Multi-cloud Performance and Security Driven Federated Workflow Management

Matthew Dickinson, Saptarshi Debroy, Prasad Calyam, Samaikya Valluripally,
Yuanxun Zhang, Ronny Bazan Antequera, Trupti Joshi, Tommi White, Dong Xu
University of Missouri-Columbia, USA
Email: {*dickinsonmg, debroysa, calyamp, joshitr, whiteto, xudong*}@missouri.edu, {*svbqb, yzd3b, rcb553*}@mail.missouri.edu

***Abstract*—Federated multi-cloud resource allocation for data-intensive application workflows is generally performed based on performance or quality of service (i.e., *QSpecs*) considerations. At the same time, end-to-end security requirements of these workflows across multiple domains are considered as an afterthought due to lack of standardized formalization methods. Consequently, diverse/heterogenous domain resource and security policies cause inter-conflicts between application's security and performance requirements that lead to sub-optimal resource allocations. In this paper, we present a joint performance and security-driven federated resource allocation scheme for data-intensive scientific applications. In order to aid joint resource brokering among multi-cloud domains with diverse/heterogenous security postures, we first define and characterize a data-intensive application's security specifications (i.e., *SSpecs*). Then we describe an alignment technique inspired by Portunes Algebra to homogenize the various domain resource policies (i.e., *RSpecs*) along an application's workflow lifecycle stages. Using such formalization and alignment, we propose a near optimal cost-aware joint *QSpecs-SSpecs*-driven, *RSpecs*-compliant resource allocation algorithm for multi-cloud computing resource domain/location selection as well as network path selection. We implement our security formalization, alignment, and allocation scheme as a framework, viz., "OnTimeURB" and validate it in a multi-cloud environment with exemplar data-intensive application workflows involving distributed computing and remote instrumentation use cases with different performance and security requirements.**

***Keywords*-Data-intensive application workflows, End-to-end security management, Federated resource allocation.**

I. Introduction

Data-intensive science applications in fields such as bioinformatics, material science and high-energy physics are increasingly multi-domain in nature. To augment local private cloud resources, these application workflows rely on multi-institutional resources, i.e., community and public clouds as illustrated in Fig. 1 that are remotely accessible (e.g., scientific instruments, supercomputers, federated data repositories, public clouds). They execute various lifecycle stages and the data may have different security requirements as it undergoes transformations. A growing trend in multi-domain resource federations that support multi-disciplinary initiatives is to combine expertise of geographically distributed collaborators as seen in exemplar application communities such as: (a) Large Hadron Collider for physicists [1], (b) iPlant Collaborative that uses federated resources for informatics [2], and (c) cyber-enabling of expensive scientific instruments (e.g., electron microscopes, spectrometers) in fields such as material science and biochemistry [3]. Thus, secure and efficient allocation of federated multi-cloud resources comprising of multi-institution resources for data-intensive science collaborations in user communities is becoming increasingly critical.

This work was supported by the National Science Foundation under awards: ACI-1440582 and CNS-1429294. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation.

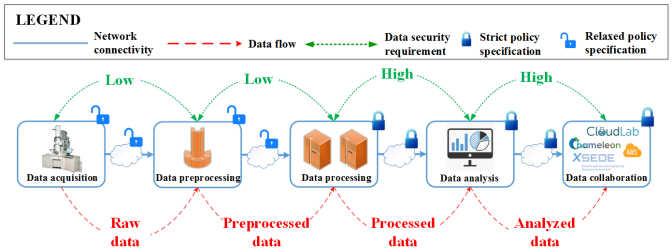


Fig. 1. End-to-end lifecycle stages of a data-intensive application with dynamic security requirements using federated multi-cloud resources from domains with diverse resource policy specifications.

Allocation of such federated multi-cloud resources has been traditionally based on applications' performance and quality of service (QoS) considerations i.e., *QSpecs*-driven (e.g., data throughput, execution time) [4], [5]. Some approaches such as [6], [7] also consider key security and privacy requirements, i.e., *SSpecs* of the applications. However, *SSpecs*-driven approaches produce resource allocation cases that contradict with diverse resource policies i.e., *RSpecs* (ranging from very strict to very relaxed as shown in Fig. 1) for real-time data processing stages. In turn, this may prevent them from using high-performance networks and Science DMZs [8] to access public clouds. They may also force the selection of compliant multi-institutional resources with *RSpecs* that have limited capacity. This in turn may limit the peak performance needed for large-scale data processing in application workflows. Thus, the complex nature of scientific workflows and their *SSpecs*, and *QSpecs* with diverse domain *RSpecs* create conflict factors that can be represented as 'frictions' among the cogs of the 3 metaphorical gears as shown in Fig. 2. Addressing the inability of traditional resource allocation schemes [4] - [7] to manage such frictions is a non-trivial challenge in federated multi-cloud environments. In addition to ensuring satisfactory performance, systematic solutions are needed to supplement current DevOps practices to address security requirements of both the application users as well as resource providers.

In this paper, we present a novel joint performance and security driven resource allocation approach that resolves the gear "frictions" shown in Fig. 2. Our approach involves three main research thrusts: (i) end-to-end security formalization, (ii) security-posture alignment, and (iii) federated multi-cloud resource allocation optimization that leverages private cloud (local) and remote (public and community) cloud resources. Our approach novelty is in the formal definition of *SSpecs* of a data-intensive application for different stages of its lifecycle (as shown in Fig. 1) utilizing resources across federated domains. For this, we extend the National Institute of Standards and Technology (NIST) SP-800E guidelines [9] to define specific security categories to create a formal *SSpecs* data structure that is intuitive and comprehensive.

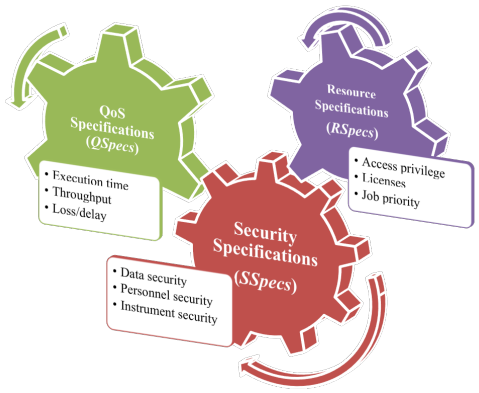


Fig. 2. Inter-conflict between application *SSpecs*, *QSpecs*, and domain *RSpecs* represented as frictions between 3 metaphorical gears.

Building upon the *SSpecs* formalization, we present a security alignment method to align diverse/heterogenous domain security postures into a homogeneous set of formal statements. This approach aids in a resource allocation that is driven by both application *SSpecs* and *QSpecs*. Although there can be overlap and connections between the specifications, we assume the metaphorical gears to be independent at the initial specification step before the alignment. Our alignment novelty is in the use of a security predicate logic, viz., "Portunes Algebra" [10] for automated drill-down of domain-specific stringent or relaxed (as shown in Fig. 1) security and resource policies from security documents into homogenous policies.

Lastly, we propose a novel cost-aware joint *QSpecs*-*SSpecs*-driven, *RSpecs*-compliant federated multi-cloud resource allocation optimization to solve the "frictions" shown in Fig. 2. For this, we use the outputs of our proposed security formalization and policy alignment methods. Our optimization problem involves resource allocation across multiple domain infrastructures with multiple constraints. Such a problem is NP-hard and known to be intractable even for a moderate number of resources, unless approximation or heuristic solutions are acceptable. Our proposed algorithm solves a multi-constrained nested Knapsack problem and takes advantage of the known benefits in considering such a structured problem. Specifically, the algorithm finds a near optimal greedy solution for network path selection, and compute location selection at each stage of application lifecycle in a *RSpecs*-compliant manner.

We operationalize our proposed security formalization, policy alignment, and joint optimization algorithm schemes through a *unified resource broker* (URB) framework viz., "OnTimeURB" for multi-cloud resource management of two exemplar data-intensive applications: (i) a Distributed Computing application viz., Soybean Knowledge Base (SoyKB) [11], and (ii) a Remote Instrumentation application viz., Electron Microscopy Core (EMC) [3]. These exemplar applications feature: (a) varied performance and QoS requirements for different stages of their lifecycles, b) different security requirements with SoyKB requiring data privacy for "data going outside", and EMC requiring instrument protection for "remote access inside", and (c) unique "private cloud – community cloud", and "private cloud – public cloud" domain policy alignment problems for SoyKB and EMC applications, respectively. We demonstrate how the "OnTimeURB" framework can formalize their *SSpecs*, align involved private-community-public cloud domains' *RSpecs*, and perform resource domain/location selection and network path selection with cost constraints.

Finally, we perform experiments to evaluate the QoS performance and security robustness of OnTimeURB resource allocation outcome for the two exemplar applications. Our first performance evaluation experiment is on a real-world SoyKB testbed. It demonstrates how OnTimeURB helps in aligning the domain security postures and pertinent resource domain/location and network path selection amongst candidate private and community cloud domains along the workflow lifecycle. Our second performance evaluation experiment is in an EMC experimental testbed. It demonstrates cost-benefit of joint performance and security-driven resource allocation between private and public cloud domains. We compare results from our joint *QSpecs*-*SSpecs*-driven approach against only *QSpecs*-driven, and only *SSpecs*-driven approaches for federated multi-cloud resource allocation. Our results show that our joint *QSpecs*-*SSpecs*-driven approach results in the most efficient resource allocation through pertinent policy alignment that satisfies application's *SSpecs*, and *QSpecs* without overriding intermediate-domain *RSpecs*. Lastly, we evaluate the robustness of the resource allocation outcome with OnTimeURB using the NIST based risk assessment model [12] for both SoyKB and EMC applications. The robustness results allows users to assess the extent to which these applications are vulnerable to well-known and relevant threats for a given set of selection choices for multi-cloud resource domains and network paths that satisfy the application requirements.

The remainder of the paper is organized as follows: Section II presents the related work categorized by the major contribution thrusts of this paper. Section III introduces the two exemplar data-intensive applications i.e., SoyKB and EMC along with their performance and security requirements. Section IV discusses our proposed scheme of security formalization, policy alignment, and federated resource allocation. Section V discusses the OnTimeURB implementation and exemplar application use case resource allocation results. Section VI concludes the paper and suggests future work.

