EDMO
European Digital Media Observatory

European Digital Media Observatory

Platform Datasets

Challenges, Insights, and Examples for Researchers under Article 40 of the Digital Services Act

Jeff Allen
Spencer Gurley
Matt Motyl

August 2025

Platform Datasets. Challenges, Insights, and Examples for Researchers under Article 40 of the Digital Services Act

The European Digital Media Observatory has received funding from the European Union under contract number LC-01935415

Abstract

This report aims to assist researchers studying risks and harms outlined in the Digital Services Act and the Code of Practice on Disinformation by enabling an understanding of the relevant data that exists within platforms and that which is accessible via APIs–and the relationships between internal data and public facing APIs.

1. Introduction

Very large online platforms and search engines (VLOPSEs) collect both an incredible amount and a wide variety of data that even newly hired employees can struggle with as they learn the internal data systems. The structure of the data–and the data warehouse that the companies use to store all their data–can vary from company to company. Some might be chaotic, with data stored in tables that employees organically produce over time, and some might be more standardized and structured with employees required to follow a strict framework. In either situation, it can still be a challenge to understand the potentially hundreds or thousands of tables that VLOPSEs maintain in their data warehouses.

Given the role of online platforms in our lives, it is not surprising that civil society and research institutions recognize the importance of this data, and in particular how it can help us better understand the risks that online platforms can pose to people, societies, and democracies. This data can also tell us how effectively the company is managing and minimizing those risks. Researchers and civil societies need to have access to datasets, both historical and real time, for studying the scale, cause, and nature of risks from the platforms. They need to be able to monitor the information environment in real time, especially around critical societal events like elections.

Article 40 of the Digital Services Act (DSA) allows for significant researcher access to VLOPSE data, creating the opportunity to answer research questions surrounding the systemic risks outlined in the DSA. This paper aims to equip vetted academic and civil society researchers with the understanding and tools necessary to best utilize their access to the data for the public good.
Specifically, this paper aims to:

1. Describe what types of data VLOPSEs collect,
2. Describe how these data are stored,
3. Describe how these data are used,
4. Provide examples of how that data can be used to answer questions,
5. Review data access APIs from VLOPSEs, and
6. Provide examples of how researchers may map the available data to the risks delineated in the Digital Services Act and the Code of Practice.

2. What Data Do Platforms Collect?

Platforms collect a large amount of personal data, network data, advertising data, and engagement data from their users. There are six main categories of data they collect.

1. Data explicitly provided by the user (user generated content, name, and other personal information),
2. Data extracted from the user's devices (IP address, geolocation, GPS coordinates) and cross-app tracking (browser cookies),
3. Data generated by user behavior (engaging with content, commenting, time spent watching content, logins, survey responses),
4. Data generated about a user based on other users' behaviors towards that user (reports, blocks, account follow request rejections, clicks on profile),
5. Inferences about the user based on data (machine learning algorithms or AI creating data), and
6. External data either created by the company or acquired through a third party (shopping behavior, websites visited, credit scores, property data, political donation history).

Not every platform collects the same amount of data, and some of the data depends on the exact nature of the platform. In addition, not all data used by the platforms actually comes from their users. For example, web crawls can be used to create datasets, such as PageRank and web authority scores, which platforms could use.

More examples of all types of data appear in the table below.

User Shared / Provided	Extracted	Created	Behaviors	Received from other users	Inferred	Purchased
Name	Devices used	Profile picture	Views	Views	Interests (e.g., cat videos, political	Shopping behavior
Vanity	Phone number	Pictures	Clicks	Clicks on profile	Monetizable	Income estimate
Phone number	GPS Location	Reels	Searches	Clicks on posts	Estimated value	Net worth
Address	Overlap between their device and other	Shorts	Edits of posts/Comments	Clicks on comments	Politics	Credit score range

User Shared / Provided	Extracted	Created	Behaviors	Received from other users	Inferred	Purchased
Email	IP addresses	Video posts	Time spent	Time viewing their content	Gender	Device preferences
Sex/Gender	Creation Time	Text posts	Sessions	Got reported	Sexual orientation	Home information
Sexual orientation	Whether other accounts use same	Comments	Session length	Blocks	Race/ethn icity	Property data
Relationship status	Language settings on device	Search queries	Scroll time		Religion	Magazine subscriptions
Religion	Wheather using a VPN	Reports	Horizontal scrolls		Income	Charitable donation history
Politics	Proximity to other accounts (e.g., on	Pages	Use on mobile vs desktop		Age	Political donation history
Age	Authoritative source (e.g., email address	Groups	Reporting others		Abusivene ss	
Public Figure		Ads	Ad clicks / views / purchases		Spammin ess	
Occupation		Events	Stickers		Public figure	
Movies			Pokes		Occupatio n	
About me			Notifications clicks		Fake	

User Shared / Provided	Extracted	Created	Behaviors	Received from other users	Inferred	Purchased
Language			Provide lightweight negative feedback		Authoritative health source	
Music			Survey participation		Authoritative news source	
Payment information			Survey responses			
Birthday			Follows			
Privacy/public			Likes			
			Reactions			
			Hides			
			Event RSVPs			
			Friend requests sent			
			Friend requests sent but ignored			
			Friend requests received and			

User Shared / Provided	Extracted	Created	Behaviors	Received from other users	Inferred	Purchased
			Friend requests received and rejected			
			Friend requests received and ignored			
			Active last 7 days			
			Blocks			

For a deeper dive into this topic, this spreadsheet provides a comprehensive overview. It contains information about the types of data stored by the platforms. The various tabs provide examples of the types of data collected from: users, content, sessions, predictive models (machine learning, artificial intelligence, etc.), surveys, and networks.

Now that we've reviewed the types of data that VLOPSEs collect, we can explore how these data are used by the platforms.

3. What Do Platforms Generate From That Data?

This data can be used by the platforms for a variety of reasons. It can power machine learning algorithms in an advertising or content recommendation system. It can also power systems designed to keep the platform safe. Machine learning algorithms use the large amount of data platforms collect to predict:

- which advertisements users are most likely to click on,
- what content is most likely to be engaged with,
- whether the user is likely to be a bot or authentic, and
- whether a piece of content is likely to violate platform policies.

The data collected is a way for the platforms to fine tune their recommendation systems, ensure engaging content on every visit, limit or remove abusive users, and/or perform tests on users to discover new ways of improving the platform. The data is significantly valuable to platforms, serving as a key reason why companies invest so much into their data infrastructure.

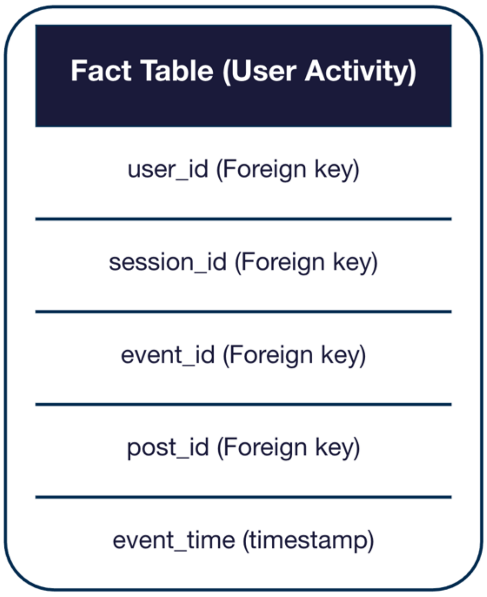
3.1. How are datasets stored and used?

