

Стандартное правило (standard)

"Обновить параметры" нужно делать в корреляторе, когда какое-либо правило меняется, чтобы подтянулись актуальные изменения в правилах в коррелятор.

<https://www.youtube.com/embed/TauttDGugBc?si=jw05NxFfgxOuyioY>

Стандартное правило (standard) — срабатывает при достижении определенного порогового значения группы событий, которые удовлетворяют условиям селектора, полей группировки событий (на основе значений поля создается группа) и времени жизни контейнера для группы.

Если **частота срабатывания (Rate limiting)** явно не указана, то устанавливается лимит умолчанию - 100 срабатываний в секунду. При превышении лимита правило ничего не делает.

Политика хранения базовых событий (Base events keep policy) - указание, какие из базовых событий должны сохраняться в корреляционном. Возможно указать одно из значений:

- first (по умолчанию) - сохранять только первое базовое событие от каждого селектора в корреляционном событии
- last - сохранять только последнее базовое событие от каждого селектора в корреляционном событии
- all - сохранять все базовые события в корреляционном событии

Типовой пример правила (обнаружение сканирования портов, перебор портов >30 назначения, от одного адреса источника и назначения в течение 60 секунд):

Общие

Селекторы

Действия

Название

[general] Обнаружено сканирование портов [B]

Тенант

test2

Тип

standard

Группирующие поля

+ Добавить поле

SourceAddress

DestinationAddress

Очистить

Уникальные поля

+ Добавить поле

DestinationPort

Очистить

Частота срабатываний

1000

Время жизни контейнера, сек.

60

Политика хранения базовых событий

all

Уровень важности

Средний

Сортировать по

Описание

Правило обнаруживает перебор различных 30 портов по одному хосту в течение 60 сек.

Общие

Селекторы

Действия

Селектор "More than 30 ports"

Параметры

Локальные переменные

Название

More than 30 ports

Порог срабатывания селектора (количество событий)

30

Фильтр

Создать

Сохранить фильтр

Условия

+ Добавить условие

+ Добавить группу

+ Добавить фильтр

И

Если

поле события

DeviceVendor

=

/1

список

×

Если

поле события

DestinationPort

<=

константа

1824

×

Если не

поле события

Type

=

константа

3

×

Если не

inActiveList

[Exclusion] General port Scan (Vertical)

SourceAddress

×

Если не

inActiveList

[Detection] General port Scan (Vertical)

