Приемы парсинга событий

Парсинг нестандартной даты

Дата Время

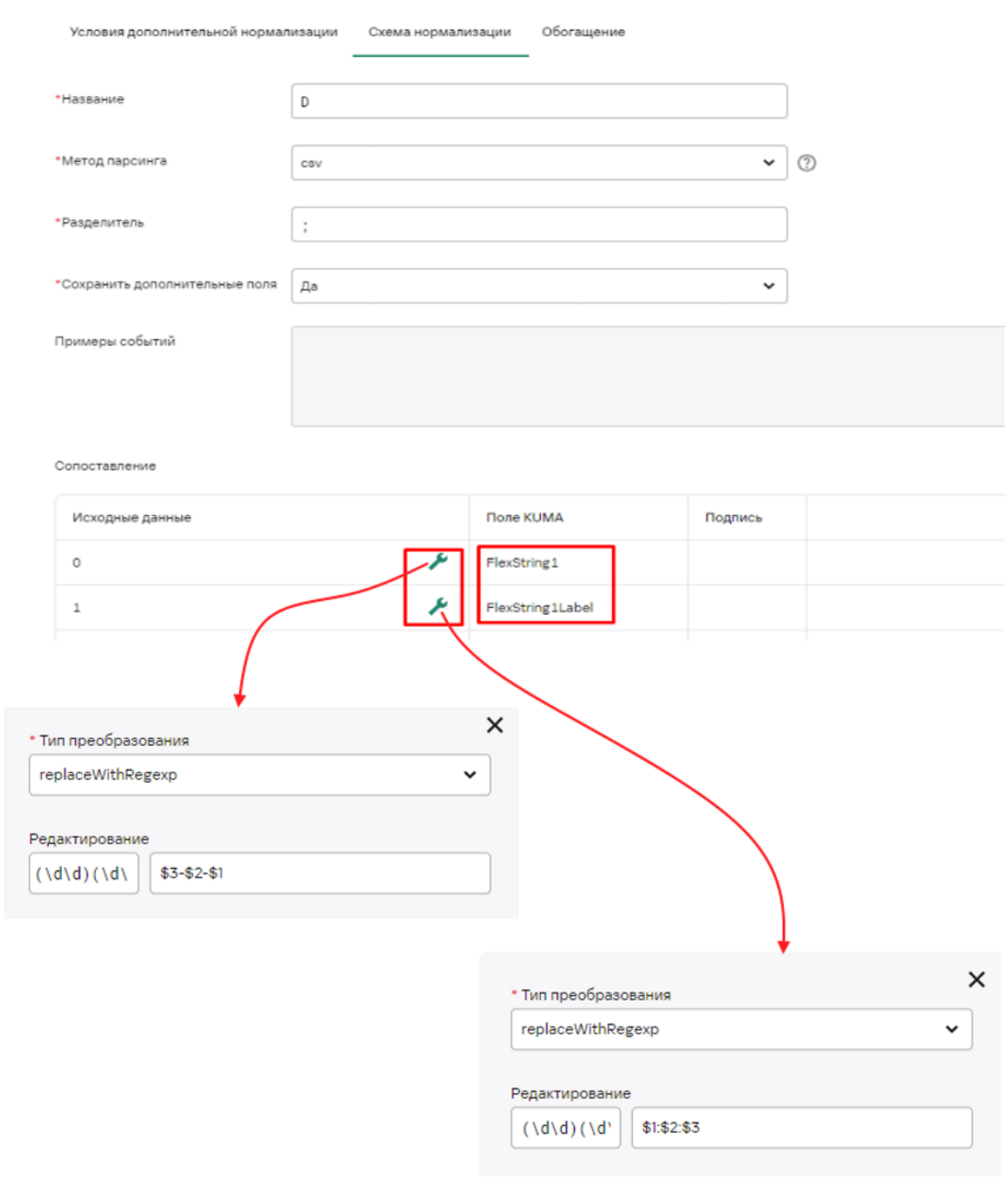
#D#
220523;080050;5522.5957;N;08605.0047;E;0;65;142.400000;22;NA;NA;NA;NA;NA;flg:1:7,mileage:2:1097.000000,timp:1:0,pwr_ext:2:13.700000,battery:1:41,sum_acc:2:1.140000,amtr_x:1:2,wln_accel_max:2:0.020000,wln_brk_max:2:0.000000,amtr_y:1:0,wln_crn_max:2:0.000000,amtr_z:1:0,hours_koef:1:16777216,tlls1:1:0,clls1:1:0,flls1:1:3,mid:1:1,cmd:1:149

Нужно привести к такому виду:

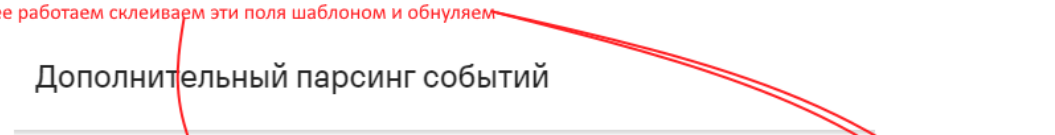
ISO8601 (Syslog timestamp)
[0-9]{4}-[0-9]{2}-[0-9]{2}T[0-9]{2}:[0-9]{2}:[0-9]{2}(\.[0-9]+)?([zZ]|([\+-])([01]\d|2[0-3]):?([0-5]\d)?)

2020-03-12T13:34:56.123Z INFO [org.example.Class]: This is a #simple #logline containing a 'value'.

Как выглядит парсер:



Далее работаем склеиваем эти поля шаблоном и обнуляем



Ветвление событий от beats в зависимости от input типа

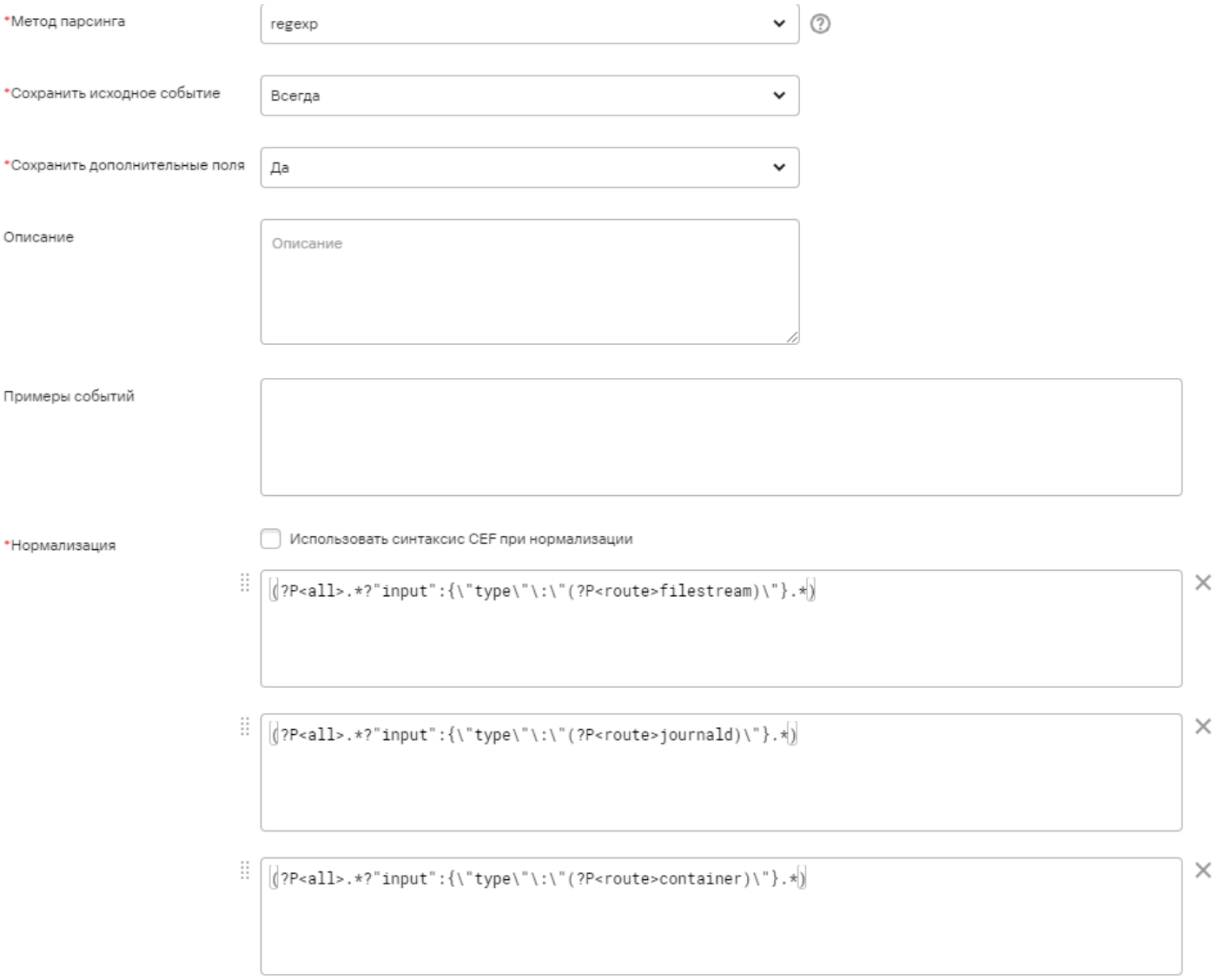
Даны следующие типы событий (содержимое тестового сообщения сокращено для лучшего понимания):

```
{"tags":["beats_input_raw_event"],"input":{"type":"filestream"}}
{"message":"I0130 14:38:47.090079 1837403 utils.go:187] ID: 544472 GRPC response:
{}","input":{"type":"container"}}
{"journal":{"system":"true"},"tags":["beats_input_codec_plain_applied"],"input":{"type":"journald"}}
{"input":{"type":"journald"},"journal":{"system":"true"},"tags":["beats_input_codec_plain_applied"]}
{"journal":{"system":"true"},"input":{"type":"journald"},"tags":["beats_input_codec_plain_applied"]}
```

