Splunk Use Cases

Tools, Tactics and Techniques

Content Sources

Consolidated and Curated by David Wells (@Epicism1)

Content Contributors:

- David Veuve (@davidveuve) – https://www.davidveuve.com
- Jim Apger (@JimApger)
- Marquis Montgomery
- Maybe others! We will update this doc

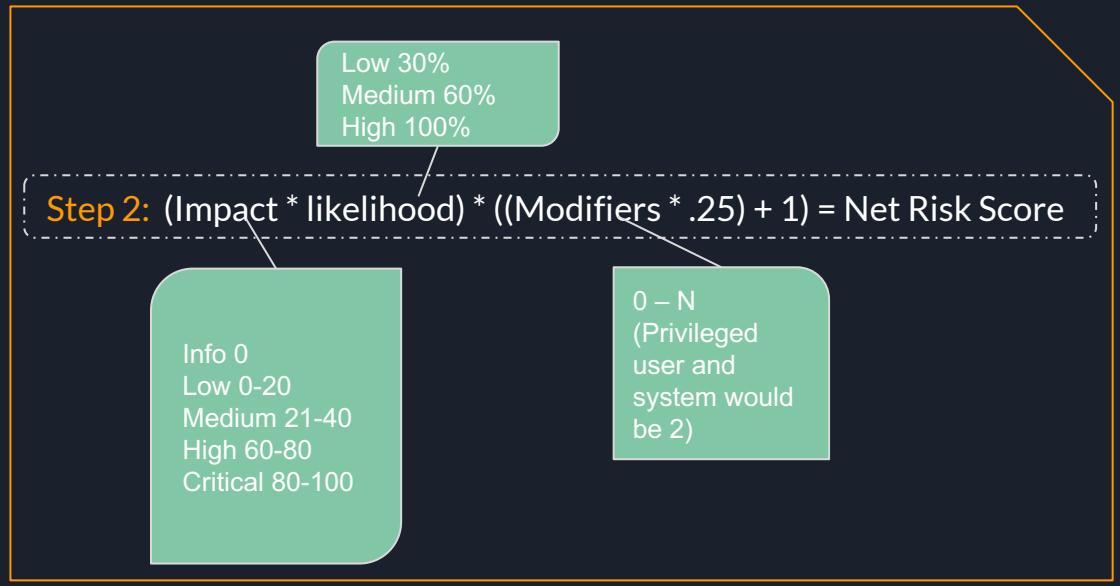
Recommended Resources:

- https://davidveuve.com/splunk.html
- https://davidveuve.com/talks/ninjutsu-part-six/Security-Ninjutsu-Part-Six-Slides-to-Source-Materials.pdf

Risk Scoring

Step 1: Risk Rule Properties

risk_score: See scoring
risk_object: The object to track
risk_object_type: The object type
risk_impact: Info to Critical
risk_liklihood: How likely event is this to occur
risk_id: This risk's unique identifier
risk_kcstage: This risk's kill chain stage



Step 3 (Example): Add Modifiers to Enhance the Risk Based on Another Field's values:

```
| eval risk = case(NumWhoReportIn>100, risk+10, risk)
| eval risk = case(like(Groups, "%OU=Groups,OU=IT Security,%"), risk + 10, risk)
| eval risk = case(like(title, "VP %"), risk+10, like(title, "Chief %"), risk+100, 1=1,
risk)
```

Risk Alerting I

Option 1: Risk Baselining with Confidence Checks
This verifies that the user is 3x their standard deviation AND there are at least 7 previous days worth of risk scores

```
Risk Baselining with Confidence Checking
<Pull risk scores>
| bucket _time span=1d
| stats sum(risk_score) as risk_score by user, _time
| stats count as num_data_samples max(eval(if(_time >= relative_time(now(), "-1d@d"), risk_score,null))) as latest
  avg(eval(if(_time<relative_time(now(), "-1d@d"), risk_score,null))) as avg stdev(eval(if(_time<relative_time(now(), "-1d@d"), risk_score,null))) as
  stdev by user
| where latest > avg + stdev * 3 AND num_data_samples > 7 AND latest > avg * 2
```