SourceAddress

×

Очистить

Действия

> На первом срабатывании правила

> На последующих срабатываниях правила

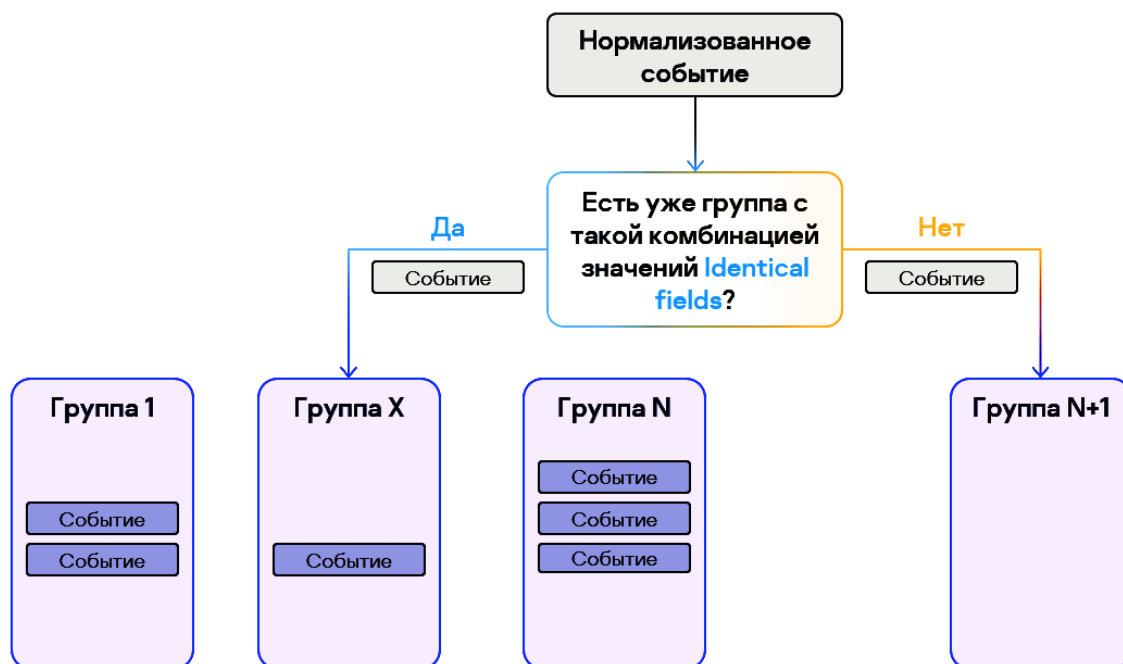
> На каждом срабатывании правила

> По истечении времени жизни контейнера

Другие подходящие примеры для стандартных правил:

- просто много обращений к опасным URL: с разных компьютеров и к разным URL
- много обращений к одному и тому же опасному URL с разных компьютеров
- много обращений к опасным URL, в том числе разным, с одного компьютера
- много обращений с одного и того же компьютера к одному и тому же опасному URL

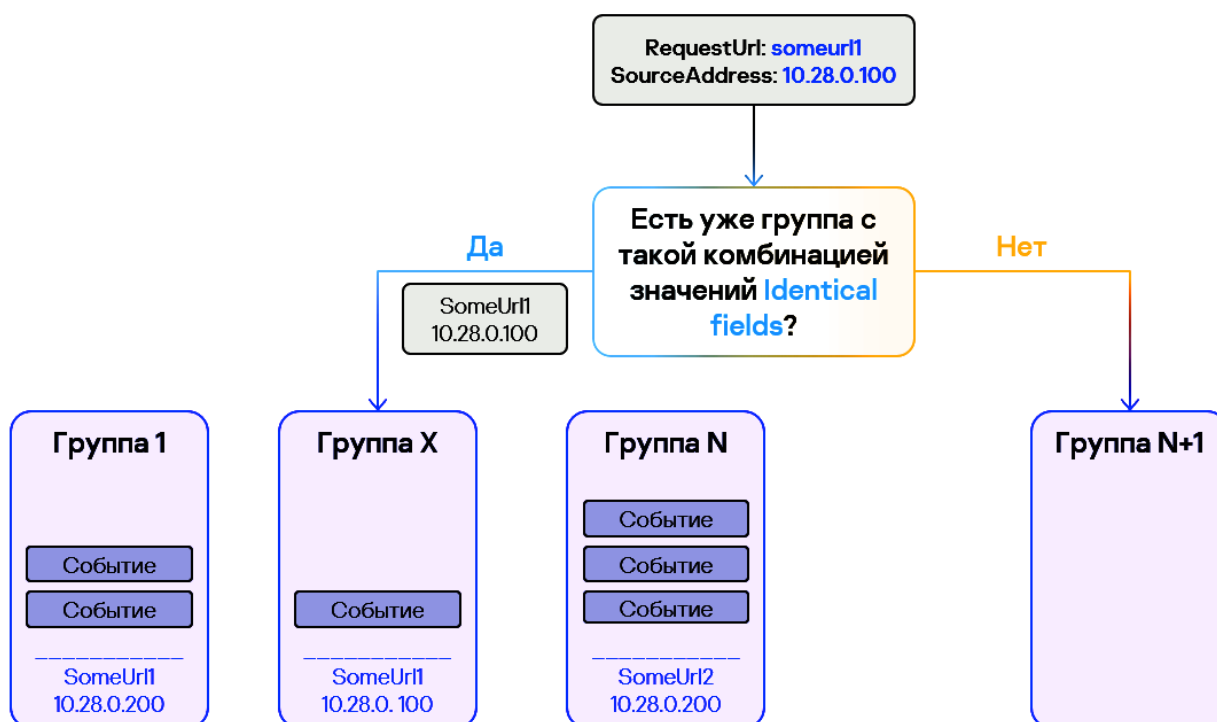
Стандартные правила разбивают все анализируемые события на группы (так называемые «корзины», buckets) с совпадающими значениями полей, перечисленных в параметре **Группируемые поля (Identical fields)** и затем обрабатывает каждую группу независимо от других. Критерии срабатывания применяются отдельно в каждой такой группе. Состояние всех групп хранится в памяти коррелятора.



