Role-Based Ecosystem for the Design, Development, and Deployment of Secure Multi-Party Data Analytics Applications

Andrei Lapets, Kinan Dak Albab, Rawane Issa, Lucy Qin, Mayank Varia, Azer Bestavros, Frederick Jansen
Boston University, 111 Cummington Mall, Boston, MA USA 02215
Email: {lapets, babman, ra1issa, lucyq, varia, best, fjansen}@bu.edu

***Abstract*—**

Software applications that employ secure multi-party computation (MPC) can empower individuals and organizations to benefit from privacy-preserving data analyses when data sharing is encumbered by confidentiality concerns, legal constraints, or corporate policies. MPC is already being incorporated into software solutions in some domains; however, individual use cases do not fully convey the variety, extent, and complexity of the opportunities of MPC. This position paper articulates a *role-based* perspective that can provide some insight into how future research directions, infrastructure development and evaluation approaches, and deployment practices for MPC may evolve.

Drawing on our own lessons from existing real-world deployments and the fundamental characteristics of MPC that make it a compelling technology, we propose a role-based conceptual framework for describing MPC deployment scenarios. Our framework acknowledges and leverages a novel assortment of roles that emerge from the fundamental ways in which MPC protocols support federation of functionalities and responsibilities. Defining these roles using the new opportunities for federation that MPC enables in turn can help identify and organize the capabilities, concerns, incentives, and trade-offs that affect the entities (software engineers, government regulators, corporate executives, end-users, and others) that participate in an MPC deployment scenario. This framework can not only guide the development of an ecosystem of modular and composable MPC tools, but can make explicit some of the opportunities that researchers and software engineers (and any organizations they form) have to *differentiate* and *specialize* the artifacts and services they choose to design, develop, and deploy. We demonstrate how this framework can be used to describe existing MPC deployment scenarios, how new opportunities in a scenario can be observed by disentangling roles inhabited by the involved parties, and how this can motivate the development of MPC libraries and software tools that specialize *not by application domain* but *by role*.

I. Introduction

Individuals and organizations face a tension between harnessing the power of data and the liability associated with the possession or potential exposure of that data, leading to lost opportunities in data analysis and the consequent benefits thereof. Software systems and applications that employ secure multi-party computation (MPC), a cryptographic technique that allows independent parties to jointly compute a shared result without revealing their private inputs to the parties performing the computation, can empower individuals and organizations to benefit from privacy-preserving computations over data in contexts where data sharing is limited by confidentiality concerns, legal constraints, or corporate policies.

In the near future, academic and industry organizations will begin drawing from a rich body of existing work on MPC to develop and implement novel software tools, systems, and platforms. These will in turn enable individuals, researchers, businesses, government agencies, and policymakers to use modern frameworks and development environments to build and deploy applications that allow cooperating parties to compute analytics over data belonging to multiple entities without the expense, liability, or privacy loss associated with contributors submitting that data to a single trusted entity. Real-world applications enabled by such an infrastructure can empower their users by allowing for new forms of collaborative data analysis.

MPC has been an active area of cryptography research for more than 35 years [1]–[4]. The past decade has seen intense focus on improving MPC algorithms and designing MPC frameworks that perform quickly enough on domain-specific functions likely to be of interest in practice [5]–[9]. Due to these efforts, MPC is starting to become a mature technology that is ripe for transition to practice in appropriate application domains: real-world deployments of MPC are accumulating, and it is clear that domain-specific solutions can have the requisite performance and usability properties that drive (or at least do not hinder) adoption of MPC. However, we can envision that at some inflection point, researchers, practitioners, organizations, and entrepreneurs will see opportunities to create more general solutions.

The opportunities and benefits associated with the use of MPC techniques can be subtle, highly sensitive to context, unfamiliar, and counterintuitive to the full range of stakeholders: software engineers, government regulators, corporate executives, end-users, and others. While acceptable performance within the target domain is a necessary requirement for adoption of MPC, it is by no means sufficient when considering the broader deployment context [10], [11]. To address the concerns of all stakeholders, performance, scalability, and modularity of techniques and functionalities can ensure the *means*, but all key players must also have the *incentive* and *capability* to use these techniques. Basic research alone cannot (and should not, given the potentially limited resources of research communities) address all of these challenges.

If other individuals and organizations wish to contribute to the development of MPC (and if those in the research

community wish to support and encourage them to do so), how can they motivate and organize their contributions? Modularity through domain specialization provides one natural answer, but this may lead to lost opportunities for reuse of knowledge, reuse of code, and exploitation of heterogeneities and asymmetries among stakeholders that occur in multiple domains. Such an approach may also overlook opportunities and incentives for those who wish to (inevitably) build and contribute general-purpose solutions that cut across domains. Through our deployment efforts, we have observed an orthogonal way to introduce modularity: via specialization of *roles* that entities can inhabit within MPC deployment scenarios.

Our explicit focus on roles may in some ways look familiar to practitioners within the broader cybersecurity domain [12]. However, the manner in which roles are incorporated into our proposed framework differs from past work: the primary reason that an entity inhabits a role in our framework is to identify *how it enables the federation (via MPC) of some functionality or responsibility* (and not, unless by coincidence or implication, to identify its capabilities or responsibilities with relation to disclosure or integrity). At a higher level, the purpose of introducing roles is also distinct from that of previous work: we argue that our role-based approach provides a *novel way to explicitly identify, articulate, and motivate future directions* for research and development efforts that incorporate MPC technologies and software solutions.

Secure MPC as an Accessible Service: Compelling real-world uses of MPC are numerous; applications include analysis of pay inequities [10], disease surveillance [13], satellite collision prevention [14], tax fraud detection [15], electricity trading markets [16], scientific discovery [17], genomics [18], and global advanced persistent threat identification in corporate network data [19].

Despite a small number of successful deployments of MPC for social good in areas such as tax fraud detection [15], the impact of MPC has mostly remained limited to proof-of-concept studies. Adoption of MPC is in part hindered by the fact that the existing frameworks are engineered to optimize for metrics that are not sufficient on their own to drive adoption in practice. Many efforts compete on computational performance for small-scale data (a domain in which all modern frameworks are adequate [20]) while not addressing human-scale performance costs: tools require users to become familiar with new domain-specific languages, to set up compatible hardware infrastructures, or to understand and appreciate the benefits of counterintuitive trust relationships. These circumstances can be burdensome for software engineers, administrators, and non-technical stakeholders.

Lessons learned from experiences in creating and successfully deploying MPC solutions for concrete applications reflect an emphasis on a *domain-specific* solutions for a *particular deployment scenario*; as expressed by others in the community [21]: "Secure computation is a general scheme; in reality one has to choose an application, starting from a very real business need, and build the solution from the problem itself choosing the right tools, tuning protocol ideas into a reasonable solution, balancing security and privacy needs vs. other constraints: legal, system setting, etc." We also agree with the assertion [21] that only after a solution addresses a concrete need is it appropriate to "Understand, employ, and generalize useful routines: Building more general routines and secure computation software packages of actual business value may, therefore, be a result of collecting various examples of actual useful deployments first, and then creating a common API/software packages based on actual use and experience, in a bottom up fashion." But in what ways is it possible to generalize solutions? Once numerous solutions exist, how do we expect the MPC landscape to develop into the future? With a desire to extrapolate from these sentiments and motivated by an examination of successful deployments, we propose a model that will help us (and others) envision and proceed into this next stage, as well as to replicate such successful sequences for new use cases and new concrete needs.

Our Vision: This position paper proposes a vision that consists of a conceptual framework for analyzing and exploiting the various heterogeneities and asymmetries in roles, concerns, incentives, and resources/capabilities of parties participating in MPC. We believe that MPC frameworks and tools can allow cryptographers and software engineers to consciously and deliberately encode these various roles within complex MPC software systems. Such tools can then be explicitly designed to assist the different parties (with varying technical capabilities) in designing, developing, and deploying these systems. Our proposed conceptual framework is informed by–and contextualizes–a number of criteria that must be satisfied to enable a successful MPC deployment. This is due to the fact that our vision is informed and motivated by our own experiences in overcoming obstacles to adoption and deployment while building MPC applications [10], [11], [22]–[24], by the identification of new applications for MPC via the disentangling of roles [25], by insights from deployment efforts undertaken by other groups, and by the rich variety of approaches to secure MPC found in related research efforts.

