CS-523 SecretStroll Report

Ugo Damiano , Eric Jollès

***Abstract*—In the last few years, the explosion of smartphones applications has led to an increasing use of location. Although, these new technologies bring a large amount of new possibilities, they also endanger user privacy. Coupled with the inherent problems of the internet and authentication, the use of encryption to protect users data has become obsolete. This paper studies how metadata can be exploited and which measures can be taken to protect the users privacy.**

I. Introduction

The goal of this project is to study the privacy aspects of an app relying on location services. On this application, called SecretStroll, the user subscribes to a set of point of interest (pois), such as restaurants, and can then get the nearest points of interest to his position for which he subscribed.

In a more general setting, most of today applications require the use of an account making the user pseudonymous. Such mechanisms are a major privacy issue and can even lead to full deanonymization if other data such as social networks profiles are provided.

In the first part we propose to use an attribute based credential scheme to cope with this issue.

In the second part we focus more on challenges related to location. We show how privacy can be achieved in settings were the service provider needs access to the user position. We also show how such mechanisms affect utility, enlightening the fact that good privacy can only be achieved by a more or less important trade-off with utility.

Lastly, we focus on a more global aspect of today's communication. By analyzing how the exchange of messages on a network such as the internet are carried out, we show how information can be revealed about their content.

II. Attribute-based credential

To give some points of interest, our application requires proof of an active subscription of a user. To protect user privacy, we used attribute-based credentials (ABCs) with Fiat-Shamir heuristic, since we want to perform offline zero-knowledge proofs. The base protocol is taken from the Pointcheval and Sanders (PS) attribute-based credential scheme[1]. In our implementation the issuer and verifier are run on the same server.

In order to map SecretStroll subscriptions to the attribute based credential we worked as follow:

First, the issuer specifies a list of the different subscriptions he wants to propose. He must create a pair of public and private keys for each of these attributes. Issuance protocol is shown in Figure 2.

In our implementation, the user needs to specify all the values of the different subscriptions, even if he doesn't want them. The full set of attributes corresponds to the list of subscriptions plus the username and secret key of the user. In order to convert the attributes and their value to exponents, we decided to hash the attribute's name concatenated with its corresponding value. Once the user has obtained a credential he can prove its possession by following the protocol of Figure 3.

The proofs of knowledge are performed using the protocol described in Figure 1.

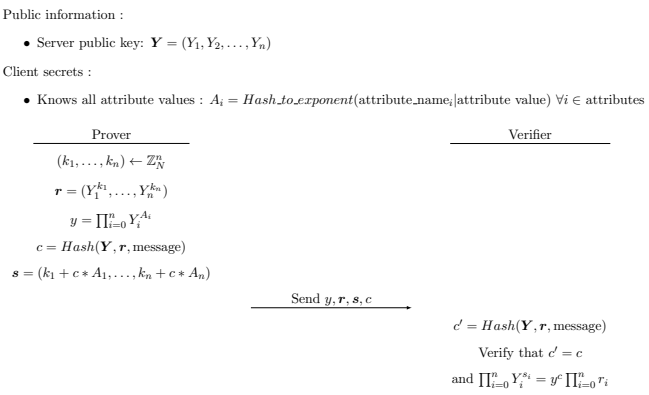

Fig. 1: Pedersen commitment

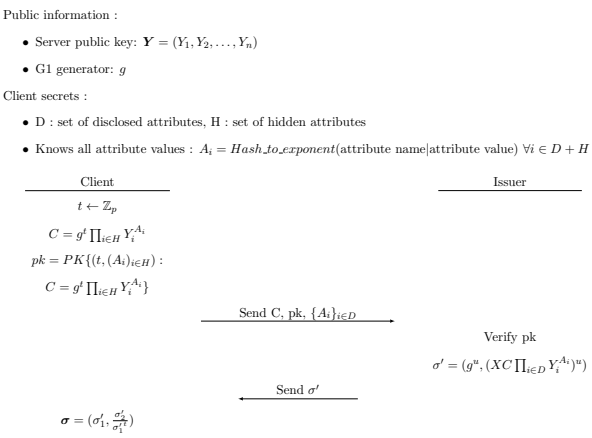

Fig. 2: Issuing a credential. In our scheme H = {user's secret key} and D contains all other attributes.

