Grant Agreement Number: 825225

Safe-DEED

www.safe-deed.eu

PSI/MPC and multi-user data aggregation protocols v2/2

Deliverable number	*D5.8*
Dissemination level	*Public*
Delivery data	*due 30.11.2020*
Status	*Final*
Authors	*Lukas Helminger, Pascal Steiner*

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 825225.

Changes Summary

Date	Author	Summary	Version
30.10.2020	Lukas Helminger, Pascal Steiner	First Draft (for internal review)	0.1
10.11.2020	Tobias Leander Welling (IFX)	Review	0.2
16.11.2020	Ludovico Boratto (EURE)	Review	0.3
16.11.2020	Lukas Helminger	Final Version (incorporated reviews)	1.0

Executive Summary

This deliverable D5.8 - PSI/MPC and multi-user data aggregation protocols v2 - is an update of D5.3. It is together with D5.7, the final outcome of task T5.1. This task was concerned with developing and improving this work package's core technologies with respect to both practical and theoretical aspects.

The research efforts conducted within the scope of this deliverable lead to two scientific papers. Both are currently in the peer-review process of major conferences in the area of cryptography and privacy. There is the possibility of a third scientific paper depending on future implementation success.

In the future, the plan is to integrate the majority of the developed protocols to the use-cases in Safe-DEED's work packages. In particular, the so-called Private Selective Aggregation protocol will be applied in WP4's Data Valuation Component. In addition, the novel Private Set Intersection protocol will enhance the security in the WP6 demonstrator and thereby reduce the necessary trust assumption.

Table of Contents

List of Figures

Abbreviations

CDR	Call Detail Record
DVC	Data Valuation Component
HE	Homomorphic Encryption
MPC	Multi-Party Computation
PSA	Private Selective Aggregation
PSI	Private Set Intersection
PVC	Public Verifiable Covert Security

1 Introduction

The purpose of this deliverable is to report the progress regarding Private Set Intersection (PSI), Multi-Party Computation (MPC), and secure data aggregation protocols. This document is the second version of this report. It provides two new protocols and an update to a previous protocol. Two of the protocols are currently under peer-review in major cryptography, respectively, privacy conferences. The papers' abstracts can be found in Appendix A.

1.1 Related Documents

D5.8 is a research-focused deliverable and an updated version of D5.3. A general introduction to the field of secure computing can be found in D5.1. Implementations to the protocols below - if already existing - are part of D5.9. This document will contribute to the scientific foundation for the demonstrator D5.11. In particular, the plan is to deploy at least two of the protocols in WP4, respectively, WP6.

1.2 Road-map

In Section 2, we describe our idea of how to improve the security of high-performance PSI protocols without much performance overhead. We continue - in Section 3 - by introducing a novel privacy-preserving protocol dubbed Private Selective Aggregation. In Section 4, we give an update to Multi-Party Computation Accumulators, which we introduced in D5.3. We defier to Appendix B for the technical details of the private set intersection protocol.

2 Private Set Intersection with Public Verifiable Covert Security

In this section, we describe an idea of how we could improve the security guarantees of PSI protocols without significant performance overhead. The main idea is to apply the work of Hong et al. [7] to the PSI setting.

2.1 Current Security Performance Trade-Off

Handling security performance trade-off is crucial when cryptographic protocols move from the academic into the real-world. So far, PSI does not offer a satisfactory trade-off because there are only two options.

First, we have the so-called semi-honest variant. In this model, the performance of PSI protocols is practical. However, this model assumes a certain amount of mutual trust between the two parties. More concretely, security holds only provided that both parties stick to the protocol. Once a party deviates from the protocol, there are no security guarantees left. Besides, an honest party can not detect a cheating party. In many use-case scenarios, this security guarantee is insufficient.

For those use-cases, one would choose PSI protocols that are malicious secure. They provide security even if a party deviates from the protocol. Until this year, it was common to experience

a ten-fold increase of runtime compared to semi-honest protocols leaving them impractical. Pinkas et al. [10] presented a protocol that reduces the communicational overhead to roughly 25% and a two-fold increase in runtime. They achieved this by introducing a new data structure.

2.2 New Approach

In contrast, we propose to close the performance gap between semi-honest security and malicious security by public verifiable covert security (PVC) [7]. PVC is a security model for general secure two-party computation that strikes a compromise between semi-honest and malicious security. Informally speaking, PVC ensures that a cheating party gets detected with a certain probability, e.g., one half. Moreover, the honest party receives a certificate that undisputedly shows the malicious behavior of the cheating party. Hence, it can hold the cheating party accountable.

This public verifiability has immense implications for real-world applications. For instance, a cheating company would have to worry about its reputation as well as legal consequences. Considering this, PVC offers close to malicious security in the realm of real-world security.

However, the performance of two-party PVC protocols is very close to semi-honest protocols. In particular, the overhead in communication is insignificant for multiplying integers. Also, the runtime increases by less than 5%. Even for more complex computations, the performance stays relatively close to the semi-honest case. Since PSI is a special two-party protocol, we believe that similar results can be achieved for PSI protocols. To the best of our knowledge, there exists no implementation of PVC security for PSI.

If the PVC's favorable performance evaluations actually hold the PSI case, it would be the best choice for many real-world applications. Our proposed solution uses only "off-the-shelf" cryptographic primitives compared to the leading malicious secure PSI protocol. This is not a negligible advantage for real-world protocols. Besides, an improvement of the underlying "off-the-shelf" cryptographic primitives directly translates to an improvement of the PSI protocol.

2.3 Definition PVC

Before formally defining the notion of PVC for PSI protocols, we give an intuitive description. PVC consists of two parts that work nicely together. First, a PSI protocol that has PVC must have covert security, i.e., a malicious party gets detected with probability (usually denoted by the Greek letter ϵ). Further, covert security is extended by PVC in the following way. Whenever the malicious party gets detected, the honest party receives a notification of the misbehavior. Moreover, the notification includes a certificate that can be publicly verified by anyone leaving no doubt about the malicious party's misbehavior.

We formalize covert security for PSI protocols via an ideal functionality - as common practice for MPC-protocols - depicted in Figure 1. More precisely, we are speaking about covert security with deterrence $\epsilon \in [0,1]$, where ϵ represents the probability that the malicious party gets caught cheating. Public verifiable covert security with deterrence ϵ extends the above notion. More concretely, PVC is augmented by two algorithms, Blame and Judge. The Blame algorithm is run whenever the honest party detects a malicious behavior. This algorithm takes as input the honest party protocol execution's view and outputs a certificate. The Judge algorithm takes as input a certificate generated from the Blame algorithm and outputs 1 with over-

whelming probability provided that there was a malicious behavior in the protocol's execution. In contrast, if there was no malicious behavior, the Judge should output 0 with overwhelming probability. This ensures that an honest party can not be falsely accused (defamation freeness).

$\mathcal{F}_{PSI}$

S sends a set X or cheat, and C sends a set Y.

