Руководство по эксплутации фрод-мониторинга.

Версия: 0.6.1

По вопросам обращайтесь: support@cybertonica.ru

Термины и определения

- Событие (event) - факт, случившийся в определенный момент времени и имеющий уникальный идентификатор. События могут иметь разную структуру, относится к разным платежным каналам, по-разному обрабатываться политиками мониторинга. События считаются неизменяемыми. Факты изменения/дополнения полетей события являются новыми событиями.
- Транзакция (transaction) - событие перехода ценности от отправителя получателю. Крому времени и идентификатора обязательно содержит так же идентификатор отправителя, получателя, сумму.
- Сессия (session) - описание непрерывного периода времени активности пользователя в фронтальной системе.
- Цифровой отпечаток (fingerprint) - набор данных о устройстве, позволяющий идентифицировать устройство с некоторой точностью.
- Трек (track) - пакет данных с информацией о поведении пользователя в клиентском веб/мобильном приложении.
- Алерт (alert) - событие сигнализирующее о угрозе.
- Очередь (queue) - упорядоченный набор алертов
- Политика (policy) - бизнес-логика с набором правил, описывающая алгоритм обработки потока событий.

Введение

Для функционирования транзакционного мониторинга необходимо провести настройку бизнес-логики, которая включает в себя политики принятия решений, справочники, модели машинного обучения(опционально).

При этом, как правило, фрод-мониторинг - это бизнес процесс, т.е. бизнес-логика должна непрерывно обновляться для реагирования на изменяющиеся угрозы и регуляторные требования. Рекомендуется руководствоваться методиками PDCA/OODA. ## Настройка политик мониторинга

Общие принципы

Правила пишутся в виде скриптов на языке Lua. Если вам нужно освежить знания Lua - рекомендуем: https://learnxinyminutes.com/docs/ru-ru/lua-ru/

При написании кода советуем придерживаться следующих принципов: - Избегайте использования глобальных переменных - Давайте переменным понятные имена, код читают чаще чем пишут - Избегайте дублирования кода, оно ведет к ошибкам в будущем - Проверяйте входные данные, им нельзя доверять - Разбивайте код на функции, код функции должен умещаться на одном экране - Разбивайте код на модули - Не забывайте использовать `pcall` в Lua для обработки ошибок. - Помните, о том что работа происходит в распределенной многопоточной среде, не забывайте про гонки (race conditions)

Принципы фрод-скоринга - Работа скоринга происходит на потоке событий, который в теории не имеет начала и конца, поэтому используйте скользящие окна в расчетах. - Общая схема работы: получение входнях данных, опрос поставщиков данных, применение моделей машинного обучения, применение правил - счетчики по временным окнам - множества по временным окнам - сравнение признаков за разные периоды - адаптивность правил - размеры выборки и статистическая значимость правил

Скрипты рекомендуется хранить в системе контроля версий (например, git).

Для обновления скриптов существует два механизма: 1) на сервере достаточно изменить файлы в папке `scripts`. Раз в секунду сервер проверяет изменения и пытается применить их. В случае ошибки пишется запись в журнал. 2) вызов API `update_script`. В случае ошибки возвращается имя скрипта и строка с найденной ошибкой.

В случае использования кластерной инсталляции рекомендуется использовать второй вариант, а при первом варианте необходимо следить за согласованным изменением файлов на всех серверах - например, в рамках CI/CD конвейера.

Потоки данных показаны на рисунке ниже.