Option 2: Identify When A User's # of Risk Kill Chain (or category) is Above 2 and the Number of Unique Risk Signatures is Above 1:

```
| stats sum(risk_score) as risk_score_aggregate values(risk_id) as risk_id values(risk_description) as risk_description values(risk_kcstage) as
  risk_kcstage by risk_object
| where mvcount(risk_kcstage)>2 AND mvcount(risk_id)>1
```

Option 3: Calculate a User's 30 Day Risk Score As a Baseline and Identify When Today's is 3x Higher Than the Average:

```
index=risk earliest=-30d
| stats values(source) as search_names sum(risk_score) as thirty_day_risk sum(eval(if(_time > relative_time(now(), "-1d"),risk_score,0))) as
  one_day_risk by risk_object
        | eval threshold_1day = 500, threshold_30day = 1200
        | eventstats avg(thirty_day_risk) as avg_thirty_day_risk stdev(thirty_day_risk) as stdev_thirty_day_risk
| where one_day_risk>threshold_1day OR thirty_day_risk>threshold_30day OR thirty_day_risk> (avg_thirty_day_risk + 3 * stdev_thirty_day_risk)
```

Risk Alerting II

Option 4: Calculate if a User is Above the One Day Risk Threshold, the 30 Day Risk Threshold or More Than 3x Its Own Standard Deviation:

```
index=risk earliest=-30d
| stats values(source) as search_names     sum(risk_score) as thirty_day_risk sum(eval(if(_time >     relative_time(now(), "-1d"),risk_score,0))) as
  one_day_risk by     risk_object
| eval threshold_1day = 500, threshold_30day = 1200
| eventstats avg(thirty_day_risk) as avg_thirty_day_risk     stdev(thirty_day_risk) as stdev_thirty_day_risk
| where one_day_risk>threshold_1day OR     thirty_day_risk>threshold_30day OR     thirty_day_risk>(avg_thirty_day_risk + 3 *
  stdev_thirty_day_risk)
| eval risk_score_reason = case(one_day_risk>threshold_1day, "One Day Risk Score above " . threshold_1day,  thirty_day_risk>threshold_30day
  . " on " . strftime(now(), "%m-%d-%Y"), "Thirty Day Risk Score above " . threshold_30day, 1=1, "Thirty Day Risk Score more than three standard
  deviations above normal (>" . round((avg_thirty_day_risk + 3 * stdev_thirty_day_risk),2) . ")")
| fields - avg* stdev*
```

Detect Rare Actions I

Good: Identify When Something Is X Times Past Their Standard Deviation:

```
<datasource> | bucket _time span=1d | stats count by <monitored> _time
| stats max(eval(if(_time >= relative_time(now(), "-1d@d"),count, null))) as latest avg(eval(if(_time < relative_time(now(), "-1d@d"),count, null))) as avg
 stdev(eval(if(_time < relative_time(now(), "-1d@d"),count, null))) as stdev by <monitored>
| where latest > avg + 6*stdev
```

Better: Adding Relative Filters to Statistical Assessments:

```
tag=authentication | bucket _time span=1d
| stats dc(dest) as count by user, _time
| stats count as num_data_samples max(eval(if(_time >= relative_time(now(), "-1d@d"), count,null))) as latest avg(eval(if(_time<relative_time(now(),
 "-1d@d"), count,null))) as avg stdev(eval(if(_time<relative_time(now(), "-1d@d"), count,null))) as stdev by user
| where latest > avg + stdev * 3 AND num_data_samples > 7 AND latest > avg * 2
```

Example: The Above Command With # Credit Cards Viewed:

```
index=crm_logs viewed card
| bin span=1d _time | stats dc(card_id) as count by user _time
| stats count as num_data_samples max(eval(if(_time >= relative_time(now(), "1d"), count, null))) as latest avg(eval(if(_time < relative_time(now(),
 "-1d"),count,null))) as average, stdev(eval(if(_time < relative_time(now(), "-1d"),count,null))) as stdev by user
| where latest> 2*stdev+average AND num_data_samples>7 AND latest > avg * 2
```

Detect Rare Actions II

Over The Time Period, Has Anyone Done X More Than Usual (Using Inter-Quartile Range Instead of Standard Deviation)

```
<datasource>
| bucket _time span=1d
| stats count by <monitored>
| eventstats perc25(count) as perc25 perc75(count) as perc75 by <monitored> | where count > perc75 + (perc75 - perc25) * 1.5
```

