hostname) as host, values(xdm.source.user.username) as user, values(xdm.target.process.pid) as process_id by xdm.source.process.name, xdm.target.process.name, xdm.source.process.executable.filename, xdm.event.operation
inputlookup installed_software_tracking_output | search (`micro_search_global_filtering_list("mscap - common abused remote access software installed raw (ccx) - summary gen")`) | where relative_time (now(), "-1d@d") < strptime(last_event_time, "%ft%t%:z") | search installed_software_product in ("*anydesk*", "*teamviewer*", "*logmein*", "*connectwise*", "*screenconnect*", "*mremoteng*", "*gotoassist*", "*zoho assist*", "*beyondtrust remote*", "*realvnc*", "*vnc connect*", "*tightvnc*", "*ultravnc*", "*bomgar*", "*splashtop*", "*atera*", "*supremo*", "*awesun*") | rename dns as src_name ip as src | fillnull ccx_customer_zone _raw vendor vendor_product value="undefined" | stats min (last_event_time) as first_event_time max(last_event_time) as last_event_time values(_raw) as orig_raw values(vendor) as vendor values(vendor_product) as vendor_product values(src) as src values(installed_software_version) as installed_software_version values(agent_uuid) as agent_uuid by ccx_customer_zone installed_software_product src_name | fillnull dest user src orig_sourcetype orig_index orig_host value="missing" | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="command and control|lateral movement", metadata_techniques="t1219|t1021.005", metadata_attack_type=null(), metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor=null(), metadata_vendor_products=null(), metadata_index_macros=null(), metadata_cim_datamodels=null(), metadata_event_codes=null()| `finalise_micro_search("mscap - common abused remote access software installed raw (ccx) - summary gen", "ccx_customer_zone,installed_software_product, src_name")`| `ccx_kill_switch`	// Title: MSCAP - Common Abused Remote Access Software Installed RAW (CCX) - Summary Gen // Description: Look up for all the software application installed in the environment. This come from tenable. There's a report in Splunk plug in that list the installed software. Run query and save to the lookup. // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: tenable_io_assets_raw // Date: 24/June/2024 dataset = tenable_io_assets_raw | alter installed_app = installed_software -> [] | alter ipv4 = ipv4s -> [] // expanding array of ipv4s and installed_software | arrayexpand ipv4 | arrayexpand installed_app | alter installed_software_product = replace(installed_app, "\"", ""), ipv4 = replace(ipv4, "\"", "") | filter installed_app in ("*anydesk*", "*teamviewer*", "*logmein*", "*connectwise*", "*screenconnect*", "*mremoteng*", "*gotoassist*", "*zoho assist*", "*beyondtrust remote*", "*realvnc*", "*vnc connect*", "*tightvnc*", "*ultravnc*", "*bomgar*", "*splashtop*", "*atera*", "*supremo*", "*awesun*") | fields created_at, fqdns as src_name, hostnames, installed_app, ipv4 as src, network_interfaces, operating_systems, * | comp count() as total_apps_installed, min(created_at) as first_event_time, max(created_at) as last_event_time, values (operating_systems) as operating_systems, values(src) as src, values(_product) as product, values(_vendor) as vendor, values(agent_uuid) as agent_uuid, values(agent_names) as agent_names, values(hostnames) as host, values (mac_addresses) as mac_addresses by src_name, installed_software_product

search `cim_event_signatures_indexes` eventtype=wineventlog_security eventcode=4769 servicename="*$" (ticketoptions=0x40810000 or ticketoptions=0x40800000 or ticketoptions=0x40810010) ticketencryptiontype=0x17 (`micro_search_global_filtering_list("mscap - kerberos service ticket request using rc4 encryption raw (ccx) - summary gen")`) | fillnull user src vendor vendor_product dest service service_id value="missing" | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | stats values(_raw) as orig_raw min(_time) as first_event_time values(vendor) as vendor values(vendor_product) as vendor_product max(_time) as last_event_time values(src) as src values(eventcode) as eventcode values(signature_id) as signature_id values(user) as user values(sourcetype) as orig_sourcetype values(host) as orig_host values(index) as orig_index by dest service service_id ticketencryptiontype ticketoptions ccx_customer_zone | eval metadata_cis20=null(), metadata_killchainstage="exploitation", metadata_tactics="credential access", metadata_techniques="t1558", metadata_attack_type=null(), metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft windows", metadata_index_macros="cim_event_signatures_indexes", metadata_cim_datamodels=null(), metadata_event_codes="4769"| `finalise_micro_search("mscap - kerberos service ticket request using rc4 encryption raw (ccx) - summary gen", "ccx_customer_zone,dest,service,service_id, ticketencryptiontype,ticketoptions")`| `ccx_kill_switch`	// Title: MSCAP - Kerberos Service Ticket Request Using RC4 Encryption RAW (CCX) - Summary Gen // Description: Service account use RC4 to log in from services like Azure...etc. // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: datamodel(xdr_data) // Date: 05/June/2024 config case_sensitive = false | datamodel dataset = microsoft_windows_raw | filter xdm.source.host.os_family = XDM_CONST.OS_FAMILY_WINDOWS | filter xdm.event.id = "4769" | alter service_name = json_extract_scalar(microsoft_windows_raw.event_data, "$.ServiceName") \| filter service_name = "*$" | alter ticket_options = json_extract_scalar(microsoft_windows_raw.event_data, "$.TicketOptions") \| filter ticket_options in ("0x40810000", "0x40800000", "0x40810010") \| alter ticket_encryption_type = json_extract_scalar(microsoft_windows_raw.event_data, "$.TicketEncryptionType") | filter ticket_encryption_type = "0x17" | alter service_id = json_extract_scalar(microsoft_windows_raw.event_data, "$.ServiceSid") \| alter user = json_extract_scalar(microsoft_windows_raw.event_data, "$.TargetUserName") | alter dest = service_name | replacenull service_id = "missing", user = "missing", xdm.source.ipv4 = "missing", dest = "missing" | comp count() as total_events, min(_time) as first_event_time, max(_time) as last_event_time, values(_vendor) as vendor, values(_product) as product, values(xdm.source.ipv4) as src, values(user) as user, values(xdm.source.host.hostname) as host by dest, service_id, service_name, ticket_encryption_type, ticket_options, xdm.event.id //, ccx_customer_zone field not found

search `cim_event_signatures_indexes` eventtype=wineventlog_windows signature_id in (5827, 5828, 5829, 5830, 5831) name="'netlogon'" (`micro_search_global_filtering_list("mscap - detect attempted exploitation of microsoft cve-2020-1472 netlogon raw (ccx) - summary gen")`) | fillnull eventcode dest signature signature_id src user value="missing" | eval signature=case(signature_id="5827", "the netlogon service denied a vulnerable netlogon secure channel connection from a machine account.", signature_id="5828", "the netlogon service denied a vulnerable netlogon secure channel connection using a trust account.", signature_id="5829", "the netlogon service allowed a vulnerable netlogon secure channel connection. ", signature_id="5830", "the netlogon service allowed a vulnerable netlogon secure channel connection because the machine account is allowed in the 'domain controller: allow vulnerable netlogon secure channel connections' group policy.", signature_id="5831", "the netlogon service allowed a vulnerable netlogon secure channel connection because the trust account is allowed in the 'domain controller: allow vulnerable netlogon secure channel connections' group policy.") | spath input=eventdata_xml output=extracted_src path=data{1} | spath input=eventdata_xml output=extracted_src_domain path=data{2} | spath input=eventdata_xml output=extracted_account_type path=data{3} | spath input=eventdata_xml output=extracted_os path=data{4} | spath input=eventdata_xml output=extracted_os_build path=data{5} | spath input=eventdata_xml output=extracted_os_service_pack path=data{6} | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | stats values(_raw) as orig_raw earliest(_time) as first_event_time latest(_time) as last_event_time values(eventcode) as eventcode values(extracted_src_domain) as extracted_src_domain values(extracted_src) as extracted_src values(extracted_account_type) as extracted_account_type values(extracted_os) as extracted_os values(extracted_os_build) as extracted_os_build values(extracted_os_service_pack) as extracted_os_service_pack values(src) as src values(host) as orig_host values(sourcetype) as orig_sourcetype values(user) as user values(index) as orig_index count by ccx_customer_zone dest signature signature_id vendor vendor_product | table orig_raw first_event_time last_event_time vendor vendor_product dest eventcode signature signature_id orig_host orig_sourcetype orig_index count src user ccx_customer_zone extracted_src_domain extracted_src extracted_account_type extracted_os extracted_os_build extracted_os_service_pack | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="defense evasion|privilege escalation", metadata_techniques=null(), metadata_attack_type="windows", metadata_nist=null(), metadata_cve="cve-2020-1472", metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft windows", metadata_index_macros="cim_event_signatures_indexes", metadata_cim_datamodels=null(), metadata_event_codes="5827|5828|5829|5830|5831"| `finalise_micro_search("mscap - detect attempted exploitation of microsoft cve-2020-1472 netlogon raw (ccx) - summary gen", "ccx_customer_zone,dest,signature,signature_id,vendor,vendor_product")`| `ccx_kill_switch`	// Title: MSCAP - Detect Attempted Exploitation of Microsoft CVE-2020-1472 Netlogon RAW (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: microsoft_windows_raw // Date: 18/June/2024 config case_sensitive = false | datamodel dataset = microsoft_windows_raw | filter xdm.source.host.os_family = XDM_CONST.OS_FAMILY_WINDOWS | filter xdm.event.id in ("5827", "5828", "5829", "5830", "5831") // | filter name = "netlogon" // no mapping know for name field | replacenull xdm.source.ipv4 = "missing" | alter signature = if(xdm.event.id = "5827", "the netlogon service denied a vulnerable netlogon secure channel connection from a machine account.", xdm.event.id = "5828", "the netlogon service denied a vulnerable netlogon secure channel connection using a trust account.", xdm.event.id = "5829", "the netlogon service allowed a vulnerable netlogon secure channel connection.", xdm.event.id = "5830", "the netlogon service allowed a vulnerable netlogon secure channel connection because the machine account is allowed in the domain controller: allow vulnerable netlogon secure channel connections group policy.", xdm.event.id = "5831", "the netlogon service allowed a vulnerable netlogon secure channel connection because the trust account is allowed in the domain controller: allow vulnerable netlogon secure channel connections group policy.") | alter src_domain = json_extract_scalar(microsoft_windows_raw.event_data, "$.domain"), machine_sam_account_name = json_extract_scalar(microsoft_windows_raw.event_data, "$.machineSamAccountName"), account_type = json_extract_scalar(microsoft_windows_raw.event_data, "$.accountType"), os = json_extract_scalar(microsoft_windows_raw.event_data, "$.os"), os_build = json_extract_scalar(microsoft_windows_raw.event_data, "$.osBuild"), os_service_pack = json_extract_scalar(microsoft_windows_raw.event_data, "$.osServicePack") | comp count() as total_events, earliest(_time) as first_event_time, latest(_time) as last_event_time, values(xdm.observer.vendor) as vendor, values(xdm.observer.product) as product, values(os) as extracted_os, values(os_build) as extracted_os_build, values(os_service_pack) as extracted_os_service_pack, values(src_domain) as extracted_src_domain, values(xdm.source.ipv4) as src, values(xdm.source.host.hostname) as host, values(machine_sam_account_name) as extracted_src by xdm.event.id, signature, xdm.event.original_event_type // data fetch from event_data as per assumption but corect data path to be validated once we have data available for the event ids.

No query, not anymore in list of detections, but should be provided for the Request, plus Multiple failed connection 4625 to an endpoint followed by a Lockout Event. Event code 4740 and where count>5	// Title: MSCAP - Detect Excessive Account Lockouts From Endpoint ACC (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: microsoft_windows_raw // Date: 07/June/2024 config case_sensitive = false | datamodel dataset = microsoft_windows_raw | filter xdm.source.host.os_family = XDM_CONST.OS_FAMILY_WINDOWS | filter xdm.event.id = "4625" // logon using RDP and device | filter xdm.logon.type in ("INTERACTIVE", "REMOTE_INTERACTIVE") // discus if any other login type are to be added // | fields _time, xdm.event.original_event_type, xdm.event.id, xdm.logon.type, xdm.source.host.hostname, xdm.source.ipv4, xdm.target.user.upn, xdm.target.user.username, xdm.observer.vendor, xdm.observer.product, microsoft_windows_raw.event_data, * | comp count() as total_failed_attempts, values(xdm.event.original_event_type) as orig_event_type, values(xdm.source.host.hostname) as host, values(xdm.source.ipv4) as src, values(xdm.observer.vendor) as vendor, values(xdm.observer.product) as product, min(_time) as login_failed_start_time, max(_time) as login_failed_end_time by xdm.target.user.username, xdm.target.user.upn | filter total_failed_attempts > 5 // filtering for login failed attempts count greater then 5 for a user | join (datamodel dataset = microsoft_windows_raw | filter xdm.source.host.os_family = XDM_CONST.OS_FAMILY_WINDOWS | filter xdm.event.id = "4740" | fields _time as account_locked_time, xdm.target.user.username as locked_account_user, xdm.source.host.hostname as locked_account_host) as locked_users locked_users.locked_account_user = xdm.target.user.username and timestamp_diff(locked_users.account_locked_time, login_failed_end_time, "SECOND") >= 0 | fields total_failed_attempts, login_failed_start_time, login_failed_end_time, account_locked_time, src, host, xdm.target.user.username, xdm.target.user.upn, vendor, product

tstats allow_old_summaries=f summariesonly=t fillnull_value="missing" count values(host) as host values(authentication.vendor_product) as vendor_product values(authentication.vendor) as vendor values(authentication.dest) as dest values(sourcetype) as sourcetype from datamodel=authentication where (`public_cidr_all(authentication.src)` authentication.action="success") and (`micro_search_global_filtering_list("mscap - geographically improbable access acc (ccx) - summary gen")`) by authentication.user _time authentication.src authentication.app authentication.signature authentication.action index span=1s | rename authentication.* as * | where `public_cidr_match_all(src)` | eval user=lower(user) | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | stats values(host) as orig_host values(sourcetype) as orig_sourcetype values(index) as orig_index values(dest) as dest values(vendor) as vendor values(vendor_product) as vendor_product values(action) as action count by user ccx_customer_zone _time src app signature | iplocation src | where isnotnull(lat) and isnotnull(lon) | rename country as country region as region city as city | streamstats current=f window=1 global=f last(lat) as prev_lat last(lon) as prev_lon last(country) as prev_country last(_time) as prev_time last(src) as prev_src last(app) as prev_app last(signature) as prev_signature by user ccx_customer_zone | eval lat1_r='lat' * pi() / 180, lat2_r='prev_lat' * pi() / 180, delta=('prev_lon' - 'lon') * pi() / 180, r=6372.8, "distance"=round(r * acos(sin(lat1_r) * sin(lat2_r) + cos(lat1_r) * cos(lat2_r) * cos(delta)), 2), time_diff=if((_time - prev_time) == 0, 1, _time - prev_time), speed=round(distance * 3600 / time_diff, 2) | where speed >= 800 and country != prev_country | foreach src country app signature [| eval <<field>>=mvappend(<<field>>, prev_<<field>>)] | lookup asn_lookup_by_cidr ip as src output description as src_asn_description | lookup asn_lookup_by_cidr_ipv6 ip as src outputnew description as src_asn_description | fillnull src_asn_description value="missing" | stats min(_time) as first_event_time max(_time) as last_event_time values(src) as src values(dest) as dest values(src_asn_description) as src_asn_description values(country) as country values(app) as app values(signature) as signature values(orig_host) as orig_host values(orig_sourcetype) as orig_sourcetype values(orig_index) as orig_index values(vendor) as vendor values(vendor_product) as vendor_product values(action) as action by user ccx_customer_zone | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="resource development|initial access", metadata_techniques="t1584.001|t1584.002|t1584.003|t1584.004|t1584.005|t1584.006|t1078.001|t1078.002|t1078.003|t1078.004", metadata_attack_type="pre|windows|macos|linux|o365|azuread|googleworkspace|saas|iaas|network|containers", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor=null(), metadata_vendor_products="windows|macos|linux|office 365|azure active directory|google workspace", metadata_index_macros=null(), metadata_cim_datamodels="authentication", metadata_event_codes=null()| `finalise_micro_search("mscap - geographically improbable access acc (ccx) - summary gen", "ccx_customer_zone,user")`| `ccx_kill_switch`	// Title: MSCAP - Geographically Improbable Access ACC (CCX) - Summary Gen // Author: Deven Amode, damode@paloaltonetworks.com // Date: 26/June/2024 datamodel dataset in (microsoft_windows_raw,msft_o365_azure_ad_raw) |filter xdm.event.outcome = XDM_CONST.OUTCOME_SUCCESS |comp values(xdm.source.host.hostname) as host, values(_product) as vendor_product, values(_vendor) as vendor, values(xdm.target.ipv4) as dest, values(xdm.observer.product) as sourcetype by xdm.source.user.username, _time, xdm.source.ipv4//, authentication.signature span=1s | iploc xdm.source.ipv4 loc_continent AS Continent, loc_country AS Country, loc_region AS Region, loc_city AS City, loc_latlon AS lon | comp count_distinct(lon) as location_count by xdm.source.user.username, xdm.source.ipv4 | filter location_count > 1

tstats allow_old_summaries=f summariesonly=t fillnull_value="missing" values(sourcetype) as sourcetype values(host) as host earliest(_time) as first_event_time latest(_time) as last_event_time values(authentication.src) as src values(authentication.vendor_product) as vendor_product values (authentication.vendor) as vendor values(authentication.dest) as dest count from datamodel=authentication where (`micro_search_global_filtering_list("mscap - azure aad mfa high failure ratio acc (ccx) - summary gen")`) and (sourcetype="azure:aad:signin" authentication.user="*@*" not authentication.authentication_method in ("null", "previously satisfied", "password", "passwordless phone sign-in")) by authentication.authentication_method authentication.action authentication.user index | rename authentication.* as * | search authentication_method != "missing" | eval success=if(action="success", count, null()), failure=if(action="failure", count, null()) | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | stats values (action) as action min(first_event_time) as first_event_time max(last_event_time) as last_event_time sum(success) as success sum(failure) as failure values(host) as orig_host values(sourcetype) as orig_sourcetype values(index) as orig_index values(src) as src values(dest) as dest values(vendor) as vendor values(vendor_product) as vendor_product by user authentication_method ccx_customer_zone | fillnull success failure value="0" | eval failure_ratio=round(failure / (success + failure), 2) | where `ms_threshold_filter_azure_aad_mfa_high_failure_ratio` | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="initial access", metadata_techniques=null(), metadata_attack_type="azuread|iaas", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft azure", metadata_index_macros=null(), metadata_cim_datamodels="authentication", metadata_event_codes=null()| `finalise_micro_search ("mscap - azure aad mfa high failure ratio acc (ccx) - summary gen", "ccx_customer_zone,user, authentication_method")`| `ccx_kill_switch`	// Title: MSCAP - Azure AAD MFA High Failure Ratio ACC (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: msft_azure_ad_raw // Date: 10/June/2024 config case_sensitive = false | dataset = msft_azure_ad_raw | alter authentication_method = json_extract_scalar(authenticationDetails, "$.0.authenticationMethod") \| replacenull authentication_method = "missing" \| filter authentication_method != "missing" \| filter authentication_method not in ("null", "previously satisfied", "password", "passwordless phone sign-in") \| alter authentication_step_req = json_extract_scalar(authenticationDetails, "$.0.authenticationStepResultDetail") | alter error_code = status -> errorCode | alter failure_reason = status -> failureReason // | alter additional_details = status -> additionalDetails | alter city = location -> city | alter country = location -> countryOrRegion | alter state = location -> state | alter location = format_string("%s | %s | %s", city, state, country) | comp count() as total_event, count(if(error_code = "0", True)) as success_count, count(if(error_code != "0", True)) as failed_count, earliest(createdDateTime) as first_event_time, latest(createdDateTime) as last_event_time, values(if (error_code != "0", error_code)) as failure_code, values(if(error_code != "0", failure_reason)) as failure_reason, values (ipAddress) as src_ip, values(appDisplayName) as app, values(location) as location, values(_product) as product, values(_vendor) as vendor by userPrincipalName, userDisplayName, authentication_method | alter failure_ratio = divide(failed_count, total_event) | filter failure_ratio > 0.5
search `cim_event_signatures_indexes` eventcode=4719 (`micro_search_global_filtering_list("mscap - windows server defence evasion codes to monitor raw (ccx) - summary gen")`) | fillnull eventcode dest signature signature_id value="0" | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | stats values(_raw) as orig_raw earliest(_time) as first_event_time latest(_time) as last_event_time values(eventcode) as eventcode values(src) as src values(index) as orig_index values(host) as orig_host values(sourcetype) as orig_sourcetype values(user) as user count by dest signature signature_id vendor vendor_product ccx_customer_zone | table orig_raw first_event_time last_event_time vendor vendor_product dest signature eventcode signature_id orig_host orig_sourcetype orig_index count src user ccx_customer_zone | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="defense evasion", metadata_techniques="t1562.001|t1562.002|t1562.003|t1562.004|t1562.006|t1562.007|t1562.008|t1562.009|t1562.010", metadata_attack_type="containers|iaas|linux|network|office 365|macos|windows", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft windows|linux", metadata_index_macros="cim_event_signatures_indexes", metadata_cim_datamodels=null(), metadata_event_codes="4719"| `finalise_micro_search("mscap - windows server defence evasion codes to monitor raw (ccx) - summary gen", "ccx_customer_zone,dest,signature,signature_id,vendor, vendor_product")`| `ccx_kill_switch`	datamodel dataset = microsoft_windows_raw |alter Category = arrayindex(regextract(microsoft_windows_raw.message, "Category:\s*(.+?)\s+\w+:"),0), Subcategory = arrayindex(regextract(microsoft_windows_raw.message, "Subcategory:\s*(.+)"),0), Changes = arrayindex(regextract(microsoft_windows_raw.message, "Changes:\s*(.+)"),0) | filter xdm.event.id ="4719" |comp min(_time) as first_event_time, max(_time) as last_event_time, values(xdm.event.id) as eventcode, values(xdm.source.ipv4) as src , values(xdm.source.host.hostname) as orig_host , values(xdm.observer.type) as orig_sourcetype , values(xdm.source.user.username) as user, values(Category) as Category, values(Subcategory) as Subcategory, values(Changes) as Changes by xdm.target.ipv4, _vendor, _product