A. Test

In order to test the system we performed a simulation of the protocol with the following different cases. The client :

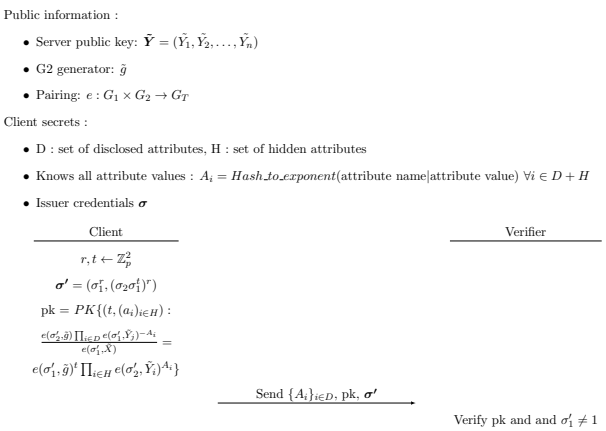

Fig. 3: Proving possession of a credential

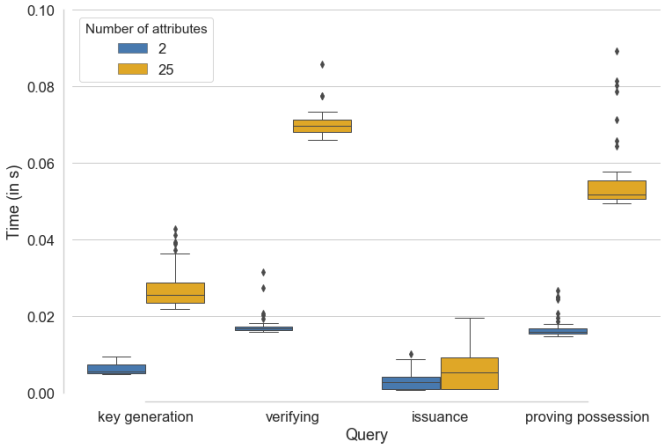

Fig. 4: Computation times of the different components of the credentials for 2 and 25 attributes.

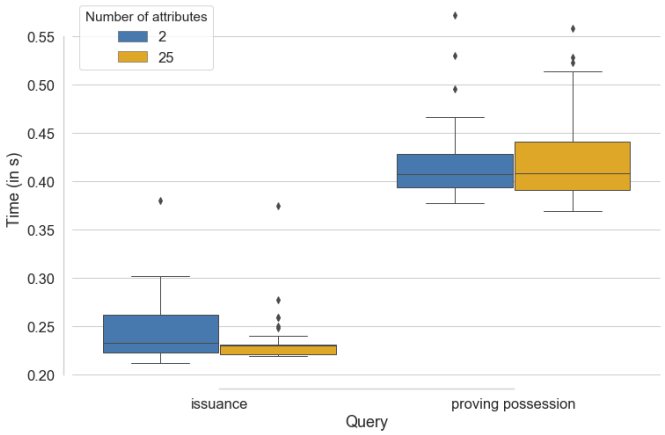

Fig. 5: Communication times of the different components of the credentials for 2 and 25 attributes.

1) subscribes and then asks for one of its subscribed point of interests.
2) reveals all its attributes, secret key included.
3) changes one of its attributes value and tries to produce a valid signature.
4) sends to the server a message m and a signature on a different message m'.

We also checked if the server correctly aborts the exchange when the client tries to issue or prove invalid attributes. These tests check the basic functionalities of the system and that the server reacts correctly when passed incorrect input. They also check that the server will not accept random message, signature pairs or proofs he receives.

B. Evaluation

As shown in Figures 4 and 5, the slowest operation is the verification phase. As opposed to the other phases, which are performed at the start of the system, the verification is performed over and over again, which could be an issue for a frequent user. However, the runtime of this phase is still less than 1 second which still makes it usable. Although, the computation times scales with the number of attributes, it only varies by a few hundredths of seconds showing that even with a realistic number of subscriptions the scheme is still usable.

