CONFIANT

The Yandex Leak: How a Russian Search Giant Uses Consumer Data

Kaileigh McCrea, Privacy Engineer, Confiant

About Me

Kaileigh McCrea

- Privacy Engineer at Confiant (3 yrs)
- Software Engineer (6 years)
- Cybersecurity Nerd
- Recovering Political Science major
- Twitter: @kaileighrose

What we're talking about

Yandex

YANDEX SERVICES SOURCE CODE LEAK

SHORT OVERVIEW OF BREACH CONTENTS

PUBLISHED THU, JAN 26, 2023 BY ARSENIY SHESTAKOV

BLEEPINGCOMPUTER

Search Site

NEWS ▾ DOWNLOADS ▾ VPNS ▾ VIRUS REMOVAL GUIDES ▾ TUTORIALS ▾ DEALS ▾

Home > News > Security > Yandex denies hack, blames source code leak on former employee

Yandex denies hack, blames source code leak on former employee

By Bill Toulas

January 26, 2023 09:44 AM 1

yandex git sources

by [redacted] - Wednesday January 25, 2023 at 03:48 PM

Yesterday, 03:48 PM (This post was last modified: 2 hours ago by dedale.) #1

Repositories only, no data. Size 44.71 G
Copied almost completely except for the anti-spam rules
Downloaded by me on 07.2022

magnet:?
xt=urn:[redacted]
.openbit[redacted]acker
acker.ope[redacted]
%3A%2F%2F[redacted]%2Fan
nounce[redacted]

BreachForums User

MEMBER

Posts:	1
Threads:	1
Joined:	Jan 2023
Reputation:	0

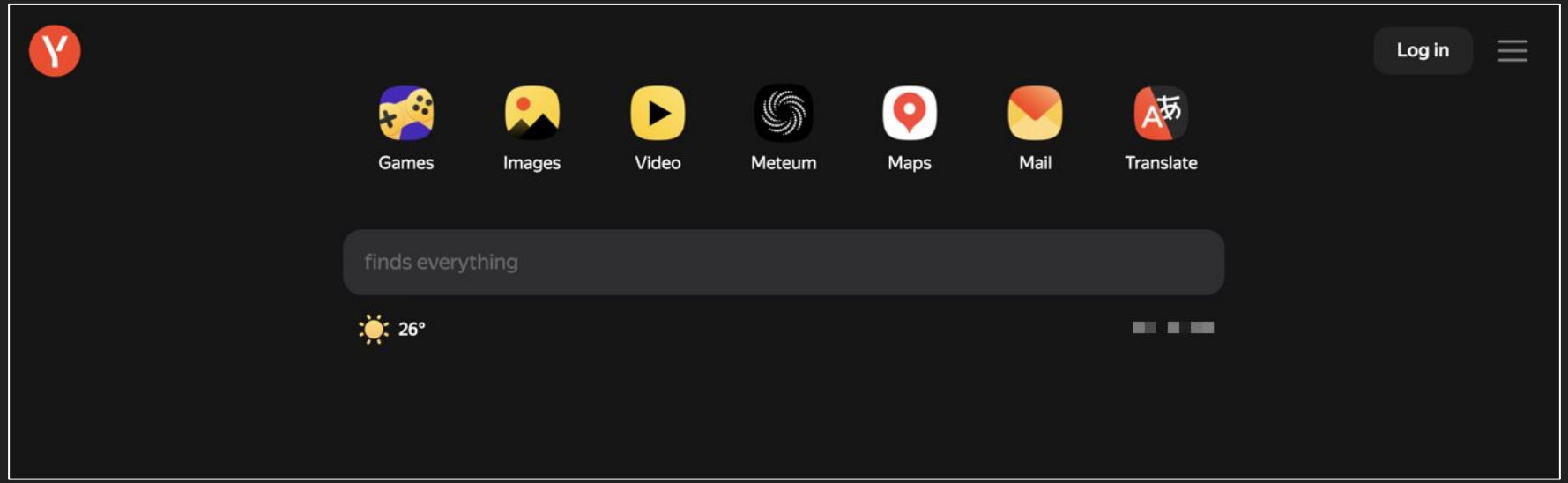
PM Find Reply Quote Report

Roadmap

- Background on Yandex Leak
- Dive into code:
  - What data Yandex is collecting
  - What Yandex is doing with that data
  - Who Yandex is sharing that data with
- Conclusions and wrap up
- Q&A

Yandex 101

Yandex 101


Log in
Games
Images
Video
Meteum
Maps
Mail
Translate
finds everything
26°

Yandex 101

Key Businesses

- **Search and Portal**
  - Yandex Search
  - Geo Services
  - Yandex Browser
  - Yandex Q
  - Other
- **Business Units**
  - E-commerce, Mobility and Delivery
    - Mobility
    - E-commerce
    - Other O2O
  - Plus and Entertainment Services
    - Yandex Plus
    - Yandex Music
    - Kinopoisk
    - Yandex Afisha
    - Yandex Studio
  - Classifieds
    - Auto.ru
    - Yandex Realty
    - Yandex Rent
    - Yandex Travel
- **Other Business Units and Initiatives**
  - Yandex SDG
  - Yandex Cloud
  - Yandex Education
  - Devices and Alice
  - Other

Yandex 101

<u>AppMetrica</u>: “In-depth analytics for product and growth teams”

<u>Audiences</u>: allows you to pull data from several sources to generate your own targeted segments

<u>Crypta</u>: “helps to identify important user characteristics for advertisers”

Yandex 101

Yandex LLC

Head office in Russia: Moscow

Head office

16, Leo Tolstoy St., Moscow, Russia 119021
tel.: +7 495 739-70-00
fax: +7 495 739-70-70

Advertising clients

tel.: +7 495 739-37-77
fax: +7 495 739-23-32
adv@yandex-team.ru

Investor Relations

tel.: +7 495 974-35-38
askIR@yandex-team.ru

Public relations

pr@yandex-team.ru

Corporate Secretary

secretary@yandex-team.ru

Sustainability

sustainability@yandex-team.com

Yandex N.V.

Registered office in Amsterdam

Schiphol Boulevard 165, 1118 BG Schiphol, The Netherlands
tel.: +31 0 20 206 6970

Official Telegram channel for individual investors https://t.me/yndx_forinvestors (in Russian only)

Yandex: A Drama

Russia

This article is more than **3 years old**

Russian internet giant grants veto powers to Kremlin-linked body

Yandex agrees to corporate restructuring in move likely to increase government oversight

Andrew Roth *in Moscow*

Mon 18 Nov 2019 06.30 EST

Arkady Volozh, the chief executive of Yandex, said the company would maintain control over its daily operations. Photograph: Mikhail Metzel/Tass

Warnings raised over Russian tech giant Yandex's UK operation

MPs want restrictions placed on the company, known as Russia's Google, which also runs the Yango Deli grocery service

Russia-Ukraine war: live news

Shanti Das

Sat 5 Mar 2022 15.02 EST

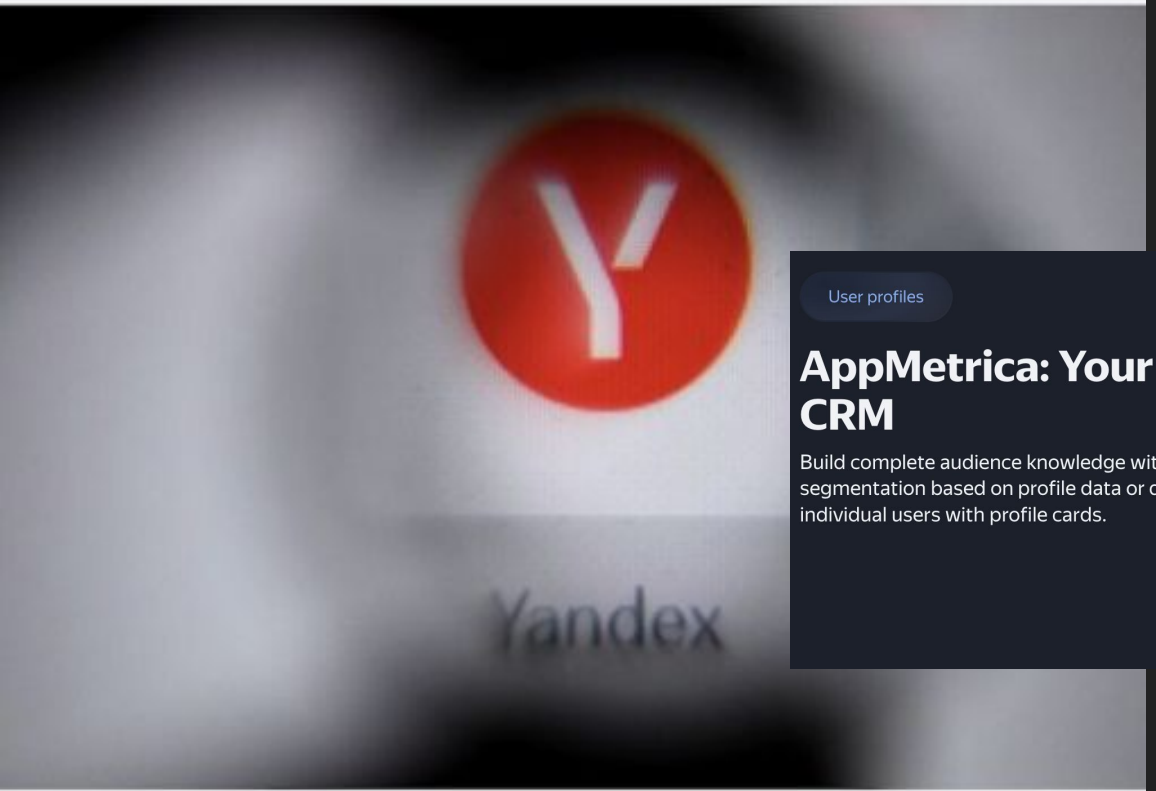
A Yango Deli driver on an electric moped delivers to homes in London. The service is expanding across the city. Photograph: John Sibley/Reuters

BOUNTIFUL HARVEST —

Data-harvesting code in mobile apps sends user data to “Russia’s Google”

Data from apps on Apple- and Google-powered mobile devices is sent to Russian servers.

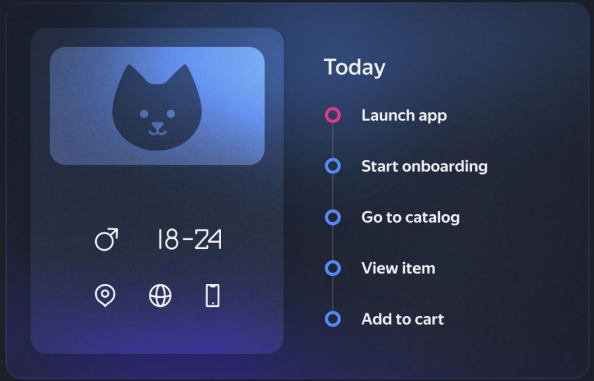
PATRICK MCGEE, FINANCIAL TIMES - 3/29/2022, 7:18 AM



User profiles

AppMetrica: Your app’s CRM

Build complete audience knowledge with segmentation based on profile data or dive into individual users with profile cards.