search `cim_event_signatures_indexes` eventtype=wineventlog_security eventcode=4776 user != "*$" status=0xc0000064 action=failure (`micro_search_global_filtering_list("mscap - invalid users failing to auth from host using ntlm raw (ccx) - summary gen")`) | bucket span=2m _time as bucket_time | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product source_workstation value="missing" | stats values(_raw) as orig_raw dc(user) as unique_accounts values(user) as user values(eventcode) as eventcode values(signature_id) as signature_id values(host) as orig_host values(sourcetype) as orig_sourcetype values(index) as orig_index values(src) as src values(dest) as dest values(vendor) as vendor values(vendor_product) as vendor_product earliest(_time) as first_event_time latest(_time) as last_event_time by bucket_time source_workstation ccx_customer_zone | eventstats avg(unique_accounts) as comp_avg stdev(unique_accounts) as comp_std by source_workstation ccx_customer_zone | eval upperbound=(comp_avg + comp_std * 3) | eval isoutlier=if(unique_accounts > 10 and unique_accounts >= upperbound, 1, 0) | search isoutlier=1 | eval upperbound=(comp_avg + comp_std * 3) | eval metadata_cis20=null(), metadata_killchainstage="exploitation", metadata_tactics="credential access", metadata_techniques="t1110.003", metadata_attack_type="windows", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft windows", metadata_index_macros="cim_event_signatures_indexes", metadata_cim_datamodels=null(), metadata_event_codes="4776"| `finalise_micro_search("mscap - invalid users failing to auth from host using ntlm raw (ccx) - summary gen", "ccx_customer_zone,source_workstation,bucket_time")`| `ccx_kill_switch`	// Title: MSCAP - Invalid Users Failing To Auth From Host Using NTLM RAW (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: // Date: 06/June/2024 config case_sensitive = false | datamodel dataset = microsoft_windows_raw | filter xdm.source.host.os_family = XDM_CONST.OS_FAMILY_WINDOWS | filter xdm.event.id = "4776" | filter xdm.target.user.username != "*$" \| filter xdm.event.outcome = "0xc0000064" // field actioin = failure not found \| alter workstation = json_extract_scalar(microsoft_windows_raw.event_data, "$.Workstation") | replacenull workstation = "missing" | fields _time, xdm.target.user.username, xdm.event.id, xdm.event.outcome, xdm.event.original_event_type, microsoft_windows_raw.event_data, workstation, xdm.observer.type, xdm.observer.vendor, xdm.source.host.hostname, * | comp count() as total_event, count_distinct(xdm.target.user.username) as unique_accounts_count, values(xdm.target.user.username) as user, values(xdm.source.host.hostname) as host, values(xdm.event.original_event_type) as source_type, values(xdm.observer.vendor) as vendor, values(xdm.observer.product) as product, min(_time) as first_event_time, max(_time) as last_event_time, values(microsoft_windows_raw.event_data) as orig_raw by workstation, xdm.event.id, xdm.event.outcome | filter unique_accounts_count > 10 // filtering for unique accounts more than 10
search `cim_event_signatures_indexes` (eventcode in (4769, 4768, 4771) status in (0x9, 0xa, 0xb, 0xf, 0x10, 0x11, 0x13, 0x14, 0x1a, 0x1f, 0x21, 0x22, 0x23, 0x26, 0x27, 0x28, 0x29, 0x2c, 0x2d, 0x2e, 0x2f, 0x31, 0x32, 0x3e, 0x3f, 0x40, 0x41, 0x43, 0x44)) (`micro_search_global_filtering_list("mscap - kerberos manipulation raw (ccx) - summary gen")`) | rename host as orig_host sourcetype as orig_sourcetype index as orig_index | fillnull dest value="missing" | lookup index_to_ccx_customer_zone_lookup index_match as orig_index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | stats values(_raw) as orig_raw values(vendor) as vendor values(signature_id) as signature_id values(vendor_product) as vendor_product earliest(_time) as first_event_time latest(_time) as last_event_time values(orig_host) as orig_host values(orig_sourcetype) as orig_sourcetype values(orig_index) as orig_index values(src) as src values(user) as user by dest eventcode status ccx_customer_zone | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="credential access", metadata_techniques="t1212", metadata_attack_type="windows", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft windows", metadata_index_macros="cim_event_signatures_indexes", metadata_cim_datamodels=null(), metadata_event_codes="4768|4769|4771"| `finalise_micro_search("mscap - kerberos manipulation raw (ccx) - summary gen", "ccx_customer_zone,dest,eventcode,status")`| `ccx_kill_switch`	// Title: MSCAP - Kerberos Manipulation RAW (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: // Date: 14/June/2024 config case_sensitive = false | datamodel dataset = microsoft_windows_raw | filter xdm.source.host.os_family = XDM_CONST.OS_FAMILY_WINDOWS | filter xdm.event.id in ("4769", "4768", "4771") | alter status = json_extract_scalar(microsoft_windows_raw.event_data, "$.Status") \| filter status in ("0x9", "0xa", "0xb", "0xf", "0x10", "0x11", "0x13", "0x14", "0x1a", "0x1f", "0x21", "0x22", "0x23", "0x26", "0x27", "0x28", "0x29", "0x2c", "0x2d", "0x2e", "0x2f", "0x31", "0x32", "0x3e", "0x3f", "0x40", "0x41", "0x43", "0x44") \| alter xdm.target.user.username = json_extract_scalar(microsoft_windows_raw.event_data, "$.TargetUserName") | alter ticket_options = json_extract_scalar(microsoft_windows_raw.event_data, "$.TicketOptions") \| alter service_name = json_extract_scalar(microsoft_windows_raw.event_data, "$.ServiceName") | comp count() as total_events, earliest(_time) as first_event_time, latest(_time) as last_event_time, values(xdm.target.user.username) as user, values(ticket_options) as ticket_options, values(xdm.observer.vendor) as vendor, values(xdm.observer.product) as product, values(xdm.source.ipv4) as src, values(xdm.source.host.hostname) as host, values(service_name) as service_name by xdm.event.id, status

```
search `cim_alerts_indexes` `ccx_o365_management_activity_sourcetypes` workload="securitycompliancecenter" comments="new alert" (`micro_search_global_filtering_list("mscap - microsoft o365 securitycompliancecenter alerts raw (ccx) - summary gen")`) | rename relativeurl as resources id as id operation as detection_operation name as detection_title entitytype as signature comments as detection_status resultstatus as detection_action workload as vendor_product | search not detection_title in ("a potentially malicious url click was detected", "a user clicked through to a potentially malicious", "activity from a password-spray associated ip address", "activity from a tor ip address", "activity from an anonymous proxy", "admin triggered user compromise investigation", "dlp-high volume of content detected australia financial data", "email sending limit exceeded", "failed exact data match upload", "form blocked due to potential phishing attempt", "form flagged and confirmed as phishing", "messages containing malicious entity not removed after delivery", "potential nation-state activity", "powerbi administrative activity", "ransomware activity", "suspicious connector activity", "suspicious email deletion activity", "suspicious email forwarding activity", "suspicious email sending patterns detected", "suspicious inbox forwarding rule", "suspicious inbox manipulation", "suspicious tenant sending patterns observed", "tenant restricted from sending email", "tenant restricted from sending unprovisioned email", "unusual volume of external file sharing", "user restricted from sending email", "user restricted from sharing forms and collecting responses") | eval orig_data=data, orig_index=index | rex field=data max_match=0 "\"(ad|dm)?\"\:\"(?<detection_description>(.*?))\"\," | rex field=data max_match=0 "\"(f3u|trc|suid)?\"\:\"(?<user>(.*?))?\"\," | rex field=data max_match=0 "\"(tht|op|lon)?\"\:\"(?<signature>(.*?))?\"\," | rex mode=sed field=detection_description "s/\\\//g" | eval user=lower(user), user=if(mvcount(user)=1, split(user, ","), mvdedup(user)), signature=if(isnull(signature), detection_title, signature), detection_description=if(isnull(detection_description), detection_title, detection_description), detection_action=lower(detection_action), action=case(detection_action="succeeded", "success", detection_action="invalid", "error", detection_action="failed", "failure", detection_action="cancelled", "error", detection_action="interrupted", "error", detection_action="pending", "pending", 1=1, "missing"), severity='severity' | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product user id alertid detection_title value="missing" | stats values(_raw) as orig_raw earliest(_time) as first_event_time latest(_time) as last_event_time values(user) as user values(vendor) as vendor values(detection_action) as detection_action values(action) as action values(detection_operation) as detection_operation values(signature) as signature values(detection_description) as detection_description values(detection_status) as detection_status values(category) as category values(severity) as severity values(host) as orig_host values(sourcetype) as orig_sourcetype values(index) as orig_index values(src) as src values(dest) as dest values(orig_data) as orig_data values(vendor_product) as vendor_product by ccx_customer_zone detection_title alertid id  | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="initial access", metadata_techniques="t1078.004", metadata_attack_type="o365", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft office 365", metadata_index_macros="cim_alerts_indexes", metadata_cim_datamodels=null(), metadata_event_codes=null()| `finalise_micro_search("mscap - microsoft o365 securitycompliancecenter alerts raw (ccx) - summary gen", "ccx_customer_zone, alertid,id,detection_title", "ccx_customer_zone,detection_title,alertid")`| `ccx_kill_switch`
```

```
// Title: MSCAP - Microsoft O365 SecurityComplianceCenter Alerts RAW (CCX) - Summary Gen
// Author: Sahil Sharma, ssharma7@paloaltonetworks.com
// Datasets: msft_o365_general_raw
// Date: 14/June/2024

config case_sensitive = false
| dataset = msft_o365_general_raw

| filter Workload = "securitycompliancecenter"
| filter Comments = "new alert"

| filter Name not in ("a potentially malicious url click was detected", "a user clicked through to a potentially malicious",
"activity from a password-spray associated ip address", "activity from a tor ip address", "activity from an anonymous proxy",
"admin triggered user compromise investigation", "dlp-high volume of content detected australia financial data",
"email sending limit exceeded", "failed exact data match upload", "form blocked due to potential phishing attempt",
"form flagged and confirmed as phishing", "messages containing malicious entity not removed after delivery",
"potential nation-state activity", "powerbi administrative activity", "ransomware activity", "suspicious connector activity",
"suspicious email deletion activity", "suspicious email forwarding activity", "suspicious email sending patterns detected",
"suspicious inbox forwarding rule", "suspicious inbox manipulation", "suspicious tenant sending patterns observed",
"tenant restricted from sending email", "tenant restricted from sending unprovisioned email", "unusual volume of external file sharing",
"user restricted from sending email", "user restricted from sharing forms and collecting responses")

| alter f3u = data -> f3u,
      trc = data -> trc,
      suid = data -> suid,
      ad = data -> ad,
      dm = data -> dm,
      op = data -> op,
      lon = data -> lon,
      tht = data -> tht

| alter user = coalesce(f3u, trc, suid),
      detection_description = coalesce(ad, dm),
      signature = coalesce(tht, op, lon)

| replacenull detection_description = Name, signature = Name, Id = "missing", AlertId = "missing", user = "missing"

| alter action = if(ResultStatus = "succeeded", "success",
            ResultStatus in ("invalid", "cancelled", "interrupted"), "error",
            ResultStatus = "failed", "failure",
            ResultStatus = "pending", "pending",
            "missing")

| comp count() as total_event, earliest(CreationTime) as first_event_time, latest(CreationTime) as last_event_time,
values(user) as user, values(_vendor) as vendor, values(_product) as product, values(ResultStatus) as detection_action,
values(action) as action, values(operation) as detection_operation, values(signature) as signature, values
(detection_description) as detection_description, values(Category) as category, values(Severity) as severity, values
(Source) as source, values(RelativeUrl) as resources, values(EntityType) as entityType, values(Id) as Id by Name,
AlertId, Workload, Comments

| fields total_event, first_event_time, last_event_time, user, detection_description, AlertId, Id, Comments as
detection_status, action, Severity, Workload as vendor_product, resources, source, detection_operation, Name as
detection_title, signature, entityType, detection_action, category, vendor, product
```

search `cim_event_signatures_indexes` (eventcode in (4769, 4768, 4771) status in (0x9, 0xa, 0xb, 0xf, 0x10, 0x11, 0x13, 0x14, 0x1a, 0x1f, 0x21, 0x22, 0x23, 0x26, 0x27, 0x28, 0x29, 0x2c, 0x2d, 0x2e, 0x2f, 0x31, 0x32, 0x3e, 0x3f, 0x40, 0x41, 0x43, 0x44)) (`micro_search_global_filtering_list("mscap - kerberos manipulation raw (ccx) - summary gen")`) | rename host as orig_host sourcetype as orig_sourcetype index as orig_index | fillnull dest value="missing" | lookup index_to_ccx_customer_zone_lookup index_match as orig_index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | stats values(_raw) as orig_raw values(vendor) as vendor values(signature_id) as signature_id values(vendor_product) as vendor_product earliest(_time) as first_event_time latest(_time) as last_event_time values(orig_host) as orig_host values(orig_sourcetype) as orig_sourcetype values(orig_index) as orig_index values(src) as src values(user) as user by dest eventcode status ccx_customer_zone | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="credential access", metadata_techniques="t1212", metadata_attack_type="windows", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft windows", metadata_index_macros="cim_event_signatures_indexes", metadata_cim_datamodels=null(), metadata_event_codes="4768|4769|4771"| `finalise_micro_search("mscap - kerberos manipulation raw (ccx) - summary gen", "ccx_customer_zone,dest,eventcode,status")`| `ccx_kill_switch`	// Title: MSCAP - Kerberos Manipulation RAW (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: // Date: 14/June/2024 config case_sensitive = false | datamodel dataset = microsoft_windows_raw | filter xdm.source.host.os_family = XDM_CONST.OS_FAMILY_WINDOWS | filter xdm.event.id in ("4769", "4768", "4771") | alter status = json_extract_scalar(microsoft_windows_raw.event_data, "$.Status") \| filter status in ("0x9", "0xa", "0xb", "0xf", "0x10", "0x11", "0x13", "0x14", "0x1a", "0x1f", "0x21", "0x22", "0x23", "0x26", "0x27", "0x28", "0x29", "0x2c", "0x2d", "0x2e", "0x2f", "0x31", "0x32", "0x3e", "0x3f", "0x40", "0x41", "0x43", "0x44") \| alter xdm.target.user.username = json_extract_scalar(microsoft_windows_raw.event_data, "$.TargetUserName") | alter ticket_options = json_extract_scalar(microsoft_windows_raw.event_data, "$.TicketOptions") \| alter service_name = json_extract_scalar(microsoft_windows_raw.event_data, "$.ServiceName") | comp count() as total_events, earliest(_time) as first_event_time, latest(_time) as last_event_time, values(xdm.target.user.username) as user, values(ticket_options) as ticket_options, values(xdm.observer.vendor) as vendor, values(xdm.observer.product) as product, values(xdm.source.ipv4) as src, values(xdm.source.host.hostname) as host, values(service_name) as service_name by xdm.event.id, status

```
tstats allow_old_summaries=f prestats=t summariesonly=t fillnull_value="missing" earliest(_time) latest(_time) values(host) values(sourcetype) count from datamodel=intrusion_detection.ids_attacks where (`cap_filter_vms_scanners("ids_attacks.src")`) not ids_attacks.severity="informational" by ids_attacks.src ids_attacks.dest ids_attacks.signature ids_attacks.vendor_product ids_attacks.category ids_attacks.severity ids_attacks.user index ids_attacks.transport ids_attacks.action ids_attacks.dest_port ids_attacks.vendor | rename ids_attacks.* as * | fillnull datamodel value="intrusion detection" | tstats append=t allow_old_summaries=f prestats=t summariesonly=t fillnull_value="missing" earliest(_time) latest(_time) values(host) values(sourcetype) count from datamodel=malware.malware_attacks by malware_attacks.dest malware_attacks.signature malware_attacks.vendor_product malware_attacks.category malware_attacks.action malware_attacks.user malware_attacks.file_hash index malware_attacks.src malware_attacks.vendor | rename malware_attacks.* as * | fillnull datamodel value="malware" | tstats append=t allow_old_summaries=f prestats=t summariesonly=t fillnull_value="missing" earliest(_time) latest(_time) values(host) values(sourcetype) count from datamodel=web where   [| inputlookup high_risk_url_category_list_by_vendor_lookup where vendor_product=* category=*   | fields vendor_product category   | rename * as web.* ] by web.src web.dest web.action web.url web.user web.vendor_product index web.category web.vendor | rename web.* as * | fillnull datamodel value="web" | search (`micro_search_global_filtering_list("mscap - multi vendor detection - attacked internal ip acc (ccx) - summary gen")`) | lookup high_risk_url_category_list_by_vendor_lookup vendor_product category output category as match_category | where (isnotnull(match_category) and datamodel="web") or datamodel != "web" | fillnull src severity action dest_port file_hash url signature category vendor transport value="missing" | eval user=lower(user), ip=mvappend(src, dest) | mvexpand ip | where cidrmatch("10.0.0.0/8", ip) or cidrmatch("172.16.0.0/12", ip) or cidrmatch("192.168.0.0/16", ip) or cidrmatch("fc00::/7", ip) | stats earliest(_time) as first_event_time latest(_time) as last_event_time values(host) as host values(sourcetype) as sourcetype count by ip src dest category signature severity vendor vendor_product action dest_port file_hash url user index datamodel transport | eval signature=if(datamodel="web", category, signature), attack=vendor_product + ":" + signature, src_ip=src, src_ip_24=if(cidrmatch("0.0.0.0/0", src_ip), src_ip, "missing"), dest_ip=dest, dest_ip_24=if(cidrmatch("0.0.0.0/0", dest_ip), dest_ip, "missing") | `convert_24_rex(src_ip_24)` | `convert_24_rex(dest_ip_24)` | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | stats count(attack) as attack_count dc(attack) as attack_dc dc(vendor_product) as vendor_product_dc values(attack) as attack min(first_event_time) as first_event_time max(last_event_time) as last_event_time values(src) as src values(src_ip) as src_ip values(src_ip_24) as src_ip_24 values(dest) as dest values(dest_ip) as dest_ip values(dest_ip_24) as dest_ip_24 values(category) as category values(severity) as severity values(signature) as signature values(dest_port) as dest_port values(transport) as transport values(action) as action values(file_hash) as file_hash values(url) as url values(user) as user values(vendor_product) as vendor_product values(vendor) as vendor values(datamodel) as datamodel values(index) as orig_index values(host) as orig_host values(sourcetype) as orig_sourcetype by ccx_customer_zone ip | where attack_dc >= 3 and vendor_product_dc > 1  | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics=null(), metadata_techniques=null(), metadata_attack_type=null(), metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor=null(), metadata_vendor_products=null(), metadata_index_macros=null(), metadata_cim_datamodels="intrusion_detection.ids_attacks|malware.malware_attacks|web", metadata_event_codes=null()| `finalise_micro_search("mscap - multi vendor detection - attacked internal ip acc (ccx) - summary gen", "ccx_customer_zone,ip")`| `ccx_kill_switch`
```

```
// Title: MSCAP - Multi Vendor Detection - Attacked Internal IP ACC (CCX) - Summary Gen
// Author: Sahil Sharma, ssharma7@paloaltonetworks.com
// Datasets: panw_ngfw_threat_raw
// Date: 27/June/2024

config case_sensitive = false
| datamodel dataset = panw_ngfw_threat_raw

| join (
    dataset = high_risk_url_category_list_by_vendor_lookup
    | fields url_category, vendor_name
) as high_risk_url high_risk_url.url_category = panw_ngfw_threat_raw.url_category

| filter incidr(xdm.source.ipv4, "10.0.0.0/8") = true or
       incidr(xdm.source.ipv4, "172.16.0.0/12") = true or
       incidr(xdm.source.ipv4, "192.168.0.0/16") = true or
       incidr(xdm.source.ipv4, "fc00::/7") = true or
       incidr(xdm.target.ipv4, "10.0.0.0/8") = true or
       incidr(xdm.target.ipv4, "172.16.0.0/12") = true or
       incidr(xdm.target.ipv4, "192.168.0.0/16") = true or
       incidr(xdm.target.ipv4, "fc00::/7") = true

| replacenull panw_ngfw_threat_raw.severity = "missing", xdm.observer.action = "missing", xdm.target.file.sha256 = "missing", panw_ngfw_threat_raw.threat_category = "missing"


| comp count(panw_ngfw_threat_raw.threat_name) as attack_count, count_distinct(panw_ngfw_threat_raw.threat_name) as attack_dc, values(_vendor) as vendor, values(_product) as product, values(panw_ngfw_threat_raw.threat_name) as attack, min(_time) as first_event_time, max(_time) as last_event_time, values(panw_ngfw_threat_raw.threat_category) as category, values(panw_ngfw_threat_raw.severity) as severity, values(xdm.target.port) as dest_port, values(xdm.observer.action) as action, values(xdm.target.file.sha256) as file_hash, values(panw_ngfw_threat_raw.users) as user, values(panw_ngfw_threat_raw.cloud_hostname) as cloud_hostname by xdm.source.ipv4, xdm.target.ipv4

// | filter attack_dc >= 3  // as of now this condition is not satisfed. So do we have to filter for 3 or more attacks
```