Data storage and processing is expensive, especially at the scale that VLOPSEs operate. Given the massive volume of data collected by VLOPSEs, the companies operating these services need to split their data into more manageable chunks to reduce the computational and financial expense of accessing the data. This is achieved by reducing the size of the data in tables and reducing the amount of data that needs to be processed or scanned when queried by employees. There isn't a single uniform way that all companies organize their data warehouses. But a common data warehousing practice is splitting data into fact and dimension tables.

The following sections describe fact tables and dimension tables, the types of variables they will contain, and how these tables can connect with each other.

3.2. Fact tables

Fact tables describe discrete actions or events and contain id variables that can be used to connect the events to other information stored in dimension tables. The table below represents a simple fact table. Here, it contains the id number corresponding to each user, each session, the event that occurred, and the posts impacted.



Fact tables are often in "long" format, which means there are fewer columns of different variables but more rows containing data. Critically, "long" format is useful if individual events (e.g., logins, searches, purchases, clicks) could appear multiple times. This is particularly important when dealing with the types of data collected by VLOPSEs. For example, some users may login multiple times per day (thereby creating multiple sessions), while other users might only login once a month (thereby creating a single session per month). Long format tables permit different numbers of rows per user or per action.

Fact table dictionary example

Typically, there is a "dictionary" which defines what variables appear in a fact table, and includes some information about the features of those data. For the User Activity example fact table (above), the corresponding dictionary might look like the table below.

Variable	Description	Data Type	Retention	Restricted access?
user_id	Assigned numeric ID for each user	INTEGER	30	N
session_id	Assigned numeric ID for each user session	INTEGER	30	N
event_id	Label for each event type	STRING	30	N
post_id	Assigned numeric ID for each post	INTEGER	30	N
event_time	The time when event occurred	TIMESTAMP	30	N

The level of internal documentation at companies can vary considerably, but some companies have infrastructure that tries to fill out these dictionaries based on artificial intelligence, machine learning, and/or natural language processing. For example, table metadata will specify the data type of the variables. And if a variable in a particular table is joined from another table, then the documentation for the variable in the upstream table can be carried to the downstream table.

Tables have retention windows that may be determined by company policy, law, or by an ad hoc decision of the data scientist, engineer, or researcher who created the table. Certain variables may also be deemed sensitive, which may mean that anyone trying to access them needs additional permissions, which can depend on having a legitimate business use case that requires access. If someone tries to access data that are older than the retention window, or if they lack the approval to see the restricted variables, their search query will fail.

Fact table data example

The data stored in the table above corresponds to the example table below. Some features of the data in this example table are worth pointing out explicitly. Despite this table having five rows of data:

1. There are only two distinct users in this table (two different user_id values),
2. There are three different sessions (three different session_id values),
3. There are four different event types (view, like, reply, share),
4. The user in the first three rows had two sessions – one session where they liked one post, and one session where they viewed and liked another post), and
5. The user in the last row logged replied to and shared one post.

user_id	session_id	event_id	post_id	event_time
12	5021	LIKE	90	2025-01-01T00:10:10Z
12	5021	VIEW	93	2025-01-01T00:20:15Z
12	5022	LIKE	93	2025-01-01T00:20:20Z
53	5023	REPLY	143	2025-01-01T00:30:23Z
53	5023	SHARE	143	2025-01-01T00:30:24Z

Extracting metrics from fact tables

Fact tables like this are informative and can be used as key performance indicators for the company that owns these platforms. For example:

1. **Total daily platform users** = sum of distinct user_id values grouped by date

```
SELECT
  CAST(event_time AS DATE) AS date,
  COUNT (DISTINCT user_id) AS total_users
FROM fact_table_example
```

```
        GROUP BY date
```

This query would return output that looks like the following table.

date	total_users
2025-01-01	9000

2. **How many posts viewed per session** = Average count of distinct post_id

```
    values per session_id
        SELECT AVG(posts_per_session) as
    average_posts_views_per_session
        FROM (
            SELECT
                session_id,
                COUNT(DISTINCT post_id) as posts_per_session
            FROM fact_table_example
            GROUP_BY session_id )
```

This query would return output that looks like the following table.

average_post_views_per_session
14.129838

At the same time, though, these data are rather blunt and gloss over important details that provide more context. This is where dimension tables come into play.

3.3. Dimension tables

Dimension tables describe attributes of the data in the fact table. For example, the fact table example above only includes long strings of numbers that identify a user, session, event, and post. Dimension tables would provide additional information about those users, sessions, events, and posts. Attributes of those variables are theoretically infinite, so it is common for dimension tables to have dozens or hundreds of columns of variables (vs. the fact tables which tend to have a much smaller number of columns of variables).

To help illustrate this, we have generated an example dimension table dictionary and sample data below to show more information about users. This table might include attributes that are relatively constant across a given user, such as their name, email, date they created their account, birthday or age, country, age, gender, etc.

Dimension table data dictionary example

Variable	Description	Data Type	Retention	Restricted access?
user_id	Assigned numeric ID for each user	INTEGER	90	N
name	User's name	STRING	90	Y
email	User's email address	STRING	90	Y
date_joined	Date user created account	DATE	90	N
country	User's specified country	STRING	90	N
age	User's specified age	INTEGER	90	N