II. Related Work

A. Security Requirement Formalization and Modeling

In related works for security formalization, domain security assessment has generally been performed by quantification, or risk analysis as suggested by NIST. The current literature lacks solutions to directly compare different domains' security requirements. In [10], a formal approach is proposed to align security policies within an organization between physical, digital, and social domains. The security profiles are updated as wrappers along the organization hierarchy. The alignment of domain security policies is performed informally by authors in [13] to assess organizational goals. Authors in [14] propose a modification of the Klaim language [15] to create high-level security models of systems. Work in [16] discusses policies in terms of sequences of actions; their framework allows refinement of both systems and policies based on UML specifications. However, they do not explicitly address end-to-end security policies spanning multiple domains that affect application performance. We explored the approach in [17], where ontologies are maintained within conceptual models to formalize security within a domain. We also considered the work in [18], where security policies are quantified and formalized. In contrast to these approaches, we found that the Portunes algebra is more comprehensive from the CIA (Confidentiality, Integrity and Availability) security tria perspective. It was also more amenable to adapt with the NIST security control rules. Thus, *our novel scheme uses Portunes algebra and NIST guidelines in order to convert diverse/heterogeneous or even mutually orthogonal domain security postures into a homogenous set of formal statements.*

B. Domain Security Posture Alignment

With regards to the *alignment of diverse/heterogeneous domain-security postures* using a standardized technique, the current literature lacks foundational methods. Traditional models have proposed security methods, such as a state machine

for enforcing access control in order to preserve confidentiality [19] or integrity [20] [21]. Although NIST has provided guidelines for Security and Privacy Controls for Federal Information Systems and Organizations [9], the security postures of SoyKB related institutions such as e.g., University of Missouri (MU) [22], University of Texas at Austin (UT) [23], and University of Southern California (USC) [24] are heterogenous and disproportionate. Institution policy documents range from 5 to 150 pages with limited, and in some cases - no alignment with the NIST guidelines. Prior works such as [25] are similar to our work, but present a top-down approach to implement trust management using declarative language in federated infrastructures. In contrast, *our work presents a bottom-up approach that compares existing institutional policies and aligns them in terms of low-level policy statements to meet security requirements of users using the NIST guidelines.*

C. Traditional Resource Allocation Approaches

In terms of scheduling of data-intensive workflows using traditional QoS-driven resource allocation, different approaches based upon the broad goals of end-to-end performance and dependability are proposed in [4], [5], [26]. Authors in [4] show how QoS requirements are determined by the application traffic type, such as multimedia or file transfer. In our earlier work on ADON (Application-Driven Overlay Network-as-a-service) [5], we used hybrid cloud computing architectures for on-demand and concurrent application handling for accelerated performance of data-intensive application workflows. The authors in [26] deal with end-to-end QoS constraints in resource allocation of web services. In all of these works, security among domains is not explicitly stated within the service level objectives, when choosing amongst a set of resource domains or interconnecting paths. Other works such as [6], [7] extend end-to-end QoS-driven approaches to include security requirements. However, security alignment as a requirement for resource allocation has not been addressed in an end-to-end manner. Many applications opt to instantiate point solutions such as SDN [27], and Network Function Virtualization (NFV) [28] to mitigate attacks such as DDoS [29] along with methods to test the solutions' impact on federated cloud security. *In contrast, our work is unique because we address the end-to-end multi-domain security design by defining and aligning SSpecs of a data-intensive application along its workflow lifecycle stages.*

D. Security Architecture in Data-intensive Communities

Existing works pertaining to *security and dependability for federated multi-cloud resources in data-intensive research communities* mostly deal with security measures and point solutions to counter confidentiality, availability, and integrity threats. They also do not consider end-to-end security design that helps in dynamic allocation and adaptation using such measures. Exemplar solutions to Big Data transfer in a federated environment include Globus efforts [33] that provide the ability to use point solutions such as InCommon [30], OpenID [31] and X.509 [32] to access resources. The Large Synoptic Survey Telescope (LSST) community on the other hand provides detailed guidelines for multi-domain cybersecurity compliance with a list of threat mitigating capabilities at involved domains [34]. *These communities can benefit from our formal approach of resource allocation based on multi-domain security requirements, and augment their current approach of manual co-ordination of policies to achieve end-to-end security alignment.*

III. Performance and Security Requirements of Exemplar Data-intensive Applications

In this section, we present two uniquely different use-cases that motivate our proposed security formalization and alignment methods for joint *QSpecs*-*SSpecs*-driven, *RSpecs*-compliant federated workflow management.

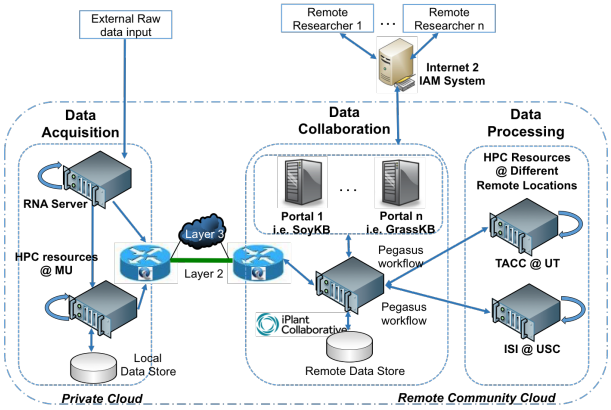


Fig. 3. SoyKB user workflows with different lifecycle stages and federated multi-cloud resource requirements.

A. Distributed Computing Use Case

Knowledge Base (KB) approaches, in areas such as translational genomics and molecular breeding, allow researchers to have a single point of information source. It can enable the collaboration of experts in different areas such as Grass KB (GrassKB), Rice KB (RiceKB), and Soybean KB (SoyKB). An exemplar in this area is the SoyKB [11], which is a comprehensive web resource developed at MU for soybean translational genomics and breeding. The SoyKB application handles the integration of soybean genomics and multi-omics data along with gene function annotations, biological pathway and trait information. The SoyKB has been featured as a model distributed computing use case in the systems biology area within the Open Science Grid community [35].

1) SoyKB performance requirements: The SoyKB workflow relies on the Pegasus [36] Workflow Management System, which splits the workflow into MPI jobs and maps them to available high-performance computing (HPC) resources. The workflow lifecycle stages involve large data sets being transferred to MU for pre-processing. These datasets are subsequently transferred to either a private cloud (at MU) or community cloud sites of XSEDE (Extreme Science and Engineering Discovery Environment) [37] (e.g., ISI [38], TACC [39]) for analysis as shown in Fig. 3. Following computation completion, the resultant processed data with meaningful results (a portion of $\approx$ 50 Terabytes of data accessible via iRODS [40] at iPlant datastore after removing the intermediate analysis files) is transferred to MU, and publicly available for the worldwide user community.

The performance (i.e., *QSpecs*) of the above end-to-end lifecycle stages when handling the datasets (often several Terabytes in size) can suffer from long execution times due to dynamic workloads at HPC sites that lead to long queues, and restrictive resource usage policies given that iPlant is operated as a shared environment supporting several simultaneous users. Various custom metrics at the system, network and application-levels are instrumented with perfSONAR extensions [41] to obtain timely and accurate performance intelligence of workflow status (e.g., log analysis and anomaly event notifications) along the different lifecycle stages. This instrumentation guides optimization decisions that determine: (a) how and where to run the data processing tasks between the various available HPC site choices, and (b) whether data transfers need a dedicated high-speed Layer 2 connectivity, or can go over the regular Layer 3 Internet paths with firewall restrictions.

2) SoyKB security requirements: In order to understand the SoyKB security requirements, we first study the threats pertaining to the multi-cloud resources, software services and user roles relevant to the SoyKB application workflow management. In particular, we model the potential threats in a structured approach using the Microsoft STRIDE [42] model

that helps us to better understand the SoyKB data Confidentiality, Integrity, and Availability threats. The Microsoft STRIDE model has been applied to a number of applications such as, online banking and scientific computing to model and analyze system vulnerability. Due its standardized approach, we use and modify STRIDE as a lifecycle based model to assess the security threats for our exemplar data-intensive application. Following this, we translate them into formal *SSpecs* in a NIST compatible way in order to aid security-driven resource allocation. Readers can refer to Section IV-B for a specific example where ensure NIST compatibility is ensured with TACC policies alignment for the SoyKB application.

We have noted that a more common concern for users relates to the data access levels (i.e., Loss of Confidentiality) for data-in-motion/data-at-rest/data-in-use within the various lifecycle stages. There is concern of unintended users having access to data due to over-provisioning of privileges amongst the various roles (e.g., KB administrator, domain scientist, data consumer). There is also concern on the ability of strangers to make data copies prior to formal publication of manuscripts, which then may reduce the 'value' of the data (considering data is treated as a currency in today's science communities). Domain scientists are also worried about data integrity issues (i.e., Loss of Integrity) when data leaves their institutional boundary and infrastructure is beyond their direct control. There are known cases where data may be corrupted due to administrator error in handling databases, or if a user action exceeds available space for an analysis process. There are always the chances of threats due to malicious attackers who may seek to deliberately compromise data integrity to tarnish a scientist's reputation, or to make social statements on sensitive debate topics in our society (e.g., in context of healthcare, finance or climate change results). Lastly, there is a threat of users not being able to access their data when needed (i.e., Loss of Availability) due to e.g., administrator error involving inadvertent system management actions. Such actions could cause major changes to storage that may not get notified to users in a timely manner, which may then result in partial or full loss of their data. The security threats to SoyKB application at different lifecycle stages are described using STRIDE model in Table I.

B. Remote Instrumentation Use Case

Increased access to high-speed networks and growing data resolutions has made remote access of sophisticated scientific instruments such as microscopes and spectrometers widely-feasible and essential for domain scientists in areas such as biochemistry, and material science/engineering. Cyber-enabling expensive instruments (some could cost several hundred-thousand dollars) via "remote instrumentation" allows remote users to utilize these instruments when they are not being used by local users, or to conduct collaborative studies that require multiple experts at various sites.