1. If S sends a set X then compute and output $X \cap Y$.
2. If S sends cheat then:
   - Send cheated to both parties, with probability ϵ.
   - Send (undetected, Y) to S. Then wait to receive set Z from S, output Z.

Figure 1: Ideal functionality $\mathcal{F}_{PSI}$ for covert security with deterrence ϵ for PSI.

2.4 Protocol

Pascal Steiner's master thesis's goal is to design and implement the first PSI protocol with PVC. In other words, he tries to realize the ideal functionality of $\mathcal{F}_{PSI}$ with public verifiability. The idea is to combine the works of Hong et al. [7] and the PSI protocol of Kales et al. [8]. This particular PSI protocol was used because it is currently integrated in WP6's demonstrator. The actual description of the protocol is complex and needs an advanced understanding of several cryptographic primitives. Therefore, we refer the interested reader to Appendix B. Note that the necessary preliminaries will be published and discussed in Pascal Steiner's master thesis.

3 Private Selective Aggregation

In this section, we describe a protocol that we dubbed Private Selective Aggregation (PSA). The goal was to develop a secure aggregation protocol that is suitable for enterprise-scale and government-scale data sets. The general protocol and the Covid-19 Heat-map use-case were introduced in the research paper [2]. The following is a high-level description of the paper for details, please see the full paper.

Our two-party protocol enables a client to retrieve aggregated information about a server's database privately. In addition, the client has the opportunity to choose which database entries should be in the aggregation.

3.1 Threat Model

The threat model of this protocol is two-fold. On the one hand, the server should neither learn which entries will be aggregated nor the aggregated value itself. Translated to our example Figure 2: Netflix should not learn that Paul wants to know how long Amy and Jack are streaming.

On the other hand, the client should not learn individual database entries - Paul should not know that Amy streams 2.0 hours or that Jack streams 2.5 hours.

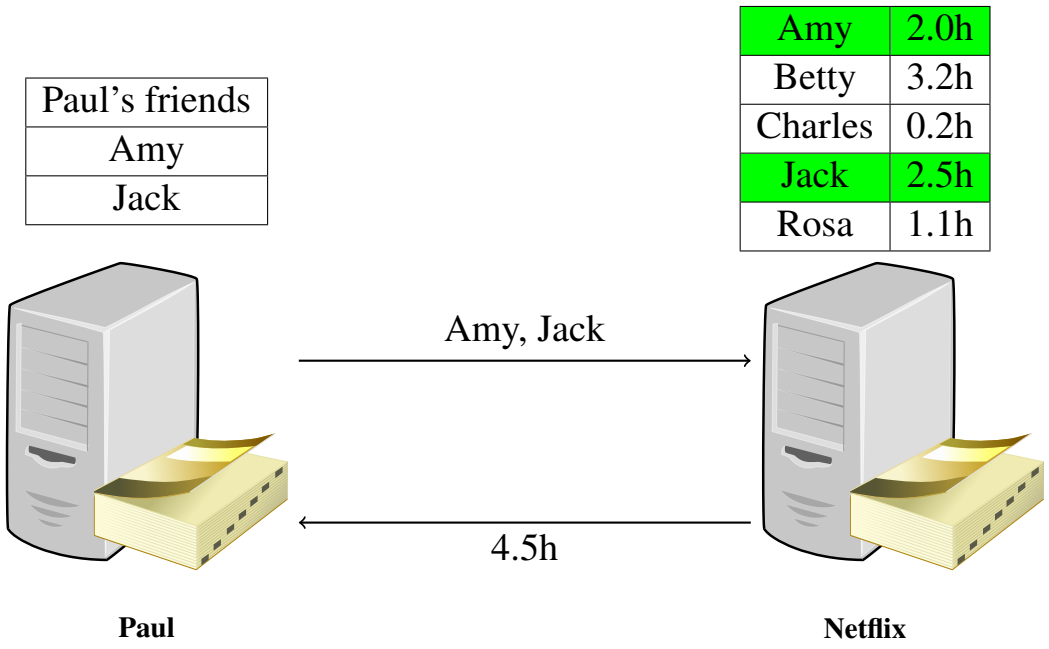


Figure 2: Naive protocol.
Paul wants to know how long combined his friends Amy and Jack are streaming from Netflix.

3.2 Protocol

To achieve the privacy goals outlined above, we use various state-of-the-art privacy-preserving primitives. In particular, we apply homomorphic encryption, zero-knowledge proof techniques, and differential privacy. Homomorphic encryption (HE) protects the client's request from the server. Due to the nature of homomorphic encryption, the server can perform the data aggregation without decrypting the client's request. To prevent the client from learning individual database entries, we ensure that the client's request has a specific minimum size by applying so-called zero-knowledge proof techniques. The server can also add noise - in the sense of differential privacy - to the aggregated value before sending it back to the client. This becomes necessary if the aggregated value would still be a privacy issue. Informally, our protocol is secure as long as the underlying homomorphic encryption scheme is secure.

More formally, we defined our protocol as an ideal functionality, which is a common practice for secure computation protocols. We showed input privacy in the presence of a maliciously controlled mobile operator provided that the homomorphic encryption scheme is semantically secure.

We implemented the protocol for the two use-cases below. The implementations are open source [1]. For a description of the implementations, see Safe-DEED deliverable D5.9 - "Implementation of cryptographic building blocks and specialized protocols v2/3".

[1] `https://github.com/IAIK/CoronaHeatMap`

3.3 Use-Case Privacy-Preserving Questionnaire

One component of Safe-DEED is concerned with data valuation (for details, see Safe-DEED deliverables D4.1 and D4.2). The purpose of the Data Valuation Component (DVC) is to provide companies with a tool for estimating their data's value. Roughly speaking, this assessment is split into two sub-components:

- Qualitative Information Extracting and Data Scoring Sub-Component (QDSC)
- Automatic Data Analysis and Scoring Sub-Component

In this report, we show how to perform the QDSC in a privacy-preserving way.

3.3.1 Qualitative Information Extracting and Data Scoring Sub-Component (QDSC)

This stage of the data valuation process requires users to provide information about their data set. The users are asked to perform a questionnaire that comes in three parts.

- Business Intelligence: Data acquisition cost and business impact.
- Domain-Specific: Generation of data.
- Data Science: Technical details.

If the DVC is offered as a Software as a Service (SaaS), then the questionnaire could be seen as a liability by the users. Namely, not every company will be comfortable to share the answers to the following questions with Eurecat.

- Is the data already producing money?
- Is the data usage shared with partners?
- Is the data used to establish a new business/R&D direction?
- Does processing involve significant costs?

Therefore, it is important to assure the companies of the confidentiality of their data even against the malicious behavior of Eurecat.

3.3.2 Privacy-Preserving QDSC

A customized version of our protocol enables users to perform the QDSC in a privacy-preserving way. More concretely, the answers to the questionnaire get homomorphically encrypted before sending them to Eurecat. Due to the nature of homomorphic encryption, Eurecat can perform the valuation without decrypting the user's answers. (The user is the only one who has the decryption key.) After the valuation, Eurecat ends up with the encrypted score for the user's answers. Eurecat sends the score back to the user, which can then decrypt the result and thereby receive its score.