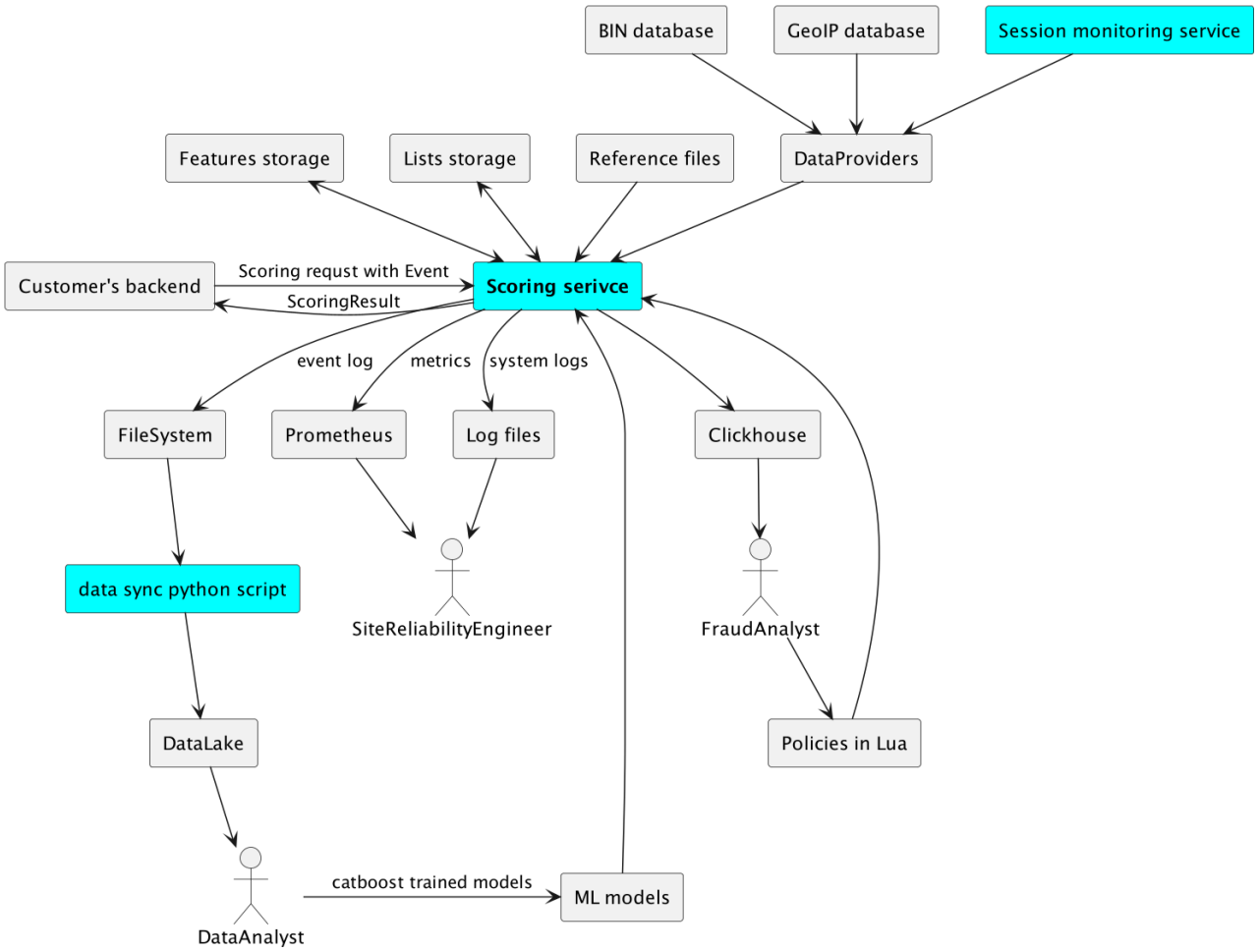

Figure 1: Dataflow diagram

Рассмотрим их подробнее.

Входные данные (event)

В новом протоколе “событие” включает в себя как создание некоторого факта (createEvent старого протокола), так и обновление статуса. Различать планируется по полю `channel`. Для обновлений каналы назвать `update_{channel}`.

События состоит из 4 групп полей: 1) общие поля - идентификатор события, время `t_ms`, `session_id`, `chanenl`, `sub_channel`. 2) поля входящего запроса, `request` 3) поля ответа: `action, score, queues, tags, comments`. Они могут пригодиться, при обработке предыдущего уже проскоренного события. 4) технические поля (tx_id, hostname, timings_mcs, version) - эти поля хотя и доступны в Lua, но бизнес-логике не нужны.

Структура события из ядра системы:

```
pub struct EventRecord {
    /// uuid
    pub tx_id: String,
    /// unix time milliseconds
    pub t_ms: u64,

    ///       ,       "scoring"
```

```
    ///          event log
    pub log_type: String,
    ///
    pub team: String,
    ///
    pub version: u16,
    ///              AS IS

    pub session_id: Option<String>,
    pub user_id: Option<String>,
    pub channel: String,
    pub sub_channel: String,
    pub prev_event_id: Option<String>,
    pub prev_event: Option<Box<EventRecord>>, // TODO: do not write to historyDB to avoid infinite link

    // todo: data_providers: binbase, geoip, fixer, session
    pub session: Option<SessionPublic>,

    pub ip_score: Option<ScoreIpPublic>,

    pub action: String,
    pub score: f32,
    pub queues: Vec<String>,
    pub tags: Vec<String>,
    pub comments: Vec<String>,

    pub error: Option<String>,
    //                       :
    ///                        -
    pub timings_mks: Vec<(String, u64)>,
    ///       , /etc/hostname
    pub hostname: String,
    ///
    pub seq_id: u64,
}
```

Результат скоринга (ScoringResult)

Эту структуру данных должен возвращать обработчик Lua `on_event`.

```
pub struct ScoringResponse {
    pub action: String,
    pub score: f32, // todo: changed from int 0..1000!
    pub queues: Vec<String>,
    pub tags: Vec<String>,
    pub comments: Vec<String>,
    pub ip_score: Option<ScoreIpPublic>
    pub extra: serde_json::Value
}

pub struct ScoreIpPublic {
    pub ip: String,
    pub geo_city_name: String,
    pub geo_country_code: String,
    pub geo_iso_code: String,
    pub geo_latitude: f64,
    pub geo_longitude: f64,

    pub blacklisted: bool,
    #[serde(default)]
    pub tags: Vec<String>,
```

}

- `action` - одно из значений `ALLOW, CHALLENGE, DENY`.
- `score` - число от 0 до 1. Замечание: в ранних версих протокола было от 0 до 1000.
- `ip_score` - результат скоринга ip-адреса, доступен только если в createEvent было передано поле `ip`.
- `extra` - произвольная таблица, которая позволяет расширять скоринг.

Работать с `ScoringResponse` удобнее через объектно-ориентированную обертку, см. utils.lua.

Расчет признаков

Временные ряды timeseries

Набор счетчиков по времени.

```
add(key, period, t_ms, value)
key: String, period: String, t_ms: u64, value: f32
period: {number}{unit}, etc. 10m, 1h, 7d
```

Внутренний формат хранения: набор пар время-счетчик, при этом счетчик отвечает за суммирование за период (, +period)

```
sum(key, t_ms, period)
```

Возвращает 0, если ключ не существует.

Множества элементов ограниченное по времени (time sets)

Набор уникаольных строк по времени.

```
add(key: String, period: String, t_ms: u64, value: String)
```

```
nunique(key, t_ms, period) -> int
```

Возвращает 0, если ключ не существует.

```
item(key, t_ms, period) -> [[times], [values]]
```

Возвращает элементы вместе с временными метками. Временные метки - unix time в миллисекундах. Хранение внутри *в секундах*.

