To appear in the Proceedings of the 31st USENIX Security Symposium (USENIX Security'22). August 2022. Boston, MA.

Measurement by Proxy: On the Accuracy of Online Marketplace Measurements

Alejandro Cuevas[*1], Fieke Miedema[*2], Kyle Soska[3,4], Nicolas Christin[1,4], and Rolf van Wegberg[2]

[1]*Carnegie Mellon University*
[2]*Delft University of Technology*
[3]*University of Illinois Urbana Champaign*
[4]*Hikari Labs, Inc.*

Abstract

A number of recent studies have investigated online anonymous ("dark web") marketplaces. Almost all leverage a "measurement-by-proxy" design, in which researchers scrape market public pages, and take buyer reviews as a proxy for actual transactions, to gain insights into market size and revenue. Yet, we do not know if and how this method biases results.

We build a framework to reason about marketplace measurement accuracy, and use it to contrast estimates projected from scrapes of *Hansa Market* with data from a back-end database seized by the police. We further investigate, by simulation, the impact of scraping frequency, consistency and rate-limits. We find that, even with a decent scraping regimen, one might miss approximately 46% of objects – with scraped listings differing significantly from not-scraped listings on price, views and product categories. This bias also impacts revenue calculations. We find Hansa's total market revenue to be US $50M, which projections based on our scrapes underestimate by a factor of four. Simulations further show that studies based on one or two scrapes are likely to suffer from a very poor coverage (on average, 14% to 30%, respectively).

A high scraping frequency is crucial to achieve reliable coverage, even without a consistent scraping routine. When high-frequency scraping is difficult, e.g., due to deployed anti-scraping countermeasures, innovative scraper design, such as scraping most popular listings first, helps improve coverage. Finally, abundance estimators can provide insights on population coverage when population sizes are unknown.

1 Introduction

Paradoxically, the emergence of online crime has made measuring criminal activities easier. In particular, the digitization of crime and the adoption of anonymization technologies has sparked the existence of anonymous marketplaces to support the underground economy. Although anonymous, most of these markets are publicly accessible, and, as a result, data are far easier to collect than for their street crime counterparts.

Online anonymous marketplaces have been the focal point of numerous measurement efforts of the underground economy [6, 16, 20, 41, 47, 50]. To gain insight into the size and scope of illegal activities on these markets, and how these evolve over time, most of the earlier work captured the markets' nature and their size – investigating the types of illicit products traded, and deriving the amount of listings, vendors and estimating its revenue.

Although these established insights help us understand trends in volume and types of crimes facilitated by online anonymous markets, the vast majority of earlier work is limited by their common measurement approach. All perform their analysis based on data collected through web scraping – i.e., collecting the content of public web pages displayed by the markets. This scraping is done in a measurement environment that is both inherently challenging (markets often run on low-availability servers [41] with high latencies due to the use of Tor or i2p hidden services), and even adversarial due to the market operators' extensive use of rate-limiting mechanisms such as CAPTCHAs, or their attempts to detect and ban automated activity [45]. As a result, researchers have to take missing and incomplete data for granted.

Furthermore, because they generally do not have access to the markets internal databases, researchers must use certain proxies – e.g., reviews instead of documented transactions, or listing counts – when performing analyses of, for instance, economic volumes. This "measurement-by-proxy" results in additional errors, whose size and influence on the results of the analysis are unknown. Because most of the approximations are due to missing, rather than incorrect, data, we know that many online anonymous market measurements can provide reliable lower bounds on economic activity. But by how much are they underestimating actual activity?

The potential for measurement errors does not only influence scientific research. If the confiscation of illegal assets by law enforcement is based on projected revenue calculated based on only data measured by proxy, the seized amount

[*]Both authors contributed equally.

will often be lower than the actual turnover of the seller. In short, estimating the size of measurement error on these marketplaces, as well as what influences these errors, is not only important to validate the outcomes of previous work, but, more importantly, understanding the origins of these errors should also help shape best practices for measurements of these marketplaces moving forward.

We make the following contributions:

- We provide the first overview of measurement methodologies used in online anonymous market research and show that very few papers explain their scraping and pre-processing routines.
- We build a framework to reason about online anonymous marketplace data collection and projections. Specifically, we mathematically define a model to express possible sources of inaccuracies in online anonymous market measurements.
- Using back-end data from a seized market, we empirically measure coverage statistics and find that scraped listings differ significantly from not-scraped listings on features such as price, product category and visibility.
- We validate revenue calculation approaches and show that taking reviews as a proxy for transactions can lead to underestimating the total market revenue by a factor of four.
- Through simulations seeded by actual market data, we estimate the coverage impact of various scraping methodologies, rate limits, and the precision of abundance estimation techniques.

The rest of this paper is structured as follows. Section 2 synthesizes related work on measuring online anonymous markets. Section 3 describes our experimental methods. Section 4 introduces a mathematical model to reason about online marketplaces. Section 5 explains our data sources, including our simulated marketplace. Section 6 presents our real-world comparison between scrapes and back-end data, to empirically uncover coverage and scraping bias. Section 7 discusses price estimation methods. Section 8 describes our experiments and results with simulated marketplaces. Section 9 describes our ethical considerations, and discusses limitations and public policy take-aways. Finally, Section 10 concludes.

2 Measuring Marketplaces

An extensive body of work studying online anonymous markets has provided us with substantial insights into market economics. Since 2013, over 60 papers covering a broad range of disciplines have used data from online anonymous marketplaces or their dedicated forums. From analyzing the first modern[1] online anonymous market, Silk Road, in 2013, to evaluating a whole ecosystem of competing markets, measuring marketplaces has evolved from studying a single market to analyzing market economics [16, 17, 41, 47], security practices [28], buyer and vendor behavior [44, 46, 51] and the impact of police interventions [22, 48]. Research using scraped data of online anonymous marketplaces has also shed light on the relationship between online and offline drug trade [20].

We first summarize how, at a high level, researchers can in turn exploit publicly available market data to produce measurements in Section 2.1. Then, in Section 2.2, we survey the literature for measurement methods used in previous research.

2.1 Background

Anonymous online marketplaces are similar to regular online markets on the clear web such as eBay, Alibaba, or Amazon Marketplace: they serve as a platform for vendors to post listings about products or services for buyers to purchase.

Figure 1 shows the Alphabay marketplace as an example, whose main features carry over to most online anonymous markets.[2] The landing page users initially reach (Fig. 1(a)) often features a menu with product categories, a search bar and an overview of popular listings on the market, as well as listing counts by category.

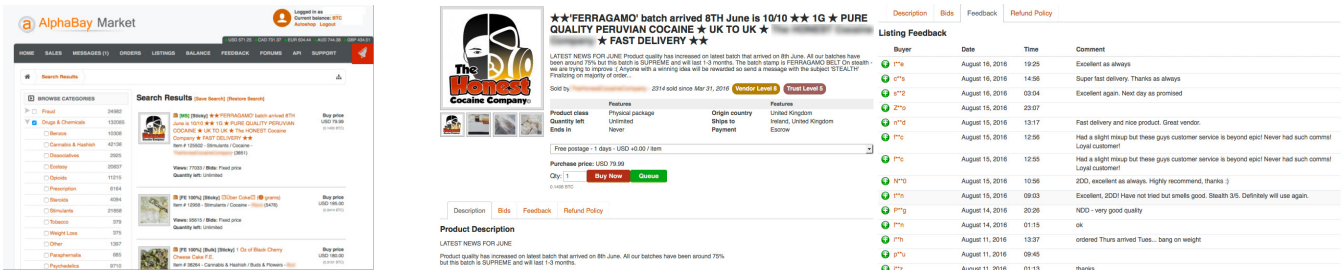
A more precise estimate of revenue can *a priori* be obtained by looking at each listing in more details. Specifically, Fig. 1(b) represents a typical listing page: title, description, geographical origin, vendor information, price, and, in some cases, total number of sales (2,314 here). To get a more precise picture of revenue over time, one may need to look at the feedback received by the vendor about the relevant item (Fig. 1(c)): the review timestamps can provide an approximate idea of the purchase dates. However, if buyers are not required to leave feedback for every single purchase, using reviews as a proxy for transactions will result in under-counts.

2.2 Literature survey

We next present an overview of measurement methodologies used in online anonymous market research. As we are interested in methodologies for scraping and pre-processing, which proxies and heuristics were used, and if/how external validation was done, we focus this overview on papers that performed this complete process – from data collection to analysis – and that investigated one or more complete markets. We thus exclude papers that use existing data, papers that fo-

[1] That is, the first online market to rely on a combination of network anonymization (Tor) and distributed cryptocurrency (Bitcoin). Other "proto-markets," such as The Farmer's Market, existed prior to Silk Road, but did not rely on the combination of cryptocurrencies with anonymizing technology, and were arguably far less influential.

[2] Alphabay was reportedly one of the largest online anonymous markets ever. It was seized in July of 2017, in a one-two punch that involved Hansa.



(a) Landing page (b) Item page (c) Review page

Figure 1: Marketplace example. Revenue can be estimated for each item, based on data present on the item and reviews page.

cus on just one product category or country, as well as papers based on other methods, e.g., user surveys or interviews.

Scraping methods. The first step in acquiring and analyzing dark net market data is to *scrape* the relevant markets; that is, to capture copies of the web pages describing item listings, vendors, and feedback so that they can be subsequently used for further processing and analysis. Relatively few authors [15, 16, 41] provide extensive details on their scraping methods: number of scrapes, frequency, crawling mechanics, size, design goals or explanations of failed scrapes. Baravalle et al. [11], Baravalle and Lee [10], and Hayes et al. [24] all describe the technical implementation of the scraper, however they do not explicitly mention the number of scrapes they collect nor the frequency. Dittus et al. [20] and Aldridge et al. [4, 6] use a single-shot scrape and merely provide details on the hyperlinks the scraper collected and followed. Similarly, Dolliver [21] uses a single scrape but only discusses the scraper design and mechanics. Van Wegberg et al. [47] describe the number and frequency of their scrapes, but do not discuss their scraper design. This finding of limited disclosure of crawling approaches in research is similar to the general survey on crawling methods in research from Ahmad et al., who sampled 350 papers that use a crawling methodology and found that 36% of their sample can be classified as *not repeatable* [3].

Post-processing. Once a market has been scraped, the relevant pages need to be post-processed before they can be analyzed. Basically, this means 1) parsing each page to extract salient information – e.g., listing title and vendor name, and 2) "cleaning up" the parsed data. More precisely, we look for discussion of parsing, deduplication, recoding, review-to-item listing matching, and completeness validation. Surprisingly to us, post-processing pipelines are seldom discussed across previous work. Completeness validation are most often discussed [5, 10, 15, 20, 41], but techniques are not standardized: authors instead employ a variety of custom strategies to assess the completeness of their datasets. Similarly, only a few authors describe their parsing procedures [10, 15, 41] and deduplication methods [15, 16, 41]. In two cases, the authors describe their recoding procedures [20, 21].