II. Existing Approaches in Related Work

This position paper draws on prior work to advocate that developers of MPC libraries, systems, and applications targeting real-world scenarios and practical settings should adopt a particular design perspective. We briefly review related efforts in developing MPC frameworks, deploying MPC applications in the real world, and addressing usability and deployability of applications that utilize MPC.

The past few years have seen several successful deployments of MPC [26]–[28]. Further, an array of software frameworks is available. These range from proprietary implementations [7], [29] to open-source, proof-of-concept work [6], [8], [9], [30]–[34]. Available frameworks vary not only in software maturity, security guarantees, and APIs but also in the roles that parties must inhabit to participate, the requirements that can be satisfied, and the communication topology of the participating parties. We note that these frameworks are often designed with a specific setting in mind, and make a variety

of implicit assumptions about the concerns, incentives, and capabilities of participating parties. This leads to restrictions on the various roles that a party is allowed to inhabit, or the interactions it is allowed to perform, which may diverge from many interesting real world settings. These restrictions also apply to many recent highly-engineered MPC protocols [35]–[38] that exploit properties of the specific setting for which they are designed to increase efficiency or achieve desired properties.

Most frameworks do not clearly distinguish between compute parties, data analysts, data contributors, and policy experts. However, two predominant communication models can be identified: (1) *peer-to-peer* frameworks in which all participating parties are connected to each other and are required to install the MPC framework locally on their own computing infrastructure to participate, and (2) *client-server* frameworks in which there is a central server running an installation of the MPC framework while the other parties are mutually unreachable, web-based clients. This distinction most closely relates to the features of our proposed framework and is of particular interest with regard to accessibility of deployed MPC solutions. We examine some of the frameworks available in each category and highlight some feature-enhancing opportunities and adaptation challenges to be addressed in incorporating these frameworks into the proposed ecosystem.

Peer-to-peer: The majority of existing MPC frameworks follow the peer-to-peer communication model. Among these, most frameworks are open-source research prototypes [6], [8], [9], [30]–[34]. In many cases, the framework's authors explicitly discourage use in production [6], [8], [9], [31], [34]; furthermore, in these frameworks all participating parties, including contributors, must deploy the software framework on mutually available servers that remain online for the duration of the protocol execution [30], [32], [33], [39]. The proprietary frameworks [7], [29] are closed-source, which does not allow for auditing by the public (auditability is one of the criteria defined in section IV-A for successful MPC deployments). It is important to note that there is a variation of the peer-to-peer model [7] that enhances usability by distinguishing between the role of a contributor and service provider. In this model contributors secret-share their input data among multiple service providers via the browser. The service providers then jointly compute the required analytics over the secret-shared data.

Client-server: In the client-server model one entity effectively acts as a service provider while the analyst and contributors can be viewed as resource-constrained clients. Unfortunately, this setting has received far less attention in the community and available frameworks are sparse [40], [41]. The work by Schröpfer et al. [40] is a web-based implementation of Yao's garbled circuit scheme that allows two browser clients to perform a secure two-party computation leveraging a web server for communication and code delivery. The framework is closed-source and restricted to two parties. Canon-MPC [41] offers a web-based system that supports MPC with symmetric binary functions. Each participant registers an account on the system, can start a session by choosing a symmetric binary function, and can indicate which other registered users are contributors. The contributors evaluate the agreed-upon function with the aid of a service provider. Each contributor interacts with the service provider exactly once, simultaneously submitting data and performing part of the computation. At the end of the computation the contributors receive the outcome of the function. While Canon-MPC addresses many of the shortcomings of the traditional peer-to-peer model, it falls short of meeting several usability requirements that we will introduce in Section IV: auditability (the cryptographic library is delivered as a compiled binary and runs in Google's proprietary NativeClient), accessibility (only Google Chrome is supported because of the reliance on NativeClient; Google has also indicated it ceased development on NativeClient [42]), asynchronicity (while participants do not have to enter data in any particular order, no two participants can access the system at the same time; this does not scale past a few participants), and idempotence (data cannot be corrected after submission). Furthermore, we remark that if not all contributors submit their data, a second pass is required to finish the session. Importantly, Canon-MPC does not distinguish between the contributor and service provider roles: any party that contributes data must also actively participate in the computation. In terms of usability this is, in a sense, a step backwards from the model adopted in peer-to-peer frameworks such as Sharemind, which do distinguish between contributors and service providers. Minimizing the participation effort for contributors and increasing the robustness of the platform to contributors failing or neglecting to participate can be crucial as it addresses the fact that, in some deployment scenarios, contributors only have weak incentive to contribute data. At the same time, it may be natural to expect more active and reliable involvement from the analysts since these parties directly benefit from a successful completion of the computation.

The underlying implicit assumptions in these frameworks motivate our discussion of the role-based framework in the next section, where these assumptions are made explicit and configurable by the various participating parties. An MPC system should allow solutions to reflect and utilize these asymmetries in a deployment scenario, and to assist in choosing the optimal primitives and protocols used in the system [43].

III. Role-Based Framework

Transitioning MPC techniques to practice in a way that drives adoption necessitates a multi-faceted approach that begins well before software engineering efforts commence. MPC's social benefits cannot be realized unless the decision makers that ultimately choose to adopt MPC (*i.e.*, executives, directors, and legal advisors) have a clear and confident understanding of exactly what role they (and other entities) play in the process, as well as how MPC protects their sensitive data and mathematically guarantees compliance with data sharing restrictions. Once a solution is accepted, it should ideally be easily and rapidly deployable at little or no cost, and should

not necessitate changes to existing infrastructures and internal data-management processes (*e.g.*, the various schemas that participants use internally). Finally, it should in all aspects support the needs of the users who interact with the solution.

One way to introduce MPC capabilities into practical scenarios in a manageable way is to separate existing functionalities into distinct, modular, reusable, and composable components. These components should individually satisfy concrete needs identified by early adopters of MPC facing real-world use cases. These can then be generalized through incorporation of alternative and complementary techniques and software systems found in the literature. Finally, they can be applied in the construction and deployment of novel MPC-based solutions.

When examining a deployment scenario, it is critical to evaluate the roles that various parties take on, what their concerns are, what incentives exists for each party to participate, and their capabilities (as enumerated in Table I).

Roles	Concerns	Incentives	Resources
Decision Maker	Cost	Legal Compliance	Web Browser
Policy Expert	Security	Business Need	Web Server
Data Contributor	Privacy	Data Access	Database
Programmer	Liability	Compensation	Cloud Service
Data Analyst	Reputation		CDN
Recipient	Usability		
Storage Service	Participation		
Compute Service	Correctness		
Code Auditor			
Code Distributor			

TABLE I
ROLES AND DIMENSIONS OF INTEREST TO PARTICIPATING PARTIES

A. Roles.

An effective infrastructure should have functionalities subdivided into distinct roles (see Figure 1) that might be inhabited by different agents (such as servers, mobile devices, desktops, human users, organizations, and so on) within a broader context. In a given MPC solution, each agent can inhabit multiple roles. We advocate for a clean separation of the duties and functionalities that may be required for an MPC application into roles that are distinct, modular, and composable. Together, we can view all these as being supported by an *ecosystem* of interdependent and interacting software (and potentially hardware) solutions.

The design of ecosystem components (including libraries and applications) can be organized around these roles, and should not predetermine which roles the agents (*i.e.*, servers, devices, application instances, and human users) must inhabit. Ultimately, functionalities can then be exposed via a framework and API as roles that are coupled with information about which agents can inhabit those roles and which needs they satisfy. We enumerate *some* of the roles we have identified (acknowledging that there may be others) and note in a scenario each agent may inhabit more than one role simultaneously.

