A Privacy-Preserving Platform for User-Centric Quantitative Benchmarking

Dominik Herrmann, Florian Scheuer, Philipp Feustel,
Thomas Nowey, and Hannes Federrath

University of Regensburg, 93040 Regensburg, Germany
http://www-sec.uni-regensburg.de/

Abstract. We propose a centralised platform for quantitative benchmarking of key performance indicators (KPI) among mutually distrustful organisations. Our platform offers users the opportunity to request an ad-hoc benchmarking for a specific KPI within a peer group of their choice. Architecture and protocol are designed to provide anonymity to its users and to hide the sensitive KPI values from other clients and the central server. To this end, we integrate user-centric peer group formation, exchangeable secure multi-party computation protocols, short-lived ephemeral key pairs as pseudonyms, and attribute certificates. We show by empirical evaluation of a prototype that the performance is acceptable for reasonably sized peer groups.

1 Introduction

During financial crisis banks have an increased need to take their bearings on the market. The comparison of own key performance indicators (KPI) to competitors' ones or to the overall baseline within the industry is a promising approach. Relevant KPIs may be for instance the proportion of subprimes in the asset portfolio or the return per worker in segment. Such benchmarks are traditionally calculated by a trusted third party (TTP) – which inevitably learns the KPI values in the process.

Benchmarking becomes non-trivial, though, when competitors are mutually distrustful or unwilling to hand over their KPI values to a TTP. Although the application of *secure multi-party computation* (SMC) protocols is a frequently cited solution, significant results can only be obtained if users can ensure comparability of their KPIs. An intuitive solution for this task is *user-controlled peer-group formation*, i. e. the initiator of a benchmarking may specify concrete selection criteria all participants have to meet (cf. Table 1).

To the best of our knowledge there is currently no practical solution with acceptable security properties for this challenge. In this paper we present a platform that allows privacy-preserving comparison of KPIs over

Table 1. Example of selection attributes, attribute values of an individual user i, and selection criteria used for peer-group formation of benchmarking b.

Selection attribute s_j	Attribute values $a_j^{(i)}$	Selection criteria $c_j^{(b)}$	a_j matches c_j
Location (country)	Germany	Germany	true
Business area	financial services	financial services	true
Sales in 1000 US$	517,000	$100 < x \leq 1000$	false

the Internet – not relying on a TTP at all. It integrates user-controlled peer group formation with exchangeable SMC protocols and offers its users practical anonymity. We discuss the requirements and the attacker model addressed by the platform, outline the architecture and protocols employed and evaluate the performance of a prototypical implementation.

2 Related Work

There have been a couple of initiatives to develop Internet-based benchmarking systems (cf. [1, 2]). However they neglect important privacy issues. Those issues can be tackled with SMC protocols. Numerous variants have been proposed in the last decades, among them generic ones [3–6] as well as protocols tailored for specific scenarios [7–9].

The work by Kerschbaum et al. [10–12] is especially focused on privacy-preserving benchmarking between (mutually distrustful) organisations. Their concept of peer-group formation using adapted k-means clustering is fundamentally different from our user-controlled approach, though. While Kerschbaum's proposal in [12] only allows users to be part of one peer group at a given time, we enable participation in various peer groups. Furthermore, our approach offers more privacy to the users, which may increase adoptability. Finally, while Kerschbaum et al. advocate an SMC protocol based on homomorphic encryption for their benchmarking platform, we additionally consider a secret sharing scheme from [3].

Catrina et al. [13] advocate the involvement of a central service provider that does not take part in benchmarking computations, but only offers communications services to the clients. Furthermore, they suggest that clients delegate work to multiple identical service providers to distribute trust among them and to reduce the risk of compromise.