1) EMC performance requirements: Fig. 4 shows the data acquisition, analysis and collaboration lifecycle stages that is common for application workflows. The stages pertain to e.g., materials modeling or biological specimen analysis that generate large amounts of raw and processed data sets and image files. Assuming a sample has been loaded into an electron microscope, there are two basic workflow patterns. The first is *remote observation*, where a remote user or multiple remote users only view the instrument-control software screens in real-time. The second workflow pattern is *remote steering*, where one or more remote users also control the instrument using an instrument control-lock passing feature. In addition to access to electron microscopes, remote users may also access the analysis tools and resources located at EMC or at a remote computing facility for *remote analytics*. To support this workflow pattern, data import from the instrument of large data files (e.g., a tilt series data set can be >2GB)

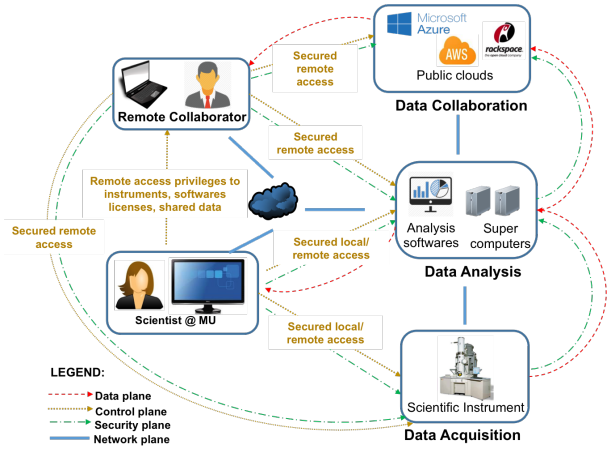


Fig. 4. EMC user workflows with different stages of the lifecycle and annotated with resource security requirements.

to a local or remote compute site with expensive/proprietary and computation-intensive software packages is allowed. The analysis results could add up to several Terabytes of data that needs to be archived, annotated and widely shared amongst the relevant domain science user communities. The unique challenges are to handle the various user roles in these workflows (e.g., system administrator, remote collaborator, remote observer), and managing the various instrument and file system mapping (e.g., one user's files could involve many instruments, and file access permissions vary by user roles).

2) EMC security requirements: Using the extended STRIDE model, we study the security threats to EMC workflow lifecycle stages, especially at the instrument-level to formalize the *SSpecs*. The most common concern again relates to the control of data access levels (i.e., Loss of Confidentiality) within the lifecycle stages. Instrument access conflicts can also occur that invade privacy of users when long-standing passwords of a remote desktop protocol enabled server (mostly maintained and modified manually) are shared with remote users hoping they maintain confidentiality. If undesired password propagation occurs, unauthorized users may access instrument functions without the anticipation or permission of instrument administrators. The instrument administrators are also particularly worried about remote access to local resources or automatic software patches to instrument computers that may break the functioning of the instrument, or cause physical damage to scientific instrument parts. This can in turn be prohibitively expensive to repair/restore, and causes highly undesirable downtime especially in case of popularly used instruments. There are chances of threats of outages (i.e., Loss of Availability) or incorrect data acquisition (i.e., Loss of Integrity) when a user is accessing an instrument remotely and is unfamiliar with the microscope's advanced features; he/she may unknowingly enter a keystroke/mouse-click when the microscope is completing a previous task, which then could cause devastating damage either to the microscope itself or other components, such as the holder or camera. Moreover, multiple concurrent logins to control an instrument remotely have led to cases where the control software auto-locks and crashes occur. This results in loss of data and wasted labor hours, if a user is in the middle of an acquisition or analysis process. The security threats to EMC application at different lifecycle stages are described using STRIDE model in Table II.

IV. Solution Methodology: Formalization, Alignment, and Allocation

In this section, we present the details about our data-intensive applications' security requirements *formalization* approach. We also propose methods to *align* the multi-institution

TABLE I
SOYKB APPLICATION SECURITY THREAT DESCRIPTION FOR DIFFERENT LIFECYCLE STAGES USING A MODIFIED STRIDE MODEL.

Type of Threat	Data Acquisition	Data Processing	Data Collaboration
Identity Spoofing	Illegitimate user accesses data from an archive	Authorized need to process data in computing sites	Access required to store data in repository
Tampering of Data	Modify data that transits the network	During the analysis of the data with the resources	Data transit from the computing sites to the iPlant storage
Repudiation	Not Applicable	Adversary alters performance of resources in computing sites	Not Applicable
Information Disclosure	Data flows in network get exposed if not encrypted	Disclose data using unauthorized access in computing sites	Not Applicable
Denial of Service	Not Applicable	User fails to access resources due to lack of permission	Denial in data transfer to iPlant Storage
Elevation of Privilege	Outsider gains privileged access to compromise data	Access resources and violate permissions	Person with low privileges may claim and access storage

TABLE II
EMC APPLICATION SECURITY THREAT DESCRIPTION FOR DIFFERENT LIFECYCLE STAGES USING A MODIFIED STRIDE MODEL

Type of Threat	Data Acquisition	Data Processing	Data Collaboration
Identity Spoofing	Access data using valid credentials in spite being a impostor	Authentication to access resources in computing sites	To share data login details to public clouds are required
Tampering of Data	Authorized personnel may alter the data in the instruments	Modify the data in transit from instrument to analytical tools	Encryption needed for a data flow from a public cloud
Repudiation	Security measures of an instrument compromised illegally	Analysis on compromised data affects analytical software	Not Applicable
Information Disclosure	Not Applicable	Not Applicable	Not Applicable
Denial of Service	Compromise the instrument illegally and block access	Denial due to lack of privileges need not be by a malicious user	Data transfer denied to public clouds over a public network
Elevation of Privilege	User not supposed to access data from storage/instrument	Data analysis is performed by an underprivileged person	Performance of resources as accessed by privileged user

security policies, and provide an example set of activities that can show how to analyze heterogeneous security policies from multiple domains in a resource federation. Finally, we present our joint *QSpecs*-*SSpecs*-driven and *RSpecs*-complaint federated multi-cloud resource allocation scheme.

Fig. 5 shows the big picture of our overall scheme to customize the collection of user's security requirements in an automated manner for a data-intensive application workflow. It shows a set of high-level steps we envision for our proposed novel method to formalize the security requirements i.e., *SSpecs* by using a predicate logic (Portunes Algebra [10]) and to align diverse domain security policies using NIST guidelines [9] for various application lifecycle stages to aid the overall resource allocation process. Once a resource allocation is determined, risk assessment of the allocation outcome considering the security threats in Tables I and II is performed (as detailed in Section V-C) to check the correctness.

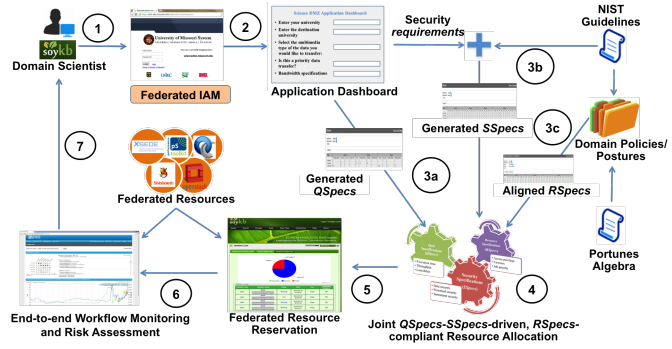


Fig. 5. Overview of security requirements collection, formalization, alignment, and resource allocation steps for data-intensive application workflows.

A. End-to-End Formalization of User SSpecs

As the first step towards joint *QSpecs*-*SSpecs*-driven, *RSpecs*-complaint resource allocation, we define and formalize a data-intensive application's *SSpecs*. Although existing cloud environments, such as the Global Environment for Network Innovations (GENI) [43], make use of *RSpecs* and *QSpecs* to request available resources, definition and characterization of *SSpecs* from an application's security requirements is not addressed directly in the orchestration of the workflow lifecycle. In the following, we list the key challenges in formalizing of security requirements to *SSpecs*.

1) Challenges in formalizing SSpecs: The main challenges in formalizing *SSpecs* of any data-intensive application from the application's data-level and instrument-level requirements can be listed as follows: **(a)** As the size, nature, and structure of data for a data-intensive application changes along the workflows lifecycle, so do the security requirements change associated with the data. Representing such changes in a standardized manner becomes even more complicated when the different stages are executed within multiple domains. **(b)** Not all domains in a resource federation will treat data classification with a unified standard, because domains carry their own classification of data in their security postures. Moreover, not all domains will be willing to accept an overriding security classification standard on top of their own domain posture. **(c)** A data-intensive application's data security classification is generally defined according to the originating domain's security posture. However, due to the diversity of different domain postures, in most cases such classifications become reduced to a conservative setup, with the originating domain bearing the bulk of the risk for ensuring end-to-end security. **(d)** Although security requirements of a data-intensive application are predominantly data-driven, they directly correspond to multi-cloud (network/compute/storage) resources where the data resides. **(e)** For any given stage in a workflow's lifecycle, additional security requirements may be added on top of any initial security requirements, such as scientific instruments generating the data, software analysis tools, and personnel associated and their privileges. The formal security requirements definition needs to have flexibility to accommodate such provisions.

Bearing in mind such challenges, we define *SSpecs* of a data-intensive application as:

Definition 1 (SSpecs): A formal data structure to describe the security requirements of an application for any stage during the workflow lifecycle at the granularity of both the data-level and instrument-level, which includes factors of multi-cloud resources for handling the data, and software tools or personnel for handling scientific instruments i.e., major resources such as spectrometers, microscopes, and telescopes.

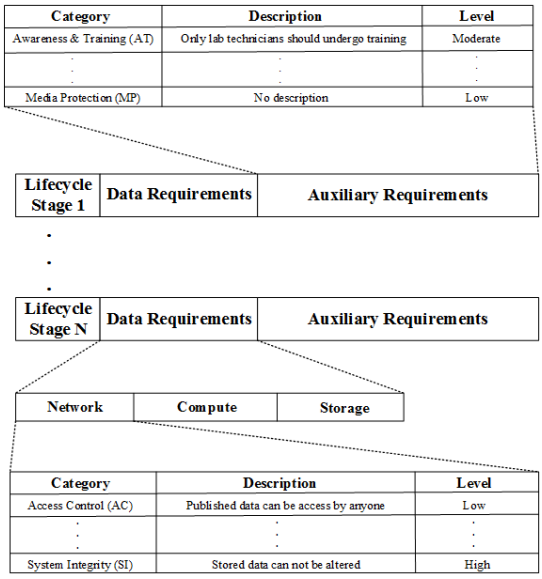


Fig. 6. A data-intensive application's proposed *SSpecs* in terms of data and auxiliary requirements.

Example: Fig. 6 shows a pictorial representation of our proposed *SSpecs* formalization for a typical data-intensive application. The data structure is divided into lifecycle stages with each lifecycle stage further divided into Data and Auxiliary security requirements. The Data requirements are again divided in to specific multi-cloud resource requirements in terms of network, compute and storage resources in order to facilitate joint *QSpecs*-*SSpecs*-driven, *RSpecs*-compliant resource brokering. Both Data and Auxiliary requirements are expressed using security requirement categories that follow the: (a) NIST guidelines, (b) description of the requirement for that category and (c) level (High/Medium/Low) of that requirement.