Dimension table data example

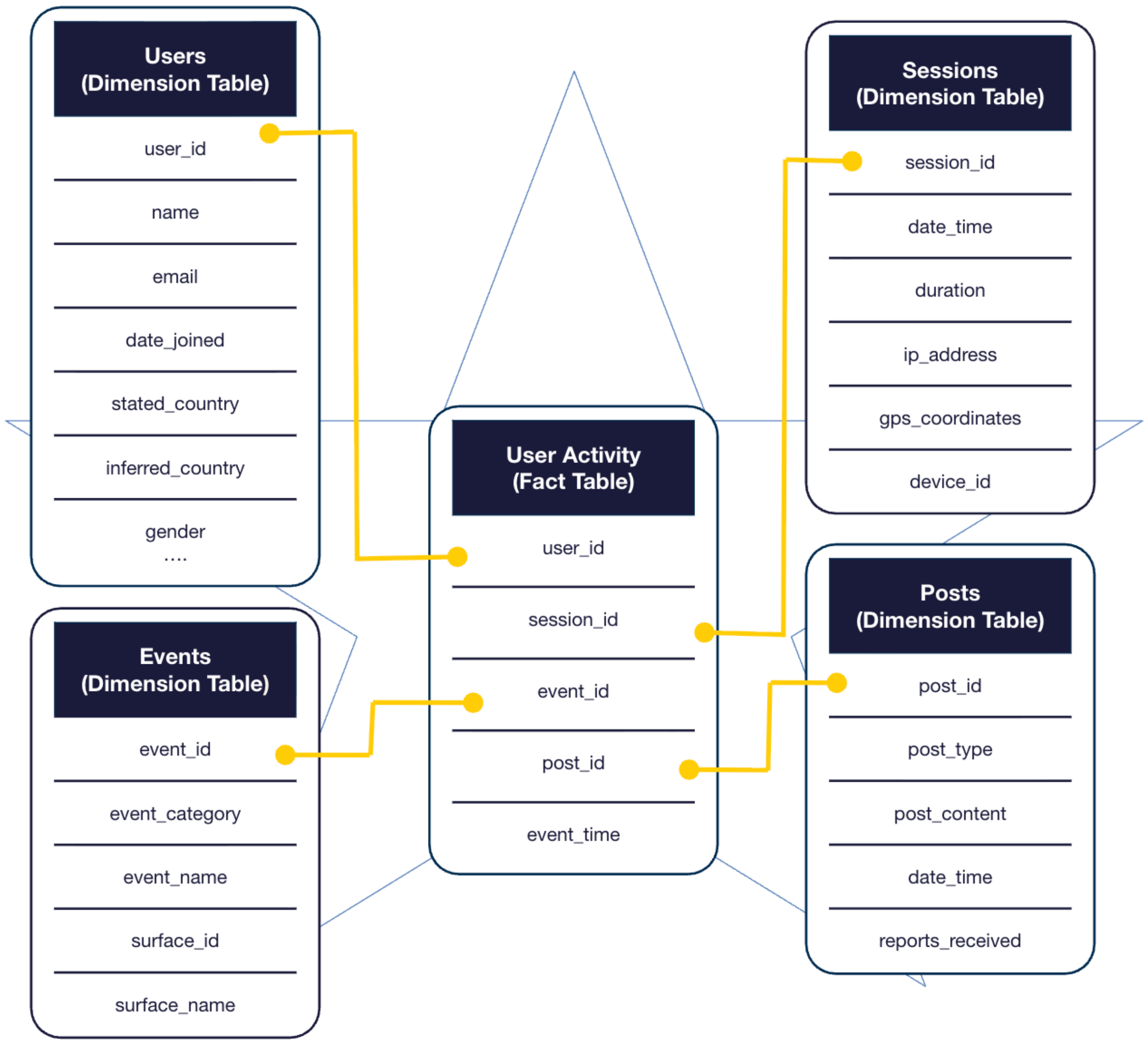
user_id	name	email	date_joined	country	age
12	John Doe	johndoe@example.com	2013-01-01	UK	45
53	Jane Doe	janedoe@example.com	2014-01-01	US	63
891	Fulano	fulano@example.com	2020-01-01	MX	32
156	Mengano	mengano@example.com	2010-01-01	CO	22
932	Zutano	zutano@example.com	2025-01-01	ES	15

Dimension tables tend to be in "wide" format where there are many columns of variables for each observation (e.g., users, posts). Unlike fact tables, dimension tables must have only one row of data per observation. Dimension tables also must contain a primary key that can be matched with a variable in the fact table.

3.4. The relationship between fact and dimension tables

Given the relative constancy of dimension table variables (people don't often change their name or email address), it would be highly inefficient to include this data for every row of the fact table, because the exact same information would be duplicated in a large number of rows. Separating fact variables from dimension variables allows companies to use fewer resources to access the data that they need (meaning, employees don't need to query giant tables that contain far more data than is necessary for the task at hand). For example, users' names aren't necessary for counting the distinct number of users, and including such data in the fact table would increase its size and the amount of memory required to access and filter the table down to the minimum number of variables needed to calculate the number of users (i.e., user_id). It can also help keep sensitive variables separated from data that needs routine access for business purposes.

Often, the relationship between fact and dimension tables are depicted as a star schema where the fact table is in the center and the dimension tables are the flares coming off of the star. The diagram below depicts the example fact table on user activity and a dimension table with additional information on each of the variables in the fact table. This data warehouse structure requires a common key variable between the fact table and the relevant dimension table. It is this id variable that permits accessing additional information about each variable in the fact table.



Joining data across multiple fact and dimension tables

To calculate the number of users of each surface by country, for example, we could write a query that joins in country and surface. Given that these variables exist in different tables, we must "join" those tables together. We've written one SQL query below that:

1. Selects the necessary columns from the 3 different tables (2 dimension tables and the 1 fact table),
2. Joins them together based on the relevant key variables (user_id for the country variable, and event_id for the surface_name),
3. Counts the number of unique user_ids grouped by each country and each surface.

Extracting and joining data across multiple tables

```
SELECT
    d1.country,
    d2.surface_name,
    COUNT(DISTINCT f.user_id) as user_count
FROM fact_table_example f
JOIN dim_user d1
    ON f.user_id = d1.user_id
JOIN dim_event d2
    ON f.event_id = d2.event_id
GROUP BY
    d1.country,DE
    d2.surface_name
```

This query should return results that look like the following table.

country	surface_name	user_count
IN	FOLLOWING_FEED	12000
IN	RECOMMENDED_FEED	10000
US	SHOPPING_FEED	9000
DE	RECOMMENDED_FEED	6000
US	PROFILE_PAGE	2000
MX	SEARCH_RESULTS_PAGE	1500

These are simple examples that most VLOPSEs almost certainly use, but they likely also perform many more complex computations that utilize additional discrete information the user provides when creating their profile. Incorporating more user information in the query allows platforms to predict the probability of a user clicking on an ad, buying a product, commenting on a post, or disseminating harmful content.

3.5. Probabilistic variables

Because of the sheer volume of content posted by millions or billions of users, it is impossible for platforms to manually review each post before distributing that post. Furthermore, many platforms and search engines "personalize" results for each user, meaning that VLOPSEs need to be able to predict what results are relevant to and desired by each user. For example, if someone living in London asks a search engine for "restaurants near me," modern search engines would likely return restaurants located in London and not restaurants in New York City. Similarly, an online short-form video platform will show videos related to each user's particular interests.

Advanced computational methods, like machine learning (e.g., gradient boosted decision trees, random forest models, support vector machines, neural networks;), topic modeling (e.g., latent dirichlet allocation, latent semantic indexing), and mixture models, are used to characterize entities (like users and posts) on VLOPSEs. The details of these specific models are beyond the scope of this overview, however, some fundamental information needed to understand how the core methods of these platforms function includes:

1. These models rely on using data to predict some specific outcome (e.g., the likelihood of a user to click on an ad).
2. The models are typically trained on existing data with known behavioral outcomes (e.g., predicting which users clicked on an ad).
3. The models then integrate many other variables (also called features or predictor variables) that can be combined to predict the likelihood of the outcome.
4. These models will generate a score for each element that they are trying to characterize (e.g., each ad).
5. The models are evaluated based on how accurately the score they generated predicts the behavioral outcome of interest (i.e., if one ad receives a score of 0.80 and another ad receives a score of 0.10, the ad receiving a score of 0.80 should generate the more clicks than the ad receiving a score of 0.10; otherwise, the model would be deemed poor because it doesn't improve the prediction of the behavior of interest).

These methods can be used to predict very concrete behaviors, like a click or posting a comment, but also more subjective characteristics of users or content, like the probability that a user is being deceptive about their location or that a piece of content may be harmful.

In the next sections, we walk through examples for how one might create a model to predict two important tendencies relevant to the Code of Practice on Disinformation and risks delineated in the Digital Services Act.

