Privacy-aware Adaptive Scheduling for Coalition Operations

Karen L. Myers,[1] Thomas Lee,[1] Laura Tam,[1] José Manuel Calderón Trilla,[2] Benjamin Davis,[2] Stephen Magill[2]
Artificial Intelligence Center, SRI International[1]
333 Ravenswood Ave., Menlo Park, CA 94025
firstname.lastname@sri.com
Galois, Inc.[2]
421 SW 6th Avenue, Suite 300
Portland, Oregon 97204
firstname.lastname@galois.com

Abstract

Coalition operations are essential for responding to the increasing number of world-wide incidents that require large-scale humanitarian assistance. Many nations and non-governmental organizations regularly coordinate to address such problems but their cooperation is often impeded by limits on what information they are able to share. In this paper, we consider the use of an advanced cryptographic technique called secure multi-party computation to enable coalition members to achieve joint objectives while still meeting privacy requirements. Our particular focus is on a multi-nation aid delivery scheduling task that involves coordinating when and where various aid provider nations will deliver relief materials after the occurrence of a natural disaster. Even with the use of secure multi-party computation technology, information about private data can leak. We describe how the emerging field of quantitative information flow can be used to help data owners understand the extent to which private data might become vulnerable as the result of possible or actual scheduling operations, and to enable automated adjustments of the scheduling process to ensure privacy requirements.

Introduction

Coalition operations are becoming an increasing focus for many nations. The need for collaboration derives from increasing awareness of mutual interests among both allies and nations that traditionally have not worked closely together. For example, coalition operations for Humanitarian Assistance and Disaster Relief (HADR) have increased substantially in recent years in both numbers and scope. With the impact of global warming, it is anticipated that there will be even more large-scale humanitarian crises resulting from adverse weather-related events and sea-level rises. These coalitions often involve not just government and military organizations but also non-governmental organizations (NGOs) and commercial entities.

A key challenge facing coalitions is how to collaborate without releasing information that could jeopardize national (or organizational) interests. Information security mechanisms for the military to date have focused on safeguarding information by limiting its access and use. This approach has lead to a significant *undersharing* problem, which impedes effective joint operations. Recent work on defining access control mechanisms is useful for enabling selective sharing of information that is safe to release (Phillips, Ting, and Demurjian 2002). However, many coalition tasks require information that participants do not wish to make available to others.

For example, consider the problem of scheduling the delivery of aid (food, water, medicine, fuel) to impacted nations after a natural disaster. Historically, international response has involved dozens of countries and NGOs, each with the desire to contribute in interconnected and potentially conflicting ways. Ideally coordination would be achieved by directly sharing information about the amount of aid each contributor has available or can produce, where that aid is situated, the position and storage capacity of ships for delivering the aid, harbor facilities where aid ships could dock, etc. But the owners of this data may be unwilling to directly share it with coalition partners, for fear of revealing information that impacts national security (e.g., ship locations) or competitive advantages (e.g., a company's backlog of inventory).

To address this problem of coalition collaboration without revealing private information, we exploit a cryptographic technology called *secure multi-party computation (MPC)* (Yao 1982). MPC protocols enable mutually distrusting parties to perform joint computations on private information while it remains encrypted. In our work, we use MPC to enable privacy-preserving computations over several types of coordination tasks related to scheduling aid delivery.

While participants cannot directly learn others' private inputs from the process/protocol of a secure multi-party computation, they may be able to infer something about private data based on the results of the computation. For this reason, our privacy-aware scheduling solution makes use of a complementary technology called *quantitative information flow (QIF)* to measure the degree to which private data used in the scheduling task might leak to other participants. Insights gained from this analysis are folded back into the scheduling process via an adaptive workflow capability, to ensure that the vulnerability of private data stays within acceptable thresholds.

The paper is organized as follows. We begin by summarizing our aid distribution task, including a description of the core scheduling problem and the data (private and non-private) belonging to the various coalition members. We then provide a short overview of secure multi-party computation followed by a description of how we employ it within a broader adaptive workflow framework to address the aid delivery scheduling task. Next, we describe our use of quantitative information flow to assess the vulnerability of private information as the scheduling progresses and to adapt the scheduling workflow in light of those assessments to ensure adherence to predefined privacy requirements. We conclude by identifying directions for future work and summarizing our contributions.

HADR Aid Delivery Problem

The aid delivery problem that we consider involves two categories of participants:

- N *Aid Provider* nations, each of which has some number S_n of ships with aid (e.g., food, medicine) to be delivered
- a single *Aid Recipient* nation that has P ports to which aid can be delivered

Collectively, these participants need to formulate a schedule for delivering aid on board the Aid Provider ships to ports belonging to the Aid Recipient, ensuring delivery prior to a specified deadline. As summarized in Figure 1a , a solution involves assigning to each Aid Provider ship: a port to which its aid should be delivered, a berth at the port, and a docking time. These assignments are subject to various constraints on ship and port characteristics to ensure physical compatibility between the ship and the port/berth, schedule availability of the berth, and the ability for the ship to completing the docking before the assigned deadline. We further seek an assignment that optimizes according to the following load-balancing criteria.