search `cim_event_signatures_indexes` eventtype=wineventlog_security eventcode=4771 status=0x18 user != "*$" (`micro_search_global_filtering_list("mscap - multiple users failing to authenticate from host using kerberos raw (ccx) - summary gen")`) | bucket span=2m _time as bucket_time | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product ipaddress value="missing" | stats values(_raw) as orig_raw dc(user) as unique_accounts values(eventcode) as eventcode values(signature_id) as signature_id values(user) as user values(host) as orig_host values(sourcetype) as orig_sourcetype values(index) as orig_index values(src) as src values(dest) as dest values(vendor) as vendor values(vendor_product) as vendor_product earliest(_time) as first_event_time latest(_time) as last_event_time by bucket_time ipaddress ccx_customer_zone | eventstats avg(unique_accounts) as comp_avg stdev(unique_accounts) as comp_std by ipaddress ccx_customer_zone | eval upperbound=(comp_avg + comp_std * 3) | eval isoutlier=if(unique_accounts > 10 and unique_accounts >= upperbound, 1, 0) | search isoutlier=1 | eval metadata_cis20=null(), metadata_killchainstage="exploitation", metadata_tactics="credential access", metadata_techniques="t1110.003", metadata_attack_type="windows", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft windows", metadata_index_macros="cim_event_signatures_indexes", metadata_cim_datamodels=null(), metadata_event_codes="4771"| `finalise_micro_search("mscap - multiple users failing to authenticate from host using kerberos raw (ccx) - summary gen", "ccx_customer_zone,ipaddress,bucket_time")`| `ccx_kill_switch`	// Title: MSCAP - Multiple Users Failing To Authenticate From Host Using Kerberos RAW (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: microsoft_windows_raw // Date: 06/June/2024 config case_sensitive = false | datamodel dataset = microsoft_windows_raw | filter xdm.source.host.os_family = XDM_CONST.OS_FAMILY_WINDOWS | filter xdm.event.id = "4771" | alter status = json_extract_scalar(microsoft_windows_raw.event_data, "$.Status") \| filter xdm.target.user.username != "*$" | filter status = "0x18" | alter ticket_options = json_extract_scalar(microsoft_windows_raw.event_data, "$.TicketOptions") \| alter service_name = json_extract_scalar(microsoft_windows_raw.event_data, "$.ServiceName") | fields _time, xdm.target.user.username, xdm.event.id, status, xdm.event.original_event_type, ticket_options, service_name, microsoft_windows_raw.event_data, xdm.observer.type, xdm.observer.vendor, xdm.source.host.hostname, xdm.source.ipv4, * | comp count() as total_event, count_distinct(xdm.target.user.username) as unique_accounts_count, values(xdm.target.user.username) as user, values(xdm.source.host.hostname) as host, values(xdm.event.original_event_type) as source_type, values(xdm.observer.vendor) as vendor, values(xdm.observer.product) as product, min(_time) as first_event_time, max(_time) as last_event_time by xdm.source.ipv4, xdm.event.id, status | filter unique_accounts_count > 10 // filtering for unique accounts more than 10
search `cim_endpoint_indexes` sourcetype=xmlwineventlog ([| inputlookup windows_security_group_change_violation_security_group_lookup | table specific_event_code_include_list_1 | rename specific_event_code_include_list_1 as eventcode | search eventcode=* and eventcode != "" | format]) and ([| inputlookup windows_security_group_change_violation_security_group_lookup | table specific_group_name_include_list_2 | rename specific_group_name_include_list_2 as group_name | search group_name=* and group_name != "" | format]) group_domain != "builtin" (`micro_search_global_filtering_list("mscap - user added or removed from privileged group raw (ccx) - summary gen")`) | eval src_user=caller_user_name, signature=subject, group_name=group_name, orig_index=index, orig_sourcetype=sourcetype, orig_host=host | eval membername=replace (membername, "\\\\,", "<comma>") | rex field=membername "cn=(?<user2>[^,]+)" | eval user2=replace (user2, "<comma>", ",") | eval user=coalesce(user2, user, src_user) | search eventcode != "4735" and src_user != "*$" | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product dest value="missing" | fillnull src_user src signature group_name user dest value="missing" | stats values(action) as action values(_raw) as orig_raw values(orig_host) as orig_host values(orig_index) as orig_index values(src) as src values(orig_sourcetype) as orig_sourcetype values(vendor) as vendor values(vendor_product) as vendor_product earliest(_time) as first_event_time latest(_time) as last_event_time values(signature_id) as signature_id by eventcode src_user signature group_name user dest ccx_customer_zone | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="persistence", metadata_techniques="t1098.001|t1098.003", metadata_attack_type="active directory|windows", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft windows", metadata_index_macros="cim_endpoint_indexes", metadata_cim_datamodels=null(), metadata_event_codes="4735"| `finalise_micro_search("mscap - user added or removed from privileged group raw (ccx) - summary gen", "ccx_customer_zone, eventcode,src_user,signature,group_name,user,dest")`| `ccx_kill_switch`	// Title: MSCAP - User Added or Removed From Privileged Group RAW (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: microsoft_windows_raw // Date: 17/June/2024 /* Alert Suppression : 6h Suppression Fields : xdm.event.id, xdm.event.original_event_type, user, group_name */ config case_sensitive = false | datamodel dataset = microsoft_windows_raw | filter xdm.source.host.os_family = XDM_CONST.OS_FAMILY_WINDOWS | filter xdm.event.id in ("4727", "4728", "4729", "4730", "4731", "4732", "4733", "4734", "4735", "4737", "4754", "4755", "4756", "4757", "4758", "4764") | filter xdm.target.user.username not in ("", null) | filter xdm.target.user.domain != "Builtin" | filter xdm.event.id != "4735" and xdm.source.user.username != "*$" \| alter membername = replace(json_extract_scalar(microsoft_windows_raw.event_data, "$.MemberName"), "\,", " ") | alter user = arrayindex(regextract(membername, "CN=([^,]+)"), 0) | replacenull xdm.source.user.username = "missing", user = "missing" | comp count() as total_event, min(_time) as first_event_time, max(_time) as last_event_time, values(xdm.source.host.hostname) as host, values(xdm.observer.vendor) as vendor, values(xdm.observer.product) as product, values(xdm.target.user.username) as group_name, values(membername) as member_name by xdm.event.id, xdm.event.original_event_type, user, xdm.source.user.username

```
search `cim_vulnerabilities_indexes` eventtype=vulnerabilities sourcetype="tenable:*" severity != informational ("plugin.family"=backdoors or "plugin_family"=backdoors and signature=*) or ("plugin.family"=* or "plugin_family"=* and signature=implant) or ("plugin.family"="web servers" or "plugin_family"="web servers" and signature="backdoor detection") or ("plugin.family"=* or "plugin_family"=* and signature="*malicious process detection*") and (acceptrisk != "true" or severity_modification_type != "accepted") tag=vulnerability tag=report (`cap_filter_vms_scanners("ipv6")`) (`cap_filter_vms_scanners("ipv4")`) (`micro_search_global_filtering_list("mscap - tenable compromised host detection raw (ccx) - summary gen")`) | rename output as orig_output | eval detection_name=if(isnull(plugin_name), pluginname, plugin_name), description=if(isnull(description), 'plugin.description', description), detection_family=if(isnull('family.name'), plugin_family, 'family.name'), detection_solution=if(isnull(solution), 'plugin.solution', solution), detection_id=if(isnull(pluginid), plugin_id, pluginid), detection_app=if(isnull(service), 'port.service', service), last_detected=if(isnull(lastseen), last_found, strftime(lastseen, "%ft%t%:z")), detection_output=if(isnull(orig_output), "unknown", orig_output), detection_severity=lower(severity), src_ip=if(isnotnull(ipv4), ipv4, ipv6), src_ip=if(cidrmatch("0.0.0.0/0", src_ip) or cidrmatch("::/0", src_ip), src_ip, "0") | fillnull description detection_family detection_solution detection_id detection_output detection_name detection_app value="unknown" | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | rex mode=sed field=detection_output "s/\n//g s/^\s+//g" | stats values(_raw) as orig_raw values(index) as orig_index values(src) as src values(sourcetype) as orig_sourcetype values(host) as orig_host values(user) as user values(dest) as dest values(detection_family) as detection_family values(detection_name) as detection_name values(description) as description values(detection_solution) as detection_solution values(detection_app) as detection_app values(port) as port values(detection_severity) as detection_severity values(detection_output) as detection_output values(vendor) as vendor values(vendor_product) as vendor_product count earliest(_time) as first_event_time latest(_time) as last_event_time by src_ip signature ccx_customer_zone  | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics=null(), metadata_techniques=null(), metadata_attack_type=null(), metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="tenable", metadata_vendor_products="tenable", metadata_index_macros="cim_vulnerabilities_indexes", metadata_cim_datamodels=null(), metadata_event_codes=null()| `finalise_micro_search("mscap - tenable compromised host detection raw (ccx) - summary gen", "ccx_customer_zone,src_ip,signature")`| `ccx_kill_switch`
```

```
// Title: MSCAP - Tenable Compromised Host Detection RAW (CCX) - Summary Gen
// Author: Deven Amode, damode@paloaltonetworks.com
// Date: 14/June/2024

datamodel dataset = tenable_io_vulnerabilities_raw
| alter
    signature = json_extract_scalar(tenable_io_vulnerabilities_raw.plugin,"$.synopsis"),
    port = json_extract_scalar(tenable_io_vulnerabilities_raw.port,"$.port"),
    last_detected = format_timestamp("%ft%t%:z", tenable_io_vulnerabilities_raw.last_found),
    xdm.target.host.ipv4_addresses = arraystring(xdm.target.host.ipv4_addresses, "|"),
    xdm.target.host.ipv6_addresses = arraystring(xdm.target.host.ipv6_addresses, "|")

| fields *, xdm.alert.category as detection_family, xdm.alert.severity as detection_severity, xdm.alert.description as description,xdm.alert.name as detection_name,xdm.alert.original_alert_id as detection_id,xdm.event.description as detection_output, tenable_io_vulnerabilities_raw.severity_modification_type as severity_modification_type
| filter
    detection_severity != "informational" and xdm.target.host.ipv4_addresses not in ("bvmswhp01", "10.10.144.10", "btenwh01","bvmsppp01", "10.40.144.12", "btenpp01","bvmsndp01", "10.50.144.10", "btenb201","bnesswh01", "10.10.143.21", "btenwh02","btenmi01", "10.25.144.10") and (severity_modification_type != "accepted") and xdm.target.host.ipv6_addresses not in ("bvmswhp01", "10.10.144.10", "btenwh01","bvmsppp01", "10.40.144.12", "btenpp01","bvmsndp01", "10.50.144.10", "btenb201","bnesswh01", "10.10.143.21", "btenwh02","btenmi01", "10.25.144.10") and
    (detection_family="backdoors" and signature="*") or
    (detection_family="*" and signature="implant") or
    (detection_family="web servers" and signature="backdoor detection") or
    (detection_family="*" and signature="*malicious process detection*")


| comp
    values(xdm.observer.type) as orig_sourcetype,
    values(xdm.observer.name ) as orig_host, //_collector_name
    values(xdm.target.host.ipv4_addresses) as dest_ipv4,
    values(xdm.target.host.ipv6_addresses) as dest_ipv6,
    values(detection_family) as detection_family,
    values(detection_name) as detection_name,
    values(description) as description,
    values(port) as port,
    values(detection_severity) as detection_severity,
    values(detection_output) as detection_output,
    values(_vendor) as vendor,
    values(_product) as vendor_product,
    count(), earliest(_time) as first_event_time, latest(_time) as last_event_time
    by signature

/*acceptrisk, plugin.solution, solution fields not found
port.service field not found and doesn't exist in Splunk Tenable addon either
 src_ip=if(isnotnull(ipv4), ipv4, ipv6) - ipv4 and ipv6 are mapped to src_ip in SPL, however, in Splunk addon, it is mapped to dest_ip
*/
```

search `cim_event_signatures_indexes` (eventcode=4741 samaccountname="*$") or (eventcode=4781 oldtargetusername="*$" newtargetusername != "*$") (`micro_search_global_filtering_list("mscap - new or renamed user account with '$' in attribute 'samaccountname' raw (ccx) - summary gen")`) | eval samaccountname=if(mvcount (samaccountname) > 1, mvindex(samaccountname, 1), samaccountname) | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull src dest src_user vendor vendor_product samaccountname oldtargetusername newtargetusername value="missing" | rename host as orig_host sourcetype as orig_sourcetype index as orig_index | stats values(_raw) as orig_raw earliest(_time) as first_event_time latest(_time) as last_event_time values(signature) as signature values(user) as user values(samaccountname) as samaccountname values(oldtargetusername) as oldtargetusername values(vendor) as vendor values(vendor_product) as vendor_product values(orig_host) as orig_host values(orig_sourcetype) as orig_sourcetype values(orig_index) as orig_index values(src) as src by dest src_user eventcode newtargetusername ccx_customer_zone | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="defense evasion", metadata_techniques="t1036.005", metadata_attack_type=null(), metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft windows", metadata_index_macros="cim_event_signatures_indexes", metadata_cim_datamodels=null(), metadata_event_codes="4741|4781"| `finalise_micro_search("mscap - new or renamed user account with '$' in attribute 'samaccountname' raw (ccx) - summary gen", "ccx_customer_zone,dest,src_user,eventcode, newtargetusername")`| `ccx_kill_switch`	// Title: MSCAP - New or Renamed User Account with '$' in Attribute 'SamAccountName' RAW (CCX) EventCode=4741 - Summary Gen /// Author: Deven Amode, damode@paloaltonetworks.com // Datasets: microsoft_windows_raw // Date: 06/June/2024 datamodel dataset = microsoft_windows_raw \|alter samaccountname=json_extract_scalar(microsoft_windows_raw.event_data,"$.SamAccountName") //| alter samaccountname=if(array_length(samaccountname) > 1, arrayindex(samaccountname, 1), samaccountname) |filter xdm.event.id = "4741" and samaccountname = "*$" | replacenull xdm.source.user.username = "missing", samaccountname = "missing" |comp values(microsoft_windows_raw._raw_log), earliest(_time) as first_time, latest(_time) as last_time, values(xdm.source.user.username) as user, values(samaccountname) as samaccountname, values(_vendor) as vendor, values(_product) as product, values(xdm.observer.name) as orig_host by xdm.target.ipv4,xdm.event.id
search `cim_event_signatures_indexes` eventtype=wineventlog_security eventcode=4741 serviceprincipalnames in ("*host/*", "*restrictedkrbhost/*") and newuacvalue=0x80 (`micro_search_global_filtering_list("mscap - windows computer account with spn raw (ccx) - summary gen")`) | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product dest logon_id subjectusersid value="missing" | stats values(vendor) as vendor values(vendor_product) as vendor_product values(sourcetype) as orig_sourcetype values(host) as orig_host values(index) as orig_index values(_raw) as orig_raw min(_time) as first_event_time max(_time) as last_event_time values(eventcode) as eventcode values(signature_id) as signature_id values(targetdomainname) as targetdomainname values(targetsid) as targetsid values(targetusername) as user values(src) as src values(subjectusername) as subjectusername values(dnshostname) as dnshostname values(serviceprincipalnames) as serviceprincipalnames by ccx_customer_zone dest logon_id subjectusersid | eval metadata_cis20="cis 3|cis 5|cis 16", metadata_killchainstage="installation", metadata_tactics="credential access", metadata_techniques="t1558", metadata_attack_type=null(), metadata_nist="de.cm", metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft windows", metadata_index_macros="cim_event_signatures_indexes", metadata_cim_datamodels=null(), metadata_event_codes="4741"| `finalise_micro_search("mscap - windows computer account with spn raw (ccx) - summary gen", "ccx_customer_zone,dest,logon_id,subjectusersid")`| `ccx_kill_switch`	// Title: MSCAP - Windows Computer Account With SPN RAW (CCX) - Summary Gen /// Author: Deven Amode, damode@paloaltonetworks.com // Datasets: microsoft_windows_raw // Date: 07/June/2024 datamodel dataset = microsoft_windows_raw |alter serviceprincipalnames=json_extract_scalar(microsoft_windows_raw.event_data,"$.ServicePrincipalNames"), newuacvalue=json_extract_scalar(microsoft_windows_raw.event_data,"$.NewUacValue"), targetsid=json_extract_scalar(microsoft_windows_raw.event_data,"$.targetsid"), dnshostname=json_extract_scalar(microsoft_windows_raw.event_data,"$.DnsHostName") | filter xdm.event.id ="4741" and serviceprincipalnames in ("*host/*", "*restrictedkrbhost/*") and newuacvalue="0x80" |comp values(_vendor) as vendor , values(_product) as vendor_product , values(xdm.observer.type) as orig_sourcetype , values(xdm.source.host.hostname) as orig_host , values(microsoft_windows_raw._raw_log) as orig_raw, min(_time) as first_event_time, max(_time) as last_event_time , values(xdm.event.id) as eventcode , values(xdm.target.user.domain) as targetdomainname , values(targetsid) as targetsid , values(xdm.source.user.username) as user , values(xdm.source.ipv4) as src , values(dnshostname) as dnshostname , values(serviceprincipalnames) as serviceprincipalnames by xdm.target.ipv4, xdm.source.user.identifier

search `cim_change_indexes` sourcetype=azure:aad:audit activitydisplayname="*consent to application*" result=success (`micro_search_global_filtering_list("mscap - initial access consent grant attack via azure application raw (ccx) - summary gen")`) | rename initiatedby.user.userprincipalname as user targetresources{}.displayname as app | eval user=lower(user), signature=activitydisplayname | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | rename host as orig_host sourcetype as orig_sourcetype index as orig_index | rename _raw as orig_raw | table status orig_raw app user signature action orig_host orig_sourcetype orig_index src dest vendor vendor_product ccx_customer_zone _time | rename _time as event_time | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="initial access|credential access", metadata_techniques="t1566.002|t1528", metadata_attack_type="azuread|iaas", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft azure", metadata_index_macros="cim_change_indexes", metadata_cim_datamodels=null(), metadata_event_codes=null()| `finalise_micro_search("mscap - initial access consent grant attack via azure application raw (ccx) - summary gen", "*")`| `ccx_kill_switch`	// Title: MSCAP - Initial Access Consent Grant Attack via Azure Application RAW (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: msft_azure_ad_audit_raw // Date: 10/June/2024 config case_sensitive = false | dataset = msft_azure_ad_audit_raw | filter activityDisplayName = "*consent to application*" | filter result = "success" | alter user = json_extract_scalar(initiatedBy, "$.user.userPrincipalName") \| alter src = json_extract_scalar(initiatedBy, "$.user.ipAddress") | alter app = json_extract_scalar(targetResources, "$.0.displayName") | fields activityDateTime, user, src, app, activityDisplayName, operationType, category, _vendor, _product, result, * | comp count() as total_events, values(src) as src_ip, earliest(activityDateTime) as first_event_time, latest (activityDateTime) as last_event_time, values(app) as app, values(_vendor) as vendor, values(_product) as produt by activityDisplayName, user, result
search `cim_change_indexes` `ccx_o365_management_activity_sourcetypes` workload=azureactivedirectory operation="add app role assignment grant to user." and (`micro_search_global_filtering_list("mscap - o365 add app role assignment grant user raw (ccx) - summary gen")`) | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product user actoripaddress dest resultstatus value="missing" | stats values(status) as status values(action) as action count min(_time) as first_event_time max(_time) as last_event_time values(vendor) as vendor values(vendor_product) as vendor_product values(index) as orig_index values(_raw) as orig_raw values(sourcetype) as orig_sourcetype values(host) as orig_host values(src) as src values (actor{}.id) as actor.id values(actor{}.type) as actor.type by ccx_customer_zone user actoripaddress dest resultstatus | eval metadata_cis20=null(), metadata_killchainstage="exploitation", metadata_tactics="persistence", metadata_techniques="t1136.003|t1136", metadata_attack_type=null(), metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft office 365", metadata_index_macros="cim_change_indexes", metadata_cim_datamodels=null(), metadata_event_codes=null()| `finalise_micro_search("mscap - o365 add app role assignment grant user raw (ccx) - summary gen", "ccx_customer_zone,user, actoripaddress,dest,resultstatus")`| `ccx_kill_switch`	// Title: MSCAP - O365 Add App Role Assignment Grant User RAW (CCX) - Summary Gen // Author: Deven Amode, damode@paloaltonetworks.com // Date: 17/June/2024 config case_sensitive = false | dataset = msft_o365_azure_ad_raw | alter action = if(ResultStatus = "succeeded", "success", if(ResultStatus in ("invalid", "cancelled", "interrupted"), "error", if (ResultStatus = "failed", "failure", if(ResultStatus = "pending", "pending", "missing")))), actor_extended = Actor -> [] | arrayexpand actor_extended |alter actor_id = actor_extended -> ID, actor_type = actor_extended -> Type | filter Workload = "*activedirectory*" and operation = "add app role assignment grant to user*" | comp values(ResultStatus) as status , values(action) as action, count(), earliest(_time) as first_event_time, latest (_time) as last_event_time , values(_vendor) as vendor , values(_product) as vendor_product, values(_collector_type) as orig_sourcetype , values(_reporting_device_name) as orig_host , values(ClientIP) as src , values(actor_id) as actor_id, values(actor_type) as actor_type by UserId, ActorIpAddress
search `cim_authentication_indexes` sourcetype=azure:aad:signin status.errorcode=0 appdisplayname="*powershell*" tokenissuertype="azuread" (`micro_search_global_filtering_list ("mscap - azure active directory powershell sign-in raw (ccx) - summary gen")`) | rename userprincipalname as user appdisplayname as app | eval user=lower(user), action=if(status . errorcode == "0", "allowed", status . errorcode), signature=activitydisplayname | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | rename host as orig_host sourcetype as orig_sourcetype index as orig_index | rename _raw as orig_raw _time as event_time | table orig_raw event_time app user signature action orig_host orig_sourcetype orig_index src dest vendor vendor_product ccx_customer_zone | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="initial access|defense evasion|persistence|privilege escalation", metadata_techniques="t1078.004", metadata_attack_type="azuread|iaas|windows", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft azure|azure active directory", metadata_index_macros="cim_authentication_indexes", metadata_cim_datamodels=null(), metadata_event_codes=null()| `finalise_micro_search("mscap - azure active directory powershell sign-in raw (ccx) - summary gen", "*")`| `ccx_kill_switch`	// Title: MSCAP - Azure Active Directory PowerShell Sign-in RAW (CCX) - Summary Gen /// Author: Deven Amode, damode@paloaltonetworks.com // Datasets: msft_azure_ad_raw // Date: 11/June/2024 config case_sensitive = false |dataset = msft_azure_ad_raw |fields _time, appdisplayname as app, userprincipalname as user , tokenIssuerType, status,_collector_type, _reporting_device_name, _vendor,_product |alter status_error_code = json_extract_scalar(status, "$.errorCode") |filter app ="*powershell*" and tokenIssuerType ="azuread" and status_error_code = "0" |alter user = lowercase(user),action = if(status_error_code = "0", "allowed", status_error_code)

tstats allow_old_summaries=t summariesonly=t fillnull_value="missing" earliest(_time) as first_event_time latest(_time) as last_event_time values(web.vendor) as vendor values(web.vendor_product) as vendor_product values(sourcetype) as orig_sourcetype values(web.user) as user values(host) as orig_host values(web.action) as action from datamodel=web where (`micro_search_global_filtering_list("mscap - web jsp request via url acc (ccx) - summary gen")`) and (web.http_method in ("get") web.url in ("*.jsp?cmd=*", "*j&cmd=*")) by web.http_user_agent web.http_method web.url web.url_length web.src web.dest index | rename web.* as * | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | stats min(first_event_time) as first_event_time max(last_event_time) as last_event_time values(orig_sourcetype) as orig_sourcetype values(user) as user values(vendor) as vendor values(vendor_product) as vendor_product values(orig_host) as orig_host values(index) as orig_index values(action) as action by http_user_agent http_method url url_length src dest ccx_customer_zone | eval metadata_cis20="cis 3|cis 5|cis 16", metadata_killchainstage="exploitation", metadata_tactics="persistence", metadata_techniques="t1505.003|t1505|t1190", metadata_attack_type=null(), metadata_nist="de.cm", metadata_cve="cve-2022-22965", metadata_detectframework="springsteen", metadata_vendor=null(), metadata_vendor_products=null(), metadata_index_macros=null(), metadata_cim_datamodels="web", metadata_event_codes=null()| `finalise_micro_search("mscap - web jsp request via url acc (ccx) - summary gen", "ccx_customer_zone,http_user_agent,http_method,url,url_length,src,dest")`| `ccx_kill_switch`	// Title: MSCAP - Web JSP Request via URL ACC (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: datamodel // Date: 12/June/2024 config case_sensitive = false | datamodel dataset in (mimecast_mimecast_raw, panw_ngfw_url_raw, msft_o365_azure_ad_raw) | filter xdm.event.type = "threat" | filter xdm.network.http.method = XDM_CONST.HTTP_METHOD_GET | filter xdm.network.http.url in ("*.jsp?cmd=*", "*j&cmd=*") // filtering for inbound requests | filter incidr(xdm.source.ipv4, "10.0.0.0/8") = false and incidr(xdm.source.ipv4, "172.16.0.0/12") = false and incidr(xdm.source.ipv4, "192.168.0.0/16") = false // filter source ip not local ip | comp count() as total_http_requests, earliest(_time) as first_seen, latest(_time) as last_seen, values(xdm.source.user.username) as user, values(xdm.observer.vendor) as vendor, values(xdm.observer.product) as product by xdm.source.user_agent, xdm.network.http.method, xdm.network.http.url, xdm.source.ipv4, xdm.target.ipv4
search `cim_event_signatures_indexes` eventcode=4624 logon_type=10 action=success dest_priority=critical ((not user_category="domain admins") or user_category="dont_expire_password") (`micro_search_global_filtering_list("mscap - service account or non domain admin rdp login to domain controller raw (ccx) - summary gen")`) | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product src dest user action value="missing" | stats values(logon_type) as logon_type values(process) as process values(vendor) as vendor values(vendor_product) as vendor_product values(_raw) as orig_raw values(sourcetype) as orig_sourcetype values(host) as orig_host values(index) as orig_index min(_time) as first_event_time max(_time) as last_event_time values(action) as action by ccx_customer_zone src dest user | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="lateral movement", metadata_techniques="t1021.001", metadata_attack_type=null(), metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft windows", metadata_index_macros="cim_event_signatures_indexes", metadata_cim_datamodels=null(), metadata_event_codes="4624"| `finalise_micro_search("mscap - service account or non domain admin rdp login to domain controller raw (ccx) - summary gen", "ccx_customer_zone,src,dest,user")`| `ccx_kill_switch`	// Title: MSCAP - Service Account or Non Domain Admin RDP Login to Domain Controller RAW (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: microsoft_windows_raw // Date: 18/June/2024 config case_sensitive = false | datamodel dataset = microsoft_windows_raw | filter xdm.source.host.os_family = XDM_CONST.OS_FAMILY_WINDOWS | filter xdm.event.id = "4624" | filter xdm.logon.type = "REMOTE_INTERACTIVE" // dest_priority and user_category field not found // filter dest_priority = "critical" // filter user_category != "domain admins" or user_category = "dont_expire_password" | replacenull xdm.source.ipv4 = "missing", xdm.target.user.username = "missing" | comp count() as total_event, earliest(_time) as first_event_time, latest(_time) as last_event_time, values(xdm.observer.product) as product, values(xdm.observer.vendor) as vendor, values(xdm.source.host.hostname) as host by xdm.source.ipv4, xdm.target.user.username, xdm.event.original_event_type // dest_priority, user_category
search `cim_change_indexes` sourcetype=azure:aad:audit activitydisplayname="add service principal" result=success (`micro_search_global_filtering_list("mscap - azure service principal addition raw (ccx) - summary gen")`) | rename initiatedby.user.userprincipalname as user targetresources{}.displayname as app | eval user=lower(user), signature=activitydisplayname | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | rename host as orig_host sourcetype as orig_sourcetype index as orig_index | rename _raw as orig_raw | rename _time as event_time | table status orig_raw event_time app user signature action attributevalue orig_host orig_sourcetype orig_index src dest vendor vendor_product ccx_customer_zone | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="defense evasion|lateral movement", metadata_techniques="t1550.001", metadata_attack_type="azuread|iaas", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft azure", metadata_index_macros="cim_change_indexes", metadata_cim_datamodels=null(), metadata_event_codes=null()| `finalise_micro_search("mscap - azure service principal addition raw (ccx) - summary gen", "*")`| `ccx_kill_switch`	// Title: MSCAP - Azure Service Principal Addition RAW (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: msft_azure_ad_audit_raw // Date: 07/June/2024 config case_sensitive = false | dataset = msft_azure_ad_audit_raw | filter activityDisplayName = "Add service principal" | filter result = "success" | alter user = json_extract_scalar(initiatedBy, "$.user.userPrincipalName") \| alter app = json_extract_scalar(targetResources, "$.0.displayName") | alter src = json_extract_scalar(initiatedBy, "$.user.ipAddress") | comp count() as total_events, values(src) as src_ip, earliest(activityDateTime) as first_event_time, latest(activityDateTime) as last_event_time, values(app) as app, values(_vendor) as vendor, values(_product) as produt by activityDisplayName, user, result