Featured Article

Russia's war hits Yandex, the 'Google of Russia'

Sources say the company is seeking a media exit as top exec hit with sanctions over propaganda charge

Natasha Lomas, Ingrid Lunden / 12:20 PM PDT • March 16, 2022

Comment

Premium HOME > TECH

'I bought a plane ticket and left 12 hours later': Engineers at Yandex, Russia's Google rival, are fleeing abroad and leaving spouses and salaries behind

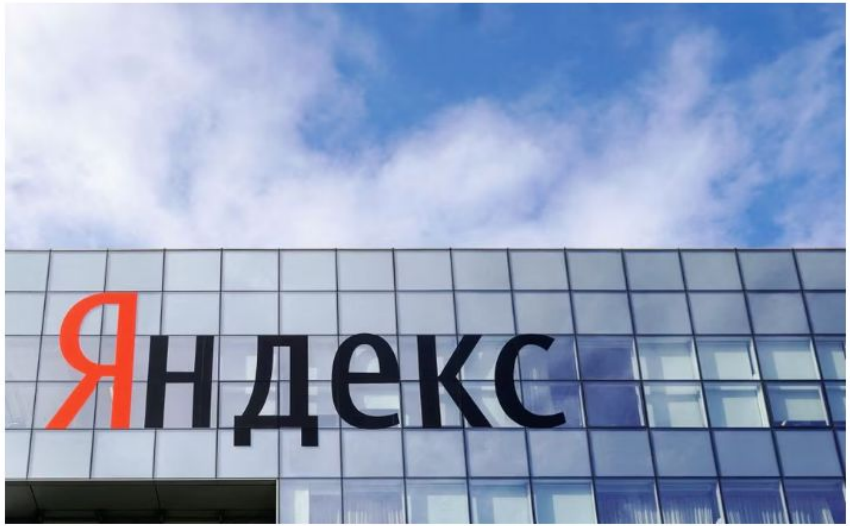
Rosie Bradbury Apr 12, 2022, 3:35 AM PDT

Technology

Yandex CEO resigns after being targeted by EU sanctions

Reuters

June 3, 2022 7:35 AM PDT · Updated a year ago



The logo of Russian internet group Yandex is pictured at the company's headquarter in Moscow, Russia October 4, 2018. REUTERS/Shamil Zhumatov

June 3 (Reuters) - Russian internet giant Yandex (YNDX.O) said on Friday that Arkady Volozh had stepped down as CEO and left the board of directors after the European Union included him on its latest list of sanctions against Russian entities and individuals.

Media & Entertainment

Yandex's sale of media assets to VK includes yandex.ru homepage

Natasha Lomas @riptari / 12:05 AM PDT • August 23, 2022

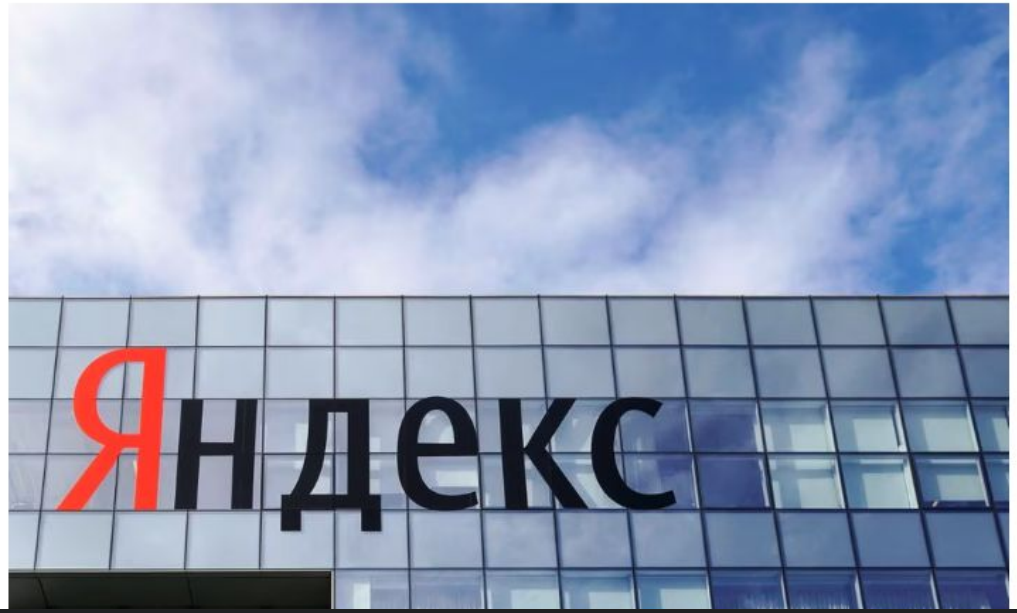
Comment

Deals

Yandex parent to review ownership of Russian tech giant, seek divestment

By **Alexander Marrow**, **Darya Korsunskaya** and **Polina Devitt**

November 25, 2022 7:31 AM PST · Updated 8 months ago



REUTERS

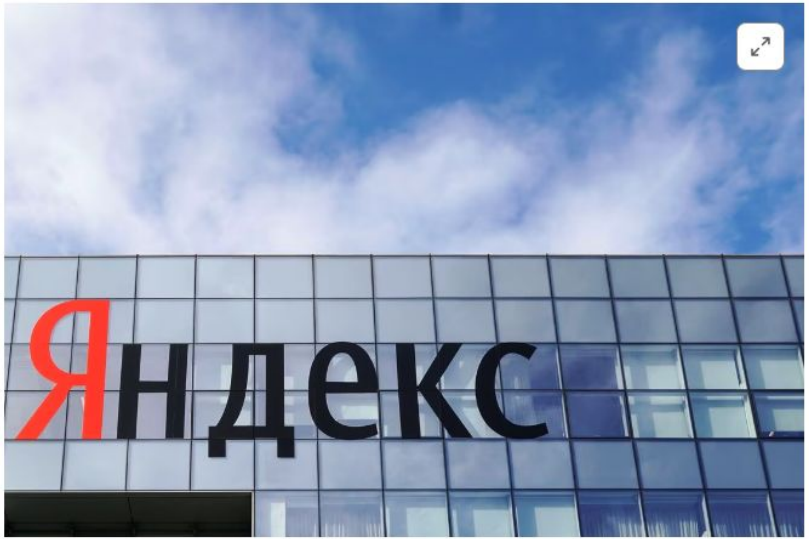
World Business Markets Sustainability Legal More

Europe

Putin, Kudrin touch on future of Yandex in late-night meeting -sources

Reuters

November 25, 2022 4:19 AM PST · Updated 8 months ago



[1/2] The logo of Russian internet group Yandex is pictured at the company's headquarter in Moscow, Russia October 4, 2018. REUTERS/Shamil Zhumatov

The Moscow Times

30 YEARS

INDEPENDENT NEWS FROM RUSSIA

NEWS UKRAINE WAR BUSINESS OPINION ARTS AND LIFE PODCASTS

Kremlin Ally Kudrin Confirms Move to Tech Giant Yandex

Dec. 5, 2022

PERSONAL WEBSITE

ABOUT ME CONTACTS NOT A CV PROJECTS

YANDEX SERVICES SOURCE CODE LEAK

SHORT OVERVIEW OF BREACH CONTENTS

PUBLISHED THU, JAN 26, 2023 BY ARSENIY SHESTAKOV

Just a few hours ago I found mention on Twitter that proprietary source code of Russian giant Yandex been leaked on online community called *BreachForums*. In this post I'll share results of my **friend** digging into said archives.

Important details about torrent:

- It just content of repository without anything else.
- All files are dated back to 24 February 2022.
- It does not contain git history, mostly just code
- No pre-built binaries for most of software with only few exceptions
- There are no pre-trained ML models with some exceptions

The Moscow Times
30 YEARS
INDEPENDENT NEWS FROM RUSSIA

Russian Billionaires Line Up to Buy Yandex – Reports

May 4, 2023

MOST READ JUST IN

1 OVERNIGHT STRIKE
Russia Says Ukrainian Drones
Moscow, Crimea

2 NO PASSAGE
Russia Blocks Cargo Ship Ov
Traces'

3 POLITICAL PRISONER
Navalny Ally Jailed 9 Years fo

4 MORE MANPOWER
Russia Raises Upper-Age Lim
Reservists

5 MONEY DRAIN

Technology

Russia's Yandex fined for refusing to share user information with security services

Reuters

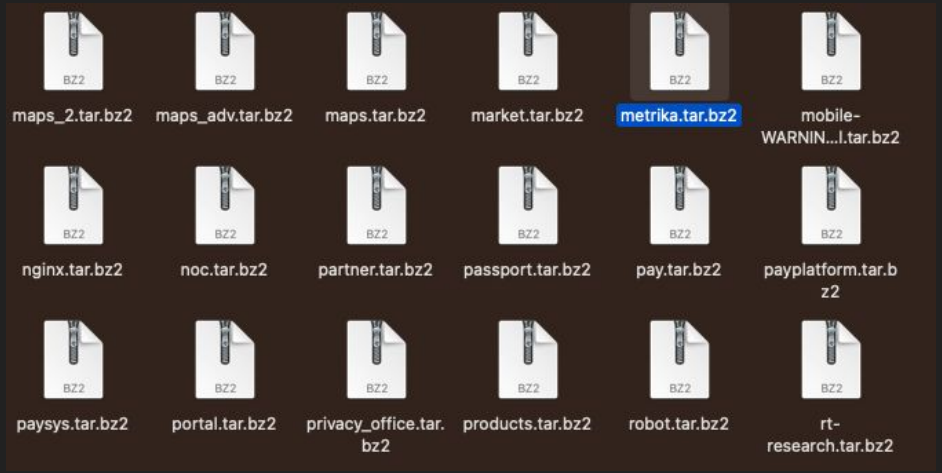
June 18, 2023 3:20 PM PDT · Updated a month ago

Roadmap

- Background on Yandex Leak
- Dive into code:
  - What data Yandex is collecting
  - What Yandex is doing with that data
  - Who Yandex is sharing that data with
- Conclusions and wrap up
- Q&A

