Overview

This document contains hints, use cases and FAQs related to the development of statements using Esper's event processing language (EPL) with Security Analytics' ESA. For recommendations on deployment of rules, please see the ESA Enablement Guide.

Performance Hints

The following table outlines a few of the areas where performance optimizations may be made to some of the most common statement syntax

From the Esper documentation, there are some additional general performance tips here http://espertech.com/esper/release-5.3.0/esper-reference/html_single/index.html#performance-tips.

Recommendation	Example Rule	Esper Documentation
Use hint with view .std:groupwin	HTTP Uploads - Office Documents Volume – High	Whenever using view .std:groupwin or aggregation 'group by' add Hint to reclaim group for length of time for statement http://espertech.com/esper/release-5.3.0/esper-reference/html_single/index.html#view-std-groupwin
Write strict MATCH RECOGNIZE patterns	Account Created And Deleted In Short Period Of Time	Avoid using quantifiers of + or * within MATCH REGOGNIZE statements if possible. The system will need to track fewer states. Consider writing a PATTERN-based statement in you need to use a loose pattern based on requirements http://espertech.com/esper/release-5.3.0/esper-reference/html_single/index.html#match-recognize-patternops-precedence
Suppress Duplicate Matches	Detecting Dyreza infections	Use every-distinct for pattern matches or in 10.5 and higher @SuppressOverlappingMatches annotation

		http://espertech.com/esper/release-5.3.0/esper-reference/html_single/index.html#pattern-logical-everydistinct
Use output rate limiting	Sources Communicating With Known Bad Hosts – High	Use output rate limiting such as OUTPUT FIRST EVERY X MINUTES to suppress generation of alerts if use case allows. This will save database space. http://espertech.com/esper/release-5.3.0/esper-reference/html_single/index.html#appendix_outputspec Using .win:time() behaves differently than .win:time_length_batch. Events are not removed from the window once logic is triggered and may create additional alerts with the same events. If only need the first within a timeframe using output rate limiting
Avoid the creation of many grouping windows	Excessive upload to File Cloud Services – High	Avoid the creation of many grouping windows (e.g. hundreds of thousands) over a long period of time if many unique values are expected
Avoid very general statements	-	Alerts or general filter conditions that could cause hundreds of thousands of events to be added to a window. You could use the filter on the Investigation over the proposed time window to gauge the level of events that may match. For health and wellness type alerts, see: Health and Wellness Monitoring and Alarms

Use hint with view .std:groupwin: HTTP Uploads - Office Documents Volume – High

The @Hint("reclaim_group_aged=*age_in_seconds*") hint instructs the engine to discard aggregation state that has not been updated for *age_in_seconds* seconds. The age_in_seconds should match the time window added to the statement.

```
@Hint('reclaim_group_aged=60')

SELECT * FROM Event(

medium = 32

AND (msg_id.toLowerCase() = 'post')

AND user_dst IS NOT NULL

AND extension IN ('pdf','doc','xls','docx','xslx','ppt','pptx')          AND rbytes > 10000

AND device_type='cacheflowelff'

AND alert IN ('VIP', 'CritAsset')

).std:groupwin(user_dst)

.win:time_length_batch(60 seconds,20)

GROUP BY user_dst HAVING COUNT(*) = 20;
```

Write strict MATCH RECOGNIZE patterns: Account Created And Deleted In Short Period Of Time

The pattern defined for the match recognize statement should be strict if at all possible. This essentially means that repetition operators such as + or * should be eliminated.