"search `cim_endpoint_indexes` sourcetype="ccx:o365:management:activity" operation="anonymouslinkcreated" (`micro_search_global_filtering_list("msana - large amount of anonymous link created")`) | bucket _time span=1h | stats dc(dest_name) as file_count values(dest_name) as file_list by _time userid appaccesscontext.clientappname operation sourcetype index | where file_count>20 | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | table vendor vendor_product ccx_customer_zone userid appaccesscontext.clientappname operation sourcetype index file_count file_list | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="transfer data to cloud account", metadata_techniques=t1537, metadata_attack_type=null(), metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen" | `finalise_micro_search("msana - large amount of anonymous link created o365","*")`"	// Title: MSANA - Large Amount of Anonymous Link Created O365 // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: msft_o365_sharepoint_online_raw // Date: 06/June/2024 config case_sensitive = false | dataset = msft_o365_sharepoint_online_raw | filter Operation = "anonymouslinkcreated" | alter client_app_name = AppAccessContext -> ClientAppName | alter client_app_id = AppAccessContext -> ClientAppId | comp count(SourceFileName) as file_count, values(SourceFileName) as file_list, earliest(CreationTime) as first_event_time, latest(CreationTime) as last_event_time, values(_vendor) as vendor, values(_product) as product, values(ClientIP) as src by UserId, client_app_id, client_app_name, operation | filter file_count > 20 // filtering for files count greater than 20
search `cim_vulnerabilities_indexes` eventtype=vulnerabilities vendor_product="tenable*" signature="microsoft windows startup software enumeration" output in ("*anydesk*", "*teamviewer*", "*logmein*", "*connectwise*", "*screenconnect*", "*mremoteng*", "*gotoassist*", "*zoho assist*", "*beyondtrust remote*", "*realvnc*", "*vnc connect*", "*tightvnc*", "*ultravnc*", "*bomgar*", "*splashtop*", "*atera*", "*supremo*", "*awesun*") or plugintext in ("*anydesk*", "*teamviewer*", "*logmein*", "*connectwise*", "*screenconnect*", "*mremoteng*", "*gotoassist*", "*zoho assist*", "*beyondtrust remote*", "*realvnc*", "*vnc connect*", "*tightvnc*", "*ultravnc*", "*bomgar*", "*splashtop*", "*atera*", "*supremo*", "*awesun*") (`cap_filter_vms_scanners("ipv6")`) (`cap_filter_vms_scanners("ipv4")`) (`micro_search_global_filtering_list("mscap - common abused remote access windows startup items - tenable raw (ccx) - summary gen")`) | rename output as orig_output | eval detection_name=if(isnull(plugin_name), pluginname, plugin_name), description=if(isnull(description), 'plugin.description', description), detection_family=if(isnull('family.name'), plugin_family, 'family.name'), detection_solution=if(isnull(solution), 'plugin.solution', solution), detection_id=if(isnull(pluginid), plugin_id, pluginid), detection_app=if(isnull(service), 'port.service', service), last_detected=if(isnull(lastseen), last_found, strftime(lastseen, "%ft%t%:z")), detection_output=if(isnull(orig_output), "unknown", orig_output), detection_severity=lower(severity), src_ip=if(isnotnull(ipv4), ipv4, ipv6), src_ip=if(cidrmatch("0.0.0.0/0", src_ip) or cidrmatch("::/0", src_ip), src_ip, "0") | fillnull description detection_family detection_solution detection_id detection_output detection_name detection_app value="unknown" | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | rex mode=sed field=detection_output "s/\n//g s/^\s+//g" | stats values(_raw) as orig_raw values(index) as orig_index values(sourcetype) as orig_sourcetype values(src) as src values(host) as orig_host values(user) as user values(dest) as dest values(detection_family) as detection_family values(detection_name) as detection_name values(description) as description values(detection_solution) as detection_solution values(detection_app) as detection_app values(port) as port values(detection_severity) as detection_severity values(detection_output) as detection_output values(vendor) as vendor values(vendor_product) as vendor_product count earliest(_time) as first_event_time latest(_time) as last_event_time by src_ip signature ccx_customer_zone | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="command and control|lateral movement", metadata_techniques="t1219|t1021.005", metadata_attack_type=null(), metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="tenable", metadata_vendor_products="tenable", metadata_index_macros="cim_vulnerabilities_indexes", metadata_cim_datamodels=null(), metadata_event_codes=null()| `finalise_micro_search("mscap - common abused remote access windows startup items - tenable raw (ccx) - summary gen", "ccx_customer_zone,src_ip,signature")`| `ccx_kill_switch`	// Title: MSCAP - Common Abused Remote Access Windows Startup Items - Tenable RAW (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: tenable_io_vulnerabilities_raw // Date: 18/June/2024 /* Alert Suppression : 6h Suppression Fields : xdm.alert.name, src_ip, detection_output */ config case_sensitive = false | datamodel dataset = tenable_io_vulnerabilities_raw | filter xdm.alert.name = "microsoft windows startup software enumeration" | filter xdm.event.description in ("*anydesk*", "*teamviewer*", "*logmein*", "*connectwise*", "*screenconnect*", "*mremoteng*", "*gotoassist*", "*zoho assist*", "*beyondtrust remote*", "*realvnc*", "*vnc connect*", "*tightvnc*", "*ultravnc*", "*bomgar*", "*splashtop*", "*atera*", "*supremo*", "*awesun*") | alter detection_solution = json_extract_scalar(tenable_io_vulnerabilities_raw.plugin, "$.solution"), detection_family = json_extract_scalar(tenable_io_vulnerabilities_raw.plugin, "$.family") | alter ipv4 = json_extract_scalar(tenable_io_vulnerabilities_raw.asset, "$.ipv4"), ipv6 = json_extract_scalar(tenable_io_vulnerabilities_raw.asset, "$.ipv6") // macros logic | filter ipv4 not in ("bvmswhp01", "10.10.144.10", "btenwh01","bvmsppp01", "10.40.144.12", "btenpp01","bvmsndp01", "10.50.144.10", "btenb201","bnesswh01", "10.10.143.21", "btenwh02","btenmi01", "10.25.144.10") | filter ipv6 not in ("bvmswhp01", "10.10.144.10", "btenwh01","bvmsppp01", "10.40.144.12", "btenpp01","bvmsndp01", "10.50.144.10", "btenb201","bnesswh01", "10.10.143.21", "btenwh02","btenmi01", "10.25.144.10") | alter src_ip = if(ipv4 in (null, ""), ipv6, ipv4) | comp count() as total_events, earliest(_time) as first_event_time, latest(_time) as last_event_time, values(xdm.target.host.hostname) as host, values(xdm.observer.vendor) as vendor, values(xdm.observer.product) as product, values(detection_family) as detection_family, values(xdm.target.host.os) as os, values(xdm.alert.description) as description, values(detection_solution) as detection_solution, values(xdm.alert.severity) as detetction_severity, values(xdm.target.host.fqdn) as host_fqdn, values(xdm.event.description) as detection_output by xdm.alert.name, src_ip

tstats allow_old_summaries=f summariesonly=t fillnull_value="missing" values(all_changes.result_id) as result_id count values(all_changes.vendor) as vendor values(all_changes.vendor_product) as vendor_product values(sourcetype) as orig_sourcetype values(all_changes.src) as src values(host) as orig_host from datamodel=change where (all_changes.result_id=4720 or all_changes.result_id=4726) and (`micro_search_global_filtering_list("mscap - short lived windows accounts acc (ccx) - summary gen")`) by _time all_changes.user all_changes.src all_changes.dest index host span=4h all_changes.account_management.src_user | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | rename all_changes.* as * | rename account_management.* as * | search result_id=4720 result_id=4726 | transaction user connected=f maxspan=240m | rename index as orig_index host as orig_host | eval event_time=strftime(_time, "%ft%t%:z") | table ccx_customer_zone orig_index orig_host orig_sourcetype event_time count user src_user src dest result_id vendor vendor_product | eval metadata_cis20="cis 16", metadata_killchainstage="exploitation", metadata_tactics="persistence", metadata_techniques="t1136.001|t1136", metadata_attack_type=null(), metadata_nist="pr.ip", metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft windows", metadata_index_macros=null(), metadata_cim_datamodels="change", metadata_event_codes="4720|4726"| `finalise_micro_search("mscap - short lived windows accounts acc (ccx) - summary gen", "*")`| `ccx_kill_switch`	// Title: MSCAP - Short Lived Windows Accounts ACC (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: microsoft_windows_raw // Date: 13/June/2024 config case_sensitive = false | datamodel dataset = microsoft_windows_raw | filter xdm.source.host.os_family = XDM_CONST.OS_FAMILY_WINDOWS | filter xdm.event.id in ("4726", "4720") | alter user_principal_name = json_extract_scalar(microsoft_windows_raw.event_data, "$.UserPrincipalName") \| alter user_display_name = json_extract_scalar(microsoft_windows_raw.event_data , "$.DisplayName") | alter user_creation_time = if(xdm.event.id = "4720", _time) | alter user_deletion_time = if(xdm.event.id = "4726", _time) | fields xdm.event.id, xdm.event.original_event_type, xdm.target.user.username, xdm.source.user.username, user_principal_name, user_display_name, microsoft_windows_raw.event_data, * | comp values(xdm.source.user.username) as activity_by_user, min(user_creation_time) as creation_time, max(user_deletion_time) as deletion_time, values(user_principal_name) as user_principal_name, values(user_display_name) as user_display_name, values(xdm.observer.vendor) as vendor, values(xdm.observer.product) as product by xdm.target.user.username | alter create_delete_time_diff = timestamp_diff(deletion_time, creation_time, "MINUTE") | filter create_delete_time_diff >= 0 and create_delete_time_diff <= 240 // in minutes
search `cim_authentication_indexes` sourcetype in ("azure:aad:risk:detection", "azure:aad:identity_protection:risk_detection") riskstate="atrisk" (`micro_search_global_filtering_list("mscap - azure active directory risky sign-in raw (ccx) - summary gen")`) | rename ipaddress as src_ip risklevel as severity userprincipalname as user riskeventtype as signature activity as operation | eval src=src_ip | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product userid userdisplayname id operation riskdetail signature additionalinfo value="missing" | stats values(action) as action values(_raw) as orig_raw values(dest) as dest values(host) as orig_host values(sourcetype) as orig_sourcetype values(index) as orig_index values(vendor) as vendor values(vendor_product) as vendor_product count earliest(_time) as first_event_time latest(_time) as last_event_time by additionalinfo src src_ip severity user userid userdisplayname id operation riskdetail signature riskstate ccx_customer_zone | eval _raw=replace(additionalinfo, "(?<=\"),(?=\"value)", ";") | extract kvdelim=":" pairdelim=";{}" mv_add=t | rex mode=sed field=value "s/\[|\]//g s/,/;/g" | eval _raw=mvjoin(mvzip(key, value, "="), ",") | extract kvdelim="=" pairdelim="," | fields - key value _raw count | eval riskreasons=split(riskreasons, ";") | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="initial access|defense evasion|persistence|privilege escalation", metadata_techniques="t1078.004", metadata_attack_type="azuread|iaas", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft azure", metadata_index_macros="cim_authentication_indexes", metadata_cim_datamodels=null(), metadata_event_codes=null()| `finalise_micro_search("mscap - azure active directory risky sign-in raw (ccx) - summary gen", "ccx_customer_zone,additionalinfo,src,src_ip,severity,user,userid, userdisplayname,id,operation,riskdetail,signature,riskstate")`| `ccx_kill_switch`	// Title: MSCAP - Azure Active Directory Risky Sign-in RAW (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: msft_azure_ad_raw // Date: 14/June/2024 config case_sensitive = false | dataset = msft_azure_ad_raw | filter riskState = "atrisk" | alter error_code = status -> errorCode, additional_details = status -> additionalDetails | alter browser = deviceDetail -> browser, os = deviceDetail -> operatingSystem | alter city = location -> city, country = location -> countryOrRegion, state = location -> state | alter location = format_string("%s | %s | %s", city, state, country) | replacenull userId = "missing", userDisplayName = "missing", riskDetail = "missing", riskEventTypes = "missing", additionalData = "missing" | comp count() as total_events, min(createdDateTime) as first_event_time, max(createdDateTime) as last_event_time, values(location) as location, values(browser) as browser, values(os) as operating_system, values(riskDetail) as risk_detail, values(additional_details) as additional_details, values(error_code) as error_code, values(ipAddress) as src_ip, values(riskEventTypes) as signature, values(riskLevelDuringSignIn) as severity by userDisplayName, userPrincipalName, riskState

search `cim_event_signatures_indexes` ((eventcode="4738" not (allowedtodelegateto="<value not set>" or allowedtodelegateto="-" or not allowedtodelegateto="*")) or ((eventcode="5136" attributeldapdisplayname="msds-allowedtodelegateto") or (eventcode="5136" objectclass="user" attributeldapdisplayname="serviceprincipalname") or (eventcode="5136" attributeldapdisplayname="msds-allowedtoactonbehalfofotheridentity"))) (`micro_search_global_filtering_list("mscap - active directory user backdoors raw (ccx) - summary gen")`) | rename host as orig_host sourcetype as orig_sourcetype index as orig_index | fillnull dest attributeldapdisplayname objectclass value="missing" | lookup index_to_ccx_customer_zone_lookup index_match as orig_index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | stats values(_raw) as orig_raw count values(vendor) as vendor values(signature_id) as signature_id values(vendor_product) as vendor_product values(orig_host) as orig_host values(orig_sourcetype) as orig_sourcetype values(orig_index) as orig_index values(src) as src values(user) as user values(src_user) as src_user earliest(_time) as first_event_time latest(_time) as last_event_time by dest eventcode ccx_customer_zone attributeldapdisplayname objectclass | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="persistence", metadata_techniques="t1098.001", metadata_attack_type=null(), metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft windows", metadata_index_macros="cim_event_signatures_indexes", metadata_cim_datamodels=null(), metadata_event_codes="4738|5136"| `finalise_micro_search("mscap - active directory user backdoors raw (ccx) - summary gen", "ccx_customer_zone,dest,eventcode,attributeldapdisplayname,objectclass")`| `ccx_kill_switch`	// Title: MSCAP - Active Directory User Backdoors RAW (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: microsoft_windows_raw // Date: 11/June/2024 config case_sensitive = false | datamodel dataset = microsoft_windows_raw | filter xdm.source.host.os_family = XDM_CONST.OS_FAMILY_WINDOWS | alter allowed_to_delegate_to = json_extract_scalar(microsoft_windows_raw.event_data, "$.AllowedToDelegateTo") \| alter attribute_ldap_display_name = json_extract_scalar(microsoft_windows_raw.event_data, "$.AttributeLDAPDisplayName") | alter object_class = json_extract_scalar(microsoft_windows_raw.event_data, "$.ObjectClass") | replacenull attribute_ldap_display_name = "missing", object_class = "missing" | alter p1 = if(xdm.event.id = "4738" and allowed_to_delegate_to not in ("-", "*", "<value not set>"), true, false) // do we need to filter out "" and null values for allowed_to_delegate_to | alter p2 = if(xdm.event.id = "5136" and attribute_ldap_display_name = "msds-allowedtodelegateto", true, false) | alter p3 = if(xdm.event.id = "5136" and object_class = "user" and attribute_ldap_display_name = "serviceprincipalname", true, false) | alter p4 = if(xdm.event.id = "5136" and attribute_ldap_display_name = "msds-allowedtoactonbehalfofotheridentity", true, false) | alter result = if(p1 or (p2 or p3 or p4), true, false) | filter result = true | comp count() as total_events, earliest(_time) as first_event_time, latest(_time) as last_event_time, values(xdm.source.host.hostname) as host, values(xdm.observer.vendor) as vendor, values(xdm.observer.product) as product, values(xdm.source.user.username) as user by xdm.event.id, xdm.event.original_event_type, object_class, attribute_ldap_display_name
search `cim_event_signatures_indexes` eventtype=wineventlog_security eventcode=4768 status=0x6 user != "*$" (`micro_search_global_filtering_list("mscap - kerberos user enumeration raw (ccx) - summary gen")`) | bucket span=2m _time as bucket_time | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product ipaddress value="missing" | stats values(_raw) as orig_raw dc(user) as unique_accounts values(user) as user values(host) as orig_host values(eventcode) as eventcode values(signature_id) as signature_id values(sourcetype) as orig_sourcetype values(index) as orig_index values(src) as src values(dest) as dest values(vendor) as vendor values(vendor_product) as vendor_product earliest(_time) as first_event_time latest(_time) as last_event_time by bucket_time ipaddress ccx_customer_zone | eventstats avg(unique_accounts) as comp_avg stdev(unique_accounts) as comp_std by ipaddress ccx_customer_zone | eval upperbound=(comp_avg + comp_std * 3) | eval isoutlier=if(unique_accounts > 10 and unique_accounts >= upperbound, 1, 0) | search isoutlier=1 | eval metadata_cis20=null(), metadata_killchainstage="exploitation", metadata_tactics="credential access", metadata_techniques="t1110.003", metadata_attack_type="windows", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft windows", metadata_index_macros="cim_event_signatures_indexes", metadata_cim_datamodels=null(), metadata_event_codes="4768"| `finalise_micro_search("mscap - kerberos user enumeration raw (ccx) - summary gen", "ccx_customer_zone,ipaddress,bucket_time")`| `ccx_kill_switch`	// Title: MSCAP - Kerberos User Enumeration RAW (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: microsoft_windows_raw // Date: 11/June/2024 config case_sensitive = false | datamodel dataset = microsoft_windows_raw | filter xdm.source.host.os_family = XDM_CONST.OS_FAMILY_WINDOWS | filter xdm.event.id = "4768" | alter status = json_extract_scalar(microsoft_windows_raw.event_data, "$.Status") \| filter status = "0x6" \| filter xdm.target.user.username != "*$" | replacenull xdm.source.ipv4 = "missing" | fields _time, xdm.event.id, status, xdm.source.host.hostname, xdm.target.user.username, xdm.source.ipv4, xdm.target.ipv4, microsoft_windows_raw.event_data, * | comp count() as total_events, count_distinct(xdm.target.user.username) as unique_user_count, values(xdm.target.user.username) as user, earliest(_time) as first_event_time, latest(_time) as last_event_time, values(xdm.observer.vendor) as vendor, values(xdm.observer.product) as product by xdm.event.original_event_type, xdm.event.id, xdm.source.ipv4 | filter unique_user_count > 10 // filtering for more than 10 attempts to enumerate user accoutns user Kerbros

search `cim_alerts_indexes` sourcetype="azure:securitycenter:task" (`micro_search_global_filtering_list("mscap - microsoft azure security center tasks raw (ccx) - summary gen")`) | rename properties.securitytaskparameters.category as category properties.securitytaskparameters.name as signature properties.securitytaskparameters.policyname as description properties.state as detection_status properties.securitytaskparameters.resourcetype as detection_resource_type name as detection_id properties.creationtimeutc as detection_time properties.securitytaskparameters.resourceid as orig_resource_id properties.securitytaskparameters.vmname as vm_name | eval detection_status=lower(detection_status), description=if(isnull(description), signature, description) | rex field=orig_resource_id ".+\/(?<dest>(.*$))" \| eval temp_field=mvjoin(mvappend(signature, detection_id, detection_time, category, description, detection_status, detection_resource_type, dest), "@@@@") \| rex field=temp_field max_match=0 "^(?<signature>.*)@@@@(?<detection_id>.*)@@@@(?<detection_time>.*)@@@@(?<category>.*)@@@@(?<description>.*)@@@@(?<detection_status>.*)@@@@(?<detection_resource_type>.*)@@@@(?<dest>.*)$" | fields - temp_field | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | stats values(_raw) as orig_raw earliest(_time) as first_event_time latest(_time) as last_event_time values(severity) as severity values(vendor) as vendor values(vendor_product) as vendor_product values(detection_time) as detection_time values(detection_id) as detection_id values(dest) as dest values(category) as category values(description) as description values(detection_resource_type) as detection_resource_type values(detection_status) as detection_status values(host) as orig_host values(sourcetype) as orig_sourcetype values(index) as orig_index values(src) as src values(user) as user by signature ccx_customer_zone | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics=null(), metadata_techniques=null(), metadata_attack_type="azuread|iaas", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft azure", metadata_index_macros="cim_alerts_indexes", metadata_cim_datamodels=null(), metadata_event_codes=null()| `finalise_micro_search("mscap - microsoft azure security center tasks raw (ccx) - summary gen", "ccx_customer_zone,signature")`| `ccx_kill_switch`	// Title: MSCAP - Microsoft Azure Security Center Tasks RAW (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: msft_graph_security_alerts_raw // Date: 16/July/2024 config case_sensitive = false | dataset = msft_graph_security_alerts_raw | alter userPrincipalName1 = json_extract_scalar(evidence, "$.0.primaryAddress"), userPrincipalName2 = json_extract_scalar(evidence, "$.0.userAccount.userPrincipalName"), accountName = json_extract_scalar(evidence, "$.0.userAccount.accountName") | alter userPrincipalName = coalesce(userPrincipalName1, userPrincipalName2) | comp count() as total_event, earliest(createdDateTime) as first_event_time, latest(createdDateTime) as last_event_time, values(createdDateTime) as detection_time, values(detectorId) as detection_id, values(alertWebUrl) as alert_web_url, values(id) as alertId, values(incidentId) as incidentId, values(incidentWebUrl) as incident_web_url, values(category) as cateory, values(mitreTechniques) as mitreTechniques, values(description) as description, values(detectionSource) as orig_sourcetype, values(status) as detection_status, values(severity) as severity, values(_vendor) as vendor, values(_product) as product, values(userPrincipalName) as userPrincipalName, values(accountName) as accountName by title | fields total_event, first_event_time, last_event_time, detection_time, accountName, userPrincipalName, title as signature, description, detection_status, detection_id, alertId, alert_web_url, incidentId, incident_web_url, cateory, mitreTechniques, orig_sourcetype, severity, vendor, product
search index=* user in ("jack.yangbin@otmlhome.onmicrosoft.com" "guise.wartoto@otmlhome.onmicrosoft.com") sourcetype="azure:aad:signin" and (`micro_search_global_filtering_list("msana - breakglass account login detection (ps) (ccx) - summary gen")`) | rename networklocationdetails{}.networknames{} as network_country, authenticationdetails{}.authenticationmethod as authentication_method_details, authenticationdetails{}.authenticationmethoddetail as authentication_how, authenticationdetails{}.authenticationstepresultdetail as authentication_result | rename _raw as orig_raw | table _time user src_ip action status.errorcode status.failurereason appdisplayname clientappused location.countryorregion network_country authentication_method_details authentication_how authentication_result sourcetype index vendor_product orig_raw | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | table user src_ip action status.errorcode status.failurereason appdisplayname clientappused location.countryorregion network_country authentication_method_details authentication_how authentication_result sourcetype index vendor_product vendor orig_raw | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics=null(), metadata_techniques=null(), metadata_attack_type=null(), metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen" | `finalise_micro_search("msana - breakglass account login detection (ps) (ccx) - summary gen", "*")` | `ccx_kill_switch`	// Title: MSANA - Breakglass Account Login Detection (PS) (CCX) - Summary Gen // Author: Deven Amode, damode@paloaltonetworks.com // Date: 17/June/2024 dataset = msft_azure_ad_raw |alter user = userPrincipalName, action = json_extract_scalar(status,"$.errorCode"), status_failurereason = json_extract_scalar(status,"$.failureReason"), src_ip = ipAddress, location_countryorregion = json_extract_scalar(location,"$.countryOrRegion"), network_country= json_extract_scalar(networkLocationDetails,"$.networknames"), authentication_method_details = json_extract_scalar(authenticationdetails,"$.authenticationmethod"), authentication_how = json_extract_scalar(authenticationdetails,"$.authenticationmethoddetail"), authentication_result = json_extract_scalar(authenticationdetails,"$.authenticationstepresultdetail") |filter user in ("jack.yangbin@otmlhome.onmicrosoft.com", "guise.wartoto@otmlhome.onmicrosoft.com") | fields user, src_ip, action, status_failurereason, appdisplayname, clientappused, location_countryorregion, network_country, authentication_method_details, authentication_how, authentication_result, _collector_type, _product, _vendor