- A *decision maker* makes informed decisions on behalf of organizations and businesses and in return increases the willingness of participants to contribute their sensitive data. Decision makers must first gain confidence in the technologies proposed, appreciate that it would impose no significant burdens on their staff and infrastructure, and be assured that features such as idempotence and asynchrony would make deployment logistically feasible and likely to produce meaningful results.
- A *policy expert* can provide expertise on how to specify appropriate policies over the data schema (*e.g.*, given legal restrictions, best practices, or constraints chosen by the data contributors). The policy expert may have the expertise to participate in the development of an application, or may have no such expertise and would require an accessible way to express policies on contributed data that the application would then enforce.
- A *data contributor* is the user (an individual or an organization) of an application that supplies the sensitive data to be analyzed. Note that the data contributor may not know in advance the particular analyses that others may want to apply to the contributed data (in other words, any policies the data contributor may be able to specify may be *analysis-agnostic* at the time the data is being contributed). Note also that a contributor may not have the sophistication necessary to deploy virtual machines or servers when participating in a protocol (for example, it may be an individual using a mobile device or a web browser).
- A *data analyst* (often the same as the result *recipient*) specifies the analytics to be computed on data. This is normally done by an analyst with full knowledge of the schema of the data, but not necessarily with advance knowledge about who owns the data or what data sharing constraints apply to that data (in other words, the data analyst's algorithm specification may be *policy-agnostic*).
- The *recipient* is the party that receives the result of computing the secure multi-party analytic. Only recipients can benefit from the result of the computation, but different parties may receive different kinds of information (*e.g.*, an individual user might only have access to information about how their own metrics stack up against other users, while a service provider might see aggregate metrics across all users). As before, the recipient may not have sophisticated hardware, software, or technical expertise.
- A *compute service* has the capacity to deploy and maintain computational resources to allow analytics to be computed (*e.g.*, on data that may already be secret-shared). Once again, for simple applications, the service provider may only have a mobile device or web browser to contribute; in more complex scenarios, the service provider maybe be a large organization or even an entire cloud provider. Such a service provider may also act as a proxy or aid for data contributors or analysts that have limited resources [44].
- A *code auditor* verifies the correctness of the code to be

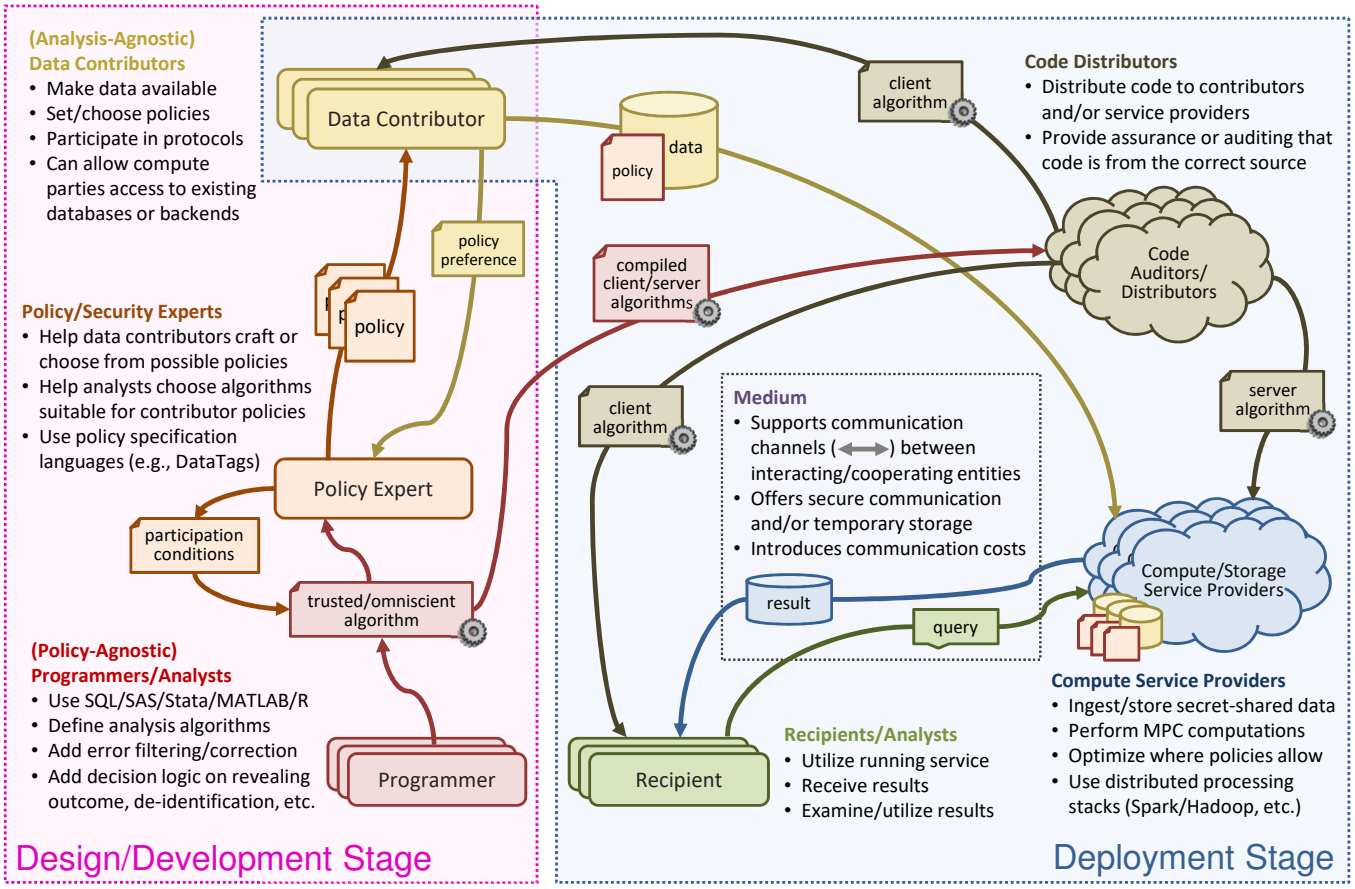

Fig. 1. An instance of the ecosystem that includes a collection of entities, the roles they inhabit, and the workflows that exist among them; a distinction is made between the design/development stage during which entities agree on the solution, and the deployment stage during which it is executed.

executed by compute services and data contributors, and must participate in additional mechanisms for ensuring integrity of delivered code.

- A *code distributor* makes available the actual application (or the specification of the particular computation) to be executed by compute parties on data provided by the data contributors. Since the integrity of the computation (and its conformance to the policy specialist's constraints) relies on the trustworthiness of the delivery mechanism, it should be possible to federate trust among code distributors [45] by allowing multiple entities to inhabit this role.

B. Concerns.

Within the ecosystem, entities have varying concerns that must be accounted for from design to deployment. These concerns may span between roles, be unique to a single entity, and change depending on the use case. Properly addressing them can reduce the barrier to participation and enhance trust that each entity is carrying out their role properly. The disentangling of roles may also create new opportunities to exploit trade-offs between competing concerns (*e.g.*, data contributors with limited computing resources need not be computing parties), or to turn concerns into incentives by taking advantage of market competition (*e.g.*, an organization with a lower liability profile than others may choose to take on additional liability and enhance the performance of a computation by operating on data in the clear).

- *Cost* in terms of efficiency, deployment expenses, and efforts required by users and analysts to learn and use the application (including any schema transformation or internal data collection and curation). Various trade-offs in costs can be tailored to account for different kinds of use cases.
- *Security* concerns may disincentivize participation if no guarantee is made to certify that only users with the correct privileges may have access to different pieces of computational data and to ensure that no malicious user may influence the result of the computation.
- *Privacy* may be vital in the scenario and is the key criteria that may call for its use in a scenario.
- *Usability* dictates that a successful deployment should not assume that data contributors and data analysts have any knowledge of MPC. Therefore usability needs to be emphasized in all design choices to lower the barrier to participation as much as is possible. For MPC to be widely adopted, the platforms developed need to be

intuitively understood by users and mimic workflows that users are already familiar with. Facilitating the user through simple contribution processes and proactive error-handling can also help ensure correctness.
- *Correctness* is critical for the legitimacy of the application and a shared concern among many parties as it may impact the reputation of various entities. It is also critical for the data analyst, in particular, for the integrity of the a deployment.
- *Participation* from entities in the ecosystem is critical for any well-designed architecture. Each entity must fulfill their respective roles during deployment. Even though clear instructions are made, data contributors may drop out of participating while the application is deployed. Managing incentives is crucial to maintaining a meaningful level of participation, thus enhancing the privacy and integrity of the final results.
- *Liability* concerns impede participation if the security guarantees of MPC are not well understood by all. Compute services may be hesitant to be a conduit for data if there are potential legal risks.
- The *reputation* of various entities may be staked to the success of an MPC deployment. These concerns can drive incentives and also prevent against malicious behavior such as colluding with other parties to reconstruct data. It may also influence participation and disincentivize data contributors who are concerned about the impact to their reputation based on the results. Choices should be made with the consideration of what is at stake and for whom to navigate the incentives around proper role fulfillment.