Predicting whether a user is misleading about their location

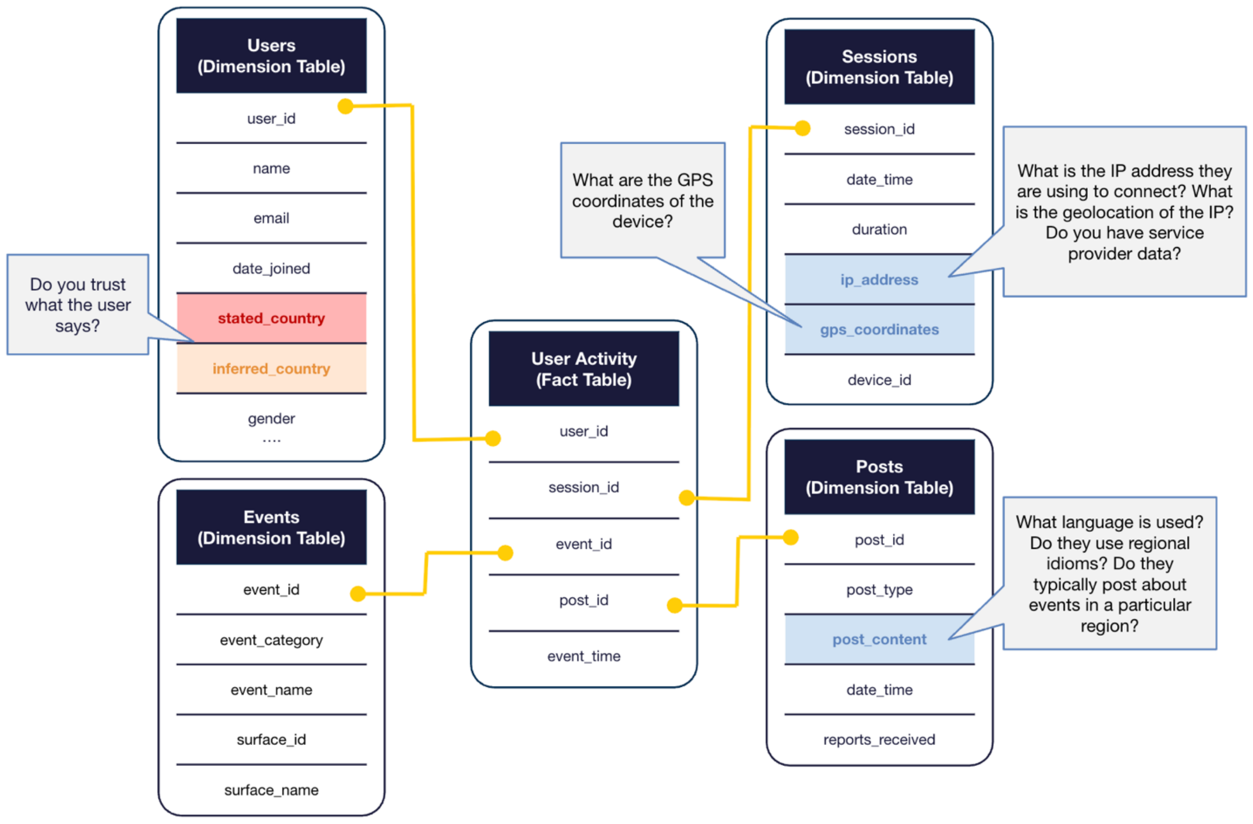
VLOPSEs generally have an incentive to use machine learning models to predict the location of users. This is very useful for ad targeting purposes. For example, a user might report living in one location, but spend a lot of time in another location. Advertisers might want to target that user based on where they spend the most time, and so platforms will generally keep track of both where users report to live and also create a predicted location of the user.

Sometimes, the signals received from users about their location might be contradictory. For example, the user may self-report living in one country, but only login to the service from IP addresses or provide GPS locations from another. Sometimes this is understandable, for example if the user lives under a government that doesn't respect human rights, they may need to mask their true location or use VPN services to hide their activity. But this can also be a signal that the user is trying to engage in deceptive activities. It can be a smart tactic for VLOPSEs to give extra scrutiny to accounts with contradictory location signals, particularly if the accounts are engaged in high risk activities like purchasing political advertisements or messaging minors.

One strategy used to interfere with a foreign country's elections involves an interested group creating fake accounts which appear to originate in the target country where the election is taking place. The foreign group will use these accounts to spread their propaganda, with the hopes of influencing or interfering with the election of the target country. Sometimes this is to advance a specific policy agenda, and sometimes this is to generally create chaos and confusion. Therefore, VLOPSEs can be skeptical of the self-reported location that is provided when an account is created.

Given the massive amount of data VLOPSEs collect, there are many variables that could be useful in trying to estimate where an account is actually based. In the diagram below, we highlight the probabilistic variable we want to infer – inferred_country – in yellow, and some variables that may be useful in helping us predict where an account is based in light blue.

How might a company infer a user's location?


Users (Dimension Table)
user_id
name
email
date_joined
stated_country
inferred_country
gender
....
Do you trust what the user says?
Events (Dimension Table)
event_id
event_category
event_name
surface_id
surface_name
User Activity (Fact Table)
user_id
session_id
event_id
post_id
event_time
What are the GPS coordinates of the device?
Sessions (Dimension Table)
session_id
date_time
duration
ip_address
gps_coordinates
device_id
What is the IP address they are using to connect? What is the geolocation of the IP? Do you have service provider data?
Posts (Dimension Table)
post_id
post_type
post_content
date_time
reports_received
What language is used? Do they use regional idioms? Do they typically post about events in a particular region?

For example, platforms typically extract massive amounts of data from the device or devices that an account uses to access the platform. All of these devices should have an IP address. IP addresses include a lot of explicit information regarding location (e.g., country, region, city, approximate latitude and longitude, and telephone area code), along with more implicit indicators that may also be related to location, such as:

1. **The internet service provider**–many providers are regional, so if someone claims to be from one region where that service provider is unavailable, that is suspicious;

2. **Network type**–whether the network is residential, commercial, governmental, mobile, etc. If a government network is being used for many accounts that are posting about an election in some faraway country without making their government affiliation clear, that is suspicious;

3. **Location masking**–whether the user is connected to a VPN or proxy, or are using a Tor browser. VPNs and proxies can be used to mask someone's location and/or to encrypt their behaviors so that some intermediate party cannot view their activity. Tor browsers use "onion routing," which means that they encrypt the user's behavior in multiple layers, and then route that data through nodes in many different places, and conceal IP addresses. There are many valid reasons to use VPNs and Tor. However, they are also useful for illegal and deceptive activities.

4. **Time zone**–if the IP-based time zone does not match the stated location time zone, or if the user's activity is at unusual times for the time zone.

5. **Language preferences**–if the typical language for an IP differs from the dominant language in a region.

Because IP addresses can also be spoofed using VPNs, Tor browsers, and other tools, they shouldn't be relied on as the sole input in predicting someone's location. Fortunately for these platforms, the devices that accounts use to access the platforms increasingly include precise GPS coordinates that can be accurate enough to identify the device's location to within a few tens of meters (based on satellites, cellular towers, or wifi signals).

VLOPSEs may also integrate other variables, like information about the types of content that a user posts. Some location-relevant information in posts are:

1. **Language**: Is the post written in the predominant language of a given region?
2. **Idioms**: Are the idioms used consistent with the regional dialect?
3. **Events**: Do they post about events that occur near their supposed location?

None of these signals can be used in isolation to make a definitive assessment of a location mismatch between stated and inferred country. A predictive model might look at many of these variables when examining accounts that had previously been confirmed to be inaccurately reporting their location, and then generate a score for each account based on these variables. It is important to combine many variables here (across many prediction models) because there are justifiable reasons why someone's device GPS may say that they are somewhere different from where they usually access the platform, like if they are on a vacation. However, the more of these indicators that are suspicious, the higher confidence the platform can have that the account is misleading about its location.