search `cim_change_indexes` sourcetype=azure:aad:user accountenabled=* (`micro_search_global_filtering_list("mscap - azure aad user enabled from disabled state raw (ccx) - summary gen")`) [| search `cim_change_indexes` accountenabled=* sourcetype=azure:aad:user earliest=-48h latest=now | fillnull userprincipalname value="missing" | stats values(action) as action dc(accountenabled) as unique_status by userprincipalname | where unique_status > 1 | table action userprincipalname] | sort 0 userprincipalname ccx_customer_zone _time | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | streamstats current=f window=1 global=f last(accountenabled) as previous_status last(_time) as last_seen_disabled_time by userprincipalname ccx_customer_zone | where _time > relative_time (now(), "-4h") and previous_status="false" and accountenabled="true" | eval current_status=if (accountenabled="true", "account enabled", "account disabled"), previous_status=if(previous_status="false", "account disabled", "account enabled"), event_time=_time, user=userprincipalname, job_title=jobtitle | eval orig_host=host, orig_sourcetype=sourcetype, orig_index=index, orig_raw=_raw | fillnull src dest vendor vendor_product value="missing" | table action event_time user displayname current_status previous_status last_seen_disabled_time job_title id ccx_customer_zone src dest vendor vendor_product orig_host orig_sourcetype orig_index orig_raw | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="persistence", metadata_techniques="t1078.004", metadata_attack_type="azuread|iaas", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft azure", metadata_index_macros="cim_change_indexes", metadata_cim_datamodels=null(), metadata_event_codes=null()| `finalise_micro_search("mscap - azure aad user enabled from disabled state raw (ccx) - summary gen", "*")`| `ccx_kill_switch`	// Title: MSCAP - Azure AAD User Enabled from Disabled State RAW (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: msft_azure_ad_audit_raw // Date: 14/June/2024 config case_sensitive = false | dataset = msft_azure_ad_audit_raw | alter modified_properties = json_extract_array(targetResources, "$.0.modifiedProperties") \| alter type = json_extract_scalar(targetResources, "$.0.type") | filter type = "User" | alter account_enabled = arraymap(modified_properties, if("@element" -> displayName = "accountEnabled", true)) | filter account_enabled = true | filter result = "success" | alter previous_status = arraymap(modified_properties, if("@element" -> displayName = "accountEnabled", "@element" -> oldValue)) | filter previous_status contains "false" | alter current_status = arraymap(modified_properties, if("@element" -> displayName = "accountEnabled", "@element" -> newValue)) | filter current_status contains "true" | alter activity_by_user = json_extract_scalar(initiatedBy, "$.user.userPrincipalName") \| alter userPrincipalName = json_extract_scalar(targetResources, "$.0.userPrincipalName") | replacenull userPrincipalName = "missing" | comp min(activityDateTime) as last_seen_disabled, values(activityDisplayName) as action by userPrincipalName, activity_by_user, result, operationType, category
search `cim_change_indexes` sourcetype=azure:aad:audit activitydisplayname="add service principal credentials" result=success (`micro_search_global_filtering_list("mscap - azure service principal credentials added raw (ccx) - summary gen")`) | rename initiatedby.user.userprincipalname as user targetresources{}.displayname as app | eval user=lower(user), signature=activitydisplayname | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | rename host as orig_host sourcetype as orig_sourcetype index as orig_index | rename _raw as orig_raw | table status orig_raw app user signature action orig_host orig_sourcetype orig_index src dest vendor vendor_product ccx_customer_zone _time | rename _time as event_time | eval metadata_cis20=null(), metadata_killchainstage="actions on objective", metadata_tactics="impact|persistence", metadata_techniques="t1496|t1136.003", metadata_attack_type="azuread|iaas", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft azure", metadata_index_macros="cim_change_indexes", metadata_cim_datamodels=null(), metadata_event_codes=null()| `finalise_micro_search("mscap - azure service principal credentials added raw (ccx) - summary gen", "*")`| `ccx_kill_switch`	// Title: MSCAP - Azure Service Principal Credentials Added RAW (CCX) - Summary Gen // Author: Deven Amode, damode@paloaltonetworks.com // Datasets: msft_azure_ad_audit_raw // Date: 11/June/2024 config case_sensitive = false | dataset = msft_azure_ad_audit_raw | filter activityDisplayName = "*add service principal credentials*" | filter result = "success" | alter user = json_extract_scalar(initiatedBy, "$.user.userPrincipalName") \| alter src = json_extract_scalar(initiatedBy, "$.user.ipAddress") | alter app = json_extract_scalar(targetResources, "$.0.displayName") | fields _time, app, user, activityDateTime as signature, _collector_type , category, _vendor, _product

search `cim_change_indexes` sourcetype=azure:aad:audit activitydisplayname="*add owner to application*" result=success (`micro_search_global_filtering_list("mscap - user added as owner for azure application raw (ccx) - summary gen")`) | rename initiatedby.user.userprincipalname as user targetresources{}.displayname as app | eval user=lower(user), signature=activitydisplayname | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | rename host as orig_host sourcetype as orig_sourcetype index as orig_index | rename _raw as orig_raw | table status orig_raw app user signature action orig_host orig_sourcetype orig_index src dest vendor vendor_product ccx_customer_zone _time | rename _time as event_time | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="persistence", metadata_techniques="t1098.001", metadata_attack_type="azuread|iaas", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft azure", metadata_index_macros="cim_change_indexes", metadata_cim_datamodels=null(), metadata_event_codes=null()| `finalise_micro_search("mscap - user added as owner for azure application raw (ccx) - summary gen", "*")`| `ccx_kill_switch`	// Title: MSCAP - User Added as Owner for Azure Application RAW (CCX) - Summary Gen // Author: Deven Amode, damode@paloaltonetworks.com // Datasets: msft_azure_ad_audit_raw // Date: 11/June/2024 config case_sensitive = false | dataset = msft_azure_ad_audit_raw | filter activityDisplayName = "*add owner to application*" | filter result = "success" | alter user = json_extract_scalar(initiatedBy, "$.user.userPrincipalName") \| alter src = json_extract_scalar(initiatedBy, "$.user.ipAddress") | alter app = json_extract_scalar(targetResources, "$.0.displayName") | fields activityDateTime, user, src, app, activityDisplayName, operationType, category, _vendor, _product, result, * | comp count() as total_events, values(src) as src_ip, earliest(activityDateTime) as first_event_time, latest(activityDateTime) as last_event_time, values(app) as app, values(_vendor) as vendor, values(_product) as product by activityDisplayName, user, result
search `cim_event_signatures_indexes` eventtype=wineventlog_windows signature_id in ("4706", "4707", "4716") (`micro_search_global_filtering_list("mscap - windows domain trust modification via windows event code raw (ccx) - summary gen")`) | eval user=mvappend(user, caller_user_name) | fillnull eventcode dest signature signature_id src user value="missing" | eval signature=case (signature_id="4706", "a new trust was created to a domain.", signature_id="4707", "a trust to a domain was removed.", signature_id="4716", "trusted domain information was modified.") | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | stats values (_raw) as orig_raw earliest(_time) as first_event_time latest(_time) as last_event_time values (eventcode) as eventcode values(src) as src values(caller_domain) as src_nt_domain values(user) as user values(host) as orig_host values(sourcetype) as orig_sourcetype values(index) as orig_index count by ccx_customer_zone dest signature signature_id vendor vendor_product | table orig_raw first_event_time last_event_time vendor vendor_product dest signature eventcode signature_id orig_host orig_sourcetype orig_index count src user ccx_customer_zone src_nt_domain | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="defense evasion|privilege escalation", metadata_techniques="t1484.002", metadata_attack_type="windows", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft windows", metadata_index_macros="cim_event_signatures_indexes", metadata_cim_datamodels=null(), metadata_event_codes="4706|4707|4716"| `finalise_micro_search("mscap - windows domain trust modification via windows event code raw (ccx) - summary gen", "ccx_customer_zone,dest,signature, signature_id,vendor,vendor_product")`| `ccx_kill_switch`	// Title: MSCAP - Windows Domain Trust Modification via Windows Event Code RAW (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: microsoft_windows_raw // Date: 17/June/2024 config case_sensitive = false | datamodel dataset = microsoft_windows_raw | filter xdm.source.host.os_family = XDM_CONST.OS_FAMILY_WINDOWS | filter xdm.event.id in ("4706", "4707", "4716") | alter signature = if(xdm.event.id = "4706", "a new trust was created to a domain.", xdm.event.id = "4707", "a trust to a domain was removed.", xdm.event.id = "4716", "trusted domain information was modified.") | replacenull xdm.source.ipv4 = "missing", xdm.source.user.username = "missing" | fields xdm.event.id, signature, xdm.event.original_event_type, xdm.source.ipv4, xdm.source.user.username, microsoft_windows_raw.event_data, * | comp count() as total_events, min(_time) as first_event_time, max(_time) as last_event_time, values(xdm.source.user.domain) as src_nt_domain, values(xdm.source.user.user_type) as user, values(xdm.source.host.hostname) as host, values(xdm.observer.product) as product, values(xdm.observer.vendor) as vendor by xdm.event.id, signature, xdm.event.original_event_type
search `cim_change_indexes` sourcetype=azure:aad:audit activitydisplayname="*add owner to service principal*" result=success (`micro_search_global_filtering_list("mscap - user added as owner for azure service principal raw (ccx) - summary gen")`) | rename initiatedby.user.userprincipalname as user targetresources{}.displayname as app | eval user=lower(user), signature=activitydisplayname | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | rename host as orig_host sourcetype as orig_sourcetype index as orig_index | rename _raw as orig_raw | table status orig_raw app user signature action orig_host orig_sourcetype orig_index src dest vendor vendor_product ccx_customer_zone _time | rename _time as event_time | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="persistence", metadata_techniques="t1098.001", metadata_attack_type="azuread|iaas", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft azure", metadata_index_macros="cim_change_indexes", metadata_cim_datamodels=null(), metadata_event_codes=null()| `finalise_micro_search("mscap - user added as owner for azure service principal raw (ccx) - summary gen", "*")`| `ccx_kill_switch`	// Title: MSCAP - User Added as Owner for Azure Service Principal RAW (CCX) - Summary Gen // Author: Deven Amode, damode@paloaltonetworks.com // Datasets: msft_azure_ad_audit_raw // Date: 11/June/2024 config case_sensitive = false | dataset = msft_azure_ad_audit_raw | filter activityDisplayName = "*add owner to service principal*" and result = "success" | alter user = json_extract_scalar(initiatedBy, "$.user.userPrincipalName"), src = json_extract_scalar(initiatedBy, "$.user.ipAddress"), app = json_extract_scalar(targetResources, "$.0.displayName"), user_type = json_extract_scalar(targetResources, "$.0.type") | fields _time, app, user, user_type, activityDisplayName as signature, _collector_type , _vendor, _product

search `cim_event_signatures_indexes` (((name="'microsoft-windows-security-auditing'" eventcode="4732") (targetusername="administr*" or targetsid="s-1-5-32-544")) not (subjectusername="*$")) (`micro_search_global_filtering_list("mscap - user added to local administrators raw (ccx) - summary gen")`) | rename host as orig_host sourcetype as orig_sourcetype index as orig_index | fillnull dest src user value="missing" | lookup index_to_ccx_customer_zone_lookup index_match as orig_index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | stats values(vendor) as vendor values(_raw) as orig_raw values (vendor_product) as vendor_product values(signature_id) as signature_id values(orig_host) as orig_host values(orig_sourcetype) as orig_sourcetype values(orig_index) as orig_index values(src) as src earliest(_time) as first_event_time latest(_time) as last_event_time by subjectusername dest user eventcode ccx_customer_zone | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="privilege escalation|persistence", metadata_techniques="t1078|t1098", metadata_attack_type=null(), metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft windows", metadata_index_macros=" cim_event_signatures_indexes", metadata_cim_datamodels=null(), metadata_event_codes="4732"| `finalise_micro_search("mscap - user added to local administrators raw (ccx) - summary gen", "ccx_customer_zone,subjectusername,user,dest,eventcode")`| `ccx_kill_switch`	// Title: MSCAP - User Added to Local Administrators RAW (CCX) - Summary Gen // Author: Deven Amode, damode@paloaltonetworks.com // Date: 13/June/2024 config case_sensitive = false | datamodel dataset = microsoft_windows_raw | alter xdm.target.user.username = json_extract_scalar(microsoft_windows_raw.event_data, "$.TargetUserName"), xdm.target.resource.id =json_extract_scalar(microsoft_windows_raw.event_data,"$.targetsid"), subject_username = json_extract_scalar(microsoft_windows_raw.event_data, "$.SubjectUserName") \| filter xdm.event.id = "4732" and xdm.target.user.username = "administr*" or xdm.target.resource.id="s-1-5-32-544" and subject_username != "*$" | fields *, xdm.source.user.username as user,microsoft_windows_raw._raw_log, xdm.event.id as event_code,xdm.target.ipv4 as dest | comp values(microsoft_windows_raw._raw_log) as orig_raw, values(_vendor) as vendor , values(_product) as vendor_product , values(xdm.source.host.hostname) as orig_host, values(xdm.observer.type)as orig_sourcetype, earliest(_time) as first_event_time, latest(_time) as last_event_time by xdm.target.user.username , dest, user, event_code
search `cim_change_indexes` `ccx_o365_management_activity_sourcetypes` app=exchange operation="new-inboxrule" resultstatus=true (`micro_search_global_filtering_list("mscap - o365 new inbox rule raw (ccx) - summary gen")`) | eval user=lower(user), signature=activitydisplayname | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | rename host as orig_host sourcetype as orig_sourcetype index as orig_index | rename _raw as orig_raw _time as event_time | table status orig_raw event_time app user signature action orig_host orig_sourcetype orig_index src dest vendor vendor_product ccx_customer_zone | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="collection", metadata_techniques="t1114.003", metadata_attack_type="o365", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft office 365", metadata_index_macros="cim_change_indexes", metadata_cim_datamodels=null(), metadata_event_codes=null()| `finalise_micro_search("mscap - o365 new inbox rule raw (ccx) - summary gen", "*")`| `ccx_kill_switch`	// Title: MSCAP - O365 New Inbox Rule RAW (CCX) - Summary Gen // Author: Deven Amode, damode@paloaltonetworks.com // Date: 13/June/2024 config case_sensitive = false | dataset = msft_o365_exchange_online_raw | filter operation = "new-inboxrule" and ResultStatus="true" | fields ResultStatus as status, _time, UserId as user, operation as action, _reporting_device_name as orig_host, _collector_type, ClientIP as src,_product,_vendor
search `cim_event_signatures_indexes` eventcode=4624 logon_type=10 ipaddress in ("127.0.0.1", "::1") (`micro_search_global_filtering_list("mscap - rdp login from localhost raw (ccx) - summary gen")`) | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | rename host as orig_host sourcetype as orig_sourcetype index as orig_index | fillnull dest value="missing" | stats values(_raw) as orig_raw earliest(_time) as first_event_time latest(_time) as last_event_time values(signature_id) as signature_id values(vendor) as vendor values (vendor_product) as vendor_product values(orig_host) as orig_host values(orig_sourcetype) as orig_sourcetype values(orig_index) as orig_index values(src) as src values(user) as user by dest eventcode logon_type ccx_customer_zone | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="lateral movement", metadata_techniques="t1021.001", metadata_attack_type=null(), metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft windows", metadata_index_macros=" cim_event_signatures_indexes", metadata_cim_datamodels=null(), metadata_event_codes="4624"| `finalise_micro_search("mscap - rdp login from localhost raw (ccx) - summary gen", "ccx_customer_zone,dest,eventcode,logon_type")`| `ccx_kill_switch`	// Title: MSCAP - RDP Login from Localhost RAW (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: microsoft_windows_raw // Date: 07/June/2024 config case_sensitive = false | datamodel dataset = microsoft_windows_raw | filter xdm.source.host.os_family = XDM_CONST.OS_FAMILY_WINDOWS | filter xdm.event.id = "4624" | filter xdm.logon.type = "REMOTE_INTERACTIVE" | filter xdm.source.ipv4 in ("127.0.0.1", "::1") | fields _time, xdm.event.id, xdm.event.original_event_type, xdm.logon.type, xdm.source.host.hostname, xdm.source.ipv4, xdm.observer.vendor, xdm.observer.product, xdm.target.user.username, xdm.target.user.upn, xdm.source.host.hostname, xdm.event.description

tstats allow_old_summaries=f summariesonly=t fillnull_value="missing" count min(_time) as first_event_time max(_time) as last_event_time values(all_email.vendor) as vendor values(all_email.vendor_product) as vendor_product values(sourcetype) as orig_sourcetype values(all_email.src) as src values(all_email.dest) as dest values(host) as orig_host from datamodel=email where (all_email.file_name in ("*.avi.com", "*.avi.exe", "*.doc.com", "*.doc.exe", "*.docx.com", "*.docx.exe", "*.jpg.com", "*.jpg.exe", "*.jpeg.com", "*.jpeg.exe", "*.mpg.com", "*.mpg.exe", "*.mpg2.com", "*.mpg2.exe", "*.mpeg.com", "*.mpeg.exe", "*.pdf.com", "*.pdf.exe", "*.png.com", "*.png.exe", "*.ppt.com", "*.ppt.exe", "*.pptx.com", "*.pptx.exe", "*.swf.com", "*.swf.exe", "*.xls.com", "*.xls.exe", "*.xlsx.com", "*.xlsx.exe", "*.zip.com", "*.zip.exe", "*.bat", "*.chm", "*.com", "*.cmd", "*.cpl", "*.exe", "*.hlp", "*.hta", "*.jar", "*.js", "*.msi", "*.pif", "*.ps1", "*.rar", "*.reg", "*.scr", "*.vbe", "*.vbs", "*.wsf")) and (`micro_search_global_filtering_list("mscap - suspicious email attachment extensions acc (ccx) - summary gen")`) by all_email.src_user all_email.file_name all_email.message_id index host | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | rename all_email.* as * | stats values(vendor) as vendor values(vendor_product) as vendor_product min(first_event_time) as first_event_time max(last_event_time) as last_event_time values(src) as src values(dest) as dest values(host) as orig_host values(orig_sourcetype) as orig_sourcetype values(index) as orig_index values(file_name) as file_name by src_user ccx_customer_zone message_id | eval metadata_cis20="cis 3|cis 7|cis 12", metadata_killchainstage="delivery", metadata_tactics="initial access", metadata_techniques="t1566.001|t1566", metadata_attack_type=null(), metadata_nist="de.ae|pr.ip", metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor=null(), metadata_vendor_products=null(), metadata_index_macros=null(), metadata_cim_datamodels="email", metadata_event_codes=null()| `finalise_micro_search("mscap - suspicious email attachment extensions acc (ccx) - summary gen", "ccx_customer_zone,src_user,message_id")`| `ccx_kill_switch`	// Title: MSCAP - Suspicious Email Attachment Extensions ACC (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: // Date: 13/June/2024 config case_sensitive = false | datamodel dataset = mimecast_mimecast_raw | filter xdm.email.attachment.filename in ("*.avi.com", "*.avi.exe", "*.doc.com", "*.doc.exe", "*.docx.com", "*.docx.exe", "*.jpg.com", "*.jpg.exe", "*.jpeg.com", "*.jpeg.exe", "*.mpg.com", "*.mpg.exe", "*.mpg2.com", "*.mpg2.exe", "*.mpeg.com", "*.mpeg.exe", "*.pdf.com", "*.pdf.exe", "*.png.com", "*.png.exe", "*.ppt.com", "*.ppt.exe", "*.pptx.com", "*.pptx.exe", "*.swf.com", "*.swf.exe", "*.xls.com", "*.xls.exe", "*.xlsx.com", "*.xlsx.exe", "*.zip.com", "*.zip.exe", "*.bat", "*.chm", "*.com", "*.cmd", "*.cpl", "*.exe", "*.hlp", "*.hta", "*.jar", "*.js", "*.msi", "*.pif", "*.ps1", "*.rar", "*.reg", "*.scr", "*.vbe", "*.vbs", "*.wsf") | comp count() as total_events, values(xdm.observer.vendor) as vendor, values(xdm.observer.product) as product, earliest(_time) as first_event_time, latest(_time) as last_event_time, values(xdm.email.attachment.filename) as file_name by xdm.email.sender, xdm.email.message_id | join (datamodel dataset = mimecast_mimecast_raw | alter email_recipients = arraystring(xdm.email.recipients, ",") // filtering for null or empty results in attachment filename, message_id and recipients | filter xdm.email.attachment.filename in ("", null) | filter xdm.email.message_id not in ("", null) | filter email_recipients not in ("", null) | fields xdm.email.message_id as message_id, xdm.email.sender as email_sender, email_recipients | comp values(email_recipients) as email_recipients by message_id, email_sender | limit 10000000) as recipient_data recipient_data.message_id = xdm.email.message_id | fields total_events, first_event_time, last_event_time, xdm.email.message_id, xdm.email.sender, email_recipients, file_name, vendor, product

search `cim_change_indexes` sourcetype=azure:aad:audit activitydisplayname="update conditional access policy" result=success (`micro_search_global_filtering_list("mscap - azure conditional access policy modified raw (ccx) - summary gen")`) | rename initiatedby.user.userprincipalname as user targetresources{}.displayname as app | eval user=lower(user), signature=activitydisplayname | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | rename host as orig_host sourcetype as orig_sourcetype index as orig_index | eval targetresources_modifiedproperties=mvrange(0, mvcount('targetresources{}.modifiedproperties{}.displayname'), 1) | rex mode=sed field=targetresources_modifiedproperties "s/.*/ccx_placeholder_value/g" | foreach targetresources{}.modifiedproperties{}.* [| eval "<<field>>"=mvmap('<<field>>', replace('<<field>>', "^", "<<matchstr>>: "))] | foreach targetresources.modifiedproperties.* [| eval targetresources_modifiedproperties=mvzip(targetresources_modifiedproperties, '<<field>>', ", ")] | rex mode=sed field=targetresources_modifiedproperties "s/ccx_placeholder_value(,)?//g s/^{//g s/$/}/g" | rex mode=sed field=targetresources_modifiedproperties "s/\\\\\\"//g s/\"//g" | fillnull targetresources_modifiedproperties value="missing" | rename _time as event_time | rename _raw as orig_raw targetresources.modifiedproperties.newvalue as targetresources_modifiedproperties_newvalue targetresources.modifiedproperties.oldvalue as targetresources_modifiedproperties_oldvalue | table status orig_raw event_time targetresources_modifiedproperties targetresources_modifiedproperties_newvalue targetresources_modifiedproperties_oldvalue app user signature action orig_host orig_sourcetype orig_index src dest vendor vendor_product ccx_customer_zone | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="persistence", metadata_techniques="t1098.001", metadata_attack_type="azuread|iaas", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft azure", metadata_index_macros="cim_change_indexes", metadata_cim_datamodels=null(), metadata_event_codes=null()| `finalise_micro_search("mscap - azure conditional access policy modified raw (ccx) - summary gen", "*")`| `ccx_kill_switch`	// Title: MSCAP - Azure Conditional Access Policy Modified RAW (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: msft_azure_ad_audit_raw // Date: 12/June/2024 config case_sensitive = false | dataset = msft_azure_ad_audit_raw | filter activityDisplayName = "update conditional access policy" | alter user = json_extract_scalar(initiatedBy, "$.user.userPrincipalName") \| alter src = json_extract_scalar(initiatedBy, "$.user.ipAddress") | alter app = json_extract_scalar(targetResources, "$.0.displayName") \| filter result = "success" \| alter modified_properties = json_extract_array(targetResources, "$.0.modifiedProperties") | alter modified_properties_old_value = arraymap(modified_properties, if("@element" -> displayName = "ConditionalAccessPolicy", "@element" -> oldValue)) | alter modified_properties_new_value = arraymap(modified_properties, if("@element" -> displayName = "ConditionalAccessPolicy", "@element" -> newValue)) | fields activityDateTime, activityDisplayName, category, operationType, result, user, src, app, modified_properties, modified_properties_old_value, modified_properties_new_value, _vendor, _product
search index=idm_azure or index=ccx_synthetiks sourcetype in ("azure:aad:risk:detection", "azure:aad:identity_protection:risk_detection") location.countryorregion!=au location.countryorregion!=pg | rename ipaddress as src userprincipalname as user location.countryorregion as tld | iplocation src | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | table vendor vendor_product ccx_customer_zone _time user src tld | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics=null(), metadata_techniques=null(), metadata_attack_type=null(), metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen" | `finalise_micro_search("msana - azure risky sign-in outside au/pg (ccx) - summary gen","*")`	// Title: MSANA - Azure Risky Sign-in Outside AU/PG (CCX) - Summary Gen // Author: Deven Amode, damode@paloaltonetworks.com // Date: 12/June/2024 dataset = msft_azure_ad_raw |alter country_or_region = json_extract_scalar(location , "$.countryOrRegion") |filter country_or_region not in ("au", "pg", "") | iploc ipAddress loc_continent AS Continent, loc_country AS Country, loc_region AS Region, loc_city AS City, loc_latlon AS lon |fields _time, ipAddress as src,userprincipalname as user, country_or_region as tld , _vendor ,_product