Proxies and heuristics. Parsed and cleaned data are then analyzed to provide insights about the market. This is where, for instance, researchers extrapolate from reviews to get a sense of economic revenue. Revenue can be defined as `price` $\times$ `sales`. Since both these features are not always directly scrapable, proxies have been used for analysis. We did find some consensus across the use of proxies and heuristics.

As hinted above, authors frequently use reviews left on listings as a proxy of transactions [4, 6, 16, 20, 41]. However, Celestini et al. cast doubt on this procedure [15]. Because some buyers will not leave a review, the number of reviews should always be considered as a lower bound for the number of sales. Most authors [4, 6, 10, 16, 41, 47] use the listing price as a proxy for the paid sales price. This proxy has two drawbacks, which are discussed in the aforementioned papers.

First, "holding prices," where vendors increase listing prices astronomically to signal an item is out of stock, are a known phenomenon across online anonymous markets, and authors employed various heuristics, mostly grounded in domain expertise and manual analysis, to filter/include them [6, 10, 41]. Second, the listing price changes over time, which means that a later price might differ from an earlier sales price. Only Soska and Christin [41] account for this by using the listing price scraped closest in time to a review timestamp.

Procedures to estimate the number of vendors either count the number of scraped pages directly [4, 15], or conditioned on activity, defined as having a listing in a given period of time [16, 41]. In terms of proxies not related to transactions, there is an apparent consensus in using shipping to/from destination for determining geographic location of items [11, 15, 16, 20, 21]. Item categorization (e.g., to determine whether a product is a narcotic, a prescription drug, or a weapon) sometimes relies on the marketplace's advertised categories [15, 16], sometimes on machine learning models [41, 47], or sometimes on manual analysis [20, 21].

External validation. Validating the reliability of collected data is an important step in online measurement studies. Past work employs a variety of external data sources to assess the reliability of the collected data. For instance, Soska

and Christin compare the data they collected against data contained in trial evidence, criminal complaints, and leaked pages [41]. Van Wegberg et al. also used criminal complaints for validation [47]. Similarly, Tai et al. also use court records in the context of vendor tracing across marketplaces [44]. Tai et al. complement their evaluation with a publicly available (at the time) crowd-sourced vendor database [8]. Last, Wang et al. compare their collected data against past studies [46].

Closest to our work, Rossy et al. use data collected by police following shutdown operations [37], and two efforts use ground truth data from back-end sources. Van de Laarschot and Van Wegberg use data from Hansa [28], and Bradley uses (partial) data from Silk Road 2.0 [12]. Interestingly, neither effort uses this back-end data for validation, but instead relies on it as ground truth for analysis.

For the validation of our own measurements, we will use all papers that have used Hansa data (either scraped or the database). These are Kruithof et al. [27], Lewis [29], Dittus et al. [20] and Van de Laarschot and Van Wegberg [28].

3 Methodology

We first formalize an abstraction for online anonymous marketplaces in Section 4. This abstraction can be used to test the impact of different hypothetical scenarios – e.g., what coverage do we get as we scrape more? Additionally, with accurate parameters, we can extend the insights that we derive from the specific marketplace we study, Hansa, to other marketplaces. We will later use this abstraction to model data collection and data analysis methods in simulated experiments.

We leverage three datasets for our analysis of losses when measuring marketplaces. The first dataset consists of scrapes collected from the public view of Hansa (Section 5.1). The second dataset is the Hansa database, i.e., the administrator's view, seized in the Hansa takedown operation by the Dutch National Police (Section 5.2). The third dataset is a set of simulated marketplaces that are scraped with different scraping procedures and parameters (Section 5.3).

We provide the first empirical measurement of scraping coverage based on ground truth data from Hansa in Section 6.1. By matching listings, reviews and users in a scrape to the same objects in the database at that moment in time, we measure both instantaneous and cumulative coverage. This experiment first confirms that not all objects are captured. In Section 6.2, we divide the objects in groups of *scraped* and *not-scraped* objects and show that significant differences exist between them, evidencing different biases in scraped data.

Revenue calculations are a key part of marketplace research. To better understand the impact of the biases that originate from incomplete scrapes and conservative heuristics, we calculate the revenue of one month of Hansa's revenue based on different data sources and different proxies and heuristics in Section 7. Based on these different revenues, we can define the different loss categories and their size. For example, how much revenue do you underestimate by using reviews as a transaction, versus orders as a transaction?

Finally, we conduct a series of simulations to understand the collection loss incurred by different scraping approaches in Section 8. We compare the coverage of one and two-shot scrapes and we estimate the impact of scraping consistency on coverage. We then explore the effectiveness of different abundance estimators. Certain pages may yield higher coverage than others (e.g. a listing with many reviews vs. one with none). Thus, given the adversarial environment of online anonymous marketplaces and the heavy impact that rate limiting has on coverage, we evaluated the design of a scraper that splits its scraping budget between rescraping listings with most feedback growth and discovering new listings.

4 Modeling Marketplaces

To reason about marketplace data collection and projections, we first need to mathematically define a model to express what a market is. We describe the model components in Section 4.1. We then describe data collection and analysis methods in Section 4.2 and Section 4.3, and the types of losses that arise from those functions in Section 4.4.

4.1 Model Components

Our model describes the relationships between the components of a marketplace: 1) the *states* of a marketplace, 2) the core *objects* of a marketplace, 3) the *views* that actors – such as a marketplace customer, vendor, or administrator – have of the state, and 4) the means by which states are *altered*. For instance, we consider a scrape to be a representation of one state, which observes various objects – such as reviews – through the public-facing view of the marketplace – i.e., what a customer would see – which does not alter the state.

States. The *state* of the marketplace, denoted σ_t for each time t, contains all of the information currently stored on the marketplace's back-end servers. Our focus in this work is on centralized marketplaces where a state takes the form of a database, which contains tables on marketplace objects. The marketplace *transcript* at time t is the complete history of all states from the beginning of the marketplace until t, namely $T_t = \bigcup_t \sigma_t$. If the marketplace does not support deletions of states then $T_t = \sigma_t$.

Objects. *Objects* are the core elements which constitute a marketplace. They are contained in a state, can be seen in a view, and can be altered through an operation. Objects include *users* (containing both vendors and customers), *item listings*, *reviews* and *transactions*. While there may be other objects that exist in a marketplace database (e.g., cryptocurrency wallets), a marketplace at least contains these. The objects themselves can have different attributes related to them. For example, a listing can have attributes such as price, shipping origin and item description.

Views. At any point in time, a marketplace offers different views to different actors. Most commonly, a customer can observe the marketplace state σ_t from a public view, which we denote σ_t^{public}. This view allows the actor to observe item price and previous reviews but may not have any information on hidden listings, or on listings deleted before time t. On the other hand, a marketplace administrator may be able to see all the information from the marketplace. The administrator view provides access to the collection of states σ_t^{admin}, which if complete, represent the marketplace transcript.

For the remainder of the paper, we assume that scrapes always rely on public views of the marketplace, while a marketplace take-down by law enforcement allows access to either the complete transcript (e.g., if the administrators kept backups of old states), or at least a partial view of the transcript. While out of scope for this work, it would also be desirable to consider vendor and moderator views, as law enforcement has been known to infiltrate these accounts, which represent a practical vantage point through which different signals can be extracted from a state. While these views are not as comprehensive as the administrator view, they should provide more information than is available in public views.

Operations. The state of the marketplace evolves via the *insertion* and *deletion* of objects, where *updating* the marketplace state is modeled as a deletion followed by an insertion. These operations affect σ_t, and thus all views of the marketplace and imply that future states are generally neither a proper subset nor a superset of previous states. Some operations can also affect specific views of the marketplace state. For instance, a *hide* operation on a listing, affects the public view but not the administrator view. On the other hand, the deletion of database backups or logs affect the administrator view but not the public view.

4.2 Data Collection

We define data collection functions as those which aim to retrieve the state of the marketplace at a time t and with a given *view*. The most common collection function is the *scraping* function, which uses $view = public$. We model a scraper that collects marketplace information from time m to time n as:

$$S_{m\to n}^{public} = \bigcup_{i\in[m,n]} x_i \leftarrow s(\sigma_i^{public}) \,. \tag{1}$$

Here, $s(\cdot)$ is a scraping function that takes in a marketplace state and returns the subset of data sampled according to a certain distribution. Typically, this function will either return the empty set when no information is collected on a particular state, or pieces of data representing the collective information on a few pages that were scraped.

4.3 Data Analysis

Data analysis is using data that has been collected, to measure any characteristic of the marketplace. Analysis functions mostly focus on taking the objects available in the public view (item listings, users and reviews) to approximate the objects available in the admin view (transactions). For instance, if one uses reviews as a proxy for transactions, we formally have:

$$|Tr_l^{admin}| \geq |R_l^{public}| \,, \tag{2}$$

that is, the number of actual transactions Tr for a listing l in the admin view will always be greater or equal to the number of reviews R for l present the public view. In other words, the number of reviews is a lower bound for the number of transactions. As discussed in Section 2, the functions applied to transform the "raw" collected object files to analyzable datasets are often overlooked in data analysis. These are mostly functions that combine different approximated states to one approximated transcript of a marketplace.

4.4 Losses

We define two broad types of loss in our model: collection loss and inference loss. First, *collection loss* results from any process which causes the data collection of a state to be different from the true state. Formally, between times m and n, for a given *view*, we have:

$$Collection\ Loss_{m\to n}^{view} = \left[\bigcup_{i\in[m,n]} \sigma_i^{view}\right] - S_{m\to n}^{view} \,. \tag{3}$$

(In the present discussion, $view = public$, but the loss definition generalizes to other views.) There are numerous sources of collection loss, including technical sources of loss (e.g., network errors, rate-limiting, backup loss), scraping-related losses (e.g., scraper design and website layout), and simply data loss that occurs over time due to data updates (e.g., deletion of objects from public view). In practice, collection loss can be defined as $1 - coverage$.

Second, we consider *inference loss*. For instance, to infer transactions, we need to match reviews to their corresponding listing. In this process, we may find matching and/or duplication issues which can lead to loss. For instance, attempting to detect when two *a priori* different reviews match the same sale (i.e., the buyer simply updated their feedback message) may lead to a loss, when this matching process reaches an incorrect conclusion.