2) Application workflow lifecycle based SSpecs structure: Every data-intensive application workflow can be viewed as a collection of lifecycle stages such as: acquisition, preprocessing, processing, analysis, and collaboration as illustrated earlier in Fig. 1. Given that the lifecycle stage division is associated with *QSpecs* requirements, our approach is to follow a natural progression for defining *SSpecs* based on the compute, storage and network resources in the different domains involved in the distributed resource allocation.

Defining *SSpecs* on the basis of the application workflow lifecycle serves two purposes. Firstly, a lifecycle based approach obviates the need for having the application user(s) original site's security posture as an overarching influence on the application *SSpecs*. This in turn removes any chance of potential conflict between the data classification rules used to define the *SSpecs* and any other domain's (along the lifecycle) security posture. Secondly, such a lifecycle based characterization makes the addition of *SSpecs* into any resource allocation algorithm that uses *QSpecs* as the differentiating factor in choosing ideal domains and paths.

TABLE III
NIST SP 800E GUIDELINES COMPATIBLE DATA AND AUXILIARY SECURITY CATEGORIES AND FAMILY NAMES

Data Requirements		Auxiliary requirements	
ID	**Family**	**ID**	**Family**
AC	Access Control	AT	Awareness and Training
AU	Audit and Accountability	CM	Configuration Management
CA	Security Assessment and Authorization	CP	Contingency Planning
IA	Identification and Authentication	IR	Incident Response
SA	System and Services Acquisition	MA	Maintenance
SC	System and Communication Protection	MP	Media Protection
SI	System and Information Integrity	PE	Physical and Environmental Protection
		PL	Planning
		PM	Program Management
		PS	Personnel Security
		RA	Risk Assessment

3) Data and Auxiliary Security Requirements based SSpecs structure: For any given stage of the workflow lifecycle, application *SSpecs* is divided into Data and Auxiliary security requirements. The Data requirements are based upon the factors (categories) that directly relate to the data security involving the multi-cloud resources. We use the NIST SP 800E guidelines [9] for the comprehensive list of 18 security and privacy control categories for federal information systems. Table III shows the list of 18 categories divided into Data and Auxiliary security requirements and their corresponding NIST SP 800E descriptions. We remark that the Table III represents a comprehensive list of security related categories/issues that can be adapted for any given data-intensive application workflow.

The category description in the *SSpecs* data structure (as shown in Fig. 6) differs from that of the description in Table III. Descriptions in the data structure represents the security requirements that are specified by the application for that particular category. Depending upon the description, the corresponding category at a particular lifecycle stage is given either a High/Moderate/Low level ranking based on the NIST guidelines. As an example, in Fig. 6, public data accessed by anyone in a particular domain/stage is deemed as Low level security requirement by NIST guidelines. In general, applications do not specify any particular requirements for most categories at different lifecycle stages. Further, if security requirements are not mentioned for any category for a particular lifecycle stage or even for the entire workflow, the security requirement level is set as Default.

Auxiliary requirements are the security specifications imposed on a workflow on top of the Data requirements involving multi-cloud resources. These mainly deal with security requirements that cannot be categorized into network, compute or storage resources. They involve combinations of scientific instruments, analysis tools and related software licenses, and users/personnel requirements of the workflow lifecycle stages. We plan to associate 11 out of the 18 NIST security control categories with Auxiliary security requirements of a data-intensive application. The method of assigning levels for such an Auxiliary requirement category is the same as in the case of the Data requirements discussed in the previous section and shown in Fig. 6. For example, a scientific instrument at lifecycle stage 1 of a data-intensive application might require expert handling by technical staff with sufficient training. However, the security requirements might consider faculty and graduate students as sufficiently trained and only requires technician staff in the lab to be adequately trained. According to the NIST security descriptions from [44], such requirements fall under the category of 'Awareness and training' (AT) and can be ranked as Moderate as shown in Fig. 6.

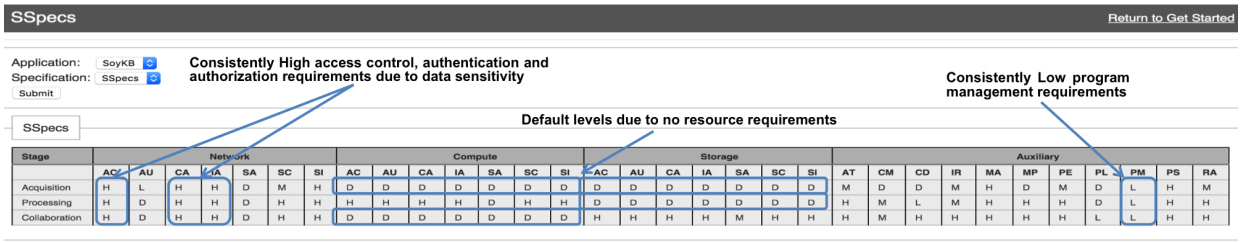


Fig. 7. SoyKB application *SSpecs* for different stages of the application lifecycle.

Similarly, user and personnel related security requirements can also be included into the Auxiliary requirements of *SSpecs*. The reputation/trust of inter- or intra-domain users associated with the workflow is an important factor in deciding whether the security (in terms of confidentiality, integrity and availability) of the overall workflow is being compromised or not. Thus, for each given user associated with any stage of the workflow, the history of the user using that application, or similar applications, or even resources associated with that application could be taken into account. They can be specified as a part of a security category description. It can subsequently be scored with a level according to the NIST SP 800E guidelines. We identify, categories such as 'Personnel Security' (PS) and partly 'Risk Assessment' (RA) as the areas where such user related requirements can be specified. We note that such user related security requirements are not the same as the 'Access Control' (AC) category, which mostly deals with access to data at different lifecycle stages.

Fig. 7 shows an example mockup of our SoyKB *SSpecs* for all the lifecycle stages from the application security requirements. The requirements need to be collected through a careful and relevant discussion with the application users for network, compute, storage resources for different stages of the workflow. At the Acquisition stage, the security requirements mostly pertain to network resources, with compute and storage security requirements set to Default. Compute and storage security requirements tend to be more elaborate for processing and collaboration stages, respectively. Whereas the auxiliary requirements could be more all-encompassing where requirements, such as awareness and training, personnel security, and risk assessment require consistently substantial protection throughout the application lifecycle.

B. Domain/resource policy alignment across multiple domains

One of the major barriers to joint performance and security-driven resource management is the fact that in most cases the domain security postures involving *RSpecs* are diverse. Thus, they cannot be easily compared with application security requirements for compliance. This is especially true for many public and private institutions (especially universities) in the United States who separate their security classification levels for different resources from anywhere between three to six levels that are difficult to align. We analyzed the data classification policies of a selection of universities related to the SoyKB application as shown in Table IV: on one hand, institutions such as, TACC [23], ISI [24] classify the data into three different categories. Whereas, MU [22] on the other hand, classifies data into four different categories.

Given that the security policies for each of these classifications vary between institutions, we address the challenges of ensuring security specification compliance while reserving federated resources through a homogenization method that we developed for meeting needs of both application users and the resource providers. More specifically, we study how inconsistencies between the classification levels of institutions

TABLE IV
COMPARISON OF SECURITY POLICY LEVELS OF DIFFERENT INSTITUTIONS

MU	TACC	ISI
Highly Restricted	Confidential	Highly Sensitive
Restricted	Controlled	Sensitive
Sensitive	Published	Private
Public	-	-

can cause confusion in how the domain resources should be classified. TACC for instance, does not classify any of its information as 'Public', thus making all the resources apparently secured. Whereas the definition of 'Sensitive' at MU is different than the same at ISI. In addition to resource classification, the length and complexity of the security posture differs greatly with one institution having a 7-page generic posture with high-level policies and 7 different categories, whereas another institution having a very detailed posture with intricate policies of over 160 pages and over 30 categories.

Thus, in order to align such diverse/heterogeneous security postures into homogenous policy statements that can make the domain *RSpecs* comparable to the application's *SSpecs* for resource brokering, we propose a 3-step security alignment scheme. Fig. 8 shows our envisioned 3-step scheme, where we first categorize the policies based on the type of resources (i.e., network, compute or storage). Following this, we drill down security policies pertaining to each of the resource types into homogenous formal policy statements using the "Portunes Algebra" [10]. Finally, we assign security levels to each such resources by applying the NIST SP 800E guidelines. The outcome of such a process is a homogenous categorization of different domain *RSpecs* that can be consistent with a data-intensive application's *SSpecs*.

1) Portunes Algebra: Most of the times domain policies pertaining to resources are set at a high-level that need to be drilled-down to low-level formal security statements as shown in Fig. 8. To this end, we use Portunes algebra to perform the first step of Fig. 8, where we align physical, digital, and social security polices within a domain to remove inconsistencies. The alignment converts the diverse/heterogeneous security postures of multiple domains into a homogenous set of formal statements. The Portunes architecture comprises of an environment layer that provides a conceptual overview at a higher-level of abstraction, and a predicate logic that helps expressing high-level policies as low-level symbolic representation of variables and combinations. It divides the environment for which specific policies are defined into three layers: spatial, object and data, as shown in Fig. 9. The spatial layer includes the physical locations of resources specified in a policy; the object layer consists of objects, such as actual resources in policy descriptions; the data layer presents the digital data of interest mentioned in the policies.

Although, the authors in [10] proposed the Portunes algebra to align and find inconsistencies in polices within a domain, our work is the first to use the Portunes framework to drill-down both generic postures (i.e., high level policies, e.g., MU) and fine-grained postures (i.e., low level policies, e.g., TACC).