We also noticed in Figure 5 that the communication time does not depend on the number of attributes and that the proving possession of attributes part takes more time since we don't send only the proof but also the message.

III. (De)Anonymization of User Trajectories

A. Privacy Evaluation

1) Adversary Model: In this part, the attacker should be able to see the content of the messages. The attacker should be able to access the server or to eavesdrop between the client and the server if the data is sent in the clear. The attack we will present can be mounted if we are provided with a dataset containing queries with locations and timestamps. Also, even though the assumption that users have a fixed IP address might sound simplistic it might actually apply to real life examples. In fact, depending on the service provider, the IP address is assigned only once when signing into the network with a phone. If the user never signs out the address might stay fixed for a long time thus, allowing tracking.

Indeed, in this part, each client is pseudonymized with its IP address. Moreover, for the moment, the server receives in each query precise client location. Thus, by analyzing the queries the service provider can recover location patterns. As an example, where the client lives or where he works. Home and work location, when coupled together, can be used for unique identification [2]. The privacy issue is that we probably can identify one person by his home and working place. Thus, we will try to hide this information from the server.

2) Attack Demonstration: We tried to find a way to infer the living and working place for each client. Since there is no ground truth to ascertain the truthfulness of our predictions, we will, in a first time, try 2 different analyses to then cross-check the results.

The first analysis is based on the number of different days

on which a query was conducted from a specific position. We can observe in Figure 6 that there are 2 distinct peaks with the same height (200, which is the number of users). Each user has one element in each of these peaks. As we have 20 days starting on a Monday we have overall 15 weekdays where people are assumed to go to work. Thus home locations are to appear in the right peak and working places are to appear in the left peak.

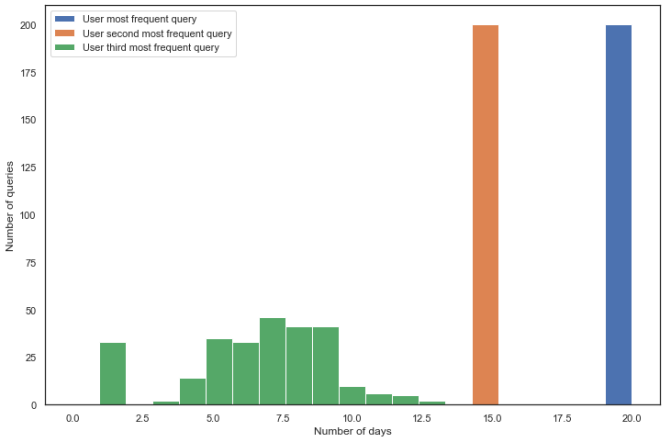

Fig. 6: Distribution of number of distinct days when a query with specific location was made

The second analysis will focus on the most frequent position used for working days queries. We will split them according to 2 distinct time intervals :

1) morning/early afternoon (from 6 am to 3 pm) – office hours
2) evening (from 4 pm to 11 pm) – home hours

This is shown in Figure 7.

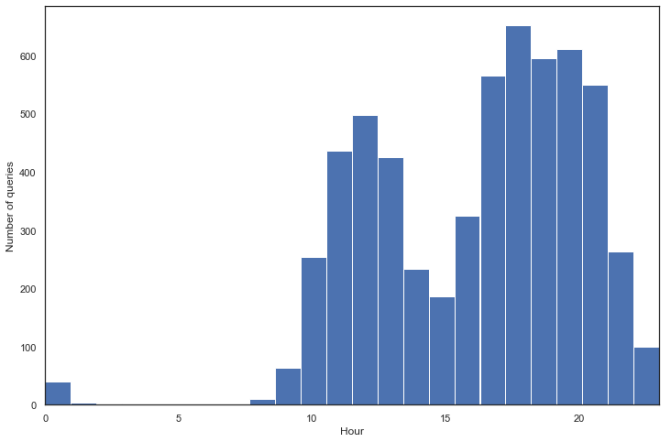

Fig. 7: Query distribution per hour

We stated that the most used location during Interval 1 corresponds to the office location, and the most used in Interval 2 corresponds to the home location.