From:

```
SELECT * FROM Event(

medium = 32  AND ec_subject='User'  AND ec_outcome='Success'  AND user_src is NOT NULL
AND ( ec_activity='Create' OR  ec_activity='Delete' )    ).win:time(7200 seconds)
match_recognize (   partition by user_src   measures C as c, D as d   pattern (C+ D)   define   C
as C.ec_activity='Create' ,   D as D.ec_activity='Delete');
```

To:

```
SELECT * FROM Event(  medium = 32  AND ec_subject='User'  AND ec_outcome='Success'  AND
user_src is NOT NULL  AND ( ec_activity='Create' OR  ec_activity='Delete' )    ).win:time(7200
seconds)   match_recognize (   partition by user_src   measures C as c, D as d   pattern (C D)
define   C as C.ec_activity='Create' ,   D as D.ec_activity='Delete');
```

Suppress Duplicate Matches: Administrative Activity followed by User Creation

A new thread will be created for every 'a' event in the first statement below. This means that multiple 'a' events will match with the same 'b' event. This could result in unexpected and undesirable number of alerts for the same user during the time window. It is recommended to use the hint @SuppressOverlappingMatches with the PATTERN syntax using every.

From:

```
SELECT * FROM PATTERN  [

every a = Event(device_class='Web Logs'

AND host_dst = 'icanhazip.com')

-> b = Event(category LIKE '%Botnet%' AND device_class='Web Logs'  AND
user_dst=a.user_dst)

where timer:within(300 seconds)
```

To:

```
SELECT * FROM PATTERN @SuppressOverlappingMatches [

every a = Event(device_class='Web Logs'

AND host_dst = 'icanhazip.com')

-> b = Event(category LIKE '%Botnet%' AND device_class='Web Logs'  AND
user_dst=a.user_dst)

where timer:within(300 seconds)
```

Use output rate limiting: Sources Communicating With Known Bad Hosts – High

If a rule is triggering frequently, you may only want to store the first occurrence within a time period per unique meta value. This means the alerts are not stored and taking up space in the database. This is separate from notification suppression, which may be done within the UI and does not influence alert storage.

The alerts below are suppressed per user_dst using 'output first every' syntax. Only the first alert within the 60 minute time window will be stored in the database and alerted. Allow constituent events to be retained within the alert by using window aggregation 'window(*)' Result without window aggregation would be only the first of the 20 events per user_dst.

From:

```
@Hint('reclaim_group_aged=300')

SELECT * FROM Event(
medium = 32
AND alert_id LIKE 'known_bad%'
AND alert IN ('VIP' , 'CritAsset')
 ).std:groupwin(ip_src)
.win:time(1 Minutes)
GROUP BY ip_src HAVING COUNT(*) >= 20;
```

To:

```
@Hint('reclaim_group_aged=300')

SELECT window(*) FROM Event(
medium = 32
AND alert_id LIKE 'known_bad%'
AND alert IN ('VIP' , 'CritAsset')
 ).std:groupwin(ip_src)
.win:time(1 Minutes)
GROUP BY ip_src HAVING COUNT(*) >= 20 ouptut first every 60 minutes;
```

Use Cases

The solutions are explained through a series of comments within the Solution box. The comments are surrounded by /* */.

Alert if events occur within a time interval and the absence of another event is detected

Statement Summary:

Alert after receiving 10 different IDS events from the same source within 10 minutes but only if within those 10 minutes we don't see a TCP RST sent from the destination IP. This is an example of correlating packet and log data. Our F5's will do a TCP RST on the inbound web requests for unknown paths, so in this instance I only want to be alerted when a source receives 10 unique attacks to a single destination and that destination hasn't responded to the web requests.

Solution:

```
/* Required annotation to trigger an alert with the advanced statement */


@RSAAlert

/* Instruct the engine remove duplicate matches to the first ‘a’ event */
SELECT * FROM pattern @SuppressOverlappingMatches
[
/* Intrushield policy event */

every a=Event (

device_type IN ( 'intrushield' )
AND ip_src is not null
AND ip_dst is not null
AND policy_name is not null
AND policy_name NOT LIKE '%P2P%'
)
-> (


/* The 10 minute time window following the first event  */

timer:interval(600 seconds)

AND

/* 9 more Instrushield policy violations each with a unique policy_name and the same ip_src
and ip_dst.   */


[9] b= Event (
device_type IN ( 'intrushield' )
AND ip_src = a.ip_src
AND ip_dst = a.ip_dst
AND policy_name is not null
AND policy_name NOT LIKE '%P2P%'
AND policy_name != a.policy_name
)

/* Both the statement for event b and event c must evaluate to true for the syntax to match. In
other words, no TCP RST can occur to match the pattern. */

AND NOT
c=Event (medium=1 AND tcp_flags_seen ='rst' AND ip_dst=a.ip_dst)
```

```
)
]
/* Ensure all b events for unique policy names between them. */

where b.distinctOf(i => i.policy_name).countOf() = 9;
```

How to correlate events that arrive out of order?