C. Incentives.

Different parties often have several different incentives to participate in an MPC computation, these incentives are often closely related to the party's role, and may provide insight into the expectations of the party, and what it can be trusted to do (*e.g.*, which parties it is most trusted not to collude with). Our ecosystem can leverage the asymmetry in these incentives to encourage participation, increase trust in the system, and improve efficiency. The ecosystem can perform as a *marketplace*, creating new incentives and magnifying existing ones. We discuss this in greater details in section IV-D.

D. Resources and Capabilities.

Capabilities will vary for each use case but the core resources that may be needed include web browser, web server, database, cloud service, and content distribution network (CDN). It is worthwhile to determine which entities can provide each of these resources while in the design phase.

IV. Benefits of the Roles Framework in Practice

In our work developing and deploying MPC solutions, there has been a need to satisfy a number of concrete criteria that are derived from the constraints imposed by target users (including their level of technical literacy, their access to appropriate kinds of IT infrastructure, and their logistical constraints). We note that some of these needs may need to be met in order to satisfy more general, overarching goals or to provide essential incentives (such as improving usability to drive adoption by a greater number of participants).

A. Criteria For Successful MPC Deployment.

We enumerate a number of these criteria, explicitly referencing how they relate back to the roles described in Section III.

- *Comprehensibility*: To drive initial adoption, either the MPC protocols themselves or the novel secure computing opportunities they introduce must be straightforward enough to explain so that decision-makers and users who might not possess technical expertise can be confident that they understand their operation or, at least, their security guarantees. This is most important for data contributors and policy experts, who are responsible for deciding whether participation in a protocol is appropriate, desirable, and safe.
- *Auditability*: To inspire trust, applications must have complete transparency, with open-source code and/or support for outside auditing. In most cases, those most closely involved with a particular application (analysts, contributors, and policy experts) will not possess the resources to inspect source code or perform auditing over the entire application stack. However, this presents an opportunity for those in the code distributor role (*e.g.*, cloud distribution networks may choose to offer this as a service coupled with application delivery).
- *Accessibility*: To minimize any hurdles that might discourage participation by data contributors and data analysts, solutions must be easily and rapidly deployable, requiring no setup, specialized software, specialized hardware, or public Internet addresses for agents behind firewalls. The way that compute service providers, code distributors, brokers, and the communications medium are incorporated can all have an effect on an application's accessibility.
- *Simplicity*: The software must be usable within a relatively narrow time window by non-expert human contributors and data analysts whose technical expertise may only include basic familiarity with common software applications.
- *Asynchronicity*: Agents only need to be online while actively performing relevant tasks (*i.e.*, not throughout the duration of a protocol's operation). This would primarily apply to data analysts and data contributors, and may be enabled through the inclusion of compute service providers.
- *Idempotence*: Contributors must be able to resubmit (*i.e.*, update) their data if they discover the data they submitted was corrupted (either through human error, through a software application failure, or both).
- *Non-commitment*: Contributors may decide not to participate, or may be unable to participate due to technical or logistical issues. The protocol should not require

advance knowledge of which parties will participate or their quantity.

- *Robustness*: Incorrect or malformed data from even one contributor destroys the value of aggregate analytics. Hence, interfaces must proactively warn users about spurious data. Furthermore, data analysts should have some assistance in assembling algorithms that are robust to outliers or common errors (one natural source of such expertise might be the security experts, though it could come from a distinct entity or toolchain).

Many of the existing protocols and tools discussed in Section II can be wrapped, adapted, adopted, or translated into a common infrastructure (using a shared modern language and platform such as JavaScript and Node.js) that arranges them into role variants that satisfy different combinations of the requirements enumerated above (and that can be inhabited by different agents). However, substantial software engineering efforts may be required to do so; in many cases a framework's authors explicitly discourage direct use of their tools in production [6], [8], [9], [31], [34].

B. Accessibility and Composability.

With regard in particular to accessibility: we believe that there is no "one size fits all" accessibility measurement. Different agents in the ecosystem have different usability concerns. For example, different potential participants "speak" different (programming) languages: distributed systems engineers on the cloud speak Spark, data analysts speak R, and lawyers and privacy experts speak their own less-structured languages. As a result, one of the most important accessibility metrics is the participants' ability not to be burdened by understanding the actions and responsibilities of roles they do not inhabit.

Consequently, the viability and success of such an ecosystem depends on the secure distributed applications being *composable* at the software and algorithmic layers. Composability allows each participant to use and build upon prior contributions while only learning succinct API-level specifications of functionality and security. It also facilitates upgrading: if one component is improved, all derived software packages gain its benefits too.

As we mentioned in Section III, policy agnostic programming [46]–[48] can provide composability at the software layer. At the algorithmic layer, one way to provide separation of responsibilities is via *universal composability* (UC) [49]. UC has been specialized and simplified for the MPC setting [50], and the potential value of UC has been shown throughout the computing stack [51]–[53].

C. Assisted Design.

Our proposed ecosystem consists of several frameworks and tools that facilitate and assist in the design, analysis, and deployment of MPC systems. In particular, we envision a set of tools that utilize information about the roles provided by the system designers and various participating parties, to generate parts of the system, analyses the efficiency of its protocols, and manage and deploy its components.

Given a component or a protocol of an MPC system, as well as auxiliary information about the roles and capabilities of the different parties, a static analysis tool may calculate an estimated performance evaluation, which may be used to determine the different trade-offs in costs in various settings. The system may make black-box usage of some primitive or functionality, this functionality may be achieved using a variety of protocols (*e.g.*, PRF protocols [54], protocols for generating multiplication triplets in Preprocessing [55], each protocol relies on different assumptions or models (e.g. preprocessing model), and the computation or communication complexity of these protocol may grow differently on different set of parameters (e.g. number of bits, number of parties). The performance evaluation tool can produce various metrics and charts (e.g. number of messages as a function of the number of parties or size of the input) that help the designers choose the optimal underlying protocols.

Additionally, the ecosystem can contain a knowledge base containing a variety of popular protocols for generic primitives or functionality (*e.g.*, comparisons, encryption, sorting, and so on). The knowledge base encodes the various properties and assumptions of these protocols. An automated tool may search the knowledge base to find suitable protocols given information about the roles of the parties and their privacy policies. The performance evaluation tool can be utilized to automatically determine which implementation of the primitives to use, or to reduce the number of choices the designer has to manually analyze. There has been recent advances in this direction [43], where a large knowledge base spanning over 180 papers from the literature was compiled in addition to several metrics for comparing MPC protocols.

This separation introduces the possibility of synthesizing potential compatible policies from analysis algorithm definitions [55] or interviews of data contributors [56]. The policy expert could then take on the task of evaluating or curating such policies. The policies can be encoded in a format that the framework understands, as well as a human-friendly format which the system displays to the data contributors before they submit their data.

D. Incentives: the Ecosystem as a Marketplace.

The population of the role-based ecosystem with appropriate, scalable, and usable MPC systems, applications, tools, and libraries provides an opportunity to explicitly address and leverage the *incentives* that may drive the agents that inhabit various roles. This naturally suggests the notion of a *marketplace*, and such a framing mechanism can further motivate, delineate, and constrain development efforts.

The major challenges of populating the marketplace, in our opinion, lie less with the design of faster MPC algorithmic building blocks but with developing developer- and user-centric software packages that span the range of possible security, usability, and performance trade-offs. Interfacing with existing languages and platforms already in widespread use within the community allows use of existing code distribution

infrastructures and enables a broader population of users to participate immediately.