Функции работы с проверочными списками (check_lists)

Проверочные списки вида ключ-значение.

Проверка наличия ключа в списке.

```
check_lists:contains(list_name, key)
```

Возвращает `bool`. Если key `nil`, то возвращает `false`. В случае истекшего элемента возвращает `false`. В случае несуществующего списка вызывает ошибку.

Получение записи из списка.

```
check_lists:get(name, key)
```

Возвращает `nil` или таблицу вида:

```
key: str
value: str
comment: str
expires_at:        unix time   ms
created_at:
```

Если key `nil`, то возвращает `nil`.

Добавление элемента в список.

```
check_lists:add(name, key, value, current_time_ms, ttl_period)
```

В случае вставки в несуществующий список возвращает ошибку.

todo: в случае больших списков или распределенной инсталяции эта функция имеет специфику!

Хранилище таблиц kvstorage

Хранилище произвольных таблиц.

Получение элементов

```
mget
Type:
table -> table
[str] -> [Optional[table]]
```

Сохранение элементов

```
mset
Type:
table, table -> nil
[str], [table] -> nil
```

Поставщики данных

Геолокация IP-адреса (GeoIP)

```
tools:geoip(ip)
string or nil -> table or nil
```

Пример:

```
[(postal_code => 127976),(country_name => Russia),(country_code => RU),(country_subdivision_iso => MOW)
```

Справочник по карточным Bank Identification Code (BIN)

```
tools:bininfo(b)
string or nil -> table or nil
```

Сессионный мониторинг

В объектах `EventRecord` есть поле `session`. Оно может быть либо пустым, либо содержать информацию о сессии с идентификатором `session_id`.

Структура записи `SessionPublic`:

```rust
pub struct SessionPublic {
    ///            , GUID
    pub session_id: String,
    ///              (    ,                          )
    pub device_id: String,
    ///            ,        unix time (UTC).
    pub start_t_ms: u64,
    ///           ,        unix time (UTC),
    ///                                   ,
    ///
    pub end_t_ms: u64,
    ///       fingerprint            ,
    pub n_fingerprints: u16,
    ///       tracking             ,
    pub n_tracks: u32,
    ///                fingerprint
    pub last_fingerprint_t_ms: u64,
    ///                       tracking
    pub last_track_t_ms: u64,
    /// UserAgent
    pub user_agent: String,     // TODO:               -         ?
    ///                     - web, ios, android
    pub platform: String,
    ///         (             )
    pub alerts: Vec<SessionAlert>,  // TODO: public alerts?
    ///     IP     ,
```

```
    pub ips: Vec<String>,
    // TODO: geoip?
    ///     ,
    pub referrers: Vec<String>,
}
```

Структура `SessionAlert`:

```
pub struct SessionAlert {
    ///              , unix time milliseconds.
    pub t_ms: u64,
    ///             (         tid)
    pub session_id: String,
    ///       ,     , BOT, SUSPICIOUS_IP, COPY_PASTE, DEV_TOOLS, ATO,...
    pub name: SessionAlertName,
    ///
    pub message: String,
    ///            (confidence score            ),       0   1
    pub confidence: f32,
}
```

Внешние справочники

Есть возможность загружать произвольные справочники в систему с доступом на чтение.

Формат:

текстовый файл в формате JSON

Поля:

`datetime` - дата и время формирования файла в целях аудита, YYYY-MM-DD HH:MM:SS

`data`: словарь ключ-значение.

Замечание: механизм справочников не нужно путать с механизмом `kvstorage`. Справочники предназначены для внешних статических (например, меняющихся раз в день) данных. `kvstorage` - для работы с произвольными данными в Lua.

Сервер проверяет с периодичностью раз в секунду изменения в файлах справочников и обновляет их в фоновом потоке.

Анализ данных

Мы хотим достичь следующих целей: - иметь как можно больше данных для анализа - предоставить удобный способ работы с данными - система должна быть простой и дешевой в эксплуатации.

Для достижения этих целей система мониторинга построена на основе принципа Event sourcing и при обрабокте событий пишет документы в журнал. Каждый такой документ содержит информацию о входящих и выходящих данных, а также служебную информация (например, времена задержек по этапам обработки).

Структура журнальных файлов

Журналы хранятся в отдельный папках по именам заказчика и датам.

Именование файлов и папок:

`{bucket}/{type}/{environment}/{date}/{date}_{time}_{type}_{subtype}.jsons.sz`

где: - bucket - имя корзины объектного хранилища - type - тип журналов, например, `session` или `scoring` - environment - имя серверной среды - date - дата формирования файла в формате `YYYYMMDD` - time - время формирования файла в формате `HHMMSS` - subtype - подтип файла, например, `traffic`, `sessions`, `api`.

Формат файлов: JSON записи под одной на строку с сжатием snappy.

Сессионные данные

Идентификаторы

`session_id` (ранее поле называлось `tid`) - глобально уникальный идентификатор (UUID) сгенерированный SDK, внутренняя структура идентификатора - см. руководство по SDK.

`device_id` - генерируется SDK, см. руководство по SDK.

`application_id` - UUID генерируемый SDK, обладает большей уникальностью чем `device_id`, но меньшей стабильностью.

`user_id` - (опциональное) уникальный идентификатор учетной записи пользователя на стороне заказчика.

Принципы

Везде где возможно используется время в миллисекундах с 1 января 1970 (unix epoch time, UTC).

Данные в хранилище неизменяемые (immutable), т.е. не меняются после записи.