Бакет (Окно корреляции):

1. Бакет открывается на событие из любого селектора, не важно в каком они порядке в правиле, порядок проверяется после наполнения бакета!
2. Для каждого набора Identical Fields создается свой бакет.
3. Когда событие подпадает под селектор, коррелятор смотрит, есть ли уже бакет с нужным набором полей Identical Fields, если нет - создает, если есть - событие отправляется в существующий.
4. Когда под селектор с Unique fields подпадает событие, то проверяется, есть ли уже в бакете события с таким же набором значений для Unique Fields, если есть, то событие не учитывается.

Пример для Identical Fields с полями RequestUrl и SourceAddress:



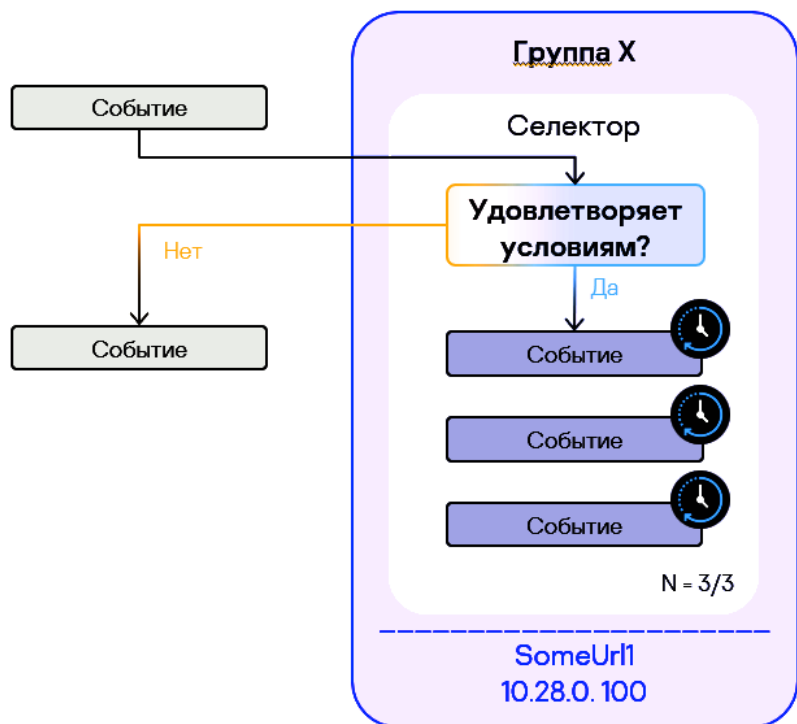
В более общем смысле параметр **Время жизни контейнера (Window)** определяет время жизни группы. Принцип такой:

- identical fields определяет, на какие группы разбивать события
- селекторы определяют, какие события будут включены в группы (если событие не соответствует ни одному селектору, оно не попадет ни в одну группу)
- если событие соответствует селектору и уже есть группа с таким же набором значений идентичных полей, как в событии, событие добавляется в эту группу
- если событие соответствует селектору, но его значения идентичных полей не соответствуют ни одной группе, создается новая группа с временем жизни, заданным параметром Window
- по истечении времени жизни группы, группа удаляется
- если позже поступает новое событие с набором значений идентичных параметров группы, которой больше нет, создается новая группа и отсчет времени жизни