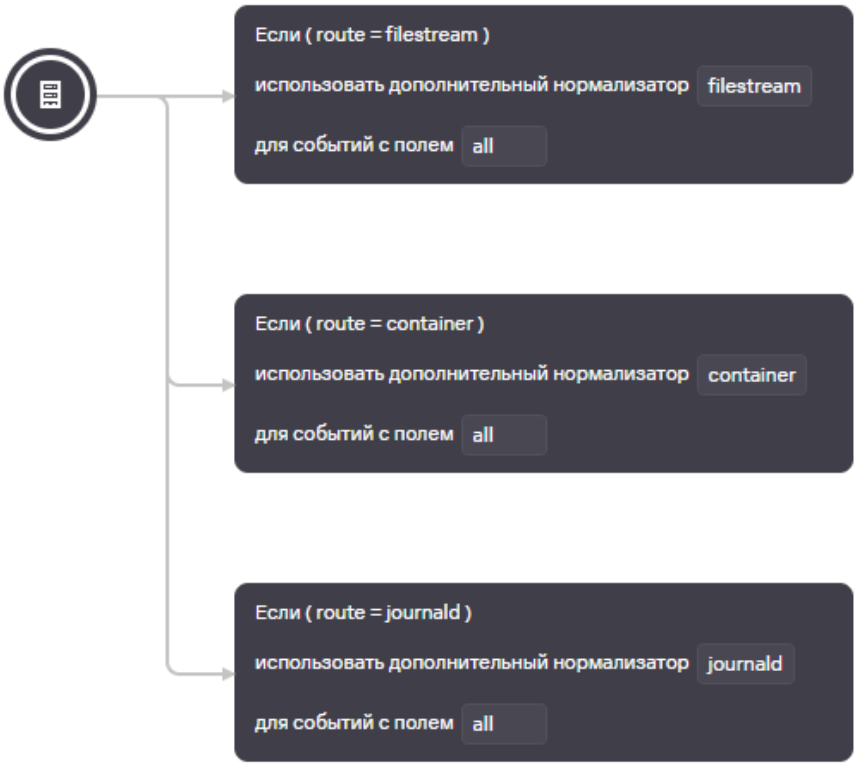
Необходимо в парсинге разветлять (тк у каждого типа свой набор полей) парсинг в зависимости от типа input поля, мы имеем три типа в данном примере:

- "input":{"type":"container"}
- "input":{"type":"journald"}
- "input":{"type":"filestream"}

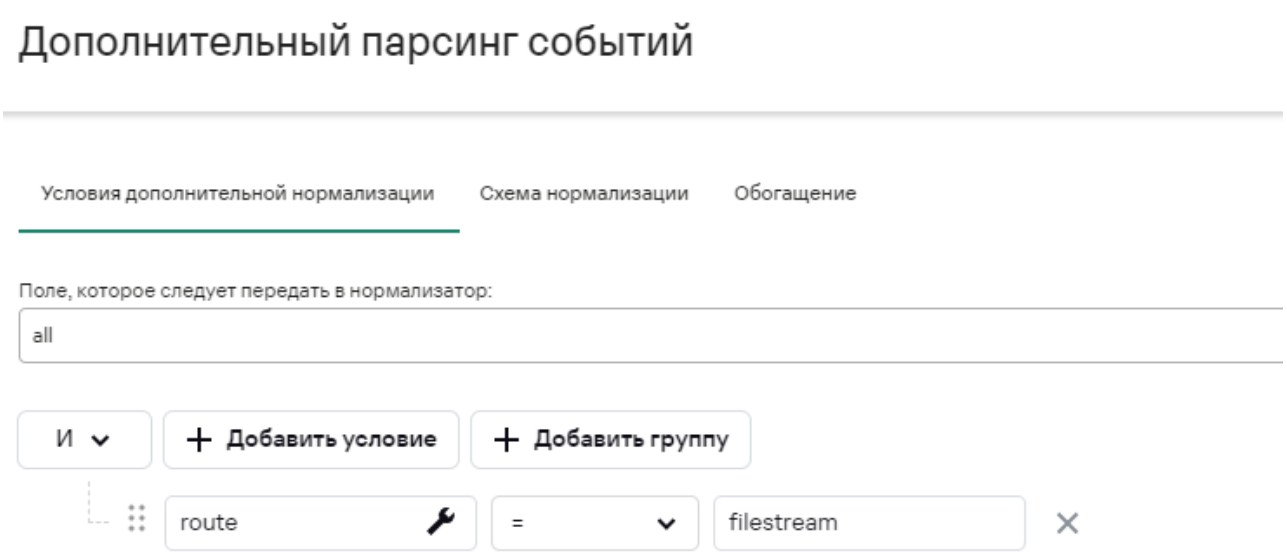
Причем, поле input может находиться как в начале, так и в середине, и в конце сообщения. Поэтому для ветвления в первом шаге парсинга будут использоваться регулярные выражения:



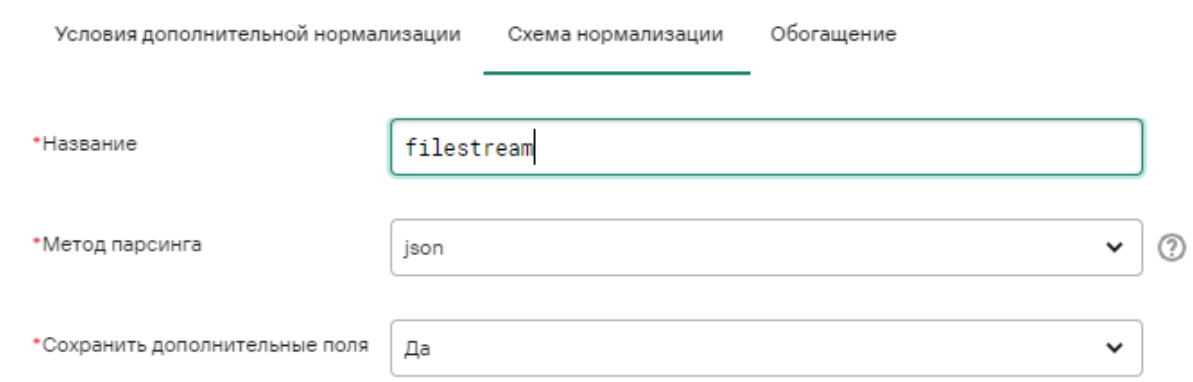
Поле из regex с наименованием route будет использоваться для маршрутизации по условию в нужный парсер, поле all необходимо для передачи полного содержимого в подпарсер. Структура парсера выглядит следующим образом:



Рассмотрим один подпарсер, например, filestream:



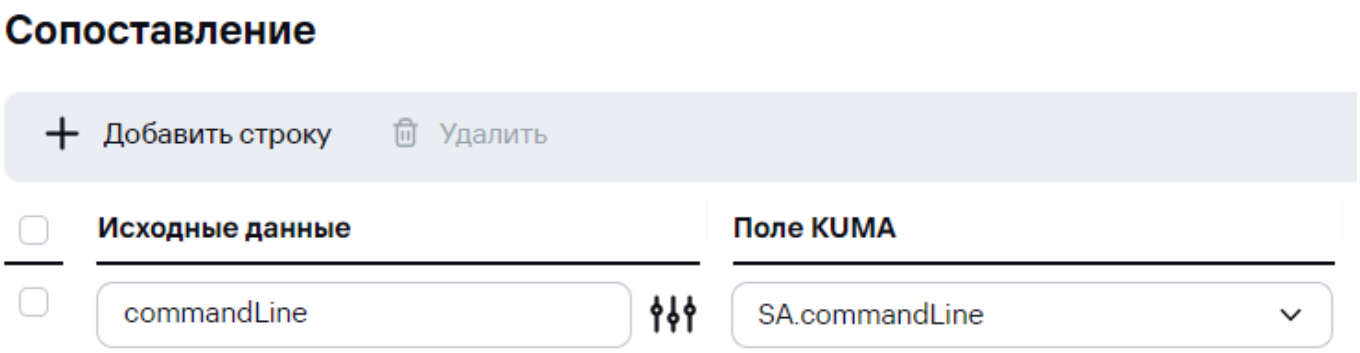
Тк общая структура сообщения формата JSON, используется соответсвующий коробочный парсер:



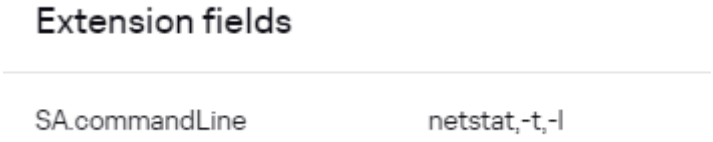
Парсинг массивов

> Актуально для KUMA 3.0+

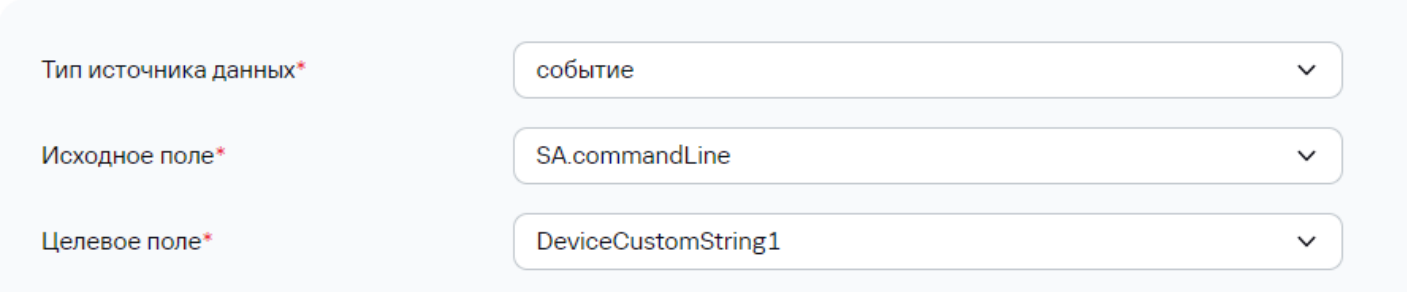
В KUMA 3.0.2 появилась возможность создания кастомных полей типа "массив" (SA, NA, FA), доступные для методов парсинг JSON и KV. Чтобы записать массив в дополнительное поле, достаточно его указать в маппинге:



В событии это будет выглядеть следующим образом:



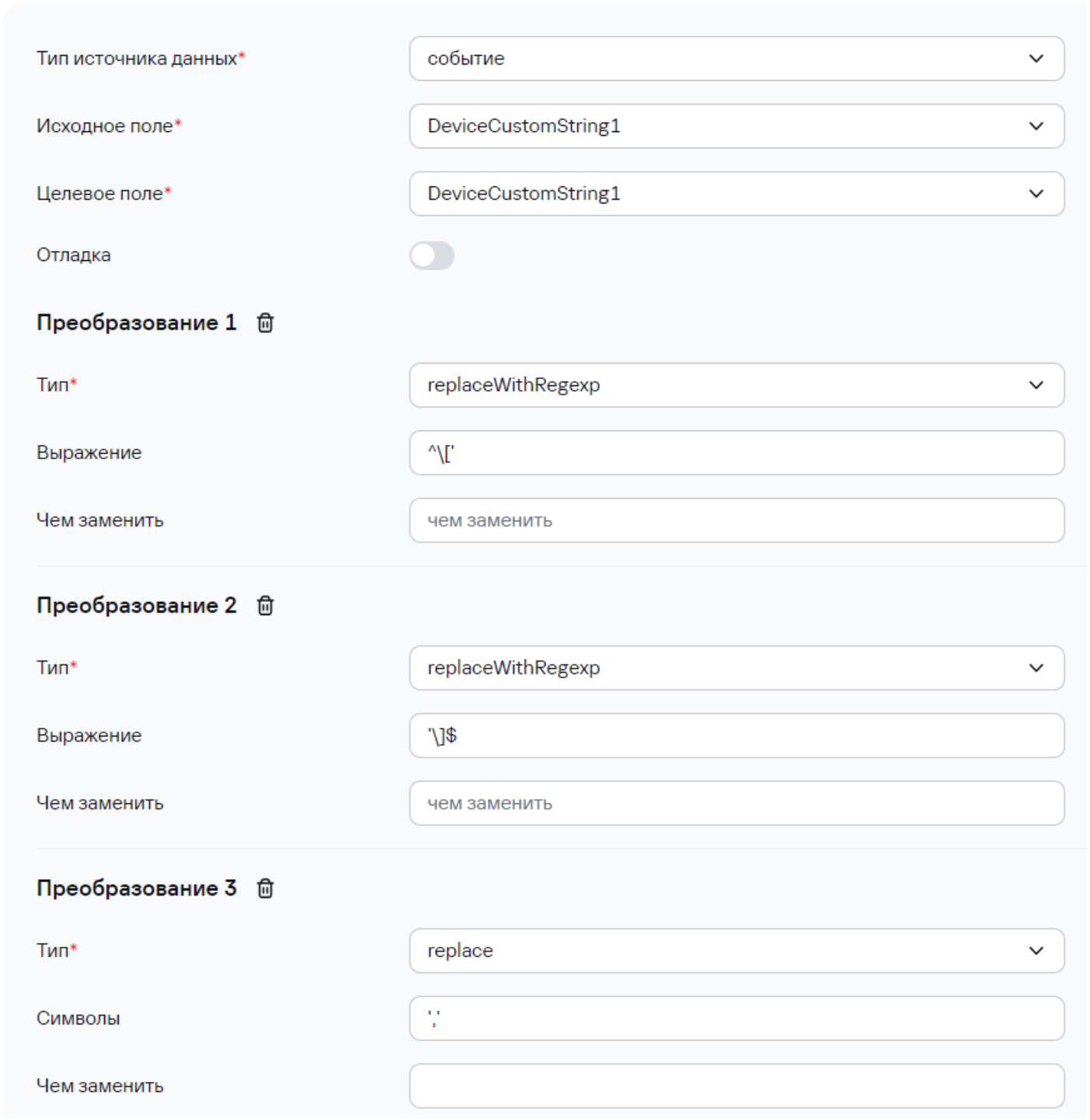
Если с массивом в таком случае работать не удобно и нужно все элементы из массива "склеить" через делиметр и записать в отдельное поле, можно воспользоваться обогащением. Для этого сначала массив мапится на строковое поле:



В таком случае в событии данное поле будет представлять собой массив переведенный в строку:

DeviceCustomString1	['netstat','-t','-l']

Чтобы привести ее в более "приятный" вид можно выполнить следующие преобразования:



После этого в DeviceCustomString1 будут записаны все элементы массива через выбранный в последнем (3) преобразовании делиметр (в данном примере это "пробел"):