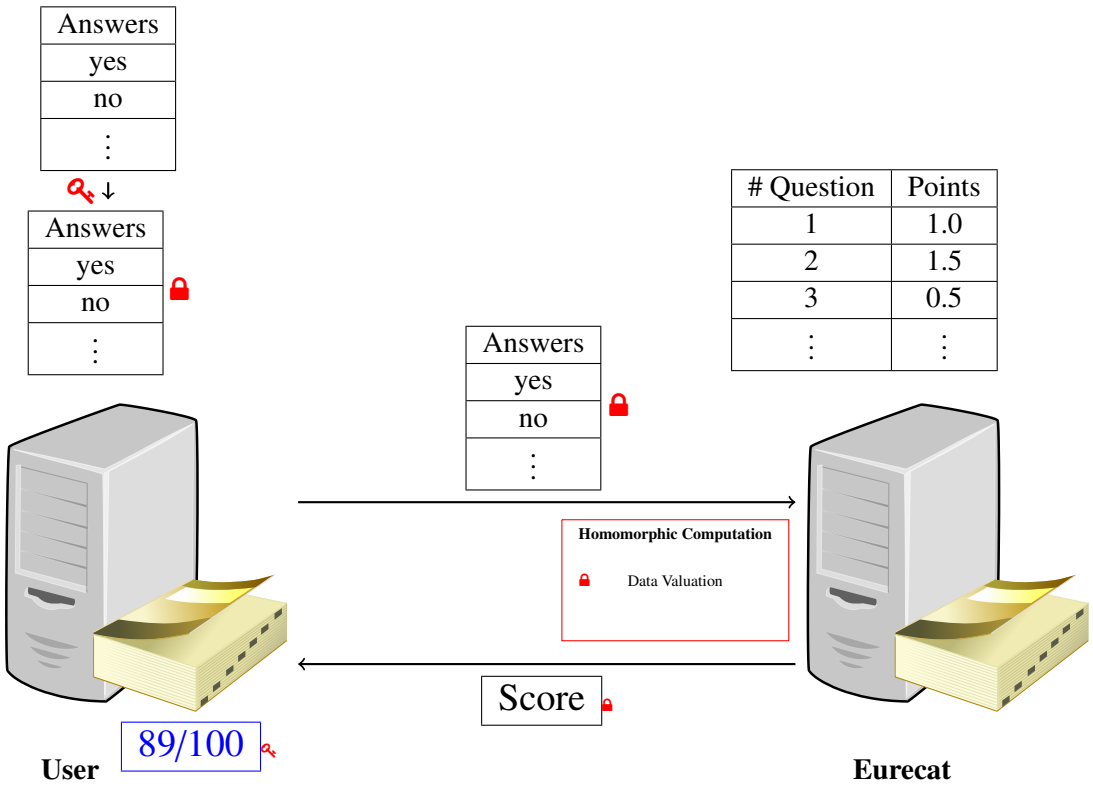


Figure 3: Privacy-Preserving QDSC
The user encrypts his answers to the questionnaire before sending them. Eurecat posses an evaluation table that assigns each answer to a particular value. This table gets matched with the answer by homomorphic means, i.e., Eurecat does not learn the user's answers. Eurecat then sends the encrypted score back to the user, decrypting it with the private key (red).

3.4 Use-Case Covid-19 Heat Map

3.4.1 Connecting Mobility to Infectious Diseases

Human mobility is undisputedly one of the critical factors in infectious disease dynamics. On the one side, increased human mobility may account for more contacts between receptive and infected individuals. On the other side, human travel may introduce pathogens into new geographical regions. Both cases can be responsible for an increased prevalence and even an outbreak of the infectious disease [12]. In particular, human travel history has been shown to play a critical role in the propagation of infectious diseases, like influenza [3] or measles [5]. Therefore understanding the spatiotemporal dynamics of an epidemic is closely tied to understanding the movement patterns of infected individuals.

Until a few years ago, researchers had to rely on static data – relative distance and population distribution – to model human mobility, which was then combined with a transmission model of a particular disease resulting in an epidemiological model. This model was then used to improve the understanding of the geographical spread of epidemics. Mobile phones and their location data have the unique potential to improve these epidemiological models further. Indeed, recent works [13, 14] have consistently been showing that substituting the static mobility data

with mobile phone data leads to significantly more accurate models. Integrating such up-to-date mobility patterns allowed them to identify hotspots with a higher risk of contamination, enabling policymakers to apply focused measures.

While prior studies have exclusively relied on a mobile operator's subscribers' aggregated data, it may be preferable to contemplate aggregated mobility data of infected individuals only. Indeed, a cholera study [4] observed that although their model has high accuracy, it performs less well when the cumulative incidence is low. They speculated that demographic stochasticity could be one reason for the bad performance of their model. In other words, the infected individuals' mobility pattern may not be precisely reflected by the population's mobility if the prevalence is low. In order to mitigate this problem, we propose the usage of infected individuals' mobile phone data, which should lead to an improvement in the predictive capabilities of epidemiological models, especially in highly dynamic situations.

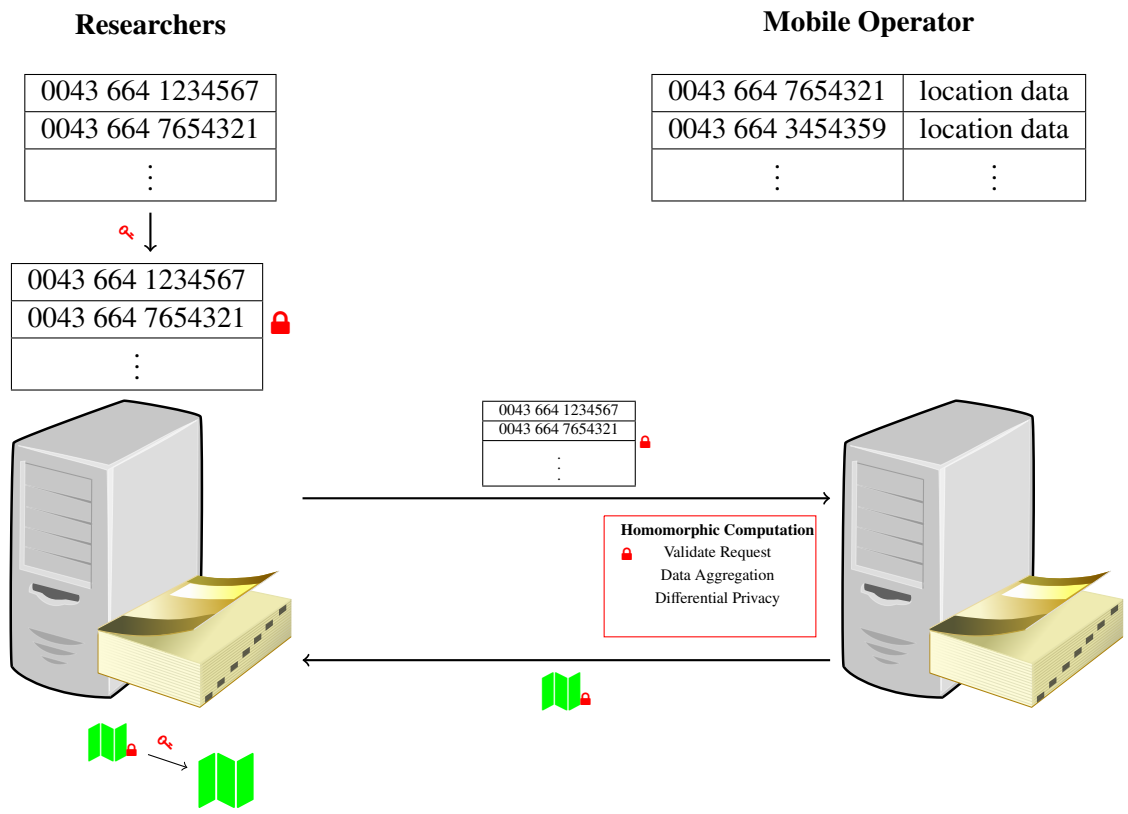
Clearly, naively linking mobile phone data with infected individuals would massively intrude on privacy. Namely, either the mobile network operator would have to know which subscribers are infected, or the epidemiological researchers would have to get access to non-anonymized data records. As a result, previous studies considered the availability of travel history information from patients as not possible and attempted to control possible biases in the results manually [11].