Optimization Criteria: Load-balancing across ports. Let *Assigned*(Port_i) designate the number of aid provider ships assigned to aid recipient port Port_i. An optimal solution is a set of assignments that minimizes

$$\text{MAX}_{j,k}(|\,\textit{Assigned}(\text{Port}_j) - \textit{Assigned}(\text{Port}_k)\,|)$$

Generating a solution to this scheduling problem requires information from the various parties about their assets (ships, ports), some of which they would prefer not to share with other coalition members. Figure 1b summarizes this private data. In both Figure 1b and Figure 1a the problem data is color-coded, with blue used for private data belonging to an Aid Provider nation and green for private data belonging to the Aid Recipient nation. As the coloring clearly shows, determining a solution requires combining private information from multiple parties, which motivates our use of secure multi-party within our scheduling algorithm.

Secure Multi-Party Computation

To address privacy concerns in our aid delivery use case, we leverage MPC. MPC protocols compute a function and reveal the output of the computation without revealing private inputs to any other participant. Early MPC work was based on two-party problems (Yao 1982), and subsequent work produced general approaches for any number of participants to participate in shared computation without learning private information (Goldreich, Micali, and Wigderson 1987).

Most MPC approaches involve modeling the desired algorithm as a boolean circuit (fixed-time algorithms expressed as a combination of `AND`, `OR`, and `NOT` logic gates). However, circuit-based approaches may require unrolling loops, introducing potential performance implications. Alternate approaches based on Oblivious RAM (Goldreich and Ostrovsky 1996) do not require this full unrolling though have other performance trade-offs. We use a circuit-based MPC approach in our scenario, though our privacy analysis and adaptive scheduling do not depend on the specifics of the underlying MPC approach.

MPC has proven to be a powerful tool to enable a wide range of use cases. For example, private keys can be split among several hosts in order to reduce the number of hosts that must be compromised in order to obtain the key. A computation that requires the decryption operation can then be done under MPC between the hosts, ensuring that the whole key is never revealed in the clear (Archer et al. 2018). MPC can also be used between companies in collaborative supply-chain management, where the reluctance to share sensitive data (such as costing and capacities) can lead to sub-optimal production plans. The use of MPC allows for collaborative supply-chain planning without compromising sensitive data (Kerschbaum et al. 2011).

In our setting, each data owner provides their private input to a MPC circuit designed such that their inputs are not revealed to other participants. Then participants follow the protocol for the multi-party computation, which allows them to collectively compute the solution to the planning and scheduling optimization problem. In the end, only the final result (schedule) is revealed to the relevant parties.

Scheduling Workflow

Creating a single MPC circuit to solve the full optimized scheduling problem in a general way is not practical at this point in time. For this reason, our solution approach consists of a workflow that decomposes the scheduling task into the following sequence of steps.

Step A. Collect relevant inputs:

- Aid Provider: determine ports that can be reached by each ship before the deadline D
- Aid Recipient: select ports to which aid will be accepted

Step B. Determine ports that satisfy joint ship/harbor physical compatibility constraints:

- Aid Provider ship draft is compatible with the harbor depth in the port
- Aid Recipient port has sufficient offload capacity

Step C. Determine all berths within each feasible port from Step B that satisfy joint ship/berth compatibility and berth availability constraints:

- Aid Provider ship fits within the berth

For a **Ship** to be scheduled to dock at a given **Docking Time** at a **Berth** belonging to a **Port**, the following must hold:

Port

- **ship-draft** $\leq$ **port-harbor-depth**
- ship-cargo-amount $\leq$ **port-cargo-offload-capacity**
- **ship-earliest-arrival** $= \frac{\text{EarthDistance}(\textbf{ship-location}, \text{port})}{\textbf{ship-maxspeed}}$
- **ship-earliest-arrival** $\leq$ deadline

Berth in Port

- **ship-length** $\leq$ **berth-length**

Docking Time at Berth

- **ship-earliest-arrival** $\leq$ docking-time $\leq$ deadline
- **berth-availability** (pairs of **berth-occupied-start** and **berth-occupied-end**) for berth at docking-time

(a) Constraints in the Scheduling Problem

Aid Provider	Aid Recipient	
Ship Info	Port Info	Berth Info
ship-location	**port-available**	**berth-length**
ship-length	**port-cargo-offload-capacity**	**berth-occupied-start**
ship-draft	**port-harbor-depth**	**berth-occupied-end**
ship-maxspeed		

(b) Summary of Private Data by Data Owner

Figure 1: Constraints over the private data and the relationship of private data to the data owners

- Aid Recipient berth is available at planned ship arrival time

Step D. Schedule the aid ships across possible ship-port-berth-arrival-time options (from Step C) in accordance with the optimization goal of load-balancing across ports.

