June 2023

Synthetized Homotopy Based Privacy for Device Analytics

n/a

Synthetized Homotopy Based Privacy for Device Analytics

ABSTRACT

Data analytics on mobile devices is important for improving user experience and developing better products and services. Device analytics need to be performed in compliance with regulations. A privacy compliance utility needs to establish univalence before and after privacy compliance to generate reliable and accurate data for device analytics. This disclosure provides a specification for a privacy utility based on synthetized homotopy that can be utilized by services that perform device analytics. The techniques described in this disclosure can also be used to enable privacy compliant content sharing with nearby devices. The techniques described in this disclosure can be utilized for privacy compliant device analytics without relying on the classic set theory and logic assumptions on axiom of choice (AC) and the law of excluded middle (LEM).

KEYWORDS

- Device analytics
- Device privacy
- Synthetized homotopy
- Pseudonymization

BACKGROUND

Data analytics on mobile devices is important for improving user experience and developing better products and services. Device analytics need to be performed in compliance with regulations such as the European Union's General Data Protection Regulation (GDPR) [1]. For very large datasets (e.g., device populations running into hundreds of millions or billions), privacy compliance requires a model without classic set theory and logic assumptions on axiom

of choice and the law of excluded middle.

A privacy compliance utility needs to establish univalence (equivalent mathematical structures being indistinguishable) before and after privacy compliance to generate reliable and accurate data for device analytics. Partitions between private and non-private data can be dynamic and complex. Privacy compliance requires a rule to compute the target non-private data and a stable privacy function to generate private data. Additionally, a validation and recovery method can ensure that the output private data generated is accurate. Privacy compliance also requires that the original files in the private data are permanently removed.

DESCRIPTION

This disclosure provides a specification for a privacy utility based on synthetized homotopy that can be utilized by services that perform device analytics. The techniques described herein can also be used to allow privacy compliant content sharing, e.g., with nearby devices.

Pseudonymization is a technique used to protect data privacy by replacing identifiable information with a pseudonym or code. The techniques described herein involve synthetizing the foundation of pseudonymization and the deduction of its consequences. The techniques also involve the derivation of a rule to calculate target log directories and files, pseudonymization of private data (e.g., device ID), validation and recovery method, and output pseudonymization to permanently remove original data files. For stability, a permanent service key is specified for identifier pseudonymization.

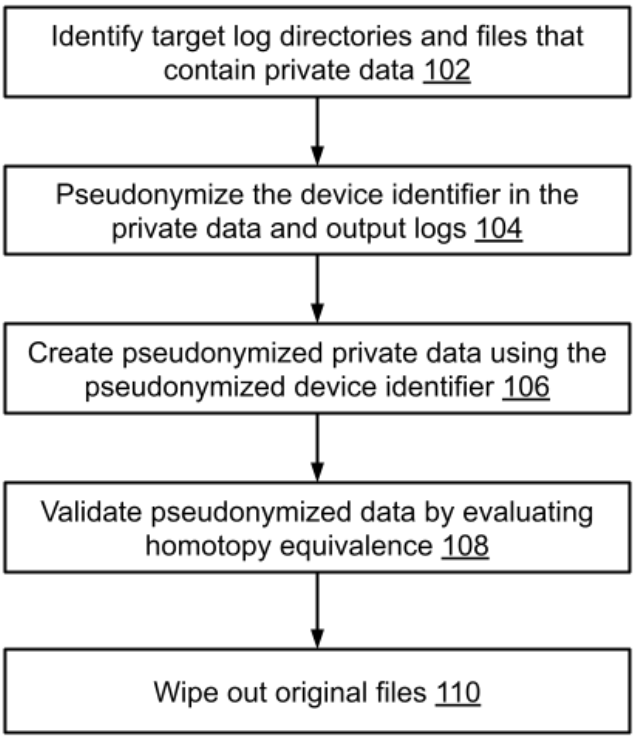

Fig. 1: Example method for pseudonymization of private data

Fig. 1 illustrates an example method for pseudonymization of private data, per techniques described in this disclosure. The target log files and directories that contain private data are identified (102). The device identifier in the private data and output logs is pseudonymized (104). A permanent service key is specified for identifier pseudonymization to ensure stability. Pseudonymized private data is calculated using the pseudonymized device identifier (106). The pseudonymized data is validated by checking univalence between the original private data and the pseudonymized data through homotopy equivalence (108). The original data files are wiped out (permanently deleted) once the pseudonymized data has been successfully validated (110).

Different types of models can be used for pseudonymization. These are described below.