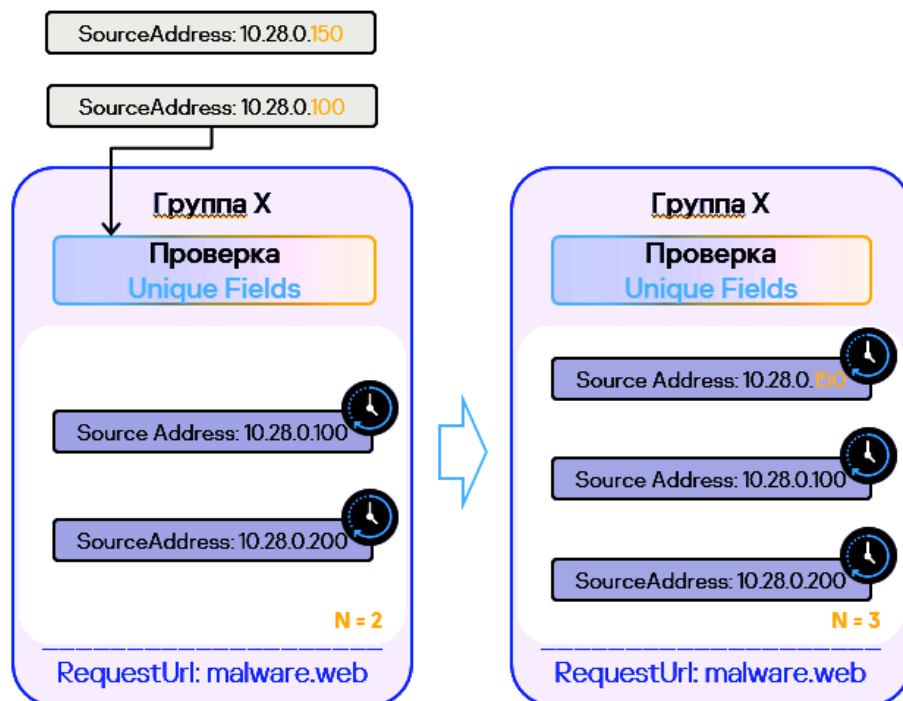
начинается заново

Т.е. все свое время жизни (или окно наблюдения) группа накапливает события, после чего удаляется, и накопление событий может начаться заново. Для каждой комбинации значений идентичных полей это происходит параллельно и независимо.

Правило срабатывает, если группа за время жизни накапливает число событий, указанное в параметре **Порог срабатывания (Threshold)** селектора.



В общих настройках стандартного правила есть еще параметр **Уникальные поля (Unique fields)**. Он не является обязательным, но он позволяет считать в группах только события с уникальными значениями выбранных полей. При добавлении событий в группу правило будет сравнивать значение уникальных полей нового события со значениями уникальных полей событий, которые уже есть в группе. Если комбинация значений уникальных полей нового событий уже встречается у одного из событий в группе, новое событие отбрасывается и в группу не попадает.



Возможные действия (допускается указать одно или более) правила:

- On first threshold — создавать корреляционное событие только после первого превышения порога, а двукратное, трехкратное и т.д. превышение порога за время жизни группы игнорировать. Например, если при пороге 3 за 30 секунд группа накопит 10 событий, корреляционное событие все равно будет одно - после третьего накопленного события
- On every threshold — создавать корреляционное событие после каждого превышения порога за время жизни группы. Если группа накопит 10 событий при пороге 3, будет создано 3 корреляционных события: после 3-го, 6-го и 9-го события в группе
- On subsequent threshold — создавать корреляционные события при всех превышениях порога, кроме первого. Например, при пороге 3 после 6-го, 9-го и т.д. события. Так можно по-разному реагировать на первое переполнение порога и последующие. Например, можно настроить отправлять на вход коррелятора только первое корреляционное событие, но в хранилище писать все. Или пополнять активный список только данными из первого события, а при последующих превышениях порога этого не делать, так как в последующих событиях нет новых артефактов для списка
- On timeout — в стандартных правилах есть еще возможность настройки действий по окончании времени жизни группы. Это действие используется в связке с опцией **Recovery (Обнуление)** в настройках селектора, в каких случаях это уместно и как именно это работает рассматривается ниже. Обнуляющие селекторы можно использовать и не только с действием onTimeout.

Можно также использовать **несколько селекторов**. Например, несколько неудачных попыток брутфорса (ловится на основе сработки другого правила корреляции) и успешный вход. В общем случае правило, в котором задано несколько селекторов, срабатывает при

одновременном превышении порогов во всех селекторах.

Пример правила с несколькими селекторами:

ОбщиеСелекторыДействия

Название

[Linux] обнаружен успешный брутфорс

Тенант

test2

Тип

standard

Группирующие поля

+ Добавить полеDeviceHostNameDestinationUserNameСбросить

Уникальные поля

+ Добавить поле

Частота срабатываний

0

Время жизни контейнера, сек.

900

Политика хранения базовых событий

all

Уровень важности

Высокий

Сортировать по

Описание

Правило обнаруживает события успешного входа в систему после большого количества неудачных попыток авторизации от 10.