3.4.2 Connecting Mobility to Infectious Diseases via Applied Cryptography

Our protocol can be applied to report the aggregated mobile phone location data of infected individuals while still maintaining compliance with privacy expectations. We use various state-of-the-art privacy-preserving cryptographic primitives to design a two-party protocol that achieves the following: The epidemiological researchers (or a health authority) input patients' identifiers, whereas the mobile operator inputs call data records (CDRs) of its subscribers. The protocol outputs the patients' aggregated location data from the CDRs to the researchers. Informally, neither do the researchers access individuals' CDRs nor do the mobile operator learn which subscribers were involved in the computation, and therefore, who is infected.

For an interested audience with little security and cryptography background, we created a webpage[2] that describes our approach and basically has the following message: Even in times of crisis where it is tempting to (temporarily) lower data protection standards for purposes of big data analytics, there are technical methods to keep data protection standards high. Moreover, those technical methods are practical and available.

[2] `https://covid-heatmap.iaik.tugraz.at`



3.5 Scalability

Despite using homomorphic encryption - which usually has a very high computational overhead - the protocol is practical for real-world parameters. More concretely, the protocol scales linearly in the dimension of the database of the server, depicted Figure 4.

Benchmarks for the Covid-19 Heat-map use-case were able to show that privacy-preserving health data analytics is possible even on a national scale. We tested our protocol for parameters suitable for Austria (20000 cell towers and 8 million mobile network operator subscribers), i.e., we have a database of dimension $2^{23} \times 2^{15}$) on the server-side.

We ran several tests with different security levels. For the most popular security level (80-bit) in the HE setting, the protocol takes between half an hour and four hours depending on the statistical security parameter. The data transfer includes a one-round trip with less than 1GiB. For details, we have included the tables 1 and 2 from the paper.

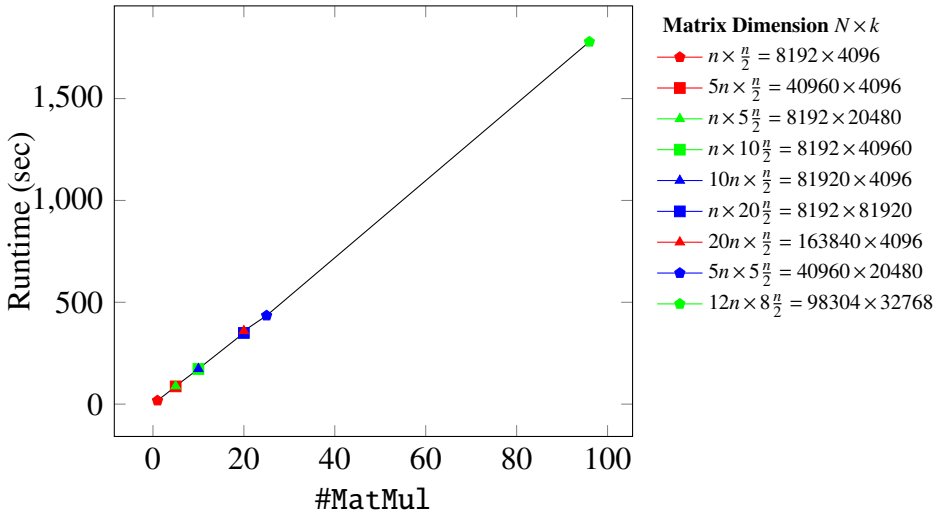


Figure 4: Linear dependency of the runtime of the overal matrix multiplication to the number of `MatMul` evaluations. BFV parameters are: $\log_2(p) = 33$, $\log_2(q) = 218$, $n = 8192$, $\kappa = 128$.

Table 1: Runtime for the Data Aggregation Phase for different parameters using 88 threads. The column Masking indicates whether this parameter set is only able to evaluate the matrix multiplication (✗), or gives the statistical privacy ν (in bits) provided by the masking value.

Nr.	BFV $\log_2(p)$	BFV $\log_2(q)$	BFV n	BFV κ	Matrix N	Matrix k	#MatMul total / per thread	Masking ν	Runtime min
1	33	218	8192	128	2^{23}	2^{15}	8192 / 96	31	59.36
2	60	218	8192	128	2^{23}	2^{15}	8192 / 96	✗	89.87
3	60	438	16384	128	2^{23}	2^{15}	2048 / 24	58	267.19
4	33	162	4096	80	2^{23}	2^{15}	32768 / 384	✗	33.55
5	33	329	8192	80	2^{23}	2^{15}	8192 / 96	31	89.32
6	60	329	8192	80	2^{23}	2^{15}	8192 / 96	58	140.82

Table 2: Data transmission in MiB for the different parameters in Table 1. Values include keys for evaluating the masking value when applicable.

Nr.	Client ct	Client gk	Client rk	Client Total	Server ct	Total
1	256.2	87.6	1.3	345.1	1.0	346.1
2	256.2	81.4	-	337.6	2.0	339.6
3	512.1	639.2	9.0	1160.3	2.0	1162.3
4	128.3	6.2	-	134.5	1.0	135.5
5	384.2	183.9	2.6	570.7	1.0	571.7
6	384.2	183.9	2.6	570.7	2.0	572.7

4 Multi-Party Computation Accumulators: Update

In this section, we are reporting an update regarding the researcher of Multi-Party Computation Accumulators (MPC-accumulators). MPC-accumulators were co-developed by researchers from Safe-DEED, see Safe-DEED deliverable D5.3 and D5.5 for the initial work. Since then, there have been substantial changes, including the new title: "Multi-Party Revocation in Sovrin: Performance through Distributed Trust" [6].