3 Architecture Overview

Many SMC protocols (e. g. [3]) rely on fully meshed peer-to-peer communications. Catrina et al. [13] argue that this network topology is not

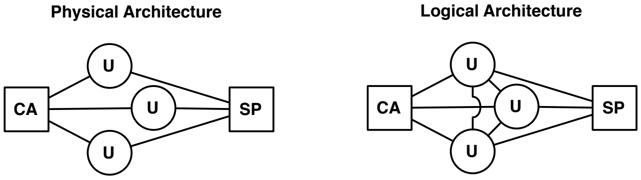

Fig. 1. Involved parties: users (U), service provider (SP), certification authority (CA)

suitable in a real-world setting, though. They advocate the involvement of a central service provider (SP) that reduces complexity, acting as a communication mediator for the clients (star topology). We have designed our platform following this reasoning. However, in contrast to their proposal to employ multiple identical service providers, we include a certification authority (CA) in our architecture. We believe that users are more likely to trust a dedicated CA than multiple (possibly relatively unknown) service providers (cf. Kerschbaum's argumentation in the introduction of [11]).

Figure 1 shows the physical and logical architecture of the platform. Clients do not communicate directly with each other, but only via the public SP. The SP stores incoming messages temporarily, and the clients poll the SP to retrieve new messages in regular intervals. For addressing a specific recipient, clients use public key certificates as pseudonyms. Moreover, the SP allows clients to request new benchmarkings or discover currently running ones as well as to retrieve the results of past ones. The CA issues certificates used for various purposes.

4 Requirements and Attacker Model

Table 2 contains the set of functional and non-functional requirements which are addressed by our privacy-preserving benchmarking platform.

Our attacker model is based on the definitions for multilateral security [14]. We differentiate between the group of *all users of the platform (users)* and the (usually smaller) group of *participating users (participants)* of an ongoing benchmarking. Following classical SMC research principles [15], we limit our attacker model to participants who are semi-honest ("honest but curious") for now: they follow the protocol and always tell the truth[1]. However, they may try to learn other participants' KPI values

[1] For sake of brevity we neglect free-riding and truth-telling issues in this paper. Refer to Ziv's model [16] for a detailed treatment of these issues. In future research, we will look into incentives that encourage truth-telling.

Table 2. The benchmarking platform addresses functional (F) and non-functional, i. e. security (S) and usability (U), requirements.

#	**Description**
F1	**User-driven Initiation.** Each user of the platform can initiate a benchmarking of a particular KPI whenever need arises.
F2	**User-controlled Peer Group Formation.** An accurate definition of the peer group is a prerequisite to gain useful insights from the benchmarking. The initiator must be able to control the peer-group formation process on a per-benchmarking level so that the participants meet certain criteria. The criteria should be specified with real-world attributes to facilitate intuitive classification.
F3	**Support for Various Statistics.** The platform must support the calculation of various statistics (e. g. mean, median and variance) for different KPI data types. Apart from integer and real values, benchmarking of categorical values may also be of interest.
S1	**Anonymity of Users.** The identity of the users must remain private. Other parties (including the platform provider) may neither learn who is participating in a benchmarking nor whether a particular organisation is participating.
S2	**Confidentiality of KPIs.** As KPI values may contain sensitive information about a user's organisation, they must not be revealed to third parties (including the platform provider). The platform has to ensure that the benchmarking results are aggregated in a fashion that prohibits the recovery of individual KPI values.
S3	**Confidentiality of Selection Attributes.** If other parties learned the selection attributes of a participant, they could deduce the identity of a participant, either with context knowledge or by an intersection attack (cf. Sect. 5.2). Therefore, the values of selection attributes used during peer-group formation must not be revealed to other parties (including the service provider).
S4	**Enforced Peer Group Formation.** The platform has to enforce the peer-group formation as requested by the initiator. Users who do not meet the requested criteria may not join a benchmarking.
U1	**Off-the-Shelf Technologies.** The platform should utilise existing technologies and communications infrastructure, i. e. the Internet, to lower entry barriers for users and platform providers.
U2	**Polling** The platform should not require its users to run servers which require inbound connections. Instead, the client software should use polling to retrieve new messages in short intervals.
U3	**Near-time Availability of Results.** Benchmarking results have to be available within a short time to enable users to set up multiple consecutive benchmarkings over various KPIs and peer groups to gain information exploratively. Results should be made available to the other participants as an incentive to participate. They schould be archived and published on the platform.
U4	**Scalability** The system is supposed to scale with increasing numbers of participants and benchmarkings.

or identities. Multiple participants may collude to achieve this goal. We leave malicious participants, who inject or forge messages, for future work.