5 Datasets

For our analysis we leverage three sources of data: Hansa scrapes (Section 5.1), Hansa database (Section 5.2) and simulations (Section 5.3).

5.1 Public View – Hansa Scrapes

We built our scraper using Scrapy [2], on top of Tor [19]. We scraped Hansa 17 times between late 2015 and mid-2017, collecting a total of 332,795 pages amounting to 39.5 GB of data.[3] The precise scrape dates can be found in Appendix 12. The scrapes provide a picture of Hansa during three periods of time: its initial stages (late 2015), its mature stage (mid-2016), and its peak prior to takedown (mid-2017). Out of the 17 scrapes, 3 of them failed due to authentication problems (due to cookies being invalidated), leaving 14 scrapes for analysis. Following the scraping, we proceeded to parse the pages and deduplicate entries. Below, we describe each of these processes.

Scraping procedure. We designed the scraper with *reliability* (to reduce data loss) and *stealth* (to prevent evasion) as primary goals. Our scraping algorithm was *depth-first* across parallel Tor circuits. To build the scraper we first performed a manual analysis of Hansa's layout. We then built a set of regular expressions for the URLs in the marketplace. This also allowed us to restrict certain requests to be sent when following links – e.g., add items to cart, checkout, etc. On session start, we provided the scraper with a session cookie manually obtained after solving a CAPTCHA. Scraping sessions ranged from a few minutes to a few days. When carrying out requests, our scraper randomly selects among a set of pre-built Tor circuits as a way of bypassing anti-DDoS mechanisms by "spreading the load" over multiple connections.

Ideally, we would want our scraper to instantaneously capture a snapshot of a marketplace, and to do so frequently. This would allow us to capture changes in the marketplace state as they happen and avoid missing objects that may be changed or deleted as time passes. In practice, however, we need to limit our requests so that we 1) do not alert the marketplace's operators and resultingly get blocked, and 2) do not significantly impact marketplace operations by flooding it with traffic. We performed approximately 12 requests per minute.

Timeline. During our initial scrapes (late 2015, early 2016) we observed slow growth in daily revenue – on average ∼$2,000 per day. As a result, we decreased our scraping rate throughout 2016 and early 2017, where Hansa had modest growth, and remained far behind competing marketplaces, notably Alphabay. Finally, following the Alphabay takedown in July 2017, Hansa saw a surge in popularity, so we began scraping frequently again.

Parsing and deduplication. We then extracted information from our scrapes through a parsing process. We iteratively adapted our parser to account for changes in the Hansa website over time which caused information, such as data fields, to be added, modified, and/or removed. One of our main parsing objectives was to ensure reviews are correctly paired with item listings, since this forms the basis of revenue calculation.

[3]The sanitized scrape data can be found at https://arima.cylab.cmu.edu/markets/

Scraping provides a snapshot of the marketplace (public) view at one point in time. Subsequent scrapes capture new information as well as substantial duplicate information. Deduplicating listings and vendors is trivial since they have unique identifiers. However, review deduplication is more challenging. We consider a review to be a duplicate if the author,[4] message, and timestamp[5] are the same and correspond to the same listing. We note that the review editing feature that Hansa provided may have caused a few overcounts given that it alters the timestamp of the review.

Author / Scrape Date	Vendors	Listings	Reviews	Est. revenue
Kruithof et al. (2016/1/11 → 2016/1/15)	219	4,829	–	–
This work (→ 2016/1/17)	282	5,987	2,847	$134,145
Lewis (2016/12/10→ 2016/12/16)	–	43,841	–	∼$3,000,000
This work (→ 2016/12/14)	840	21,185[6]	64,123	$2,885,133
Dittus et al. (2017/6 → 2017/7)	2,300	51,800	91,900[7]	–
This work (→ 2017/7/7)	1,639	48,330	186,893	$10,305,493

Table 1: Comparisons between Hansa studies. We include counts of reviews without price information. However, we omit them when estimating revenue.

External validation. We first validated the completeness of our scrapes by comparing them to information contained in other work on Hansa. Table 1 summarizes this comparison. Kruithof et al. conducted a scrape between January 11th and January 16th, 2016 [27]. Lewis conducted a scrape between December 10th and December 16th, 2016 [29] and Dittus et al. conducted a scrape "in late June to early July 2017" [20].

For all three datasets, we can directly compare our review counts since reviews are timestamped, which allows us to drop all reviews which do not fall in the scraping dates mentioned by the authors. However, in terms of listings and vendors, we can only do direct comparisons with Kruithof et al. and Dittus et al.'s datasets, since we have a Hansa scrape on January 17th 2016, and on July 7th 2017. This is because vendor and listing pages are not timestamped, so we cannot determine how many listings or vendors were present at the time of Lewis's scrape. Instead, we approximate the listings and vendors we had at the time of Lewis's scrape by only counting listings (and their corresponding vendors) which we had seen prior to July 7th and had more than one review. Table 1 shows that our scrapes mostly match measurements of earlier work. This is reassuring, given the scraping gap between 2016 and 2017.

5.2 Admin View – Hansa Database

We next use Hansa data obtained by the Dutch National Police on July 20, 2017 when the market was taken down [23]. At

[4]Regardless of username length, Hansa only displayed the first and last character of a review author with three asterisks in-between, e.g. a***b.

[5]Hansa originally provided timestamps with a one-minute granularity, before switching to a one-day granularity.

[6]We skipped 27,145 listings, unable to confirm their scrape date.

[7]The review discrepancy is likely caused by the fact that Dittus et al. focus on scraping "product catalogs," missing reviews left on vendor pages.

that time, the Dutch National Police had been running Hansa through a covert operation for exactly a month, starting on June 20, 2017. Using this data raises ethical considerations that we discuss in Section 9.

This data we have at our disposal is, in practice, a copy of the Hansa "back-end" database, that consists of 64 tables created by the marketplace administrators, as well as 76 back-up tables containing data from specific, earlier time periods. Using our earlier notations, we thus have both the "final state" of the marketplace, $\sigma_t^{admin} = \sigma_t$ (where $t =$"July 20, 2017") and some of the $\sigma_{t'}^{admin}$ for $t' < t$. We focus on quantifying measurement loss that occurs when we rely on scrapes of public views to reconstruct the entire market transcript (see Section 4 for definitions). For this analysis, we only need the data that pertains to the main objects (see Section 4.1) of the back-end database: listings, reviews, users, orders and transactions. Because older data was deleted as time went by, the final state of the Hansa market is not identical to the complete transcript of the market. Fortunately, the presence of back-up databases allowed us to partially recover that transcript. To that effect, we took the following preprocessing steps.

First, we noticed that a number of objects were present in different back-up tables. For each object type (e.g., orders, users, ...), we combined all of these records into a single, merged "complete" table. Whenever we found multiple records corresponding to a single object, we kept the most recent record. Second, we then pruned these complete tables to ensure they only hold data pertaining to "finalized" purchases, as opposed to aborted attempts. For instance, we filtered out of the complete order table entries referring to 1) orders without an associated transaction (money transfer), 2) orders that were declined by the seller, and 3) orders that were refunded. Similarly, we removed transactions between internal wallets to avoid double-counting transactions. Third, we checked data completeness in each table. Each table has an incremental unique identifier, which we can use to infer the amount of records purged from the database, simply by comparing the record count with the highest unique identifier.

Table 2 summarizes the outcome of our data processing. It shows the time period data is available from, the amount of records, the highest identifier, the percentage of missing data and finally the total amount of records available for analysis after filtering. The order table seemingly only holds roughly 50% of all orders, even when all the available backup tables are used. However, plotting the data over time, in Figure 2, shows a much more nuanced picture: order data is sporadically highly available, and sometimes completely missing. This shows that even after seizing a marketplace, one does not necessarily possess the ability to completely recreate the whole transcript of everything that happened during the marketplace's lifespan. In contrast with Van de Laarschot and Van Wegberg [28], who used the same dataset, we do not reconstruct purged orders by using their reviews as a proxy.[8]

Object	Time period	Records	Highest ID	Missing (%)	After filtering
Listings	2015/3/19–2017/7/20	123,143	123,969	0.67%	123,133
Reviews	2015/3/19–2017/7/20	258,184	260,853	1.02%	258,184
Users	2015/3/18–2017/7/20	419,323	432,287	3.00%	419,323
Orders	2015/6/17–2017/7/20	312,128	589,038	47.01%	192,708
Transactions	2016/1/28–2017/7/20	1,686,919	1,715,485	1.67%	505,883

Table 2: Marketplace objects from Hansa back-end

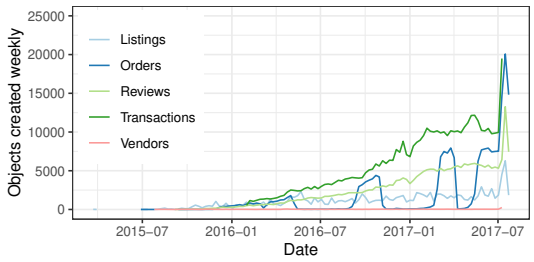


Figure 2: Weekly counts of objects from the Hansa back-end

5.3 Public View – Simulation

To derive insights on the impact of different scraping regimens and abundance estimation techniques on the quality of revenue estimation, we first generate (by simulation) fictitious marketplaces, that are similar to the Hansa marketplace [9], i.e., they feature similar objects and similar statistical parameters. As we discuss in Section 8.5, with the right choice of parameters, such simulations could reproduce other markets like Alphabay, Evolution, White House Market, Silk Road, etc. We then simulate different scraping routines on these markets. We begin with a formal description of the marketplace generation and scraping simulation processes, based on the model defined in Section 4. Figure 3 shows the entire process.

Following our model in Section 4.1, our simulation consists of four main *objects*: listings, vendors, reviews, and transactions. Additionally, we implement *operations* on each of these objects. Vendors and reviews can only be created, whereas listings can be created, deleted, set to hidden, or set to visible. Our simulations need five *inputs*: probability spaces, assignment functions, growth functions, a shaping function and a scraping function.

Probability space ❶ The probability space determines the sampling probability of each operation, e.g., probability a listing gets deleted, that a vendor is "created" (i.e., appears on the market), etc. Since the Hansa database provides final counts for the objects and operations we defined, we use this information to empirically define a probability space.

[8] Van de Laarschot and Van Wegberg used feedback to reconstruct missing order data, whereas we - for investigating differences between (scraped) reviews and orders - turned to data from the previously untapped and more complete transaction table.