When we compared the 2 results, we figured out that all home locations were the same according to the 2 analyses. However, 6 locations over 200 that we classified as work places were different.

In these cases, we noticed that the predicted office location in first analysis corresponds to the second most frequent query in interval 2 in analysis 2. This can be linked to the fact that some workers can be at home during the morning.

To avoid these cases, we would use the first analysis if we had to mount an attack.

B. Defences

Our first idea to protect the user privacy is to send the cell id in a query in place of location. The user will receive the same query answer since the points of interest sent in return are linked to the cell id. However, we noticed that in the given files, the pair working cell id, home cell id is a quasi identifier (84% of the users have a unique pair). Also, the privacy of the user depends on the size of the cells, which characterized by the server. We thus need to find an additional defence mechanism.

We thus decided to use spatial obfuscation, namely perturbing locations with noise. In this case we do not have a lot of points per user thus, we added Laplacian noise at the client side to each point and keep the obtained noised point for future queries to avoid point averaging. In order to compare how well this mechanism works, we try to infer the same information as before, work place and house. We compare our new estimations and the previous ones with L^2 norm[1]. We define the utility as the average L^2 norm between a person location and the query response points. When we make a query, the server responds with points of interest of the actual cell. However, when we add noise to positions, the cell can change and thus the points of interest. It can happen that a query has no answer in one cell, but has several answers in another. When we compute the utility of our defence, we do not take into account these queries.

Threat model: For this defence to work, we assume that the adversary has only access to the location with timestamps of the user and not any other data. In fact, only seeing the packet metadata might be enough to get information about its content. So, it might be possible for the attacker to get this information by eavesdropping the traffic entering the server. We will demonstrate this in part IV. We also assume that the location service provider is trustworthy.

Utility and privacy trade-off: The more privacy we have from noise, the less useful we are. With a Laplace noise with scale 0.002 and 0 mean[2], 55% of the packets stay on the same cell. As can be seen in Figure8 points stay close. However with a Laplace noise with scale 0.04 and 0 mean, only 3% of the packets stay on the same cell, as it can be noted on Figure 9. The utility is thus impacted by the Laplace scale, as shown in Figures 10 and 11, respectively. Note that, when either the

[1]The norm is computed with latitude and longitude. So the result is in degrees.

[2]Whenever we talk about the noise we mean by that a two-dimensional Laplacian distributed noise where each dimension is independently drawn from the mentioned distribution.

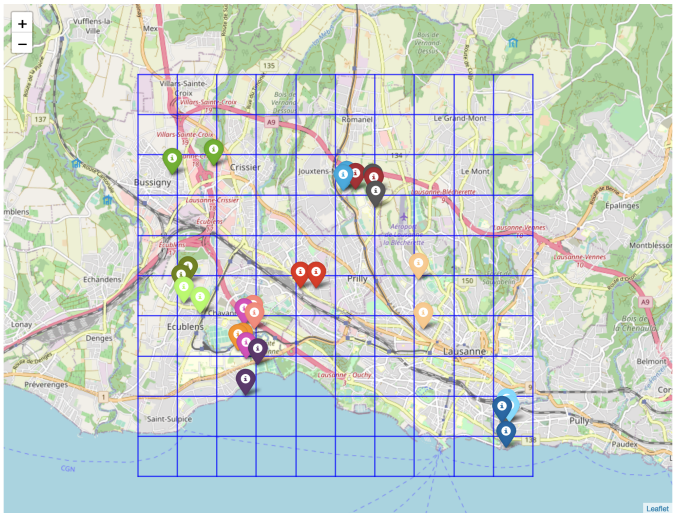

Fig. 8: Original query point with noised point (both points have the same color) with a Laplace noise with scale 0.002 and 0 mean added to the location for the user with IP address 34.101.177.245