Furthermore, we limit ourselves to SPs and CAs who carry out their duties honestly. While the CA is considered trustworthy the SP may maliciously try to gain information about the benchmarkings. The last group we consider are outsiders who are not using the platform. They may either passively or actively try to gain knowledge about running benchmarkings, the identity of participants or their KPI values. To this end, they may record network traffic or even maliciously inject messages.

5 Addressing Core Requirements

5.1 Secure Multi-Party Computation of Mean Value

Our benchmarking platform is independent of the SMC protocols. For evaluation purposes we have implemented two popular SMC protocols prototypically (results are provided in Sect. 7). Both of them facilitate the distributed computation of the sum of multiple values. The arithmetic mean of the values is easily found by dividing the sum by the number of participants. Of course, more advanced protocols addressing other statistics can be implemented in the future.

SumSecureSplit. The first one is an extension to Atallah's Secure Split Protocol (cf. [3] for a detailed explanation of that building block). The Secure Split protocol allows a group of m participants P_j, $1 \leq j \leq m$, who provide private numeric inputs $x^{(j)}$, to securely share these values among themselves. As output of the protocol each participant obtains m blinded values $z^{(j)}$ such that $\sum_{j=1}^{m} x^{(j)} = \sum_{j=1}^{m} z^{(j)}$. In addition to Atallah's specification, our scheme requires each participant P_j to send his value $z^{(j)}$ to all the other participants, enabling each of them to compute the final result $z = \sum_{j=1}^{m} z^{(j)}$. The presented protocol uses a collusion threshold $k = m - 1$ for best privacy protection.

SumHomomorphic. The second protocol employs a homomorphic cryptosystem with an additive property $E_{\mathrm{hPub}}(\sum_{j=1}^{m} x^{(j)}) = \prod_{j=1}^{m} E(x^{(j)})$ [17]. Using a shared homomorphic key pair $K_{\mathrm{h}} = (K_{\mathrm{hPub}}, K_{\mathrm{hSec}})$ participants $P_j, 1 < j < m$ submit their encrypted private numeric inputs $E_{\mathrm{hPub}}(x^{(j)})$ to a party unaware of K_{hSec} (the SP), which will execute the multiplication and distribute the still encrypted result to the participants afterwards. Finally they gain the result $\sum_{j=1}^{m} x^{(j)}$ by decrypting the received result with K_{hSec}.

5.2 Preserving User's Anonymity

Users should connect to the server through an anonymisation service (like Tor or JonDonym)[2] to protect their IP addresses. On the platform, they are addressed with a pseudonym, which cannot be linked to their identity. Our protocol uses public key certificates as pseudonyms. However, this scheme is not sufficient: it is conceivable that the set of selection attribute values may uniquely identify a participant, i. e. when only one organisation exists in the world which matches those criteria. Our protocol ensures that neither the service provider nor other clients learn the selection attribute values.

If other participants can observe that a single client (identified by a static pseudonym) takes part in consecutive benchmarkings, they can mount an intersection attack by slightly varying the selection criteria and observing whether the targeted client takes part in the benchmarking or not. In order to protect clients from intersection attacks, clients use a new pseudonym *(ephemeral public key certificate)* for every benchmarking.

5.3 Enforcing Peer-Group Formation

Given our set of requirements, the SP cannot enforce peer-group formation on its own, because it must not learn the selection attribute values (cf. requirement S3). We assume that – as a trusted party – the CA may learn these values and is able to validate their correctness, though. Therefore, the client has to present the selection attribute values to the CA in order to receive an *attribute certificate* containing them.