We use the term *marketplace* because a successful ecosystem must provide rational incentives for parties to interact. The multiplier effect of composition is not merely a benefit to software engineers but an incentive to deliver modular code in the first place (a principle that is well-known to all software engineers but is sorely lacking in the security domain). The marketplace must also incentivize data owners, who have not yet contributed data, to make their data available for secure analyses (*e.g.*, by giving responses quicker [57] or more accurately to those who provide more data). Additionally, data owners can secret-share their data with a custom privacy policy, so that many computations may utilize their data in the future, given that the stated policy of the computation respects the policy of the data owners. Data owners may be compensated for providing valuable data (monetarily or otherwise), and the value of the data can be determined under MPC without compromising its confidentiality. Next, by being open to (and in some cases requiring) multiple compute service providers [58], our vision reduces the current cloud computing incentives toward vendor lock-in and instead favors compute service providers who compete and specialize in areas like trusted client application code delivery, untrusted high-performance computing, data analysis, policy synthesis, and reliable data delivery. Finally, a marketplace allows for new types of actors to emerge who *broker* the exchange of information or offer suggested advice on privacy policies governing this exchange. This has the effect of further reducing barriers to entry: data contributors and analysts may not need to work as hard to understand the details of MPC in order to receive its benefits (as others may be incentivized to help them understand or to shield them from having to do so). Finally, we observe that such a marketplace model naturally points to *mechanism design* [59], [60] as an important area of MPC research–one that can lead to opportunities to exploit the benefits of secure computing for maximal social good.

V. Use Cases in the Ecosystem Context

The recommendations in this report are in part informed by planning, development, and deployment efforts on a number of real-world applications (including each of our own MPC deployment experiences [10], [22], [24] and development efforts [61]) with a need for accessible, secure analytics solutions that are critical to their missions. We briefly discuss three use cases within the context of the proposed role-based ecosystem.

A. MPC For Tax Fraud Detection

In 2013, estimates quantified Estonia's losses over VAT evasion at 220 millions euros a year. Companies offered resistance against legislation that required them to declare their purchase and sales invoices to tax authorities, both due to the sensitive nature of this data and due to the burden it places on companies to curate the data. An MPC system for detecting such evasion was proposed as a way to protect companies' confidentiality [15]. Companies use the system to secret share their VAT declarations, which are then analyzed under MPC to calculate risk scores associated with each company. The results are then revealed to the Estonian Tax and Customs Board (MTA). The system was presented to decision makers at the MTA, who understood the value of utilizing MPC in this application and considered using MPC for future applications. However, the MPC system was not adopted.

This use case offers a variety of interesting insights into the importance of the role-based ecosystem and its benefits. The companies with VAT declarations are the data contributors in this system; they secret share their declarations to the compute parties, who then execute the MPC analysis. The companies do not participate in the analysis and are not required to operate a sophisticated computing stack or powerful hardware; this is consistent with their concerns about the burden of maintaining or acquiring appropriate resources and expertise. The two main compute parties between whom trust is federated are the MTA and the Estonian Trade Association (ETA). These two parties can be trusted not to collude because their respective incentives are entirely different: the MTA is a government agency concerned with detecting tax fraud, while the ETA is a representative body of the companies (data contributors) that is concerned with protecting their privacy and interests. The Information Systems and Registers Center (under the Estonian Justice Department) is the third compute party. Although the third party is also trusted not to collude with the others due to incentives and various organizational independence requirements, the authors explicitly stated that its participation improves performance: having three compute parties increases efficiency by enabling the use of more optimized MPC protocols (the Sharemind framework also requires exactly three compute parties [7]). The MTA is the analyst (since it specifies the analytics to compute) and the recipient of the results of the computation. Additionally, the MTA plays the role of a decision maker: the MPC protocols and their security guarantees had to be explained to the MTA, and the MTA raised multiple concerns about the system.

The authors state that two concerns unique to the MTA hindered the adoption of their MPC system: (1) the performance of their system was considerably slower than that of a non-MPC system (processing 30 days of VAT data would require spending about 10 days on the computation, as opposed to three days using the non-MPC system), and (2) the MTA was concerned about the transparency of the risk scoring algorithms (since the other compute parties must jointly execute the algorithms using MPC, they must know the code describing the algorithms).

The authors attempted to utilize the capabilities and incentives of the participating parties to argue against these two concerns. First, the protocol was split into two stages to increase efficiency: (1) a parallel MPC task executed by each contributing company to aggregate that company's data, and (2) a global aggregation and analysis task. This optimization is possible for two reasons: the desired analytics can be parallelized by input company before the global aggregation takes place, and the compute parties are capable of acquiring and

managing a powerful computing infrastructure. Second, the authors argued that transparency can improve the acceptance of the MPC system, since the ETA and other companies can inspect the analysis algorithms. The MTA agreed that the transparency and auditability of the MPC system is a step towards incentivizing tax payers to think of paying taxes as a social obligation at the grassroots level. However, the MTA remained concerned about the changes that would need to be instituted across the analysis process in order to enable sharing of risk analysis algorithms with other parties.

Both of these attempts can be understood through the lens of the role inventory and dimensions discussed in Section III. This highlights the importance of designing and deploying MPC solutions while maintaining a role-based perspective, as this can help stakeholders enumerate possible alternatives when incentives are misaligned, can help drive adoption of MPC, and can more explicitly guide selection of improvements to the quality of MPC systems (*e.g.*, in terms of their usability and/or efficiency). The MTA's concerns about transparency exemplify an interesting tension between the auditability of software and the incentives some parties have to keep their analysis algorithms secret. This raises the possibility of developing MPC protocols where the program itself is garbled as part of the input (potentially at the cost of efficiency): the generic protocol can be auditable and public, but the inputs and program can be secret. Such a protocol might need to analyze the program to confirm that it satisfies necessary privacy policies (without revealing it or running it on inputs).

B. Accessible MPC for Privacy-Preserving Data Analytics.

In 2013, the Boston Women's Workforce Council (BWWC) [62] initiated a study of wage gaps among employers within Greater Boston. Uniquely, they planned to use employer-provided data from over over 200 companies to quantify and track progress over time [63]. For an organization, participating in the tracking portion involves aggregating data internally and then contributing that data to a computation across all organizations. The BWWC's efforts were initially stymied by privacy and legal barriers to sharing of sensitive payroll data. Companies refused to submit data to a "trusted third party", and conversely the BWWC had difficulties recruiting an entity to serve in a trusted data collection role due to the risk of being held liable by Compact signers if the payroll data were accidentally leaked or breached. Ultimately, the BWWC used a web-based MPC system for three data analysis sessions to collect payroll analytics securely, thus sidestepping the legal risks involved in handling payroll data [64]. This system used a lightweight solution: a simple browser-based application that could accommodate the familiar look and feel of a spreadsheet [22], with transparent open-source code to enable outside auditing. A variant of a simple, well-known protocol [23], [65] met users' needs and was just expressive enough for the deployment scenario while still being comprehensible enough for key decision makers to feel comfortable.

In this scenario, the signing companies play the role of data contributors (and, indirectly, recipients). The BWWC is the analyst and also a recipient, viewing the employee earnings totals aggregated across all companies while individual company aggregates remain private. The BWWC's collaborating partner (*i.e.*, Boston University) facilitates the private aggregation by supplying the personnel (thus fulfilling the policy expert role) and by installing and running the application back-end on an Amazon Web Services (AWS) cloud-based server (thus acting together with AWS as a compute service provider). The latter arrangement also means that AWS acts as the sole code distributor. The Internet is the communications medium, with secure end-to-end communications achieved via TLS.

The choice of MPC protocol was determined by negotiation and reconciliation of a number of trade-offs and constraints, including which of the roles from Section III participants could inhabit and which of the requirements listed in Section IV-A had to be satisfied. An asynchronous variant of a secret-sharing protocol was used that allows multiple parties to collectively compute a sum of their individual quantities [65]. This solution was comprehensible, required only one compute service provider, did not burden the data contributors or recipient, and had acceptable performance characteristics. However, it was also limited in its expressiveness and in its security guarantees: (1) it only allowed for the computation of linear combinations on the input data and (2) it did not protect against collusion between the compute party and recipient.