[9] The code used for simulations can be found at: https://github.com/aledcuevas/dnm-simulation

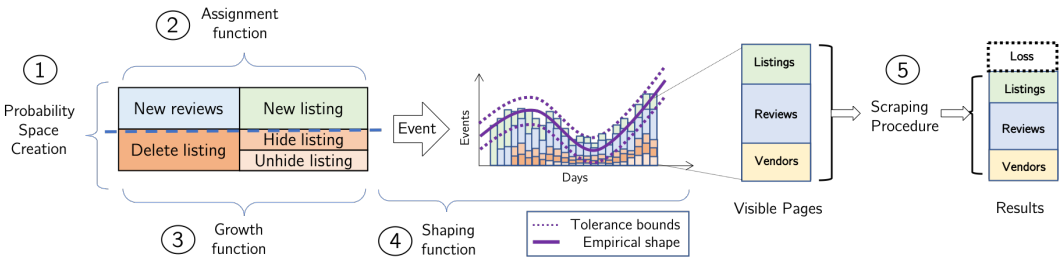


Figure 3: Steps involved in the generation and scraping of a simulated online marketplace.

Assignment function 2 Our objects have ordering and a set preference. For instance, vendors can exist in isolation, however listings must be created by a vendor, and reviews must belong to a listing. The way that each object is assigned to another is important in the context of hide and delete operations. For example, the distribution of reviews over listings can greatly impact a scraper's coverage, since the deletion of a listing with many reviews will cause a bigger loss if the scraper did not manage to capture this information before deletion. Thus, we define distributions for each assignment function (i.e., reviews to listings, listings to vendors).

Growth function 3 Certain operation probabilities depend on the quantity of the objects in the market. For example, the probability of deleting a listing is zero when no listing exists. However, as the market grows and the number of listings increases, the probability of a delete operation will also increase. As such, we define a growth function which adapts our probability space as the quantity of objects increase.

Shaping function 4 Once we have the probability of each operation, we need to add a time abstraction for the occurrence of events. For this, we employ a shaping function. Its purpose is to organize (or *shape*) the sequence of operations that take place over the lifetime of the marketplace simulation. Without a shaping function, each operation corresponds to a state transition from $\sigma_t \rightarrow \sigma_{t+1}$. Shaping allows the state transition to be over *epochs* corresponding to a number of operations.

Here, we define each epoch to represent a day. We allow a certain number of operations to take place before we proceed to the next epoch. So, we compute the moving average of the objects over the lifespan of the marketplace in days and summed them to derive an approximate shape for our events. Then, we define *tolerance bounds* around the average and allow the number of allowed events to be picked uniformly within the bounds. The tightness of the bounds determines the variability between each simulation. The simulation ends once either a certain number of operations have taken place or a certain number of epochs have passed. We also allow tolerance bounds around the allowed number of operations/days.

Scraping procedure 5 Last, we define a simulated scraping procedure. Our basic scraping procedure is parameterized by the frequency at which scrapes are conducted, the number of requests the scraper is allowed to conduct, and an error probability characterizing the risk of failure of the request. The scrapes are instantaneous. For each request, the scrape has access to the public view of the marketplace, that is, all public listings, vendors, and reviews; we call these *pages*. A page is scraped by uniformly drawing from the list of public pages. Each page retrieval counts towards the request cap, as well as a failed request.

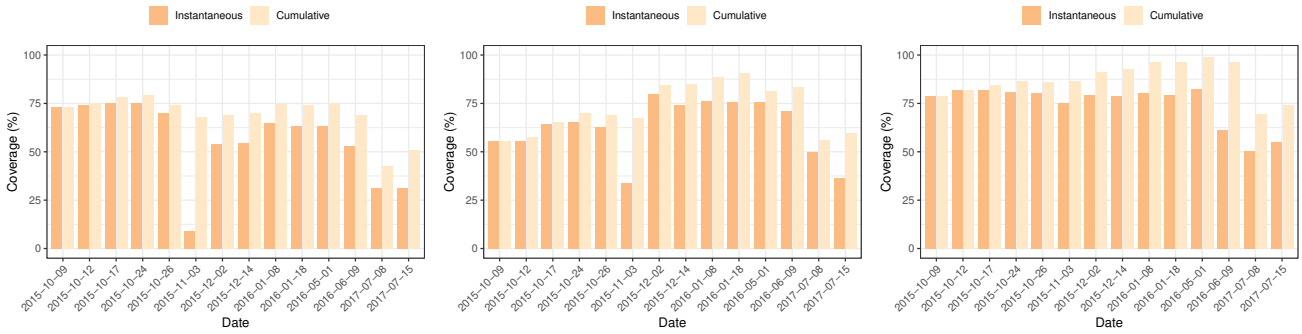
Simulation setup. We summarize the marketplace simulation and describe the parameterization we used. The probability spaces determine the frequency at which operations take place. Because we do not know the precise ordering of operations in Hansa's transcript, we assume that the probability of a specific operation is equal to the number of times the operation occurred over the total number of operations. The assignment function determines how objects are assigned to their parent sets. We compute the empirical distributions of object assignments (e.g., distribution of reviews across listings) from the back-end to handle this sampling. Since we do not know how the conditional probabilities of operations evolve as the number of objects vary, we lack empirical data to parameterize our growth function. Instead, we assume that probabilities scale linearly. For our shaping functions, we allow the tolerance bounds to be within $\pm 25\%$. That is, in a given epoch, we allow a minimum of 75% and maximum of 125% operations over our empirical values. Lastly, we allow $\pm 1\%$ bounds for the number of operations/days. These bounds are much narrower since we have more precise information to parameterize them.

6 Coverage and Bias

We measure scraping coverage by comparing scrapes and back-end data in Section 6.1. We then measure differences between the *scraped* and *not-scraped* objects to to empirically uncover scraping bias in Section 6.2.

6.1 Scraping Coverage

We define coverage as the percentage of objects from a scrape that can be matched to the database on the scrape date. We measure listing coverage, review coverage and active vendor



(a) Listings (b) Reviews (c) Vendors

Figure 4: Per-scrape instantaneous and cumulative coverage.

coverage.[10] For each scrape, we parsed and deduplicated the captured pages as explained in Section 5.1. This results in listing, review and vendor tables, which we use for our analysis. We compare these tables to the tables on listings, reviews and users directly derived from the Hansa back-end database (see Section 5.2). More precisely, we use the object creation dates to slice the Hansa back-end data into 42 sub-tables: one for each of the three object type (listings, reviews, vendors), on one of each of the 14 successful scrape dates. Each sub-table contains the relevant object data present in the market up to the corresponding scrape date.

We match listings between both datasets (back-end and scrapes) using the "listing ID," a numerical identifier present both in the database, and in the listing URL observable in the public view. We match vendors using vendor name. We match reviews using the tuple (review date, buyer name, vendor name, review message). For each of the 14 scrapes, we calculate the listing (L), review (R) and vendor (V) coverage using the following procedure. The input is an array T of 14 dates, the sets from a scrape (L_t^s, R_t^s, V_t^s) and the sets from a database slice $(L_t^{db}, R_t^{db}, V_t^{db})$. Then, for each $t \in T$, the coverage of that object type is calculated by taking the intersection between the scrape set and the database slice set, followed by calculating the percentage of the intersection to the database slice total size. For listings this is for example: $(L_t^s \cap L_t^{db})/L_t^{db} \times 100$.

Figure 4 shows the coverage over time for each object. The mean coverage is 56.61% for listings, 62.66% for reviews and 74.71% for vendors. Looking at the scrape dates, there is a large time gap between the 12th scrape (2016/6/9) and the 13th scrape (2017/7/8). Unsurprisingly, the coverage of the last two scrapes is as a result much lower than the average of the first twelve scrapes. This can be explained by these scrapes not capturing all of the listings and reviews that have been created and then hidden or deleted for the public view in the time between scrapes. A different explanatory factor can be the increased size of Hansa, which grew from 28,700 listings and 20,100 reviews in mid-2016 to 112,800 listings and 233,600 reviews in mid-2017, making it more likely for a scrape in 2017 to be unable to capture all objects in one go.

In general, the vendor coverage is the highest type of coverage with almost 75% of all active vendors being captured on average by scrapes. Comparing the listing coverage and the review coverage over time, we observe the review coverage to be lower than the listing coverage for the first six scrapes. From December 2015 onward, however, the review coverage of each scrape is higher than its listing coverage. This could indicate that while a scrape captures less of the total inventory of listings, the listings it does capture are responsible for a larger proportion of all available reviews.

To give insights on how subsequent scrapes influence the cumulative coverage, Figure 4 shows the cumulative coverage when all scrapes are combined. While instantaneous scrape coverage does not improve, the increase in cumulative coverage shows that consecutive scrapes capture different objects. Thus, in most cases the combination of two consecutive scrapes leads to a higher cumulative coverage than the average of the two scrapes separately. The cumulative coverage of our scrapes for the market up to and including 2017/07/15 is 50.83% for listings, 59.49% for reviews and 73.93% for vendors. Hence, the empirical collection loss on Hansa is 49.17%, 40.51% and 26.07% for listings, reviews and vendors respectively. The average of these coverages weighted by their counts is 53.84%, meaning that on average just a bit more than half of all available objects was scraped.

6.2 Scraping Bias

We just showed that even after 14 scrapes, a non-negligible number of listings, reviews and vendors have still not been captured. From the back-end data, we also know that listings could be hidden and reviews deleted, making them disappear from the public view. In what way then is a scrape a truly random sample from the total population of available objects?

[10] We distinguish active vendors, as public views do not provide information on inactive vendors – i.e., those that have no listings on Hansa.

	Tests			Scraped listings $n = 61,248$				Not-scraped listings $n = 61,885$			
Variable	Test	Statistic	p-value	M	μ	σ	min-max	M	μ	σ	min-max
usdPrice	M-W U	1.6×10^9	0.00	30.00	390.18	2,739.08	0.01–3.2×10^5	66.48	625.50	6,508.99	0.01–1.0×10^6
views	M-W U	1.4×10^9	0.00	637.00	2820.82	12,569.63	0.00–270,251	232.50	1,536.93	5,438.36	0.00–251,554
numReviews	M-W U	1.8×10^9	≤ 0.001	0.00	2.90	18.71	0–1,313	0.00	1.30	11.47	0.00–2,114.00
ageListing	M-W U	1.7×10^9	≤ 0.001	239.00	267.64	206.86	5–728.00	207.00	224.96	149.12	1.00–855.00
isHidden	χ^2 test	5.9×10^3	0.00	0.00	0.05	0.22	0–1	0.00	0.20	0.40	0–1
isDeleted	χ^2 test	2.4×10^4	0.00	0.00	0.05	0.22	0–1	0.00	0.20	0.40	0–1
soldNoReview	χ^2 test	558.61	≤ 0.001	0.00	0.05	0.21	0–1	0.00	0.02	0.15	0–1
category	χ^2 test	9.0×10^3	0.00								

Table 3: Results of the Mann-Whitney U and χ^2 tests between scraped and not-scraped listings

To answer this, we analyze the differences between scraped and not-scraped listings. Differences between scraped and not-scraped vendors and reviews come down to whether or not the corresponding listings are scraped. Indeed, comparing the characteristics of scraped and not-scraped vendors shows that 99.95% of the scraped vendors have a listing and 98.86% have a listing that is scraped. For reviews (given the necessary pairing between a review and its listing) the percentages are even higher, with 100% of the scraped reviews having their paired listing scraped and 99.84% having the corresponding vendor scraped. This means that whether a review or vendor is scraped ultimately depends on whether the listing is scraped. This is because a review is scraped only when the vendor or the corresponding listing is scraped, and a vendor is scraped when A) it has a listing that B) is scraped. (See Tables 5 and 6 in the appendix for the descriptive statistics and tests for vendors and reviews.)