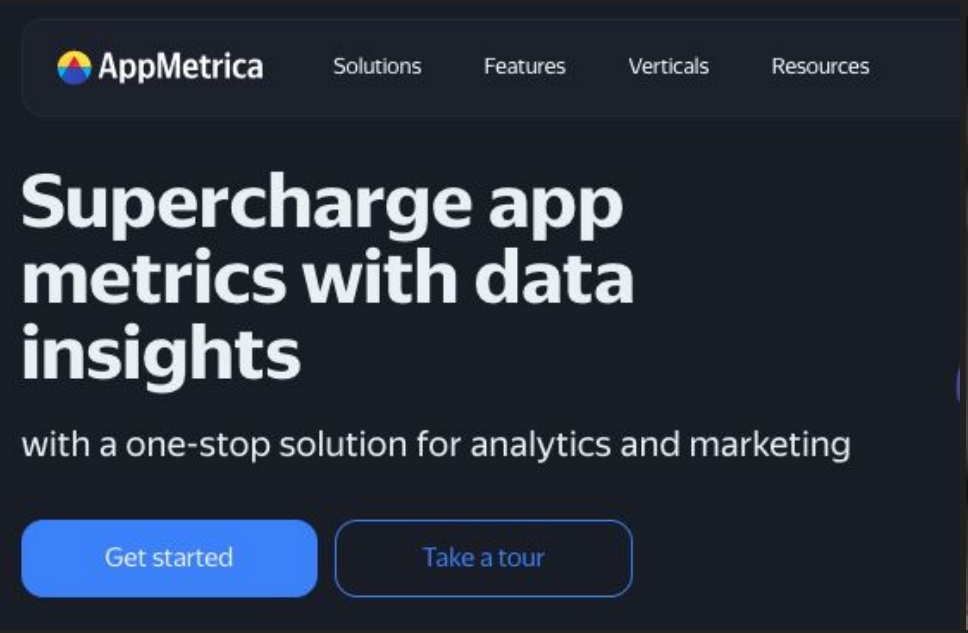
Yandex Codebase

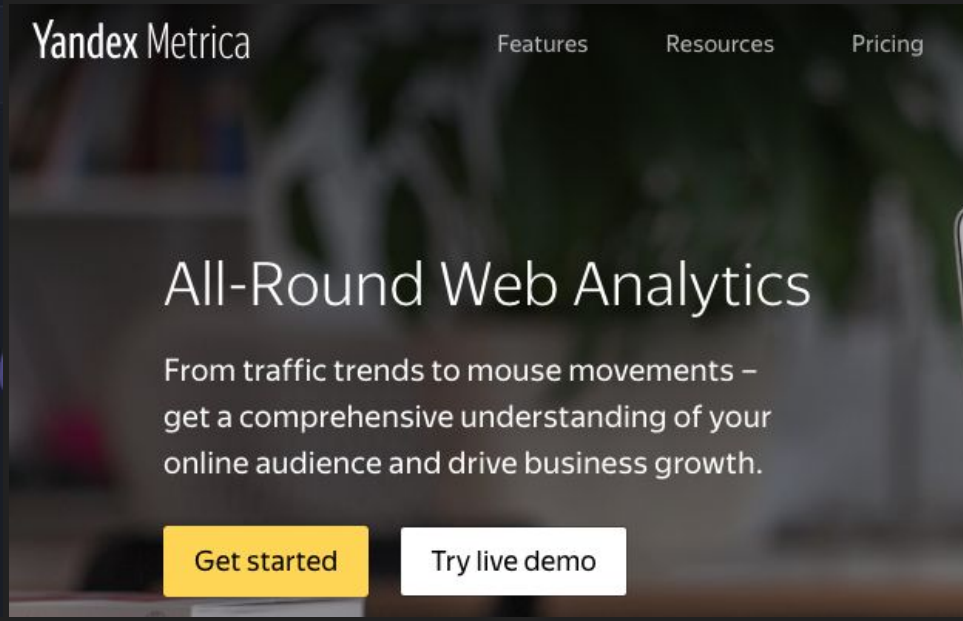
Codebase


maps_2.tar.bz2
maps_adv.tar.bz2
maps.tar.bz2
market.tar.bz2
metrika.tar.bz2
mobile-WARNIN...l.tar.bz2
nginx.tar.bz2
noc.tar.bz2
partner.tar.bz2
passport.tar.bz2
pay.tar.bz2
payplatform.tar.bz2
paysys.tar.bz2
portal.tar.bz2
privacy_office.tar.bz2
products.tar.bz2
robot.tar.bz2
rt-research.tar.bz2

Metrika

Metrika


AppMetrica
Solutions
Features
Verticals
Resources
Supercharge app metrics with data insights
with a one-stop solution for analytics and marketing
Get started
Take a tour


Yandex Metrica
Features
Resources
Pricing
All-Round Web Analytics
From traffic trends to mouse movements – get a comprehensive understanding of your online audience and drive business growth.
Get started
Try live demo

Example Raw Data Fields that AppMetrica Logs

analytics > appmetrica-location-log-anonymizer > convert_log.yql

```
insert into `//home/metrika-logs/anonym-appmetrica-location-log/1d/{table_date}`
with truncate
    select
        String::HexEncode(Digest::Blake2B(`DeviceID`, seed)) as `DeviceID`,
        String::HexEncode(Digest::Blake2B(`ADVID`, seed)) as `ADVID`,
        String::HexEncode(Digest::Blake2B(`IFA`, seed)) as `IFA`,
        String::HexEncode(Digest::Blake2B(`UUID`, seed)) as `UUID`,
        String::HexEncode(Digest::Blake2B(`AndroidID`, seed)) as `AndroidID`,
        `APIKey`,
        `AppBuildNumber`,
        `AppFramework`,
        `AppID`,
        `AppPlatform`,
        `AppVersionName`,
        `Cells_AreConnected`,
        `Cells_CellsIDs`,
        `Cells_CountriesCodes`,
        `Cells_Lacs`,
        `Cells_LastVisibleTimeOffset`,
        `Cells_OperatorsIDs`,
        `Cells_OperatorsNames`,
        `Cells_PhysicalsCellsIDs`,
        `Cells_SignalsStrengths`,
        `Cells_Types`,
        `ChargeType`,
        `ClientIP`,
        `ClientIPHash`,
        `CollectTimestamp`,
        `CollectTimestampBootOffset`,
        `CollectionMode`,
        `DeviceType`,
        `EventID`,
        `IncrementalID`,
        `IsExtraLocationEvent`,
        `IsRooted`,
        `KitBuildNumber`,
        `KitBuildType`,
        `KitVersion`,
```

Ln 117

analytics > appmetrica-location-log-anonymizer > convert_log.yql

```
        `IsExtraLocationEvent`,
        `IsRooted`,
        `KitBuildNumber`,
        `KitBuildType`,
        `KitVersion`,
        `Latitude`,
        `LatitudeLBS`,
        `LocationAltitude`,
        `LocationDirection`,
        `LocationEnabled`,
        `LocationPrecision`,
        `LocationPrecisionLBS`,
        `LocationSource`,
        `LocationSpeed`,
        `LocationTimestamp`,
        `LocationTimestampBootOffset`,
        `Longitude`,
        `LongitudeLBS`,
        `OSApiLevel`,
        `OSVersion`,
        `OperatingSystem`,
        `OriginalCollectTimestamp`,
        `OriginalLocationTimestamp`,
        `ReceiveTimestamp`,
        `RequestID`,
        `SendTimestamp`,
        `Wifi_AreConnected`,
        `Wifi_LastVisibleTimeOffset`,
        `Wifi_Macs`,
        `Wifi_SignalsStrengths`,
        `Wifi_Ssids`,
        `_logfeller_index_bucket`,
        `_logfeller_timestamp`,
        `_rest`,
        `_stbx`,
        `iso_eventtime`,
        `source_uri`,
        `subkey`,
```

Anonymized identifiers

```
91    select
92        String::HexEncode(Digest::Blake2B(`DeviceID`, seed)) as `DeviceID`,
93        String::HexEncode(Digest::Blake2B(`ADVID`, seed)) as `ADVID`,
94        String::HexEncode(Digest::Blake2B(`IFA`, seed)) as `IFA`,
95        String::HexEncode(Digest::Blake2B(`UUID`, seed)) as `UUID`,
96        String::HexEncode(Digest::Blake2B(`AndroidID`, seed)) as `AndroidID`,
```

Location Fields

```
126        `KitVersion`,
127        `Latitude`,
128        `LatitudeLBS`,
129        `LocationAltitude`,
130        `LocationDirection`,
131        `LocationEnabled`,
132        `LocationPrecision`,
133        `LocationPrecisionLBS`,
134        `LocationSource`,
135        `LocationSpeed`,
136        `LocationTimestamp`,
137        `LocationTimestampBootOffset`,
138        `Longitude`,
139        `LongitudeLBS`,
140        `OSApiLevel`,
141        `OSVersion`,
142        `OperatingSystem`,
143        `OriginalCollectTimestamp`,
144        `OriginalLocationTimestamp`,
```

Wifi Fields Collected By AppMetrica

```
147    `SendTimestamp`,
148    `Wifi_AreConnected`,
149    `Wifi_LastVisibleTimeOffset`,
150    `Wifi_Macs`,
151    `Wifi_SignalsStrengths`,
152    `Wifi_Ssids`,
153    `_logfeller_index_bucket`,
```

Those fields in Crypta

graph > fuzzy > lib > yql > export_ssid_devid_day_table.yql

```
26
27  $list_metrika_log = (
28      select coalesce(DeviceID, "") as DeviceID,
29             coalesce(OriginalDeviceID, "") as OriginalDeviceID,
30             $MakeStringList(Wifi_Macs) as Wifi_Macs,
31             $MakeStringList(Wifi_Ssids) as Wifi_Ssids,
32             $MakeIntList(Wifi_SignalsStrengths) as Wifi_SignalsStrengths,
33             $MakeIntList(Wifi_AreConnected) as Wifi_AreConnected
34      from `{source_mmetric_table}`
35      where DeviceID is not null
36  );
```

Dev Id and SSID Associated with Yandex UID

graph > fuzzy > lib > yql > export_ssid_yuids.yql

```
$mobile_all_table = (
    select distinct mmetric_devid, ssid
    from concat({sources})
);

$mmetric_to_devid = (
    select mmetric_devid, devid,
           coalesce(cast(yuid as uint64), 0) as yuid
    from `{source_nolimit_table}`
);
```