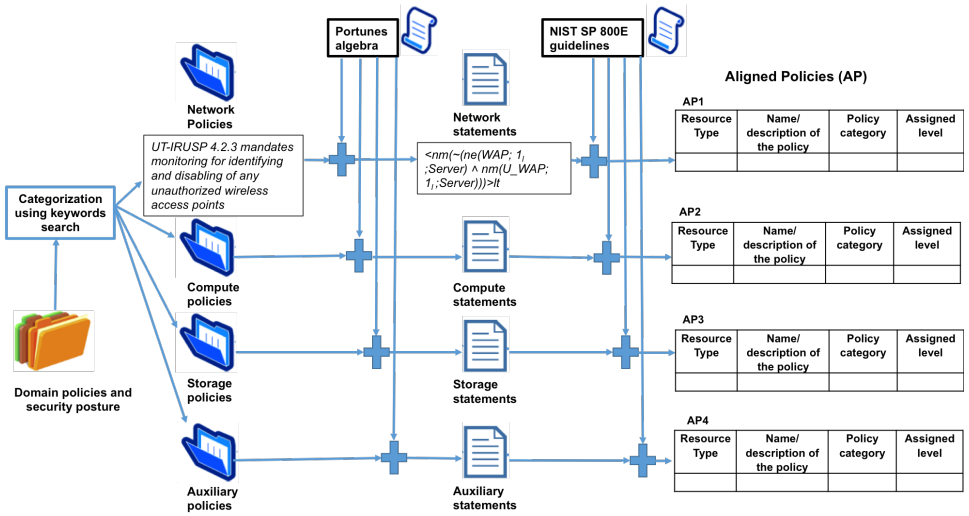


Fig. 8. Logical steps to align diverse domain policies to obtain standardized NIST-compatible security policies.

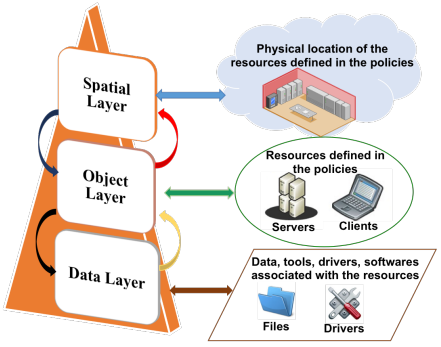


Fig. 9. Logical representation of Portunes algebra with low-level policies described around resources and data in spatial, object and data layers

We convert spatial, object, and data layer specific policies into formal homogenous statements. Such a vertical alignment of policies of each domain using Portunes algebra in turn helps horizontal alignment and homogenization of policies across domains, thus facilitating joint *QSpecs-SSpecs*-driven, *RSpecs*-compliant resource allocation. Below, we discuss using examples, the relevant concepts in applying Portunes algebra to represent simple and complex policies.

Definition 2 (Policy): A policy is a theory Θ in first-order predicate logic, with behaviors $T \in \mathcal{T}$, and $P(\_)$, a distinguished prefix-closed predicate over behaviors.

The formula $P(T)$ means that behavior or action T is permitted or possible; $\sim P(T)$ means that a behavior T is forbidden or impossible. If neither $P(T)$ nor $\sim P(T)$ can be derived from a policy, then the permissibility of T is undecided. In Portunes, a behavior is expressed as a function of actions, and it uses three specific functions, such as *netelav (ne)*, *netmove (nm)*, and *netcopy (nc)* to express all types of action in a policy.

Definition 3 (Neteval): Any action involving delegation of tasks corresponds to the semantic function *neteval (ne)* in Portunes.

To elucidate, when the main server assigns a task to another server, function *ne* is used to express such an action.

Definition 4 (Netmove): Any action involving data movement or transfer from one location to another corresponds to the semantic function *netmove (nm)* in Portunes.

For instance, a *nm* function is used for a server executing login and logout operations, as credential data transfer takes place while login and logout.

Definition 5 (Netcopy): Any action involving data not leaving a location, but simply copied and pasted to another corresponds to the semantic function *netcopy* (nc) in Portunes.

For example, if a server performs a backup operation, function *nc* is used to express such an action.

Definition 6 (Simple policy): A simple policy is a set of sentences Θ of the form $P(T)$ or $\sim P(T)$, with T being a behavior.

A simple policy can be understood as assigning to each behavior a value among: don't care, permitted, forbidden, or contradiction. Many policies allow certain behavior, however they require that a certain result can be achieved in relation to an institution goal. Often, it is not of essential importance how this result is achieved. For example, there should be at least one possible way to change the configuration of an e-mail server. This means that security policies can forbid all but one of the concerned behaviors, as long as this one behavior remains possible. We can thus have a situation where out of a set of behaviors, at least one should be possible.

Example 6.1: Remote access of public data is permitted by using Secure Shell (SSH).

The corresponding Portunes representation of the policy will be $nc(publicData, ssh)$.

Definition 7 (Extended policy): An extended policy Θ is a set of sentences of the form $\phi_1 \vee \phi_2 \vee \cdots \wedge \phi_n$, where each ϕ_i is of the form $P(T)$ or $\sim P(T)$, with T being a behavior, and $\vee$ and $\wedge$ denote conjunction and disjunction of policies, respectively.

The extended policies are only "extended" with respect to simple policies, not with respect to the general notion of policy. Extended policies are a subset of high-level policies to be found in generic postures, and simple policies are a subset of extended policies to be found in detailed postures.

Example 7.1: The server data should never leave a secure server.

The corresponding Portunes representation of the policy will be $nm(serverData, secureServer) \wedge nc(serverData, secureServer)$ assuming *netmove* and *netcopy* being the only two actions permitted on secure server data.

Example 7.2: The server should make sure only a valid user who is authenticated and authorized accesses the system.

The Portunes representation here will be $ne(users, Server) \wedge -ne(authUsers, Server)$, assuming *neteval* is the only action to know whether the person is authenticated.

2) Converting TACC RSpecs into Portunes statements:: Herein, we demonstrate examples of TACC *RSpecs*, i.e., U. of Texas at Austin (UT) security policies (UT-IRUSP) statements being converted into Portunes statements and categorized into network, compute, storage resources, or auxiliary policies. Although the UT-IRUSP statement document is more than 150 pages long, below we show few examples of simple and extended policy statement alignment using Portunes algebra for the SoyKB application workflow.

UT-IRUSP 4.2.3 mandates *monitoring for identifying and disabling of unauthorized (i.e., rogue) wireless access points*. The corresponding Portunes statement is:

$$\langle nm(\sim (ne(WAP, 1_l, Server) \ \vee \ nm(U\_WAP, 1_l, Server)))\rangle l_t$$

where *WAP* is Wireless Access Point and *U_WAP* is unauthorized WAP. The location predicate 1_l denotes that the server can be located at any generic location, and the entire statement is true of all such generic locations signified by the expression l_t. This statement belongs specifically to a network resource.

UT-IRUSP 4.3 recommends that *UT Austin must approve all network hardware connected to UT network system resource in order to ensure integrity*, the corresponding Portunes statement is as follows:

$$\langle \sim nm(System, 1_l, UT\_SytemResources) \vee nm(auth\_sys, 1_p, UT\_SystemResources)\rangle tt$$

Similar to the location predicate, 1_p denotes the process predicate for any generic process, and tt signifies satisfaction of all kinds of predicate logic. This statement can also be categorized exclusively to network resources.

For compute resource specific policies, UT-IRUSP 17.1.3 states *vulnerability assessments are performed annually, at minimum, to identify software and configuration weaknesses within information systems*, yielding the Portunes statement

$$\langle (nc(results, file, 1_l) \wedge ne(assess, software, InformationSystems))\rangle tt$$

An example of a specific policy UT-IRUSP 4.2.1 requires to *establish and communicate to users the roles and conditions under which remote or wireless access to information resources containing confidential data is permitted*, leading to the Portunes statement:

$$\langle \sim ne(person, 1_p, PrimaryServer) \vee nm(person, 1_p, server)\rangle tt$$

The above examples are only a small subset of Portunes statement that we generated from TACC and ISI domain/resource security policies. For resource allocation of SoyKB application using OnTimeURB, we converted all the policies belonging to UT and USC that are relevant to the SoyKB application.

3) NIST conversion: As shown in Fig. 8, once the security postures are translated into Portunes statements, we apply NIST guidelines from NIST SP 800E document [9]. By this, we determine the security level (High/Medium/Low/Default) of each type of resource and for each of the 18 categories. To elaborate, we first take Portunes statements of the security policies of various domains along with the NIST specification of a security rule where we have 25 subrules, and compare these to the Portunes statements considered initially. Based on these steps, we determine whether a chosen security policy belongs to a High, Medium or Low category. We take the help of the detailed descriptions provided in [9] and perform our proposed 3-step scheme to get the aligned *RSpecs* of a domain that can be easily compared with application *SSpecs* for joint *QSpecs*-*SSpecs*-driven, *RSpecs*-compliant resource allocation.

C. Resource allocation optimization

As the final step of our solution approach (Step 4 in Fig. 5), we model the joint *QSpecs*-*SSpecs*-driven, *RSpecs*-compliant federated resource allocation problem along the lifecycle stages of a data-intensive application as a 0-1 Knapsack problem that will: (a) be practical, and easy to operationalize in practice, (b) consider the overall budget of the domain scientists for resource allocation, (c) take into account the cost of resource reservations in different domains, given that cost to the researcher will be different at private, community and public clouds, and (d) ensure non-polynomial time complexity.

The simplistic 0-1 Knapsack problem formulation is as follows:

$$maximize \sum_{d \in D} R_d \times x_d$$

$$subject\ to \sum_{d \in D} C_d \times x_d \leq B \ and \ x_d \in \{0, 1\}$$

where R_d is the resource allocation from domain d, C_d is the cost of the corresponding resources from domain d, D is the total set of the candidate domains, and B is the overall budget of the domain scientist for resource allocation.

The above simplistic formulation becomes non-trivial to solve due to the following constraints. Number of domains, the domains involved, and their corresponding resource types vary between different stages of the application lifecycle. Hence, the entire private, community, and public cloud domain pool D is divided into overlapping domains sets D=$\{D_1, D_2, \cdots, D_N\}$ based on N lifecycle stages. At the same time, the allocated resource R_d from a domain $d \in D$ at any lifecycle stage is a vector represented in terms of network, compute and storage resources that should satisfy the *QSpecs* of the data-intensive application. However, the *QSpecs* of any application is an end-to-end requirement and is represented as a vector of relevant performance and QoS metrics such as execution time and end-to-end throughput. Thus, the mapping of end-to-end *QSpecs* into lifecycle stage resource allocations and subsequent optimization for the entire lifecycle engenders new challenges.

Furthermore, at each stage of the lifecycle, the resource allocation outcome should satisfy both the application's *SSpecs* which can be achieved by considering only certain domains whose aligned *RSpecs* satisfy application's requirements. Thus, we design the entire problem as a multi-constrained nested Knapsack problem for each stage of the application lifecycle. We consider the resources, security, and budget requirements of the applications as well as the resource, security, and cost constraints of the domains. We remark that our approach requires *SSpecs* to be applied for each of the different lifecycle stages separately. This approach is beneficial particularly in cases where each lifecycle stage is handled by different domains with heterogeneous resource policies. In addition, each lifecycle stage might involve different sets of resources, and thus it is relatively simplistic to apply *SSpecs* separately for each of the lifecycle stages.