This experience showed that for data contributors and recipients the computational efficiency of the MPC component is not the primary performance bottleneck. For simple analytics over relatively small sets of data, all modern frameworks perform rather well (*i.e.*, seconds to minutes) [20]. However, other seemingly unrelated considerations such as choice of client-side cryptographic library have a substantial effect [64]. Furthermore, human time can dominate computing time when a window spanning multiple days is required to collect data from a large number of human contributors with incompatible schedules; asynchrony was an essential protocol feature.

Passive (also known as *semi-honest*) security [66] suffices in this scenario because incentives derived from existing privacy laws are leveraged: the service provider and analyst lack any clear incentive to falsify the results of the aggregation or to learn private input data. To the contrary, completing the study successfully is directly beneficial to the BWWC (as the initiator of the study) as well as to the compute service provider (as an institution reliant upon a reputation of integrity). Additionally, obtaining any of the contributors' private data would create a liability risk for the service provider and analyst (as would any other type of collusion). Thus, the passive model is natural in this case: the service providers are protected from the usual legal risks of processing sensitive data so long as the parties follow the protocol.

An important insight from this deployment was the recognition (through discussions with data contributors about the potential attack surfaces of the software application) that one of the compute service providers and the code distributor were one and the same. It was recognized that this risk can be mitigated by federating the code distribution service itself

using MPC [45].

In a similar use case to that of the BWWC, the Greater Boston Chamber of Commerce (GBCC) launched an economic inclusion initiative called Pacesetters in 2018 [10]. The goal of the initiative is to track (in aggregate) corporate spending on contracts with minority and female-owned businesses within the region. As part of the initiative, companies must contribute information about their spending on a national, state, and local level. MPC is used to derive global insights on corporate spending toward minority and female-owned businesses. The same web-based MPC software is used for this initiative, demonstrating ease of re-usability and how a generalizable solution can arise from this design and development process.

C. Scalable MPC Supporting Heterogeneous Data Stacks.

The goal of our *Conclave* framework [61], [67] is to enable deployment of MPC in scenarios that involve multiple organizations that want to collectively analyze their large data sets. The roles of data contributor and compute service are inhabited by the same entities, and the framework assumes that all parties have the capacity (in terms of access to hardware resources and of availability of software frameworks such as Hadoop) both to store large data sets and to compute over them. Conclave is explicitly designed to exploit this assumption to enhance performance: a data query workflow is optimized to use the local computing resources available to each party as much as possible (limiting MPC computation and communication overhead to only the part of the workflow for which it is essential).

In terms of usability, this framework seeks to eliminate the burden on data contributors who also act as compute parties of tasking their in-house software developers and IT administrators with adopting brand new data storage and computation stacks. The design also seeks to allow data analysts to use a familiar data analysis language rather than learning a new domain-specific language, and to avoid dealing with the issue of data sharing policies (which should be the domain of the data contributors or policy experts).

Conclave is also designed to support the separation of a policy expert (who specifies what data is sensitive and must remain private) from a data analyst (who specifies the data query). The data contributors specify which columns in their contributed data sets are sensitive and must remain private. They can do so without prior knowledge of the query that will be applied to the data sets. At the same time, the automated process that Conclave uses to identify which portion of a computation must utilize MPC allows the query itself to be agnostic with respect to the policies that govern the input data sets. Thus, the author of a query need not be a policy (or cryptography) expert; they only need to concern themselves with authoring the query as if the data is available in the clear.

Also particularly relevant is Conclave's ability to improve an expensive MPC computation's performance via the introduction of a *selectively trusted party* (STP). If directed to do so, Conclave can expose the plaintext values of some columns in the data to the STP in order to achieve significant performance improvements [61], [68]. Such a party may exist for a number of reasons within a deployment scenario. For example, certain organizations that act as STPs may choose to establish legal relationships with the data contributors via non-disclosure agreements or other mechanisms. If MPC is being used to mitigate liability, such organizations may choose to take advantage of insurance and to offer their computing resources in order to improve the performance of third-party MPC computations in which they have no interest otherwise.

More generally, Conclave's features can be viewed as enabling the navigation of *policy-performance trade-offs* for expensive MPC computations over large data sets. This is accomplished by exploiting asymmetries in the cost and privacy constraints among the compute parties. Note the manner in which Conclave's target use cases are distinguished from the previous category of web-based data analytics application use cases: Conclave targets scenarios in which the organizations inhabit the same roles but may have different cost and privacy constraints, while the former software applications target scenarios in which the organizations necessarily inhabit different roles due to resource constraints but mostly have identical security constraints.

VI. Conclusions and Future Work

In this work we have introduced a collection of organizing principles and a conceptual framework for identifying, characterizing, and effectively addressing scenarios that can benefit from the introduction of MPC technologies. We have also shown how this approach can be used to model and examine past attempts to deploy MPC.

The proposed model can help guide future directions in several areas of fundamental research. Work on new MPC primitives, protocols, and technologies can explicitly disentangle roles, concerns, incentives, and capabilities and can aim to provide a modular collection of building blocks that can be used to address regions of the scenario space. Work can also explore the compositionality of protocols along these new dimensions. Formal modeling and analysis of protocol definitions can incorporate these concepts, and work on static analysis techniques can explore how policy agnostic programming and related approaches can enable the level of modularity necessary to support the envisioned ecosystem.

On the implementation and deployment end, the concepts presented can provide a starting point for the vocabulary used within standards and APIs of software solutions that utilize or enable MPC. Designs of open source MPC libraries can aim to support the full range of scenarios, or particular subsets that represent interesting or valuable trade-offs between privacy, liability, performance, and other metrics relevant to a scenario.

Acknowledgment

This work is partially supported by the NSF (under Grants #1430145, #1414119, #1718135, and #1739000) and the Honda Research Institutes. We also thank our reviewers for their detailed and insightful feedback.

REFERENCES

[1] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[2] A. C. Yao, "Protocols for secure computations," in *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*, ser. SFCS '82. Washington, DC, USA: IEEE Computer Society, 1982, pp. 160–164. [Online]. Available: http://dx.doi.org/10.1109/SFCS.1982.88

[3] D. W. Archer, D. Bogdanov, B. Pinkas, and P. Pullonen, "Maturity and performance of programmable secure computation," *IEEE Security Privacy*, vol. 14, no. 5, pp. 48–56, September 2016.

[4] E. Shen, M. Varia, R. K. Cunningham, and W. K. Vesey, "Cryptographically secure computation," *IEEE Computer*, vol. 48, no. 4, pp. 78–81, 2015. [Online]. Available: http://dx.doi.org/10.1109/MC.2015.101

[5] Y. Lindell and B. Pinkas, "Secure multiparty computation for privacy-preserving data mining," *Journal of Privacy and Confidentiality*, vol. 1, no. 1, p. 5, 2009.

[6] "VIFF, the Virtual Ideal Functionality Framework," http://viff.dk/, [Accessed: August 12, 2019].

[7] D. Bogdanov, S. Laur, and J. Willemson, "Sharemind: A Framework for Fast Privacy-Preserving Computations," in *Proceedings of the 13th European Symposium on Research in Computer Security - ESORICS'08*, ser. Lecture Notes in Computer Science, S. Jajodia and J. Lopez, Eds., vol. 5283. Malaga, Spain: Springer Berlin / Heidelberg, October 2008, pp. 192–206.

[8] C. Liu, X. S. Wang, K. Nayak, Y. Huang, and E. Shi, "ObliVM: A programming framework for secure computation," in *Proceedings of the 36th IEEE Symposium on Security and Privacy*, San Jose, CA, USA, May 2015.

[9] K. Nayak, X. S. Wang, S. Ioannidis, U. Weinsberg, N. Taft, and E. Shi, "GraphSC: Parallel secure computation made easy," in *Proceedings of the 36th IEEE Symposium on Security and Privacy*. San Jose, CA, USA: IEEE Computer Society, May 2015, pp. 377–394. [Online]. Available: http://dx.doi.org/10.1109/SP.2015.30

[10] F. Jansen, K. D. Albab, A. Lapets, and M. Varia, "Accessible Privacy-Preserving Web-Based Data Analysis for Assessing and Addressing Economic Inequalities," in *Proceedings of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies*, San Jose, CA, USA, June 2018.