search `cim_change_indexes` sourcetype=azure:aad:audit activitydisplayname="*member to role*" result=success "targetresources{}.displayname"="global administrator" or "targetresources{}.modifiedproperties{}.newvalue"="\"62e90394-69f5-4237-9190-012177145e10\"" (`micro_search_global_filtering_list("mscap - azure aad global administrator role assignment raw (ccx) - summary gen")`) | rename initiatedby.user.userprincipalname as user initiatedby.app.displayname as app initiatedby.user.ipaddress as src targetresources{}.* as target_* target_modifiedproperties{}.* as property_* | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | eval user=lower(user), signature=activitydisplayname | eval target_role=mvindex(target_displayname, mvfind(target_type, "role")), target_user=mvindex(target_userprincipalname, mvfind(target_type, "user")), target_directory=mvindex(target_displayname, mvfind(target_type, "directory")), target_templateid=mvindex(property_newvalue, mvfind(property_displayname, "templateid")), target_roledisplayname=mvindex(property_newvalue, mvfind(property_displayname, "displayname")), target_rolewellknownname=mvindex(property_newvalue, mvfind(property_displayname, "wellknownobjectname")) | fillnull user target_user value="null" | rename user as src_user | eval user=target_user | stats values(status) as status earliest(_time) as first_event_time latest(_time) as last_event_time values(dest) as dest values(_raw) as orig_raw values(vendor) as vendor values(vendor_product) as vendor_product values(signature) as signature values(app) as app values(src) as src values(target_directory) as target_directory values(target_role) as target_role values(target_roledisplayname) as target_roledisplayname values(target_rolewellknownname) as target_rolewellknownname values(target_templateid) as target_templateid values(action) as action values(host) as orig_host values(sourcetype) as orig_sourcetype values(index) as orig_index values(src_user) as src_user values(target_user) as target_user by user ccx_customer_zone | foreach * [| eval "<<field>>"=mvfilter(not match(<<field>>, "null"))] | fillnull vendor vendor_product src dest user src_user app target_directory target_role target_roledisplayname target_rolewellknownname value="missing" | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="defense evasion|persistence|privilege escalation|initial access", metadata_techniques="t1078.004", metadata_attack_type="azuread|iaas", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft azure", metadata_index_macros="cim_change_indexes", metadata_cim_datamodels=null(), metadata_event_codes=null()| `finalise_micro_search("mscap - azure aad global administrator role assignment raw (ccx) - summary gen", "ccx_customer_zone,user")` | `ccx_kill_switch`	// Title: MSCAP - Azure AAD Global Administrator Role Assignment RAW (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: // Date: 12/June/2024 config case_sensitive = false | dataset = msft_azure_ad_audit_raw | filter activityDisplayName = "*member to role*" | filter result = "success" | alter modified_properties = json_extract_array(targetResources, "$.0.modifiedProperties") \| alter target_role_display_name = arrayindex(arraymap(modified_properties, if("@element" -> displayName = "Role.DisplayName", "@element" -> newValue)), 0) \| alter target_template_id = arrayindex(arraymap(modified_properties, if("@element" -> displayName = "Role.TemplateId", "@element" -> newValue)), 0) \| filter target_role_display_name contains "global administrator" or target_template_id contains "62e90394-69f5-4237-9190-012177145e10" \| alter src_user = json_extract_scalar(initiatedBy, "$.user.userPrincipalName") | alter src = json_extract_scalar(initiatedBy, "$.user.ipAddress") \| alter app = json_extract_scalar(initiatedBy, "$.app.displayName") | alter target_user = json_extract_scalar(targetResources, "$.0.userPrincipalName") \| alter target_display_name = json_extract_scalar(targetResources, "$.0.displayName") | alter target_role_wellknown_name = arrayindex(arraymap(modified_properties, if("@element" -> displayName = "Role.WellKnownObjectName", "@element" -> newValue)), 0) | replacenull src_user = "missing", src = "missing", app = "missing", target_role_display_name = "missing", target_template_id = "missing", target_role_wellknown_name = "missing" | fields activityDateTime, activityDisplayName, result, category, operationType, app, src_user, target_user, target_role_display_name, target_template_id, target_display_name, target_role_wellknown_name, modified_properties, _vendor, _product, * | comp count() as total_events, values(activityDateTime) as first_event_time, values(activityDateTime) as last_event_time, values(_vendor) as vendor, values(_product) as product, values(src_user) as src_user, values(src) as src, values(app) as app, values(target_role_display_name) as target_role_display_name, values(target_template_id) as target_template_id, values(target_role_wellknown_name) as target_role_wellknown_name by target_user, activityDisplayName, result, operationType
search `cim_change_indexes` sourcetype=azure:aad:audit activitydisplayname="*set federation settings on domain*" result=success (`micro_search_global_filtering_list("mscap - azure modified domain federation trust settings raw (ccx) - summary gen")`) | rename initiatedby.user.userprincipalname as user targetresources{}.displayname as app | eval user=lower(user), signature=activitydisplayname | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | rename host as orig_host sourcetype as orig_sourcetype index as orig_index | rename _raw as orig_raw | table status orig_raw app user signature action orig_host orig_sourcetype orig_index src dest vendor vendor_product ccx_customer_zone _time | rename _time as event_time | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="defense evasion|privilege escalation|credential access", metadata_techniques="t1134.003", metadata_attack_type="azuread|network|windows|o365|iaas", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft azure|office 365|azure active directory", metadata_index_macros="cim_change_indexes", metadata_cim_datamodels=null(), metadata_event_codes=null()| `finalise_micro_search("mscap - azure modified domain federation trust settings raw (ccx) - summary gen", "*")` | `ccx_kill_switch`	// Title: MSCAP - Azure Modified Domain Federation Trust Settings RAW (CCX) - Summary Gen // Author: Deven Amode, damode@paloaltonetworks.com // Datasets: msft_azure_ad_audit_raw // Date: 11/June/2024 config case_sensitive = false | dataset = msft_azure_ad_audit_raw | filter activityDisplayName = "*set federation settings on domain*" and result = "success" | alter user = json_extract_scalar(initiatedBy, "$.user.userPrincipalName"), src = json_extract_scalar(initiatedBy, "$.user.ipAddress"), app = json_extract_scalar(targetResources, "$.0.displayName"), user_type = json_extract_scalar(targetResources, "$.0.type") | fields _time, app, user, user_type, activityDisplayName as signature, _collector_type , _vendor, _product

search `cim_event_signatures_indexes` eventcode=4769 servicename != "*$" (ticketoptions=0x40810000 or ticketoptions=0x40800000 or ticketoptions=0x40810010) ticketencryptiontype=0x17 (`micro_search_global_filtering_list("mscap - kerberoasting spn request against excessive accounts raw (ccx) - summary gen")`) | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product user dest service service_id value="undefined" | stats values(_raw) as orig_raw values(src) as src values(dest) as dest values (service) as service dc(service) as service_count values(service_id) as service_id values (ticketencryptiontype) as ticketencryptiontype values(ticketoptions) as ticketoptions values(eventcode) as eventcode values(signature_id) as signature_id values(action) as action values(vendor) as vendor values(vendor_product) as vendor_product values(sourcetype) as orig_sourcetype values(host) as orig_host values(index) as orig_index min(_time) as first_event_time max(_time) as last_event_time by user ccx_customer_zone | search service_count >= 10 | eval metadata_cis20="cis 8|cis 16", metadata_killchainstage="exploitation", metadata_tactics="credential access", metadata_techniques="t1558|t1558.003", metadata_attack_type=null(), metadata_nist="de.cm", metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft windows", metadata_index_macros="cim_event_signatures_indexes", metadata_cim_datamodels=null(), metadata_event_codes="4769"| `finalise_micro_search("mscap - kerberoasting spn request against excessive accounts raw (ccx) - summary gen", "ccx_customer_zone,user")`| `ccx_kill_switch`	// Title: MSCAP - Kerberoasting SPN request against Excessive Accounts RAW (CCX) - Summary Gen // Author: Deven Amode, damode@paloaltonetworks.com // Date: 13/June/2024 config case_sensitive = false | datamodel dataset = microsoft_windows_raw | alter service_name = json_extract_scalar(microsoft_windows_raw.event_data, "$.ServiceName"), service_id = json_extract_scalar(microsoft_windows_raw.event_data, "$.ServiceSid"), ticket_options = json_extract_scalar(microsoft_windows_raw.event_data, "$.TicketOptions"), ticket_encryption_type = json_extract_scalar(microsoft_windows_raw.event_data, "$.TicketEncryptionType") | filter xdm.event.id = "4769" and service_name != "*$" and ticket_options in ("0x40810000","0x40800000","0x40810010") and ticket_encryption_type="0x17" | fields *, xdm.source.user.username as user,microsoft_windows_raw._raw_log | comp values(microsoft_windows_raw._raw_log) as orig_raw, values(xdm.source.ipv4) as src, values(xdm.target.ipv4) as dest ,values(service_name) as service_name, count_distinct(service_name) as service_count, values(service_id) as service_id, values(ticket_encryption_type) as ticket_encryption_type, values(ticket_options) as ticket_options, values (xdm.event.id) as eventcode, values(_vendor) as vendor , values(_product) as vendor_product , values(xdm.source.host.hostname) as orig_host, values(xdm.observer.type)as orig_sourcetype, earliest(_time) as first_event_time, latest (_time) as last_event_time by user
search `cim_event_signatures_indexes` eventtype=wineventlog_security eventcode=4776 user != "*$" status=0xc000006a action=failure (`micro_search_global_filtering_list("mscap - multiple users failing to authenticate from host using ntlm raw (ccx) - summary gen")`) | bucket span=2m _time as bucket_time | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product source_workstation value="missing" | stats values(_raw) as orig_raw dc(user) as unique_accounts values(eventcode) as eventcode values(signature_id) as signature_id values(user) as user values (host) as orig_host values(sourcetype) as orig_sourcetype values(index) as orig_index values(src) as src values(dest) as dest values(vendor) as vendor values(vendor_product) as vendor_product earliest (_time) as first_event_time latest(_time) as last_event_time by bucket_time source_workstation ccx_customer_zone | eventstats avg(unique_accounts) as comp_avg stdev(unique_accounts) as comp_std by source_workstation ccx_customer_zone | eval upperbound=(comp_avg + comp_std * 3) | eval isoutlier=if(unique_accounts > 10 and unique_accounts >= upperbound, 1, 0) | search isoutlier=1 | eval metadata_cis20=null(), metadata_killchainstage="exploitation", metadata_tactics="credential access", metadata_techniques="t1110.003", metadata_attack_type="windows", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft windows", metadata_index_macros="cim_event_signatures_indexes", metadata_cim_datamodels=null(), metadata_event_codes="4776"| `finalise_micro_search("mscap - multiple users failing to authenticate from host using ntlm raw (ccx) - summary gen", "ccx_customer_zone,source_workstation, bucket_time")`| `ccx_kill_switch`	// Title: MSCAP - Multiple Users Failing To Authenticate From Host Using NTLM RAW (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: // Date: 13/June/2024 config case_sensitive = false | datamodel dataset = microsoft_windows_raw | filter xdm.source.host.os_family = XDM_CONST.OS_FAMILY_WINDOWS | filter xdm.event.id = "4776" | filter xdm.event.outcome = "0xc000006a" | alter workstation = json_extract_scalar(microsoft_windows_raw.event_data, "$.Workstation") | replacenull workstation = "missing" | comp count() as total_events, count_distinct(xdm.target.user.username) as unique_accounts, values(xdm.source.host.hostname) as host, values(xdm.target.user.username) as user, earliest(_time) as first_event_time, latest(_time) as last_event_time, values(xdm.event.original_event_type) as event_type, values(xdm.observer.vendor) as vendor, values (xdm.observer.product) as product by workstation, xdm.event.id, xdm.event.outcome | filter unique_accounts > 10 // filtering for more than 10 unique user accounts

search `cim_event_signatures_indexes` eventtype=wineventlog_security eventcode=4625 logon_type=3 src != "-" (`micro_search_global_filtering_list("mscap - multiple users remotely failing to authenticate from host raw (ccx) - summary gen")`) | bucket span=2m _time as bucket_time | eval destination_account=user | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product workstationname value="missing" | stats values(_raw) as orig_raw dc(destination_account) as unique_accounts values(eventcode) as eventcode values(signature_id) as signature_id values(user) as user values(host) as orig_host values(sourcetype) as orig_sourcetype values(index) as orig_index values(dest) as dest values(vendor) as vendor values(vendor_product) as vendor_product earliest (_time) as first_event_time latest(_time) as last_event_time by bucket_time src workstationname ccx_customer_zone | eventstats avg(unique_accounts) as comp_avg stdev(unique_accounts) as comp_std by src workstationname ccx_customer_zone | eval upperbound=(comp_avg + comp_std * 3) | eval isoutlier=if(unique_accounts > 10 and unique_accounts >= upperbound, 1, 0) | search isoutlier=1 | eval metadata_cis20=null(), metadata_killchainstage="exploitation", metadata_tactics="credential access", metadata_techniques="t1110.003", metadata_attack_type="windows", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft windows", metadata_index_macros="cim_event_signatures_indexes", metadata_cim_datamodels=null(), metadata_event_codes="4625"| `finalise_micro_search("mscap - multiple users remotely failing to authenticate from host raw (ccx) - summary gen", "ccx_customer_zone,src,bucket_time, workstationname")`| `ccx_kill_switch`	// Title: MSCAP - Multiple Users Remotely Failing To Authenticate From Host RAW (CCX) - Summary Gen // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: microsoft_windows_raw // Date: 13/June/2024 config case_sensitive = false | datamodel dataset = microsoft_windows_raw | filter xdm.source.host.os_family = XDM_CONST.OS_FAMILY_WINDOWS | filter xdm.event.id = "4625" | filter xdm.logon.type = "NETWORK" | filter xdm.source.user.username != "-" | alter workstation_name = json_extract_scalar(microsoft_windows_raw.event_data, "$.WorkstationName") | replacenull workstation_name = "missing" | comp count() as total_events, count_distinct(xdm.target.user.username) as unique_accounts, values(xdm.source.host.hostname) as host, values(xdm.target.user.username) as target_user, earliest(_time) as first_event_time, latest(_time) as last_event_time, values(xdm.event.original_event_type) as event_type, values(xdm.observer.vendor) as vendor, values(xdm.observer.product) as product by workstation_name, xdm.source.user.username, xdm.event.id | filter unique_accounts > 10 // filtering for more than 10 unique user accounts
search `cim_event_signatures_indexes` eventcode=4673 service="lsaregisterlogonprocess()" keywords="0x8010000000000000" not subjectusersid in ("nt authority\\network service", "nt service\\msolap$datainsight") (`micro_search_global_filtering_list("mscap - user couldn't call a privileged service lsaregisterlogonprocess raw (ccx) - summary gen")`) | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product value="missing" | rename host as orig_host sourcetype as orig_sourcetype index as orig_index | fillnull dest value="missing" | stats values(_raw) as orig_raw values(vendor) as vendor values(signature_id) as signature_id values(vendor_product) as vendor_product values(orig_host) as orig_host values(orig_sourcetype) as orig_sourcetype values(orig_index) as orig_index values(src) as src values(user) as user earliest (_time) as first_event_time latest(_time) as last_event_time by dest eventcode service keywords ccx_customer_zone | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics="lateral movement|privilege escalation", metadata_techniques="t1558.003", metadata_attack_type=null(), metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft windows", metadata_index_macros="cim_event_signatures_indexes", metadata_cim_datamodels=null(), metadata_event_codes="4673"| `finalise_micro_search("mscap - user couldn't call a privileged service lsaregisterlogonprocess raw (ccx) - summary gen", "ccx_customer_zone,dest,eventcode,service,keywords")`| `ccx_kill_switch`	// Title: MSCAP - User Couldn't Call a Privileged Service LsaRegisterLogonProcess RAW (CCX) - Summary Gen // Author: Deven Amode, damode@paloaltonetworks.com // Date: 13/June/2024 config case_sensitive = false | datamodel dataset = microsoft_windows_raw | alter service = json_extract_scalar(microsoft_windows_raw.event_data, "$.service"), subjectusersid = json_extract_scalar(microsoft_windows_raw.event_data, "$.subjectusersid") | filter xdm.event.id = "4673" and service="lsaregisterlogonprocess()" and microsoft_windows_raw.keywords = "0x8010000000000000" and subjectusersid not in ("nt authority\\network service", "nt service\\msolap$datainsight") \|fields _time,_vendor as vendor,_product as vendor_product, microsoft_windows_raw._raw_log as orig_raw, xdm.source.host.hostname as orig_host, xdm.target.ipv4 as dest,xdm.observer.type as orig_sourcetype,xdm.source.ipv4 as src, xdm.source.user.username as user, microsoft_windows_raw.keywords as keywords,service, xdm.event.id as eventcode \| comp values(orig_raw) , values(vendor) , values(vendor_product) , values(orig_host) , values(orig_sourcetype), values(src), values(user), earliest(_time) as first_event_time, latest(_time) as last_event_time by dest, eventcode, service, keywords // Author: Deven Amode, damode@paloaltonetworks.com // Date: 13/June/2024 config case_sensitive = false \| datamodel dataset = microsoft_windows_raw \| alter service_name = json_extract_scalar(microsoft_windows_raw.event_data, "$.ServiceName"), service_id = json_extract_scalar(microsoft_windows_raw.event_data, "$.ServiceSid"), ticket_options = json_extract_scalar(microsoft_windows_raw.event_data, "$.TicketOptions"), ticket_encryption_type = json_extract_scalar(microsoft_windows_raw.event_data, "$.TicketEncryptionType") \| filter xdm.event.id = "4769" and service_name != "*$" and ticket_options in ("0x40810000","0x40800000","0x40810010") and ticket_encryption_type="0x17" | fields *, xdm.source.user.username as user,microsoft_windows_raw._raw_log | comp values(microsoft_windows_raw._raw_log) as orig_raw, values(xdm.source.ipv4) as src, values(xdm.target.ipv4) as dest ,values(service_name) as service_name, count_distinct(service_name) as service_count, values(service_id) as service_id, values(ticket_encryption_type) as ticket_encryption_type, values(ticket_options) as ticket_options, values(xdm.event.id) as eventcode, values(_vendor) as vendor , values(_product) as vendor_product , values(xdm.source.host.hostname) as orig_host, values(xdm.observer.type)as orig_sourcetype, earliest(_time) as first_event_time, latest(_time) as last_event_time by user

search `cim_event_signatures_indexes` eventtype=wineventlog_security eventcode=4648 (`micro_search_global_filtering_list("mscap - users attempting to auth using explicit credentials raw (ccx) - summary gen")`) | bucket span=2m _time as bucket_time | eval source_account=src_user | eval destination_account=user | search source_account != "*$" source_account != "-" destination_account != "*$" | lookup index_to_ccx_customer_zone_lookup index_match as index output ccx_customer_zone | fillnull ccx_customer_zone value="undefined" | fillnull vendor vendor_product computer processname value="missing" | stats values(_raw) as orig_raw dc (destination_account) as unique_accounts values(eventcode) as eventcode values(signature_id) as signature_id values(user) as user values(host) as orig_host values(sourcetype) as orig_sourcetype values(index) as orig_index values(src) as src values(dest) as dest values(processname) as processname values(vendor) as vendor values(vendor_product) as vendor_product earliest(_time) as first_event_time latest(_time) as last_event_time by bucket_time computer source_account ccx_customer_zone | eventstats avg(unique_accounts) as comp_avg stdev(unique_accounts) as comp_std by computer ccx_customer_zone | eval upperbound=(comp_avg + comp_std * 3) | eval isoutlier=if(unique_accounts > 10 and unique_accounts >= upperbound, 1, 0) | search isoutlier=1 | eval metadata_cis20=null(), metadata_killchainstage="exploitation", metadata_tactics="credential access", metadata_techniques="t1110.003", metadata_attack_type="windows", metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="microsoft", metadata_vendor_products="microsoft windows", metadata_index_macros="cim_event_signatures_indexes", metadata_cim_datamodels=null(), metadata_event_codes="4648"| `finalise_micro_search("mscap - users attempting to auth using explicit credentials raw (ccx) - summary gen", "ccx_customer_zone,bucket_time,computer,source_account")`| `ccx_kill_switch`	// Title: MSCAP - Users Attempting To Auth Using Explicit Credentials RAW (CCX) - Summary Gen /// Author: Deven Amode, damode@paloaltonetworks.com // Datasets: microsoft_windows_raw // Date: 07/June/2024 datamodel dataset = microsoft_windows_raw | filter xdm.event.id ="4648" | replacenull xdm.source.ipv4 = "-" | comp values(microsoft_windows_raw._raw_log) as orig_raw, min(_time) as first_event_time, max(_time) as last_event_time, values(xdm.event.id) as eventcode, values(xdm.source.ipv4) as src , values(xdm.source.host.hostname) as orig_host , values(xdm.observer.type) as orig_sourcetype , values(xdm.source.user.username) as user by _vendor, _product
index=* sourcetype="tenable:io:vuln" plugin.name="*silverlight*" state=open OR state=reopened asset_fqdn="*" | rename plugin.name AS pluginName | fillnull value=NULL cvss3_base_score | eval pluginAndCVEState = pluginName." CVSS2: ".cvss2_base_score." CVSS3: ".cvss3_base_score." ".state | stats values(pluginAndCVEState) by asset_fqdn	// Title: Silverlight Fortnightly report // Description: Report for Joanna Field to review // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: tenable_io_vulnerabilities_raw // Date: 25/June/2024 config case_sensitive = false | datamodel dataset = tenable_io_vulnerabilities_raw | filter xdm.alert.name = "*silverlight*" | filter xdm.source.host.fqdn not in ("", null) | alter state = tenable_io_vulnerabilities_raw.state, cvss3_base_score = json_extract_scalar(tenable_io_vulnerabilities_raw.plugin, "$.cvss_base_score") | filter state in ("OPEN", "REOPENED") | replacenull cvss3_base_score = "NULL" | alter pluginAndCVEState = format_string("%s CVSS3: %s %s", xdm.alert.name, cvss3_base_score, state) | fields tenable_io_vulnerabilities_raw.plugin, xdm.alert.name, xdm.source.host.fqdn, state, cvss3_base_score, pluginAndCVEState, * | comp count() as total_events, min(_time) as s_time, max(_time) as e_time, values(xdm.alert.severity) as severity, values(xdm.event.description) as output, values(xdm.alert.description) as description, values(pluginAndCVEState) as pluginAndCVEState, values(xdm.alert.name) as plugin_name, values(cvss3_base_score) as cvss3_score, values(state) as state by xdm.source.host.fqdn | fields pluginAndCVEState, plugin_name, cvss3_score, state, xdm.source.host.fqdn as asset_fqdn, total_events, s_time, e_time, severity, output, description