Firstly, we considerably improved the performance of MPC-accumulators. Not only were we able to reduce the online phase by a magnitude of two, but we could also reduce the offline runtime by a factor of approximately five. These values hold for parameters expected in real-world applications. As a representative example, 1 shows the evaluation algorithm's performance for malicious security in the Wide Area Network (WAN) setting. The full benchmarks can be found in the paper.

Secondly, our MPC-accumulators can apply to a wider range of trust settings. So far, MPC-accumulators worked only as long as all parties cooperated. In contrast, now MPC-accumulators work in a so-called threshold environment. The MPC-accumulator's operator can define that it should work as long as a certain number of parties cooperate. This greatly enhances the flexibility of our scheme.

Besides these technical upgrades, we describe a new use-case in the paper. MPC-accumulators have great potential for revocation in distributed credential systems such as Sovrin[3]. Currently, this requires a trusted credential issuer, which is a single point of failure. Applying MPC-accumulators in this scenario results in a reduction of the trust assumption into the Sovrin foundation.

5 Conclusion

In this deliverable, we achieved three different objectives. At first, we outlined how to get a better security performance trade-off for PSI protocols. The idea is to take the fastest protocol and enhance real-world security by talking business incentives into consideration. We will work further on an actual implementation of this protocol. Secondly, we developed a novel protocol for private, selective data aggregation, which is highly scalable. The scalability was achieved by combing the strengths of several privacy-enhancing technologies. Also, this protocol is immensely versatile, as highlighted by completely different use-cases. At last, we showed a significant performance improvement for MPC-accumulators.

[3] https://sovrin.org/

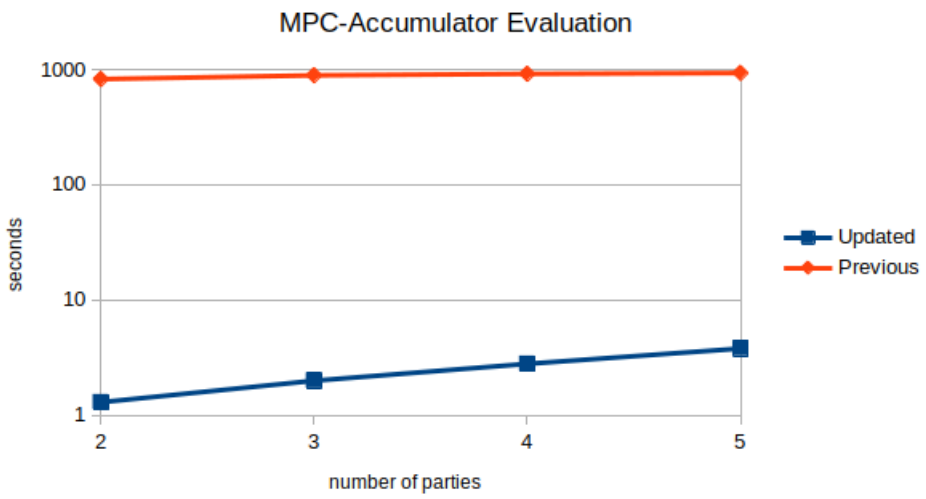


Figure 5: Runtime MPC-Accumulator Eval Algorithm

6 References

[1] Gilad Asharov and Claudio Orlandi. Calling out cheaters: Covert security with public verifiability. In *ASIACRYPT*, volume 7658 of *Lecture Notes in Computer Science*, pages 681–698. Springer, 2012.

[2] Alessandro Bruni, Lukas Helminger, Daniel Kales, Christian Rechberger, and Roman Walch. Privately connecting mobility to infectious diseases via applied cryptography. *arXiv e-prints*, pages arXiv–2005, 2020.

[3] Neil M Ferguson, Derek AT Cummings, Simon Cauchemez, Christophe Fraser, Steven Riley, Aronrag Meeyai, Sopon Iamsirithaworn, and Donald S Burke. Strategies for containing an emerging influenza pandemic in southeast asia. *Nature*, 437(7056):209–214, 2005.

[4] Flavio Finger, Tina Genolet, Lorenzo Mari, Guillaume Constantin de Magny, Noël Magloire Manga, Andrea Rinaldo, and Enrico Bertuzzo. Mobile phone data highlights the role of mass gatherings in the spreading of cholera outbreaks. *Proceedings of the National Academy of Sciences*, 113(23):6421–6426, 2016.

[5] Bryan T Grenfell, Ottar N Bjørnstad, and Jens Kappey. Travelling waves and spatial hierarchies in measles epidemics. *Nature*, 414(6865):716–723, 2001.

[6] Lukas Helminger, Daniel Kales, Sebastian Ramacher, and Roman Walch. Multi-party revocation in sovrin: Performance through distributed trust.

[7] Cheng Hong, Jonathan Katz, Vladimir Kolesnikov, Wen-jie Lu, and Xiao Wang. Covert security with public verifiability: Faster, leaner, and simpler. In *EUROCRYPT (3)*, volume 11478 of *Lecture Notes in Computer Science*, pages 97–121. Springer, 2019.

[8] Daniel Kales, Olamide Omolola, and Sebastian Ramacher. Revisiting user privacy for certificate transparency. In *EuroS&P*, pages 432–447. IEEE, 2019.

[9] Yehuda Lindell and Benny Pinkas. An efficient protocol for secure two-party computation in the presence of malicious adversaries. In *EUROCRYPT*, volume 4515 of *Lecture Notes in Computer Science*, pages 52–78. Springer, 2007.

[10] Benny Pinkas, Mike Rosulek, Ni Trieu, and Avishay Yanai. PSI from paxos: Fast, malicious private set intersection. In *EUROCRYPT (2)*, volume 12106 of *Lecture Notes in Computer Science*, pages 739–767. Springer, 2020.

[11] Andrew J Tatem, Zhuojie Huang, Clothilde Narib, Udayan Kumar, Deepika Kandula, Deepa K Pindolia, David L Smith, Justin M Cohen, Bonita Graupe, Petrina Uusiku, et al. Integrating rapid risk mapping and mobile phone call record data for strategic malaria elimination planning. *Malaria journal*, 13(1):52, 2014.

[12] Amy Wesolowski, Caroline O Buckee, Kenth Engø-Monsen, and Charlotte Jessica Eland Metcalf. Connecting mobility to infectious diseases: the promise and limits of mobile phone data. *The Journal of infectious diseases*, 214(suppl_4):S414–S420, 2016.

[13] Amy Wesolowski, Nathan Eagle, Andrew J Tatem, David L Smith, Abdisalan M Noor, Robert W Snow, and Caroline O Buckee. Quantifying the impact of human mobility on malaria. *Science*, 338(6104):267–270, 2012.

[14] Amy Wesolowski, Taimur Qureshi, Maciej F Boni, Pål Roe Sundsøy, Michael A Johansson, Syed Basit Rasheed, Kenth Engø-Monsen, and Caroline O Buckee. Impact of human mobility on the emergence of dengue epidemics in pakistan. *Proceedings of the National Academy of Sciences*, 112(38):11887–11892, 2015.