Over The Time Period, Has Anyone Done X More Than Usual (Using Inter-Quartile Range Instead of Standard Deviation) - tStats Version

```
| tstats count from datamodel=<datamodel> where earliest=-30d@d by <monitored> _time span=1d
| eventstats perc25(count) as perc25 perc75(count) as perc75 by <monitored> | where count > perc75 + (perc75 - perc25) * 1.
```

Detect Rare Actions III: Using tStats

Unusual Detection with tStats

```
| tstats count latest(_time) as latest from datamodel=<...> where earliest=-30d@d by <monitored>
| eventstats sum(count) as total
| where count / total < 1/20000 AND latest > relative_time(now(), "-1d@d")
```

Detect unusual errors

```
| tstats count latest(_time) as latest from datamodel=Example_AWS_Security where earliest=-30d@d by cloudtrail.errorCode
| eventstats sum(count) as total
| where count / total < 1/20000 AND latest > relative_time(now(), "-1d@d")
```

Detect users with an unusual MFA Status

```
| tstats count latest(_time) as latest from datamodel=Example_AWS_Security where earliest=-30d@d by cloudtrail.mfaAuthenticated
 cloudtrail.userIdentity.arn | eventstats sum(count) as total
| where count / total < 1/20000 AND latest > relative_time(now(), "-1d@d")
```

Identifying First Time Event Attacks

Detect If This Is The first time that X has been Seen:

```
*Search Criteria
| stats earliest(_time) as earliest latest(_time) as latest by <field(s)>
| eval isOutlier=if(earliest >= relative_time(now(), "-1d@d"), 1, 0)
```

Detect if This Is The First Time Seen With tStats:

```
| tstats summariesonly=t allow_old_summaries=t min(_time) as earliest max(_time) as latest from datamodel=<..> by <..>
| where earliest > relative_time(now(), "-1d@d")
```

First Logon to New Server
sourcetype=win*security
| stats earliest(_time) as earliest latest(_time) as latest by user, dest
| eval isOutlier=if(earliest >= relative_time(now(), "-1d@d"), 1, 0)

Authentication against a New Domain Controller
sourcetype=win*security
| stats earliest(_time) as earliest latest(_time) as latest by user, dc
| eval isOutlier=if(earliest >= relative_time(now(), "-1d@d"), 1, 0)

First Access to a New Source Code Repository
sourcetype=source_code_access
| stats earliest(_time) as earliest latest(_time) as latest by user, repo
| eval isOutlier=if(earliest >= relative_time(now(), "-1d@d"), 1, 0)

First External Email Claiming to be Internal from Server
sourcetype=cisco:esa src_user=*@mycompany.com src!=10.0.0.0/8
| stats earliest(_time) as earliest latest(_time) as latest by user, src
| eval isOutlier=if(earliest >= relative_time(now(), "-1d@d"), 1, 0)

Familiar Filename on a New Path
Sourcetype=win*security EventCode=4688 `IncludeMicrosoftFiles`
| stats earliest(_time) as earliest latest(_time) as latest by filename, path
| eval isOutlier=if(earliest >= relative_time(now(), "-1d@d"), 1, 0)

New Database Table Accessed
sourcetype=database
| stats earliest(_time) as earliest latest(_time) as latest by user, table
| eval isOutlier=if(earliest >= relative_time(now(), "-1d@d"), 1, 0)

New Interactive Logon by Service Account
sourcetype=win*security user=srv_* Logon_Type=2 OR .. 11 .. 12
| stats earliest(_time) as earliest latest(_time) as latest by user, dest
| eval isOutlier=if(earliest >= relative_time(now(), "-1d@d"), 1, 0)

New Parent Process for cmd.exe
sourcetype=win*security EventCode=4688 filename=4688
| stats earliest(_time) as earliest latest(_time) as latest by parent_process
| eval isOutlier=if(earliest >= relative_time(now(), "-1d@d"), 1, 0)

Example: Detect When Users Take High Risk Actions From A New Country:

```
| tstats summariesonly=t allow_old_summaries=t min(_time) as earliest max(_time) as latest from datamodel=Example_AWS_Security where cloudtrail.HighRiskAPICalls>0 by cloudtrail.sourceIPAddress_Contry
| where earliest > relative_time(now(), "-1d@d")
```