In order to participate in a benchmarking a client has to forward the list of requested selection criteria together with its own attribute certificate to the CA. The CA checks whether the client's attributes match the requirement and creates a *participation certificate*, which includes the current pseudonym of the client and the satisfied selection criteria, on success. The client has to present the participation certificate as a means of authorisation to the SP. In order to maintain unlinkability (cf. Sect. 5.2), the client must request a fresh participation certificate for each benchmarking – even if it contains the same statement.

6 Protocol

The protocol proceeds in four phases. Figure 2 outlines the involvement of the various parties in each phase. We assume that all connections between

[2] cf. `http://www.torproject.org/` and `http://www.jondonym.de/`, respectively

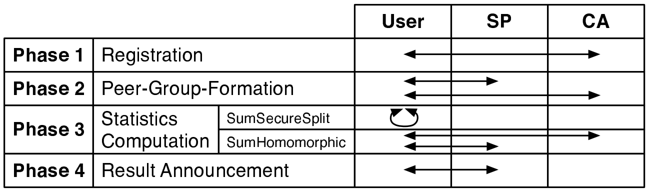

Fig. 2. Involvement of parties in the course of the protocol

clients, SP and CA are encrypted using TLS to protect against attacks from outsiders. For practical purposes the SP offers a predefined list of selection attributes S_{global} and KPIs X_{global} for various subject areas to choose from. The following sections outline the building blocks of the communication protocol.

Registration at the CA. In *phase 1* a user i creates a permanent key pair $K^{(i)}_{\text{perm}} = (k^{(i)}_{\text{pPub}}, k^{(i)}_{\text{pSec}})$. Then, he registers at the CA providing $k^{(i)}_{\text{pPub}}$, his real-world identity, and a list of j selection attributes $\mathbf{s}^{(i)}_{\text{client}} = (s_1, \ldots, s_j)$ with their corresponding attribute values $\mathbf{a}^{(i)}_{\text{client}} = (a^{(i)}_1, \ldots, a^{(i)}_j)$ which shall be used during peer-group formation. If the verification of the supplied data succeeds the CA will sign the user's long-lasting public key $k^{(i)}_{\text{pPub}}$ for future authentication. The user also receives an attribute certificate $\text{Cert}^{(i)}_{\text{attributes}} = (k^{(i)}_{\text{pPub}}, \mathbf{s}^{(i)}_{\text{client}}, \mathbf{a}^{(i)}_{\text{client}})$, which ties the validated selection attribute values $\mathbf{a}^{(i)}_{\text{client}}$ to his identity represented by $k^{(i)}_{\text{pPub}}$. Once the registration is completed, a user can either initiate a new benchmarking or participate in a pending one.

Creating Ephemeral Key Pairs. In *phases 2–3* users are identified by ephemeral public-key certificates. $K^{(i)}_{\text{perm}}$ cannot be used for communication with the SP or other clients in order to maintain unlinkability. Whenever a user i wants to set up or participate in a benchmarking, he creates a short-lived (ephemeral) key pair $K^{(i)}_{\text{eph}} = (k^{(i)}_{\text{ePub}}, k^{(i)}_{\text{eSec}})$, which is exclusively used for one benchmarking and cannot be linked to $K^{(i)}_{\text{perm}}$.

Benchmarking Setup. When a new benchmarking is set up *phase 2* is entered. A user i can issue an ad-hoc request for a new benchmarking b of a specific KPI $x \in X_{\text{global}}$ at any time. Therefore, he requests the CA to sign an ephemeral public key $k^{(i)}_{\text{ePub}}$ after having authenticated with $k^{(i)}_{\text{pPub}}$. Then, the user sends a benchmarking request $\text{BR} = (k^{(i,b)}_{\text{ePub}}, x, t, \mathbf{s}_{\text{req}}, \mathbf{c}_{\text{req}})$

to the SP. The BR includes a vector of selection attributes $\mathbf{s}_{\text{req}}$ and the requested criteria $\mathbf{c}_{\text{req}}$. The initiator is willing to wait for participants until deadline t. The SP validates the signature on $k_{\text{ePub}}^{(i)}$ and adds the request to the list of benchmarking announcements on success. All clients (including the initiator) can now indicate their interest to participate. When the deadline expires, the announcement will be removed from the list and the computation starts.