A Scientific Papers' Abstracts

Privately Connecting Mobility to Infectious Diseases via Applied Cryptography

Alexandros Bampoulidis[4] Alessandro Bruni[3], Lukas Helminger[1,2], Daniel Kales[1], Christian Rechberger[1], and Roman Walch[1,2]

[1] Graz University of Technology, Graz, Austria
[2] Know-Center GmbH, Graz, Austria
[3] Katholieke Universiteit Leuven, Leuven, Belgium
[4] Research Studio Data Science, Vienna, Austria

Abstract. Human mobility is undisputedly one of the critical factors in infectious disease dynamics. Until a few years ago, researchers had to rely on static data to model human mobility, which was then combined with a transmission model of a particular disease resulting in an epidemiological model. Recent works have consistently been showing that substituting the static mobility data with mobile phone data leads to significantly more accurate models. While prior studies have exclusively relied on a mobile operator's subscribers' aggregated data, it may be preferable to contemplate aggregated mobility data of infected individuals only. Clearly, naively linking mobile phone data with infected individuals would massively intrude privacy. This research aims to develop a solution that reports the aggregated mobile phone location data of infected individuals while still maintaining compliance with privacy expectations. To achieve privacy, we use homomorphic encryption, zero-knowledge proof techniques, and differential privacy. Our protocol's open-source implementation can process eight million subscribers in one and a half hours. Additionally, we provide a legal analysis of our solution with regards to the General Data Protection Regulation.

Keywords: FHE, privacy, Covid-19, mobile data, GDPR

1 Introduction

1.1 Human Mobility and Infectious Diseases

Human mobility is undisputedly one of the critical factors in infectious disease dynamics. On the one side, increased human mobility may account for more contacts between receptive and infected individuals. On the other side, human travel may introduce pathogens into new geographical regions. Both cases can be responsible for an increased prevalence and even an outbreak of the infectious disease [55]. In particular, human travel history has been shown to play a critical role in the propagation of infectious diseases, like influenza [24] or measles [31]. Therefore understanding the spatiotemporal dynamics of an epidemic is closely tied to understanding movement patterns of infected individuals.

Multi-Party Revocation in Sovrin: Performance through Distributed Trust

Lukas Helminger[1,2], Daniel Kales[1], Sebastian Ramacher[3], and Roman Walch[1,2]

[1] Graz University of Technology, Graz, Austria
{lukas.helminger,daniel.kales,roman.walch}@iaik.tugraz.at
[2] Know-Center GmbH, Graz, Austria
[3] AIT Austrian Institute of Technology, Vienna, Austria
sebastian.ramacher@ait.ac.at

Abstract. Accumulators provide compact representations of large sets and enjoy compact membership witnesses. Besides constant-size witnesses, public-key accumulators provide efficient updates of both the accumulator itself and the witness; however, they come with two drawbacks: they require a trusted setup and – without knowledge of the secret trapdoors – their performance is not practical for real-world applications with large sets. Recent improvements in the area of secure multi-party computation allow us to replace the trusted setup with a distributed generation of the public parameters.

In this paper, we introduce multi-party public-key accumulators dubbed dynamic linear secret-shared accumulators. We present versions of dynamic public-key accumulators in bilinear groups giving access to more efficient witness generation and update algorithms that utilize the shares of the secret trapdoors sampled by the parties generating the public parameters.Specifically, for the t-SDH-based accumulators, we provide a maliciously-secure variant sped up by a secure multi-party computation (MPC) protocol (IMACC'19) built on top of SPDZ. For this scheme, a performant proof-of-concept implementation is provided, which substantiates the practicability of public-key accumulators in this setting. With our implementation in two MPC frameworks, MP-SPDZ and FRESCO, we obtain more efficient accumulators for both medium-sized (2^{10}) and large (2^{14} and above) accumulated sets.

Finally, we explore applications of dynamic linear secret-shared accumulators to revocations schemes of group signatures and credentials system. In particular, we consider it as part of Sovrin's system for anonymous credentials where credentials are issued by the a foundation of trusted nodes. Hence, our accumulators naturally fit this setting.

Keywords: multiparty computation, dynamic accumulators, distributed trust

B PSI with PVC Protocol

B.1 Covert Security With Public Verifiability

As already discussed in 2, there is another security model regarding multi party computation besides semi-honest and malicious security. Covert security aims to be only slightly slower than semi-honest security while also providing a practical level of security. The security guarantee is that a dishonest party is caught cheating by the other party with probability $1-\epsilon$, but is able to cheat successfully with a tweakable probability ϵ.
Though, the guarantee provided by covert security on its own is not enough since the cheating party can simply deny the claims of the honest participant. Hence, an honest party cannot convince anybody else that someone cheated, and reputation is only damaged towards the involved honest party. For this purpose, one needs publicly verifiable covert (PVC) security, as it was first introduced in [1]. PVC security enables the honest party to generate a certificate, which proves that a dishonest party has cheated. Due to this mechanism, the reputation of cheating parties is now damaged publicly and may have negative financial or legal consequences, which is a reasonable incentive not to try to cheat.
The PVC security protocols follow the cut-and-choose paradigm. In such a protocol, one first defines a deterrence factor λ, which indicates the probability of catching a cheating party. The garbler, called P_A in the context of this description, picks λ different seeds as well as corresponding witnesses, while the evaluator, denoted as P_B, picks a random evaluation index $\hat{\jmath} \in (1,\ldots,\lambda)$. Then signed oblivious transfer is run on the seeds generated before. Signed OT is a process where a party obliviously learns one out of two inputs plus a signature on the learned value.
The input of the evaluator for the oblivious transfer is always 0 except for $\hat{\jmath}$th instance of the OT, where the input is 1 instead. After having executed the signed OT, P_B knows all seeds except $seed_{\hat{\jmath}}$. For the evaluation instance, P_B learns the witness $witness_{\hat{\jmath}}$ corresponding to $seed_{\hat{\jmath}}$ instead. Next, P_A generates λ garbled circuits, using $seed_j$ as randomness in the jth instance.
Furthermore, commitments to the generated garbled circuits are made, signed, and finally sent to the evaluator. In order to let the party, P_B learn the wire labels for its input, signed OT is performed once again. For every check instance, that is for every index $j \neq \hat{\jmath}$, the commitments are checked with the help of each seed $seed_j$. If one circuit commitment is invalid, a certificate is issued that consists of the inconsistent values as well as the signatures on them. If every commitment is correct, party P_B reveals its chosen evaluation index $\hat{\jmath}$ to P_A. P_A sends the $\hat{\jmath}$th garbled circuit to P_B, along with P_A's own input wire labels, allowing P_B to finally evaluate the garbled circuit.
The only way a dishonest party P_A is able to cheat successfully is to correctly guess which one of the λ instances is the $\hat{\jmath}$th instance, the evaluation instance. With a total of λ instances, this means P_A is caught with a probability of $1-\frac{1}{\lambda}$. Unfortunately, the described form of a PVC secure protocol has a few major problems. First off, signed oblivious transfers are computationally expensive and also complex to implement. Secondly, the certificates can become quite large since they depend on the size of the circuit.
Additionally, it is possible for the garbler to conduct selective-failure attacks. The vulnerability occurs when P_A sends a single incorrect wire label for the input of P_B for the $\hat{\jmath}$th garbled

circuit. For instance, if P_A corrupts an input wire label for the one-value and P_B aborts, then P_A learns that the value on that input wire was indeed one. Technically there is a solution to prevent these selective-failure attacks, namely a technique called XOR-tree (see [9]), but this method is not very practical since it needs additional signed oblivious transfers.