Full Pseudonymization

This model reads all files in the directories which expired after a certain period for

pseudonymization within the private data. All output files are generated and original files are wiped out.

The pseudonymization interval, denoted as I, is the simplest higher inductive type. It is generated by:

a point 0 of I : I,

a point 1 of I : I,

a path segment:0 of I = 1 of I.

The recursion principle for the interval says that given a type B along with

a point b0 : B,

a point b1 : B,

a path s:b0 = b1,

Pseudonymization function f : I → B such that f(0I) ≡ b0, f(1I) ≡ b1, and f(segment) = s.

The induction principle says that given P : I → U along with

a point b0 :P(0 of I),

a point b1 :P(1 of I),

a path s: b0 = b 1 for P on segment

There is a function f : $\prod_{(x:I)} P(x)$ such that $f(0_I) \equiv b_0$, $f(1_I) \equiv b_1$, and $apd_f(seg) = s$.

The full pseudonymization function constructs the higher inductive interval type for the private data. Output files are generated and the original files are wiped out.

Block Pseudonymization

This model can be used for pseudonymization of a specific list of date filters. This model can also be used to force pseudonymization for a list of specific dates, e.g., that may have earlier been missed or in case of erroneous pseudonymization. The list of dates can be used to match

with target directories and files for pseudonymization. Output files are generated and the original files are wiped out.

Block pseudonymization function constructs the higher inductive W-type. W-types are a generalization of natural numbers, lists, and binary trees, which are sufficiently general to encapsulate the recursion aspect of any inductive type. Block pseudonymization can define the type of lists over A as a W-type with 1 + A many labels: one nullary label for the empty list, plus one unary label for each a : A, corresponding to appending *a* to the head of a list: List(A) :≡ $W_{(x:1+A)}$ $rec_{1+A}(U, 0, \lambda a.\ 1, x)$.

Block pseudonymization constructs finite types and has no consistent form of choice to pick a date. Block pseudonymization does propositional truncation in the domain or co-domain such that the values are not chosen or specified in any known way. Block pseudonymization does propositional truncation in co-domain without determining in any known way. As a result, the validation function for block pseudonymization does not require the traditional axiom of choice (LEM) for pseudonymization.

Incremental Pseudonymization

Incremental pseudonymization is encoded with dates to be triggered at periodic intervals, e.g., daily. The single date is used to match with the target directory and files for pseudonymization. Output files are generated and original files wiped out. The pseudonymization pipeline is scheduled to run periodically such that incremental pseudonymization is the regular operation mode. Pseudonymization performed through regular operation of incremental pseudonymization is equivalent to full pseudonymization.

Pseudonymization requires the rules that can be used to process any infinite nearby logs.

The simplest infinite type is the type N : U of natural numbers. Incremental pseudonymization constructs the homotopy type of natural numbers. The elements of N are constructed using 0 : N and the successor operation succ : N → N. The pseudonymization functions can be defined by recursion and proved by induction.

Given a starting point c_0: C and a next step function c_s : N → C → C, pseudonymization pipeline constructs a non-dependent function f : N → C out of the natural numbers by recursion, Then f can be defined by the following primitive recursion:

$$f(0) :\equiv c0,$$

$$f\,(succ(n)) :\equiv cs\,(n, f\,(n)).$$

The above function, definable only using the primitive recursion principle, is computable and constructive. The encode-decode method is used to characterize the path space of the natural numbers as dates, which are also a positive type. In this case, rather than fixing one endpoint, the two-sided path space is characterized all at once. Thus, the codes for identities are a type family code : N → N → U. A dependent function *r* is also defined by recursion as described below:

$$r : \prod(n:N)\ code(n, n), \text{ with}$$

$$r(0) :\equiv \star$$

$$r(succ(n)) :\equiv r(n).$$

The models described in this disclosure represent pseudonymization by types, which can be regarded simultaneously as both constructions and assertions (propositions as types). Pseudonymization regards a term a : A as both an element of the type A, and at the same time, a proof of the proposition A.

Pseudonymization validation and recovery method

The validation method is triggered upon the completion of pseudonymization and is used to evaluate equivalence between the original private data and the pseudonymized data by means of homotopy equivalence. Validation recovers a missed entry by joining the collection of processed entries and writing to log directories. Original logs are overwritten upon successful validation and recovery. If a pseudonymization job does not function, it is shut down and results in the termination of pseudonymization. In such cases, block pseudonymization can be used to catch up with the process of original logs. In case a pseudonymization job has read/write errors, pseudonymization errors, the validation and recovery method logs and reports the errors by failure counters but the job proceeds.