Therefore, the new multi-constrained nested Knapsack problem can be formulated as:

$$maximize \sum_{i=1}^{N} \sum_{d \in D_i} R_d \times x_d$$

$$subject\ to \sum_{d \in D} C_d \times x_d \leq B$$

$$\& \quad x_d \in \{0, 1\}$$

$$\& \quad SecLevelOf(d) \odot SecLevelOf(a) = 1 \ \forall \ x_d = 1$$

Operation $\odot$ on $SecLevelOf()$ function with domain *RSpecs* and application *SSpecs* as arguments is assumed to ensure conformity. Although this is a NP-hard problem, close to optimal solution can be achieved through well-known dynamic programming and greedy approaches. Our proposed

cost-aware joint *QSpecs*-*SSpecs*-driven, *RSpecs*-compliant optimization algorithm shown in Algorithm 1 tries to solve the 3-gear optimization problem shown in Fig. 2 with a close to optimal greedy approach. The algorithm takes an application's *SSpecs* SS_l, *QSpecs* QS_l for each lifecycle stage l and different domain's resource availability RA_d, and aligned *RSpecs*/domain policies DP_d as inputs where $d \in D$, and D is the set of domains. The outcome of the algorithm is a resource allocation vector A_a of an application a in terms of network, compute, and storage resources at optimal private, public, and community cloud locations. This outcome is based on the different lifecycle stages and corresponding cost C_a that satisfies both the *SSpecs* and *QSpecs*, and is within the overall application resource budget B.

Algorithm 1 follows a greedy heuristic where the available resources at each domain are checked for satisfiability of the application QoS requirements. If satisfied, then each of the network, compute, storage, and auxiliary security requirements of the application *SSpecs* is compared with the aligned *RSpecs* of the domain. If the domain *RSpecs* have equal or higher security level than the corresponding application *SSpecs*, then that domain is considered to satisfy the application *SSpecs*. This way when both the *QSpecs* and *SSpecs* requirements' satisfiability are evaluated for all the domains, the domain offering the minimum resource cost is chosen to be the destination domain for that particular lifecycle stage. Similarly domains with minimum cost but satisfying the requirements are chosen for each lifecycle stage. The total cost of allocation is thus minimized, with application requirements also being satisfied. Such greedy allocation may not always guarantee the maximum amount of resources allocated; however, it can ensure user satisfaction and near optimal allocation. Next, in Section V, we show the implementation outcome of this algorithm for exemplar data-intensive applications.

Algorithm 1: Cost aware joint optimization based resource allocation algorithm

Data: *SSpecs* $SS_l = \{S_l^N, S_l^C, S_l^S, S_l^A\}$ of application for each lifecycle stage l
Data: *QSpecs* $QS_l = \{Q_l^N, Q_l^C, Q_l^S\}$ of application for each lifecycle stage l
Data: Overall resource budget $\mathcal{B}$ of the application
Data: Resource availability $RA_d = \{R_d^N, R_d^C, R_d^S\}$ of each candidate domain d
Data: Unit cost $C_d = \{C_d^N, C_d^C, C_d^S\}$ of each candidate domain d
Data: Aligned *RSpecs* or domain policies $DP_d = \{P_d^N, P_d^C, P_d^S, P_d^A\}$ of each candidate domain d
Result: Resource allocation A_a and total cost C_a of application a
for *each lifecycle stage* $l \in L$ **do**
  /* Finding the domains that satisfy *QSpecs* and *SSpecs* */
  for *all candidate domains* $d \in D$ **do**
    if $(R_d^N \geq$ *resourceEquv*(Q_l^N) && $R_d^C \geq$ *resourceEquv*(Q_l^C) && $R_d^S \geq$ *resourceEquv*$(Q_l^S))$ **then**
      if $($*SecLevelOf*$(S_l^N) \geq$ *SecLevelOf*(P_d^N) && *SecLevelOf*$(S_l^C) \geq$ *SecLevelOf*(P_d^C) && *SecLevelOf*$(S_l^S) \geq$ *SecLevelOf*(P_d^S) && *SecLevelOf*$(S_l^A) \geq$ *SecLevelOf*$(P_d^A))$ **then**
        $A_d^N =$ *resourceEquv*(Q_l^N);
        $A_d^C =$ *resourceEquv*(Q_l^C);
        $A_d^S =$ *resourceEquv*(Q_l^S);
      else
        $A_d^N = 0$; $A_d^C = 0$; $A_d^S = 0$;
    else
      $A_d^N = 0$; $A_d^C = 0$; $A_d^S = 0$;
    $A_d = \{A_d^N, A_d^C, A_d^S\}$;
    $\mathcal{A}_t = append(A_d)$
  end
  /* Selecting a domain by minimizing unit cost */
  $A_f =$ *minimize*$(\mathcal{A}_t, C_d)$;
  $A_l = append(A_f)$;
  $C_f = CostOf(A_f)$;
  $\mathcal{C} = \mathcal{C} + C_f$;
end
if $\mathcal{C} \leq \mathcal{B}$ **then**
  Return $A_a = \{A_l\}$;
  Return $C_a = \mathcal{C}$;
else
  Return Failure;

V. OnTimeURB Implementation and Evaluation

In this section, we first discuss the implementation of the proposed security formalization, policy alignment, and joint *QSpecs*-*SSpecs*-driven, *RSpecs*-compliant federated resource allocation schemes in our OnTimeURB framework. Next, we will discuss the use of the framework to provision federated resources for distributed computing (i.e., SoyKB) and remote instrumentation (i.e., EMC) use cases and evaluate the resource allocation outcome performance results. Finally, we will discuss the results of NIST based risk assessment and robustness evaluation of our resource allocation outcome.

A. OnTimeURB Implementation

We implement our proposed security formalization, policy alignment, and optimization schemes shown in Fig. 5 by developing software elements and tools as part of a *unified resource broker* framework, viz., OnTimeURB. It intelligently uses multi-cloud resources, and point solutions to dynamically manage and adapt federated resources in an agile, timely and policy-compliant manner. It does so by implementing *SSpecs* formalization using "*SSpecs* Generator", *RSpecs* alignment using "*RSpecs* Generator", and joint optimization algorithm using "Optimization and Orchestration" modules. OnTimeURB modules such as "'Network Controller" and "Compute Controller" allocate network and compute resources based on the optimization outcome. OnTimeURB is built using RESTful APIs [45] that are modular, and interoperable with common data-intensive application deployments. This design approach also enables OnTimeURB to be integrated with e.g., popular Shibboleth-based [46] federated authentication and authorization frameworks.

B. Resource Allocation and Performance Evaluation

Herein, we discuss the resource allocation outcome for SoyKB and EMC applications as part of the OnTimeURB framework evaluation.

1) SoyKB performance evaluation: We use our OnTimeURB framework to provision resources for the SoyKB application, specifically for compute location selection for the Processing stage and ensuing network path selection to the final storage facility at iPlant. This is because the SoyKB application mandates data to be collected at MU at the Acquisition stage and final public access from the iPlant data store. The three choices for compute location selection were: a private cloud at MU, and HPC clusters at community clouds such as: TACC, and ISI. The outcome of OnTimeURB joint *QSpecs*-*SSpecs*-driven resource allocation scheme is shown in Fig. 10 where TACC community cloud resources are chosen over MU and ISI for data processing. Fig. 10 also shows the outcome of our previous ADON [5] based resource allocation that uses a traditional *QSpecs*-driven allocation, and only *SSpecs*-driven allocation. The figure shows that in case of ADON, local MU private cloud resources are chosen over community cloud resources, and for only *SSpecs*-driven allocation remote community cloud at ISI is chosen over TACC.

Figs. 11(a) and 11(b) show the reason behind the choice of MU private cloud over community cloud with ADON-based *QSpecs*-driven allocation. Fig. 11(a) shows that due to local resource availability and high priority task scheduling at local site (higher queue), the compute time for different sizes of soybean genomic data at MU private cloud is much faster than community clouds, such as TACC and ISI which are quite similar in resource characteristics. Fig. 11(b) shows longer latency between local MU site to iPlant community cloud; however data movement within community clouds (TACC, ISI, and iPlant) is much faster due to higher bandwidth availability. Finally, Fig. 11(c) presents the overall data transfer time showing individual portions of times responsible for data

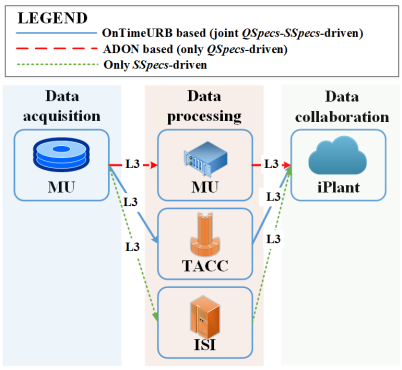


Fig. 10. Comparing network path and domain selection outcomes for different lifecycle stages for SoyKB workflow between only *QSpecs*-driven, only *SSpecs*-driven and joint *QSpecs*-*SSpecs*-driven allocation.

transfer: (a) from local site to HPC site, and (b) from HPC site to iPlant storage. It is important to note that the majority of the transfer time for MU is comprised by time taken to transfer genome data from local machine to MU private cloud which is due to the last mile problem. The overall longer transfer time for private cloud can also be attributed to the lack of sophisticated data management tools (e.g., iRODS with parallel file transfer functions) availability resulting in longer transfer times. Although transfer time performance for MU private cloud is worse than to other community clouds, the scale of transfer time deterioration (in seconds) is no match to the scale of improvement in compute time (in minutes). Therefore, the overall end-to-end execution time remains superior at MU and consequently the *QSpecs*-driven resource allocation chooses MU private cloud over community clouds at TACC or ISI.

However, Fig. 12 shows a different story as far as satisfying the SoyKB *SSpecs* is concerned. We can particularly note the three domains' aligned NIST-inspired *RSpecs* for network, compute and auxiliary security policies (no storage, as it is irrelevant for Processing stage) and their compliance with SoyKB *SSpecs* for Processing stage (as shown in Fig. 7). The levels representing lower and higher (or equal) than the SoyKB application's *SSpecs* levels for different categories are shown using annotations in Fig. 12. We see that MU, due to its low-granularity and high level policies and small number of security APIs, exhibits less than required security standards and a lack of transparency against possible threats. Whereas, community clouds, such as TACC and ISI that are designed for remote HPC collaboration, with more descriptive and fine-grained security postures, support all of the SoyKB security requirements. Therefore OnTimeURB resource allocation algorithm (Algorithm 1) chooses TACC as the final Processing site due to two reasons: (i) compliance with SoyKB *SSpecs*, and (ii) marginally better performance than ISI in terms of end-to-end execution (compute and transfer time) time.