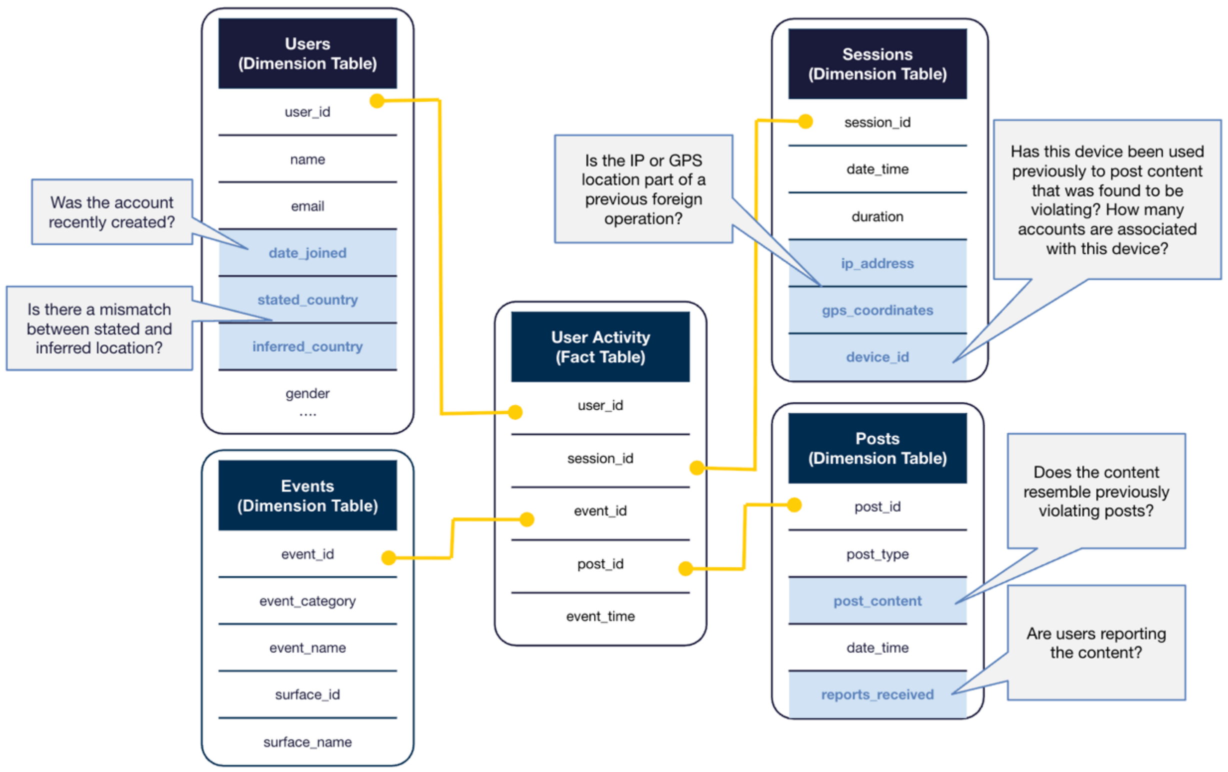
If this location prediction model is accurate, then accounts that score high may be subject to manual review by content moderators or receive rate limits applied. This can be justified because accounts that have a mismatch between stated and inferred location have significantly higher rates of posting policy violating content or engaging in policy violating behaviors. Location verification is certainly relevant to whether an account is authentic, trustworthy, and likely to contribute positive (or negative) value to a platform–but it is typically not against a platform's terms of service to not verify, and a location mismatch normally won't warrant deleting or suspending the account on its own.

Predicting whether a piece of content is policy violating

Platforms develop many predictive models to identify the likelihood that a piece of content is policy violating and could cause harm. The specific features used to predict harms differ. For example, a model predicting the likelihood of a post being spam likely will include features such as the rate of posting of the account (e.g., do they post/comment too frequently?) and the diversity of the contents of the post (e.g., are they copying and pasting the same post/comment in lots of places?). On the other hand, a model predicting the likelihood of a post containing incitement to violence would likely include information about the language that the post is in, whether that post contains specific hateful phrases or words, and whether the author of the post has posted policy violating content in the past. It is also possible to generate an overall likelihood of a post violating any policy (e.g., spam, hate speech, disinformation, child sexual abuse material).

In the diagram below, we have highlighted some variables across different dimension tables that could be useful in predicting harmfulness in light blue.

How to infer the likelihood of a post violating policies?


Users (Dimension Table)
user_id
name
email
date_joined
stated_country
inferred_country
gender
....
Was the account recently created?
Is there a mismatch between stated and inferred location?
Events (Dimension Table)
event_id
event_category
event_name
surface_id
surface_name
User Activity (Fact Table)
user_id
session_id
event_id
post_id
event_time
Is the IP or GPS location part of a previous foreign operation?
Sessions (Dimension Table)
session_id
date_time
duration
ip_address
gps_coordinates
device_id
Has this device been used previously to post content that was found to be violating? How many accounts are associated with this device?
Posts (Dimension Table)
post_id
post_type
post_content
date_time
reports_received
Does the content resemble previously violating posts?
Are users reporting the content?

As described in the previous section, if an account's inferred location is *substantially different* from its self-reported location, that would likely increase the risk that that account may be more likely to create harm. It is critical to highlight the qualifier *substantially different,* because it's common for people to access platforms on their devices as they move about their community, and accessing a platform from 5 kilometers away from one's home is normal. If someone is always accessing the platform from 10,000 kilometers away from their home, then that is suspicious, and is an indicator of elevated riskiness of that user.

Additionally, some other variables that might be useful to include in training this predictive model could be:

1. **How recently the account was created**: Some people create new or "burner" accounts so that they can engage in behavior that they wouldn't want associated with their primary account. Therefore, low account age is a factor that increases the risk of an account being violative.

2. **Have other violating accounts logged in using the same device**: If an account accesses the platform on a device that is associated with other violating accounts, or with an older account that was terminated due to violations, then that new account on a known device is also likely to be riskier, because it could represent a user, which previously violated policies, that is making a new account.

3. **User reports**: If a post or an account is being reported by other users for violating the terms of service or for engaging in violative behavior (e.g., posting harmful misinformation, trying to incite violence, harassing protected groups), then that increases the likelihood that that account is engaging in violative behavior.

Therefore, features such as these may be entered into a predictive model to try to predict the likelihood of a post violating policies. Again, a predictive model like this would be trained on previously collected data where the outcome is clear (i.e., a post was evaluated and found to be violating or not). The model would then calculate a score based on some combination of these features and then try to predict whether a post was deemed violating vs. not violating. If the model seems to differentiate between violating and non-violating content well, it would then be put into operation and run on new, incoming content where it is not known whether the content is violating.

3.6. Other data reduction considerations

In addition to using fact and dimension tables, VLOPSEs further reduce the expense and time required to process the data by partitioning or splitting the data into tables by:

1. **Platform**: If a company owns multiple platforms, they may have separate tables for each platform.

2. **Surface**: Most platforms have different ways for users to interact with the platform, and these different ways are often referred to as "surfaces." Examples of surfaces could include, but are not limited to, a feed limited to accounts the user follows, a feed for recommended content for the user, an account profile page, a shopping section, a messaging section.
3. **Date**: In addition to the retention periods, VLOPSEs may also partition large tables by day. In other words, if a table has a 90 day retention period and is partitioned

by day, then the cost of querying data from one day in the table is 1.1% (1 / 90) of what it would be if the table weren't partitioned by day.