[11] A. Bestavros, A. Lapets, and M. Varia, "User-centric distributed solutions for privacy-preserving analytics," *Communications of the ACM*, vol. 60, no. 2, pp. 37–39, February 2017.

[12] D. D. Clark and D. R. Wilson, "A Comparison of Commercial and Military Computer Security Policies," *Proceedings of the 1987 IEEE Symposium on Research in Security and Privacy (SP'87)*, pp. 184–193, May 1987.

[13] K. El Emam, J. Hu, J. Mercer, L. Peyton, M. Kantarcioglu, B. Malin, D. Buckeridge, S. Samet, and C. Earle, "A secure protocol for protecting the identity of providers when disclosing data for disease surveillance," *Journal of the American Medical Informatics Association: JAMIA*, vol. 18, no. 3, pp. 212–217, May 2011. [Online]. Available: http://europepmc.org/articles/PMC3078664

[14] L. Kamm and J. Willemson, "Secure floating point arithmetic and private satellite collision analysis," *Int. J. Inf. Secur.*, vol. 14, no. 6, pp. 531–548, Nov. 2015. [Online]. Available: http://dx.doi.org/10.1007/s10207-014-0271-8

[15] D. Bogdanov, M. Jõemets, S. Siim, and M. Vaht, *How the Estonian Tax and Customs Board Evaluated a Tax Fraud Detection System Based on Secure Multi-party Computation*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 227–234. [Online]. Available: http://dx.doi.org/10.1007/978-3-662-47854-7_14

[16] A. Abidin, A. Aly, S. Cleemput, and M. A. Mustafa, *An MPC-Based Privacy-Preserving Protocol for a Local Electricity Trading Market*. Milan, Italy: Springer International Publishing, November 2016, pp. 615–625. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-48965-0_40

[17] "Open Science Data Cloud," https://www.opensciencedatacloud.org/, [Accessed: August 12, 2019].

[18] "NCI Cloud Resources," https://cbiit.nci.nih.gov/ncip/cloudresources, [Accessed: August 12, 2019].

[19] "ThreatExchange," https://developers.facebook.com/products/threat-exchange, [Accessed: August 12, 2019].

[20] D. W. Archer, D. Bogdanov, B. Pinkas, and P. Pullonen, "Maturity and performance of programmable secure computation," *IACR Cryptology ePrint Archive*, vol. 2015, no. 1039.

[21] M. Yung, "From mental poker to core business: Why and how to deploy secure computation protocols?" in *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '15. New York, NY, USA: ACM, 2015, pp. 1–2. [Online]. Available: http://doi.acm.org/10.1145/2810103.2812701

[22] L. Qin, P. Flockhart, A. Lapets, F. Jansen, K. D. Albab, M. Varia, S. Roberts, and I. Globus-Harris, "From Usability to Secure Computing and Back Again," in *Proceedings of SOUPS 2019: The 15th Symposium on Usable Privacy and Security*, Santa Clara, CA, USA, August 2019.

[23] A. Lapets, N. Volgushev, A. Bestavros, F. Jansen, and M. Varia, "Secure Multi-Party Computation for Analytics Deployed as a Lightweight Web Application," in *2016 IEEE Cybersecurity Development (SecDev)*, November 2016, pp. 73–74.

[24] A. Rajan, L. Qin, D. W. Archer, D. Boneh, T. Lepoint, and M. Varia, "Callisto: A Cryptographic Approach to Detecting Serial Perpetrators of Sexual Misconduct," in *Proceedings of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies*, San Jose, CA, USA, June 2018.

[25] F. Jansen, K. D. Albab, A. Lapets, and M. Varia, "Brief Announcement: Federated Code Auditing and Delivery for MPC," in *Proceedings of SSS 2017: The 19th International Symposium on Stabilization, Safety, and Security of Distributed Systems*, Boston, MA, USA, November 2017.

[26] P. Bogetoft, D. L. Christensen, I. Damgård, M. Geisler, T. Jakobsen, M. Krøigaard, J. D. Nielsen, J. B. Nielsen, K. Nielsen, J. Pagter, M. Schwartzbach, and T. Toft, "Financial cryptography and data security," R. Dingledine and P. Golle, Eds. Berlin, Heidelberg: Springer-Verlag, 2009, ch. Secure Multiparty Computation Goes Live, pp. 325–343. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-03549-4_20

[27] I. Damgård, K. Damgård, K. Nielsen, P. S. Nordholt, and T. Toft, "Confidential benchmarking based on multiparty computation," in *Financial Cryptography*, 2016, pp. 169–187. [Online]. Available: https://doi.org/10.1007/978-3-662-54970-4_10

[28] D. Bogdanov, L. Kamm, B. Kubo, R. Rebane, V. Sokk, and R. Talviste, "Students and Taxes: a Privacy-Preserving Study Using Secure Computation," *PoPETs*, vol. 2016, no. 3, pp. 117–135, 2016. [Online]. Available: https://eprint.iacr.org/2015/1159.pdf

[29] J. Launchbury, I. S. Diatchki, T. DuBuisson, and A. Adams-Moran, "Efficient lookup-table protocol in secure multiparty computation," in *Proceedings of the 17th ACM SIGPLAN International Conference on Functional Programming*, ser. ICFP '12. New York, NY, USA: ACM, 2012, pp. 189–200. [Online]. Available: http://doi.acm.org/10.1145/2364527.2364556

[30] M. Burkhart, M. Strasser, D. Many, and X. Dimitropoulos, "SEPIA: Privacy-preserving aggregation of multi-domain network events and statistics," in *Proceedings of the 19th USENIX Conference on Security*, ser. USENIX Security'10. Berkeley, CA, USA: USENIX Association, 2010, pp. 15–15. [Online]. Available: http://dl.acm.org/citation.cfm?id=1929820.1929840

[31] A. Rastogi, M. A. Hammer, and M. Hicks, "Wysteria: A programming language for generic, mixed-mode multiparty computations," in *Proceedings of the 35th IEEE Symposium on Security and Privacy*, ser. SP '14. Washington, DC, USA: IEEE Computer Society, 2014, pp. 655–670. [Online]. Available: http://dx.doi.org/10.1109/SP.2014.48

[32] Y. Ejgenberg, M. Farbstein, M. Levy, and Y. Lindell, "SCAPI: The secure computation application programming interface," Cryptology ePrint Archive, Report 2012/629, 2012, https://eprint.iacr.org/2012/629.

[33] A. Ben-David, N. Nisan, and B. Pinkas, "FairplayMP: a system for secure multi-party computation," in *Proceedings of the 15th ACM conference on Computer and Communications Security*. Toronto, Canada: ACM, 2008, pp. 257–266.

[34] D. Demmler, T. Schneider, and M. Zohner, "ABY–a framework for efficient mixed-protocol secure two-party computation," in *Proceedings of the 2015 Network and Distributed System Security (NDSS) Symposium*, San Diego, CA, USA, 2015.

[35] A. R. Choudhuri, M. Green, A. Jain, G. Kaptchuk, and I. Miers, "Fairness in an unfair world: Fair multiparty computation from public bulletin boards," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '17. New York, NY, USA: ACM, 2017, pp. 719–728. [Online]. Available: http://doi.acm.org/10.1145/3133956.3134092

[36] S. Kamara, "Server-aided private set intersection (psi) with data transfer," Oct. 13 2015, uS Patent 9,158,925.

[37] P. Scholl, N. P. Smart, and T. Wood, "When It's All Just Too Much: Outsourcing MPC-Preprocessing," in *Cryptography and Coding*, M. O'Neill, Ed. Oxford, UK: Springer International Publishing, 2017, pp. 77–99. [Online]. Available: https://doi.org/10.1007/978-3-319-71045-7_4

[38] Y. Lindell, B. Pinkas, N. P. Smart, and A. Yanai, "Efficient constant round multi-party computation combining BMR and SPDZ," in *Advances in Cryptology – CRYPTO 2015*, R. Gennaro and M. Robshaw, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 319–338.