ОбщиеСелекторыДействия

Селектор "BruteForce"

Параметры

Локальные переменные

Название

BruteForce

Порог срабатывания селектора (количество событий)

1

Фильтр

Создать

Сохранить фильтр

Условия

И+ Добавить условие+ Добавить группу+ Добавить фильтр

Если

поле события

Name

=

константа

Обнаружена попытка брутфорса

Если

поле события

Type

=

константа

3

Обучение

Удалить селектор

Селектор "Login_Success"

Параметры

Локальные переменные

Название

Login_Success

Порог срабатывания селектора (количество событий)

1

Фильтр

Создать

Сохранить фильтр

Условия

И+ Добавить условие+ Добавить группу+ Добавить фильтр

[Filter] Linux AuditD base events

Если

поле события

DeviceEventClassID

=

список

USER_LOGIN, USER_AUTH

Если не

поле события

DestinationUserName

=

список

splunk, zabbix, (unknown)

Если

поле события

EventOutcome

=

константа

success

Обучение

Удалить селектор

+ Добавить селектор

Действия

На первом срабатывании правила

На последующих срабатываниях правила

На каждом срабатывании правила

Отправить событие на дальнейшую обработку

Отправить событие снова в коррелятор

Не создавать alert

Обогащение

+ Добавить обогащение

Обновление активных листов

+ Добавить действие с активным листом

Изменение категорий

+ Добавить категоризацию

По истечении времени жизни контейнера

Можно также **рекавери правило**. Например, когда событие типа «Вредоносное ПО удалено» не обнаружено в течение 5 минут после получения события «Вредоносное ПО обнаружено».

ОбщиеСелекторыДействия

Название

[KSC] обнаружено ВПО, невозможно удалить

Тенант

test2

Тип

standard

Группирующие поля

+ Добавить полеDeviceCustomString1DeviceCustomStringLabelDeviceProductDeviceVendorFilenameFilepathDestinationHostNameСбросить

Уникальные поля

+ Добавить поле

Частота срабатываний

0

Время жизни контейнера, сек.

300

Политика хранения базовых событий

all

Уровень важности

Высокий

Сортировать по

Описание

Неудачная попытка лечения или удаления вредоносного объекта. Срабатывает когда событие типа «Вредоносное ПО удалено» не обнаружено в течение 5 минут после получения события «Вредоносное ПО обнаружено».

ОбщиеСелекторыДействия

Селектор "KSC Virus Found"

Параметры

Локальные переменные

Название

KSC Virus Found

Порог срабатывания селектора (количество событий)

1

Фильтр

[Filter] KSC Virus Found

Сохранить фильтр

Условия

И+ Добавить условие+ Добавить группу+ Добавить фильтр

[Filter] KSC Base Events

Если не

поле события

DeviceCustomString1

startsWith

константа

not-found

Если

поле события

DeviceEventClassID

startsWith

константа

SMI_EVT_VIRUS_FOUND

Обучение

Удалить селектор

Селектор "KSC Virus Deleted"

Параметры

Локальные переменные

Название

KSC Virus Deleted

Порог срабатывания селектора (количество событий)

1

Фильтр

[Filter] KSC Object Deleted

Сохранить фильтр

Условия

И+ Добавить условие+ Добавить группу+ Добавить фильтр

Если

поле события

DeviceEventClassID

=

константа

SMI_EVT_OBJECT_DELETED

[Filter] KSE Base Events

Обучение

Удалить селектор

+ Добавить селектор

Действия

На первом срабатывании правила

На последующих срабатываниях правила

На каждом срабатывании правила

По истечении времени жизни контейнера

Отправить событие на дальнейшую обработку

Отправить событие снова в коррелятор

Не создавать alert

Обогащение №1

Тип источника данных: шаблон

Шаблон: ВПО было обнаружено и не было удалено на хосте {{DestinationHostName}}.

Целевое поле: Message

Отладка: Выключено

Удалить обогащение

+ Добавить обогащение

Recovery селектор (Обнуление):

1. Бакет открывается только на событие из обычного селектора, на событие из recovery-селектора бакет не открывается никогда!
2. Место нахождения селектора с recovery не имеет значения, как только в бакет попадут все нужные recovery-события бакет будет закрыт!
3. На recovery-селектор не влияет настройка фильтра Order By.
4. Если нужно, чтобы произошло событие А, и не произошло событие Б, при этом событие Б может произойти раньше А, нужно использовать активные листы, т.к. с помощью recovery-селектора такой логики не достичь (см п.1).

Revision #11

Created 9 August 2023 12:44:28 by Boris RZR

Updated 24 December 2024 09:40:05 by Boris RZR