4. **Sensitivity**: More sensitive data may be stored in separate tables where access is limited to people with the appropriate approval.

While external researchers accessing platform data likely won't need to do much data reduction with the data they obtain, they will need to be mindful of how the data are structured and partitioned to be able to query the data. If a query requests data from a table outside of the partition window, the query may fail or provide an inaccurate result. Once a researcher knows that a table is partitioned on a variable, then it is straightforward to add a line to the query that filters the data based on the partition. For example, if a table is partitioned by date, the researcher could add a filter such as "WHERE date = '2024-12-12'".

Now that we have reviewed the types of data platforms collect, and how they process, use, and store those data, we are ready to learn more about how API and internal data access across VLOPSEs can empower research.

4. Data Comparisons: API vs. Internal Data

When talking with external researchers across the EDMO hubs, we found several common themes in their experiences. Some of these themes are important to address, but are beyond the scope of this guide. Nonetheless, we include all common themes below, and then will go into greater detail about the in-scope themes.

1. **Researchers are struggling to gain approval to use the data access APIs.** For example, while nearly every researcher we spoke to had previously had access to recently deprecated tools from VLOPSEs, almost none had access to the more recent replacements. Many researchers report waiting months without hearing back from the VLOPSEs regarding their application to gain access to the API. Even after waiting for months, some researchers only received access after explicitly reminding the VLOPSEs that they would be subject to fines under the Digital Services Act if they were not granted access. Other researchers simply had not received access. This is important to flag, and there are efforts to track the response of VLOPSEs, but it is beyond the scope of this report.

2. Some researchers who were granted access received it for **parameters that did not match their original request**. For example, one researcher proposed a 6 month project, but was only granted access for 3 months of data. Other researchers did not receive approval until after a significant global event, such as an election, that they proposed to study using the platform data had occurred. This is important to flag, because timely access to the data sets is key for monitoring risks around critical societal events like elections, but it is beyond the scope of this report.

3. Some of the APIs had **very limited quotas** so that a researcher might be limited to returning 500 or 1000 observations of data per API call, and are limited to a small number of API calls per day. When studying relatively rare events (e.g., with prevalence rates < 2%), it is probable that API calls might not return any relevant content, which makes it difficult to study the prevalence of harms. This is important to flag, because even though harmful or violating incidents might have low prevalence on VLOPSEs, because VLOPSEs have billions of users globally and tens of hundreds of millions of users in the EU, the total number of occurrences of harms can be significant and have significant impact on people and societies, but this problem is beyond the scope of this report.

4. **The numbers in the data do not reflect the numbers observed on the same piece of content on the platform.** For example, a post might have 500 engagements in the data download, but only 10 engagements when looking at that same post live on the platform. It is possible that there may be some logging delay between when the behavior happens on the platform and when it is logged into the data table connected to the API, but we cannot say that this explains differences between the data downloaded and the data observed on the platform. This is important to flag, because researchers obviously need accurate and trustworthy data to properly study online risks, but this is beyond the scope of this report.

5. **Using the APIs is rather technical, and requires knowledge that many researchers do not have** (e.g., coding knowledge in Python). This is important to

flag, because the data and tools provided by the VLOPSEs need to be usable by the target audience, but it is beyond the scope of this report.

6. **Many variables that researchers want are not available in the API.** We go into this in more detail.

Generally, VLOPSEs only include simple data like post/reply text or videos, engagement with that content (views, reactions/favs/likes, shares, replies), the date the content was created, and some basic information about the entity creating that content (number of followers, their free-text biography from their profile, whether the account is verified). While these data are useful, they are lacking. If a researcher wants to verify the accuracy of a VLOPSE's transparency report, they would not be able to do so because the VLOPSEs do not provide access to the variables related to the systemic risks that researchers are supposed to be able to study.

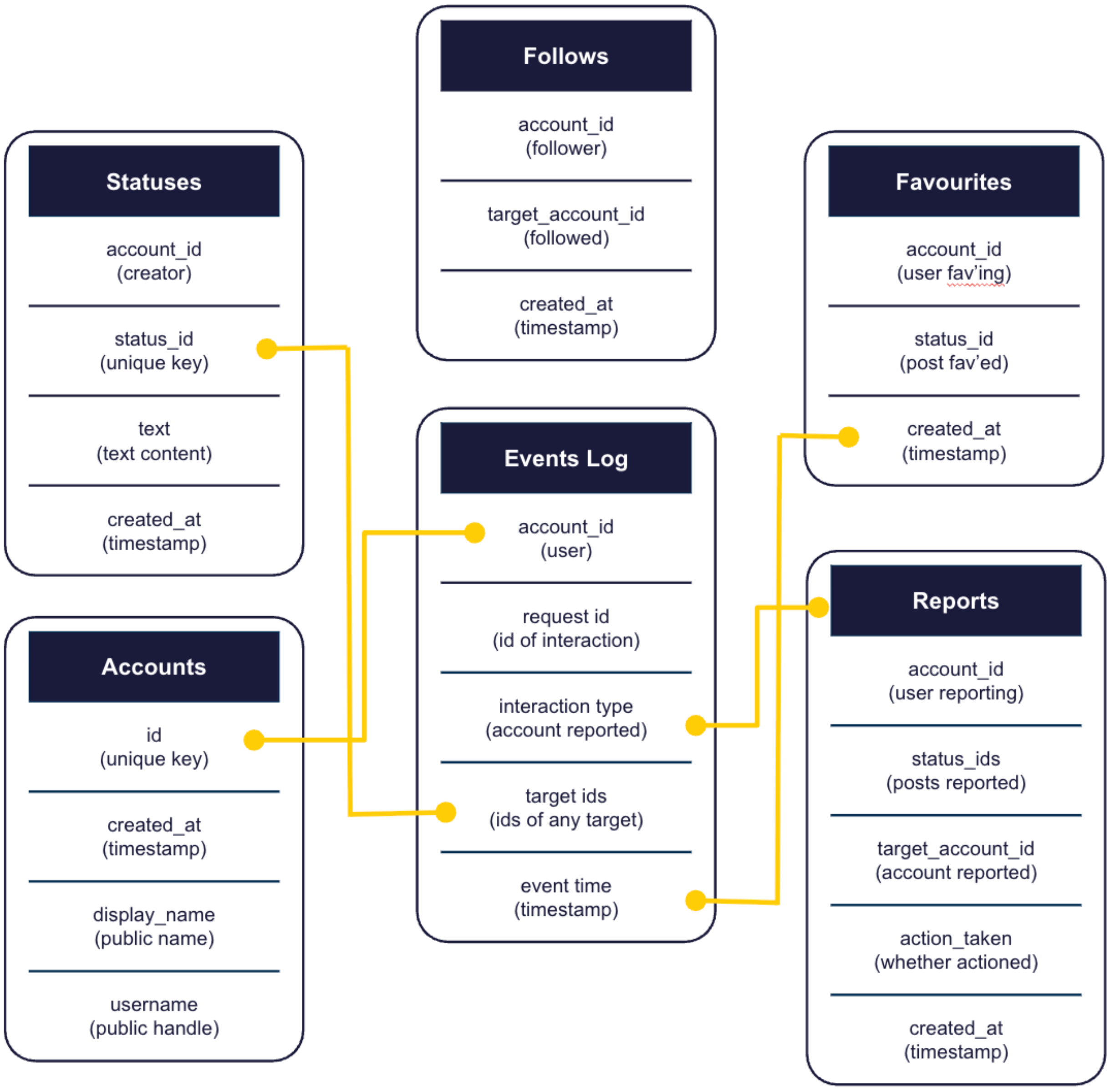
There is also considerable variability in the data available via the APIs. To illustrate this point, we've generated three examples and created tables comparing variables available at three different VLOPSEs (Facebook, X, TikTok) and one non-VLOPSE that is open source and thus we can evaluate the data generated internally (Mastodon). We created separate tables for two different types of risk and filled in the cells with variables available via the API for each platform that may be useful in evaluating whether the content may contain each specific harm.