Step A is a simple information gathering/filtering task performed locally by individual participants. Step B requires computation over private inputs from both the Aid Provider and the Aid Recipient. Steps C and D combine private inputs with intermediate results from earlier steps. For these reasons, we use three secure multi-party circuits within our workflow:

- A two-party circuit for the physical compatibility test in Step B (Aid Provider and Aid Recipient)
- A two-party circuit for the viability test in Step C (Aid Provider and Aid Recipient)
- An N-party circuit for optimizing berth allocation in Step D (all N Aid Providers)

The circuits for Steps B and C are straightforward but must be run for each possible ship/port (Step B) and ship/berth (Step C) combination. The circuit for Step D is more complex. Because it is not possible to implement a truly optimized solution in the MPC circuit model, we instead opted for the following greedy approach to load balancing across ports.

1. Initialization: set the list of ship-port-berth solutions to be empty and the working set of options to be the set of ship-port-berth-unload time entries from Step C
2. Select a port P with the fewest number of assignments and for which there remain options
3. Select earliest ship-port-berth-unload time entry for P and add to the solutions list
4. Remove from the set of options all entries for the ship and berth selected in Step 3
5. Repeat steps 2-4 until no more ship-port-berth solutions remain
6. Output the solution list

We use the Lumen agent technology (described in (Myers et al. 2011)) to provide an adaptive workflow capability for executing this scheduling process. Although we depict only one workflow here, more generally our adaptive workflow capability draws from a library of alternative approaches to a range of aid distribution scheduling problems, enabling solution approaches to be matched to the specifics of a given situation. For example, we have defined alternative workflows that embed different scheduling and optimization strategies and that make use of secure multi-party computation in different ways as a means of investigating tradeoffs between efficiency and privacy.

Quantifying Information Vulnerability

In this section we describe what is involved in using QIF to reason about questions of how private data can be inferred from the results of a computation.

Limitations of Multi-Party Computation

The “Millionaires’ Problem” (Yao 1982) is an early use-case for multi-party computation in which two people, Alice and Bob, use MPC to securely determine which of them is richer without revealing their actual wealth to each other. But while they don’t learn the other’s precise values, they do learn *something* about each other.

The nature of the relationship between the revealed output and the private inputs determines how much one may infer about the inputs from the result. For example, if Alice and Bob use MPC to instead compute the arithmetic mean of their net worths, Alice could use the input she provided and the resulting mean from the computation to solve for Bob's exact net worth. Alice may not be able to extract Bob's input via examination of the MPC circuit itself, but MPC cannot change the relationship between inputs and output that exists outside the circuit.

In our Aid Delivery scenario, private "inputs" may involve sensitive capability and operational details. As the scenarios increase in scope and complexity, it becomes difficult to reason informally about what an adversary may be able to infer.[1] We use Quantitative Information Flow (QIF) to address these concerns by characterizing an adversary's ability to make these inferences based on information-theoretic bounds on the relationship between results and private inputs.

Modeling Workflows

A QIF analysis begins by transforming a program (such as our aid distribution workflow) into a model that represents the relationship between the (private) inputs and outputs. Specifically, these models are based on information-theoretic *channels*, which we use to construct a mapping from prior[2] distributions on private data to distributions on posterior distributions (Smith 2011; Smith 2009).

We use these models to support "predictive-mode" adaptive workflows in which we reason about what an adversary could learn from any possible set of private inputs, as well as "posterior-mode" in which we determine how much an adversary may learn from some specific concrete result.

Quantifying Inference Capabilities

We can construct games in which we quantify the adversary's inference capability by measuring how their chances of "winning" (i.e., guessing a piece of private data) improve given additional information. We call the probability that an adversary can with one chance correctly guess a piece of private data the *vulnerability* of that variable.

In the "prior game," the defender picks a private input value by sampling from the prior distribution of possible input values. The adversary makes their best guess of the defender's input based only on their knowledge of the prior distribution of possible input values. The *prior vulnerability* is the probability the adversary will correctly guess the defender's private input (without any additional information).

In the "posterior game," the defender again picks a private input, but then performs the computation using that input and shares (only) the result with the adversary. We also assume the adversary has full knowledge of how the inputs relate to outputs in the computation (e.g., could construct an identical channel-based model of the computation). Here the chance the adversary correctly guesses the defender's private input is called the *posterior vulnerability*.

To summarize, the prior game is how likely an adversary is to guess the private input *before* seeing the result of the computation, the posterior game is how likely the adversary is to guess the private input *after* seeing the *specific output* of the computation.

The Use of Vulnerability as a Metric

Vulnerability as a metric of risk is appealing because of its relatively intuitive nature. The caveat to vulnerability as a metric is that it is extremely sensitive to the chosen prior belief. Therefore, care must be taken in deciding upon a prior in order to assure that the metric reflects reality as much as possible.