B.1.1 PVC Secure Protocol Improvements

The work of [7] manages to efficiently solve the problems described in the previous section. The basic structure is similar, though. The two parties are once again the garbler P_A (the server) with input $x \in \{0,1\}^{n_1}$ and key-pair (pk, sk) for a signature scheme, and the evaluator P_B (the client) with input $y \in \{0,1\}^{n_2}$. It is assumed that P_B has gained knowledge of the public key pk beforehand.
Again, λ instances of the garbled circuit are created. One instance, enumerated with the uniformly chosen index $\hat{\jmath}$, constitutes the evaluation instance, which is the actual instance that is evaluated in order to calculate the output. The other instances are called check instances and are used to find out whether the participants of the protocol are honest or not.
The garbling scheme is defined by two algorithms. The garbling algorithm **Gb** takes the security parameter 1^κ and some circuit C with $n = n_1 + n_2$ input wires and n_3 output wires as input, and produces input wire labels pairs $\{X_{i,0}, X_{i,1}\}_{i=1}^{n}$, output wire label pairs $\{Z_{i,0}, Z_{i,1}\}_{i=1}^{n_3}$ and the garbled circuit GC as the output. **Eval** is the algorithm for the evaluation and outputs output wire labels $\{Z_i\}_{i=1}^{n_3}$ upon supplying the input, which is consisting of a garbled circuit GC and the corresponding input wire labels $\{X_{i,0}, X_{i,1}\}_{i=1}^{n}$.
One very important change to the standard PVC secure protocol is that the seeds $\{seed_j^A\}_{j\in[\lambda]}$, which are generated by party P_A, do not only derandomize the creation of the garbled circuits GC_j and the corresponding commitments c_j on them, but the entire execution of the remaining protocol. This includes the oblivious transfer, denoted as $\prod_{OT}$, employed in order to let the client learn its input wire labels. Hence, signed OTs are no longer required in this step. As the entire execution of the protocol is determined by the seeds $\{seed_j^A\}_{j\in[\lambda]}$ the client receives from the server in the initial OT, selective-failure attacks are also prevented.
Furthermore, P_A must sign the transcript of each instance in the protocol. In case P_A cheats, P_B is able to create a certificate using the seed of the inconsistent instance and the signed transcript. The signed transcript σ_j includes the circuit C, the instance index j, the seed commitments h_j of P_B, the transcript $trans_j$ of the OT for the seeds, the transcript hash $\mathcal{H}$ of the OT for the input wire labels, and the commitment on the whole circuit including the labels, c_j. A transcript hash $\mathcal{H}$ is defined as $\mathcal{H} = (H(m_1), H(m_2), ...)$, where H is a hash function with output length 2κ and $(m_1, m_2, ...)$ represent the messages that the two parties exchange alternately during a conversation.
On the other hand, defamation freeness is achieved by P_B, also committing to its randomness, that is, its seeds $\{seed_j^B\}_{j\in[\lambda]}$. The commitments on the seeds $h_j = \mathbf{Com}(seed_j^B)$ are included in P_A signatures σ_j, which makes it impossible for P_B to falsely claim that P_A has cheated, since P_B derives all its randomness from those seeds. Deriving its randomness from a seed means that the party is using a pseudo random function in CTR mode with said seed as key. **Com** denotes a commitment scheme, the corresponding decommitment $decom$ constitutes the randomness used for the commitment.
The last major problem that remains is the signed oblivious transfers that are put to use when

letting P_B learn the seeds $\{seed_j^A\}_{j\in[\lambda]}$ of P_A for all but one garbled circuit instance. The need for the signed OTs in this step disappears due to the commitments h_j of P_B to its seeds $\{seed_j^B\}_{j\in[\lambda]}$, because the seeds $\{seed_j^A\}_{j\in[\lambda]}$ that P_B learns can be reconstructed from the transcripts $trans_j$ of the execution of the OT protocol $\prod_{OT}$ plus the seeds $\{seed_j^B\}_{j\in[\lambda]}$. The transcript $trans_j$ is also signed by the garbler, which enables anyone to verify the seeds used by P_A publicly. The certificate size is also constant by having P_B generate all its randomness solely from a short seed.
The protocol in full detail is depicted in Figure 6.
Our first proposal for a PSI protocol with PVC can be found in Figure 7.