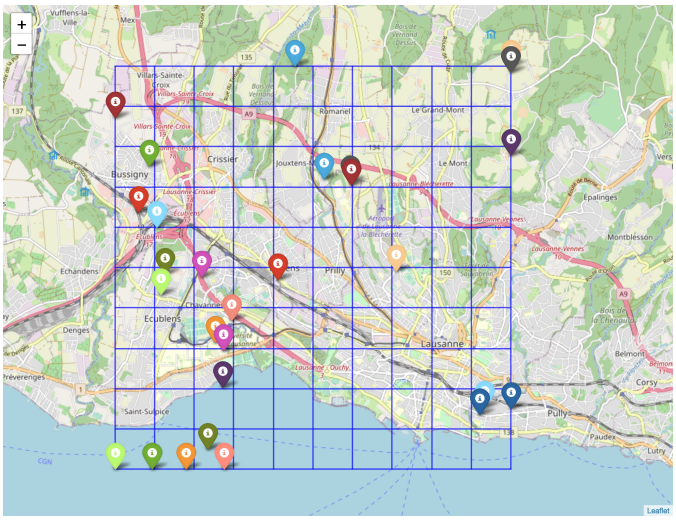

Fig. 9: Original query point with noised point (both points have the same color) with a Laplace noise with scale 0.04 and 0 mean added to the location for the user with IP address 34.101.177.245

latitude or the longitude of the point is out of the grid, the point is set to the nearest location in the grid.

Thus, the limit of our technique is that we need to give to the server a cell id. Indeed, when the noise is too big, we give most of the time a corner cell (cell 1, 10, 91 and 100) which causes the utility of the service to drop.

IV. Cell Fingerprinting via Network Traffic Analysis

A. Implementation details

For each cell we collected 100 different packet sequences. For that we used tcpdump at the client side. Once, tcpdump

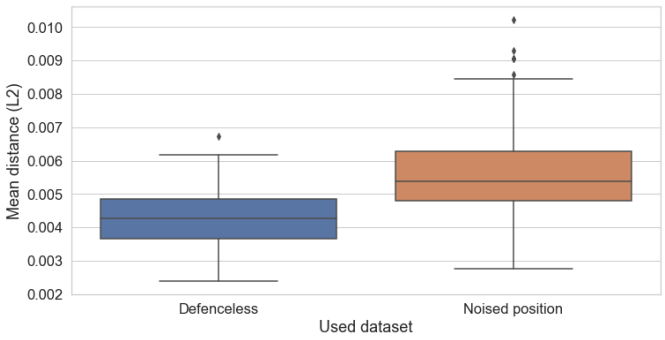

Fig. 10: Mean distance to points of interest for each query with a Laplace noise with scale 0.002 and 0 mean

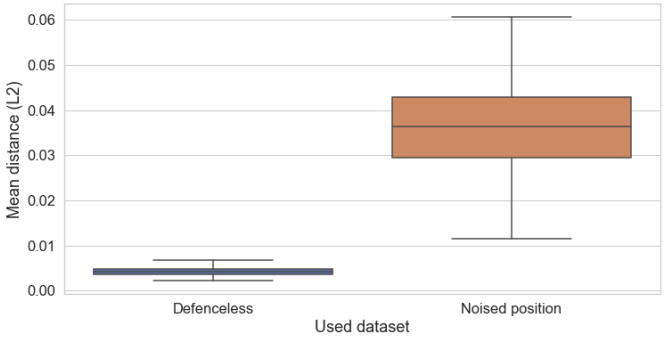

Fig. 11: Mean distance to points of interest for each query with a Laplace with scale 0.04 and 0 mean

is launched, a query for a cell is made. When the query is over, a pcap file is generated based on this communication. In order to get all the packets related to the tor communications we only keep the ones containing ports 443 (HTTPS) or in between 8'000 and 10'000 (tor related ports) [3]. Then, the size of the packets and their timing is kept and written into arrays. Here, we present the different methods we tried and results they achieved.