DeviceCustomString1	netstat -t -l

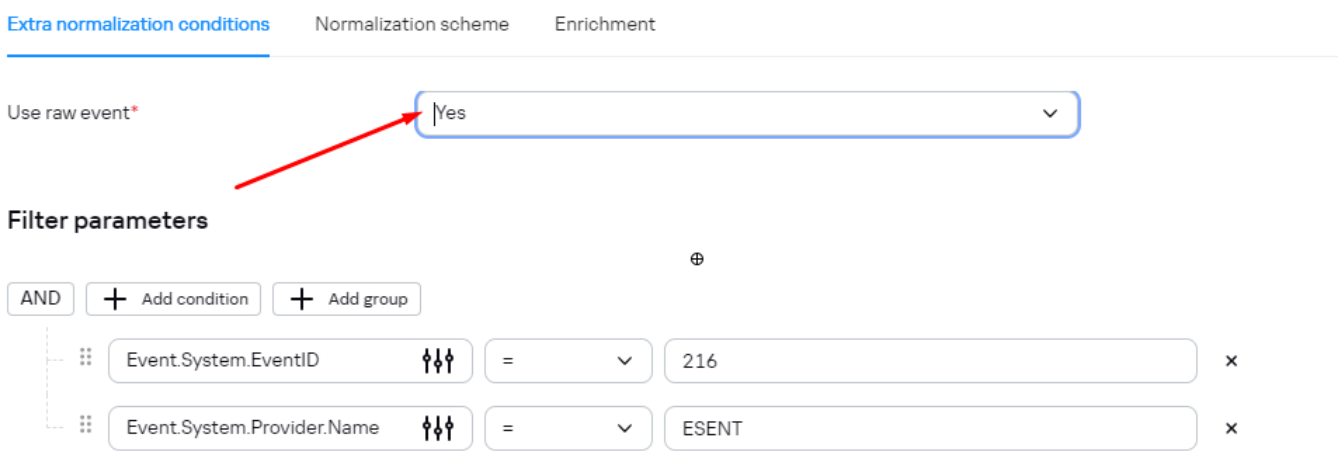
Передача сырого события в экстранормализатор, для доступа к элементам массива

> Актуально для KUMA 3.0+

Для передачи «сырого» события в экстра-нормализатор необходимо:

- открыть нормализатор событий;
- перейти в меню «Условия дополнительной нормализации»;
- активировать параметр «Использовать сырое событие».

> По умолчанию параметр «Использовать сырое событие» не активен.



Рекомендуется активировать параметр «Использовать сырое событие» в нормализаторах типа **«xml», «json»**.

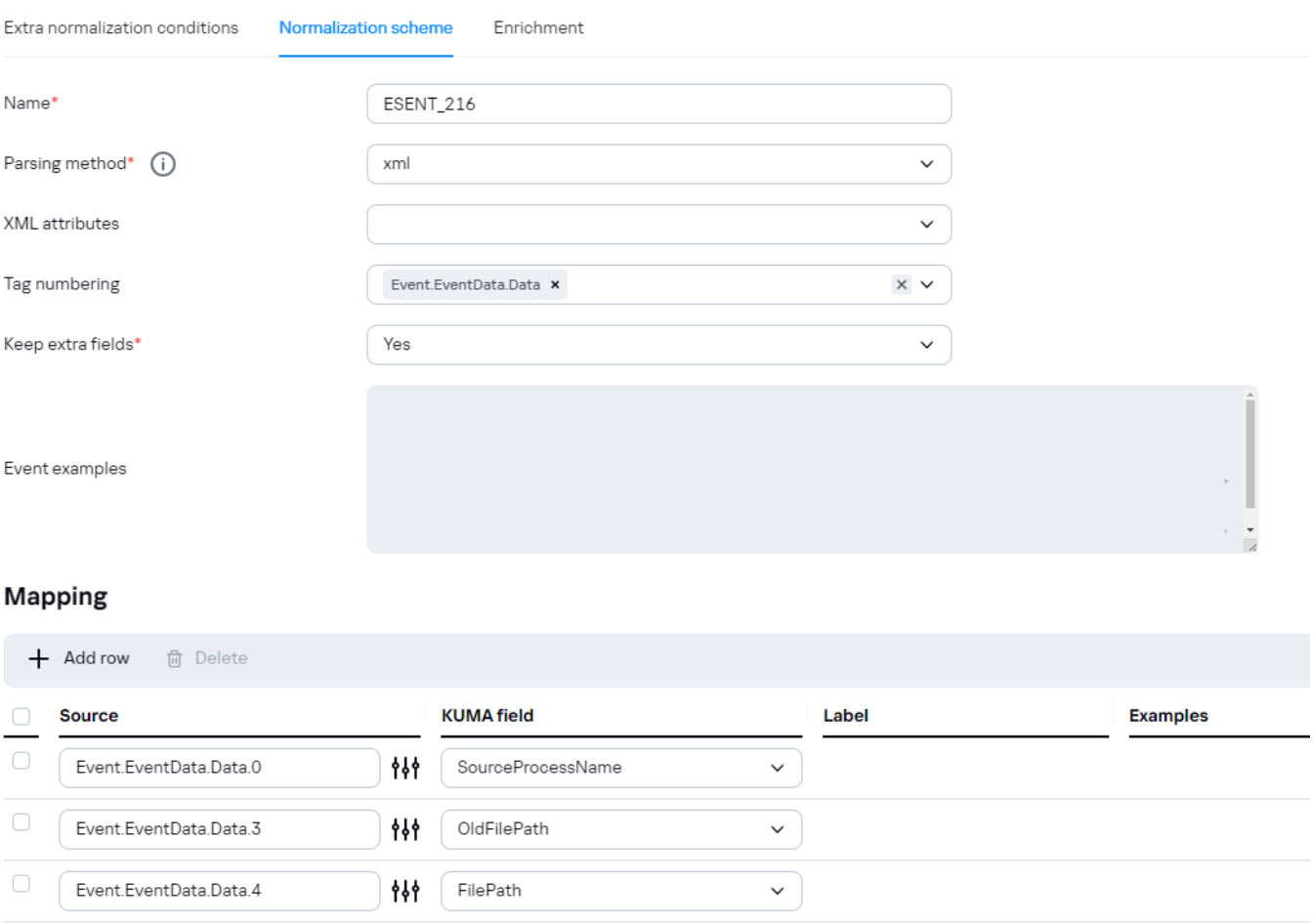
Для передачи «сырого» события в экстра-нормализатор второго, третьего и более глубоких уровней вложенности необходимо последовательно включить параметра «Использовать сырое событие» в каждом экстра-нормализаторе по пути следования события в целевой экстра-нормализатор и непосредственно в целевом экстра-нормализаторе.

В качестве примера работы данной функции вы можете обратиться к нормализатору Microsoft Products для KUMA 3.0.1: параметр «Использовать сырое событие» включен последовательно в экстра-нормализаторах «AD FS» и «424».

В качестве примера, событие:

<Event xmlns='http://schemas.microsoft.com/win/2004/08/events/event'><System><Provider Name='ESENT'/><EventID Qualifiers='0'>216</EventID><Level>4</Level><Task>3</Task><Keywords>0x80000000000000</Keywords><TimeCreated SystemTime='2024-01-20T20:06:07.144730300Z'/><EventRecordID>870234</EventRecordID><Channel>Application</Channel><Computer>COMPANY.COM</Computer><Security/></System><EventData><Data>lsass</Data><Data>724,R,98</Data><Data></Data><Data>C:\Windows\NTDS\ntds.dit</Data><Data>\\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy50\Windows\NTDS\ntds.dit</Data></EventData></Event>

При парсинге ID события 216:



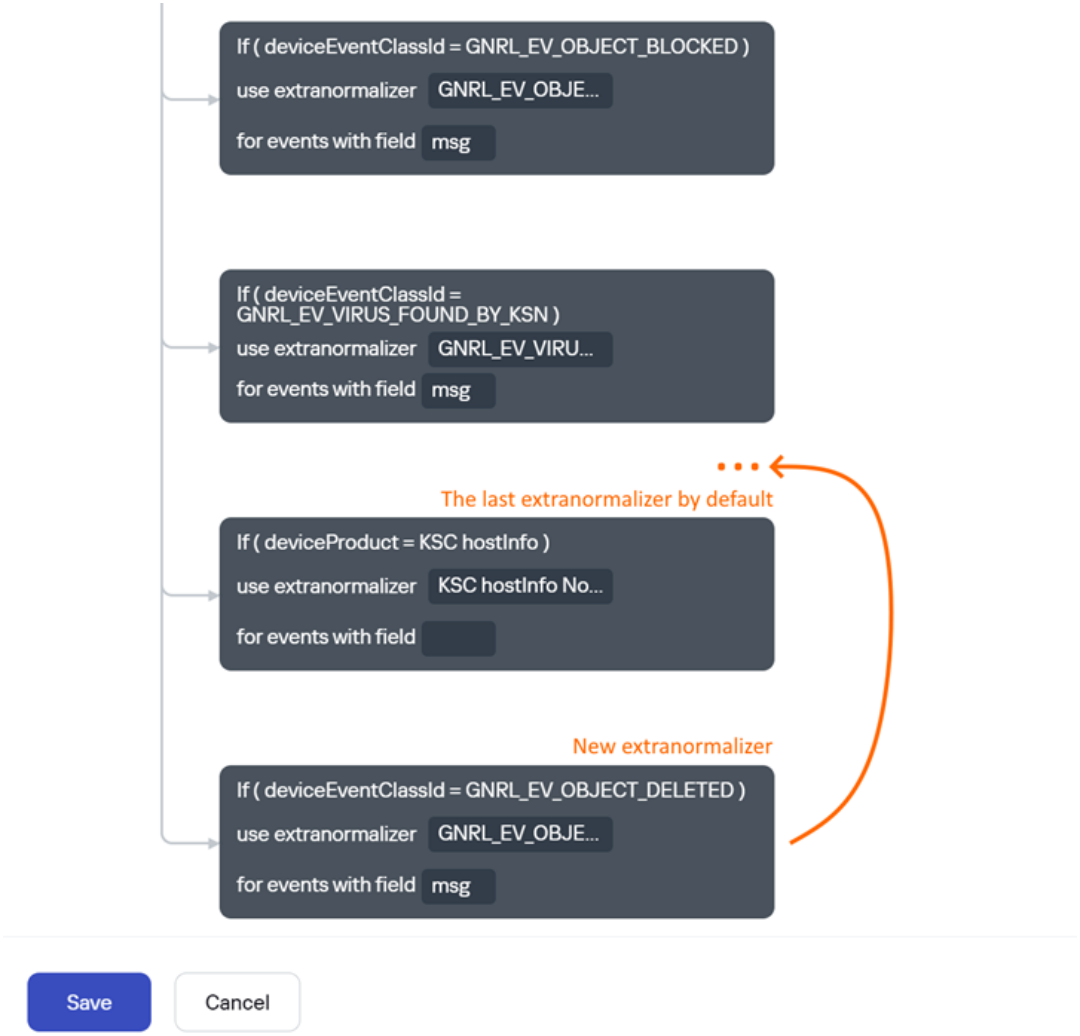
Будет корректно разбираться:

SourceProcessName	lsass
Service	BorisTest (tcp/5577)
ExternalID	1276974
FilePath	\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy47 \Windows\NTDS\ntds.dit
OldFilePath	C:\Windows\NTDS\ntds.dit
Type	Base

Смена порядка следования экстранормализаторов

> Материал был предостален пользователем комьюнити ❤

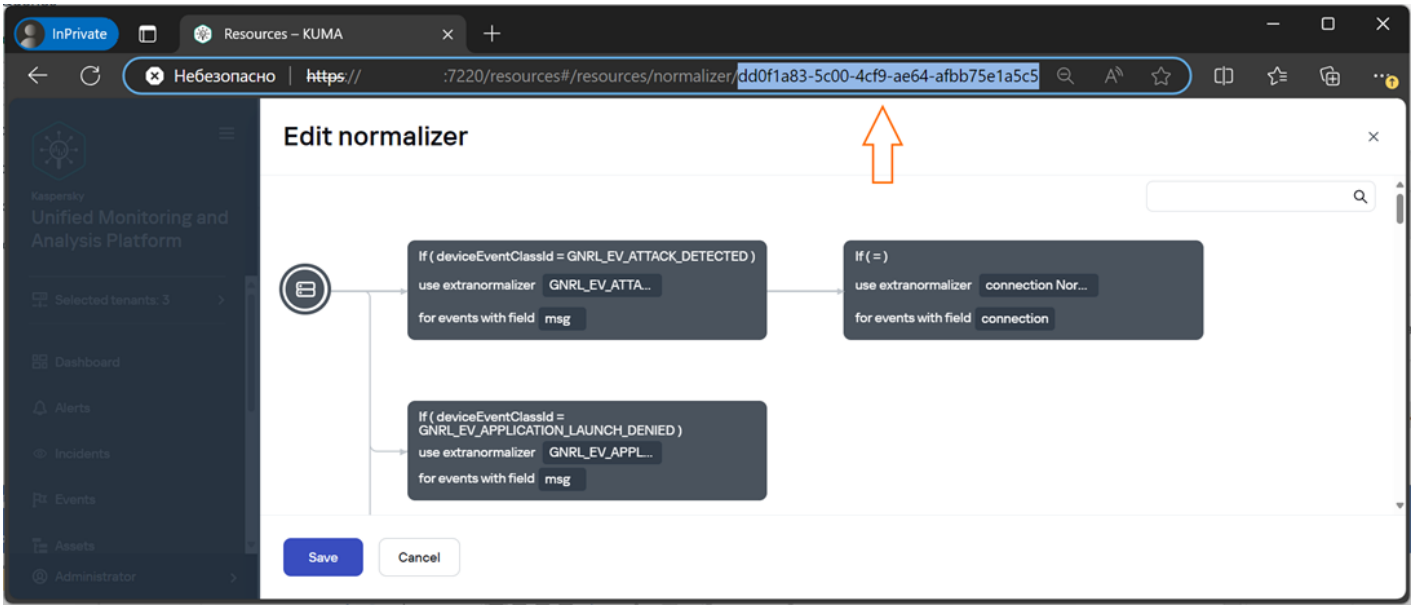
По умолчанию в GUI KUMA отсутствует возможность перемещать экстранормализаторы внутри правила нормализации и менять их местами. Однако, в ряде случаев данная операция всё же требуется. Например, когда нужно добавить блок с экстранормализатором выше уже существующих, так как они проверяются последовательно. Через веб-интерфейс это сделать проблематично, т.к. потребуется удаление и пересоздание заново всех блоков экстранормализаторов идущих ниже.



Ниже описан workaround, который позволяет получить нужное правило нормализации в виде JSON файла из MongoDB и отредактировать его, задав нужную последовательность экстранормализаторов.

> **Необходимые меры предосторожности**:
> 1. Все действия по подготовке нужного правила настоятельно рекомендуется выполнять на тестовом стенде (не на продуктовой инсталляции), так как предполагается прямой доступ и запись данных в MongoDB (основную базу настроек KUMA). Нельзя исключать риск нарушения работы инсталляции KUMA из-за возможных ошибок.
> 2. Предварительно рекомендуется сделать выгрузку контента и бэкап самой базы: средствами kuma tools (old), по API или через утилиту mongodump.

1. Разместить на стенде KUMA правило нормализации, которое будет подлежать редактированию. Открыть его в браузере и скопировать его UUID из строки URL.



> Необходимые утилиты под вашу ОС можно загрузить отсюда: https://www.mongodb.com/try/download/database-tools Документация по утилитам: https://www.mongodb.com/docs/v4.2/reference/program/

2. С помощью встроенной консольной утилиты mongoexport выполнить подключение к базе kuma и экспорт нужного правила нормализации в файл:

```
/opt/kaspersky/kuma/mongodb/bin/mongoexport --db=kuma --collection=resources --query='{"_id": "your_normalizer_id"}' > normalizer.json
```

Пример успешного экспорта:

```
[root@kuma ~]# /opt/kaspersky/kuma/mongodb/bin/mongoexport --db=kuma --collection=resources
 --query='{"_id": "dd0f1a83-5c00-4cf9-ae64-afbb75e1a5c5"}' > normalizer.json
2024-06-20T11:17:45.779+0300    connected to: mongodb://localhost/
2024-06-20T11:17:45.798+0300    exported 1 record
[root@kuma ~]#
```

3. Открыть полученный JSON файл в редакторе, поддерживающим форматирование JSON и работу с объектами (например, Notepad++ с плагином JSTool).
— Сразу поменять uuid в полях "_id", "exportID", "id" на новый. Он должен быть уникальным в рамках всех остальных ресурсов KUMA для успешного импорта правила обратно.
Сгенерировать UUID:
cat /proc/sys/kernel/random/uuid
— Сразу поменять значение поля "name", задав новое название правила или его версию.
— Найти в структуре файла блок "extra", содержащий экстранормализаторы. Развернуть его и выполнить поиск нужного блока экстранормализации который требуется переместить.