Supporting Workflow Adaptation

Our predictive-mode adaptation strategy is based on QIF *predictive leakage*, which attempts to predict how much will be leaked about the private data before the computation is run on the concrete private values. We can also approximate predictive leakage using Monte Carlo simulation, permitting the use of these metrics even in scenarios where operational requirements make it infeasible to complete the precise predictive leakage within a desired time frame. Incorporating predictive leakage metrics into our workflows enables identification of potentially higher-risk situations where it may be preferable to adapt or halt the workflow (based on some policy) rather than participating in a computation that may reveal too much.

Our posterior-mode adaptation strategy is based on QIF *dynamic leakage*, which takes into account the actual results of a computation. As opposed to the predictive leakage's assessments, that average all possible private input values, dynamic leakage enables our workflows to incorporate more accurate assessments of what was actually revealed in the specific ongoing workflow.

Sample Vulnerability Analysis

As discussed previously, the HADR aid delivery problem provides a strong foundation for analyzing adaptive workflows in a privacy-aware setting. In this section we describe how the notion of *vulnerability* can be used to inform the cooperating parties of risk to their private data.

A Simplifying Example

To begin our discussion we first describe a simplified version of the HADR aid delivery problem involving a single ship, S, and a single port, P. In this simplified scenario, the only private data is S's location (`ship_loc`). We assume that

[1]In this work, an 'adversary' is a so-called *honest-but-curious* actor who attempts to infer the values of private data by observing the public results of each step in the workflow. Adversaries could be other coalition partners or an unrelated third party that has access to the computational framework.

[2]A 'prior' can be thought of as the initial set of beliefs that an adversary has about a system. An adversary may have a prior over the lengths of naval ships (e.g., between 10 and 1,500 feet long), or the possible locations of the ships. If an adversary has no reason to believe one value is more likely than any other then the prior is a uniform distribution over the space of secrets, hence it is known as a *uniform* prior.

other relevant data (such as ship maximum speed) is known publicly.

```
def reachable(deadline):
  dist = distance(ship_loc, port_loc)
  return (dist <= (max_speed * deadline))
```

Figure 2: The pseudo-code for a simple query

The channel is "is S able to reach port P within d hours?". As pseudo-code we write this channel as a function of d, over some global set of private (`ship_loc`) and public (`port_loc`, `max_speed`) variables.

Prior Belief In our simplified scenario an adversary who wants to determine S's private data will have some *prior* belief about that data. In this case, the prior belief will be some area of the ocean where S could be located. This may or may not be informed by other information available to the adversary. While QIF can be calculated over non-uniform priors, in this exposition we assume uniform priors for the sake of simplicity. Figure 3a shows a graphical representation of a possible prior for the simplified scenario.

Because our prior is uniform, the adversary's chance of guessing the ship's position (the *prior vulnerability*, V_{prior}), is simply

$$V_{\text{prior}} = \frac{1}{\text{number of possible locations in the prior}}$$

This makes intuitive sense: under a uniform prior, the adversary's likelihood of winning the prior game is equivalent to choosing a point out of the prior at random.

Posterior Belief When ship S, cooperating in the simplified scenario, responds to the `reachable` query in Figure 2, the adversary is able to observe the *output* of the channel. This observation allows the adversary to rule out entire sections of the space of possible locations.

If the result of the query is `True`, then the adversary can infer that the location of S is within a circle of radius r, where r is the distance that the ship can travel at `max_speed` with `deadline` amount of time. Inversely, if the result of the query is `False`, then the adversary could infer that the ship must be *outside* of the same circle. Observing `False` is the circumstance illustrated in Figure 3b.

The important point is to see that *regardless of the result* of the query, the adversary may be able to infer rule out subsets of possible values for the private data.

The probability the adversary has of guessing the ship's position after seeing the channel output (the *posterior vulnerability*, V_{post}) is simply

$$V_{\text{post}} = \frac{1}{\text{number of possible locations in the posterior}}$$

Further Queries Because cooperation is the goal, it is likely that the coordinator of our simple scenario will need to query S multiple times. This is particularly true if the initial query's result was `False`. The coordinator may query S with `reachable(d2)`, where $d2$ is a new, longer, deadline. If the result is then `True` then the adversary is able to infer that S resides within the ring formed by two overlapping circles: an inner circle with radius r, described above, and a wider circle with radius $r2$, where $r2$ is the distance that S is able to travel at max speed in time $d2$. This circumstance is illustrated in Figure 3c.

Worst-Case Simple Scenario If we add another port, $P2$, to the simple scenario above, but keep all other details the same, it may be possible for S's location to be determined within a very small tolerance. If the result of `reachable` is `True` for both P and $P2$, then (using the same process as above) the adversary would intersect the appropriate circles. The smaller the intersection, the more the adversary knows about the ship's position.