The validation method proves equivalence by calculating global algebraic invariants associated with a space. This includes the homotopy groups and homology and cohomology groups. The equivalent spaces must have isomorphic homotopy/(co)homology groups. This means that if two spaces have different groups, then they are not equivalent and validation fails.

Specifically, these algebraic invariants provide global information about the original private data space and the pseudonymized data space. Global information provided by algebraic invariants complements local information provided by notions such as continuity and can be used to differentiate between spaces with original identifiers and pseudonymized identifiers, which are local variants.

The fundamental group of a Nearby space is a simplest global invariant, which is written $\pi 1(X, x0)$: Given a space X and a point x0 in it, a group can be made whose elements are loops at x0 (continuous paths from x0 to x0), considered up to homotopy, with the group operations given by the identity path (standing still), path concatenation, and path reversal. The identity of

each log space is a such loop.

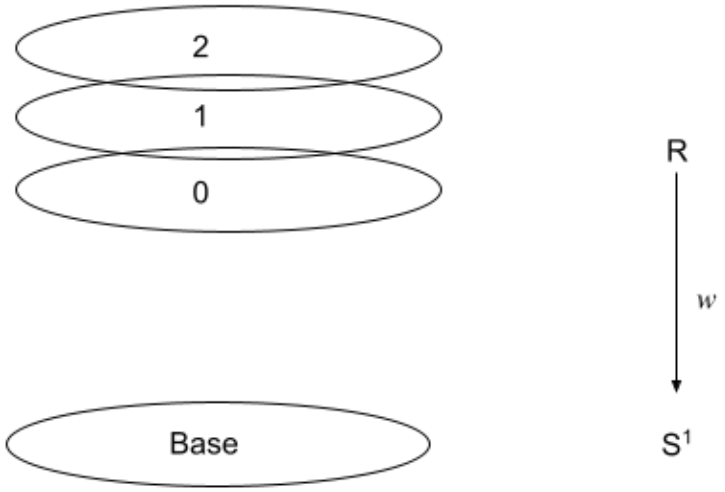

Fig. 2: Loops for calculation of global invariants

Using the techniques described in this disclosure, the validation method evaluates both homotopy and pointwise equivalence of the original private data and the pseudonymized data. The pseudonymized private data maps to a higher inductive interval compared to the original data. This is important since it enables the recovery and validation process to check univalence between the original private data and pseudonymized data through homotopy equivalence.

There are multiple alternatives to implement the pseudonymization techniques described in this disclosure. The simplest and low-risk option is to use separated pipeline pseudonymization. However, this requires reprocessing and reproduction of logs which adds overhead. Other alternatives include integrated inline pseudonymization and multi-phase pipeline pseudonymization.

Separated pipeline pseudonymization involves the creation of a separate pipeline where full pseudonymization, block pseudonymization, and incremental pseudonymization are implemented. This approach has benefits such as no interruption of existing jobs or dashboards,

flexible triggering of pseudonymization, and the ability to set up and configure full pseudonymization, block pseudonymization, and incremental pseudonymization without dependencies. However, there is an overhead of reprocessing and reproduction of logs.

Integrated inline implementation approach involves incorporating incremental pseudonymization into existing pipelines. A benefit of this approach is that no new pipeline is required to be created or set up. However, there may be a higher risk in rollout and difficulties in setting up experiments. Existing pipelines may be impacted by human errors or bugs, pseudonymization changes, and rollout.

In the multi-phase pipeline pseudonymization implementation, a separated pipeline can be implemented and rolled out as the first phase. After the separated pipeline runs are stabilized, incorporation of inline pseudonymization into existing pipelines can be considered to eliminate the overhead of the separated pipeline approach. Full pseudonymization and block pseudonymization are separated pipelines, while incremental pseudonymization can be integrated into existing pipelines.

The techniques described in this disclosure can ensure privacy compliance without relying on the classic set theory and logic assumptions on axiom of choice (AC) and the law of excluded middle (LEM). The proposed validation method proves both homotopy and pointwise equivalences by calculating counters and re-identification. The described techniques can be used in any device analytics pipeline and/or in other applications.

CONCLUSION

This disclosure provides a specification for a privacy utility based on synthetized homotopy that can be utilized by services that perform device analytics. The techniques described in this disclosure can also be used to enable privacy compliant content sharing with

nearby devices. The techniques described in this disclosure can be utilized for privacy compliant device analytics without relying on the classic set theory and logic assumptions on axiom of choice (AC) and the law of excluded middle (LEM).

REFERENCES

1. "General Data Protection Regulation - Wikipedia" available online at https://en.wikipedia.org/wiki/General_Data_Protection_Regulation