Detect Rare Events I: Compared to all events

Detect Very Rare Events:

```
<datasource> earliest=-30d@d
| stats count latest(_time) as latest by <monitored> [optionally: <entity>]
| eventstats sum(count) as total [optionally: by <entity>]
| where count / total < 1/20000 AND latest > relative_time(now(), "-1d@d")
```

Example: Detect Rare API Calls (eventName):

```
sourcetype=aws:cloudtrail earliest=-30d@d | stats count by eventName | eventstats sum(count) as total
| where count / total < 1/20000 AND latest > relative_time(now(), "-1d@d")
```

Example: Detect Users Making Rare API Calls:

```
sourcetype=aws:cloudtrail earliest=-30d@d
| stats count by eventName, userIdentity.arn
| eventstats sum(count) as total by userIdentity.arn
| where count / total < 1/20000 AND latest > relative_time(now(), "-1d@d")
```

Detect Rare Events II: Compared to Itself

This Command Can Be Used To Identify Higher-risk IP Addresses Based On The Uniqueness of the IPS signature:

```
tag=ids tag=attack
| bucket _time span=1d
| stats count by severity signature dest _time
| stats sum(count) as count avg(count) as avg stdev(count) as stdev sum(eval(if(_time > relative_time(now(), "-1d"),
 count, 0))) as recent_count min(_time) as earliest by severity signature dest
| eventstats avg(avg) as avg_num_per_dest avg(earliest) as avg_earliest sum(count) as sig_wide_count
 sum(recent_count) as sig_wide_recent_count by signature
| where NOT (avg_earliest < relative_time(now(), "-1y") AND sig_wide_recent_count / sig_wide_recent_count < 0.05 AND
 priority <=3)
```

Alert When Users Who Usually Log Into Very Few Systems All Of A Sudden Log Into a Lot:

```
| tstats summariesonly=true count from datamodel=Authentication where earliest=-30d@d groupby
 Authentication.dest Authentication.user _time span=1d
| rename AuthenEcaEon.dest as dest AuthenEcaEon.user as user
| eval isRecent=if(_time>relative_time(now(),"-1d"), "yes", "no")
| stats avg(eval(if(isRecent="no",count,null))) as avg first(count) as recent by user, dest
| eventstats count(eval(if(avg>0,"yes",null))) as NumServersHistorically count(eval(if(recent>0,"yes",null))) as
 NumServersRecently by user
| eval Cause=if(isnull(avg) AND NumServersHistorically!=0, "This is the first logon to this server", "")
| eval Cause=if(NumServersRecently>3 AND NumServersHistorically * 3 < NumServersRecently,
 mvappend(Cause,"Substantial increase in the number of servers logged on to"), Cause)
| where Cause!=""
```

Risk Data Modelling Tips & Tricks

Good: Combine Multiple Data Sources Together Via tstats:

```
| tstats prestats=t summariesonly=t count(Malware_Attacks.src) as malwarehits from datamodel=Malware where Malware_Attacks.action=allowed
  groupby Malware_Attacks.src
| tstats prestats=t append=t summariesonly=t count(web.src) as webhits from datamodel=Web where web.http_user_agent="shockwave flash" groupby
  web.src
| tstats prestats=t append=t summariesonly=t count(All_Changes.dest) from datamodel=Change_Analysis where sourcetype=carbon_black OR
  sourcetype=sysmon groupby All_Changes.dest
        | rename web.src as src Malware_Attacks.src as src All_Changes.dest as src
| stats count(Malware_Attacks.src) as malwarehits count(web.src) as webhits count(All_Changes.dest) as process_launches by src
```

Better: Avoid "| transaction" Commands Via Eventstats and Stats*:

```
        sourcetype=ironport OR sourcetype=cisco:esa
        | eventstats values(TLS) as TLS values(src_ip) as src_ip values(…) as … by ICID
        | stats values(icid) AS icid values(src*) AS src* by mid
        | eval recipient_count=mvcount(recipient)
```