Joining a Benchmarking. If user i wants to participate in a pending benchmarking b from the list of announcements, he will create a new ephemeral key pair $K_{\text{eph}}^{(i)}$. Then, he presents $k_{\text{ePub}}^{(i)}$, $\text{Cert}_{\text{attributes}}^{(i)}$, $\mathbf{s}_{\text{req}}^{(b)}$ and $\mathbf{c}_{\text{req}}^{(b)}$ to the CA and requests it to sign $k_{\text{ePub}}^{(i)}$ and to issue a participation certificate $\text{Cert}_{\text{p}}^{(i,b)} = (k_{\text{ePub}}^{(i)}, \mathbf{s}_{\text{req}}^{(b)}, \mathbf{c}_{\text{req}}^{(b)})$. The CA will only comply, if $\mathbf{a}_{\text{client}}^{(i)}$ matches the criteria specified by $\mathbf{s}_{\text{req}}^{(b)}$ and $\mathbf{c}_{\text{req}}^{(b)}$. After that, the client sends a *participation request* $\text{PR} = (\text{Cert}_{\text{p}}^{(i,b)}, b)$ to the server. The server validates PR and checks that $\mathbf{s}_{\text{req}}^{(b)}$ and $\mathbf{c}_{\text{req}}^{(b)}$ contained in $\text{Cert}_{\text{p}}^{(i,b)}$ are identical to the corresponding fields in BR. On success it will add the signed $k_{\text{ePub}}^{(i)}$ to the list of participants. Once the deadline t is reached, the server checks whether the security requirements are met, i. e. a certain minimum number of clients are willing to participate. It proceeds by publishing the list of participants $\text{PL}^{(b)} = \mathbf{k}_{\text{ePub}}^{(\cdot)}$, which contains a list of their signed ephemeral public keys.

Participating in a Benchmarking. In *phase 3* the distributed computation takes place. All n participants of benchmarking b retrieve $\text{PL}^{(b)}$ from the server and validate the signatures of the contained $\mathbf{k}_{\text{ePub}}^{(\cdot)}$ to prevent the SP from maliciously injecting participants. On success they proceed with the actual SMC protocol, which is not shown here in detail for conciseness. Each message m exchanged between two clients is signed and encrypted using the ephemeral keys before transmission: $m' = \text{enc}_{k_{\text{ePub}}^{(j)}}(m, \text{sig}_{k_{\text{eSec}}^{(i)}}(m))$. When the SMC protocol terminates, all clients submit their final result values to the SP, which announces them publicly *(phase 4)*. The SP may use a majority vote in case the reported values are not identical.

7 Performance Evaluation

We implemented the platform prototypically in Java SE 5 on Linux. All messages were encoded in XML and transmitted over TLS-secured sockets.

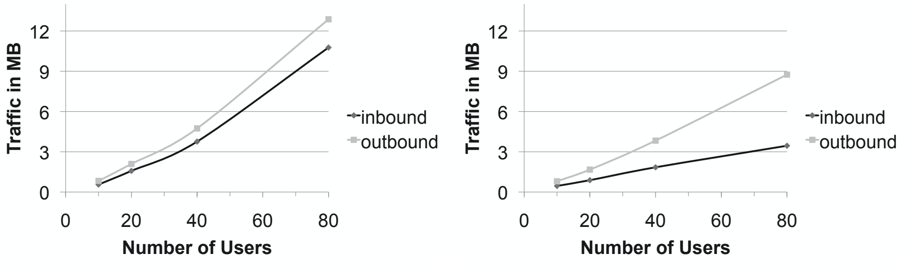

Fig. 3. Server-side traffic for *SumSecureSplit* (left) and *SumHomomorphic* (right)

Clients applied end-to-end encryption to all messages exchanged between each other using 1024 bit RSA keys. The JVMs of the CA and the SP were running on off-the-shelf hardware (2 cores, 2.20 GHz, 3 GB RAM), the clients were instantiated on a Dual Xeon 2.13 GHz machine with 2 GB of RAM and connected with 100 MBit/s.

