Tit-for-Tat Disclosure of a Binding Sequence of User Analyses in Safe Data Access Centers

Josep Domingo-Ferrer

Universitat Rovira i Virgili,
Department of Computer Engineering and Mathematics,
CYBERCAT-Center for Cybersecurity Research of Catalonia,
UNESCO Chair in Data Privacy,
Av. Països Catalans 26, 43007 Tarragona, Catalonia.
e-mail josep.domingo@urv.cat

Abstract. Safe access centers, a.k.a. trusted research environments, and digital marketplaces allow a user to submit her own statistical analyses on the data stored by controllers. As a responsible organization for the data, the controller may wish to enforce several ethical controls on the user's code. If data are personal or otherwise confidential, the controller needs to make sure the outputs of user analyses do not leak any personal information. Additionally, the data controller may require other ethical properties of the user analyses, like fairness or explainability in case an analysis consists of training a decision model. If a user analysis fails to meet the controller's ethical requirements, the controller stops running the user's code on his data. On the other side, the user would like to achieve fair disclosure of her code, that is, stop disclosing the remainder of her code to the controller in case the controller refuses to return the result of the current analysis. In this concept paper, we present a framework allowing the user: i) to bind herself to a sequence of analyses before receiving any actual output, which precludes disclosure attacks via adaptive analyses; ii) to disclose her next instruction/analysis to the controller only after the controller has returned the output of the current instruction/analysis, which ensures fair disclosure. Our aim is to protect both the controller's and the user's interests.

Keywords: Safe access centers · Digital marketplaces · Trusted research environments · Confidentiality · Output checking · Fairness · Ethics

1 Introduction

There is an increasing demand of data of all sorts for research and decision making. Yet, unrestricted sharing of data is often infeasible, for various reasons. On the one side, personal data fall under the umbrella of data protection legislation, such as the European Union's General Data Protection Regulation [7]. On the other side, there may be data that are confidential for other reasons, such as trade secrets. Anonymization [5, 10] is a possible solution to reconcile the need of data dissemination for secondary uses (different from the primary use

that motivated their collection) with the confidentiality constraints. However, anonymization entails some accuracy loss, which may be unaffordable in some secondary analyses. This creates a demand for accurate original data.

Possible workarounds to allow users to compute their analyses on original data that are confidential is for them either to access a remote access center run by the controller (where the users are under surveillance) or to submit the code of the analyses to the data controller who holds the data. This latter case is called remote execution: the controller runs the code on the user's behalf, and returns her the results if these do not leak any of the original data on which they were computed. Examples of such arrangements include:

- National statistical institutes and data archives often feature so-called safe access centers (a.k.a. research data centers or trusted research environments) as an option for researchers who cannot be satisfied with anonymized data. A safe access center may be a physical facility to which the researcher must travel or an on-line service that the researcher can remotely access. For example, Eurostat, the EU statistical office, operates both a physical safe access center in Luxembourg and an on-line safe access center (called KIOSK [6]). Whatever the case, there is an environment in which the researcher runs her analyses using software provided by the controller and is under monitoring by the controller's staff during her entire work session. Furthermore, output checking rules [1, 8] are manually enforced by the controller's staff to decide whether the outputs of each statistical calculation by the researcher can be safely returned to her.
- Some decentralized data marketplaces, like Ocean [12], complement their offer of anonymized data (data-as-a-service) with the possibility of running computations on the original data they store (compute-to-data). Yet, they have no solution to thwart data leakages arising from the results of computations.

The approaches to monitoring user analyses sketched above are at best at the level of each single statistical calculation (output checking rules) and aimed only at ensuring the confidentiality of the data held by the controller. In fact, in the case of decentralized data marketplaces, not even the above countermeasures are provided.

Contribution and plan of this paper

This is a concept paper where we consider an entire sequence of user analyses rather than a single statistical computation. In our approach:

- The user binds herself to her analysis sequence before seeing any of its outputs. In this way, adaptive analyses aimed at isolating specific confidential records are precluded: all possible strategies to isolate records must be programmed beforehand in the code.
- The user reveals her code on an instruction-by-instruction basis, that is, she discloses her $i+1$-th statistical analysis only after she has received the output

of her i-th statistical analysis. In this way, fair disclosure is achieved: if the controller aborts a certain statistical analysis, the user can avoid disclosing the remainder of her planned analyses to the controller.
– Since the user binds herself to a sequence, this sequence can also be certified by some certification authority as being compliant with other ethical properties. For example, if the analyses include training machine learning models, the fairness and explainability of the trained models are relevant ethical properties.

Section 2 gives background on output checking rules. Section 3 presents our scheme for user's binding to a sequence of analyses with tit-for-tat instruction disclosure. Section 4 analyzes how our framework can be leveraged to enforce confidentiality for the controller's data and other ethical properties, while guaranteeing fair disclosure of the user's analyses. Conclusions and future research avenues are gathered in Section 5.