Сырые данные (traffic)

Сырые данные можно использовать для разработки признаков (feature engineering) и отладки. Они имеют технический характер и большой объект, не рекомендуется работать с ними напрямую. Форматы и набор полей могут отличаться между веб и мобильными, между разными версиями SDK. Рекомендуется использовать *session* файлы, потому что они имеют унифицированную и стабильную структуру.

В журнал сырых данных пишутся все корректные записи, полученные от SDK.

Замечание: записи с некорректной структурой, полученные от клиентов или IP адресов черного списока, в результате атак или от старых неподдержиываемых версий SDK в журналы не пишутся.

Существует 3 типа записей:

1) цифровые отпечатки (fingerprints)
   Отправляются когда пользователь запускает веб/мобильное приложение и заново отправляются каждые несколько часов.
2) треки (tracking)
   Отправляются каждые 2-300 секунд, в зависимости от активности пользователя. Обратите внимание, что для коротких сессий треков может не быть.
3) служебные отладочные сообщения (logEvent)

Все зависи имеют следующую общую структуру:

`team: string` - имя заказчика

`method: string` - HTTP method, etc. ‘GET’, ‘POST’

`url: string` - URL of the request

`headers: map[string, string]` - HTTP headers of the request

`ip: string` - request IP address (note, it can be some proxy, not client real IP)

`time: long` - unix time in milliseconds when then request was received.

`date: string` - human readable timestam when the request was received, etc. ‘2022.10.06-13:50:31’.

`log_type: string` - service name, etc. ‘pixel’ (for session monitoring) or ‘afs’ for transaction monitoring.

`performance_tracing: array` - array of (step_name, duration) for performance profiling.

`body: object` - payload of the request

`extra: object` - additional information about request processing, etc. scoring result.

The most important fields are `team` and `body`.

Пример цифрового отпечатка устройства

Отпечатки от веб и мобильных устройств имеют разную структуру, см. спецификацию SDK.

Высокоуровневая структура: `LogRecord -> body -> commond fields + platform_specific_fields`.

Общий поля:

`t: long` - *client-side* timestamp, note: client device clock can be inaccurate.

`platform: string` - etc. ‘web’, ‘android’, ‘ios’

`version: string` - SDK version, etc. ‘4.7.3’

`sessionId: string` - session ID.

`userId: optional[string]` - optional userID specific to the customer.