Click Event Data Being Matched to Existing Users

core > programs > clicklogd-mobile > src > C event_indexed_pool.h > TEventIndexedPool > GetIndex<TMatchCriteria>()

```
58  private:
59      template <class TMatchCriteria>
60      TIndex<TMatchCriteria>& GetIndex() {
61          if constexpr (std::is_same_v<TMatchCriteria, NMatchCriteria::TAndroidId>) {
62              return AndroidId_;
63          } else if constexpr (std::is_same_v<TMatchCriteria, NMatchCriteria::TAndroidIdMd5>) {
64              return AndroidIdMd5_;
65          } else if constexpr (std::is_same_v<TMatchCriteria, NMatchCriteria::TAndroidIdSha1>) {
66              return AndroidIdSha1_;
67          } else if constexpr (std::is_same_v<TMatchCriteria, NMatchCriteria::TDeviceIdHash>) {
68              return DeviceIdHash_;
69          } else if constexpr (std::is_same_v<TMatchCriteria, NMatchCriteria::TFingerprint>) {
70              return Fingerprint_;
71          } else if constexpr (std::is_same_v<TMatchCriteria, NMatchCriteria::TGoogleAid>) {
72              return GoogleAid_;
73          } else if constexpr (std::is_same_v<TMatchCriteria, NMatchCriteria::TGoogleAidMd5>) {
74              return GoogleAidMd5_;
75          } else if constexpr (std::is_same_v<TMatchCriteria, NMatchCriteria::TGoogleAidSha1>) {
76              return GoogleAidSha1_;
77          } else if constexpr (std::is_same_v<TMatchCriteria, NMatchCriteria::TIfa>) {
78              return Ifa_;
79          } else if constexpr (std::is_same_v<TMatchCriteria, NMatchCriteria::TIfaMd5>) {
80              return IfaMd5_;
81          } else if constexpr (std::is_same_v<TMatchCriteria, NMatchCriteria::TIfaSha1>) {
82              return IfaSha1_;
83          } else if constexpr (std::is_same_v<TMatchCriteria, NMatchCriteria::TWindowsAid>) {
84              return WindowsAid_;
85          } else if constexpr (std::is_same_v<TMatchCriteria, NMatchCriteria::TWindowsAidMd5>) {
86              return WindowsAidMd5_;
87          } else if constexpr (std::is_same_v<TMatchCriteria, NMatchCriteria::TWindowsAidSha1>) {
88              return WindowsAidSha1_;
89          } else if constexpr (std::is_same_v<TMatchCriteria, NMatchCriteria::TYmTrackingId>) {
90              return YmTrackingId_;
91          }
92      }
93
```

Socio-Demographic Attributes for DevID being Updated

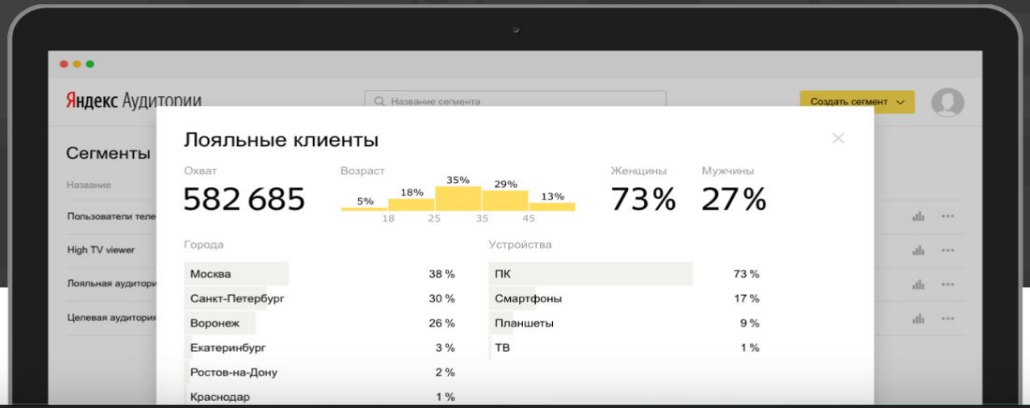
core > programs > socdem-updaterd-mobile > src > UserIdAndInfoParser.cpp > ...

```
            {
                {"0_17", AgeIntervalsCrypta::LessThan18},
                {"18_24", AgeIntervalsCrypta::Between18and24},
                {"25_34", AgeIntervalsCrypta::Between25and34},
                {"35_44", AgeIntervalsCrypta::Between35and44},
                {"45_54", AgeIntervalsCrypta::Between45and54},
                {"55_99", AgeIntervalsCrypta::MoreThan55}
            };
        ::setValue(value, exact_socdem_node, key, json_keys_to_ages_intervals);
    }

    void UserIdAndInfoParser::setValue(
        SexTypesCrypta & value,
        const NYT::TNode & exact_socdem_node,
        const TString key)
    {
        static const std::map<TString, SexTypesCrypta> json_keys_to_sex_types =
            {
                {"f", SexTypesCrypta::Female},
                {"m", SexTypesCrypta::Male}
            };
        ::setValue(value, exact_socdem_node, key, json_keys_to_sex_types);
    }

    std::string UserIdAndInfoParser::parse(const NYT::TNode & user_record)
    {
        const TString & device_id = user_record["appmetrica_devid"].AsString();
        UserInfo user_info;

        const auto & exact_socdem = user_record["exact_socdem"];
        setValue(user_info.age, exact_socdem, "age_segment");
        setValue(user_info.sex, exact_socdem, "gender");

        static const auto tail = getConstTail();

        std::ostringstream buffer;
        buffer <<
            sipHash64(device_id.data(), device_id.size()) << '\t' <<
            static_cast<int>(user_info.age) << '\t' <<
```

To come in

Create segments based on offline and online data

Create Segment



Crypta

Crypta

Yandex COMPANY JOBS FOR DEVELOPERS FOR ADVERTISERS FOR INVESTORS

About History Privacy Press Releases Blog Contact

Technologies /

Crypta

Every day, millions of web users are exposed to banner ads on the pages of Yandex's sites. Advertisers on Yandex can opt to show their ads only to that part of the viewer audience that is potentially interested in seeing them, such as people of a certain age or gender. To enable advertisers to target their ads to a specific audience, Yandex uses its own proprietary behavior analytics technology called Crypta. This technology allows classification of web users based on their online behavior. Their behaviour just has to differ somehow.

Example Segments

CRYPTA

- test
- age_segment_18_20.py
- alice_users.py
- apartment_room_number.py
- apps_users.py
- artists.py
- auto_interactions.py
- avia_travellers.py
- bank_cards.py
- bought_two_tickets.py
- business_travellers.py
- children_age_segment_clarification.py
- compulsory_auto_insurance.py
- connection_type.py
- console_gamers.py
- contest.py
- devices_without_google_services.py
- digital_viewers.py
- direct_clients_by_industry.py
- direct_product_users.py
- disk_users.py
- ecommerce_owners.py
- edadeal_offline_purchases_lal.py
- expensive_car_customers.py
- film_lovers_by_genres.py
- gas_stations.py
- industry_representatives.py
- kbt_customers.py
- kfc_visitors.py
- kinopoisk_logins.py
- kinopoisk_movie_watchers.py

CRYPTA

- kinopoisk_movie_watchers.py
- kz_users.py
- laptop_users.py
- logged_in_for_plus.py
- longterm_interest_mobile_gamers.py
- loyal_to_launcher_install.py
- macos_users.py
- mail_data.py
- manufacturer_phone_owners.py
- marketplaces_ltv_users.py
- mobile_gamers.py
- mobile_operators_users_by_prefix.py
- mobile_operators_users.py
- multidevice_puid.py
- multidevice.py
- music_genres_listeners.py
- nestle_regions.py
- phone_buyers.py
- phone_owners.py
- phone_with_esim_owners.py
- potential_aon_android_users.py
- potential_aon_ios_users.py
- preinstalled_apps.py 3
- prism.py
- proleads.py
- realty_interactions.py
- recent_passport_accounts.py
- score_users_for_telephony.py
- searched_for_phone_numbers.py
- searched_radisson_on_maps.py

CRYPTA

- mobile_operators_users_by_prefix.py
- mobile_operators_users.py
- multidevice_puid.py
- multidevice.py
- music_genres_listeners.py
- nestle_regions.py
- phone_buyers.py
- phone_owners.py
- phone_with_esim_owners.py
- potential_aon_android_users.py
- potential_aon_ios_users.py
- preinstalled_apps.py
- prism.py
- proleads.py
- realty_interactions.py
- recent_passport_accounts.py
- score_users_for_telephony.py
- searched_for_phone_numbers.py 5
- searched_radisson_on_maps.py
- seo_specialists.py
- seo_users.py
- smart_gadgets_customers.py
- smokers.py
- summer_residents.py
- travellers.py
- video_bloggers.py
- want_to_change_the_provider.py
- webmaster.py
- widgets.py
- with_children_by_ages.py

Example Segments

- smart_gadgets_customers.py
- smokers.py
- summer_residents.py
- travellers.py
- video_bloggers.py
- want_to_change_the_provider.py
- webmaster.py
- widgets.py
- with_children_by_ages.py
- va.make

Travellers

profile > runners > segments > lib > coded_segments > travellers.py > ...

```
82
83  INSERT INTO `{output_table}` WITH TRUNCATE
84  SELECT
85      id,
86      id_type,
87      segment_name
88  FROM(
89      SELECT
90          crypta_id AS id,
91          'crypta_id' AS id_type,
92          CASE
93              WHEN Geo::RoundRegionById(region, "country").id != Geo::RoundRegionById(CAST(main_region AS Int32), "country").id THEN 'internati
94              ELSE 'domestic'
95          END AS segment_name,
96          MAX(`date`) AS last_seen,
97          MIN(`date`) AS first_seen,
98          region,
99          week_end_date,
100     FROM $travell_visits
101     GROUP BY region, main_region, crypta_id, week_end_date
102 )
103 WHERE
104     last_seen <= week_end_date AND
105     DateTime::ToDays(DateTime::MakeTimestamp($parse(last_seen)) - DateTime::MakeTimestamp($parse(first_seen))) > 0 AND
106     DateTime::ToDays(DateTime::MakeTimestamp($parse(week_end_date)) - DateTime::MakeTimestamp($parse(first_seen))) <= 7
107 GROUP BY id, id_type, segment_name
108 """
109
110
```

Mail Data

profile > runners > segments > lib > coded_segments > mail_data.py > ...