Analysis of the Aid Distribution Scenario

The simplified scenario above only has one piece of private data: the ship's position. As a reminder, Figure 1b shows the private data in the full HADR scenario. In this section we present some results from an analysis over a model of the full scenario, using a single ship (Ship #9) as a running example.

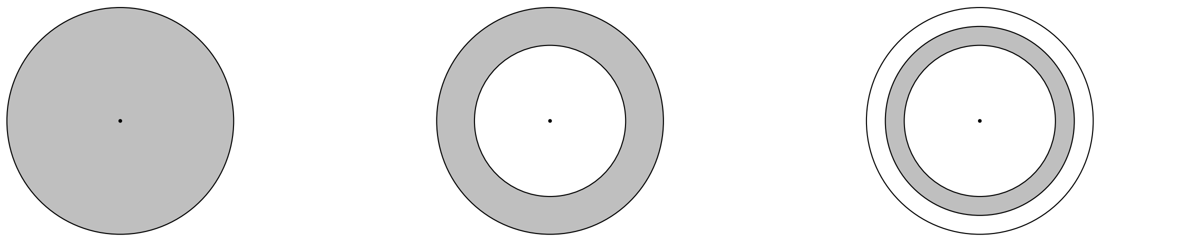
Prior Vulnerability As discussed above, an adversary has *some* notion of the possible values that a secret can take on even before running any computations over that secret data. This is referred to as the *Prior*. When analyzing a system, the analyst must *choose* appropriate ranges for the private data in question. For some variables this may be straightforward (e.g., the ocean sector in question when deciding on a prior for a ship's location), for others it may depend on domain knowledge (e.g., the appropriate range for naval ship drafts).

We can visualize the prior vulnerability over location easily. Figure 5 shows several aspects of the HADR scenario. The boundaries of the map in the image are the boundaries of the prior belief over ship position. It is important to remind ourselves that the choice of prior (part of which is determined by the geographic area under consideration) affects the vulnerability metric significantly. Increasing the area *decreases* the prior vulnerability, while decreasing the area *increases* the prior vulnerability. Because the prior models the *adversary's* view of the world, it should be constructed carefully.

Predictive Vulnerability for Steps A+B In resource-constrained systems, it is useful to be able to predict how much of a given resource would be used if a certain action were to be executed. Private data can similarly be viewed as a form of resource for which predictive analysis can be applied to assess the impact of planned or possible actions.

The ability to predict the future vulnerability can be implemented in several ways, the practicality and efficiency of different methods depends heavily on the constraints of the system, imposed by the state-space and the adversarial model.[3]

[3]In the QIF literature, this ability is knowns as *Static Leakage Analysis* (Smith 2009; Alvim et al. 2012).



(a) The initial prior belief: no reason to believe the ship is in one location over any other.

(b) Posterior belief after observing a `False`: The ship must be *outside* the distance of the `reachable` query.

(c) If a subsequent query for a further distance returns `True` the ship must be within the newly formed ring.

Figure 3: The effects of observing query results on a prior belief over the simplified scenario.

In our system we have chosen an approximate method that enables an analyst to calculate a histogram over the possible vulnerability outcomes without any access to the private data. This method uses Monte Carlo simulation to run our analysis over randomly sampled points from the space of possible private data values.

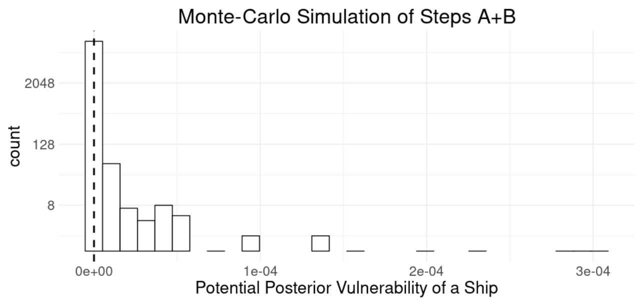


Figure 4: Predictive Vulnerability of Steps A+B

Figure 4 shows the results of a Monte Carlo simulation of the predictive vulnerability of running Steps A+B (using 13788 samples[4]). The vertical dotted line shows the median value ($6.847603\text{e}{-6}\%$) of all the samples. This method of approximation provides analysts with various options. In a scenario where the preservation of privacy is paramount, the analyst may focus on the right-hand side of the figure, where the potential vulnerability is higher, 0.03039514%, though still unlikely.

This method also adapts well to use cases where an analyst may have access to *some* of the private data. For example, an analyst for one of the coalition partner nations might have access to that nation's private data, but not the private data of the ports. Using this same method of sampling from the space of (unknown) private data, such an analyst would be able to approximate the future vulnerability of their data if they were to respond to a query.

Posterior Vulnerability for Steps A+B Once a step is taken, we can calculate the *posterior vulnerability* of the private data. Unlike the predictive analysis in the previous section, this analysis is 'exact' in that the real vulnerability cannot be more than what the analysis reports.