`deviceId: string` - client-side deviceId

Mobile SDK example:


```
{'team': 'some_customer',
 'method': 'GET',
 'time': 1667606074379,
 'date': '2022.11.04-23:54:34',
 'log_type': 'pixel',
 'body': {'apiuser': 'some_customer',
  'applicationId': '1667322915591-11ed-934f-4df3bb2f40441',
  'bluetooth': {'isEnabled': False},
  'cpu': {'cpuInfo': [{'BogoMIPS': ' 38.40',
     'CPU_architecture': ' 8',
     'CPU_implementer': ' 0x51',
     'CPU_part': ' 0x805',
     'CPU_revision': ' 14',
     'CPU_variant': ' 0xd',
     'Features': ' fp asimd evtstrm aes pmull sha1 sha2 crc32 atomics fphp asimdhp cpuid asimdrdm lrcpc
     'Processor': ' AArch64 Processor rev 15 (aarch64)',
     'coresCount': 1,
     'processor': ' 0'}]},
  'cybertonica_info': {'appName': '...',
   'className': '...application.App',
   'firstInstallTime': 1667316609602,
   'hasEmulatorDrivers': False,
   'hasEmulatorFilesystemEntries': False,
   'hasEmulatorNetworkConf': False,
   'isAdbActive': False,
   'isSimulator': False,
   'lastUpdateTime': 1667316609602,
   'minSdkVersion': 23,
   'packageName': '...app_money',
   'platform': 'android',
   'processName': '...money',
   'publicSourceDir': '/data/app/.../base.apk',
   'sharedUserLabel': 0,
   'signature': 'MF1ElhdZ58UYponfQ9mQb0eEby0=',
   'sourceDir': '/data/app/.../base.apk',
   'storageUuid': '41217664-9172-527a-b3d5-edabb50a7d69',
   'store': 'com.android.vending',
   'targetSdkVersion': 31,
   'taskAffinity': '...money',
   'uid': 10289,
   'version': '2.8.0',
   'versionCode': 15027,
   'versionName': '1.55.0'},
  'device': {'availMem': 2528624640,
```

```
 'availableProcessorsJVM': 8,
 'freeBytes': '75381207040',
 'freeMemoryJVM': 237107272,
 'icc_card': 'true',
 'java_vm_version': '2.1.0',
 'maxMemoryJVM': 268435456,
 'network_country_iso': 'ru',
 'network_operator': '25020',
 'network_operator_name': 'Tinkoff',
 'phone_type': '1',
 'phones_count': '2',
 'rooted': False,
 'sim_country_iso': 'ru',
 'sim_operator': '25062',
 'sim_operator_name': 'Tinkoff',
 'sim_state': '5',
 'totalBytes': '112419729408',
 'totalMemory': 5851787264,
 'totalMemoryJVM': 268435456},
'deviceId': '98842772774806d2',
'device_settings': {'accessibility_enabled': False,
 'development_settings_enabled': True,
 'install_non_market_apps': True},
'display_info': {'brightness': '0.6862745098039216',
 'compatibleWidthLimitDp': 0,
 'density': 2.75,
 'densityDpi': 440,
 'scaledDensity': 2.75,
 'screenResolution': '1080x2254',
 'xdpi': 397.56500244140625,
 'ydpi': 395.843994140625},
'height': 2254,
'is_mobile': True,
'motion': [{'maxRange': 78.45317840576172,
  'minDelay': 2404,
  'name': 'lsm6ds3c Accelerometer Non-wakeup',
  'power': 0.15000000596046448,
  'resolution': 0.0023928226437419653,
  'type': 1,
  'vendor': 'STMicro',
  'version': 140549},
 {'maxRange': 34.90549087524414,
  'minDelay': 2404,
  'name': 'lsm6ds3c Gyroscope Non-wakeup',
  'power': 0.5550000071525574,
  'resolution': 0.0012216945178806782,
  'type': 4,
  'vendor': 'STMicro',
  'version': 140549}],
'network': {'ip': '192.168.31.39'},
'os': {'board': 'joyeuse',
 'bootloader': 'unknown',
 'brand': 'Redmi',
 'defaultLocale': 'ru',
 'device': 'joyeuse',
 'display': 'RKQ1.200826.002 test-keys',
 'fingerprint': 'Redmi/joyeuse_ru/joyeuse:11/RKQ1.200826.002/V12.5.4.0.RJZRUXM:user/release-keys',
 'hardware': 'qcom',
 'id': 'RKQ1.200826.002',
 'localeIdentifier': 'ru',
 'manufacturer': 'Xiaomi',
```

```
   'model': 'Redmi Note 9 Pro',
   'product': 'joyeuse_ru',
   'radioVersion': 'MPSS.AT.4.4.c6-00057-1,MPSS.AT.4.4.c6-00057-1',
   'release': '11',
   'releaseVersion': '11',
   'supported_32_bit_abis': ['armeabi-v7a', 'armeabi'],
   'supported_64_bit_abis': ['arm64-v8a'],
   'systemVersion': 30,
   'tags': 'release-keys',
   'time': 1649995712000,
   'type': 'user',
   'user': 'builder',
   'version_codename': 'REL',
   'version_incremental': 'V12.5.4.0.RJZRUXM',
   'version_preview_sdk_int': 0,
   'version_security_patch': '2022-03-01'},
  'platform': 'android',
  'referrer': 'some_customer',
  'sessionId': '1667606073209-11ed-af53-83c8744ce342',
  'system': {'_errorCount': 1,
   'fp_count_fail': 0,
   'fp_history': [],
   'last_fp_t': 1667476470994},
  't': 1667606073252,
  'tid': '1667606073209-11ed-af53-83c8744ce342',
  'version': '2.8.0',
  'width': 1080},
 'url': '/fingerprint',
 'headers': {'host': 'session-srv.cbt.ydx',
  'content-length': '2815',
  'content-encoding': 'gzip',
  'user-agent': 'Dalvik/2.1.0 (Linux; U; Android 11; Redmi Note 9 Pro Build/RKQ1.200826.002)',
  'connection': 'keep-alive',
  'accept-encoding': '',
  'x-forwarded-for': '62.148.157.253, 178.154.224.139',
  'content-type': 'application/json; charset=UTF-8'},
 'ip': '62.148.157.253',
 'extra': {},
 'performance_tracing': [['parsing', 0],
  ['get_active_session', 0],
  ['velocity_check', 1],
  ['phishing_check', 0],
  ['domain_check', 0],
  ['ip_check', 0],
  ['user_agent_check', 0],
  ['analyse_session', 0],
  ['session_hijacks', 0],
  ['alerts_detection', 0],
  ['processing', 1],
  ['handle_fingerprint', 2]]}
```

Привер трека с устройства

Треки периодически отправляются устройством.

Сессия (идентифицируемая `session_id`) может иметь от нуля до множества треков.