normalizer.json

```
{
    "_id": "8ce907b9-98a4-43b9-b432-b924f35eca97",
    "exportID": "8ce907b9-98a4-43b9-b432-b924f35eca97",
    "tenantID": "f02dbc6e-14cd-44f4-9c46-3fe9054bc210",
    "kind": "normalizer",
    "name": "[PRIMER] KSC from SQL Extended",
    "description": "",
    "createdAt": 1718625123883,
    "updatedAt": 1718629908657,
    "folderID": "86cfd4d8-9036-49eb-9d23-e79de6786690",
    "userID": "56021256-4836-4356-8aee-5e1a69208095",
    "deps": null,
    "internal": false,
    "repositoryPackageID": "",
    "packageIntegrationResource": false,
    "payload": {
        "id": "8ce907b9-98a4-43b9-b432-b924f35eca97",
        "name": "[PRIMER] KSC from SQL Extended",
        "kind": "sql",
        "expressions": [""],
        "pairDelimiter": "",
        "kvDelimiter": "",
        "delimiter": "",
        "keepRaw": "always",
        "mapping": [{
        "enrichment": [{
        "extra": [{
        "xmlKeyAttrs": [],
        "xmlArrayKeys": [],
        "samples": {
            "row": null,
            "columnar": null
        },
        "saveExtra": false,
        "regexpGroupsCEF": false
    }
}
```

Change uuid

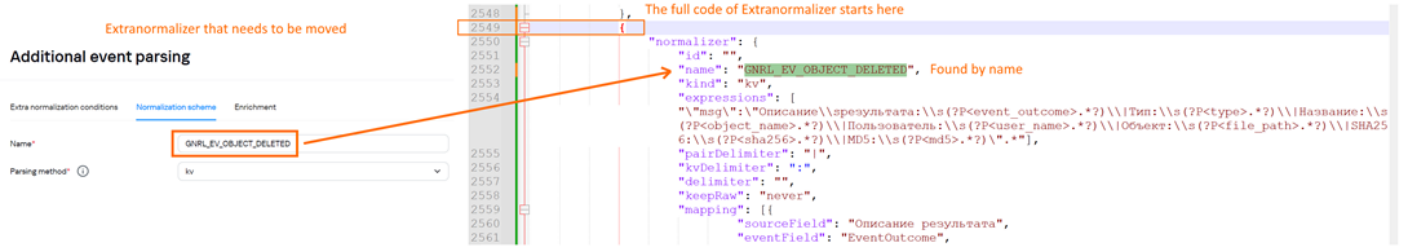
Change name

Change uuid

Change name

Go to extranormalizers

4. В блоке "extra" найти по нужный экстранормализатор, выделить и скопировать его код целиком, ориентируясь на открывающую скобку перед полем "normalizer" и соответствующую закрывающую скобку.



5. Вставить скопированный код в нужное место в блоке "extra". Например, в его начале или между требуемых экстранормализаторов (зависит от нужного вам порядка их следования). Проверить, что все скобки { } на месте.

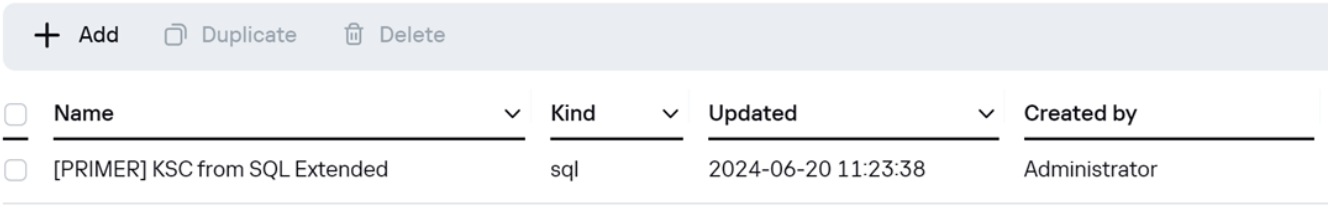
6. Сохранить получившийся JSON файл, перенести его обратно на сервер с KUMA Core и выполнить его импорт в MongoDB:

```
/opt/kaspersky/kuma/mongodb/bin/mongoimport --db kuma --collection resources --file new_normalizer.json
```

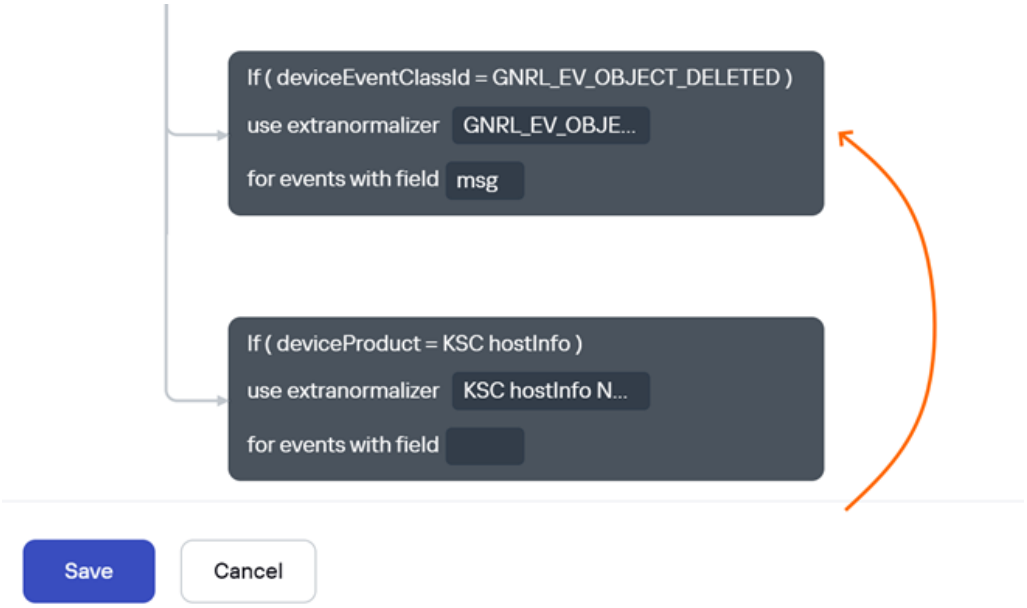
Пример успешного импорта:

```
[root@kuma ~]# /opt/kaspersky/kuma/mongodb/bin/mongoimport --db kuma --collection resources --file new_normalizer.json
2024-06-20T11:20:40.588+0300    connected to: mongodb://localhost/
2024-06-20T11:20:40.604+0300    1 document(s) imported successfully. 0 document(s) failed to import.
[root@kuma ~]#
```

7. Зайти в веб-интерфейс KUMA и проверить наличие отредактированного нормализатора (в том же тенанте и папке, т.к. они не менялись)



Результат: изменён порядок следования экстранормализаторов без их удаления и пересоздания вручную через веб-интерфейс