Correlates 3 events that populate the same ip_dst and occur within 30 of each other in any order

http://espertech.com/esper/release-5.3.0/esper-reference/html_single/index.html#epl-join-inner

```
/*
Intrusion Detection with Nonstandard HTTPS Traffic and ECAT Alert
Single host generates IPS alert on destination IP on port TCP/443
accompanied by traffic to TCP/443 that is not HTTPS with the target
host generating an ECAT alert within 5 minutes.
*/

/*
Create a window to store the IPS, nonstandard traffic and ECAT alerts
*/
@Name('create')
Create Window HttpsJoinedWindow.win:time(15 minutes)(device_class string, ip_dstport
integer, service integer , tcp_dstport integer, device_type string, ip_dst string);

/*
Insert into the window the IPS, nonstandard traffic and ECAT alerts
*/
@Name('insert')
INSERT INTO HttpsJoinedWindow
SELECT * FROM
Event
(
(ip_dst IS NOT NULL and device_class IN ('IPS', 'IDS', 'Firewall') AND ip_dstport=443)
OR
(ip_dst IS NOT NULL and service!=443 and tcp_dstport=443)
OR
(ip_dst IS NOT NULL and device_type='rsaecat')
);
```

```
/*
Alert to the combination of all three events: IPS, nonstandard traffic and ECAT alerts
*/
@RSAAlert
INSERT INTO HttpsIntrusionTrigger
SELECT * FROM
        HttpsJoinedWindow(ip_dst IS NOT NULL and device_class IN ('IPS', 'IDS', 'Firewall') AND
ip_dstport=443) as s1,
        HttpsJoinedWindow(ip_dst IS NOT NULL and service!=443 and tcp_dstport=443) as s2,
        HttpsJoinedWindow(ip_dst IS NOT NULL and device_type='rsaecat') as s3
        where s1.ip_dst = s2.ip_dst and s1.ip_dst = s3.ip_dst;

/*
Delete all events from the joined window that caused the alert so they won't be reused
*/
@Name('delete')
on HttpsIntrusionTrigger delete from HttpsJoinedWindow as j where s1.ip_dst=j.ip_dst;
```

Only fire rule within business hours

http://espertech.com/esper/release-5.3.0/esper-reference/html_single/index.html#context_def_nonoverlapping

Non working hours.

- Set the working hours as '09:00' – '18:00'
- Any 'event.cat.name LIKE system.config%' after the working hours will trigger.

```
create context NotWorkingHours start (0, 18, *, *, *) end (0, 9, *, *, *);

context NotWorkingHours select * from Event(event_cat_name LIKE 'system.config%');
```

Administrative Activity followed by User Creation

Solution:

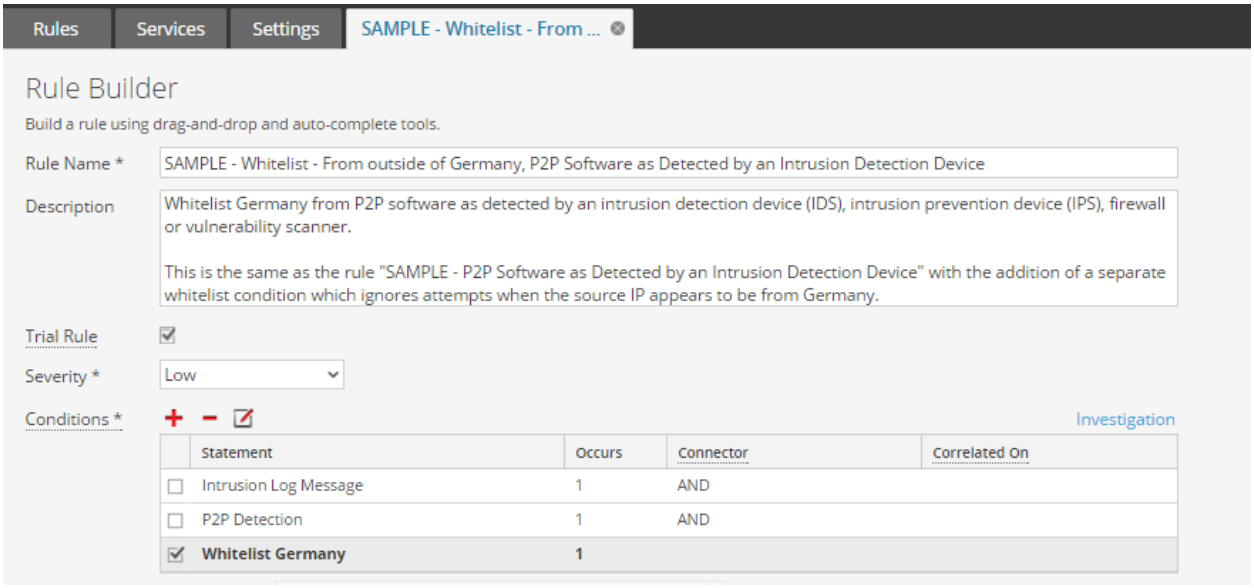
- Added parenthesis around the second and third events to require both the 2nd and 3rd events to occur after the 1st event. See Esper documentation: *7.5.1.3 Every Operator Example*

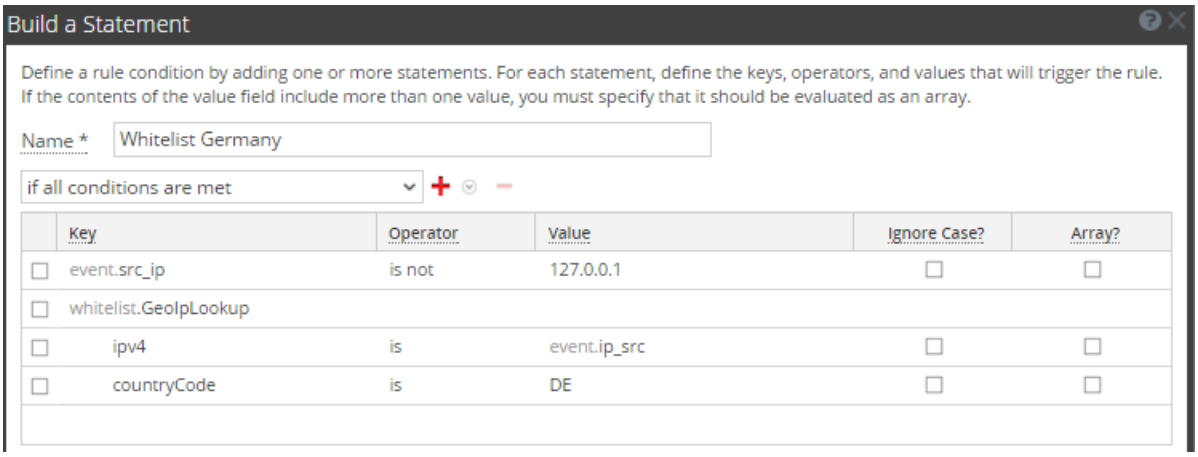
- Moved the time window of 60 minutes to the pattern statement so all 3 events would need to occur within that time
- Added @SuppressOverlappingMatches. Use the @SuppressOverlappingMatches pattern-level annotation to instruct the engine to discard all but the first match among multiple overlapping matches.

```
SELECT * from pattern @SuppressOverlappingMatches [

every e1=Event(device_class IN ('Windows Hosts') AND reference_id IN ('4720', '4722', '4724','4738') )

->

(

timer:interval(1 Seconds) AND e2=Event(event_cat_name LIKE 'Config.Changes%' AND user_dst=e1.user_src)

-> timer:interval(1 Seconds) AND e3=Event(device_class IN ('Windows Hosts') AND reference_id IN ('4726') AND user_src=e2.user_dst)

)

]
```

How to leverage enrichments (CSV files, database) within ESA Rules such as domain or IPs

Within 10.5.1 and higher, you can use the Enrichments feature. Once the enrichment is defined, it may be used within a rule condition using the Rule Builder or through an Advanced rule. Below is a sample rule that is a part of the base ESA installation. This demonstrates using the Rule Builder to whitelist German sources.