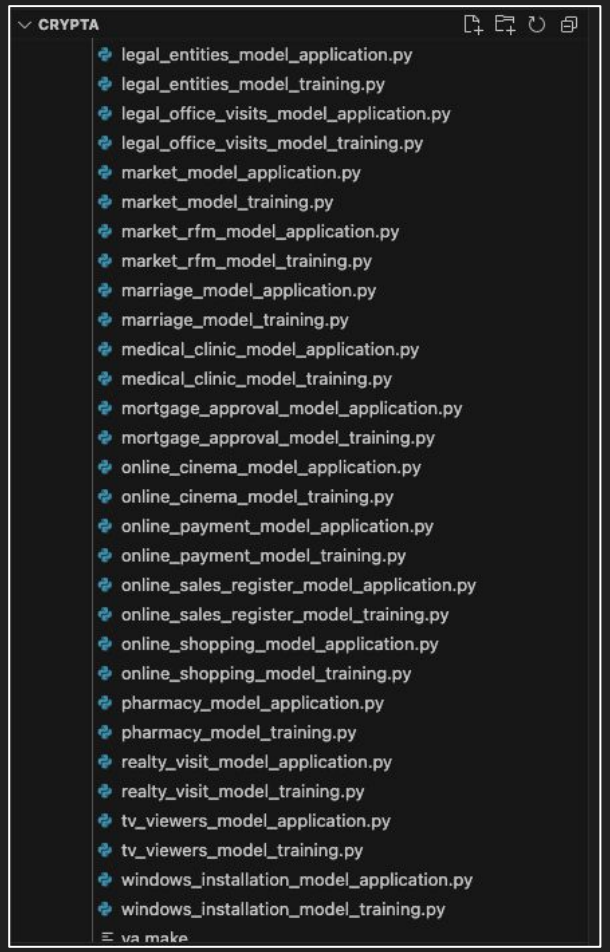
```python
segment_query = """
INSERT INTO `{output_table}` WITH TRUNCATE
SELECT id, id_type, segment_name
FROM `{mail_data_table}`;

INSERT INTO `{sample_table}` WITH TRUNCATE
SELECT
    yandexuid,
    segment_name
FROM (
    SELECT matching.yandexuid AS yandexuid, mail_data.segment_name AS segment_name
    FROM `{mail_data_table}` AS mail_data
    INNER JOIN `{indevice_yandexuid_matching}` AS matching
    USING (id, id_type)
)
GROUP BY yandexuid, segment_name
"""


class PrepareMailSampleForLalSegments(RegularSegmentBuilder):
    keyword = 549
    name_segment_dict = {
        'aviaticket': 1404,
        'boardingpass': 1405,
        'hotel': 1406,
    }
```

Gas Stations

profile > runners > segments > lib > coded_segments > gas_stations.py > ...

```python
class ProcessedDeepVisitLogForGasStations(DayProcessor):
    def requires(self):
        return deep_visits.org_visits_deep_external_input(self.date)

    def process_day(self, inputs, output_path):

        self.yql.query(
            gas_stations_query_template.format(
                organization_categories=config.ORGANIZATION_CATEGORIES,
                deep_visits=inputs.table,
                matching_idfa=get_matching_table('idfa', 'crypta_id'),
                matching_gaid=get_matching_table('gaid', 'crypta_id'),
                name_to_variable=',\n'.join(
                    [u('"{}", "{}")'.format(key, value)
                     for key, value in name_to_variable.iteritems()]
                ),
                output_table=output_path,
            ),
            transaction=self.transaction,
        )
```

Example ML Model Types



Basic example of household details

graph > metrics > household > query.sql

```
    FROM $composition
    GROUP BY $size_to_range(size) AS key;
END DEFINE;

DEFINE SUBQUERY $hh_size_by_crypta_id($title, $predicat) AS
    SELECT
        ($title || key) AS key,
        COUNT(1) AS hh_size_by_crypta_id
    FROM $composition
    WHERE $predicat(size, socdems)
    GROUP BY CAST(Yson::GetLength(Yson::Lookup(data, 'crypta_ids')) AS String) AS key;
END DEFINE;

DEFINE SUBQUERY $hh_by_socdems($title, $predicat) AS
    $hh_socdem = (
        SELECT
            hhid,
            size,
            IF((Yson::LookupInt64(info, 'female') != 0), 'female', Null) AS has_female,
            IF((Yson::LookupInt64(info, 'male') != 0), 'male', Null) AS has_male,
            IF((Yson::LookupInt64(info, 'grand') != 0), 'grand', Null) AS has_old,
            IF((Yson::LookupInt64(info, 'child') != 0), 'child', Null) AS has_child
        FROM $composition
        WHERE $predicat(size, socdems)
    );

    SELECT ($title || groups) AS key, hh_c AS hh_socdem_count
    FROM (
        SELECT groups, SUM(size) AS hh_c
        FROM $hh_socdem
        GROUP BY String::JoinFromList(
            ListSort(AsList(has_female, has_male, has_old, has_child)),
            '_') AS groups
    ) WHERE groups != "";
END DEFINE;
```

AppMetrica being used to pull wifi connection types:

profile > runners > segments > lib > coded_segments > connection_type.py > ...

```python
connection_type_query = """
INSERT INTO `{output_table}` WITH TRUNCATE
SELECT
    id,
    'mm_device_id' AS id_type,
    CASE
        WHEN types == AsSet('3g') THEN '3g'
        WHEN types == AsSet('4g') THEN '4g'
        ELSE '3g_4g'
    END AS segment_name
FROM (
    SELECT
        id,
        ToSet(AGGREGATE_LIST_DISTINCT(segment_name)) AS types
    FROM `{input_table}`
    GROUP BY id
);
"""


class ConnectionType(RegularSegmentBuilder):
    name_segment_dict = {
        '3g': (557, 17823841),
        '4g': (557, 17823853),
        '3g_4g': (557, 17823847),
    }

    number_of_days = 35

    def requires(self):
        return {
            'AppMetrica': LogProcessor(
                ProcessAppMetricaForConnectionType,
                self.date,
                self.number_of_days,
            ),
        }
```

AppMetrica data being used to separate users with common SSIDs (wifi networks)

```python
class ImportSsidMobileMetrikaTask(BaseTask):
    date = DateParameter()
    SSID_THRESHOLD = 20
    YUID_THRESHOLD = 20
    DAYS_IN_MONTH = 7

    def requires(self):
        """
        this tasks must be done to complete this task
        """
        task_list = [
            ImportSsidMobileMetrikaDayTask(date=self.date, target_date=target_date, ssid_threshold=self.SSID_THRESHOLD)
            for target_date in days_range_back(self.date, self.DAYS_IN_MONTH)
        ]
        return task_list
```

AppMetrica data being used to separate users with common SSIDs (wifi networks)

```
def _run(self):
    self.yt.create_table_with_schema(
        self.destination, self.destination_schema, strict=True, recreate_if_exists=True
    )
    with self.yt.TempTable() as unexploded, self.yt.TempTable() as not_unique:
        self.yql.execute(self.query(unexploded), syntax_version=1)
        run_native_reduce(
            reducer_name="NCommonWifiAP::TExploder",
            source=unexploded,
            destination=not_unique,
            proxy=self.yt.proxy,
            transaction=self.yt.transaction_id,
            pool=conf.Yt.POOL,
            title="Explode yandexuids with common wifi access point",
            reduce_by=["ssid"],
        )
        yuid_pair = [conf.Constants.YUID_LEFT, conf.Constants.YUID_RIGHT]
        self.yt.run_sort(not_unique, not_unique, sort_by=yuid_pair)
        run_native_reduce(
            reducer_name="NCommonWifiAP::TUnique",
            source=not_unique,
            destination=self.destination,
            proxy=self.yt.proxy,
            transaction=self.yt.transaction_id,
            pool=conf.Yt.POOL,
            title="Make yandexuids with common wifi access point unique",
            reduce_by=yuid_pair,
        )
        self.yt.run_sort(self.destination, sort_by=yuid_pair)
    self.yt.set(self.destination + "/@generate_date", self.date.isoformat())
```

Sources

graph > fuzzy > lib > config.py > GeoPaths

```
class SourceTypes(object):
    EMAIL_LOGIN = "EMAIL_LOGINS"
    EMAIL_SIMILAR = "EMAIL_SIMILAR"
    GEO_HOMEWORK = "GEO_HOMEWORK"
    HOUSEHOLD = "HOUSEHOLD"
    REQANS_LOG = "REQANS_LOG"
    SSID = "SSID"
```

Search Data

Wifi

Yandex IDs Associated with Email

```
class EmailPaths(object):
    ROOT = ROOT
    # Emails
    BASE = "{root}/email".format(root=ROOT)

    ALL_EMAILS_TABLE = "{base}/all_emails".format(base=BASE)
    ALL_EMAIL_LOGINS_TABLE = "{base}/all_email_logins".format(base=BASE)
    ALL_EMAILS_SORTED_BY_LOGIN = "{base}/all_email_logins.sorted_by_login".format(base=BASE)
    ALL_EMAIL_LOGINS_PAIRS_TABLE = "{base}/all_email_logins.pairs".format(base=BASE)
    ALL_EMAILS_GROUPED_BY_LOGIN = "{base}/all_email_logins.groups".format(base=BASE)
    ALL_YUID_PAIRS_FROM_EMAIL_LOGIN = "{base}/all_yuid_pairs_from_email_logins_matching".format(base=BASE)
    ALL_YUID_PAIRS_FROM_SIMILAR_EMAILS = "{base}/all_yuid_pairs_from_similar_emails".format(base=BASE)

    ALL_EMAILS_TABLE_SCHEMA = {"email": "string", "yuids": "any"}
    ALL_EMAIL_LOGINS_TABLE_SCHEMA = {"login": "string", "email": "string", "yuids": "any"}
    ALL_EMAILS_SORTED_BY_LOGIN_SCHEMA = {"login": "string", "email": "string", "yuids": "any"}
    ALL_EMAILS_GROUPED_BY_LOGIN_SCHEMA = {"login": "string", "all_emails": "any", "howmany": "uint64"}
    ALL_EMAIL_LOGINS_PAIRS_TABLE_SCHEMA = {
        "email_1": "string",
        "email_2": "string",
        "login": "string",
        "yuids_1": "any",
        "yuids_2": "any",
    }
    ALL_YUID_PAIRS_FROM_EMAIL_LOGIN_SCHEMA = {
        Constants.YUID_LEFT: ("uint64", True),
        Constants.YUID_RIGHT: ("uint64", True),
        "match": "any",
    }
    ALL_YUID_PAIRS_FROM_SIMILAR_EMAILS_SCHEMA = {
        Constants.YUID_LEFT: "uint64",
        Constants.YUID_RIGHT: "uint64",
        "email_left": "string",
        "email_right": "string",
        "fragment": "string",
    }
```