4.1. Mastodon as a evidence of internal data

As a way of confirming the data we understand to exist within platforms–and comparing to the APIs that VLOPSEs have made available–we can use Mastodon, an open source platform which operates at a much smaller scale than the VLOPSEs. If a variable or dataset is available in the internal data tables generated by Mastodon, then that data very likely exists internally at VLOPSEs or could be easily generated at VLOPSEs.

Mastodon has over 75 data tables. These tables represent the "production" database, meaning they are the live tables that are used to display data to users and are modified as users interact with a Mastodon instance. They primarily act as dimension tables, because the observations/rows in these tables represent individual entities rather than a list of events. In addition to these production tables, Mastodon also has a core logging functionality which can record events and can capture user interactions with Mastodon. The core logging of Mastodon acts as a fact table in the overall data schema.

Key tables within Mastodon, for our purposes, include "accounts", "statuses", "status_stats" (which aggregates engagement statistics for statuses), "favourites", "follows", "reports", and the logging (a fact table for all events as users interact with the platform). The following diagram illustrates the "star schema" nature for Mastodon data, highlights some key variables in the tables, and shows some example connections between the logging events and the production dimension tables. More details can be found in the source code of Mastodon.



Mastodon contains enough data and logging for us to study, fairly comprehensively, how risks might manifest on the platform. But, as a general philosophy, Mastodon doesn't track or keep data unnecessarily. To start, Mastodon does not have a robust definition of a "view", because Mastodon does not track what content comes into view on a user's device or how long content stays in view. For Mastodon, the best analog to a view is if the content was served to the user's device. VLOPSEs will typically have logging systems in place to track and store how long content is in the view of users' devices.

Additionally, Mastodon does not, by default, store the event log. It is possible to store it for any reasonable retention length, and so in our analysis of Mastodon, we will assume that the event log is stored for some length of time, for example 90 days. Mastodon does not have any content classifier systems built in. This means there won't be a score that predicts the likelihood of a piece of content violating any Mastodon content policies, which VLOPSEs typically have internally. Finally, Mastodon does not collect personal data about users, such as location, age, and gender. Mastodon overall collects significantly less data than VLOPSEs, so a good rule of thumb is: if Mastodon has a table that can answer your research questions, then you can be highly confident that any VLOPSE will be storing that same data.

4.2. Example 1: research by harm type

In this example, we are assuming that a researcher may have a question related to two harm areas, both related to the Code of Practice on Disinformation: disinformation and foreign information manipulation and influence (FIMI). Disinformation acts as an analogy for roughly any harm related to a content policy violation of the platform, such as hate speech or harassment. FIMI acts as an analogy for any harm related to a violation of a platform's behavioral policies, such as spam or inauthentic behavior.

For both of these harm types, we collect the relevant variables made available through the research APIs for Facebook, X, and TikTok, while for Mastodon we collect all the relevant variables available in the data tables and event logging. Researchers would likely need to collect information around the text and media of content on the platform, engagement of content on the platform, classifier scores related to the harms, user reports of content, data related to any platform action, and user controls. All these variables would be necessary to, for example, collect a random sample of content to study the prevalence of the harm, collect the top content on the platform to study prevalence of the harm among the most widely distributed content and most significant spreaders, evaluate the accuracy of any platform moderation systems, and assess any possible biases in the platform policies and moderation operation.

As demonstrated in the table, platform APIs only offer a fraction of relevant data compared to what is available internally at the companies. Platform APIs likely don't offer enough data to enable researchers to meaningfully answer research questions related to systemic risks.

	Facebook	X	TikTok	Mastodon
Disinformation				
Text/Media	post_media_text	text	video_description hashtag_names	statuses media_attachments
Engagement	post_reactions post_comments post_shares	organic_metrics nonpublic_metrics promoted_metrics	like_count comment_count share_count view_count favourites_count	favourites_count replies_count reblogs_count views (from event log)
Reports				account_warnings reports report_notes

Classifier		possibly_sensitive	video_label is_stem_verified	follow_recommendation_suppressions global_follow_recommendations
Manual review				account_moderation_notes account_notes admin_action_logs appeals
User Controls				blocks conversation_mutes custom_filters follow_recommendation_mutes
Audience Information	language_scope geographic_scope	Place: country Place: geo		
FIMI				
Text	post_media_text	text	video_description hashtag_names region_code	statuses media_attachments
Engagement	post_reactions post_comments post_shares	organic_metrics nonpublic_metrics promoted_metrics	like_count comment_count share_count view_count favourites_count	favourites_count replies_count reblogs_count views (from event log)
Reports				account_warnings reports report_notes
Classifier		possibly_sensitive	video_label	follow_recommendation_suppressions global_follow_recommendations
Survey Data				

Manual review				account_moderation_notes account_notes admin_action_logs appeals
User Controls				blocks conversation_mutes custom_filters follow_recommendation_mutes
Creator Information			region_code	Location from IP of creator

4.3. Example 2: creating core datasets for evaluating structural indicators with mastodon

There are two key datasets which would greatly empower researchers to evaluate the structural indicators of the Code of Practice on Disinformation for a given platform: a large random sample of public content weighted by views, and a large list of the most viewed public content. These datasets can be created fairly easily using Mastodon data.

To generate a random sample of content weighted by views, we will have to use the events log to create a new table. This could be done using any number of logging tools, and using this we could collect relevant “Status Load” events, which would enable us to create a table–here named “views”–which would have columns: created_at, account_id, status_id. Each observation in “views” would represent a single instance of a user being served a status.

To generate a list of randomly selected content, weighted by views, we simply need to randomly sample the views table and join the key “statuses” and “status_stats” tables with contain all the information about the sampled statuses.

```
SELECT
  status.*,
  stats.*
FROM (
 SELECT status_id
 FROM views v
 WHERE
  created_at BETWEEN <START DATE> AND <END DATE>
 ORDER BY
  random()
 LIMIT <NUMBER OF SAMPLES> ) r
LEFT JOIN
 status_stats stats
ON
```

```
  stats.id=r.status_id
LEFT JOIN
    statuses status
ON
  status.id=r.status_id
```

This query will return any desired number of sampled content between a specified date range, as well as all information related to the status, including the content, account information, and engagement statistics. This data set could be reduced to include only the essential variables necessary for researchers to evaluate if the content is disinformation. If the admins of the Mastodon instance were already labeling content as disinformation themselves, then any labels the admins applied could also be joined in from the “reports” table.