V. Evaluation

1) Wang et al attack: We decided to implement a state of the art Website Fingerprinting knn attack : Wang's et al attack [3]. The feature set is composed of data extracted from the packet sequences such as transmission size, number of incoming/outgoing packets, packets ordering etc. There is at the end more than 3000 extracted features. The corresponding machine learning model used is the well known k-Nearest Neighbor classifier with weight adjustment. The accuracy obtained with Wang attack produces an accuracy less than 0.2 over the train set whereas with real traffic this attack has accuracy around 0.9

2) k-Nearest Neighbors: We train a k-Nearest Neighbors with two features. The number of packets in a sequence and the sum of all the packets sizes in a sequence. The cell id

[3]Tor generaly uses ports in the 9000 range but it appears that in our machines ports in the 8000 range were used for this purpose.

of the query is used to label each of these sequences. The model results in a 10-fold cross validation accuracy of mean 0.5112 and variance 0.0056. This result is achieved after some polynomial expansion and hyper-parameter tuning.

3) Support Vector Machine (SVM): The third model we used is a SVM. The features set we used is composed of the number of packets in a sequence, the sum of all the packets sizes in a sequence and the total time of packet sequences. We made grid search evaluation in order to find best hyper-parameters. The model results in a 10-fold cross validation accuracy of mean 0.61 and variance 0.0002.

A. Discussion

We tried to use features such as intervals between packets, number of outgoing and incoming packets, which ports are used, ratios of incoming and outgoing packets and statistics over packet ordering list. But this didn't improve the classifier much. Why is it the case? In website fingerprinting, packets sizes and streams, namely incoming and outgoing packets, have characteristics quasi-unique for each website. However, here, our exchanges always belong to the same client-server pair and follow the same protocol, one outgoing query and then incoming points of interest. We still have different answers according to the query varying packet sizes and number of packets. Also, the fact that the delay of the packets between the client and the server is always the same, makes it harder to characterize one request. We must account for the very high number of cells which makes queries highly likely to look like a query for another cell. Clearly, this precision might increase if provided with a fair amount of data and a better filtering phase.

On the other hand, as noted in the previous part, the user performs generally the queries from the same location. So, even if the model isn't always right, by observing multiple queries of a user, one prediction might stand out. Even if the classifier does not have a really high performance, given enough query data about a user we might find something interesting. Finally, note that when trying to infer this information from a normal user, the noise added by other communications might decrease the precision of the results.

B. Countermeasures

One countermeasure could be to send the same number of packets for every query. However, it might result in high overhead and the bound is hard to find. In fact, the amount of points of interest in a cell is potentially infinite. Clearly, we should find a bound that reflects the possibility of adding new points without having to change it.

Another approach would be to make all the packets the same size, which is doable with services such as TCP.

Another way to protect the queries could be to load another random cell request in the background together with the real one. This technique will decrease the performance of the attack, but it will also increase the time of each request, which could affect the user experience. On the long run, if the same cell is asked many times it might get revealed. Finally, note that we could also cache the result if the points of interest don't change often.

VI. Conclusion

As stated in the introduction, we have seen that building a modern privacy preserving app brings quite a few challenges. At some reasonable cost, we can add anonymous credentials which allows users to use services without having to reveal unnecessary information such as age or gender. At a less reasonable cost, we can make use of location services by providing the users with means to hide their true location thus, avoiding tracking. However, we still have to make compromises. Location services still need to produce a location based on the GPS information thus, they need to be trustworthy which might not always be the case. Also, the internet still keeps some archaic principles dating back to the 1970s which makes it a perfect target for modern algorithms such as machine learning. Although encryption and its problems are widespread, the simplistic model of a message on a link cannot be simply applied to the internet. Both users and providers must make use of these solutions in order to live in a world where data has become the new gold.

References

[1] D. Pointcheval and O. Sanders, "Short randomizable signatures," 02 2016, pp. 111–126.

[2] P. Golle and K. Partridge, "On the anonymity of home/work location pairs," in *Pervasive Computing*, H. Tokuda, M. Beigl, A. Friday, A. J. B. Brush, and Y. Tobe, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 390–397.

[3] T. Wang and I. Goldberg, "Walkie-talkie: An efficient defense against passive website fingerprinting attacks," in *26th USENIX Security Symposium (USENIX Security 17)*. Vancouver, BC: USENIX Association, Aug. 2017, pp. 1375–1390. [Online]. Available: https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/wang-tao