2 Background

In [1, 8] two slightly different sets of rules of thumb are proposed to decide whether the outputs of various statistical analyses can be safely returned to a user/analyst. "Safely" in this context means without leaking the underlying confidential data.

Each rule can be formalized in terms of the following attributes: *AnalysisType*, *Output*, *Confidential*, *Context* and *Decision*. By way of example, we next quote the rule concerning frequency tables:

AnalysisType: FrequencyTable
Output: Number of units in each cell.
Confidential: YES/NO (YES means the data on which the frequency table is computed are confidential).
Decision: YES/NO

The decision is NO, that is, the output is not returned if data are confidential AND {some cell contains less than 10 units OR a single cell contains more than 90% of the total number of units in a row or column}.

Similar rules are given for other statistical analyses. For example, in [1] rules are specified for magnitude tables, maxima, minima, percentiles, modes, means, indices, ratios, indicators, concentration ratios, variances, skewness, kurtosis, graphs, linear regression coefficients, nonlinear regression coefficients, regression residuals, regression residuals plots, test statistics, factor analyses, correlations and correspondence analyses.

3 Binding to a sequence of analyses with tit-for-tat analysis disclosure

In [3,4], a hash-based coding of programs was proposed to allow a CPU to check the integrity of programs at run-time. The goal of that coding was to detect

at run-time any alteration suffered by the code after it was written, *e.g.* due to computer viruses.

We repurpose and adapt that scheme to allow the user, a.k.a. data analyst, to bind herself to a certain sequence of analyses, which she cannot alter as a result of seeing intermediate results. At the same time, the user does not disclose the next analysis in her sequence until she has obtained the results of her previous analysis.

We adapt here the above hash-based coding for a sequence of analyses. The variants in [3, 4] for non-sequential programs (with branches) could also be adapted for our purpose in a way analogous to that described in the next sections. Such an adaptation is left for future research.

3.1 Instruction preparation

Assume the user's sequence of analyses is $i_1, \ldots, i_n$, where each i_j is an instruction describing a certain statistical analysis. Let H be a specified cryptographically strong one-way hash function such that in general $H(X \oplus Y) \neq H(X) \oplus H(Y)$, where $\oplus$ is the XOR operator (addition modulo 2).

We want to transform instructions $i_1, \ldots, i_n$ so that they have a length close but not greater than the output of H. This is necessary for the security properties below.

If all instructions $i_1, \ldots, i_n$ are shorter than the output of H, then fixed-length instructions $I_1, \ldots, I_n$ can be obtained by padding with a known filler or a redundancy pattern.

However, if we want to accommodate complex statistical analyses or even small routines in each instruction i_j, the length of instructions may be greater than the length of the output of H. In this case, the user can compute the hash of i_j, $h_j = H'(i_j)$, using another cryptographically strong secure hash function H' whose output is shorter than the output of H. Then what is padded *with some redundancy* into $I_1, \cdots, I_n$ are, respectively, $h_1||p_1, \ldots, h_n||p_n$, where each key is appended a pointer p_j to a location where instruction i_j is stored. We assume the output of H' concatenated with a pointer is not longer than the output of H.

3.2 Hash-based coding and execution

We encode $I_1, \ldots, I_n$ into a *trace sequence* $T_0, T_1, \ldots, T_n$, where traces are computed in reverse order according to the following equalities:

$$\begin{aligned} T_0 &= H(T_1) \oplus I_1; \\ T_1 &= H(T_2) \oplus I_2; \\ &\vdots \\ T_{n-1} &= H(T_n) \oplus I_n; \\ T_n &= H(I_n). \end{aligned} \tag{1}$$

To "seal" the trace sequence, the first trace T_0 is signed by the user who has written the analysis sequence or by some certification authority. Call t_0 the signed version of T_0.

To start executing the analysis sequence, traces t_0, T_0 and T_1 need to be supplied to the controller/processor, who verifies the signature t_0 on T_0 and retrieves the first instruction as

$$I_1 := H(T_1) \oplus T_0.$$

In general, after instructions $I_1, \ldots, I_{j-1}$ have been retrieved, learning trace T_j allows the controller to retrieve the j-th instruction as

$$I_j := H(T_j) \oplus T_{j-1}. \tag{2}$$

The final step is to run the instruction/analysis:

- If I_j contains an instruction i_j, this instruction/analysis is run by the controller and, if appropriate, its results are returned to the user.
- If I_j contains $h_j||p_j$, then the controller fetches an analysis instruction i_j at location p_j, checks that $H'(i_j) = h_j$ and, if the check passes, runs i_j and returns its results to the user if appropriate.