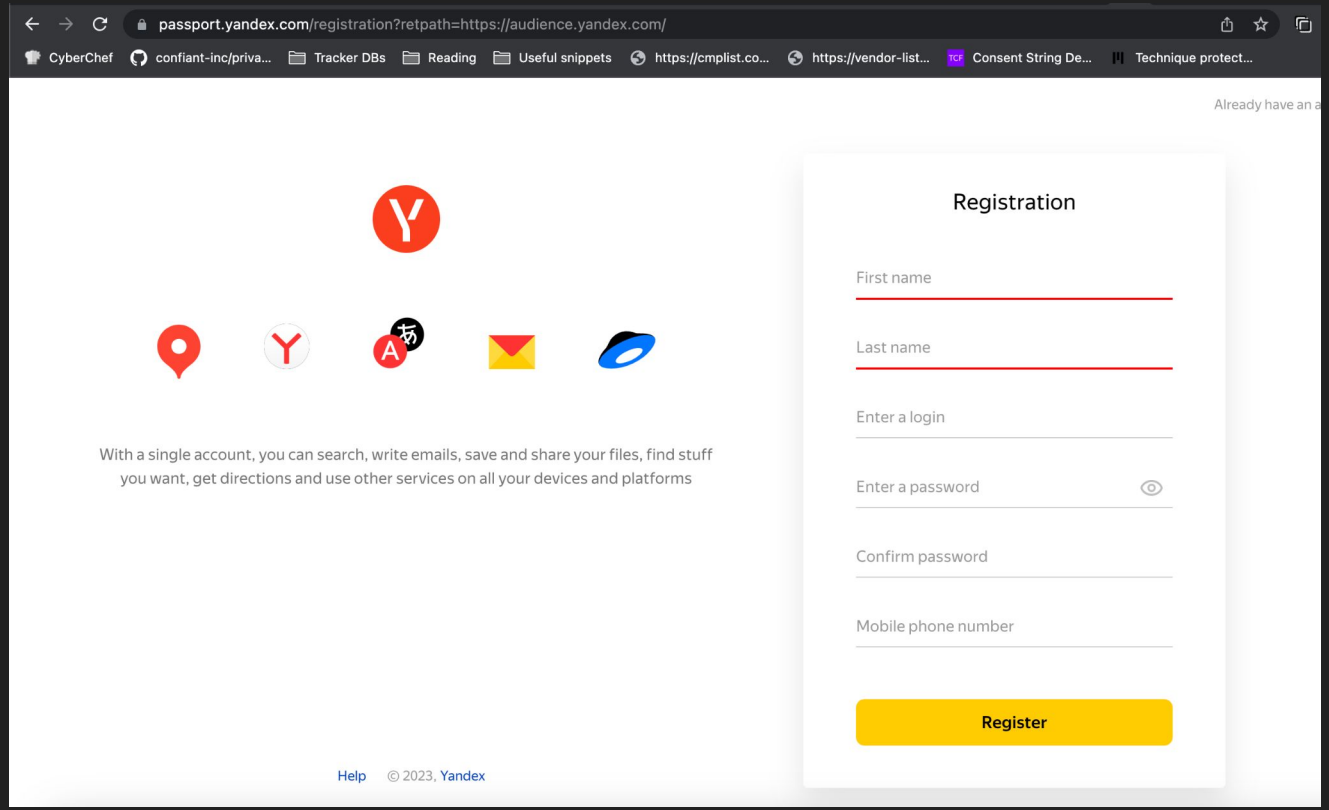
Login Data

graph > fuzzy > lib > tasks > sources > visitlog_logins > extract.py > ...

```python
    def filter_rare_logins_options(self):
        return TFilterRareLoginsOptions(Threshold=self.threshold).SerializeToString()

    @property
    def filter_keys_options(self):
        return TFilterKeysOptions(
            Keywords=[
                "login",
                "user",
                "userid",
                "clientid",
                "uid",
                "email",
                "emailhash",
                "\u043b\u043e\u0433\u0438\u043d",
                "computerid",
                "cid",
                "suserid",
            ]
        ).SerializeToString()
```

Extracting multiple types of identifiers

Passport


passport.yandex.com/registration?retpath=https://audience.yandex.com/
CyberChef
confiant-inc/priva...
Tracker DBs
Reading
Useful snippets
https://cmplist.co...
https://vendor-list...
Consent String De...
Technique protect...
Already have an a
With a single account, you can search, write emails, save and share your files, find stuff you want, get directions and use other services on all your devices and platforms
Registration
First name
Last name
Enter a login
Enter a password
Confirm password
Mobile phone number
Register
Help
© 2023, Yandex

Passport User ID Associated with Phone

graph > data_import > passport > lib > query > passport.sql

```
33
34  $out_login_tbl = $soup_output_dir || $edge(IdType::PUID(), IdType::PHONE(), SourceType::PASSPORT_PROFILE(), LogSource::PASSPORT_PHONE_DUMP());
35  INSERT INTO $out_login_tbl WITH TRUNCATE
36  SELECT
37      id1,
38      IdType::PUID() AS id1Type,
39      id2,
40      IdType::PHONE() AS id2Type,
41      SourceType::PASSPORT_PROFILE() AS sourceType,
42      LogSource::PASSPORT_PHONE_DUMP() AS logSource,
43      ListCreate(String) AS dates
44  FROM (
45      SELECT DISTINCT puid, phone
46      FROM $log FLATTEN LIST BY phones AS phone
47  ) WHERE Identifiers::IsSignificantPhone(phone)
48  GROUP BY
49      puid AS id1,
50      phone AS id2
51  ;
52
```

graph > data_import > passport > tests > fixtures > {} passport.json > ...


```
{"uid": "11111", "login": "aashinova"}
{"uid": "11112", "login": "andrei-ponomareff-1997"}
{"uid": "11113", "login": "anoshko-av"}
{"uid": "11114", "login": "bars12@161.ru"}
{"uid": "11115", "login": "ev0ngertlt"}
{"uid": "11116", "login": "evarcher"}
{"uid": "11117", "login": "lagutin2008"}
{"uid": "11118", "login": "login-for-avito"}
{"uid": "11119", "login": "modsever"}
{"uid": "11120", "login": "mouradian"}
{"uid": "11121", "login": "perschina-olga2013"}
{"uid": "11122", "login": "r.amiraslanov@dveri.ru"}
{"uid": "11123", "login": "saprovec2015"}
{"uid": "11124", "login": "stoltat"}
{"uid": "11125", "login": "sveta-aleshina2015"}
{"uid": "11126", "login": "watchradius"}
{"uid": "11127", "login": "watchradius"}
{"uid": "11128", "login": "sveta-aleshina2015"}
{"uid": "11129", "login": "watchradiusmob"}
{"uid": "123456", "login": "abc123", "phone_numbers": []}
{"uid": "123457", "login": "abc127", "phone_numbers": ["+1234567890", "+71111234567"]}
{"uid": "123458", "login": "", "phone_numbers": ["+9393939393", "+71202020201"]}
{"uid": "134614616", "login": "roscosh8"}
{"uid": "134648582", "login": "e222mn"}
{"uid": "15033290", "login": "mouradian"}
{"uid": "194502233", "login": "ingvr80"}
{"uid": "2687", "login": "govshit"}
{"uid": "2687", "login": "GOVSHIT"}
{"uid": "76667777", "login": "g8jkqqaaaaaaaaah"}
{"uid": "766679666", "login": "d6fqqaaaaaaaaah"}
{"uid": "766679777", "login": "mdmozn45"}
```

Crypta - Geo graphs

Using lat/long data associated with “predicted home”, linked to Yandex UID

graph > fuzzy > lib > tasks > sources > geo > C geo_operations.h

```
void Do(TTableReader<TNode>* input, TTableWriter<TGeoSquare>* output)
override {
    for (; input->IsValid(); input->Next()) {
        const auto& row = input->GetRow();
        if (not IsRowValid(row)) {
            continue;
        }

        const ui64 yandexuid = FromString<ui64>(row["yandexuid"].AsString());
        const auto& homeCoordinates = row["predicted_home"];
        const auto latitude = homeCoordinates["latitude"].AsDouble();
        const auto longitude = homeCoordinates["longitude"].AsDouble();
        const auto& square = computeSquare({.Lat = latitude, .Lon = longitude}, State->radius());

        /**
         *    + + -
         *    + + -
         *    + - -
         */
        for (int beltOffset : {-1, 0, 1}) {
            for (int sqOffset : {-1, 0}) {
                if (beltOffset == -1 && sqOffset == 0) {
                    continue;
                }
                const ui64 square_idx = ConvertSquareToIdx({.Belt = square.Belt + beltOffset, .Sq = square.Sq + sqOffset
                TGeoSquare out;
                out.set_yandexuid(yandexuid);
                out.set_lat(latitude);
                out.set_lon(longitude);
                out.set_squareidx(square_idx);
                output->AddRow(out);
            }
        }
    }
}
```

Crypta - Geo graphs

Then using that data to find literal neighbors within a certain radius of that home

graph > fuzzy > lib > tasks > sources > geo > C geo_operations.h

```
class TFindNeighbors: public IReducer<TTableReader<TGeoSquare>, TTableWriter<TNeighborsDistance>> {
public:
    TFindNeighbors()
            : State()
    {
    }

    TFindNeighbors(const TBuffer& buffer)
            : State(buffer)
    {
    }

    void Do(TTableReader<TGeoSquare>* input, TTableWriter<TNeighborsDistance>* output) override {
        const double radius = State->radius();
        TVector<TGeoSquare> candidates;
        for (; input->IsValid(); input->Next()) {
            const auto& row = input->GetRow();
            candidates.push_back(row);
        }
        for (auto i : xrange(candidates.size())) {
            for (auto j : xrange(i + 1, candidates.size())) {
                const auto& left = candidates.at(i);
                const auto& right = candidates.at(j);
                if (left.yandexuid() == right.yandexuid()) {
                    continue;
                }
                double distance = computeDistance({.Lat = left.lat(), .Lon  = left.lon()}, {.Lat = right.lat(), .Lon = righ
                if (distance > radius) {
                    continue;
                }
                TNeighborsDistance out;
                out.set_distance(distance);
                out.set_yandexuidleft(Min(left.yandexuid(), right.yandexuid()));
                out.set_yandexuidright(Max(left.yandexuid(), right.yandexuid()));
                output->AddRow(out);
            }
```

AppMetrica and Taxi data being used generate segments about households with children:

```
        self.yql.query(
            app_metrica_query.format(
                devid_by_app_table=self.input()['DevidByApp'].table,
                output_table=with_children_by_app_table,
                app_to_segment_name='\n'.join(app_segment_name_tuples),
            ),
            transaction=self.transaction,
        )

    def build_segment(self, inputs, output_path):
        with self.yt.TempTable() as taxi_puid_table, \
                self.yt.TempTable() as app_metrica_table:
            self.yt.run_map(
                extract_children_from_taxi,
                inputs['TaxiData'].table,
                taxi_puid_table,
            )

            self.prepare_with_children_by_app(app_metrica_table)

            self.yql.query(
                with_children_query_template.format(
                    metrics_table=inputs['ProcessedMetrics'].table,
                    reqans_table=inputs['ProcessedReqans'].table,
                    app_metrica_table=app_metrica_table,
                    taxi_data_table=taxi_puid_table,
                    id_to_crypta_id_table=config.VERTICES_NO_MULTI_PROFILE,
                    crypta_id_to_hhid_table=config.HOUSEHOLD_CRYPTA_ID_TO_HHID,
                    yandexuid_to_hhid_table=config.HOUSEHOLD_REVERSED_TABLE,
                    hhid_to_yandexuid_table=config.HOUSEHOLD_ENRICH_TABLE,
                    output_table=output_path,
                ),
```

ID mapping associations:

```
taxi_data_table=taxi_puid_table,
id_to_crypta_id_table=config.VERTICES_NO_MULTI_PROFILE,
crypta_id_to_hhid_table=config.HOUSEHOLD_CRYPTA_ID_TO_HHID,
yandexuid_to_hhid_table=config.HOUSEHOLD_REVERSED_TABLE,
hhid_to_yandexuid_table=config.HOUSEHOLD_ENRICH_TABLE,
output_table=output_path,
```

Profiles integrate biometric data, most likely from smart speakers that use Yandex's Alice smart assistant

Yandex

COMPANY JOBS FOR DEVELOPERS FOR ADVERTISERS FO

About History Privacy Press Releases Blog Contact

Press Releases / 2022 /

Yandex Launches Smart Devices With Alice in Uzbekistan

Internet, November 22, 2022. Uzbekistan's local
Stations with Alice. Upon purchasing a smart sp

Stations are smart speakers with Alice. A single Yandex Plus subscription allows you to play music, podcasts, or playlists with personal recommendations on Yandex Station. Alice, the voice assistant on board, can entertain children with an educational game or compose a fairy tale together. She will tell you about the weather or remind you to buy groceries. Alice loves talking and will hold a conversation with ease: currently, in Russian only.