[39] M. Keller, P. Scholl, and N. P. Smart, "An architecture for practical actively secure MPC with dishonest majority," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '13. New York, NY, USA: ACM, 2013, pp. 549–560. [Online]. Available: http://doi.acm.org/10.1145/2508859.2516744

[40] A. Schroepfer and F. Kerschbaum, "Demo: Secure computation in JavaScript," in *Proceedings of the 18th ACM Conference on Computer and Communications Security*, ser. CCS '11. New York, NY, USA: ACM, 2011, pp. 849–852. [Online]. Available: http://doi.acm.org/10.1145/2046707.2093509

[41] A. Jarrous and B. Pinkas, "Canon-MPC, a system for casual non-interactive secure multi-party computation using native client," in *Proceedings of the 12th ACM Workshop on Workshop on Privacy in the Electronic Society*, ser. WPES '13. New York, NY, USA: ACM, 2013, pp. 155–166. [Online]. Available: http://doi.acm.org/10.1145/2517840.2517845

[42] "Refactor NaCl integration to eliminate the trusted, in-process plugin." https://bugs.chromium.org/p/chromium/issues/detail?id=239656#c160, [Accessed: August 12, 2019].

[43] J. Perry, D. Gupta, J. Feigenbaum, and R. N. Wright, "Systematizing secure computation for research and decision support," in *Security and Cryptography for Networks*, M. Abdalla and R. De Prisco, Eds. Amalfi, Italy: Springer International Publishing, 2014, pp. 380–397. [Online]. Available: https://doi.org/10.1007/978-3-319-10879-7_22

[44] M. Dahl, V. Pastro, and M. Poumeyrol, "Private data aggregation on a budget," Cryptology ePrint Archive, Report 2017/643, 2017, http://eprint.iacr.org/2017/643.

[45] F. Jansen, K. D. Albab, A. Lapets, and M. Varia, "Brief Announcement: Federated Code Auditing and Delivery for MPC," in *Stabilization, Safety, and Security of Distributed Systems*, P. Spirakis and P. Tsigas, Eds. Boston, MA, USA: Springer International Publishing, 2017, pp. 298–302.

[46] J. Yang, T. Hance, T. H. Austin, A. Solar-Lezama, C. Flanagan, and S. Chong, "End-to-end policy-agnostic security for database-backed applications," *CoRR*, vol. abs/1507.03513, 2015. [Online]. Available: http://arxiv.org/abs/1507.03513

[47] J. Yang, K. Yessenov, and A. Solar-Lezama, "A language for automatically enforcing privacy policies," in *Proceedings of the 39th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, ser. POPL '12. New York, NY, USA: ACM, 2012, pp. 85–96. [Online]. Available: http://doi.acm.org/10.1145/2103656.2103669

[48] T. H. Austin, J. Yang, C. Flanagan, and A. Solar-Lezama, "Faceted execution of policy-agnostic programs," in *Proceedings of the Eighth ACM SIGPLAN workshop on Programming languages and analysis for security*, ser. PLAS '13. New York, NY, USA: ACM, 2013, pp. 15–26.

[49] R. Canetti, "Universally composable security: A new paradigm for cryptographic protocols," Cryptology ePrint Archive, Report 2000/067 (2013 version), 2013, https://eprint.iacr.org/2000/067/20130717:020004.

[50] R. Canetti, A. Cohen, and Y. Lindell, "A simpler variant of universally composable security for standard multiparty computation," Cryptology ePrint Archive, Report 2014/553, 2014, http://eprint.iacr.org/2014/553.

[51] R. Canetti, S. Chari, S. Halevi, B. Pfitzmann, A. Roy, M. Steiner, and W. Venema, "Composable security analysis of OS services," *IACR Cryptology ePrint Archive*, vol. 2010, p. 213, 2010. [Online]. Available: http://eprint.iacr.org/2010/213

[52] R. Canetti, K. Hogan, A. Malhotra, and M. Varia, "A universally composable treatment of network time," in *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, Santa Barbara, CA, USA, August 2017, pp. 360–375.

[53] K. Hogan, H. Maleki, R. Rahaeimehr, R. Canetti, M. van Dijk, J. Hennessey, M. Varia, and H. Zhang, "On the universally composable security of openstack," in *IEEE Cybersecurity Development (SecDev)*, 2019, https://eprint.iacr.org/2018/602.

[54] L. Grassi, C. Rechberger, D. Rotaru, P. Scholl, and N. P. Smart, "MPC-friendly symmetric key primitives," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '16. New York, NY, USA: ACM, 2016, pp. 430–443. [Online]. Available: http://doi.acm.org/10.1145/2976749.2978332

[55] N. Volgushev, A. Lapets, and A. Bestavros, "Programming support for an integrated multi-party computation and MapReduce infrastructure," in *Proceedings of HotWeb 2015: The Third IEEE Workshop on Hot Topics in Web Systems and Technologies*, Washington, D.C., USA, November 2015. [Online]. Available: http://www.cs.bu.edu/fac/best/res/papers/hotweb15.pdf

[56] M. Bar-Sinai, L. Sweeney, and M. Crosas, "Datatags, data handling policy spaces and the tags language," in *2016 IEEE Security and Privacy Workshops (SPW)*, San Jose, CA, USA, May 2016, pp. 1–8.

[57] P. D. Azar, S. Goldwasser, and S. Park, "How to incentivize data-driven collaboration among competing parties," in *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science*, M. Sudan, Ed. Cambridge, MA, USA: ACM, January 2016, pp. 213–225. [Online]. Available: http://doi.acm.org/10.1145/2840728.2840758

[58] A. Bestavros and O. Krieger, "Toward an open cloud marketplace: Vision and first steps," *IEEE Internet Computing*, vol. 18, no. 1, pp. 72–77, 2014. [Online]. Available: https://doi.org/10.1109/MIC.2014.17

[59] Y. Narahari, *Game Theory and Mechanism Design*. World Scientific Publishing Company Pte. Limited, 2014.

[60] V. Krishna, "Chapter five - mechanism design," in *Auction Theory (Second Edition)*, second edition ed., V. Krishna, Ed. San Diego: Academic Press, 2010, pp. 61 – 83. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780123745071000261

[61] N. Volgushev, M. Schwarzkopf, B. Getchell, M. Varia, A. Lapets, and A. Bestavros, "Conclave: Secure Multi-Party Computation on Big Data," in *Proceedings of EuroSys 2019: The 12th European Conference on Computer Systems*, Dresden, Germany, March 2019.

[62] "The Boston Women's Workforce Council," https://www.boston.gov/departments/womens-advancement/womens-workforce-council, [Accessed: August 12, 2019].

[63] "100% Talent: The Boston Women's Compact," https://www.boston.gov/departments/womens-advancement/womens-workforce-council#-talent-compact, [Accessed: August 12, 2019].

[64] A. Lapets, N. Volgushev, A. Bestavros, F. Jansen, and M. Varia, "Secure Multi-Party Computation for Analytics Deployed as a Lightweight Web Application," CS Dept., Boston University, Tech. Rep. BUCS-TR-2016-008, July 2016. [Online]. Available: http://www.cs.bu.edu/techreports/pdf/2016-008-mpc-lightweight-web-app.pdf

[65] C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin, and M. Y. Zhu, "Tools for privacy preserving distributed data mining," *SIGKDD Explor. Newsl.*, vol. 4, no. 2, pp. 28–34, Dec. 2002. [Online]. Available: http://doi.acm.org/10.1145/772862.772867

[66] O. Goldreich, *The Foundations of Cryptography - Volume 2, Basic Applications*. Cambridge University Press, 2004.

[67] N. Volgushev, M. Schwarzkopf, A. Lapets, M. Varia, and A. Bestavros, "DEMO: Integrating MPC in Big Data Workflows," in *Proceedings of CCS 2016: The 23rd ACM SIGSAC Conference on Computer and Communications Security*, Vienna, Austria, October 2016.

[68] N. Volgushev, M. Schwarzkopf, B. Getchell, M. Varia, A. Lapets, and A. Bestavros, "Conclave: secure multi-party computation on big data (extended TR)," arXiv, Report 1902.06288, February 2019, available at https://arxiv.org/abs/1902.06288.