Треки могут содержать внутри временные ряда с движениями мыши, касаниями экрана, кликами, событиями клавиатуры и т.д.

```
{'team': 'some_customer',
 'method': 'POST',
```

```
'time': 1667606074554,
'date': '2022.11.04-23:54:34',
'log_type': 'pixel',
'body': {'_errorCounter': 0,
 'apiUserName': 'some_customer',
 'applicationId': '1663840340110-4c5f-0d83-3d5530cb3711',
 'autofillTrack': [],
 'currentTimeForServerTime': 1667606086775,
 'deviceId': '53815381177557252552037373711617758517583720878418 6',
 'deviceMotion': {},
 'dom': {'c': [{'bg': '0,0,0,0',
    'c': [{'bg': '255,255,255',
      'c': [{'bg': '0,0,0,0', 'r': '16,-119,374,-119', 'tn': 'DIV'},
       {'bg': '0,0,0,0',
        'c': [{'bg': '0,0,0,0',
          'c': [{'bg': '0,0,0,0', 'r': '16,-119,114,-82', 'tn': 'TD'},
           {'bg': '0,0,0,0',
            'c': [{'bg': '0,0,0,0', 'r': '134,-121,205,-95', 'tn': 'SPAN'}],
            'r': '114,-119,206,-82',
            'tn': 'TD'}],
          'r': '16,-119,206,-82',
          'tn': 'TR'},
         {'bg': '0,0,0,0',
          'c': [{'bg': '0,0,0,0', 'r': '16,-82,114,-45', 'tn': 'TD'},
           {'bg': '0,0,0,0',
            'c': [{'bg': '0,0,0,0', 'r': '154,-84,205,-58', 'tn': 'SPAN'}],
            'r': '114,-82,206,-45',
            'tn': 'TD'}],
          'r': '16,-82,206,-45',
          'tn': 'TR'},
         {'bg': '0,0,0,0',
          'c': [{'bg': '0,0,0,0', 'r': '16,-45,114,-8', 'tn': 'TD'},
           {'bg': '0,0,0,0',
            'c': [{'bg': '0,0,0,0', 'r': '134,-47,205,-21', 'tn': 'SPAN'}],
            'r': '114,-45,206,-8',
            'tn': 'TD'}],
          'r': '16,-45,206,-8',
          'tn': 'TR'}],
        'r': '16,-119,206,-8',
        'tn': 'TABLE'},
       {'bg': '0,0,0,0', 'r': '16,-8,374,17', 'tn': 'DIV'},
       {'bg': '0,0,0,0',
        'c': [{'bg': '0,0,0,0',
          'c': [{'bg': '0,0,0,0',
            'c': [{'bg': '0,0,0,0',
              'c': [{'bg': '0,0,0,0',
                'c': [{'bg': '0,0,0,0', 'r': '21,46,65,90', 'tn': 'IMG'},
                 {'bg': '0,0,0,0', 'r': '73,58,164,78', 'tn': 'SPAN'},
                 {'bg': '0,0,0,0', 'r': '180,57,317,79', 'tn': 'SPAN'}],
                'r': '17,42,345,94',
                'tn': 'DIV'},
               {'bg': '0,0,0,0',
                'c': [{'bg': '0,0,0,0', 'r': '345,64,357,71', 'tn': 'IMG'}],
                'r': '345,42,373,94',
                'tn': 'DIV'}],
              'r': '16,41,374,95',
              'tn': 'DIV'},
             {'bg': '0,0,0,0', 'r': '16,95,374,95', 'tn': 'DIV'}],
            'r': '16,41,374,95',
            'tn': 'DIV'}],
          'r': '16,41,374,95',
```

```
          'tn': 'DIV'}],
        'r': '0,41,390,95',
        'tn': 'DIV'},
       {'bg': '0,0,0,0',
        'c': [{'bg': '243,244,246',
          'c': [{'bg': '0,0,0,0',
            'c': [{'bg': '0,0,0,0',
              'c': [{'bg': '0,0,0,0', 'r': '33,128,243,148', 'tn': 'H5'},
               {'bg': '0,0,0,0', 'r': '33,148,243,172', 'tn': 'P'}],
              'r': '33,128,243,172',
              'tn': 'DIV'},
             {'bg': '0,0,0,0', 'r': '243,128,287,172', 'tn': 'IMG'}],
            'r': '33,128,287,172',
            'tn': 'DIV'},
           {'bg': '0,0,0,0',
            'c': [{'bg': '255,255,255', 'r': '33,188,137,230', 'tn': 'INPUT'},
             {'bg': '0,0,0,0', 'r': '33,238,287,260', 'tn': 'P'}],
            'r': '33,188,287,260',
            'tn': 'DIV'}],
          'r': '16,111,304,277',
          'tn': 'DIV'}],
        'r': '16,111,374,277',
        'tn': 'DIV'},
       {'bg': '243,244,246,0.4',
        'c': [{'bg': '0,0,0,0',
          'c': [{'bg': '0,0,0,0',
            'c': [{'bg': '0,0,0,0', 'r': '264,412,314,429', 'tn': 'A'}],
            'r': '16,397,374,445',
            'tn': 'P'}],
          'r': '16,397,374,445',
          'tn': 'DIV'}],
        'r': '0,301,390,477',
        'tn': 'DIV'}],
      'r': '0,-207,390,477',
      'tn': 'DIV'},
     {'bg': '0,0,0,0',
      'c': [{'bg': '0,0,0,0',
        'c': [{'bg': '0,0,0,0', 'r': '111,497,279,529', 'tn': 'IMG'}],
        'r': '0,477,390,550',
        'tn': 'DIV'},
       {'bg': '0,0,0,0',