Possible Children by Voice

profile > runners > segments > lib > coded_segments > children_age_segment_clarification.py > ...

```
clarify_children_yql_template = """
$possible_children_by_voice = (
    SELECT `uuid`, TableName() AS `date`, '0_12' AS segment_name
    FROM RANGE(`{biometry_folder}`, `{biometry_first_date}`, `{biometry_last_date}`)
    WHERE bio_child > 0.8
);

$possible_children_by_voice = (
    SELECT DISTINCT `uuid`, `date`, segment_name
    FROM $possible_children_by_voice
);

$possible_children_by_voice = (
    SELECT `uuid`, segment_name
    FROM $possible_children_by_voice
    GROUP BY `uuid`, segment_name
    HAVING COUNT(*) >= 2
);

$sources_new_age = (
    SELECT matching.cryptaId AS cryptaId,
        CASE
            WHEN socdem_storage.birth_date > '{thirteenth_birthday}' THEN '0_12'
            WHEN '{thirteenth_birthday}' >= socdem_storage.birth_date AND
                socdem_storage.birth_date > '{eighteenth_birthday}' THEN '13_17'
            ELSE '18_99'
        END AS segment_name
    FROM `{socdem_storage_table}` AS socdem_storage
    INNER JOIN `{id_to_crypta_id_table}` AS matching
    ON socdem_storage.id == matching.id AND socdem_storage.id_type == matching.id_type
    WHERE socdem_storage.birth_date is not Null
    UNION ALL
    SELECT matching.cryptaId AS cryptaId, biometry.segment_name AS segment_name
    FROM $possible_children_by_voice AS biometry
    INNER JOIN `{id_to_crypta_id_table}` AS matching
    ON biometry.`uuid` == matching.id
    WHERE matching.id_type == 'uuid'
```

UI for Infographics Card

```
27
28      const marriedText = convertMarriedToSingleText(exactDemographics.gender, married);
29      const incomeText = convertIncomeSegmentToText(exactDemographics.income);
30      const hasChildrenText = convertHasChildrenToText(hasChildren);
31
32      return (
33          <div className="BasicInfoGraphics">
34              <img alt="" className="BasicInfoGraphics-Image" src={images[exactDemographics.gender]}/>
35              <div className="BasicInfoGraphics-Bubble BasicInfoGraphics-Bubble_family">{marriedText}</div>
36              <div className="BasicInfoGraphics-Bubble BasicInfoGraphics-Bubble_income">{incomeText}</div>
37              <div className="BasicInfoGraphics-Bubble BasicInfoGraphics-Bubble_children">{hasChildrenText}</div>
38              <div className="BasicInfoGraphics-Interest BasicInfoGraphics-Interest_first">
39                  <div className="BasicInfoGraphics-InterestIcon"
40                       style={{ backgroundImage: `url(${interestIcons[0]})` }}/>
41              </div>
42              <div className="BasicInfoGraphics-Interest BasicInfoGraphics-Interest_second">
43                  <div className="BasicInfoGraphics-InterestIcon"
44                       style={{ backgroundImage: `url(${interestIcons[1]})` }}/>
45              </div>
46              <div className="BasicInfoGraphics-Interest BasicInfoGraphics-Interest_third">
47                  <div className="BasicInfoGraphics-InterestIcon"
48                       style={{ backgroundImage: `url(${interestIcons[2]})` }}/>
49              </div>
50          </div>
51      );
52  }
```

- icons
  - apps
  - interests
    - agro.svg
    - animals.svg
    - appliances.svg
    - beauty.svg
    - business.svg
    - clothes.svg
    - construction.svg
    - education.svg
    - electronics.svg
    - entertainments.svg
    - family.svg
    - finance.svg
    - food.svg
    - gifts.svg
    - index.js
    - job.svg
    - realty.svg
    - rest.svg
    - sport.svg
    - stationery.svg
    - telecom.svg
    - transport.svg

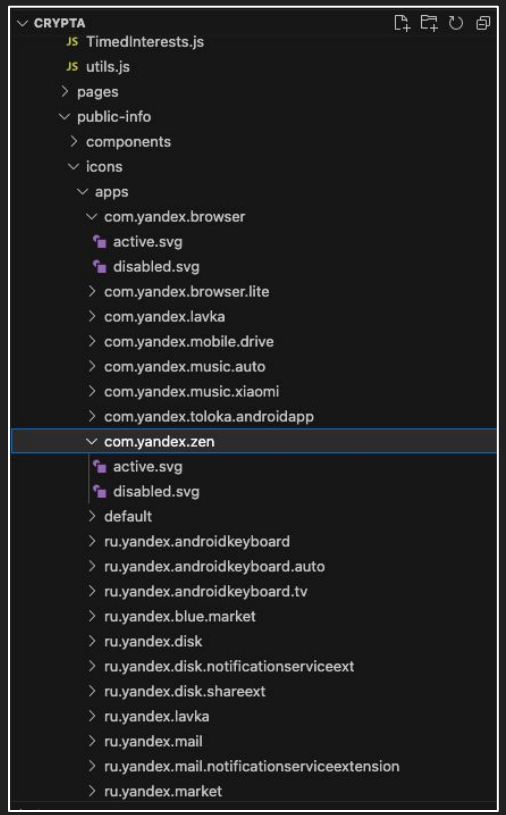
Search Profile by ID

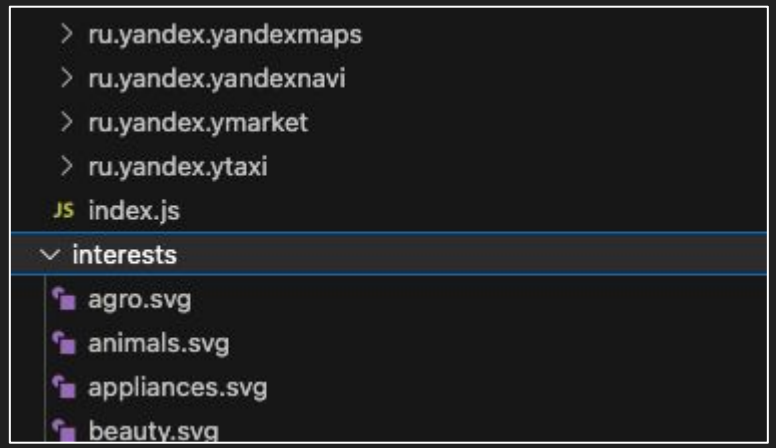
web > portal > src > graph > search > JS SearchPanel.js > ...

```
112            return (
113                <div key={"inputs-" + suffix} className="input-group">
114                    <div style={{display: showIdInput ? "block" : "none"}}>
115                        <ValueInput
116                            key="id_value"
117                            placeholder="[id value]"
118                            value={parameters["idValue" + suffix]}
119                            onChange={changeParameter("uid" + suffix)}
```

```
return (
    experiments.status !== 403 && (
        <>
            <div className="experiments-bar">
                <div className="experiments-select-uid-type">
                    <RadioButton
                        value={activeUid}
                        size="s"
                        view="default"
                        className="select-sorting"
                        onChange={(event) => selectUidType(event.target.value)}
                        options={[
                            { value: "uid", children: t("by") + " yandexuid" },
                            { value: "cryptaId", children: t("by") + " CryptaID" },
                        ]}
```

UI - Available App Icons





Ids Associated with Social Media Accounts

web > portal > src > public-info > sections > GraphSection > JS GraphSection.js > GraphSection >