Example: Avoid "| transaction" Commands Via eventstats and Stats (Full):

```
        sourcetype=cisco:esa* earliest=-20m
| eventstats values(sending_server) as sending_server values(sending_server_dns_status) as sending_server_dns_status values(sending_server_dkim) as sending_server_dkim
  values(sending_server_tls_status) as sending_server_tls_status values(sending_server_tls_cipher) as sending_server_tls_cipher values(sending_server_whitelist) as sending_server_whitelist by
  icid
| stats min(_time) as _time max(_time) as email_processing_complete_time count(eval(searchmatch("Message Finished MID"))) as complete_count count(eval(searchmatch("Start MID"))) as
  start_count values(d) as d values(message_id) as message_id values(message_subject) as message_subject values(mid) as mid values(recipient) as recipient values(sender) as sender
  values(spam_status) as spam_status values(encoding) as encoding values(subject) as subject values(attachment) as attachment values(queue) as queue values(message_scan_error) as
  message_scan_error values(message_size) as message_size values(sending_server) as sending_server values(sending_server_dns_status) as sending_server_dns_status
  values(sending_server_dkim) as sending_server_dkim values(sending_server_tls_status) as sending_server_tls_status values(sending_server_tls_cipher) as sending_server_tls_cipher
  values(sending_server_whitelist) as sending_server_whitelist values(icid) as icid values(dcid) as dcid by mid
|  where complete_count > 1 AND start_count > 1 AND email_processing_complete_time >= relative_time(now(), "-7m@m") AND email_processing_complete_time < relative_time(now(),
  "-2m@m")
        | collect index=parsed_emails
```

***This solution is 3x faster than Transaction commands.**

Tips and Tricks

Use a Subsearch as Search Input:

```
[ | inputlookup inscope_ad_users.csv
| stats values(sAMAccountName) as search
| eval search= "(user=" . mvjoin(search, " OR user=") . ")" ]
```

Pull and Update a Lookup to Act as a Cache:

```
tag=authentication
| stats earliest(_time) as earliest latest(_time) as latest by
user, dest
| inputlookup append=t login_tracker.csv
| stats min(earliest) as earliest max(latest) as latest by
user, dest
| where latest > relative_time(now(), "-90d")
| outputlookup sample_cache_group.csv
| where earliest >= relative_time(now(), "-1d@d")
```

Using Eval Within a Stats to 'tag' or Count Data of Interest:

```
tag=authentication
| stats count(eval(action="success")) as successes
 count(eval(action="failure")) as failures
 values(eval(if(action="success",user,null))) as "Successful
 Users" count(eval(if(searchmatch("example of log
 message"), 1, null))) as "example hits"
 count(eval(if(match(email,
 "\@buttercupgames\.com"),1,null))) as buttercup_emails
```

Producer-Consumer Ratio Macro:

```
[pcr(2)]
args = in,out
definition = eval pcr_total=$in$+$out$ \
| eval pcr_ratio= (($out$-$in$)/pcr_total) \
| eval pcr_source_fraction = ((1 + pcr_ratio)/2), pcr_dest_fraction = ((1 - pcr_ratio)/2) \
| eval pcr_range = case(pcr_ratio > 0.4, "Pure Push", pcr_ratio > 0, "70:30 Export", pcr_r
Exchange", pcr_ratio >= -0.5, "3:1 Import", pcr_ratio > -1, "Pure Pull")
iseval = 0
```

Linear Trendline Macro:

```
[lineartrend(2)]
args = x,y
definition = eventstats count as numevents sum($x$) as sumX sum($y$) as sumY
 sum(eval($x$*$y$)) as sumXY sum(eval($x$*$x$)) as sumX2 sum(eval($y$*$y$))
 as sumY2 \
| eval slope=((numevents*sumXY)- (sumX*sumY))/ ((numevents*sumX2) -
 (sumX*sumX)) \
| eval yintercept= (sumY-(slope*sumX))/numevents\
| eval newY=(yintercept + (slope*$x$))\
| eval R=((numevents*sumXY) - (sumX*sumY))/
 sqrt(((numevents*sumX2)-(sumX*sumX))*
 ((numevents*sumY2)-(sumY*sumY)))\
| eval R2=R*R iseval = 0
```

5 Week Forecast Macro:

```
[forecast5w(4)]
args = val,confidence,reltime,days
definition = eval w=case( \
(_time>relative_time(now(), "$reltime$@d-5w-30m") AND _time<=relative_time(now(
5w+$days$d+30m")), 5, \
(_time>relative_time(now(), "$reltime$@d-4w-30m") AND _time<=relative_time(now(
4w+$days$d+30m")), 4,\
```