We next explore features that could be correlated with the chance of an object being scraped (e.g., the object being hidden) and features that can influence revenue calculations (e.g., the price of the object). To make sure we test features that have small inter-dependencies and thus capture different variations of why an object is not scraped, we performed an exploratory factor analysis on the listing features. As we did not discover any latent factors, we will not use the factors nor loadings themselves. The analysis and descriptive statistics of the factor analysis can be found in Appendix 11.2. The subset of features then is `numReviews`, `ageListing`, `views`, `usdPrice`, `isDeleted`, `isHidden`, `category` and `soldNoReview`.

We performed Mann-Whitney U [30] and Chi-Square [35] tests between the scraped and not-scraped groups, to test for significant differences. The results in Table 3 show that *all* features differ significantly between the scraped and the not-scraped listings. Since not-scraped listings have less `views` and a lower number of reviews, `numReviews`, on average, this could point in the direction of a scrape being biased through "popularity". This is supported by a lower average `usdPrice`, as lower priced products are seen and sold more as they are more popular than higher priced listings. The features `ageListing`, `isHidden` and `isDeleted` influence the scraping process as we would expect: the longer a listing is available (and not hidden or deleted) on the market, the higher the probability the listing is scraped. The feature `soldNoReview` (i.e., the listing had sales, but no reviews) is relevant for a specific type of listing, namely *custom* listings [16]. Such listings sell a specific (larger) quantity and are created for a single buyer, who often does not leave a review. Surprisingly, a larger percentage of the scraped than the not-scraped listings was bought without anyone leaving a review. Finally, comparing the categories of scraped and not-scraped listings, we found that while on average $\approx 46\%$ of a category is scraped, "Digital Goods" listings were scraped more often ($\approx 77\%$), while "Weed" listings were more often not scraped ($\approx 35\%$).

7 Revenue Calculations

We next compare projected revenues from our scraped data, to the actual revenues we can infer from the back-end database.

Projected revenue. Projecting market revenue from scraped data requires the use multiple proxies and heuristics. First, we detect and remove holding prices. Second, we pair reviews to listings, to approximate the actual price paid by the advertised price closest in time to when the review was left. Multiplying the number of reviews left every day by the listing prices gives us daily revenues in Bitcoin, which we convert to US Dollars using exchange rates from Coincap [1] for the corresponding dates. From there, we get the total revenue for a listing by summing these daily revenues over the lifespan of the listing; and the total projected revenue for the entire market, by summing the revenues for all listings.

Actual revenue. We next compute the *actual* market revenue from the Hansa back-end database. Because the transaction table only holds data from 2016/1/28 onward, we add revenue from order data for 2015/6/17–2016/1/27 to the revenue from transaction data for 2016/1/28–2017/7/20. For the revenue computation to be perfectly reliable, we would need the complete marketplace transcript; the Hansa back-end database, albeit very comprehensive, is not perfectly complete, as described earlier. However, based on the missing data percentages from Table 2 we assume that it is a very close approximation of ground truth data.

Loss. As discussed earlier, projecting revenue from scrapes produces two loss types: (i) an inference loss, due to using proxies and (ii) a collection loss, due to using data with incomplete coverage of reviews and listings. To estimate the size of the inference loss, we reproduce our projection calculations using, this time, data from the Hansa back-end database that would have been publicly available for scraping. In essence, this allows us to simulate what we would have gotten if we had "perfect scrapes" that captured all the information ever made publicly available by the market. Since we know, from Table 2, that review and listing data is 98.98% and 99.33% complete, respectively, the difference between our earlier projected revenue computation and this computation with perfect scrapes will approximate the inference loss well.

The total market revenue projected from scrapes is $13,149,373. When the revenue is calculated based on all the reviews available in the back-end database ("perfect scrapes"), this number rises to $27,385,346. The final number of total marketplace revenue for Hansa from transactions and orders is, however, $50,056,008. Shortly stated, inference loss causes a 50% drop, and collection loss seem to cause another 50% loss, resulting in a projected number that is only slightly more than a quarter of the actual market revenue.

Where does the loss come from? We next attempt to discover the causes for these losses. We use one month – March 2017 – for this, since full order data is available for that month, Hansa had matured enough that, at that point, it was generating millions of revenue each month, but was not yet growing exponentially as it did later in 2017.

We calculate the revenue that month based on five different inputs: 1) the scraped reviews 2) the reviews from the database 3) the orders with the single quantity price 4) the orders with the item price 5) the orders with the full paid price (incl. shipping). The difference between 1) and 2) reflects the collection loss for this time period. The difference between 2) and 3) captures the inference loss from using reviews as a proxy for sales (orders), when not all customers leave reviews. The difference between 3) and 4) is the inference loss coming from assuming unit quantities for each inferred transaction. Finally the difference between 4) and 5) is the inference loss due to ignoring shipping costs.

Figure 5 shows these revenue calculations based on different inputs. The gap between scraped reviews and reviews from the database is about a factor of two – $1,179,993 and $2,548,941 respectively. This collection loss of 53.71%, is in line with our findings in Section 6. The inference loss when using reviews as a proxy for sales is 21%, which translates to $809,101 in revenue. The difference between orders with the price for a singular quantity (3) and orders with an item price (4) is $353,863 (9.18%) in revenue, and the final difference between the orders with full price paid and orders with item price is just $141,435 (3.67%).

Take-aways. In short, achieving good scraping coverage is essential to get reliable estimates. Transactions without reviews present a major challenge. Without additional information from the market (e.g., the total number of sales for an item, as displayed by Alphabay), it is impossible to infer whether the transaction occurred. The extent of this problem depends on the "social norms" of the market: the original Silk Road, for instance, reportedly strongly incentivized buyers to leave a review [16], whereas, evidently, compliance is a lot looser on Hansa. Finally, assuming away shipping costs and orders for multiple quantities of the same item seems to bear little impact on the projections.

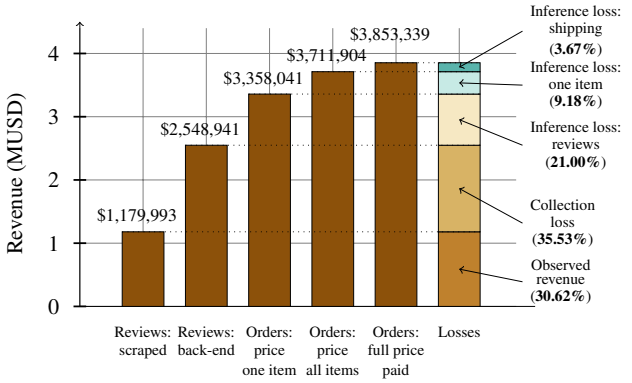


Figure 5: Calculation of Hansa's March 2017 revenue with different inputs

8 Simulation

Through simulations (see Section 5.3) we explore marketplace coverage when varying the frequency, consistency, and rate-limiting of scrapes (Section 8.1 and Section 8.2). We present a comparison of abundance estimators in Section 8.3. Last, we propose and test a new, popularity-driven, scraper design.

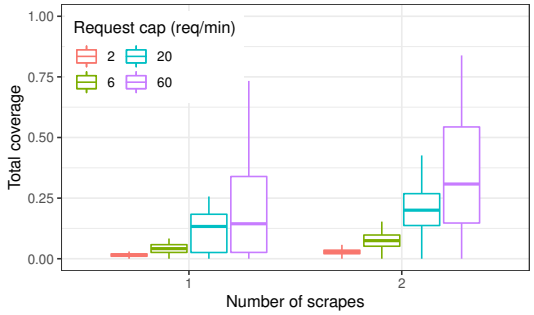


Figure 6: Distribution of coverage for one and two-shot scrapes simulated across different request limits.

8.1 Coverage of One and Two-shot Scrapes

We first quantify the coverage loss for our simulated marketplaces. Given that many studies rely on only one or two scrapes [6, 13, 20, 21], we compute the coverage distribution for both scenarios. First we simulate markets where only one scrape is available; we repeat this simulation for every single day the market is live. We then compute the expected coverage for each possible day in the simulation. Then, we simulate markets where two scrapes (taken on different days) are available. We run this simulation for every possible pair of days among the days the market was live. We then compute the expected coverage for all possible combinations of scrapes. Further, we conduct these experiments with different page request limits: 2 req./min. (2,880 daily), 6 req./min (8,640 daily), 20 req./min (28,800 daily), 60 req./min (86,400 daily). In total, we simulated 2,800 scrapes for one-shot scrapes, and over 1,897,000 two-shot scrapes.

Figure 6 shows the results, using box plots with 95% confidence intervals. Even when scraping a page every second, the median coverage is low in the one-shot case (0.144) and only moderately better in the two-shot case (0.308). The theoretical maxima are 0.733 and 0.840 for the one and two-shot cases, respectively. However, in practice, 60 req./min. is rarely achievable due to the presence of anti-scraping mechanisms (e.g., CAPTCHAs, temporary bans, rate-limiting, etc.) [45].

8.2 Coverage and Scraping Consistency

We next seek to understand how coverage increases as the number of scrapes increase. Further, given that most past work we reviewed does not follow a consistent scraping schedule, we want to differentiate the impact on performance between consistent and inconsistent scraping routines. So, we compare the final coverage of all pages obtained between: 1) evenly spaced scrapes and 2) scrapes which are done at random intervals. For both settings, we calculate the coverage as we increase the number of simulated scrapes from 3 to 30. For each setting, we conduct simulations until our results converge into a narrow 95% confidence interval; this amounts to over 30,000 simulations.