        'c': [{'bg': '0,0,0,0', 'r': '43,570,347,594', 'tn': 'IMG'}],
        'r': '0,550,390,614',
        'tn': 'DIV'}],
      'r': '0,477,390,614',
      'tn': 'FOOTER'}],
    'r': '0,-207,390,614',
    'tn': 'DIV'}],
  'r': '0,-207,390,614',
  'tn': 'BODY'},
 'editkd': [{'count_letters_begin': 0,
   'count_letters_end': None,
   'htmlClass': 'rounded',
   'id': 2872213225,
   'keyDown': [[1667606074236, 1, 0, 99]],
   'keyUp': [[1667606074262, 1, 0, 3]],
   'rect': [33, 137, 395, 437],
   't1': 1667606074236,
   't2': None}],
 'extra': {'fplat': 157,
  'ind_pr': False,
```

```
 'ls_pr': True,
 'prev_session_ids': ['1667590253a36-4735-7505-9f9ddb64b5e0',
  '16675965594c2-48fe-933b-45a505c83edd',
  '16675995978e1-44b5-7877-d0622adbea20'],
 'requestNumber': 1,
 'stat': 2,
 'stid': '1663840340110-4c5f-0d83-3d5530cb3711'},
'height': 614,
'iframes': [],
'keyDownTrack': [[1667606074236, -1, '']],
'keyUpTrack': [],
'lastKnownServerTime': 1667606070618,
'mouseTrack': [[1667606073343, 74, 399], [1667606073434, 75, 399]],
'mouseTrackClicks': [[1667606073480, 75, 399, '#input#div#div#div', '']],
'pageSelection': [],
'pageTrack': [],
'platform': 'web',
'referrer': 'pay.domain.ru/refill',
'scripts': [{'code': '!function (t, e) { "object" == typeof exports && "undefined" != typeof module ?
  'evnts': '',
  'id': 3236147597,
  'len': 512,
  't': 1667606071857,
  'tagname': 'script',
  'url': ''},
 {'code': 'var __vite_is_dynamic_import_support=false;',
  'evnts': '',
  'id': 3807647784,
  'len': 43,
  't': 1667606071857,
  'tagname': 'script',
  'url': ''},
 {'code': 'try{import("_").catch(()=>1);}catch(e){}window.__vite_is_dynamic_import_support=true;',
  'evnts': '',
  'id': 3621678075,
  'len': 85,
  't': 1667606071857,
  'tagname': 'script',
  'url': ''},
 {'code': '!function(){if(window.__vite_is_dynamic_import_support)return;console.warn("vite: loading
  'evnts': '',
  'id': 550385361,
  'len': 420,
  't': 1667606071857,
  'tagname': 'script',
  'url': ''},
 {'code': '!function(){var e=document,t=e.createElement("script");if(!("noModule"in t)&&"onbeforeload
  'evnts': '',
  'id': 1369398380,
  'len': 314,
  't': 1667606071857,
  'tagname': 'script',
  'url': ''},
 {'code': "System.import(document.getElementById('vite-legacy-entry').getAttribute('data-src'))",
  'evnts': '',
  'id': 1031040633,
  'len': 84,
  't': 1667606071858,
  'tagname': 'script',
  'url': ''}],
'scrollTrack': [[1667606073547, 207, 207, 74, 399]],
```

```
 'sessionId': '1667606070305-4098-f8b3-442ec9bcdd25',
 'signals': {'F12': 0,
  'auto': 0,
  'back': 0,
  'contextm': 0,
  'csi': 0,
  'ctrlV': 0,
  'dev': 0,
  'priv': 0,
  'rtc': '0.0.0.0',
  'tab': 0},
 't1': 1667606070254,
 't2': 1667606074453,
 'tid': '1667606070305-4098-f8b3-442ec9bcdd25',
 'touchesEllipseTrack': [[1667606073343, 24, 24, 0, 0]],
 'url': 'https://pay.domain.ru/refill?requestId=some_id',
 'userId': '',
 'version': '4.2.0',
 'width': 390,
 'zoom': 1},
'url': '/tracking',
'headers': {'content-type': 'application/json',
 'origin': 'https://pay.domain.ru',
 'x-real-ip': '85.249.167.188',
 'user-agent': 'Mozilla/5.0 (iPhone; CPU iPhone OS 16_0 like Mac OS X) AppleWebKit/605.1.15 (KHTML, li
 'content-length': '8120',
 'x-forwarded-server': 'pay.domain.ru',
 'connection': 'keep-alive',
 'cookie': 'cbtmon=xftroLofE93sEX1GLrU7T15+cNXqb+p+9NheAgX9QWQqLnXZcZV2Vhh/or3DM8xsK4i4UoZglcuKPIpfCDA
 'x-forwarded-host': 'pay.domain.ru',
 'accept-language': 'ru',
 'accept-encoding': 'gzip, deflate, br',
 'referer': 'https://pay.domain.ru/refill?requestId=someID',
 'x-forwarded-proto': 'https',
 'accept': '*/*',
 'host': 'session-srv.cbt.ydx',
 'x-forwarded-for': '85.249.167.188, 195.19.208.235'},
'ip': '85.249.167.188',
'extra': {},
'performance_tracing': [['track_decoding', 0],
 ['track_parsing', 0],
 ['get_active_session', 0],
 ['velocity_check', 0],
 ['phishing_check', 0],
 ['domain_check', 0],
 ['ip_check', 0],
 ['user_agent_check', 0],
 ['web_inject', 0],
 ['apply_bot_model', 0],
 ['apply_ato_model', 0],
 ['analyse_session', 0],
 ['session_hijacks', 0],
 ['alerts_detection', 0],
 ['track_processing', 1]]}
```