index=* sourcetype="ccx:mimecastauditst" auditType IN ("*Adjustments" "*Adjustment" "*Log Entry") | fillnull value=NULL | stats count by _time user action app auditType change_type object object_attrs object_category signature reason method	// Title: [OTML] Mimecast Auditing Events CHG0034292 // Description: Report to OTML from audit policy changes in MimeCast sent to ICT-Enterprise-Systems@oktedi.com // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Date: 25/June/2024 config case_sensitive = false | datamodel dataset = mimecast_mimecast_raw | filter xdm.event.type in ("*Adjustments", "*Adjustment", "*Log Entry") | alter //object_attrs = auditType , object =to_string(regextract(mimecast_mimecast_raw.eventInfo,"(?:Application|App):\s*([^,]+)")), object_category = if(xdm.event.type contains "*user*","user", "instance"), change_type = if(xdm.event.type contains "*user*","AAA", "filesystem"), signature = if (xdm.event.type contains "*log*" , xdm.event.type, "virus"), action = mimecast_mimecast_raw.action, reason = xdm.event.description // Used both reason and eventInfo data to evaluate Action | alter action = if(action not in ("", null), action, reason contains "Wrong Password" or reason contains "Account locked", "Failure", reason contains "White URL created", "Created", reason contains "Successful", "Success", mimecast_mimecast_raw.eventInfo contains "unlocked", "Unlocked", mimecast_mimecast_raw.eventInfo contains "locked", "Lockout", mimecast_mimecast_raw.eventInfo contains "updated", "Updated", mimecast_mimecast_raw.eventInfo contains "delete", "Deleted", mimecast_mimecast_raw.eventInfo contains "cleare", "Cleared", mimecast_mimecast_raw.eventInfo contains "create", "Created", mimecast_mimecast_raw.eventInfo not in ("", null), "Modified", null) |fields mimecast_mimecast_raw.eventInfo , _time, xdm.source.user.username, object as app, xdm.event.type, change_type, object_category, reason, action | comp count() as total_events, min(_time) as firstEventTime , max(_time) as lastEventTime by xdm.source.user.username, app, reason, action, xdm.event.type, change_type, object_category
`cim_Endpoint_indexes` sourcetype=XmlWinEventLog EventID=4728 Group_Name="*admin*" MemberName="*logicmonitor*"| stats count by EventID subject MemberName Computer Group_Domain Group_Name SubjectDomainName SubjectUserName TargetDomainName TargetUserName | lookup index_to_ccx_customer_zone_lookup index_match AS index OUTPUT ccx_customer_zone | fillnull ccx_customer_zone value="UNDEFINED" | `finalise_micro_search("MSANA - Service Logic Monitor Added to Admin Group (CS0057691) (CCX) - Summary Gen","EventID, subject, MemberName, Computer, Group_Domain, Group_Name, SubjectDomainName, SubjectUserName, TargetDomainName, TargetUserName, sourcetype, index, ccx_customer_zone")` | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics=null(), metadata_techniques=null(), metadata_attack_type=null(), metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor=null(), metadata_vendor_products=null(), metadata_index_macros=null(), metadata_cim_datamodels=null(), metadata_event_codes=null()	// Title: MSANA - Service Logic Monitor Added to Admin Group (CS0057691) (CCX) - Summary Gen // Description: report when logic Monitor is added to an administrator work, as Logic Monitor should not have this capability // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: microsoft_windows_raw // Date: 25/June/2024 config case_sensitive = false | datamodel dataset = microsoft_windows_raw | filter xdm.source.host.os_family = XDM_CONST.OS_FAMILY_WINDOWS | filter xdm.event.id = "4728" | filter xdm.target.user.username = "*admin*" | alter member_name = json_extract_scalar(microsoft_windows_raw.event_data, "$.MemberName") \| filter member_name = "*logicmonitor*" // extract user from DN \| alter membername = replace(json_extract_scalar(microsoft_windows_raw.event_data, "$.MemberName"), "\,", " ") | alter user = arrayindex(regextract(membername, "CN=([^,]+)"), 0) | comp count() as total_event, min(_time) as first_event_time, max(_time) as last_event_time, values(member_name) as member_name, values(user) as user by xdm.event.id, xdm.event.original_event_type, xdm.source.user.username, xdm.source.user.domain, xdm.target.user.username, xdm.target.user.domain, xdm.source.host.hostname

`cim_Intrusion_Detection_indexes` sourcetype="canarytools:incident" summary IN ("RDP Login Attempt", "WinRM Login Attempt", "VNC Login Attempt") "description.acknowledged"=False (`micro_search_global_filtering_list("MSCAP - Thinkst Canary Incident Triggered - Graphical Login RAW (CCX) - Summary Gen")`) (`cap_filter_vms_scanners("src")`) | eval user=coalesce(user, 'description.events{}.USERNAME') | fillnull ccx_customer_zone value="UNDEFINED" | fillnull vendor vendor_product user summary src dest value="MISSING" | stats values(_raw) AS orig_raw values (index) AS orig_index values(transport) AS transport values(sourcetype) AS orig_sourcetype values (host) AS orig_host values(vendor) AS vendor values(vendor_product) AS vendor_product count earliest(_time) AS first_event_time latest(_time) AS last_event_time BY summary user src dest ccx_customer_zone | eval metadata_cis20=null(), metadata_killchainstage=null(), metadata_tactics=null(), metadata_techniques=null(), metadata_attack_type=null(), metadata_nist=null(), metadata_cve=null(), metadata_detectframework="springsteen", metadata_vendor="Thinkst", metadata_vendor_products="Thinkst Canary", metadata_index_macros="cim_Intrusion_Detection_indexes", metadata_cim_datamodels=null(), metadata_event_codes=null()| `finalise_micro_search("MSCAP - Thinkst Canary Incident Triggered - Graphical Login RAW (CCX) - Summary Gen", "ccx_customer_zone,summary,user,src,dest")`| `ccx_kill_switch`	// Title: MSCAP - Thinkst Canary Incident Triggered - Graphical Login RAW (CCX) - Summary Gen // Description: Alerts on Thinkst Canary incidents related to graphical logins // Author: Sahil Sharma, ssharma7@paloaltonetworks.com // Datasets: microsoft_windows_raw // Date: 26/June/2024 config case_sensitive = false | dataset = thinkst_canary_generic_alert_raw | filter summary in ("RDP Login Attempt", "WinRM Login Attempt", "VNC Login Attempt") | alter src_ip = description -> src_host, src_port = description -> src_port, dst_ip = description -> dst_host, dst_port = description -> dst_port, user = json_extract_scalar(description, "$.events.0.USERNAME"), name = description -> name, source_type = _alert_data -> alert_source, alert_type = _alert_data -> alert_type, host = description -> src_host_reverse | replacenull src_ip = "MISSING", dst_ip = "MISSING", user = "MISSING" | filter src_ip not in ("bvmswhp01", "10.10.144.10", "btenwh01", "bvmsppp01", "10.40.144.12", "btenpp01", "bvmsndp01", "10.50.144.10", "btenb201", "bnesswh01", "10.10.143.21", "btenwh02", "btenmi01", "10.25.144.10") | comp count() as total_events, earliest(_time) as first_event_time, latest(_time) as last_event_time, values(name) as name, values(source_type) as orig_sourcetype, values(alert_type) as alert_type, values(host) as orig_host, values (_vendor) as vendor, values(_product) as product, values(src_port) as src_port, values(dst_port) as dst_port by summary, src_ip, dst_ip, user
index=vpn AND Cisco_ASA_message_id=734003 AND (endpoint_attribute_name="aaa.cisco. tunnelgroup") NOT user IN (V-P631547, V-P631544, V-P631546, V-P631545, V-P631543) AND endpoint_value="SS_Retailer" | rename endpoint_value AS TunnelGroup, dest AS src_ip | fields _time user TunnelGroup src_ip host | join type=inner user host [search index=vpn AND Cisco_ASA_message_id=722022] | table _time user user_nick host src_hostname TunnelGroup src_ip VPNToken	datamodel dataset = cisco_asa_vpn_raw | filter (xdm.event.id = "734003" and cisco_asa_vpn_raw ._ raw_log ~= "Session\sAttribute\saaa\.cisco\.tunnelgroup") or (xdm.event.id = "722022") and xdm.source.user.username not in ("V-P631547", "V-P631544", "V-P631546", "V-P631545", "V-P631543") and xdm.source.user.groups = "SS_Retailer" | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = xdm. source.user.username |comp values(arraystring(xdm.source.user.groups,",")) as Tunnel_Groups, count() as Count1 by xdm.source.user. username, xdm.source.ipv4, xdm. source.host.device_category
(index=vpn AND message_id=734003 AND (endpoint_attribute_name="endpoint.device.hostname")) AND user="G-PRDOFFASSESS1" | iplocation dest | eval src_hostname=mvindex(src_dns,0) | search src_hostname!="st-l1006997" | table _time, user, user_nick, user_email, user_managedBy, src_hostname , dest, Country	datamodel dataset = cisco_asa_vpn_raw | filter xdm.event.id = "734003" and cisco_asa_vpn_raw._raw_log contains "endpoint.device.hostname" and xdm.source.user.username = "G-PRDOFFASSESS1" and xdm.source.host.hostname != "st-l1006997" | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = xdm.source.user.username | *
(index=zscalerlogs OR index=proxy) dest IN ("mitarchive.info", "www.dealctr.com", "liveupdt.com") | table _time action user hostname dest url http_referrer http_method http_user_agent | sort -_time	datamodel dataset in (zscaler_nss_raw , symantec_bluecoatproxysg_raw) | filter xdm.target.url contains "liveupdat.com" or xdm.target.url contains "www.dealctr.com" or xdm.target.url contains "liveupdt.com" | fields _time, xdm.observer.action, xdm.source.host.hostname, xdm.source.user.username, xdm.network.http.domain, xdm.target.url, xdm.network.http.referrer, xdm.network.http.method, xdm.source.user_agent

index=vpn (src!="" AND src!=10.*.*.* AND src!=203.126.130.140 AND src!=165.21.21.38 AND src!=203.127.23.227 AND src!=203.125.232.138 AND src!=203.208.173.13) Cisco_ASA_message_id=722051 | iplocation src | fillnull value="Unknown" | stats count(src) AS ipCount BY user, user_nick, user_email, user_managedBy, src, _time, Country | where ipCount >=1 | table _time, user, user_nick, user_email, user_managedBy ,src, Country | eval _time=strftime(_time, "%Y-%d-%m %H:%M") | dedup src, user | stats values(_time) values(src) values(user_nick) values(user_email) values(user_managedBy) values(Country), dc(Country) AS numCountries BY user | rename values(_time) AS _time values(src) AS src_ip values(Country) AS Country user AS src_user values(user_nick) AS src_user_nick values(user_email) AS src_user_email values(user_managedBy) AS src_user_managedBy | where numCountries>1 | table _time, src_user, src_user_nick src_user_email src_user_managedBy src_ip, Country	datamodel dataset = cisco_asa_vpn_raw | filter xdm.event.id = "722051" and not incidr(xdm.source.ipv4, "10.0.0.0/8") and xdm.source.ipv4 not in ("203.126.130.140", "165.21.21.38", "203.127.23.227", "203.125.232.138", "203.208.173.13") | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = xdm.source.user.username | fields * | comp count(xdm.source.ipv4) as ip_count by xdm.source.user.username, xdm.source.ipv4, _time, xdm.source.location.country, upn, manager_sam_account_name | filter ip_count >= 1 | comp count_distinct(xdm.source.location.country) as num_of_countries , values(_time) as time, values(xdm.source.ipv4) as src_ip, values(upn) as user_upn, values(manager_sam_account_name) as user_sam_account_name, values(xdm.source.location.country) as country by xdm.source.user.username //,min(_time) as first_seen_time, max(_time) as last_seen_time | filter num_of_countries > 1
index=vpn AND (src_dns="innotek GmbH VirtualBox" OR src_dns="VMWare*" OR src_dns="Microsoft Corporation Virtual Machine" OR src_dns="Parallels*") | iplocation dest_ip | table _time, user, user_nick, user_email, user_managedBy, user_bunit src_dns, dest_ip, Country | rename user as src_user user_nick as src_user_nick user_email as src_user_email user_managedBy as src_user_managedBy user_bunit as src_user_bunit	datamodel dataset = cisco_asa_vpn_raw | filter cisco_asa_vpn_raw._raw_log contains "734003" and cisco_asa_vpn_raw._raw_log ~= "Session\sAttribute\sendpoint" //| alter xdm.source.host.device_category = arrayindex(regextract(xdm.event.description , "endpoint.anyconnect.devicetype\s*=\s*\"(.+)\""),0) | filter xdm.source.host.device_category in ("innotek GmbH VirtualBox","VMWare*","Microsoft Corporation Virtual Machine","Parallels*") | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = xdm.source.user.username | fields *
index=vpn Cisco_ASA_message_id=111010 src!=0.0.0.0 src!=10.* | table _time user user_first user_last user_managedBy object src Cisco_ASA_vendor_action command dvc	config case_sensitive = false | datamodel dataset = cisco_asa_vpn_raw | alter object = arrayindex(regextract(cisco_asa_vpn_raw._raw_log, "running\s+\'(.*)\'\s+from"),0) | filter xdm.event.id = "111010" and xdm.source.ipv4 not in ("0.0.0.0","10.*") | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = xdm.source.user.username | fields *
index=vpn Cisco_ASA_message_id=111010 command="access-list *" OR command="no access-list *" AND src!=0.0.0.0 | table _time user user_first user_last user_managedBy object src Cisco_ASA_vendor_action command dvc	config case_sensitive = false | datamodel dataset = cisco_asa_vpn_raw | alter object = arrayindex(regextract(cisco_asa_vpn_raw._raw_log, "running\s+\'(.*)\'\s+from"),0) | filter xdm.event.id = "111010" and xdm.source.ipv4 not in ("0.0.0.0") and xdm.target.process.command_line in ("access-list*","no access-list*") | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = xdm.source.user.username | fields *
index=external_firewall AND action=allowed | join type=inner src_ip [| inputlookup IOC_FW.csv | table src_ip IOC] | table _time src_ip src_port dest_ip dest_port host action direction IOC	datamodel dataset = check_point_vpn_1_firewall_1_raw | filter xdm.observer.action = "Accept" | alter deviceDirection = if(arrayindex(regextract(check_point_vpn_1_firewall_1_raw._raw_log,"deviceDirection=([^\s=]+)"),0) = "0", "inbound", arrayindex(regextract(check_point_vpn_1_firewall_1_raw._raw_log,"deviceDirection=([^\s=]+)"),0) = "1", "outbound") | join (dataset = ioc_fw | fields src_ip, IOC) as ioc_fw ioc_fw.src_ip = xdm.source.ipv4 | fields xdm.source.ipv4, xdm.source.port, xdm.target.ipv4, xdm.target.port, xdm.source.host.hostname, IOC, deviceDirection
index=external_firewall AND action=allowed | join type=inner left=l right=r where l.dest_ip=r.src_ip [| inputlookup IOC_FW.csv| table src_ip IOC] | table _time l.src_ip l.src_port l.dest_ip l.dest_port l.host l.action l.direction r.IOC | rename l.src_ip as src_ip l.src_port as src_port l.dest_ip as dest_ip l.dest_port as dest_port l.host as host l.action as action l.direction as direction r.IOC as IOC	datamodel dataset = check_point_vpn_1_firewall_1_raw | filter xdm.observer.action = "Accept" | alter deviceDirection = if(arrayindex(regextract(check_point_vpn_1_firewall_1_raw._raw_log,"deviceDirection=([^\s=]+)"),0) = "0", "inbound", arrayindex(regextract(check_point_vpn_1_firewall_1_raw._raw_log,"deviceDirection=([^\s=]+)"),0) = "1", "outbound") | join (dataset = IOC_FW | fields src_ip IOC) as ioc_fw ioc_fw.src_ip = xdm.target.ipv4 | fields xdm.source.ipv4, xdm.source.port, xdm.target.ipv4, xdm.target.port, xdm.source.host.hostname, IOC, deviceDirection

index=*firewall NOT [|inputlookup Exclude_DDOS.csv | table src_ip] action=allowed AND src_ip!=10.65.0.0/16 AND (src_ip!=165.21.21.250) | stats dc(dest_port) as num_dest_port dc(dest) as num_dest_ip count by src_ip action | where (num_dest_port>100 OR num_dest_ip>500) AND count>25000	datamodel dataset in (check_point_vpn_1_firewall_1_raw , fortinet_fortigate_raw) | filter xdm.observer.action = "accept" | join (dataset = Exclude_DDOS | fields src_ip) as ddos ddos.src_ip != xdm.source.ipv4 or ddos.src_ip != xdm.source.ipv6 | filter not incidr(xdm.source.ipv4,"10.65.0.0/16") and xdm.source.ipv4 != "165.21.21.250" | comp count_distinct(xdm.target.port) as num_dest_port, count_distinct(xdm.target.ipv4) as num_dest_ip, count() as event_count by xdm.source.ipv4, xdm.observer.action | filter (num_dest_port > 100 or num_dest_ip > 500) and event_count > 25000
index=zscalerlogs sourcetype=zscalernss-web AND action=allowed [| inputlookup IOC_Proxy.csv | table URL | rename URL AS dest] | join dest [| inputlookup IOC_Proxy.csv | table URL IOC | rename URL AS dest] | table _time src_user src_user_managedBy src_user_bunit dest url category action status bytes_in bytes_out IOC	datamodel dataset = zscaler_nss_raw |filter xdm.observer.product = "nss" and xdm.observer.action = "allowed" |join (dataset = IOC_Proxy.csv | fields URL, IOC) as ioc_proxy url.ioc = xdm.target.url | fields _time, xdm.source.user.username as src_user, xdm.source.user.ou as src_user_bunit, xdm.target.host.hostname as dest, xdm.target.url as url, xdm.network.http.url_category as category, xdm.observer.action as action, xdm.network.http.response_code as status, xdm.target.sent_bytes as bytes_in, xdm.source.sent_bytes as bytes_out, ioc_proxy
index=zscalerlogs AND (urlclass = "Advanced Security Risk" OR urlclass = "Privacy Risk") AND action = Allowed | table _time action deviceowner user src_user_nick category url transport location bytes status src_ip serverip reason	datamodel dataset in (zscaler_nss_raw) | filter (xdm.alert.category = "Security Risk" or xdm.alert.category = "Privacy Risk") and xdm.observer.action = "Allowed" | fields _time, xdm.alert.category as urlclass, xdm.observer.action as action, xdm.source.user.employee_id as deviceowner, xdm.source.user.username as user, xdm.network.http.url_category as category, xdm.target.url as url, xdm.network.application_protocol as transport, xdm.source.location.region as location, xdm.target.sent_bytes as received_bytes, xdm.source.sent_bytes as sent_bytes, xdm.network.http.response_code as status, xdm.source.host.ipv4_addresses as src_ip, xdm.target.ipv4 as serverip, xdm.event.outcome_reason as reason, zscaler_nss_raw._raw_log
index=zscalerlogs (category="Botnet Callback") | table _time action dest url threatname urlclass deviceowner user devicehostname url useragent location ClientIP clientpublicIP	datamodel dataset = zscaler_fw_raw |filter xdm.network.http.url_category = "Botnet Callback" |fields _time, xdm.event.outcome as action, xdm.target.url as url, xdm.alert.original_threat_name as threatname, xdm.alert.category as urlclass, xdm.source.user.employee_id as deviceowner, xdm.source.user.username as user, xdm.source.host.hostname as devicehostname, xdm.source.user_agent as useragent, xdm.source.location.region as location, xdm.source.host.ipv4_addresses as clientIP, xdm.source.host.ipv4_public_addresses as clientpublicIP
index=zscalerlogs threatclass="Behavior Analysis" (threatcategory!="None" AND threatcategory!="Sandbox Benign" AND threatcategory!="Submitted to Sandbox" AND threatcategory!="Sandbox Suspicious") action=allowed | table _time action dest url http_referrer threatcategory threatname urlclass deviceowner user src_user_bunit devicehostname url useragent location ClientIP clientpublicIP	datamodel dataset = zscaler_nss_raw |filter xdm.observer.product = "nss" |alter threatclass = arrayindex(regextract(zscaler_nss_raw._raw_log, "threatclass=(.+)\tdlpdictionaries"), 0) |filter threatclass = "Behavior Analysis" |filter xdm.alert.subcategory != "None" and xdm.alert.subcategory != "Sandbox Benign" and xdm.alert.subcategory != "Submitted to Sandbox" and xdm.alert.subcategory != "Sandbox Suspicious" and xdm.observer.action = "allowed" | fields _time, xdm.observer.action as action, xdm.target.host.hostname as dest, xdm.target.url as url, xdm.network.http.referrer as http_referrer, xdm.alert.subcategory as threatcategory, xdm.alert.original_threat_name as threatname, xdm.alert.category as urlclass, xdm.source.user.employee_id as deviceowner, xdm.source.user.username as user, xdm.source.user.ou as src_user_bunit, xdm.source.host.hostname as devicehostname, xdm.source.user_agent as useragent, xdm.source.location.region as location, xdm.source.host.ipv4_addresses as clientIP, xdm.source.host.ipv4_public_addresses as clientpublicIP
index=*_windows AND EventCode=4769 AND Failure_Code=0x1f | eval IPAddress = replace (src_ip,"::ffff:","") | stats count by IPAddress | sort -count | where count > 15 | rename IPAddress as src_ip	datamodel dataset = microsoft_windows_raw | alter Failure_Code = json_extract_scalar(microsoft_windows_raw.event_data ,"$.Status") | filter xdm.event.id = "4769" and Failure_Code = "0x1f" | comp count() as count by xdm.source.ipv4,xdm.source.ipv6 | filter count > 15 | sort desc count

index=*_windows NOT user="*$" EventCode=4625 src_ip!="-" NOT [|inputlookup DomainControllers2.csv | table src_ip] NOT [|inputlookup WhitelistedServers.csv | table src_ip] | table _time dest EventCode user src src_ip signature Failure_Reason | stats dc(user) as distinct_users count by src_ip EventCode signature Failure_Reason | where distinct_users > 9 | sort +count +distinct_users | rename user as src_user user_nick as src_user_nick user_email as src_user_email user_managedBy as src_user_managedBy	datamodel dataset = microsoft_windows_raw | filter xdm.source.user.username !=""*$"" and xdm.event.id = ""4625"" and XDM_ALIAS.ip != ""-"" |join type=left (dataset = singtel_DomainControllers | fields src_ip) as DomainControllers DomainControllers.src_ip!=xdm.source.ipv4 or DomainControllers.src_ip!=xdm.source.ipv6 | filter src_ip=null |join type=left conflict_strategy = both (dataset = singtel_WhitelistedServers | fields src_ip) as WhitelistedServers WhitelistedServers.src_ip!=xdm.source.ipv4 or singtel_WhitelistedServers .src_ip!=xdm.source.ipv6 | filter src_ip=null | fields _time, xdm.target.ipv4, xdm.target.ipv6, xdm.event.id, xdm.source.user.username, xdm.source.ipv4, xdm.source.ipv6, xdm.event.operation_sub_type, xdm.event.outcome_reason, * | comp count_distinct(user) as distinct_users, count() as total by xdm.source.ipv4, xdm.target.ipv6, xdm.event.id , xdm.event.operation_sub_type, xdm.event.outcome_reason | filter distinct_users > 9 | sort desc total
index=*_windows EventCode=4624 Logon_Type=10 AND NOT ([| inputlookup PIAMServers2.csv | table src_ip] OR [| inputlookup JumpHost_HarmonyVDI.csv | table src_ip]) | table _time index EventCode Logon_Type user dest src_ip | rename user as src_user dest as host	datamodel dataset = microsoft_windows_raw | filter xdm.event.id = "4624" and xdm.logon.type = XDM_CONST.LOGON_TYPE_REMOTE_INTERACTIVE | join (dataset = singtel_PIAMServers2 | fields src_ip) as PIAMServers2 PIAMServers2.src_ip = xdm.source.ipv4 or PIAMServers2.src_ip = xdm.source.ipv6 | filter src_ip = null | join conflict_strategy = both (dataset = singtel_JumpHost_HarmonyVDI | fields src_ip) as JumpHost_HarmonyVDI JumpHost_HarmonyVDI.src_ip = xdm.source.ipv4 or PIAMServers2.src_ip = xdm.source.ipv6 | filter src_ip = null | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = xdm. source.user.username | fields _time, xdm.event.id, xdm.logon.type, xdm.source.user.username as src_user, xdm.target.ipv4 as host, xdm.source.ipv4, xdm.source.ipv6, *
index=singtel_windows AND EventCode=4719 | table _time EventCode host Security_ID user Category Changes | sort - _time | rename user as src_user	datamodel dataset = microsoft_windows_raw | filter xdm.event.id = "4719" | alter Security_ID = arrayindex(regextract(xdm.event.description, "Security ID:\s*(.*)\n\t"), 0) | alter Category= arrayindex(regextract(xdm.event.description, "Category:\s*(.*)\n\t"), 0) | alter Changes= arrayindex(regextract(xdm.event.description, "Changes:\s*(.*)"), 0) | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = xdm. source.user.username |fields _time,xdm.event.id, xdm.source.host.hostname, Security_ID, xdm.source.user.username as user, Category, Changes, * |sort desc _time