Ship #9 Position	$6.847603\text{e}{-6}\%$

In this case, the posterior vulnerability coincides with the median of the predictive vulnerability. This is not too surprising as the median was also the most likely value by a significant margin.

Predictive Vulnerability for Step C Figure 6 shows the Monte Carlo simulation for vulnerability of a ship after Step C is completed.[5] The median predicted value in this instance, $6.644298\text{e}{-4}\%$, is two orders of magnitude higher than the vulnerability after Steps A+B alone. This makes intuitive sense as much more information is revealed after Step C that can be used to infer a ship's private data. Unsurprisingly, the maximum sampled predictive vulnerability is also substantially higher: 0.2012072%.

Posterior Vulnerability for Steps C As with the posterior vulnerability for Steps A+B, the posterior vulnerability for Step C is based on a sound analysis using the real results of the workflow step, and are not simulated as in the predictive vulnerability.

Ship #9 Position	$2.049806\text{e}{-4}\%$

In the case of Step C, the posterior vulnerability for Ship #9 is *lower* than the median of the predictive result ($6.644298\text{e}{-4}\%$). From an analyst's perspective, this could mean that Ship #9 has revealed even less about its private data than the 'average' ship would in this scenario.

Why Step D doesn't leak Step D has no meaningful consequences on the vulnerability of the private data for any stakeholder in the HADR scenario *if the result of Step C has been observed by the adversary*. The reason for this is that Step D's algorithm can be computed completely from the results of previous steps in the workflow, i.e. it does not require the values of the private data directly. Interestingly, this point reinforces an important aspect of QIF analysis: even though Step C's result was computed with private data, the vulnerability metrics from the QIF analysis of Step C take

[4]The sampling procedure is time based, hence the seemingly arbitrary number of samples.

[5]Note that the range of the x-axis has changed in order to better display the data.

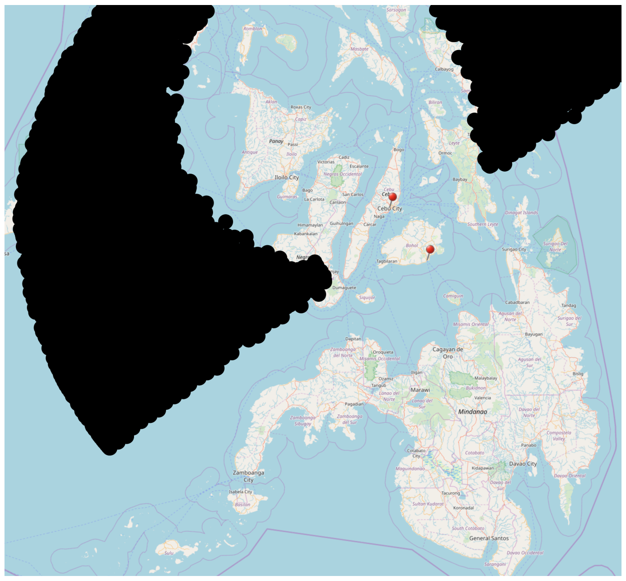


Figure 5: The posterior belief over the position of Ship #9 in the HADR scenario after cooperating in all steps. (Map cartography and tiles © OpenStreetMap contributors.)

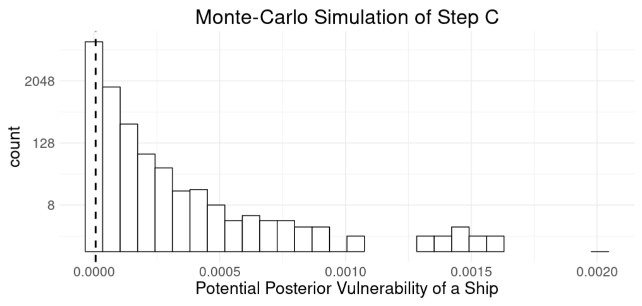


Figure 6: Predictive Vulnerability of Step C

into account *any possible use* of the result. Therefore additional computation over the results of Step C (or any prior step) does not affect the vulnerability of private data.

Privacy-aware Workflow Adaptation

The vulnerability assessment computed by the QIF capability provides insights to data owners as to the security of private information that they wish to protect. We exploit these insights within our workflow manager to adapt the scheduling process in order to ensure adherence to privacy objectives. More specifically, we use the QIF capability in two ways: in a *predictive mode* to estimate the amount of leakage associated with potentially performing a particular query or task and in a *posterior mode* to track actual leakage based on the specific values that a query or task returns.

When executing a particular task our workflow manager invokes the predictive mode to estimate leakage. If the estimate of aggregate leakage for designated private data does not exceed set thresholds, then the workflow proceeds and the posterior leakage analysis is invoked to determine actual leakage values. If the estimate does exceed the threshold then the workflow is either terminated or (if possible) modified via a *remediation strategy* to keep leakage below the threshold.