Figure 7 shows that increasing the number of scrapes yields diminishing returns as the number of scrapes increases, mirroring Soska and Christin's findings [41]. We find that not following a scraping routine is not necessarily detrimental to the coverage. However, it is important to caveat these results with the fact that the random scraping days were computed with *a priori* knowledge of the lifetime of the market. For continually growing markets (until takedown), such as Hansa, later scrapes have a greater chance of contributing more information to the final coverage. Thus, the more scrapes we have around periods of time growth, the better the coverage. On the other hand, if objects are frequently removed from the public view (e.g., deletions), then a consistent scraping routine might perform better since it has greater chance of catching data before the public view changes. In essence, we do not expect to see major differences in coverage between studies that did not follow a consistent scraping routine, as long as their scrapes are not concentrated in the early stages of the market.

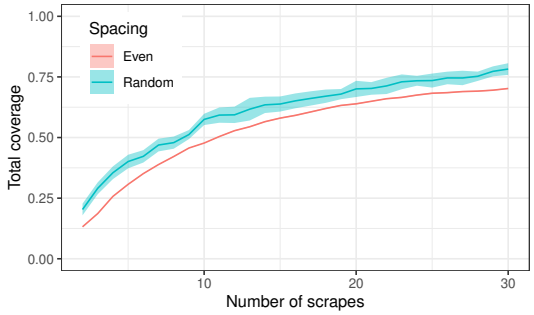


Figure 7: Scraping coverage as the number of scrapes increases, with evenly spaced scrapes and randomly spaced scrapes. The shaded area is the 95% confidence interval.

8.3 Comparison of Abundance Estimators

We have evaluated scrape coverage using the ground truth contained in the back-end data. In practice, however, public views do not always provide features to help us determine the size of the population for each object.[11] Instead, past work has relied on abundance estimators to calculate scraper coverage or collection loss. For instance, Soska and Christin used the Schnabel estimator [38] to estimate coverage [41]. Coverage estimations can then be used to extrapolate revenue, missing data, or adjust scraping regimens.

Abundance estimators, however, have not been evaluated in the context of online marketplaces. Thus, we proceed to evaluate the Schnabel estimator, along with the Lincoln-Petersen (LP) estimator, and the Jolly Seber (JS) estimator on our simulated marketplace. These estimators are part of a family of methods known as "mark and recapture," derived from tagging and recapturing experiments used to estimate wildlife populations [39]. A summary of these algorithms is given in Appendix 11.3. At a high level, LP is the simplest estimator and assumes the population is constant, and estimated from two population samples; Schnabel extends LP to account for repeated sampling; Jolly-Seber extends these algorithms to a situation, like here, where the population changes over time.

We implemented each of the three estimators and used them in our simulation. We validated the LP and Schnabel estimators using the capture histories of northern pike data [9] in the R `FSAdata` package and the procedure described by Ogle [34]. For the JS estimator, we used the implementation

[11] Most markets list the total number of items; some give the number of vendors; very few give the number of transactions per listing.

Algo. \ Coverage	Bi-Weekly Low	Bi-Weekly High	Monthly Low	Monthly High	Quarterly Low	Quarterly High
Jolly-Seber	0.501	0.081*	0.451	0.163*	0.401	0.338*
Lincoln-Petersen	0.219*	0.226	0.251*	0.249	0.358*	0.356
Schnabel	0.603	0.455	0.583	0.457	0.57	0.467

Table 4: Avg. error when estimating the number of listings across scraping intervals and using either a low request limit (2 req./min) or a high request limit (20 req./min).

provided by the `MARK` package, a well-known and widely used package for mark-and-recapture models [49].

We performed experiments in six different settings, varying the frequency and coverage of our scrapers. We tried three scraping frequencies: bi-weekly, monthly, and quarterly. We paired these with either a low request limit (2,880 requests per scrape; 2 req./min.) and a high request limit (28,880 requests per scrape; 20 req./min.). For each simulated scrape, we estimated the population of listings in the market based on prior captures and recaptures. We then computed the average collection loss for each scraper configuration across all our simulations. We repeated the simulations until we narrowed our 95% confidence interval; this took over 9,000 simulations.

Results. We summarize our results in Table 4. We observe that the JS estimator performs best in scenarios where our scrape has higher coverage. The JS estimator provides the best estimates when scraping frequently and with high coverage. However, the LP estimator performs better when coverage is poorer. This is because higher estimates are preferable when there is low coverage, and the LP estimator provides high estimates when there is low coverage. Surprisingly, the Schnabel estimator, which yielded good results in earlier work [41], performs here quite poorly across all settings.

8.4 Popularity-Driven Scraping

As explained in Section 6.2, certain pages are more critical to achieve good coverage than others. For instance, a listing page with a lot of reviews is more important to scrape properly than a listing with zero reviews. Previous work has hinted that, in terms of popularity, listings and vendors follow long-tailed distributions [41]. Thus, we hypothesize that one may achieve good coverage by primarily focusing on the most popular vendors and listings. While, ideally, one would want to scrape everything, it may not be possible: marketplaces have been deploying increasingly strict anti-scraping measures, which limit the ability of a third party to collect information [45]. We next explore whether "popularity-driven scraping" provides good coverage when facing a limited scraping "budget".

More precisely, we assume that we are given a limit ℓ on the number of requests our scraper can issue (e.g., 2 requests per minute), and that we control the proportion ρ of previously seen pages we want to scrape again. We sort listings by popularity, i.e., by the number of reviews they have.[12] We rescrape the most popular listing pages until we hit $\rho\ell$ pages; we then scrape $(1-\rho)\ell$ pages we had not seen before.

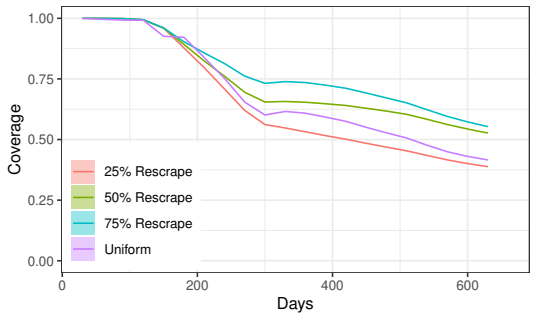


Figure 8: Average scrape coverage through a simulated market's lifetime for various popularity scraping budgets and compared to our uniformly random scraping baseline.

We simulate three different parameters choices for ρ: 25%, 50%, and 75%, with $\ell = 6$ req./minute, over a 30-day interval. We conduct experiments until we sufficiently narrow our 95% confidence interval; here, this takes slightly over 20,000 simulations. We compare popularity-driven scrapers against our baseline, which is to scrape uniformly at random from the set of available pages. We present the mean coverage at each scrape date over all our simulated markets.

Results. Figure 8 shows that the scraper with $\rho = 75\%$, performs the best, with an average coverage of 0.765, followed by the scraper with a $\rho = 50\%$ rescraping budget (average coverage of 0.725). The baseline, random scraper, achieves a 0.674 coverage. Perhaps surprisingly, a scraper $\rho = 25\%$ budget performs *worse* than the baseline, with a 0.638 coverage. In short, a popularity-driven scraping approach can substantially increase coverage—as much as 10% higher than the baseline—as long as it is properly parameterized. Also, the difference in coverage widens as the market grows, which, in Hansa, was the case toward the end of the market's life.

8.5 Extrapolation

An optimal scraper for Hansa is contingent on a set of features that may not be shared by other markets. Hansa was a market with no established deletion policy, as opposed to others. For instance, Dream Market deleted reviews older than 150 days [18]. Likewise, the recently deposed Russian-language Hydra Marketplace[13] purged reviews older than 240 days.

Thus, a scraper that follows a consistent routine would likely ensure more reliability and coverage. Hansa also experienced a burst of growth, following the Alphabay takedown, which occurred and lasted for a small period of time towards the end of the market. With constrained resources (i.e., limited number of scrapers and number of allowed requests), an optimal routine would have sporadically scraped Hansa during its

[12] For the first scrape, all listings are assumed to be equally popular.

[13] No relationship to an older Hydra market active in 2014–2015 [41].

slow period and aggressively scraped during its meteoric rise.

However, not only it may be hard to establish this routine *a priori*, but other markets follow different patterns, even following takedowns. For instance, Soska and Christin [41] show that older markets like Pandora or Agora had various bursts of revenue throughout their lifetime. These different growth patterns may call for different routines. Thus, when facing a new market, researchers may want to simulate different possible growth patterns and market lifetimes and choose the most robust strategy.

Lastly, a popularity-driven approach is an efficient choice for studies where we can infer where high-yield objects will be located (such as a revenue estimation study). For example, reviews on Hansa were largely concentrated among a handful of vendors, which is intuitive since listing popularity on anonymous markets has historically tended to follow Pareto-like distributions.

9 Discussion

This works brings up ethical considerations, especially as they relate to the use of seized data, which we discuss next. Second, while our results show that scraping as a measurement approach can introduce significant losses, we explain why this paper should not be seen as an indictment of scraping—quite the contrary. Third, we discuss other contexts such as fora and other online shops. Last, from our observations, we derive a set of best practices for scraping online markets.

Ethics. For our scraping measurements, we followed Martin and Christin's recommendations [31], and took proactive steps to minimize direct and indirect consequences that our measurements may have had on marketplace participants and on Tor users. (For instance, we purposefully limited our scraping regimen, did not interact with marketplace actors, etc.) Similar to earlier work [28,33], all of our analyses of the back-end data were conducted on-site at Dutch law enforcement agencies, and the data was stored and protected under their safety and security guidelines. The data was made accessible to us for academic research purposes. Extracting aggregate data points for our tables and figures was done under strict supervision through one specific monitored channel. A Dutch law enforcement privacy-officer vetted that the data contains no personally identifiable information.

As we obtained the approval of the Dutch Public Prosecution Service for our analysis, the Delft IRB viewed this work as outside of their jurisdiction and were satisfied with this assessment. The three authors at US institutions did not directly interact with back-end data. The Carnegie Mellon IRB had earlier opined, and confirmed, that scraping marketplace data (without personal identifiers) did not constitute human-subject research.

Most importantly, this study does not, and does not seek to, provide any legal proof of criminal conduct.

Value of scraping. While our results show that scraping can result in significant loss, ground-truth data is rarely, if ever, available. Seized back-ends are rare – and may be very far from complete when they exist. We discovered that Hansa's database holds many features unavailable in the public views. However, a major drawback is that this database only contains a *single* record for each object. Absent any back-up (which were available here, due to the Hansa administrators espousing questionable data retention practices), one would only be able to see the *latest* version of each object. On the other hand, a consistent weekly scraping regime could have captured 108 versions of each object in Hansa's lifetime. Doing so allows to understand historical price developments, vendor PGP-keys changes, and vendor geographic shipping information – all important data points for revenue analysis and vendor matching [44].