Альтернативный вариант с VS Code

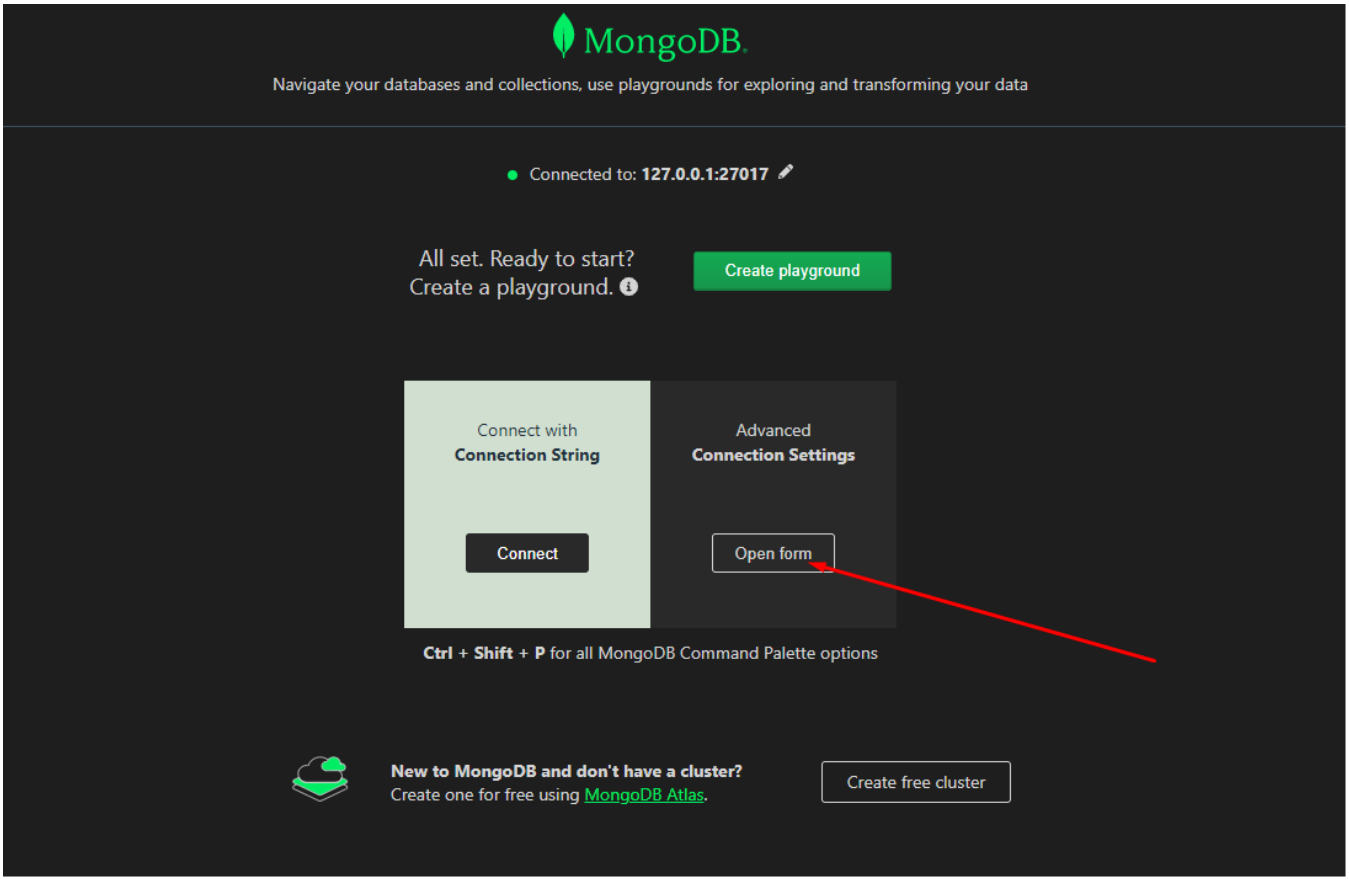
> Потребуется приложение: https://code.visualstudio.com/ и плагин для работы с MongoDB: https://marketplace.visualstudio.com/items?itemName=mongodb.mongodb-vscode

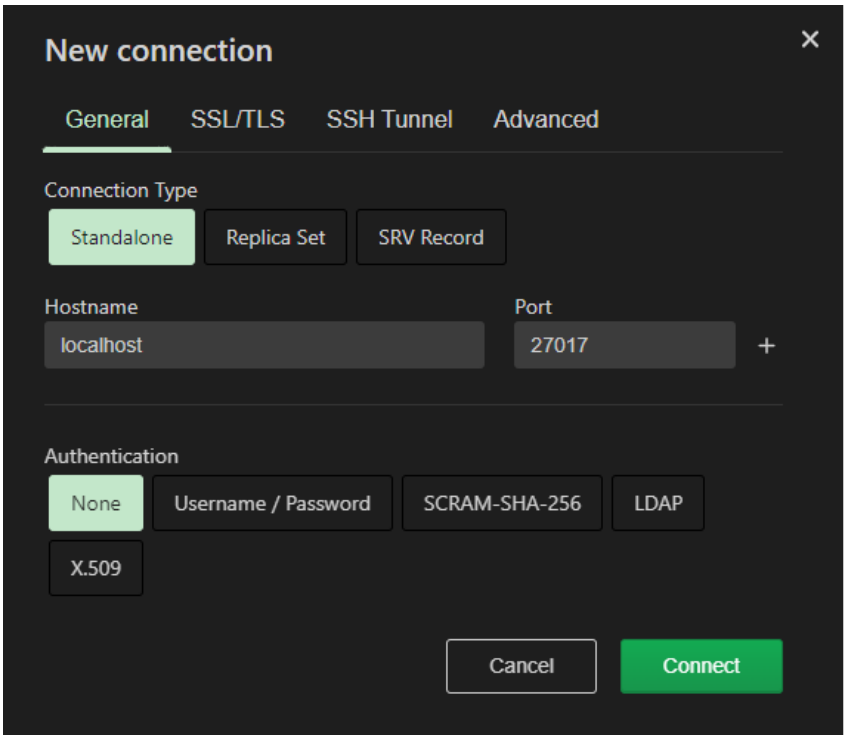
>

> Также можно использовать клиент MongoDB Compass для подключения к MongoDB: https://www.mongodb.com/products/tools/compass

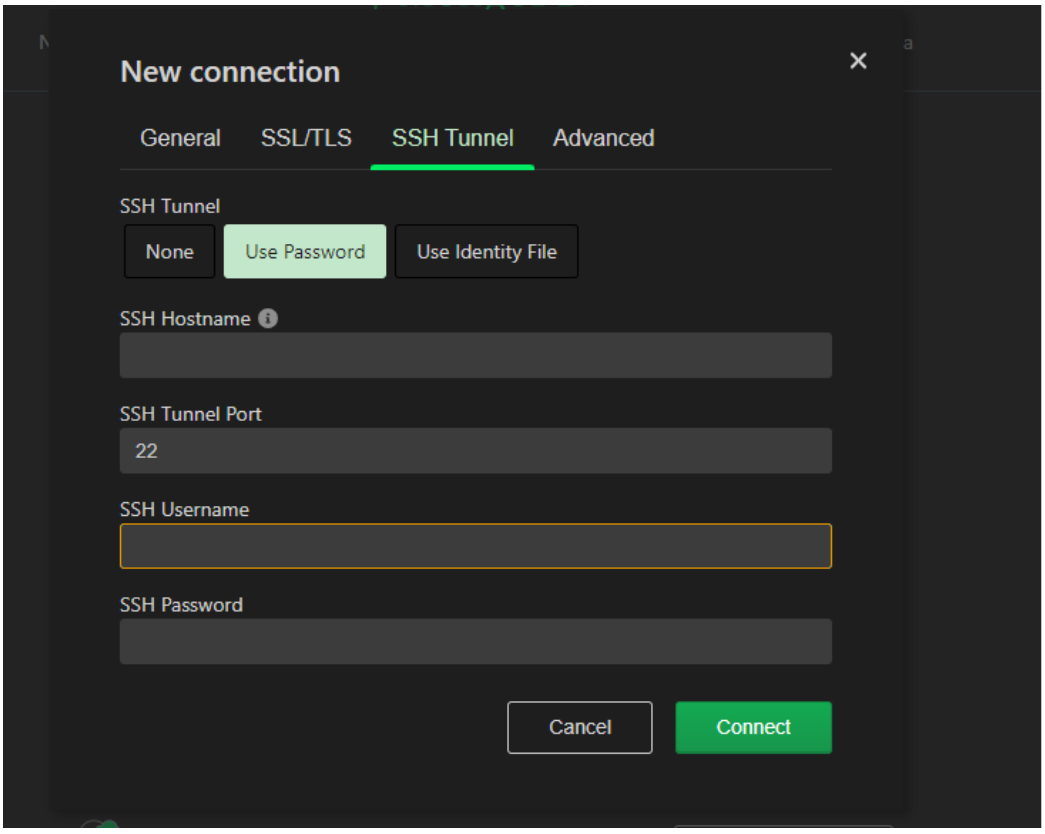
> Рекомендуется работать с копией / дубликатом ресурса, чтобы предотвратить возможные проблемы

1. Подключение к монго через SSH:

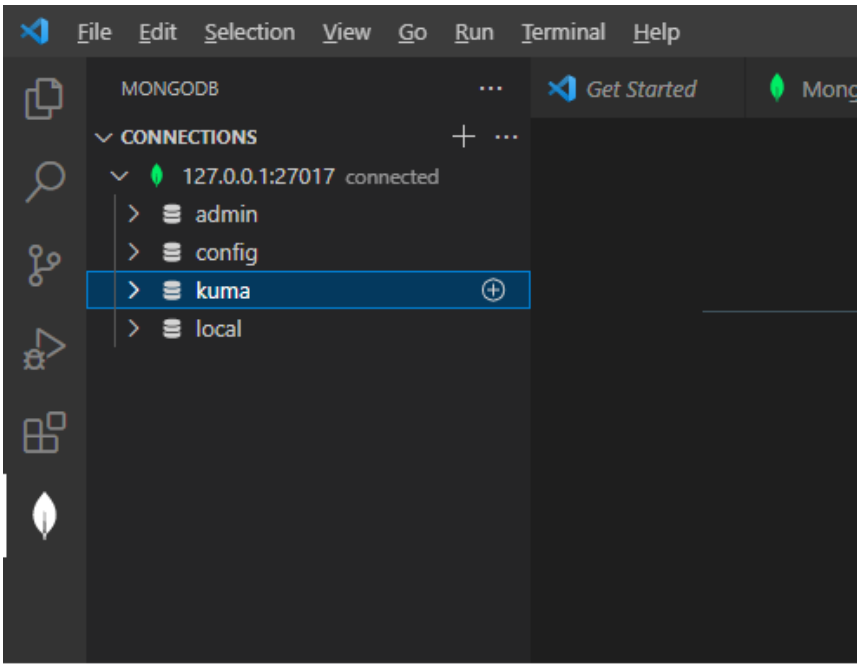




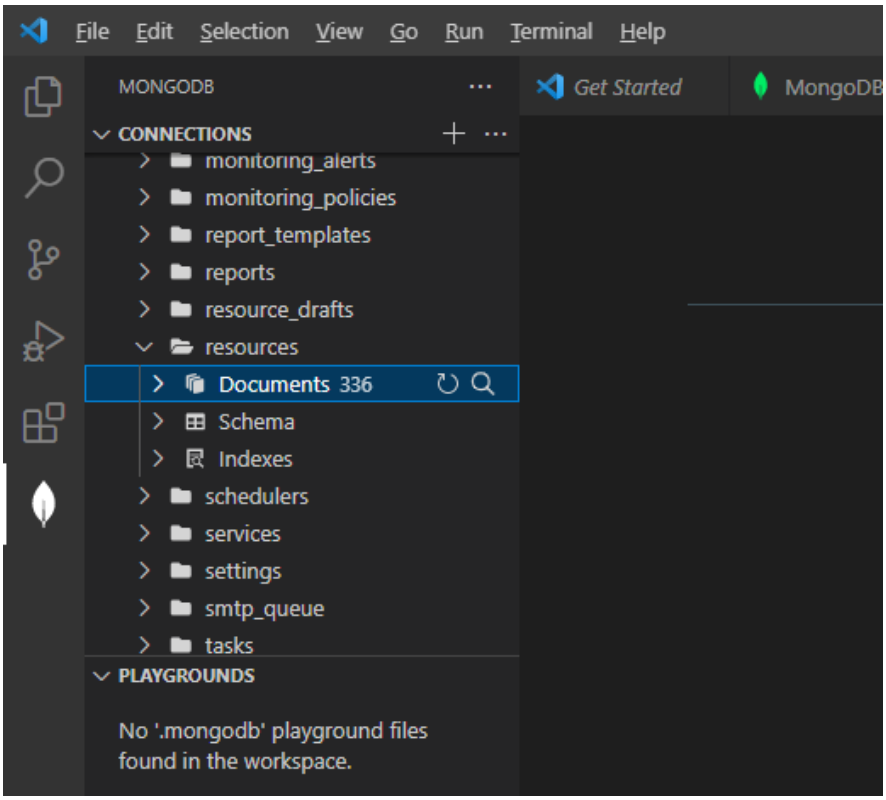
Прописываем адрес, логин и пароль для SSH:



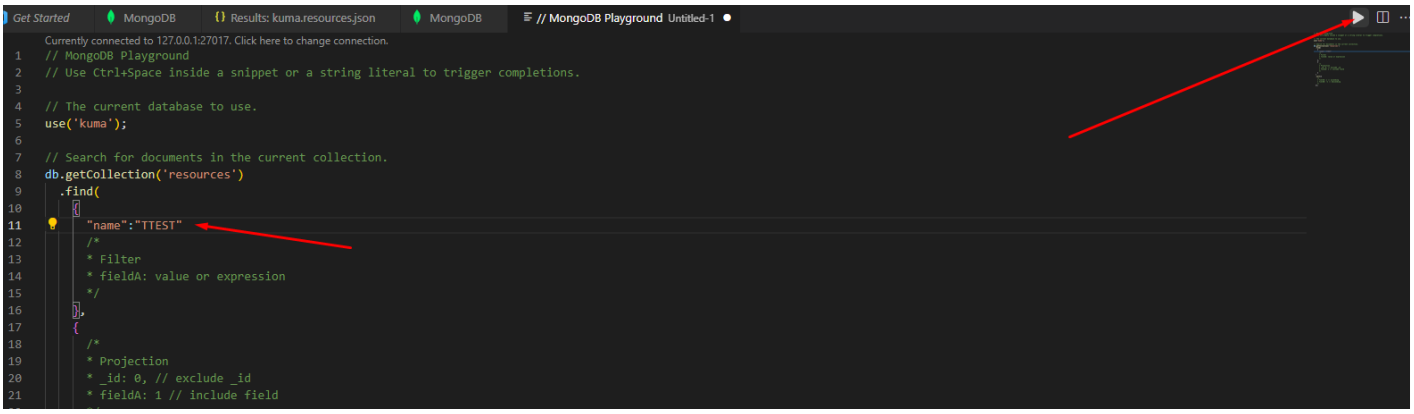
2. Заходим в БД kuma



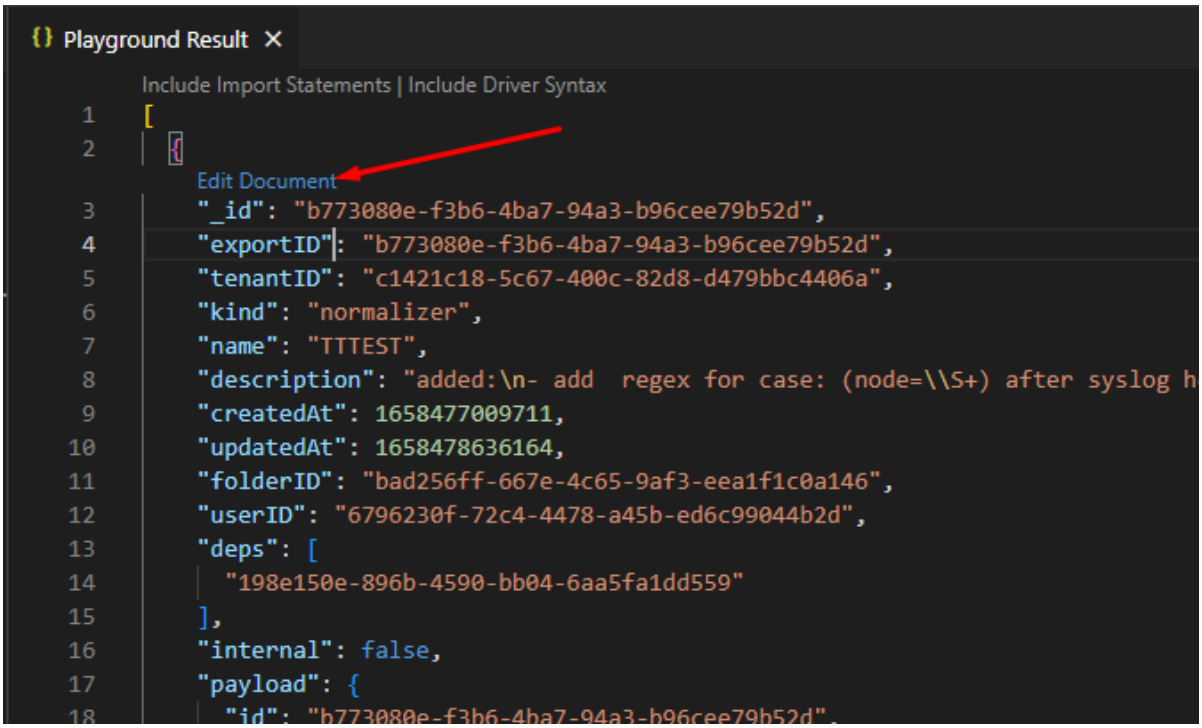
3. Переходим в коллекцию resoureces и нажимаем на значок поиска



4. Ищем ресурс по имени или ID (кому как удобно):



5. Переходим в режим редактирования и далее можно перемежать блоки JSON нормализатора и других ресурсов, как вам удобно, для сохранения используте комбинацию клавиш Ctrl+S:



Revision #12
Created 2 November 2023 08:26:27 by Boris Rzr
Updated 19 September 2024 08:21:40 by Boris Rzr