The idea behind a remediation strategy is to modify the problem or state in ways that will likely reduce impacts on private data. For example, our aid delivery problem requires computing the reachability of a port by a given deadline. Knowing that a given ship can reach the port by that deadline reveals information about the combination of ship position and maximum speed. One simple remediation strategy is to postpone the deadline, which will reveal less information about the position and speed values (e.g., the fact that a ship can reach a port by a given deadline reveals something about the lower bound for its max-speed; a later deadline introduces greater uncertainty as to what that speed might be by decreasing that lower bound). Our Lumen-based adaptive workflow engine includes such remediation strategies to enable adaptivity based on QIF predictive analyses.

Privacy thresholds are implemented using an existing policy framework within Lumen that was developed previously to enable users to impose boundaries on the behaviors of autonomous agents (Myers and Morley 2003). The privacy policies have the general form:

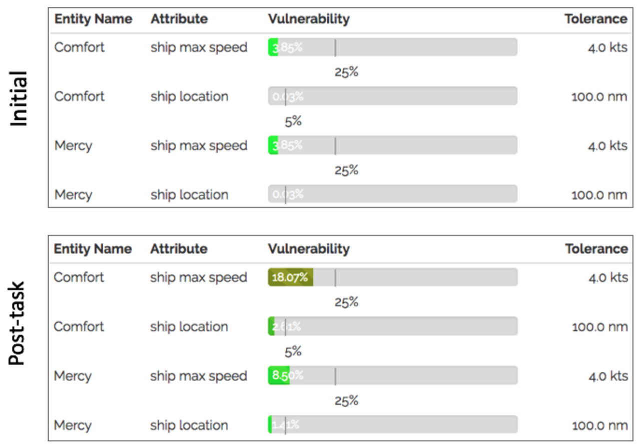
```
Keep below <percentage>
the probability of knowing
<private-data> within <tolerance>
```

Below we show two examples used in the system, one for the Aid Provider nation and one for the Aid Recipient nation.

Aid Provider Sample Policy:

"Keep below 10 % the probability of knowing the location of my ships within 50 NMs"

Aid Recipient Sample Policy:



Initial

Entity Name	Attribute	Vulnerability	Tolerance
Comfort	ship max speed	3.85%	4.0 kts
		25%	
Comfort	ship location	0.03%	100.0 nm
		5%	
Mercy	ship max speed	3.85%	4.0 kts
		25%	
Mercy	ship location	0.03%	100.0 nm

Post-task

Entity Name	Attribute	Vulnerability	Tolerance
Comfort	ship max speed	18.07%	4.0 kts
		25%	
Comfort	ship location	2.61%	100.0 nm
		5%	
Mercy	ship max speed	8.50%	4.0 kts
		25%	
Mercy	ship location	1.41%	100.0 nm

Figure 7: Comparison of initial (prior) and post-task (posterior) vulnerability assessments

"Keep below 20 % the probability of knowing the port harbor depth within 40 feet"

Figure 7 shows sample vulnerability assessments for two types of private data (max-speed, location) for a select set of ships belonging to an individual Aid Provider nation. The top image shows the initial vulnerabilities of the data, prior to performing any computations; the bottom image shows the vunlerabilities after workflow completion. The display shows the QIF-derived vulnerability level as a colored bar representing the adversary's likelihood of guessing the private data within the specified tolerance. The vertical line bisecting the display for each piece of private data marks the policy-prescribed threshold of acceptability for the vulnerability. We note that the initial vulnerabilities for the ship locations are non-zero but so small as to not be perceptible in the image.

Future Work

Here, we consider two avenues for future work.

Analogs can be drawn between our use of the QIF analysis to predict and track vunlerabilities of private data within the scheduling workflow and prior work on estimating resource usage in workflows (Morley, Myers, and Yorke-Smith 2006). Although we currently consider individual actions incrementally as the workflow executes, we envision performing predictive vulnerability assessments of entire workflows prior to execution, to enable informed choices about alternative approaches before any information usage has occurred. Generating useful assessments will require predictive QIF techniques that consider cases beyond worst-case leakage, whose inherent pessimism can make them of limited value for certain vulnerability assessment tasks. Longer term, such analyses could also potentially open the door to using first-principles planning techniques to synthesize privacy-aware workflows on an as needed basis that are tailored to the specifics of a given task and privacy requirements.

The scalability of QIF techniques can be an issue for systems where there are complex relationships between sets of variables. Some work has been done on attaining scalability by enhancing static analysis techniques with approximations that speed up analysis with probabilistic bounds on certainty (Sweet et al. 2018). However, there is still further work required before the QIF analysis of arbitrary channels could scale to use cases such as the end-to-end HADR scenario considered in this paper. In particular, the methods described in this paper utilize bespoke analyses for the channels under consideration, providing a more scalable approach at the cost of generality. One future direction may be to design Domain Specific Languages that enable description of a scenario and its analysis in tandem.