Note that only *SSpecs*-driven allocation chooses ISI over TACC because of the fact that ISI *RSpecs* are more stringent in a few categories than TACC (annotated in Fig. 12) making ISI a marginally better option than TACC in terms of security. This proves the counter-intuitive argument that remote community clouds, such as TACC and ISI that are tailor-made for data-intensive computation such as SoyKB, are more suited to satisfy both performance and security requirements over local campus private clouds.

2) EMC performance evaluation: In the second experiment, we use the OnTimeURB framework to perform an image processing task at EMC. The two candidate domains for the experiment were a private cloud at MU and a remote public cloud instance at Amazon Web Services (AWS) [47]. This setup was based on a cost calculation that considered the overall budget set aside for the image processing task. The image processing is performed using IMOD software suite [48], which is a set of image processing, modeling and display programs used for tomographic reconstruction and for 3D reconstruction of EM serial sections and optical sections. We used the `newstack`' command in IMOD which takes raw image data from the microscope and converts it into an image stack file. Size of image data varies depending on the raw image resolution and pixel count. For the experimental setup, we installed GPU instances at both private cloud (MU) and public cloud (AWS) because the IMOD program is built with CUDA. For the local MU private cloud, we used a single GPU instance as the cost of computation does not vary significantly; whereas at AWS, we used 2 GPU instances with varying computational capability (AWS has only 2 GPU instances in comparison to 20 CPU instances) and prices. The standard instance (AWS-Ins2) was priced at $0.65/hr, whereas the faster instance (AWS-Ins1 with 4x speedup) that was priced at $2.60/hr.

The outcome of OnTimeURB joint *QSpecs*-*SSpecs*-driven resource allocation scheme is shown in Fig. 13 where AWS-Ins2 is chosen over MU and AWS-Ins1. Fig. 13 also shows the outcome of our previous ADON [5] based resource allocation that uses a traditional *QSpecs*-driven allocation, and only *SSpecs*-driven allocation. The reason for such an outcome can be explained with results shown in Fig. 14 where we see that the computation time for both AWS instances are higher than that of the MU private cloud for all sizes of raw image files, with AWS-Ins1 being marginally better than AWS-Ins2. Thus, in terms of only performance, i.e., ADON based allocation, AWS-Ins1 is chosen. Now in terms of security, public cloud resources (AWS-Ins1 or AWS-Ins2) having better classification of security policies and more fine grained posture become an obvious choice for only *SSpecs*-driven allocation. However, OnTimeURB chooses AWS-Ins2 for two reasons: (i) AWS better satisfies EMC application's security requirements than MU private cloud, and (ii) AWS-Ins2 guarantees almost similar computation time with AWS-Ins1 for all sizes of raw image file, however, at a much lower (4x) cost. Overall, the EMC application performance results demonstrate the utility of a public cloud i.e., AWS to provide better performance and security over local private clouds, however at a cost to the domain scientists, which at times can be a significant amount depending on the size and type of data being processed.

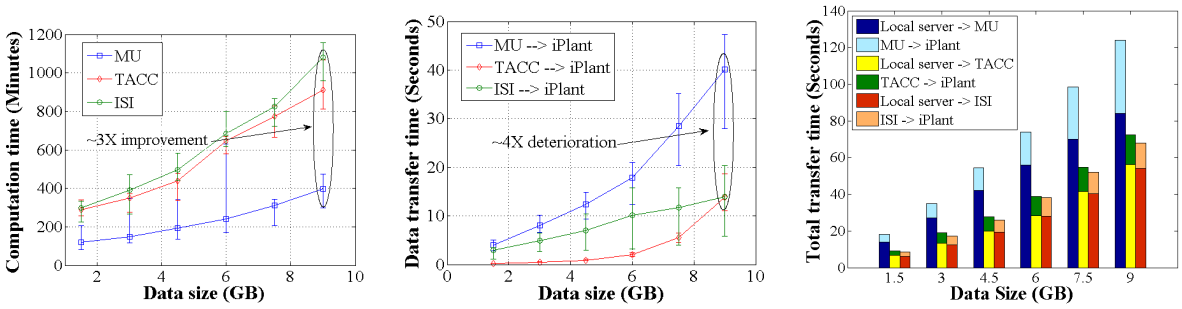
C. Security Risk Assessment and Robustness Evaluation

In order to evaluate the robustness of the OnTimeURB optimization and domain selection outcome against well known and relevant cyber attack threats, we use the NIST [12] based risk assessment method.

1) Assessment methodology: The NIST [12] method for conducting risk assessments is a widely accepted procedure to analyze the security robustness and dependability of a system. The risk assessment study allows us to evaluate the security robustness of our joint *QSpecs*-*SSpecs*-driven, *RSpecs*-compliant federated resource allocation for the two exemplar data-intensive applications i.e., SoyKB and EMC. The risk calculation from a threat event using the NIST method is shown in Fig. 15, and involves the following steps:

- Assess the likelihood of threat occurrence on basis of probability of initiation and success.
- Assess the level of Impact in event of a successful attack.
- The overall Risk score is a combination of the likelihood and impact.

Of all the possible threat events in the NIST guidelines, we identified six events with one for each STRIDE model threat category that are potentially 'High' to 'Moderate' security risks (based on 'Impact' values) to a candidate domain for the



(a) Compute time comparison (b) Processed data transfer time comparison (c) Overall transfer time comparison

Fig. 11. SoyKB application performance comparison at different processing locations for different sizes for genome data.

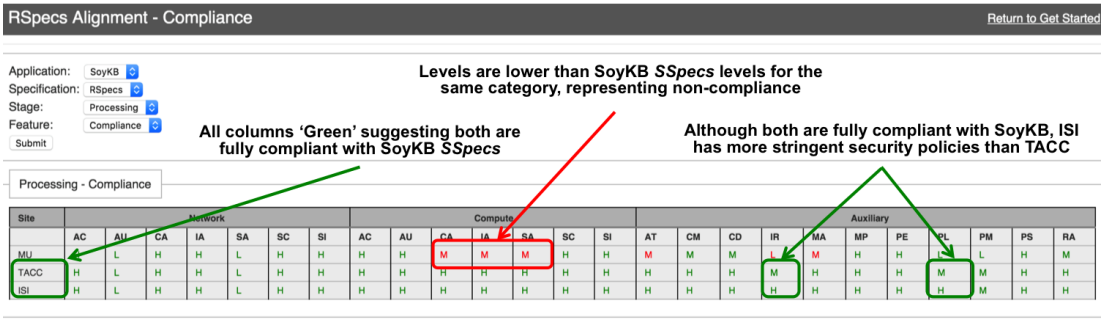


Fig. 12. *RSpecs* alignment outcome of SoyKB application candidate sites/domains and their annotated compliance and non-compliance with SoyKB *SSpecs*

.

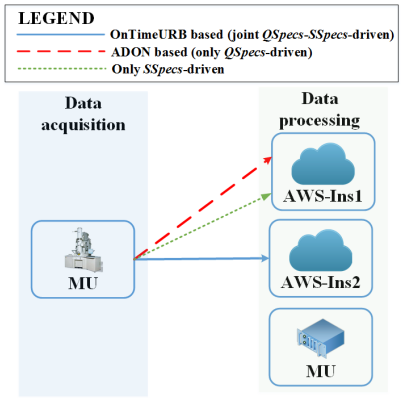


Fig. 13. Comparing image processing selection outcomes for computation stage for EMC application between only *QSpecs*-driven, only *SSpecs*-driven and joint *QSpecs-SSpecs*-driven allocation.

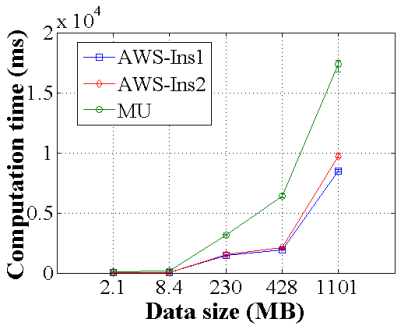


Fig. 14. EMC application image processing performance comparison at MU (private cloud) and AWS (public cloud) sites for different image file sizes.

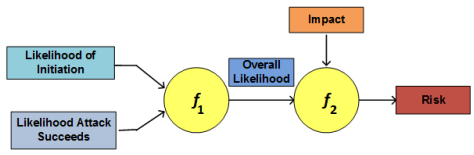
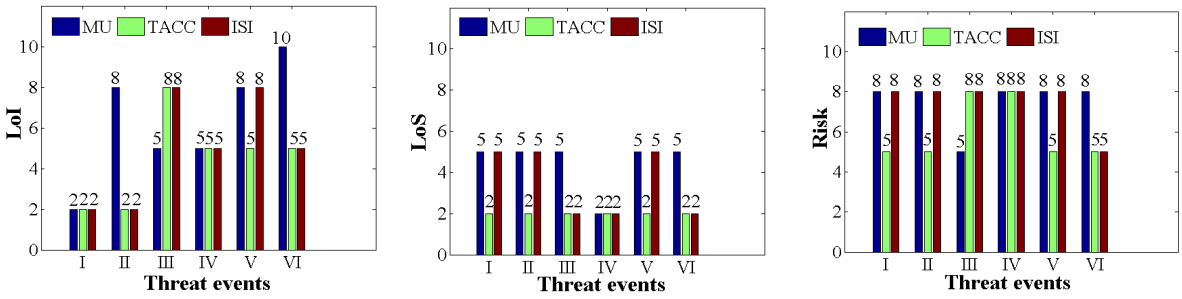


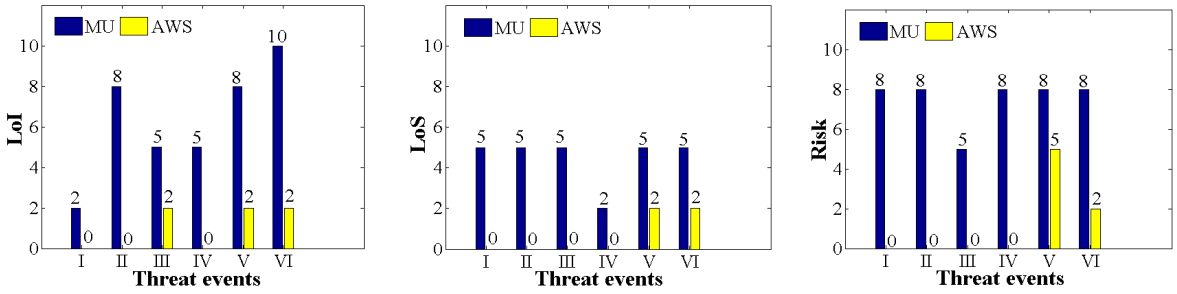
Fig. 15. NIST based Risk assessment model from likelihood and threat impact factors.