P_A **(Garbler)**		P_B **(Evaluator)**
Input: $x \in \{0,1\}^{n_1}$, keys (pk, sk)	Agree on circuit C, security parameter κ, deterrence factor λ	Input: $y \in \{0,1\}^{n_2}$, knows pk
		Choose κ-bit $\{seed_j^B\}_{j\in[\lambda]}$
	$\{h_j\}_{j\in[\lambda]}$ ←	Set $h_j \leftarrow \mathbf{Com}(seed_j^B)\ \forall j$
Choose κ-bit $\{seed_j^A, witness_j\}_{j\in[\lambda]}$		Choose uniform $\hat{j} \in [\lambda]$
		Set $b_{\hat{j}} = 1$ and $b_j = 0$ for $j \neq \hat{j}$
$(seed_j^A, witness_j)$ →	$\boxed{\prod_{OT} \forall j}$	$b_j, seed_j^B$ ←
$trans_j$ ←		$\{seed_j^A\}_{j\neq\hat{j}}, witness_{\hat{j}}, trans_j$ →
$\forall j \in [\lambda]$ garble C using $seed_j^A$		
Obtain GC_j,		
input-wire labels of P_A $\{A_{j,i,b}\}_{i\in[n_1],b\in\{0,1\}}$,		
input-wire labels of P_B $\{B_{j,i,b}\}_{i\in[n_2],b\in\{0,1\}}$,		
and output-wire labels $\{Z_{j,i,b}\}_{i\in[n_3],b\in\{0,1\}}$		
$\{(B_{j,i,0}, B_{j,i,1})\}_{i=1}^{n_2}, seed_j^A$ →	$\boxed{\prod_{OT} \forall j}$	y if $\hat{j} = j, 0^{n_2}$ otherwise, $seed_j^B$ ←
$\mathcal{H}_j$ ←		$\{B_{j,i,y[i]}\}_{i\in[n_2]}, \mathcal{H}_j$ →
Compute $h_{j,i,b}^A \leftarrow \mathbf{Com}(A_{j,i,b})\ \forall j,i,b$		
$c_j \leftarrow \mathbf{Com}(GC_j, \{h_{j,i,b}^A\}_{i\in[n_1],b\in\{0,1\}},$		
$\{Z_{j,i,b}\}_{i\in[n_3],b\in\{0,1\}})$, where each pair $(h_{j,i,0}^A, h_{j,i,1}^A)$		
is randomly permuted. (Randomness from $seed_j^A$)		
$\forall j \in [\lambda] : \sigma_j \leftarrow \mathbf{Sign}_{sk}(C, j, h_j, trans_j, \mathcal{H}_j, c_j)$	$\{c_j\}_{j\in[\lambda]}, \sigma_j$ →	
		Check if σ_j is valid $\forall j$,
		abort with output $\perp$ otherwise
		$\forall j \neq \hat{j}$ use $seed_j^A$ to compute $\hat{\mathcal{H}}_j, \hat{c}_j$
		Check if $(\hat{\mathcal{H}}_j, \hat{c}_j) = (\mathcal{H}_j, c_j)\ \forall j \neq \hat{j}$
		If check fails output corrupted, send
		$cert = (j, trans_j, \mathcal{H}_j, c_j, \sigma_j, seed_j^B, decom_j^B)$
		to P_A, and halt
	$\hat{j}, \{seed_j^A\}_{j\neq\hat{j}}, witness_{\hat{j}}$ ←	
Check if $\{seed_j^A\}_{j\neq\hat{j}}, witness_{\hat{j}}$ are correct,		
otherwise abort		
	$GC_{\hat{j}}, \{A_{\hat{j},i,x[i]}\}_{i\in[n_1]}$ →	
	$\{h_{\hat{j},i,b}^A\}_{i\in[n_1],b\in(0,1)}$ →	
	$\{Z_{\hat{j},i,b}\}_{i\in[n_3],b\in(0,1)}$ →	
	$decom_{\hat{j}}, \{decom_{\hat{j},i,x[i]}^A\}$ →	
		Check if $c_{\hat{j}} =$
		$\mathbf{Com}(GC_{\hat{j}}, \{h_{\hat{j},i,b}^A\}, \{Z_{\hat{j},i,b}\}, decom_{\hat{j}})$ and
		$\mathbf{Com}(A_{\hat{j},i,x[i]}; decom_{\hat{j},i,x[i]}^A) \in \{h_{\hat{j},i,b}^A\}_{b\in\{0,1\}} \forall i$,
		otherwise abort with output $\perp$
		Evaluate $GC_{\hat{j}}$ using $\{A_{\hat{j},i,x[i]}\}_{i\in[n_1]}$ and
		$\{B_{\hat{j},i,y[i]}\}_{i\in[n_2]}$ to obtain $\{Z_i\}_{i\in[n_3]}$
		Then, $\forall i \in [n_3]$: If $Z_i = Z_{\hat{j},i,0}$, set $z[i] = 0$
		If $Z_i = Z_{\hat{j},i,1}$, set $z[i] = 1$,
		else abort with output $\perp$
		Output z

Figure 6: PVC secure protocol as proposed by [7]

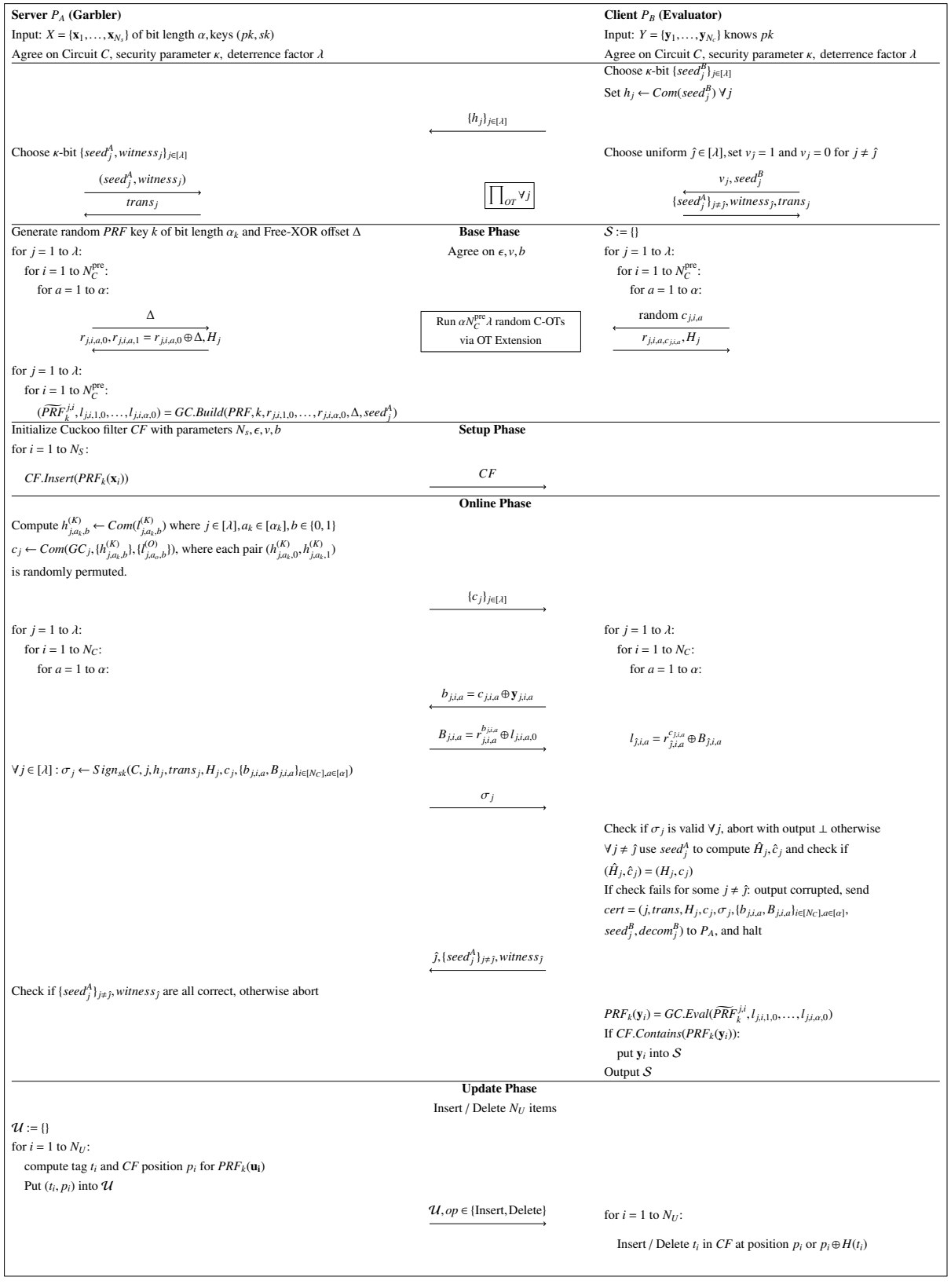


Figure 7: PSI with PVC