index=*_windows AND (EventCode=4720 OR EventCode=624) AND (Account_Name!=g-svcuam AND Account_Name!=EXCASGL06A$ AND Account_Name!=EXCASGL06B$ AND Account_Name!=EXCASGL06C$) AND domain!=CLDPRDADSG | eval src_user=mvindex(Account_Name,0) | eval dest_user=mvindex(Account_Name,1) | eval domain=mvindex(Account_Domain,0) | table _time domain host src_user src_user_nick dest_user Display_Name | rename Display_Name as dest_user_nick | sort -_time	datamodel dataset = microsoft_windows_raw | filter xdm.event.id in ("4720", "624") | alter Display_Name = json_extract_scalar(microsoft_windows_raw.event_data ,"$.DisplayName") \| filter xdm.source.user.username not in ("g-svcuam","EXCASGL06A$","EXCASGL06B$","EXCASGL06C$") and xdm.target.user.username not in ("g-svcuam","EXCASGL06A$","EXCASGL06B$","EXCASGL06C$")and xdm.source.user.domain != "CLDPRDADSG" | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = xdm.source.user.username | fields time, xdm.source.user.domain, xdm.source.host.hostname, xdm. target.user.username, xdm.source.user.username, Display_Name, * | sort desc _time
index=*_windows AND (((EventCode=4737 OR EventCode=4729 OR EventCode=4728) AND Group_Name="Domain Admins") OR ((EventCode=4732 OR EventCode=4733 OR EventCode=4746 OR EventCode=4747) AND Group_Name="*Administrators")) | eval dest_user=mvindex(Security_ID,1) | table _time host EventCode signature src_user dest_user Group_Domain Group_Name	datamodel dataset = microsoft_windows_raw | filter (xdm.event.id in ("4737", "4729","4728") and xdm.target.user.groups="Domain Admins") or (xdm.event.id in ("4732","4733","4746","4747") and xdm.target.user.groups contains "Administrators") | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = xdm. source.user.username | fields _time,xdm.source.host.hostname,xdm.event.id,xdm.event.operation_sub_type ,xdm.source.user.username,xdm.target.user.username,xdm.target.user.domain, xdm.target.user.groups
index=*_windows AND (EventCode=7045 OR EventCode=4697) NOT [|inputlookup process_whitelist.csv | table Service_File_Name] | table _time, index, EventCode, host, user, Service_Account, Service_File_Name , Service_Name, Service_Type | rename user as src_user	datamodel dataset = microsoft_windows_raw |filter (xdm.event.id in ("7045", "4697")) | alter Service_Account= json_extract_scalar(microsoft_windows_raw.event_data ,"$.ServiceAccount") \| alter Service_File_Name= json_extract_scalar(microsoft_windows_raw.event_data ,"$.ServiceFileName") |alter Service_Name= json_extract_scalar(microsoft_windows_raw.event_data ,"$.ServiceName") \|alter Service_Type= json_extract_scalar(microsoft_windows_raw.event_data ,"$.ServiceType") |join type=left (dataset = singtel_process_whitelist| fields Service_File_Name) as processlist processlist.svc_file_name = Service_File_Name | filter svc_file_name = null | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = xdm.source.user.username |fields _time, xdm.source.host.hostname ,xdm.event.id ,xdm.source.user.username, Service_Account, Service_File_Name , Service_Name, Service_Type, *

index=singtel_windows AND (EventCode=4724 OR EventCode=4723) src_user!=g-piamsvc AND src_user="itwadmin1" | eval SourceAccountName=mvindex(Account_Name,0)| eval TargetAccountName=mvindex(Account_Name,1) | table _time, EventCode, signature, host, SourceAccountName TargetAccountName | search TargetAccountName="itwadmin1" | rename SourceAccountName as src_user TargetAccountName as dest_user	datamodel dataset = microsoft_windows_raw |filter xdm.event.id in ("4724", "4723") and xdm.source.user.username!="g-piamsvc" and xdm.source.user.username="itwadmin1" |filter xdm.target.user.username="itwadmin1" | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = xdm. source.user.username |fields _time, xdm.event.id , xdm.source.host.hostname , xdm.source.user.username, xdm.target.user.username, *
index=*_windows EventCode=4728 AND (#itservicedesk OR #ITSD) | eval src_user=mvindex(Security_ID,0) | eval dest_user=mvindex(Security_ID,1) | table _time EventCode signature Group_Name src_user src_user_nick dest_user user | rename user as dest_user_nick | sort -_time	datamodel dataset = microsoft_windows_raw |filter microsoft_windows_raw.message contains "#itservicedesk" OR microsoft_windows_raw.message contains "#ITSD" |filter xdm.event.id in ("4728") |alter src_user= arrayindex(regextract(xdm.event.description, "Security ID:\s*(\S+)"), 0) |alter dest_user= arrayindex(regextract(xdm.event.description, "Security ID:\s*(\S+)"), 1) | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = xdm. source.user.username |fields _time , xdm.event.id , xdm.target.user.groups, xdm.event.operation_sub_type ,src_user, dest_user, xdm.source.user.username |sort desc _time
index=*_windows AND (EventCode=4707 OR EventCode=4706 OR (EventCode=4716 AND user!="ANONYMOUS LOGON")) | table _time index EventCode signature host status user TaskCategory Domain_ID Domain_Name dvc_nt_host Trust_Attributes Trust_Type Trust_Direction | rename user as src_user	datamodel dataset = microsoft_windows_raw |filter xdm.event.id in ("4707","4706") OR (xdm.event.id="4716" AND xdm.source.user.username!="ANONYMOUS LOGON") | alter Domain_ID= json_extract_scalar(microsoft_windows_raw.event_data ,"$.DomainSid") \| alter Trust_Attributes= json_extract_scalar(microsoft_windows_raw.event_data ,"$.TdoAttributes") | alter Trust_Type= json_extract_scalar(microsoft_windows_raw.event_data ,"$.TdoType") \| alter Trust_Direction= json_extract_scalar(microsoft_windows_raw.event_data ,"$.TdoDirection") | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = xdm. source.user.username | fields _time, xdm.event.id , xdm.event.operation_sub_type ,xdm.source.host.hostname , xdm.event.outcome ,xdm.source.user.username as user, xdm.event.original_event_type, Domain_ID, xdm.source.user.domain, xdm.target.host.hostname,Trust_Attributes, Trust_Type ,Trust_Direction, *

index=singtel_winsrv (EventCode=4624 Logon_Type IN (2, 10)) OR EventCode=4625 OR EventCode=4647 user!="DWM-*" | table _time host user user_nick EventCode Logon_Type name action src_ip | sort _time | eval Logon_Type = case (Logon_Type== 2,"Local Interactive", Logon_Type==10, "Remote Interactive") | rename host AS JumpHost, user AS UserID, user_nick AS User, name AS Description, action AS Action, src_ip AS SourceIP	datamodel dataset = microsoft_windows_raw | filter (xdm.event.id="4624" and (xdm.logon.type = "XDM_CONST.LOGON_TYPE_INTERACTIVE" or xdm.logon.type = "XDM_CONST.LOGON_TYPE_REMOTE_INTERACTIVE")) or xdm.event.id="4625" or xdm.event.id="4647" and xdm. source.user.username contains "DWM-" | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = xdm. source.user.username |fields _time, xdm.event.id , xdm.source.host.hostname , xdm.logon.type , xdm.source.user.username, xdm.target.user. username xdm.source.ipv4, * |sort desc _time
index=*_windows (EventCode=1102 OR EventCode=1100 OR EventCode=104) AND host=ADGL06B | eval dow = tonumber(strftime(_time,"%w")) | where dow != 0 AND dow != 6 | table _time EventCode signature host	datamodel dataset = microsoft_windows_raw | filter xdm.event.id in ("1102","1100","104") | alter dow = extract_time(current_time(), "DAYOFWEEK") |filter dow != 0 AND dow != 6 | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = xdm.source.user.username |fields _time, xdm.event.id ,xdm.source.host.hostname ,xdm.event.operation_sub_type, *
index=*_windows AND (EventCode=4724 OR EventCode=4723) AND user=ITWAdmin2 AND NOT (src_user=ITWAdmin2 OR src_user=g-piamsvc OR src_user=g-piambkp) | eval SourceAccountName=mvindex(Account_Name,0) | eval TargetAccountName=mvindex(Account_Name,1) | table _time, EventCode, signature, host, SourceAccountName, TargetAccountName | rename SourceAccountName as src_user TargetAccountName as dest_user	datamodel dataset = microsoft_windows_raw | filter xdm.event.id in ("4724","4723") and xdm.target.user.username="ITWAdmin2" and xdm.source.user.usernamen not in ("ITWAdmin2","g-piamsvc","g-piambkp") | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = xdm.source.user.username |fields _time, xdm.event.id , xdm.source.host.hostname , xdm.source.user.username, xdm.target.user.username, xdm. event.operation_sub_type, *
index=*_windows AND (EventCode=4720 OR EventCode=624) AND (Account_Name!=g-svcuam AND Account_Name!=EXCASGL06A$ AND Account_Name!=EXCASGL06B$ AND Account_Name!=EXCASGL06C$) AND ((EventCode=4728 AND Group_Name="Domain Admins") OR ((EventCode=4732 OR EventCode=4746) AND Group_Name="*Administrators")) | eval src_user=mvindex(Account_Name,0) | eval dest_user=mvindex(Account_Name,1) | eval domain=mvindex(Account_Domain,0) | table _time domain host EventCode Group_Name src_user src_user_nick dest_user Display_Name | rename Display_Name as dest_user_nick | sort -_time	datamodel dataset = microsoft_windows_raw | alter Display_Name = json_extract_scalar(microsoft_windows_raw.event_data ,"$.DisplayName") \|alter Group_Name= arrayindex(regextract(xdm.event.description, "Group Name:\s*(.*)\n\t"), 0) \|filter (xdm.event.id="4720" OR xdm.event.id="624") AND (xdm.source.user.username not contains ("g-svcuam","EXCASGL06A$","EXCASGL06B$","EXCASGL06C$") and xdm.target.user.username not contains ("g-svcuam","EXCASGL06A$","EXCASGL06B$","EXCASGL06C$")) AND ((xdm.event.id="4728" AND Group_Name="Domain Admins") OR ((xdm.event.id="4732" OR xdm.event.id="4746") AND Group_Name="*Administrators")) | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = xdm.source.user.username |fields _time, xdm.source.user.domain, xdm.source.host.hostname ,xdm.event.id ,Group_Name,xdm.target.user. username,xdm.source.user.username,Display_Name, * | sort desc _time

index=*_windows EventCode="4742" OR EventCode="4624" AND (src_user="*anonymous*" OR member_id="*S-1-0*") | eval local_system=mvindex(upper(split(user,"$")),0) | search host=local_system | table _time EventCode host local_system user Security_ID member_id src_nt_domain dest_nt_domain | rename user as src_user	datamodel dataset = microsoft_windows_raw |filter xdm.event.id in ("4742", "4624") | alter member_id= json_extract_scalar(microsoft_windows_raw.event_data ,"$.MemberSid") \| alter Security_ID= json_extract_scalar(microsoft_windows_raw.event_data ,"$.SubjectUserSid") | alter TargetDomainName= json_extract_scalar(microsoft_windows_raw.event_data ,"$.TargetDomainName") \| alter SubjectDomainName= json_extract_scalar(microsoft_windows_raw.event_data ,"$.SubjectDomainName") |filter xdm.source.user.username="*anonymous*" OR member_id="*S-1-0*" |alter local_system = uppercase(arrayindex(split(xdm.source.user.username,"$"),0)) |filter xdm.source.host.hostname=local_system | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = xdm. source.user.username |fields _time,xdm.event.id ,xdm.source.host.hostname ,local_system, xdm.source.user.username ,Security_ID, member_id,TargetDomainName,SubjectDomainName, *
index=*_windows AND (EventCode=5136 OR EventCode=5137 OR EventCode=5138 OR EventCode=5139 OR EventCode=5141) AND Class=groupPolicyContainer | table _time, index, EventCode, signature, src_user, host, DN, dir_svcs_action	datamodel dataset = microsoft_windows_raw | alter ObjectClass = json_extract_scalar(microsoft_windows_raw.event_data, "$.ObjectClass") |filter xdm.event.id in ("5136", "5137","5138","5139","5141") and ObjectClass = "groupPolicyContainer" | alter DN= arrayindex(regextract(xdm.event.description, "DN:\s*(\S+)"), 0) | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = xdm.source.user.username |fields _time, xdm.event.id ,xdm.event.operation_sub_type , xdm.source.user.username ,xdm.source.host.hostname , DN, dir_svcs_action, *
index=*_windows EventCode=4722 AND user=Administrator | rename user as dest_user | table _time EventCode host dest_user src_user	datamodel dataset = microsoft_windows_raw |filter (xdm.event.id="4722") |alter dest_user = coalesce(xdm.target.host.hostname , xdm.target.ipv4 , xdm.target.host.fqdn) |alter user=microsoft_windows_raw.user |filter user="Administrator" | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = user |fields _time,xdm.event.id , xdm.source.host.hostname , xdm.source.user.username , dest_user,user,*

(index=singtel_windows) EventCode=4624 (src_user="ITWAdmin1" OR src_user="ITWAdmin2" OR src_user=_logadmin OR src_user=g-svcuam OR src_user=,Äù breakglass,Äù) NOT [|inputlookup PIAMServers2.csv | table src_ip] NOT [|inputlookup DomainControllers2.csv | table src_ip] NOT [|inputlookup VDIADServers.csv | table src_ip] NOT [|inputlookup JumpHost_HarmonyVDI.csv | table src_ip] NOT [|inputlookup UAMServers.csv | table src_ip] NOT [|inputlookup WhitelistedServers2.csv | table src_ip] | table _time src_user src_ip host Logon_Type index	datamodel dataset = microsoft_windows_raw | filter xdm.event.id="4624" and xdm.source.user.username in ("ITWAdmin1","ITWAdmin2","_logadmin","g-svcuam","breakglass") | join type=left (dataset = singtel_PIAMServers| fields src_ip) as piam piam.src_ip!= xdm.source.ipv4 or piam.src_ip!= xdm.source.ipv6 | filter src_ip = null | join type=left (dataset = singtel_DomainControllers| fields src_ip) as dc dc.src_ip!= xdm.source.ipv4 or dc.src_ip!= xdm.source.ipv6 | filter src_ip = null | join type=left (dataset = singtel_VDIADServers| fields src_ip) as vdi vdi.src_ip!= xdm.source.ipv4 or vdi.src_ip!= xdm.source.ipv6 | filter src_ip = null | join type=left (dataset = singtel_JumpHost_HarmonyVDI| fields src_ip) as jumphost jumphost.src_ip!= xdm.source.ipv4 or jumphost.src_ip!= xdm.source.ipv6 | filter src_ip = null | join type=left (dataset = singtel_UAMServers |fields src_ip) as uam uam.src_ip!= xdm.source.ipv4 or uam.src_ip!= xdm.source.ipv6 | filter src_ip = null | join type=left (dataset = singtel_WhitelistedServers |fields src_ip) as whitelisted whitelisted.src_ip!= xdm.source.ipv4 or whitelisted.src_ip!= xdm.source.ipv6 | filter src_ip = null | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = xdm. source.user.username |fields _time, xdm.source.user.username , xdm.source.ipv4 ,xdm.source.host.hostname , xdm.logon.type, *
(index=ayrows_windows OR index=singtel_windows) EventCode=4624 AND [| inputlookup RUserList.csv | table user] AND NOT (user=CP862430) | table _time host user user_nick user_managedBy src_ip Logon_Type EventCode signature index | sort +_time | rename user as src_user user_nick as src_user_nick user_email as src_user_email user_managedBy as src_user_managedBy	datamodel dataset = microsoft_windows_raw |filter xdm.event.id="4624" |join type=left (dataset = singtel_RUserList|fields user) as ruser ruser.user=xdm.source.user.username |filter (xdm.source.user.username!="CP862430") | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = xdm.source.user.username |fields _time, xdm.source.user.username, xdm.source.ipv4, xdm.logon.type, xdm.event.id ,xdm.event.operation_sub_type, * |sort asc _time

index=*_windows (EventCode=4728 OR EventCode=4755 OR EventCode=4757) AND #ITWPADInfra | eval src_user=mvindex(Security_ID,0) | eval dest_user=mvindex(Security_ID,1) | table _time EventCode signature Group_Name src_user src_user_nick dest_user user | rename user as dest_user_nick | sort -_time	datamodel dataset = microsoft_windows_raw |filter xdm.event.id in ("4728","4757","4755") |filter microsoft_windows_raw.message contains "#ITWPADInfra" | alter src_user= json_extract_scalar(microsoft_windows_raw.event_data ,"$.SubjectUserSid") \| alter dest_user= json_extract_scalar(microsoft_windows_raw.event_data ,"$.MemberSid") |alter Group_Name= arrayindex(regextract(xdm.event.description, "Group Name:\s*(.*)\n\t"), 0) | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = xdm. source.user.username | fields _time, xdm.event.id, xdm.event.operation_sub_type , Group_Name, src_user, dest_user, xdm.source.user.username |sort desc _time
index=*windows Name="'Microsoft-Windows-Sysmon'" EventID=1 parent_process_name=wmiprvse.exe | where match(process, "(?i)cmd\.exe\s+\/Q\s+\/c") AND match(process, "\\\\127\.0\.0\.1\\.*") AND match(process, "__\\d{1,10}\\.\\d{1,10}") | table _time index host User parent_process_name process	datamodel dataset =microsoft_windows_raw | filter xdm.event.id ="1" and xdm.source.process.name="wmiprvse.exe" | alter match1 = arrayindex(regextract(xdm.source.process.executable.path , "(?i)cmd\.exe\s+\/Q\s+\/c"), 0) | alter match2 = arrayindex(regextract(xdm.source.process.executable.path , "\\\\127\.0\.0\.1\\.*"), 0) | alter match3 = arrayindex(regextract(xdm.source.process.executable.path , "__\\d{1,10}\\.\\d{1,10}"), 0) | filter match1 != null AND match2!=null AND match3 != null | join type = left (dataset = pan_dss_raw | fields sam_account_name, upn, department, manager_sam_account_name) as ad ad.sam_account_name = xdm. source.user.username | fields _time, xdm.source.host.hostname, xdm.source.user.username as user, xdm.source.process.name , xdm.source.process.executable.path , *
index=singtel_o365 sourcetype=o365:management:activity Workload=SecurityComplianceCenter Operation=AlertTriggered Severity = High | dedup AlertId | rex field=Data "\"(?<email>[\w\d\.\-]+\@[\w\d\.]+)\"" | stats count by _time, Severity, Status, email, Name, Comments, Category, AlertType, Data | fields - count | sort -_time	// rule created by palo //datamodel dataset in (/*msft_o365_azure_ad_raw, msft_o365_dlp_raw,*/ msft_o365_exchange_online_raw, msft_o365_exchange_online_raw, msft_o365_general_raw, msft_o365_sharepoint_online_raw) | filter xdm.observer.type = "SecurityComplianceCenter" and xdm.event.operation = "AlertTriggered" | filter xdm.alert.severity = "High" | dedup xdm.alert.original_alert_id | alter src_email = arrayindex(regextract(xdm.event.description, "\"([\w\d\.\-]+\@[\w\d\.]+)\""),0) | comp count() as count by _time, xdm.alert.severity, xdm.observer.action, src_email, xdm.network.rule, xdm.alert.subcategory, xdm.event.description | fields - count | sort desc _time //no change required dataset in (msft_graph_security_alerts_raw) | filter severity="high" and status="newAlert" and title!="External user file activity" and title!="Form blocked due to potential phishing attempt" | alter user_Name= json_extract(userStates, "$.0.accountName") \| alter device_Name=json_extract(hostStates,"$.0.fqdn") | alter url_id = if(len(id)=40, to_string(id), to_string("")) | alter URL = concat("https://security.microsoft.com/alerts/",url_id) | fields _time, title, severity, category, device_Name, user_Name, description ,URL, * | sort desc _time

	config timeframe = 7d // Corresponds to `from datamodel=Network_Traffic.All_Traffic` | datamodel dataset in (panw_ngfw_traffic_raw) // Adjust datasets as per your environment // Placeholder for the `dw_netr_00001_tstats_exceptions` macro // | filter <logic_from_dw_netr_00001_tstats_exceptions> // Corresponds to `All_Traffic.src IN (...) AND NOT All_Traffic.dest IN (...)` | filter xdm.source.ipv4 incidr "10.0.0.0/8" or xdm.source.ipv4 incidr "172.16.0.0/12" or xdm.source.ipv4 incidr "192.168.0.0/16" and not (xdm.target.ipv4 incidr "10.0.0.0/8" or xdm.target.ipv4 incidr "172.16.0.0/12" or xdm.target.ipv4 incidr "192.168.0.0/16") // Corresponds to `sum(All_Traffic.bytes_out) as volume by _time ... span=1h` | bin _time span = 1h | comp sum(coalesce(xdm.source.sent_bytes, 0)) as volume by _time, _product, xdm.source.ipv4 , xdm.source.user.username | fields volume, _time, _product as sourcetype, xdm.source.ipv4 as src, xdm.source.user.username as user // Placeholder for the `dw_netr_00001_exceptions` macro // | filter <logic_from_dw_netr_00001_exceptions> | filter timestamp_diff(_time,current_time(),"MINUTE") <= 70 // This block translates the `eventstats` logic using `windowcomp`. // It calculates the average volume for the period before the last 70 minutes. | alter historical_volume = if(timestamp_diff(_time,current_time(),"MINUTE") <= 70, volume, null) | windowcomp avg(historical_volume) by src, user as previous_avg // Corresponds to `where _time>=relative_time(now(), "-70m@m")` | filter timestamp_diff(_time,current_time(),"MINUTE") <= 70 // Corresponds to the series of `eval` commands. | alter previous_avg = round(coalesce(previous_avg, 0)), delta = if(previous_avg > 0, round(divide(volume, previous_avg)),null)
| tstats summariesonly=true allow_old_summaries=true fillnull_value="unknown" sum(All_Traffic.bytes_out) as volume from datamodel=Network_Traffic.All_Traffic where earliest=-`dw_netr_00001_timerange` latest=-10m@m `dw_netr_00001_tstats_exceptions` All_Traffic.src IN (10.0.0.0/8,172.16.0.0/12,192.168.0.0/16) AND NOT All_Traffic.dest IN (10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16) by _time index sourcetype All_Traffic.src All_Traffic.user span=1h | `dw_drop_dm_object_name("All_Traffic")` | `dw_netr_00001_exceptions` | eventstats avg(eval(if (_time < relative_time(now(), "-70m@m"),volume,null()))) as previous_avg by src user | where _time>=relative_time(now(), "-70m@m") | eval previous_avg=round(previous_avg) | eval delta = round(volume/previous_avg,2) | eval volume_GB=round(volume/1024/1024/1024,2) | eval previous_avg_GB=round(previous_avg/1024/1024/1024,6) | `dw_netr_00001_threshold` | table _time index sourcetype src priority user volume volume_GB previous_avg previous_avg_GB delta	