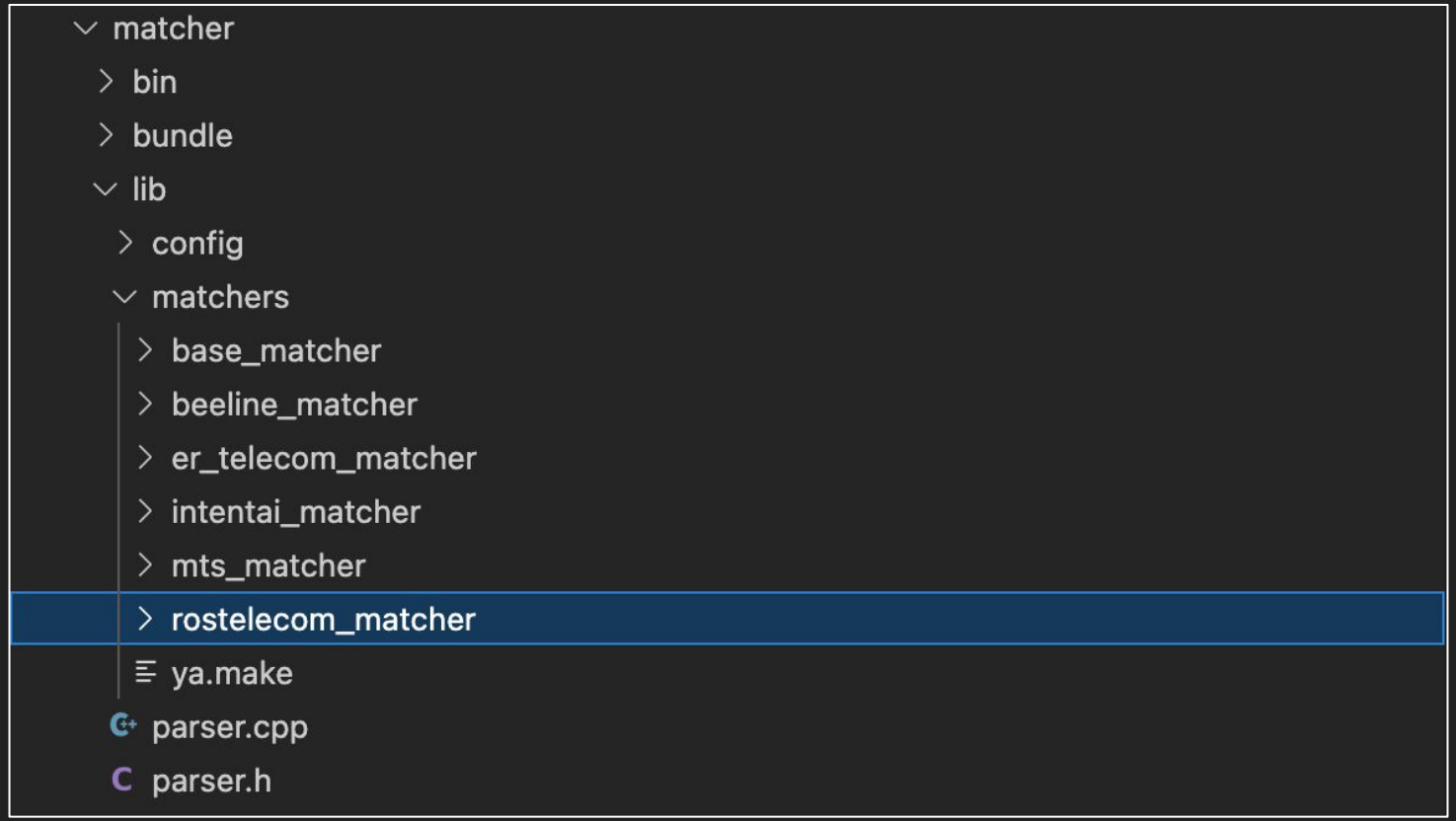
```javascript
import React, { useEffect, useMemo, useState } from "react";
import { useSelector } from "react-redux";
import { getPublicGraph, getPublicGraphLoading } from "../../store/selectors";
import { Graph, GraphSkeleton } from "../../components/Graph/Graph";
import { Section } from "../../components/Section/Section";
import { getServiceIcon } from "../../icons/services";
import { getAppIcon } from "../../icons/apps";

import "./GraphSection.scss";

import noData from "./no-data.svg";

const IMAGE_SIZE_XS = 12;
const IMAGE_SIZE_S = 36;
const IMAGE_SIZE_M = 56;
const IMAGE_SIZE_L = 80;

const NODE_MAPPING = {
    email: {
        imageSize: IMAGE_SIZE_M,
        imageHref: "mail",
    },
    yandexuid: {
        imageSize: IMAGE_SIZE_XS,
        imageHref: "yandexuid",
    },
    idfa: {
        imageSize: IMAGE_SIZE_L,
        imageHref: "ios",
    },
    gaid: {
        imageSize: IMAGE_SIZE_L,
        imageHref: "android",
    },
    oaid: {
        imageSize: IMAGE_SIZE_L,
        imageHref: "android",
    },
    login: {
        imageSize: IMAGE_SIZE_M,
        imageHref: "key",
    },
    puid: {
        imageSize: IMAGE_SIZE_M,
        imageHref: "key",
    },
    instagram_login: {
        imageSize: IMAGE_SIZE_M,
        imageHref: "instagram"
    },
    instagram_id: {
        imageSize: IMAGE_SIZE_M,
```

web > portal > src > public-info > sections > GraphSection > JS GraphSection.js > GraphSection > useEffe

```javascript
    instagram_id: {
        imageSize: IMAGE_SIZE_M,
        imageHref: "instagram"
    },
    fb_id: {
        imageSize: IMAGE_SIZE_M,
        imageHref: "facebook"
    },
    ok_id: {
        imageSize: IMAGE_SIZE_M,
        imageHref: "ok"
    },
    vk_id: {
        imageSize: IMAGE_SIZE_M,
        imageHref: "vk"
    },
    vk_name: {
        imageSize: IMAGE_SIZE_M,
        imageHref: "vk"
    },
    kp_id: {
        imageSize: IMAGE_SIZE_M,
        imageHref: "kinopoisk"
    },
}

function getNodeMapping(item) {
    if (item.idType === 'uuid') {
        // App
        return { imageHref: item.icon, imageSize: IMAGE_SIZE_S };
    }

    return NODE_MAPPING[item.icon] ?? { imageHref: "default", imageSize: IMAGE_SIZE_XS };
}

function getImage(item) {
    const disabled = !item.isActive;

    if (item.idType === 'uuid') {
        return getAppIcon(item.imageHref, disabled)
            .catch(() => getAppIcon("default", disabled));
    }

    return getServiceIcon(item.imageHref, disabled)
        .catch(() => getServiceIcon("default", disabled));
}
```

Matcher

Matcher


matcher
bin
bundle
lib
config
matchers
base_matcher
beeline_matcher
er_telecom_matcher
intentai_matcher
mts_matcher
rostelecom_matcher
ya.make
parser.cpp
parser.h

Rostelecom Matcher

ext_fp > matcher > lib > matchers > rostelecom_matcher > rostelecom_matcher.cpp

```cpp
TConnection TRostelecomMatcher::MakeConnection(const TFpEvent& event) {
    return {
        .Ip = event.GetIp(),
        .Port = event.GetPort(),
        .Timestamp = event.GetUnixtime(),
        .Domain = NMcDomain::GetMcDomainForRostelecom(event.GetDuid()),
    };
}

void TRostelecomMatcher::AddConnection(const TFpEvent& event) {
    auto connection = MakeConnection(event);

    Stats.Count->Add("events.incoming.rostelecom.count");
    Request += TStringBuilder() << connection.Ip << '\t'
                                << connection.Port << '\t'
                                << connection.Timestamp << '\t'
                                << connection.Domain << '\n';
}

TMatches TRostelecomMatcher::GetMatches() {
    if (Request.length() == 0) {
        return TMatches();
    }
    const auto& requestId = CreateGuidAsString();
    Log->info("Rostelecom request {} body:\n{}", requestId, Request);

    NNeh::TMessage message(GetApiUrl(), "");
    Y_ENSURE(NNeh::NHttp::MakeFullRequest(message, "", Request, "text/plain"), "Failed to build request to Rostelecom API");

    Stats.Count->Add("api.calls.rostelecom.count");
    const auto& resp = MakeRequest(Client, message, TDuration::MilliSeconds(Config.GetApiCallTimeoutMs()), "Rostelecom", requestId, Log);

    return ParseResponse(resp->Data);
}

TString TRostelecomMatcher::GetApiUrl() const {
    return "post://" + Config.GetApiUrl();
```

Rostelecom Matcher

ext_fp > matcher > lib > matchers > rostelecom_matcher > rostelecom_matcher.cpp

```
22  TConnection TRostelecomMatcher::MakeConnection(const TFpEvent& event) {
23      return {
24          .Ip = event.GetIp(),
25          .Port = event.GetPort(),
26          .Timestamp = event.GetUnixtime(),
27          .Domain = NMcDomain::GetMcDomainForRostelecom(event.GetDuid()),
28      };
29  }
```

Rostelecom Matcher

```
41  TMatches TRostelecomMatcher::GetMatches() {
42      if (Request.length() == 0) {
43          return TMatches();
44      }
45      const auto& requestId = CreateGuidAsString();
46      Log->info("Rostelecom request {} body:\n{}", requestId, Request);
47
48      NNeh::TMessage message(GetApiUrl(), "");
49      Y_ENSURE(NNeh::NHttp::MakeFullRequest(message, "", Request, "text/plain"), "Failed to build request to Rostelecom API");
50
51      Stats.Count->Add("api.calls.rostelecom.count");
52      const auto& resp = MakeRequest(Client, message, TDuration::MilliSeconds(Config.GetApiCallTimeoutMs()), "Rostelecom", requestId, Log);
53
54      return ParseResponse(resp->Data);
55  }
```

Test Result Data

ext_fp > matcher > bin > test > canondata > {} result.json > [ ] test_matcher.test_matcher > {} 1


```
{
    "test_matcher.test_matcher": [
        {
            "duid": 16999999761000006,
            "ext_id": "fake_ertelecom_id_for_5.3.100.0",
            "ext_source": "ertelecom",
            "hitlogid": 100506,
            "ip": "5.3.100.0",
            "log_type": "bs-watch-log",
            "logid": 0,
            "original_domain": "domain-6.ru",
            "port": 5555,
            "rtmr_timestamp": 1699999977,
            "unixtime": 1699999970,
            "user_agent": "Mozilla/5.0 (Windows NT РУС)",
            "watchid": 200000000000000006,
            "yuid": 10061699999976
        },
        {
            "duid": 16999999861000016,
            "ext_id": "mts_id_for_160.1.2.4",
            "ext_source": "mts",
            "hitlogid": 100516,
            "ip": "160.1.2.4",
            "log_type": "bs-watch-log",
            "logid": 0,
            "original_domain": "domain-16.ru",
            "port": 4444,
            "rtmr_timestamp": 1699999987,
            "unixtime": 1699999970,
            "user_agent": "Mozilla/5.0 (Windows NT РУС)",
            "watchid": 200000000000000016,
            "yuid": 10161699999986
        },
        {
            "duid": 16999999931000003,
            "ext_id": "fake_ertelecom_id_for_5.3.62.0",
            "ext_source": "ertelecom",
            "hitlogid": 100503,
            "ip": "5.3.62.0",
            "log_type": "bs-watch-log",
            "logid": 0,
            "original_domain": "domain-3.ru",
            "port": 2222,
            "rtmr_timestamp": 1699999994,
            "unixtime": 1699999990,
            "user_agent": "Mozilla/5.0 (Windows NT РУС)",
            "watchid": 200000000000000003,
            "yuid": 10031699999993
        },
```

Roadmap

- Background on Yandex Leak
- Dive into code:
  - What data Yandex is collecting
  - What Yandex is doing with that data
  - Who Yandex is sharing that data with
- Conclusions and wrap up
- Q&A

Conclusion

Wrap Up

- Yandex has access to a broad international reach of data and it has been evasive about what it can do with that data
- A small amount of data can say a lot when it is matched to entries from a company’s other data sources and analyzed
- Yandex has code to sync some of its data with a Russian-state owned entity

Takeaways

- Anonymization is very easily undone when data gets combined with pools from other sources that may contain identifying data
- Pay attention to who runs your SDKs, what data points they collect, and where they send your user data.
- Who gets access to a company’s user data when its assets are sold, the geopolitical climate changes, or a government tightens its control?

CONFIANT

Q&A

Link to Write Up:
https://bit.ly/455utBP