As a comparison with existing SMC-based platforms [11, 8] is difficult due to differing assumptions, we limited our evaluation to the SMC protocols we implemented so far, *SumHomomorphic* and *SumSecureSplit*, for varying numbers of users (10, 20, 40, 80). In each run all users registered in the first 60 seconds. Afterwards 10 benchmarkings with 10 selection criteria each were executed in sequence. Each time users had 60 seconds to join the benchmarking and to form the peer group. Clients were configured to use a polling interval of 7 seconds. On average the statistics computation was finished within 8 seconds *(SumHomomorphic)* or 17 seconds *(SumSecureSplit),* respectively.

According to our results both SMC protocols offer satisfactory performance for peer groups of reasonable size. Figure 3 indicates that *SumSecureSplit* induces more traffic than *SumHomomorphic*. As illustrated in Fig. 4 the traffic for the one-time registration is insignificant − in stark contrast to the peer-group formation. While the proportion of the SMC protocol increases rapidly with the number of users for *SumSecureSplit*, this is not an issue for *SumHomomorphic*. This finding is not surprising as *SumHomomorphic* has been designed with a client-server architecture in mind and does not rely on extensive peer-to-peer communications [11].

The CPU load of both SMC protocols is noncritical as seen in Fig. 5. The servers are idle, waiting for requests most of the time. Considering the distributed computation of the *SumSecureSplit* protocol the load of the SMC component is surprisingly high, though. This can be ascribed to the extensive processing of client-client-messages at the server.

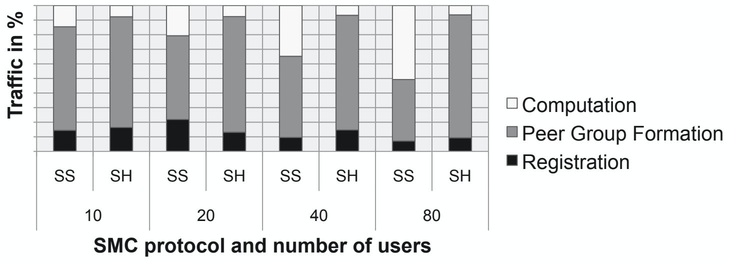

Fig. 4. Server-side traffic allocation for various protocol phases using *SumSecureSplit* (SS) and *SumHomomorphic* (SH)

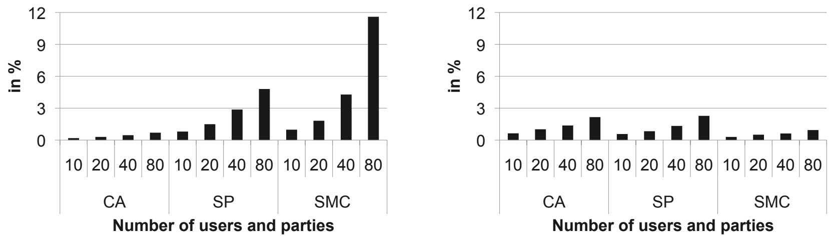

Fig. 5. Average CPU load for CA, SP and its SMC component for *SumSecureSplit* (left) and *SumHomomorphic* (right)

8 Conclusion and Future Work

We presented a centralised privacy-preserving platform for benchmarking of numeric values which is the first one to integrate intuitive, user-controlled peer-group formation with flexibly exchangeable SMC protocols. Users are free to initiate and participate in benchmarkings, while the employed SMC protocols ensure that sensitive KPI values are not disclosed at any time. Furthermore, we have demonstrated how ephemeral key pairs can be used to foil intersection attacks targeting users' anonymity. Another novel concept in our work concerns the architecture of the platform: in addition to a central service provider we deploy a dedicated CA, which ensures appropriate separation of duties. The evaluation of a prototypical implementation using off-the-shelf technologies yielded promising results. The calculation of the arithmetic mean of a set of secret KPI values with the help of two classical SMC protocols performs fast enough for practical purposes. Following up from here the modular concept of the platform allows us to easily assess the properties of many more SMC protocols addressing various other statistics in a common environment. Furthermore, the platform will allow us to study countermeasures against iterative intersection attacks, which exploit our peer-group formation scheme to identify certain participants with unique selection attribute values.