The controller can decide on the "appropriateness" of returning a result by (manually or automatically) using an output checking rule such as those recalled in Section 2.

3.3 Security properties

The above coding and execution scheme has two interesting properties.

Proposition 1 (Tit-for-tat instruction disclosure). *The user having constructed the sequence of analyses can disclose her sequence to the controller instruction by instruction, upon receipt of partial results, rather than in one shot.*

Proof. From Expressions (1), it can be seen that each trace T_i encrypts an instruction I_{i+1} by adding to it a pseudorandom value $H(T_{i+1})$ that is as long as I_{i+1}. Thus, the controller cannot decrypt I_{i+1} from T_i until the user reveals T_{i+1} to the controller. □

The second property refers to the binding nature of the sequence.

Proposition 2 (Binding sequence of analyses). *Even if the controller is only revealed the sequence instruction by instruction, he can detect any changes introduced to the sequence since its execution started.*

Proof. We will show this by induction. After the user signs T_0 and the signature t_0 is revealed to the controller, the latter can detect any change in T_0 and abort the sequence execution. For any $j \geq 1$, if $T_0, \ldots, T_{j-1}$ are not changed, the controller can detect any instruction substitution, deletion or insertion of T_j at run-time as follows: if T_j^* is supplied to the controller instead of T_j, then Expression (2) will yield a gibberish I_j^* in place of I_j (because H is a strong hash function). Two cases must be distinguished:

- If I_j contained an instruction i_j, then I_j^* will be gibberish as well, rather than a proper instruction, which will cause the execution to abort.
- If I_j contained a hash value plus a pointer, that is $h_j||p_j$, then I_j^* will be gibberish and will not contain the redundancy that should have been embedded with a valid concatenation of a hash value and a pointer. This will also result in the execution being aborted.

Thus, we have proven that changes in the sequence of traces will be detected.

An alternative for an attacker when I_j contains a hash value h_j plus a pointer p_j is to try to change the instruction i_j at address p_j, say, to i_j^*. If the attacker does this, the controller's check $H'(i_j^*) \stackrel{?}{=} h_j$ will fail and cause the execution to be aborted.

Thus, by just revealing t_0 to the controller, the user is binding herself to the entire (and yet undisclosed) sequence of analyses. □

4 Using binding sequences of user analyses to ensure ethical compliance

When the user binds herself to a sequence of analyses, she actually binds herself to a certain behavior. As the user discloses one instruction/analysis after the other, it is possible for the controller to monitor that the user's behavior stays compliant with some ethical values. We next discuss compliance with the following values: confidentiality, fairness and transparency-explainability. The latter two values are applicable when an analysis consists of training a machine learning model on the controller's original data.

4.1 Checking confidentiality

A data controller offering to run statistical analyses on his confidential data on the user's behalf wants to prevent the user from learning specific confidential records. There are two ways in which the user may seek to learn confidential information from statistical queries:

Single statistical query A single statistical query/analysis may leak individual records if it affects a sufficiently small set of records. For example, computing the average value of an attribute over a query set of size 1 obviously leaks the value of the attribute for the affected record.

Sequence of statistical queries Tracker attacks [2] have been known for about 40 years: a sequence of queries is constructed such that it pads small query sets with enough extra records to give them a size in the allowable range. Then the effect of padding is subtracted, and statistical outputs over small query sets can be obtained.

In our framework, the controller can defend against leaks from single statistical queries by resorting to output checking, discussed in Section 2. The controller

uses output checking rules to decide whether the output of the current instruction/analysis can be safely returned to the user. If it cannot, then the controller withholds the output and aborts the execution of the user's analysis sequence.

Regarding leaks caused by a sequence of instructions, they can be prevented by the controller by using query set size controls [11]. This strategy consists in controlling the size of the set of records isolated by the queries processed so far plus the current query. If the query set size is below a threshold set by the controller, he withholds the output of the current query and aborts the execution of the user sequence. Some amount of output perturbation can be used to increase the safety of query set size control, as hinted in [2].

In case of sequence abort, the user does not need to disclose the remaining instructions/analyses of her sequence. In case of non-abort, the user discloses a new trace to allow the controller to recover a new instruction/analysis.

4.2 *Ex ante* checking of explainability and fairness

If the user's sequence of analyses trains a machine learning model on the controller's data, the controller/processor may wish to make sure the resulting model is explainable or fair before returning it to the user or even before running the user's sequence that trains the model.

Whereas confidentiality can be checked as the user discloses one instruction/analysis after the other, checking that a sequence trains an explainable or a fair model may require examining the entire sequence. Yet, this can hardly be done by the controller if the next instruction is disclosed by the user only after the controller returns her the results of the previous instruction.