Using the dataset of randomly selected content above, researchers could evaluate the content and estimate the prevalence of disinformation. The prevalence, combined with a count of the total number of views, could then be used to estimate the total number of exposures as well as the total reach of disinformation on the platform.
To generate the most viewed content on the platform is also trivial using the Mastodon data. Starting again from the “views” table:

```
SELECT
    status.*,
    stats.*,
    v.view_count
FROM (
  SELECT status_id, COUNT(1) AS view_count
  FROM views v
  WHERE
    created_at BETWEEN <START DATE> AND <END DATE>
  GROUP BY status_id
  ORDER BY view_count DESC
  LIMIT <NUMBER OF SAMPLES> ) r
LEFT JOIN
  status_stats stats
ON
  stats.id=r.status_id
LEFT JOIN
    statuses status
ON
  status.id=r.status_id
```

This query will return most viewed content on the Mastodon instance, limited to whatever size is necessary. Using a dataset like this, researchers could again evaluate the content to see what is disinformation, study if there are accounts that are systematically sharing disinformation with broad reach, and then estimate the structural indicators related to the sources of disinformation.

As Mastodon shows, the data that the platforms have internally could easily be used to create data sets that would greatly empower researchers to better understand the risks and harms that can occur on platforms. The main difficulties in generating these datasets from Mastodon come mainly from the fact that Mastodon limits the amount of data it collects from users, which is not a practice that VLOPSEs typically follow.

4.4. Example 3: ability to evaluate structural indicators using api data vs. Internal data

In this example we assume that a researcher is trying to estimate the structural indicators of the Code of Practice on Disinformation. We determine whether the data available either from the TikTok researcher API or the internal Mastodon data is enough to answer questions around the indicator.

Before walking through the table, it is important to note that because *neither VLOPSEs nor Mastodon* include any variable that indicates whether they flagged a post as disinformation, the determination of what content is disinformation must be made by the **researcher**. This is **not** a table of platforms classifying disinformation, it is a table determining whether accessible data enables researchers to do research into disinformation themselves.

Platforms do have content and behavioral policies related to disinformation though. Platforms could provide metrics and data around content and accounts taken down for violating misinformation related policies or inauthentic behavior policies. Those violating pieces of content and accounts could be represented in the API which would aid in the evaluation of the structural indicators. One potential issue with this approach, though, is that when content is removed from the platform, companies tend to have all data removed. If platforms do remove the data upon removing this content (or making it nonpublic), then platforms will need to leave behind markers in the data to make note of the removed content.

To match the comprehensive coverage of the structural indicators that Mastodon provides, the platforms will have to release datasets, much like the ones in example 2, that actually empower researchers to comprehensively understand the information environments on the platform.

PLATFORM	TIKTOK API	MASTODON
SI-1: Prevalence of Disinformation		
Reach of disinformation content (unique views per member state)	FALSE	TRUE
Engagement with disinformation content (comments, shares, reactions)	TRUE	TRUE
Depth of engagement (time spent on content, watch-through rate for videos)	FALSE	FALSE

Platform recommendation source (breakdown of how users saw the content: recommendations, subscriptions, reshares)	FALSE	FALSE
Modalities of content (audio, video, text, third-party links)	TRUE	TRUE
Top disinformation samples (500 pieces per monitoring period)	FALSE	TRUE
Enforcement metrics:		
Action taken (removal, labeling, demotion)	FALSE	TRUE
Type of violation (platform policy violated)	FALSE	TRUE
Reach and engagement before and after enforcement	FALSE	TRUE
Time of publication vs. time of platform action	FALSE	TRUE
Total content subjected to enforcement	FALSE	TRUE
SI-2: Sources of Disinformation		
Characteristics of disinformation sources:		
Number of disinformation pieces shared by accounts	FALSE	TRUE
Reach, exposure, and engagement of accounts	FALSE	TRUE
Network size (followers, connections)	FALSE	TRUE
Frequency of publication	FALSE	TRUE

Geolocation	TRUE	TRUE
Account history (age, previous violations)	FALSE	TRUE
Superspreaders:		
Reach, exposure, and engagement of superspreaders	FALSE	TRUE
Network size	FALSE	TRUE
Frequency of publication	FALSE	TRUE
Superspreader distribution (percentage of exposures by top X% of spreaders)	FALSE	TRUE
Relationships and network links between superspreaders	FALSE	TRUE
Information influence operations:		
Number of operations or networks detected	FALSE	FALSE
Number of accounts involved in each operation	FALSE	FALSE
Actions taken against subjects	FALSE	FALSE
Violations and enforcement metrics (reach, engagement before and after)	FALSE	FALSE
SI-3: Audience of Disinformation		
Aggregated characteristics of disinformation audience:		
Socio-demographic and psychographic profiles	FALSE	FALSE

Geolocation	TRUE	TRUE
Platform usage history (time on platform, frequency of exposure)	FALSE	TRUE
Network size (friends, followers)	FALSE	TRUE
Algorithmic recommendation data (whether users followed the source or not)	FALSE	TRUE
Probability of inauthentic behavior (bots, fake accounts)	FALSE	FALSE
Audience engagement metrics (X exposures or X engagements with disinformation)	FALSE	TRUE
Contextual audience characteristics for popular news brands	FALSE	FALSE
Complementary survey data (self-reported capability to recognize disinformation, social media habits)	FALSE	FALSE

Based on the table, we see that Mastodon enables researchers to evaluate many of the structural indicators, as opposed to the TikTok researcher API and many other VLOPSE APIs. The key gaps in the APIs of many VLOPSEs, which need to be covered, include a means of generating a random sample of content weighted by views, a means of viewing the most viewed and engaged with content on the platform, lack of information about content recommendation information, and exclusion of moderation data. While Mastodon data does have some gaps of its own, those gaps are primarily due to Mastodon not collecting the necessary data to answer them.

5. Conclusion

In summary, this report provides an overview of:

1. What data VLOPSEs collect, use, and store;
2. How those data are structured;
3. How to query such data;
4. What specific variables are theoretically available via VLOPSE APIs; and
5. How those variables may map to delegated harms.

A crucial takeaway from this report is that VLOPSEs collect a large amount of data that is quite complex to process and use. The data that VLOPSEs make available via existing APIs indicates a step in the right direction, but ultimately, the available data falls short of providing meaningful transparency into platform behavior. This is because VLOPSEs omit many of the critical variables and means of querying the data that they use to identify harms on their own platform from their transparency reports. Therefore, even if researchers are able to obtain approval to use these APIs, they cannot determine whether a given VLOPSE's transparency report is even accurate, much less replicable.

To achieve meaningful transparency into the methods VLOPSEs employ to identify content harm areas on their platforms, these companies need to provide the variables used to construct their transparency report, which likely includes the predictive variables used to classify content. At present, there are no APIs that include these critical predictive variables. Until this level of transparency and knowledge-sharing is achieved, researchers will be forced to do the content labeling work, rather than utilizing the extensive readily-available methodology employed by platforms themselves.

In the absence of these essential data points and data sets, the gap between what is reported and what is truly happening on platforms will continue to widen, limiting the potential for external oversight and accountability. To foster public-private trust and support more effective research, VLOPSEs need to prioritize greater openness in their data-sharing practices. Only then will researchers be able to meaningfully assess the risks on platforms and contribute to a more informed dialogue about content moderation practices.

EDMO

www.edmo.eu

The European Digital Media Observatory has received funding from the European Union under contract number LC-01935415