Other contexts. The issues of incomplete data and the usage of proxies and heuristics for (revenue) calculations are not limited to the domain of online anonymous marketplaces. Other marketplace contexts, such as online fora (e.g. hacker fora) or specific web shops (e.g. pharmaceutical websites), also face the challenge of doing empirical marketplace research in adversarial contexts. This has two consequences.

First, online anonymous marketplace research can learn from approaches on these other types of marketplaces. Different internal and external validation techniques from other works could also be applied. Two notable examples are calculating completeness of a scrape through leveraging unique marketplace identifiers (e.g., changing URLs or sales counters [26]) and cross-referencing tables and checking concordances between transactional data and metadata [32].

Second, the present study can serve as a model for these other contexts. As Portnoff et al. [36] note in their analysis of an underground forum marketplace: "an analysis relying on both private and public data vs. just public may reach different conclusions about the revenue of a market." More broadly, Andreas and Greenhill in "Sex, Drugs, and Body Counts" show how scientific measurement errors often motivate inappropriate policy choices [7]. As all these types of fora or unlicensed shops primarily deal in illegal offerings, precision is of the utmost importance.

Best practices. Our findings can inform future online anonymous markets measurement studies, both for study design and for reporting results. First, we recommend *frequent* and *periodic* scraping to mitigate the impact of scraping errors, rate limits, and data deletion. When describing data collection, studies should disclose when the scrapes were obtained and the number of requests that were sent. To contextualize the potential coverage of their scrapes, studies should try to estimate the size (i.e., pages) of the site. While abundance estimation can help, markets may offer metadata that provide a better starting point for estimation. For instance, markets may disclose the number of vendors, items, or even the number of orders that each vendor has fulfilled. These results

can then be complemented with estimates derived from Jolly-Seber or Lincoln-Petersen models for high and low coverage assumptions, respectively.

In the face of limited scraping budgets (e.g., as caused by anti-scraping mechanisms), future studies should consider identifying and focusing their scraping on high-yield portions of the website, rather than scraping in a breadth-first fashion. Rate of growth can be measured through observed changes in subsequent scrapes (as we described in Section 8.4) or through metadata (e.g., a leaderboard of reputable vendors). Further, we also recommend that future studies provide more detail on their scraper *design*. We found that scraper design is often either not discussed or described with insufficient detail in the literature (Section 2.2). Yet, understanding how the scraper traverses pages, the number of requests it performs, or how it adapts to adversarial scraping environments are all important details that help contextualize the coverage of the measurements, and subsequently its impact on estimation.

Last, our research showed that the "measurement-by-proxy" approach provides a *very* conservative lower bound for revenue estimations on online anonymous marketplaces. If the assumption of similar review-to-transaction ratios holds for a newer marketplace (e.g., feedback is neither mandatory nor automatically purged over time), our loss factors from Section 7 can help calculate an upper bound for revenue projections. That way, future research can take the biases we discovered into account and reason about the impact of calculating revenue based on scraped data on measurement outcomes.

10 Conclusion

We investigated the accuracy of marketplace measurements using the Hansa Market back-end database, 14 Hansa scrapes and more than 60,000 simulations of over 2M scrapes. Our results show that "measurement-by-proxy" can result in significant collection and inference loss. We find the collection loss of Hansa scrapes to be around 46% in objects ever generated on the marketplace. Further, a scrape does not uniformly randomly draw from the population, since captured listings differ significantly from the not-scraped listings.

The inference loss introduced by proxies, such as reviews and listing prices, corresponds to 34% in monthly revenue. Unfortunately, the scarcity of complete back-end data sources present researchers with little alternatives to measuring-by-proxy, and inference loss cannot be easily mitigated.

Our main take-away is thus to focus on mitigating collection loss. Our simulations yield insights on how to achieve this objective. Scraping a marketplace just once or twice is likely to result in very poor ($< 50\%$) coverage. While scraping frequently outperforms scraping consistently, getting from 60% to 80% coverage almost requires doubling the amount of scrapes. Innovative scraper design, such as scraping most popular listings first, can help improve coverage when the scraping budget is limited. Finally, abundance estimators, albeit imperfect, can provide insights on population coverage in the absence of data on population sizes.

Acknowledgments

We are grateful to Dutch law enforcement for their enduring trust that allowed us to use the Hansa back-end for this paper. We thank our reviewers for their constructive feedback, and Gianluca Stringhini for shepherding this paper. Likewise, we thank Bas Stinenbosch, Michel van Eeten, and Gert-Jan van Hardeveld for their contributions and suggestions on earlier versions of this work. This research was partially supported by the US Dept. of Homeland Security (Office of Science and Technology) and the US Air Force Research Laboratory (AFRL) under agreement number FA8750-20-1-1003; and by the Singapore Defence Science and Technology Agency (DSTA) under agreement CNZ2000832.

References

[1] CoinCap API 2.0. https://docs.coincap.io.

[2] Scrapy: An open source web scraping framework for Python (v.1.0.0-1.4.0). http://scrapy.org.

[3] S. Ahmad, M. Dar, M. Zaffar, N. Vallina-Rodriguez, and R. Nithyanand. Apophanies or epiphanies? how crawlers impact our understanding of the web. In *Proc. Web Conf.*, pages 271–280, April 2020.

[4] J. Aldridge and D. Décary-Hétu. Not an 'Ebay for drugs': The cryptomarket 'Silk Road' as a paradigm shifting criminal innovation, May 2014. Available at SSRN: https://ssrn.com/abstract=2436643.

[5] J. Aldridge and D. Décary-Hétu. Cryptomarkets and the future of illicit drug markets. In *The Internet and drug markets*, pages 23–32. EMCDDA, 2015.

[6] J. Aldridge and D. Décary-Hétu. Hidden wholesale: The drug diffusing capacity of online drug cryptomarkets. *Int. J. Drug Policy*, 35:7–15, September 2016.

[7] P. Andreas and K. Greenhill, editors. *Sex, Drugs, and Body Counts: The Politics of Numbers in Global Crime and Conflict*. Cornell University Press, 2010.

[8] Anonymous. Grams: Search the Darknet, 2017. Was at http://grams7enufi7jmdl.onion. Taken offline in December 2017.

[9] New York Power Authority. Use of buckhorn marsh and grand island tributaries by northern pike for spawning and as a nursery. Technical report, FERC, 2004.

[10] A. Baravalle and S. Lee. Dark web markets: Turning the lights on AlphaBay. In *Proc. WISE 2018*, pages 502–514, November 2018.

[11] A. Baravalle, M. Lopez, and S. Lee. Mining the dark web: Drugs and fake ids. In *Proc. IEEE ICDM 2016 Workshops*, pages 350–356, December 2016.

[12] C. Bradley. *On the resilience of the Dark Net Market ecosystem to law enforcement intervention*. PhD thesis, University College London, 2019.

[13] J. Broséus, D. Rhumorbarbe, C. Mireault, V. Ouellette, F. Crispino, and D. Décary-Hétu. Studying illicit drug trafficking on darknet markets: Structure and organisation from a Canadian perspective. *Forensic Sci. Int.*, 264:7–14, July 2016.

[14] R. Cattell. The scree test for the number of factors. *Multi. Behavior. Res.*, 1(2):245–276, 1966.

[15] A. Celestini, G. Me, and N. Mignone. Tor marketplaces exploratory data analysis: The drugs case. In *Proc. ICG3S*, volume 630, pages 218–229, January 2016.

[16] N. Christin. Traveling the Silk Road: A measurement analysis of a large anonymous online marketplace. In *Proc. Web Conf.*, pages 213–224, May 2013.

[17] N. Christin. An EU-focused analysis of drug supply on the AlphaBay marketplace, October 2017. EMCDDA commissionned paper.

[18] N. Christin and J. Thomas. An analysis of the supply of drugs and new psychoactive substances by EU-based vendors via darknet markets in 2017–18, November 2019. EMCDDA commissionned paper.

[19] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. In *Proc. USENIX Sec. Symp.*, August 2004.

[20] M. Dittus, J. Wright, and M. Graham. Platform criminalism: The 'last-mile' geography of the Darknet market supply chain. In *Proc. Web Conf.*, pages 277–286, 2018.

[21] D. Dolliver. Evaluating drug trafficking on the Tor Network: Silk Road 2, the sequel. *Int. J. Drug Pol.*, 26(11):1113–1123, November 2015.

[22] D. Décary-Hétu and L. Giommoni. Do police crackdowns disrupt drug cryptomarkets? A longitudinal analysis of the effects of Operation Onymous. *Crime, Law and Social Change*, 67(1):55–75, February 2017.

[23] Europol. Massive blow to criminal dark web activities after globally coordinated operation, 2017. https://www.europol.europa.eu/media-press/newsroom/news/massive-blow-to-criminal-dark-web-activities-after-globally-coordinated-operation.

[24] D. Hayes, F. Cappa, and J. Cardon. A Framework for More Effective Dark Web Marketplace Investigations. *Information*, 9(8):186, July 2018.

[25] J. Horn. A rationale and test for the number of factors in factor analysis. *Psychometrika*, 30:179–185, 1965.

[26] C. Kanich, N. Weaver, D. McCoy, T. Halvorson, C. Kreibich, K. Levchenko, V. Paxson, G. Voelker, and S. Savage. Show me the money: Characterizing spam-advertised revenue. In *Proc. USENIX Sec. Symp.*, August 2011.

[27] K. Kruithof, J. Aldridge, D. Décary-Hétu, M. Sim, E. Dujso, and S. Hoorens. Internet-facilitated drugs trade: An analysis of the size, scope and the role of the Netherlands, 2016. RAND corporation. https://www.rand.org/pubs/research_reports/RR1607.html.

[28] J. van de Laarschot and R. van Wegberg. Risky business? Investigating the security practices of vendors on an online anonymous market using ground-truth data. In *Proc. USENIX Sec. Symp.*, pages 4079–4095, August 2021.

[29] S. Lewis. OnionScan report: Reconstructing the finances of darknet markets through reputation systems, January 2017. https://mascherari.press/onionscan-report-forensic-finances-dark-markets/.

[30] H. B. Mann and D. R. Whitney. On a test of whether one of two random variables is stochastically larger than the other. *Ann. Math. Stat.*, 18(1):50–60, 1947.

[31] J. Martin and N. Christin. Ethics in cryptomarket research. *Int. J. Drug Pol.*, 25:84–91, 2016.

[32] D. McCoy, A. Pitsillidis, J. Grant, N. Weaver, C. Kreibich, B. Krebs, G. Voelker, S. Savage, and K. Levchenko. PharmaLeaks: Understanding the business of online pharmaceutical affiliate programs. In *Proc. USENIX Sec. Symp.*, August 2012.