With 10.4, you can create a Named Window to store IPs and update the window based on matching filter criteria. Only trigger if a second event occurs and the IP is on the watchlist. The user is only kept on the watchlist for 15 minutes. Use may delete from a named window based on a triggering event.

http://espertech.com/esper/release-5.3.0/esper-reference/html_single/index.html#named_delete

```
create window WatchListIPs.win:time(15 min) (ip_src string);

insert into WatchListIPs select ip_src from Event(category LIKE '%scan%');

 @RSAAlert
select * from Event(category LIKE '%malicious%') WHERE ip_src in (SELECT ip_src from WatchListIPs );
```

How to computer percentages/rations/averages/counts/min/max within a given time window

Note: Computations over a large number of events and/or time periods are performance and memory intensive. Use caution when deploying the rules. See the ESA Enablement Guide for *10.5*. https://sadocs.emc.com/@api/deki/files/53102/ESA-enablement-guide-10_5.pdf

One way is to use named windows. However, this stores events in memory, which may cause issues if storing over a long period or large number of events.

```
CREATE WINDOW SizePerIP.win:length(100) (ip_src string,size long);

INSERT INTO SizePerIP SELECT ip_src AS ip_src, sum(size) AS size FROM Event.win:time_batch(1 minute) GROUP BY ip_src;

@RSAAlert(oneInSeconds=0)
SELECT ip_src FROM SizePerIP GROUP BY ip_src HAVING size > avg(size)*2;
```

Using a non-overlapping context does not retain events in memory and should be the preferred solution.

```
/*
Create a non-overlapping context to store data by second
*/
create context PerSecond start @now end after 1 second;

/*
Sum session size per second
*/

context PerSecond

insert into OneSecondBucket

select ip_src, sum(size) as size

from Event group by ip_src output snapshot when terminated;



/*
```

```
Alert if the total size for one second within an hour is two times greater than average
*/

@RSAAlert

select  ip_src from OneSecondBucket.win:time(1 hour) group by ip_src HAVING size >
avg(size)*2;
```

Frequently Asked Questions

Question: What Regex filter support is supported?

See: http://espertech.com/esper/release-5.3.0/esper-reference/html_single/index.html#epl-operator-ref-keyword-regexp

The rexexp function matches the entire region against the pattern
via java.util.regex.Matcher.matches()method. Please consult the Java API documentation for more information or refer to Regular Expression Flavors.

Question: What is the difference between the count VS time length batch

```
SELECT * FROM Event(filter_criteria)
.std:groupwin(ip_src)
.win:time_batch(1 minute)
GROUP BY ip_src
HAVING count(*) > 10;
```

When the time window of 1 minute is reached it will output everything within it that matches an ip_src. The HAVING count clause instructs the engine to only output after the time window if the count of events is greater than 10. The GROUP BY ip_src aggregation instructs the count to apply to only a single ip_src instead of across all ip_src that match the filter criteria.

Question: How to identify periods of times that exceed memory usage

With 10.5 and above, the Health and Wellness monitoring and alarms may be sufficient to monitor memory use.

https://sadocs.emc.com/0_en-us/089_105InfCtr/215_SysAdm/MonitorHlth

See also the ESA Enablement Guide for ***10.5***. It gives a good overview of

- Deployment best practices using trial rules
- Alarms for memory utilization and service status
- Per rule memory usage reporting

Question: How to feed rules into collector and create rules on top of rules

In 10.4, you would need to output the ESA alerts to syslog and create a custom parser. In 10.5 and above, you could deploy the CEF parser from Live and use this to parse the ESA alerts sent over syslog in that format. The additional ESA rules or reports would need to be written against the parser output for the ESA alerts.