A way around is for the user to have her sequence certified by a certification authority trusted by the controller/processor and the user:

1. The user binds herself to her sequence of analyses by encoding its instructions into traces and signing the first trace as explained in Section 3.
2. Then the user submits her *entire* binding sequence of analyses to the certification authority. The user trust the authority not to disseminate or misuse her sequence of analyses.
3. The authority checks the code for the required ethical value:
   - Checking for transparency-explainability can be done by making sure the code corresponds to training an explicitly explainable model (decision rules, decision trees, random forests, etc.).
   - Checking for fairness by code analysis can be done by making sure the code includes instructions for pre-processing, in-processing or post-processing anti-discrimination (*e.g.* [9]).
4. If the check passes, the authority appends its signature to the first trace of the binding sequence.
5. Upon finding the signature of the authority when being revealed the first sequence trace, the controller is reassured that the sequence will train an explainable, respectively fair, model.

5 Conclusions and further research

We have presented a new concept to make sure the sequence of analyses run by a user in a safe access center or a trusted research environment is ethically compliant. Ethical values to be enforced include respondent privacy (non-disclosure of the underlying original microdata), user privacy (fair disclosure between user and controller), and fairness of trained models if any, *inter alia*. On the one side, the user binds herself to a sequence, which means she cannot modify it depending on the intermediate results obtained. This prevents adaptive attacks and thus thwarts disclosure of the underlying respondent microdata (*respondent privacy*). If the user breaks this bond and tries to change her sequence, the controller stops returning the results of the analyses. On the other side, the user reveals her next analysis only if she obtains the result of her previous analysis. This ensures *user privacy* for the rest of the analysis sequence, and it can be construed as a form of *fairness* between user and controller.

Admittedly, our proposal can only be employed if the user's research has a pre-set purpose, so that it is possible for the user to define in advance the sequence of analyses she wants to perform.

Future work will consist of building a demonstrator in a practical trusted research environment. Also, we plan to extend the approach to non-sequential analysis programs (with loops and subroutine calls).

Acknowledgments and disclaimer

Thanks go to a reviewer for making comments and raising stimulating questions. Partial support is acknowledged from the European Commission (projects H2020-871042 "SoBigData++" and H2020-101006879 "MobiDataLab"), the Government of Catalonia (ICREA Acadèmia Prize), and UK Research and Innovation (project MC_PC_21033 "GRAIMatter"). The author is with the UNESCO Chair in Data Privacy, but the views in this paper are his own and are not necessarily shared by UNESCO or the above mentioned funding agencies.

References

1. S. Bond, M. Brandt and P.-P. de Wolf. *Guidelines for the Checking of Output Based on Microdata Research.* Deliverable D11.8, project FP7-262608 "DwB: Data without Boundaries", 2015. `https://ec.europa.eu/eurostat/cros/system/files/dwb_standalone-document_output-checking-guidelines.pdf`
2. D. E. Denning and J. Schlörer. "Inference controls for statistical databases." *Computer* 16(7): 69-82, 1983.
3. J. Domingo-Ferrer. "Software run-time protection: a cryptographic issue". In *Advances in Cryptology – EUROCRYPT'90*, Springer, pp. 474-480, 1991.
4. J. Domingo-Ferrer. "Algorithm-sequenced access control." *Computers & Security* 10(7):639-652, 1991.

5. J. Domingo-Ferrer, D. Sánchez and J. Soria-Comas. *Database Anonymization: Privacy Models, Data Utility, and Microaggregation-Based Inter-Model Connections.* Morgan & Claypool, 2016.
6. Eurostat. "Options for Decentralised/Remote Access to European Microdata." In *5th Meeting of the Working Group on Methodology*, Luxembourg, April 1, 2020 (videoconference).
7. *General Data Protection Regulation.* Regulation (EU) 2016/679. `https://gdpr-info.eu`
8. E. Griffiths, C. Greci, Y. Kotrotsios, S. Parker, J. Scott, R. Welpton, A. Wolters and C. Woods. *Handbook on Statistical Disclosure Control for Outputs (version 1.0).* Safe Data Access Professionals Working Group, 2019. `https://ukdataservice.ac.uk/media/622521/thf_datareport_aw_web.pdf`
9. S. Hajian and J. Domingo-Ferrer. "A methodology for direct and indirect discrimination prevention in data mining," *IEEE Transactions on Knowledge and Data Engineering* 242:35-48, 2013.
10. A. Hundepool, J. Domingo-Ferrer, L. Franconi, S. Gießing, E. Schulte-Nordholt, K. Spicer and P.-P. de Wolf. *Statistical Disclosure Control.* Wiley, 2012.
11. E. Nussbaum and Michael Segal. "Privacy analysis of query-set-size control." In *Privacy in Statistical Databases – PSD 2020.* Springer, pp. 183-194, 2020.
12. Ocean Protocol, accessed June 20, 2022. `https://oceanprotocol.com`