References

1. Bogetoft, P., Nielsen, K.: Internet Based Benchmarking. Technical report, Dept. of Economics, The Royal Veterinary and Agricultural University, Denmark (2002)
2. Crotts, J., Pan, B., Dimitry, C., Goldman, H.: A Case Study on Developing an Internet-Based Competitive Analysis and Benchmarking Tool for Hospitality Industry. In: Proc. Conference of Travel and Tourism Research Association. (2006)
3. Atallah, M., Bykova, M., Li, J., Frikken, K., Topkara, M.: Private collaborative forecasting and benchmarking. In: Proc. 2004 ACM workshop on Privacy in the Electronic Society, ACM (2004) 103–114
4. Yao, A.: Protocols for Secure Computations. In: Proc. 23rd annual IEEE Symposium on Foundations of Computer Science. (1982) 160–164
5. Malkhi, D., Nisan, N., Pinkas, B., Sella, Y.: Fairplay – a Secure Two-Party Computation System. In: Proc. 13th USENIX Security Symposium. (2004) 287–302
6. Liu, W., Luo, S.S., Chen, P.: A Study of Secure Multi-party Ranking Problem. In: 8th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing. (2007) 727–732
7. Bogetoft, P., Damgård, I., Jakobsen, T., Nielsen, K., Pagter, J., Toft, T.: A Practical Implementation of Secure Auctions Based on Multiparty Integer Computation. In: Financial Cryptography. (2006) 142–147
8. Bogetoft, P., Christensen, D.L., Damgard, I., Geisler, M., Jakobsen, T., Krøigaard, M., Nielsen, J.D., Nielsen, J.B., Nielsen, K., Pagter, J., Schwartzbach, M., Toft, T.: Multiparty Computation Goes Live. Cryptology ePrint Archive, Report 2008/068 (2008)
9. Feigenbaum, J., Pinkas, B., Ryger, R., Saint-Jean, F.: Secure Computation of Surveys. In: Proc. EU Workshop on Secure Multiparty Protocols. (2004)
10. Kerschbaum, F., Terzidis, O.: Filtering for Private Collaborative Benchmarking. In: Emerging Trends in Information and Communication Security. (2006) 409–422
11. Kerschbaum, F.: Practical Privacy-Preserving Benchmarking. In Jajodia, S., Samarati, P., Cimato, S., eds.: Proc. IFIP TC-11 23rd International Information Security Conference. Volume 278 of IFIP., Springer (2008) 17–31
12. Kerschbaum, F.: Building A Privacy-Preserving Benchmarking Enterprise System. Enterprise Information Systems **2**(4) (2008)
13. Catrina, O., Kerschbaum, F.: Fostering the Uptake of Secure Multiparty Computation in E-Commerce. In: Proc. Third International Conference on Availability, Reliability and Security. (2008) 693–700
14. Rannenberg, K., Pfitzmann, A., Müller, G. In: Multilateral Security in Communications. Volume 3. Addison-Wesley (1999) 21–29
15. Canetti, R., Lindell, Y., Ostrovsky, R., Sahai, A.: Universally Composable Two-Party and Multi-Party Secure Computation. In: Proc. 34th annual ACM symposium on Theory of Computing, ACM (2002) 494–503
16. Ziv, A.: Information Sharing in Oligopoly: the Truth-Telling Problem. RAND Journal of Economics **24**(3) (1993) 455–465
17. Paillier, P.: Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In: Advances in Cryptology – EUROCRYPT '99. (1999) 223–238