Individual features to Cluster Machine Learning in 3 easy steps

Step 1: Example of Taking Multiple Values of Interest and Calculating Their Count, Distinct Count, Sum, etc. to Provide a Historic Trend.

```
index=sfdc
| bucket _time span=1d
| stats dc(eval(if(like(URI_ID_DERIVED, "00140000%"),
URI_ID_DERIVED, null))) as NumAccounts
dc(eval(if(like(URI_ID_DERIVED, "0063300%"),
URI_ID_DERIVED, null))) as NumOpts
sum(ROWS_PROCESSED) as ROWS_PROCESSED
count(eval(EVENT_TYPE="Login")) as Logins
count(eval(EVENT_TYPE="Report")) as ReportsIssued
count(eval(EVENT_TYPE="API" OR
EVENT_TYPE="BulkApi" OR EVENT_TYPE="RestApi"))
as APICalls
sum(DB_CPU_TIME) as DB_CPU_Time
sum(RUN_TIME) as RUN_TIME
sum(DB_BLOCKS) as db_blocks
dc(CLIENT_IP) as UniqueIPs
dc(ORGANIZATION_ID) as NumOrganizations
dc(ENTRY_POINT) as ApexExecution_Entry_Type
by USER_ID _time
```

Step 2: Then Calculate the 'Z' Score (e.g. How Many Stdev Away From Avg It Is) Per User; Reduce the Number of Fields to 5 for Processing Efficiency; Then Machine Learning Magic It:

```
<Previous search results>
                | eventstats avg(*) as AVG_* stdev(*) as STDEV_* by
USER_ID
                | foreach *
                [ eval "Z_<>" = ('<>' - 'AVG_<>' ) / 'STDEV_<>']
                | fields - AVG_* STDEV_*
                | fillnull
                | fit PCA k=5 Z_*
                | fit KMeans k=5 PC_*
                | eventstats max(clusterDist) as maxdistance
p25(clusterDist) as
        p25_clusterDist p50(clusterDist) as p50_clusterDist
        p75(clusterDist) as p75_clusterDist dc(USER_ID) as NumIDs
        count as NumEntries by cluster
       | eval MaxDistance_For_IQR= (p75_clusterDist + 12 *
        (p75_clusterDist - p25_clusterDist))
                | where NumEntries < 5 OR clusterDist >
MaxDistance_For_IQR
```

Peer Group Comparison

The Following Search Compares a User's Actions to Their Peer Group. Peer Groups Can be Defined Via Active Directory Groups or Workday Type Platforms:

```
<Event that you want to verify if it's the first time that you've seen it>
        | stats earliest(_time) as earliest latest(_time) as latest by user, dest
        | inputlookup append=t sample_cache_group.csv
        | stats min(earliest) as earliest max(latest) as latest by user, dest
        | outputlookup sample_cache_group.csv
        | lookup peer_group.csv user OUTPUT peergroup
        | makemv peergroup delim=","
        | multireport
        [| stats values(*) as * by user dest ]
        [| stats values(eval(if(earliest>=relative_time(now(),"-1d@d"),dest ,null))) as peertoday
          values(eval(if(earliest<relative_time(now(),"-1d@d"),dest ,null))) as peerpast by peergroup dest ]
        | eval user=coalsce(user, peergroup)
        | fields - peergroup
        | stats values(*) as * by user dest
        | where isnotnull(earliest)
        | isOutlier= if(isnotnull(earliest) AND earliest>=relative_time(now(),"-1d@d") AND isnull(peerpast),1,0)
```