Сессии

Это структурированное представление пользовательских сессий, рассчитываемое на основе отпечатков и треков.

“Сессия” означает непрерывный период активности пользователя.

Сессия пишется в журнал, когда наступает одно из событий: - нет треков или отпечатков в течение определенного периода

времени (например, 15 минут) - длительность сессии достигает 24 часов.

Это означает, что может быть несколько зависей в одним и тем же `session_id`, когда пользователь приостанавливает активность, а потом возобновляет.

Каждая запись о сессии включает: - данные о сессии (время начала, конца, кол-во треков, IP адреса и т.д.) - алерты для сессии

Alerts - см. руководство по сессионному мониторингу

Features - словарь `string->float` с признаками для моделей машинного обучения. Признаки с значением 0 не пишутся, чтобы сохранить место.

Пример сессии

```
{'team': 'some_customer',
 'session_id': '1667690735a49-4e4e-5f59-31fc58df0000',
 'device_id': '53815381177557101052037373711617758517583720878418 6',
 'prev_session_id': '',
 'start_t_ms': 1667690735053,
 'end_t_ms': 1667690741917,
 'n_fingerprints': 1,
 'n_tracks': 3,
 'last_fingerprint_t_ms': 1667690735438,
 'last_track_t_ms': 1667690742129,
 'user_agent': '"Mozilla/5.0 (iPhone; CPU iPhone OS 16_0_2 like Mac OS X) AppleWebKit/605.1.15 (KHTML,
 'platform': 'web',
 'is_mobile': True,
 'width': 414,
 'height': 814,
 'mouse_track': {'points': [{'t': 1667690736961.0, 'x': 90.0, 'y': 408.0},
   {'t': 1667690737025.0, 'x': 90.0, 'y': 409.0},
   {'t': 1667690738885.0, 'x': 336.0, 'y': 361.0},
   {'t': 1667690738943.0, 'x': 337.0, 'y': 361.0},
   {'t': 1667690741824.0, 'x': 336.0, 'y': 343.0},
   {'t': 1667690741884.0, 'x': 336.0, 'y': 343.0}]},
 'mouse_clicks': [{'t': 1667690737029.0,
   'x': 90.0,
   'y': 409.0,
   'sid': '#input#div#div#div',
   'sname': '',
   'cbt_data': None},
  {'t': 1667690738959.0,
   'x': 337.0,
   'y': 361.0,
   'sid': 'button-submit',
   'sname': '',
   'cbt_data': None},
  {'t': 1667690741917.0,
   'x': 336.0,
   'y': 343.0,
   'sid': '#a#div#div$app',
   'sname': '',
   'cbt_data': None}],
 'touch_move': {'points': []},
 'touch_up': {'points': []},
 'touch_down': {'points': []},
 'key_down_track': {'ts': [1667690737820.0, 1667690738127.0, 1667690738359.0]},
 'key_up_track': {'ts': [1667690737842.0, 1667690738153.0, 1667690738368.0]},
 'mouse_down_track': {'ts': []},
 'mouse_up_track': {'ts': []},
 'alerts': [{
        "t_ms": 1664530433693,
        "session_id": "wdc471-16645303963-815c-013c94ab4c1e",
```

```
        "name": "DEV_TOOLS",
        "message": "Browser DevTools active",
        "confidence": 0.85,
        "user_id": "cbt_test_user_id",
        "ip": null,
        "details": "NONE"
      }],
 'features': {'mouse_track_vy_std': 1.2217107e-05,
  'mouse_track_vy_percentile_75': 1.919549e-05,
  'mouse_track_vy_corrcoef_1': 0.051189043,
  'velocity_device_1h': 40965.363,
  'velocity_device_4h': 227770.98,
  'mouse_track_dx_corrcoef_1': -0.33514246,
  'mouse_track_dt_corrcoef_2': 0.9999938,
  'mouse_track_vx_percentile_75': 4.1646177e-05,
  'mouse_track_dt_mean': 984.6,
  'mouse_track_vx_count': 5.0,
  'mouse_track_vy_count': 5.0,
  'mouse_track_dy_max': 0.058968067,
  'mouse_track_vy_mean': 1.1714844e-05,
  'mouse_track_dy_count': 5.0,
  'mouse_track_dx_max': 0.5942029,
  'mouse_track_dt_min': 58.0,
  'mouse_track_dt_corrcoef_1': -0.9113959,
  'mouse_track_dt_percentile_75': 1860.0,
  'mouse_track_dy_corrcoef_1': -0.694449,
  'mouse_track_dt_max': 2881.0,
  'mouse_track_vx_std': 0.00012457202,
  'mouse_track_dx_corrcoef_2': 0.4969419,
  'mouse_clicks_count': 3.0,
  'velocity_subnet_4h': 10650.389,
  'mouse_track_dy_std': 0.02285937,
  'mouse_track_vy_corrcoef_2': 0.79922456,
  'mouse_track_dt_percentile_25': 60.0,
  'mouse_track_dy_mean': 0.01646192,
  'mouse_track_dt_count': 5.0,
  'velocity_subnet_5m': 109.60095,
  'mouse_track_dy_corrcoef_2': 0.9998338,
  'mouse_track_vx_corrcoef_1': -0.2732565,
  'velocity_ip_4h': 180.6097,
  'mouse_track_dx_std': 0.23720051,
  'velocity_ip_1h': 61.485905,
  'mouse_track_vx_max': 0.0003194639,
  'mouse_track_vx_mean': 7.23897e-05,
  'velocity_ip_5m': 5.0426493,
  'mouse_track_dy_percentile_75': 0.022113025,
  'mouse_track_dx_mean': 0.11980677,
  'mouse_track_vy_max': 3.170326e-05,
  'velocity_subnet_1h': 1888.2886,
  'mouse_track_vx_corrcoef_2': -0.5860504,
  'velocity_device_5m': 2166.776,
  'mouse_track_dx_count': 5.0,
  'mouse_track_dt_std': 1176.7443,
  'mouse_track_dx_percentile_75': 0.0024154782},
 'ips': ['213.87.155.209'],
 'referrers': ['pay.domain.ru/refill', 'pay.domain.ru/success'],
 'n_sensors': 0,
 'os_board': '',
 'os_brand': '',
 'os_model': '',
 'battery': None,
```

```
 'is_vpn_connected': False,
 'boot_t_ms': None,
 'user_id': None,
 'sensors': {'accelerometer': [], 'gyro': []},
 'application_id': '166769020565f-4b75-f82a-c2ded87fd057',
 'version': '4.2.0',
 'device_is_rooted': False,
 'geo_city_name': 'Moscow',
 'geo_country_code': 'RU',
 'geo_iso_code': 0,
 'geo_latitude': 55.7483,
 'geo_longitude': 37.6171,
 'captcha_in_progress': 0,
 'captcha_correct': 0,
 'captcha_incorrect': 0,
 'fp_latency': 120,
 'tracking_latencies': [146],
 'errors_count': 0,
 'event_ids': []}
```

Python пример

Пример чтения журналов с snappy-сжатием

```
import io
import snappy
import json

def load_snappy_file(filename):
    out = io.BytesIO()
    with open(filename, 'rb') as fd:
        snappy.stream_decompress(fd, out)
    out.seek(0)
    return out

def parse_lines(stream):
    for line in stream:
        yield json.loads(line)

records = list(parse_lines(load_snappy_file('somefile.json.sz')))
```

В реальных применениях при больших объемах данных может быть целесообразно использовать dask bag/dataframe.