[33] A. Noroozian, J. Koenders, E. van Veldhuizen, C. Ganan, S. Alrwais, D. McCoy, and M. van Eeten. Platforms in everything: Analyzing ground-truth data on the anatomy and economics of bullet-proof hosting. In *Proc. USENIX Sec. Symp.*, August 2019.

[34] D. Ogle. fishR Vignette – Closed mark-recapture abundance estimates, December 2013. http://derekogle.com/fishR/examples/oldFishRVignettes/MRClosed.pdf.

[35] K. Pearson. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *London, Edinburgh, and Dublin Phil. Mag. J. Sci.*, 50(302):157–175, July 1900.

[36] R. Portnoff, S. Afroz, G. Durrett, J. Kummerfeld, T. Berg-Kirkpatrick, D. McCoy, K. Levchenko, and V. Paxson. Tools for automated analysis of cybercriminal markets. In *Proc. Web. Conf.*, pages 657–666, 2017.

[37] Q. Rossy, L. Staehli, D. Rhumorbarbe, P. Esseiva, and F. Zobel. Drogues sur Internet: État des lieux sur la situation en Suisse. Technical Report 98, Addiction Suisse & ESC/UNIL, November 2018.

[38] Z. Schnabel. The estimation of the total fish population of a lake. *Am. Math. Month.*, 45(6):348–352, 1938.

[39] C. Schwarz and A. Arnason. A general methodology for the analysis of capture-recapture experiments in open populations. *Biometrics*, 52(3):860–873, September 1996.

[40] C. Schwarz and A. Arnason. Jolly-seber models in MARK. In *MARK: A Gentle Introduction*. 8th edition, 2009.

[41] K. Soska and N. Christin. Measuring the longitudinal evolution of the online anonymous marketplace ecosystem. In *Proc. USENIX Sec. Symp.*, pages 33–48, August 2015.

[42] J. Stevens. *Applied multivariate statistics for the social sciences*. Routledge, 2012.

[43] W. Sutherland. *Ecological census techniques: a handbook*. Cambridge University Press, 2006.

[44] X. Tai, K. Soska, and N. Christin. Adversarial matching of dark net market vendor accounts. In *Proc. ACM KDD*, pages 1871–1880, July 2019.

[45] K. Turk, S. Pastrana, and B. Collier. A tight scrape: methodological approaches to cybercrime research data collection in adversarial environments. In *Proc. IEEE Euro. S&P. Workshops*, pages 428–437, 2020.

[46] X. Wang, P. Peng, C. Wang, and G. Wang. You are your photographs: Detecting multiple identities of vendors in the darknet marketplaces. In *Proc. ASIACCS*, pages 431–442, June 2018.

[47] R. van Wegberg, S. Tajalizadehkhoob, K. Soska, U. Akyazi, C. Ganan, B. Klievink, N. Christin, and M. van Eeten. Plug and prey? Measuring the commoditization of cybercrime via online anonymous markets. In *Proc. USENIX Sec. Symp.*, pages 1009–1026, August 2018.

[48] R. van Wegberg and T. Verburgh. Lost in the Dream? Measuring the effects of Operation Bayonet on vendors migrating to Dream Market. In *Proc. Evolution of the Darknet Workshop at Web. Sci. Conf.*, pages 1–5, 2018.

[49] G. White and K. Burnham. Program MARK: survival estimation from populations of marked animals. *Bird Study*, 46(sup1):S120–S139, 1999.

[50] C. Zhang, R. Wei, and X. Liu. Drugs and bitcoins: What role do bitcoins play in the darknet market? A preliminary study. *Proc. ASIS&T*, 55:944–945, January 2018.

[51] Y. Zhang, Q. Xiong, Y. Fan, W. Song, S. Hou, Y. Ye, X. Li, L. Zhao, C. Shi, and J. Wang. Your style your identity: Leveraging writing and photography styles for drug trafficker identification in darknet markets over attributed heterogeneous information network. In *Proc. Web. Conf*, pages 3448–3454, May 2019.

11 Appendix

11.1 Bias analysis

	Tests			Scraped reviews $n = 139,271$				Not scraped reviews $n = 118,913$			
Variable	Test	Statistic	p-value	M	μ	σ	min-max	M	μ	σ	min-max
listingScraped	χ^2 test	137,129.00	0.00	1.00	1.00	0.00	1–1	0.00	0.32	0.48	0–1
vendorScraped	χ^2 test	24,324.84	0.00	1.00	1.00	0.04	0–1	1.00	0.83	0.37	0–1
isEdited	χ^2 test	96.13	0.00	0.00	0.02	0.14	0–1	0.00	0.02	0.12	0–1
isPurged	χ^2 test	2,642.36	0.00	0.00	0.01	0.09	0–1	0.00	0.04	0.19	0–1

Table 5: Results of the Mann-Whitney U and χ^2 tests between scraped and not scraped reviews

	Tests			Scraped vendors $n = 1,929$				Not scraped vendors $n = 1,696$			
Variable	Test	Statistic	p-value	M	μ	σ	min-max	M	μ	σ	min-max
hasListing	χ^2 test	1,277.90	0.00	1.00	1.00	0.02	0–1	0.00	0.49	0.50	0–1
listingScraped	χ^2 test	3,517.78	0.00	1.00	0.99	0.11	0–1	0.00	0.00	0.05	0–1

Table 6: Results of the χ^2 test between scraped and not scraped vendors

11.2 Exploratory Factor Analysis

We begin by constructing a $n \times k$ data matrix, with n corresponding to the number of listings ($n = 123,133$) and k to the number of features ($k = 9$) for each listing. Since our variables are a mix of numeric and binary types, we calculate *polychronic* and *Pearson* correlations between our variables

from the $n \times k$ data matrix and use the resulting $k \times k$ heterogeneous correlation matrix as input for our exploratory factor analysis. We tested the suitability of our data for factor analysis by performing the KMO and Bartlett's tests. The results in Table 7 show already that there is a very low degree of information overlap among the variables.

Table 7: Results of the KMO and Bartlett's tests

Test	**Test statistic**	***p*-value**
KMO	0.546	
Bartlett	210072.289	0.0

Factor analysis generates a set of i latent factors, each labeled as MR_i, from our correlation matrix. We first use scree-plot analysis [14] and Horn's parallel analysis [25] to determine a suitable i, the number of latent factors to look for ($i = 4$ in our case). Given the $k \times k$ correlation matrix, we then look for three underlying latent factors using a so-called "*minres*" factor analysis method. Moreover, we also apply a so-called "*oblimin*" rotation to the resulting set of factors since we expect the resulting factors to be correlated.

Table 8: Factor Analysis Output

Variable	MR_1	MR_2	MR_3	MR_4
`numReviews`	**1.00**	0.00	0.03	-0.00
`numOrders`	**0.89**	-0.01	-0.03	0.01
`ageListing`	-0.00	**0.88**	0.00	-0.00
`isDeleted`	0.00	0.00	**0.50**	0.01
`category`	0.00	-0.01	-0.01	0.39
`views`	0.28	0.17	-0.10	-0.01
`isHidden`	0.01	0.10	0.14	0.20
`soldNoReview`	-0.04	-0.05	-0.17	0.07
`usdPrice`	-0.01	-0.01	0.00	-0.01
SS Loadings	2.06	1.25	1.07	1.00
Proportion var. explained	0.21	0.09	0.03	0.02
Cumulative var. explained	0.21	0.30	0.33	0.35

The resulting four factors, their so-called "*loadings*," in addition to several other quantities of interest in factor analysis are illustrated in Table 8. Factor loadings in Table 8 (the values reported under each MR_i column), express how much a factor can explain a corresponding variable as a number ranging from -1 to 1. Crudely put, a loading expresses association strength between the latent factor and the original variable. A loading value close to 1 or -1 indicates that a factor "loads" highly onto a variable – i.e., is strongly associated with and explains the observed variance of that variable, while a value close to 0 expresses weak association. For each factor we apply a cut-off point value of 0.4 to its set of loadings, a common threshold used in the literature, to determine the most prominent associations [42]. These are reported in **bold** font, and indicate variables strongly associated with latent factors.

In general, the four latent factors (or three, if we exclude MR_4 based on no variable surpassing the loading threshold of 0.4) only capture 0.35% of the variance. Here, we also observe that of our nine variables only two seem to be associated with the same underlying latent factor, namely `numReviews` and `numOrders`. However, we reason that this is an artifact of the market policy that forcibly associates reviews with actual orders. Thus, for our analysis of testing whether any significant differences exist between scraped and not-scraped listings, we include all variables individually.

11.3 Abundance Estimation Algorithms

We summarize here the three abundance estimation algorithms we employ.

Lincoln-Petersen (LP) The Lincoln-Petersen method estimates N, the population, as

$$\hat{N} = \frac{Kn}{k} , \tag{4}$$

where n is the number of units marked on the first sampling, K is the number of units marked in the second sampling, and k the number of recaptured units that were marked [43].

Schnabel The Schnabel method extends the LP method for situations where we have various samples:

$$\hat{N} = \frac{\sum_t (C_t M_t)}{\sum_t R_t + 1} , \tag{5}$$

where C_t are the total number of units caught at time t, R_t are the number of units already marked at time t, and M_t is the number of marked units at time $t-1$ [43]. Both the Schnabel and LP methods, however, assume that the populations are *closed*, that is, no units appear (births) nor disappear (deaths). To relax these assumptions, "open-population" models which model recruitment and survival were introduced. In this paper, we use the Jolly-Seber (JS) estimator [40].

We used the POPAN formulation [40]. We estimate $\hat{p}_t$ the probability of capture, $\hat{\phi}_t$ the probability of survival between periods, and $\hat{b}_t$ the probability of entering the population. These parameters are estimated using a Maximum Likelihood Estimation (MLE) procedure on a multinomial distribution, where each *encounter history* is a possible outcome. An encounter history is a series of observations of the studied object, encoded as a string of 0s for sampling dates when the object was not observed and 1s when it was observed. The total population N is estimated at each time t by:

$$\hat{N}_t = \hat{N}_{t-1}\hat{\phi}_{t-1} + B_{t-1} , \tag{6}$$

where B_t is the number of new entrants to the population.

12 Scrape Dates

We obtained scrapes from the Hansa marketplace taken on: October 8th 2015, October 11th 2015, October 16th 2015, October 23th 2015, October 25th 2015, November 2nd 2015, December 1st 2015, December 13th 2015, January 7th 2016, January 17th 2016, April 30th 2016, June 8th 2016, July 7th 2017 and July 14th 2017.