On Taxis and Rainbows

Lessons from NYC's improperly anonymized taxi logs

Vijay Pandurangan

Recently, thanks to a Freedom of Information request, Chris Whong received and made public a complete dump of historical trip and fare logs from NYC taxis. It's pretty incredible: there are over 20GB of uncompressed data comprising more than **173 million** individual trips. Each trip record includes the pickup and dropoff location and time, anonymized hack licence number and medallion number (i.e. the taxi's unique id number, 3F38, in my photo above), and other metadata.

These data are a veritable trove for people who love cities, transit, and data visualization. But there's a big problem: the personally identifiable information (the driver's licence number and taxi number) hasn't been anonymized properly — what's worse, it's trivial to undo, and with other publicly available data, one can even figure out which person drove each trip. In the rest of this post, I'll describe the structure of the data, what the person/people who released the data did wrong, how easy it is to deanonymize, and the lessons other agencies should learn from this. (And yes, I'll also explain how rainbows fit in).

The NYC taxi data consist of a number of CSV-files, each with lines that look like this:

```
6B111958A39B24140C973B262EA9FEA5,D3B035A03C8A34DA17488129DA581EE
```

Each of these columns correspond to the following fields:

```
medallion,hack_license,vendor_id,rate_code,store_and_fwd_flag,pi
```

It's pretty obvious what format most of the fields are in (lon/lat, timestamp) but the first two columns demonstrate that the government folks clearly intended to anonymize the medallion and licence numbers. It's obvious that the alphanumeric codes are not purely random, the same taxi and same driver always has the same code throughout the data. This isn't necessarily a problem — in fact, preserving this property is often of critical importance when anonymizing data. Imagine a search engine engineer who wants to analyse user behaviour in aggregate: it's important that the same user has the same code throughout the set of logs that are being analysed, so that you can see what a user does over time. It's obviously also critical that it's not possible to go backwards from the code to the user's name.

Someone on Reddit pointed out that one specific driver seemed to be doing an incredible amount of business. When faced with anomalous data like that, it's good practice to weed out data error before jumping to conclusions about cheating taxi drivers. Also, I couldn't shake the feeling that there was something about that encoded id number: "CFCD208495D565EF66E7DFF9F98764DA." After a little bit of poking around, I realised that that code is actually the MD5 hash of the character '0'. This proved my suspicion that this was actually a data collection error, but also made me immediately realise that the entire anonymization process was flawed and could easily be reversed.

A cryptographically secure hashing function, like MD5 is a one-way function: it always turns the same input to the same output, but given the output, it's pretty hard to figure out what the input was *as long as you don't know anything about what the input might look like*. This is mostly what you'd like out of an anonymization function. The problem, however, is that in this case we know a *lot* about what the inputs look like.

In NYC, taxi licence numbers are 6-digit *[Edit: (6-digit numbers may start with any digit and are sometimes zero-padded)]*, or 7-digit numbers starting with a 5. That means that there are only about 3M possible taxi licence

numbers. Similarly, medallion numbers conform to a very specific pattern:

```
one number, one letter, two numbers. For example: 5X55
two letters, three numbers. For example: XX555
three letters, three numbers. For example: XXX555
```

[Edit: I fixed my math thanks (thanks Paul)] There are about 1000*27²*26 = 18954000 or ~19M possible medallion numbers. So, by calculating the md5 hashes of all these numbers (only 22M!), one can completely deanonymise the entire data. Modern computers are fast: so fast that computing the 22M hashes took less than 2 minutes. The resulting table of hashed-data to input data is called a ***Rainbow Table.*** *[Edit: people have pointed out that this may not meet the exact requirements to be called a rainbow table]*

It took a while longer to de-anonymize the entire dataset, but thanks to Yelp's MRJob, I ran a map-reduce over about 10 computers on EMR and had it done within an hour. There's a ton of resources on NYC Taxi and Limousine commission, including a mapping from licence number to driver name, and a way to look up owners of medallions. I haven't linked them here but it's easy to find using a quick Google search.

Here are a few de-anonymized lines I picked at random:

```
9Y99,5296319,VTS,1,,2013-12-06 00:07:00,2013-12-06 00:16:00,5,54
```

Security researchers have been warning for a while that simply using hash functions is an ineffective way to anonymize data. In this case, it's substantially worse because of the structured format of the input data. This anonymization is so poor that anyone could, with less then 2 hours work, figure which driver drove every single trip in this entire dataset. It would be even be easy to calculate drivers' gross income, or infer where they live.

There are a number of ways these data could have been better anonymized.

Two good ones include:

- assigning a totally random number to each hack licence number and medallion number once, and re-using it throughout the dump file
- creating a secret AES key, and encrypting each value individually

[Edit: As always, things are more complicated than they seem at first. The strategies described above will anonymise the numbers of the licences and taxicabs, but commenters have pointed out that there are a number of other ways in whch PII may be reconstructed. The most interesting is Narayanan and Shmatikov's algorithm. NYC is dense enough that it may be much more challenging to target specific passengers using these data, however. Anonymizing data is really hard.]

The cat is already out of the bag in this case, but hopefully in the future, agencies will think carefully about the method they use to anonymize data before releasing it to the public.

Edit: this is up on hackernews. Please feel free to comment there