Conclusion

A key challenge facing coalitions is how to collaborate without releasing information that could jeopardize national (or organizational) interests. In this paper, we consider this challenge for a realistic scheduling problem tied to aid delivery. Our work makes several contributions. First, we show how state-of-the-art secure multi-party computation can be used to safeguard private information with an overall distributed scheduling solution to the aid delivery problem. A second contribution relates to the use of quantitative information flow (QIF): even with secure multi-party computation, scheduling outputs can reveal information about coalition members' private data. We show how QIF can be applied to assess the vulnerability of providate data for both prospective (i.e., where results are not known) and actual (i.e., where results are known) computations. As a third contribution, these assessments can be used to adapt the scheduling algorithm to ensure it remains within accepted vulnerability thresholds established by data owners.

Acknowledgments

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA), the United States Air Force, and the Space and Naval Warfare Systems Center, Pacific (SSC Pacific) under Contracts No. FA8750-16-C-0022 and N66001-15-C-4071. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of DARPA, the Department of Defense, the United States Air Force, or SSC Pacific.

The authors thank Stealth Software Technologies, Inc. (in particular; Steve Lu, Rafail Ostrovsky, Paul Bunn, and Chongwon Cho) for providing the MPC software that was used on this effort.

References

[Alvim et al. 2012] Alvim, M. S.; Chatzikokolakis, K.; Palamidessi, C.; and Smith, G. 2012. Measuring information leakage using generalized gain functions. In *2012 IEEE 25th Computer Security Foundations Symposium*, 265–279. IEEE.

[Archer et al. 2018] Archer, D. W.; Bogdanov, D.; Lindell, Y.; Kamm, L.; Nielsen, K.; Pagter, J. I.; Smart, N. P.; and

Wright, R. N. 2018. From Keys to Databases—Real-World Applications of Secure Multi-Party Computation. *The Computer Journal* 61(12):1749–1771.

[Goldreich and Ostrovsky 1996] Goldreich, O., and Ostrovsky, R. 1996. Software protection and simulation on oblivious rams. *J. ACM* 43(3):431–473.

[Goldreich, Micali, and Wigderson 1987] Goldreich, O.; Micali, S.; and Wigderson, A. 1987. How to play any mental game. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, STOC '87, 218–229. New York, NY, USA: ACM.

[Kerschbaum et al. 2011] Kerschbaum, F.; Schroepfer, A.; Zilli, A.; Pibernik, R.; Catrina, O.; de Hoogh, S.; Schoenmakers, B.; Cimato, S.; and Damiani, E. 2011. Secure collaborative supply-chain management. *Computer* 44(9):38–43.

[Morley, Myers, and Yorke-Smith 2006] Morley, D. N.; Myers, K. L.; and Yorke-Smith, N. 2006. Continuous Refinement of Resource Estimates . In *Proceedings of the 5th International Joint Conference on Autonomous Agents and Multi Agent Systems*, 858–865.

[Myers and Morley 2003] Myers, K., and Morley, D. 2003. Policy-based Agent Directability. In Hexmoor, H.; Castelfranchi, C.; and Falcone, R., eds., *Agent Autonomy*. Kluwer. 143–162.

[Myers et al. 2011] Myers, K.; Kolojejchick, J.; Angiolillo, C.; Cummings, T.; Garvey, T.; Gervasio, M.; Haines, W.; Jones, C.; Knittel, J.; Morley, D.; et al. 2011. Learning by demonstration technology for military planning and decision making: A deployment story. In *Twenty-Third IAAI Conference*.

[Phillips, Ting, and Demurjian 2002] Phillips, Jr., C. E.; Ting, T.; and Demurjian, S. A. 2002. Information sharing and security in dynamic coalitions. In *Proceedings of the Seventh ACM Symposium on Access Control Models and Technologies*, SACMAT '02, 87–96. New York, NY, USA: ACM.

[Smith 2009] Smith, G. 2009. On the foundations of quantitative information flow. In *International Conference on Foundations of Software Science and Computational Structures*, 288–302. Springer.

[Smith 2011] Smith, G. 2011. Quantifying information flow using min-entropy. In *Proceedings of the 2011 Eighth International Conference on Quantitative Evaluation of SysTems*, QEST '11, 159–167. Washington, DC, USA: IEEE Computer Society.

[Sweet et al. 2018] Sweet, I.; Calderón Trilla, J. M.; Scherrer, C.; Hicks, M.; and Magill, S. 2018. What's the over/under? probabilistic bounds on information leakage. In *International Conference on Principles of Security and Trust*, 3–27. Springer, Cham.

[Yao 1982] Yao, A. C. 1982. Protocols for Secure Computations. In *23rd Annual Symposium on Foundations of Computer Science (sfcs 1982)*, 160–164.