Geolocation Comparison

The following search compares the longitude and latitude of a user and calculates the risk:

```
index=bro_http (”public_domain1" OR ”public_domain2” OR ”public_domain.co.uk") (hv_user1 OR hv_user2 OR hv_user3 OR hv_user4)
| fields src_ip
| bin span=1d _time
| rex field=_raw "(?mi)(username|user|login)=(?<uname>.{0,35}?)(@|&)"
| eventstats count as mul._ip by _.me, uname, src_ip
| where mul._ip > 1
| stats count as count by _.me, uname, src_ip
| geoip src_ip
| fillnull value="-"
| where src_ip_region_name != "-"
| eventstats sum(count) as count_per_user by _.me, uname
| eval src_ip_lat_w_avg = src_ip_la.tude * (count/count_per_user)
| eval src_ip_lon_w_avg = src_ip_longitude * (count/count_per_user)
| eval src_ip_lat_w_stdev = sqrt((src_ip_la.tude * src_ip_la.tude)/count_per_user) * (count/count_per_user)
| eval src_ip_lon_w_stdev = sqrt((src_ip_longitude * src_ip_longitude)/count_per_user) * (count/count_per_user)
| eventstats sum(src_ip_lat_w_avg) as src_ip_lat_avg2, sum(src_ip_lon_w_avg) as src_ip_lon_avg2, sum(src_ip_lat_w_stdev) as src_ip_lat_stdev2, sum(src_ip_lon_w_stdev) as src_ip_lon_stdev2 by _time, uname
| eval src_ip_lat_threshold_low2 = src_ip_lat_avg2 -- src_ip_lat_stdev2
| eval src_ip_lat_threshold_high2 = src_ip_lat_avg2 + src_ip_lat_stdev2
| eval src_ip_lon_threshold_low2 = src_ip_lon_avg2 -- src_ip_lon_stdev2
| eval src_ip_lon_threshold_high2 = src_ip_lon_avg2 + src_ip_lon_stdev2
| where src_ip_la.tude > src_ip_lat_threshold_high2 OR src_ip_la.tude < src_ip_lat_threshold_low2 OR src_ip_longitude > src_ip_lon_threshold_high2 OR src_ip_longitude < src_ip_lon_threshold_low2
| stats count as geo_count by _time, uname, src_ip_city, src_ip_region_name, src_ip_country_code
```

Future Value Slides

One day...

User Risk Modifiers

Add a Risk Multiplier To A User Based On Their Title:

```
| inputlookup LDAPSearch
| eval risk = 1
| eval risk = case(NumWhoReportIn>100, risk+10, risk)
| eval risk = case(like(Groups, “%OU=Groups,OU=IT Security,%”), risk + 10, risk)
| eval risk = case(like(Etle, “VP %”), risk+10, like(Etle, “Chief %”), risk +100, 1=1, risk)
| fields risk sAMAccountName
| outputlookup RiskPerUser
```

Analysis of User Risk:

```
[... insert your Privileged User AcEvity Search ...]
| stats count by user
| lookup RiskPerUser sAMAccountName as user
| eval AggRisk = risk * count
| eval DescripEveRisk = case(AggRisk > 100, “very high”, AggRisk>30, “medium”, AggRisk>5, “low”, 1=1, “very low”)
```

Microsoft’s Five types of Alerts

Alerts

- These activities have a significant service impact and are rarely due to benign activity. For example, a new account being granted Domain Administrator privileges would be classified as an alert.
- An alert immediately generates an escalation / page to be reviewed.

Atomics

- These are activities that are significant, and unlikely to be benign, but don't risk the enterprise if they're not responded to in short order.
- For example, a new local account being created on an important system.
- All atomics should be reviewed, but doing so can happen in their own time.

Behavioral

- These activities may occur due to benign service operations but may also indicate unauthorized activity.
- An example of a Behavioral indicator is a new process executing that has never been observed across the service.
- These won't be reviewed on their own, but will show up if grouped with other behaviorals or atomics through the clustering described inthe post below.

Contextual

- These activities occur very frequently due to benign activity but have forensic value during an investigation. A net.exe process start is one type of Contextual indicator.
- These would never be reviewed directly on their own, but are available to analysts to provide starting points and situational awareness during an investigations.

Source: https://blogs.technet.microsoft.com/office365security/defending-office-365-with-graph-analytics/

Saved Searches

To ensure that saved searches are not skipped, enable the option "realtime_schedule = 1"

Controls the way the scheduler computes the next execution time of a scheduled search.

If this value is set to 1, the scheduler bases its determination of the next scheduled search execution time on the current time.

If this value is set to 0, the scheduler bases its determination of the next scheduled search on the last search execution time. This is called continuous scheduling.

If set to 1, the scheduler might skip some execution periods to make sure that the scheduler is executing the searches running over the most recent time range.

If set to 0, the scheduler never skips scheduled execution periods.

However, the execution of the saved search might fall behind depending on the scheduler's load. Use continuous scheduling whenever you enable the summary index option.

The scheduler tries to execute searches that have realtime_schedule set to 1 before it executes searches that have continuous scheduling (realtime_schedule = 0).

* Defaults to 1