SoyKB and EMC applications. Table V shows the NIST definitions of such events along with SoyKB and EMC application relevance, corresponding STRIDE model threat category (from Tables I, and II), and relative impact on the applications. We assess the security robustness of the candidate domains against each of the six events for the two exemplar data-intensive applications, i.e., MU, TACC, and ISI for SoyKB, and MU, and AWS for EMC. We use a pre-defined semi-quantitative scale of 0-10 as guided by NIST for the impact/likelihood event assessments, with 10 indicating *very high*, 8 indicating a *high*, 5 indicating a *moderate*, 2 indicating a *low*, and 0 indicating *very low* levels of impact. For the assessment, the three basic attack variables, e.g., Likelihood of Initiation (LoI), Likelihood of Success (LoS), and Impact of the attack are assigned values using the NIST guidelines. The final Risk value is calculated using the method illustrated in Fig. 15. In the figure, f_1 is defined as a $max()$ function with the likelihoods as arguments. Whereas, f_2 is an $\lceil avg() \rceil$ function between overall likelihood and impact. The only exception condition for the function is as follows: if any basic variable value is 0, then the overall Risk for that scenario becomes 0, as the risk for such event is completely nullified. Any value upon calculation is rounded off to the nearest upper bound integer value among the five possible pre-defined values. The rational behind such a calculation is to get the most conservative estimate of the candidate domains' security robustness.



(a) Likelihood of initiation comparison (b) Likelihood of success comparison (c) Overall risk comparison

Fig. 16. SoyKB application robustness evaluation and domain choice comparison for different threat events.



(a) Likelihood of initiation comparison (b) Likelihood of success comparison (c) Overall risk comparison

Fig. 17. EMC application robustness evaluation and domain choice comparison for different threat events.

TABLE V
NIST BASED THREAT EVENTS SHOWING THE RELEVANCE TO SOYKB AND EMC APPLICATIONS WITH STRIDE CATEGORY AND IMPACT

No.	NIST Threat Events	SoyKB relevance	EMC relevance	STRIDE category	Impact
I	Craft counterfeit certificates	Unauthorized access to HPC data processing site	Unauthorized remote access to scientific instrument	Identity Spoofing	High (8)
II	Deliver targeted malware for data exfiltration	Perform illegal data transfer from HPC to compromised site	Perform illegal data transfer from acquisition to compromised site	Data Tampering	High (8)
III	Perform network sniffing of exposed networks	Access data in transit to get a knowhow of soft-spots	Unauthorized remote sharing users accessing analyzed data in transit	Repudiation	Moderate (5)
IV	Obtain sensitive publicly available data	Access data from public clouds to find sensitive information	Remote users access sensitive data from collaboration site	Information Disclosure	High (8)
V	Conduct simple Denial of Service (DoS) attack	HPC resources made unavailable to legitimate users	Exhaustion of analysis software licenses by irregular means	Denial of Service	Moderate (5)
VI	Exploit physical access of authorized staff	Tailgate authorized users to gain access to HPC resources	Gain instrument login information from authorized users	Elevation of Privilege	Moderate (5)

2) Robustness evaluation: The assessment results comparison for SoyKB application are shown in Figs. 16(a), 16(b), and 16(c), where it is evident that the overall risk at community clouds such as TACC, or ISI is at least as high as MU private cloud; for most threats, community cloud turns out to be more secured. The reason for such an observation is the fact that community clouds such as TACC and ISI are designed for secure computation with detailed and fine-grained security postures at UT and USC having precautions for many possible threats. For example, the TACC and ISI have clearly laid out policies regarding protection against malware installation (Event II with High impact) with precautionary measures that makes it difficult for adversaries to initiate malware installation for data exfiltration (LoI at TACC and ISI is 2, with MU being 8). Similarly, the robustness evaluation for EMC application shown in Figs. 17(a), 17(b), and 17(c) demonstrates the robustness of a public cloud such as AWS with 18 security APIs against possible threats to data-intensive computation. It is interesting to see that the robustness of a public cloud is greater than community clouds because public clouds are typically designed for more mission-critical applications in both scientific and corporate cases with high stakes. Moreover, premium prices paid for such public cloud resources enable the cloud service providers to install state-of-the art security solutions and stringent resource policies.

VI. CONCLUSION

In this paper, we motivated the need to formalize *SSpecs* of distributed applications and align domain *RSpecs* for an efficient joint performance and security driven workflow management across federated multi-cloud resources. We showed how a process of breaking down the security requirements across workflow lifecycle stages and applying NIST based categorization can facilitate formalization of application *SSpecs*. Our formal *SSpecs* data structure is intuitive and comprehensive enough to account for a wide range of security requirements pertaining to data-intensive application workflows. Our unique use of Portunes algebra to align diverse domain postures resulted in homogenizing domain *RSpecs* that is easily comparable with a data-intensive application's *SSpecs* to achieve joint *QSpecs-SSpecs*-driven, *RSpecs*-compliant resource allocation. Our modeling and solution of the joint optimization problem achieves close to optimal resource allocation of federated resources across multiple domains.

Our implementation of OnTimeURB and multi-cloud environment evaluations with the SoyKB and EMC applications demonstrated the benefits of our proposed approach.

We ensured satisfaction of both performance and security requirements, without overriding any domain policies to gain performance advantages. The results of this study inform the tradeoffs for a data-intensive application user to make decisions on scale out/up from private to public, and community clouds to improve both performance and security. Our work advances the current knowledge on how to intelligently perform resource allocations among private, community and public cloud locations to reduce turnaround times in a secured and policy-compliant manner. The data-intensive application communities can benefit from our novel approach of resource allocation, and augment their current techniques of excessive manual co-ordination of policies.

In the future, we plan to study how our end-to-end security design schemes can be integrated rapidly in application development efforts utilizing federated resources. In particular, we see opportunities in integration within multi-cloud resource brokers that implement distributed trust with different personnel roles. We also plan to extend our schemes' relevance to other data-intensive enterprise application use cases in fields of e.g., cybermanufacturing and neuroscience.

REFERENCES

[1] CERN Large Hadron Collider Computing Grid - http://wlcg.web.cern.ch.
[2] The iPlant Collaborative: Cyberinfrastructure for Plant Biology - http://www.iplantcollaborative.org.
[3] University of Missouri Electron Microscopy Core Facility - http://emc.missouri.edu.
[4] W. Kim, P. Sharma, J. Lee, S. Banerjee, J. Tourrilhes, S-J. Lee, and P. Yalagandula, "Automated and Scalable QoS Control for Network Convergence", *Proc. of ACM INM/WREN*, 2010.
[5] R. B. Antequera, P. Calyam, S. Debroy, L. Cui, S. Seetharam, M. Dickinson, T. Joshi, D. Xu, and T. Beyene., "ADON: Application-Driven Overlay Network-as-a-Service for Data-Intensive Science", *IEEE Trans. on Cloud Computing*, 2016.
[6] C. Irvine,T. Levin, "Quality of Security Service", *Proc. of Workshop on New Security Paradigms*, 2000.
[7] S. Lindskog, "Modeling and Tuning Security from a Quality of Service Perspective", *PhD Thesis*, Chalmers Univ. of Tech., 2005.
[8] E. Dart, L. Rotman, B. Tierney, M. Hester, J. Zurawski, "The Science DMZ: A Network Design Pattern for Data-Intensive Science", *Proc. of IEEE/ACM Supercomputing*, 2013.
[9] "Security and Privacy Controls for Federal Information Systems and Organizations", *NIST SP800-30 Technical Report*, 2013.
[10] W. Pieters, T. Dimkov, D. Pavlovic, "Security Policy Alignment: A Formal Approach", *IEEE Systems Journal*, Vol. 7, No. 2, pp. 275-287, 2013.
[11] T. Joshi, K. Patil, M. R. Fitzpatrick, L. D. Franklin, Q. Yao, J. R. Cook, Z. Wang, M. Libault, L. Brechenmacher, B. Valliyodan, X. Wu, J. Cheng, G. Stacey, H. T. Nguyen, D. Xu, "Soybean Knowledge Base (SoyKB): A Web Resource For Soybean Translational Genomics", *BMC Genomics*, Vol. 13, No. 1, S15, 2012.
[12] R. S. Ross, "Guide for Conducting Risk Assessments", *NIST SP800-30- Rev1 Technical Report*, 2012.
[13] M. Corpuz, P. H. Barnes, "Integrating Information Security Policy Management with Corporate Risk Management for Strategic Alignment", *Proc. of World Multi-Conference on Systemics, Cybernetics and Informatics*, 2010.
[14] C. W. Probst, R. R. Hansen, and F. Nielson, "Where Can an Insider Attack?", *Proc. of FAST*, 2006.
[15] R. De Nicola, G.L. Ferrari, R. Pugliese, "Klaim: A Kernel Language for Agents Interaction and Mobility", *IEEE Trans. on Software Engineering*, Vol. 24, No. 5, pp. 315-330, 1998.
[16] B. Solhaug , K. Stølen, "Preservation of Policy Adherence Under Refinement", *Int. J. Software and Informatics*, Vol. 5, pp. 139-157, 2011.
[17] T. Pereira, H. Santos, "An Ontology based Approach to Information Security", *Proc. of Research Conference on Metadata and Semantic Research*, Springer, Berlin, Heidelberg, pp. 183-192, 2009.
[18] M. Clarkson, "Quantification and Formalization of Security", *Cornell University Dissertation*, 2010.
[19] D. . Bell, "Looking Back at the Bell-La Padula Model", *Proc. of ACSAC*, 2005.
[20] K. J. Biba, "Integrity Considerations for Secure Computer Systems", *MITRE Corp. Technical report 04*, 1977.
[21] D. D. Clark, D. R. Wilson, "A Comparison of Commercial and Military Computer Security Policies", *Proc. of IEEE Security*, 1987.
[22] University of Missouri Security Posture - public document, http://infosec.missouri.edu/classification/definitions.html.
[23] University of Texas at Austin - TACC Security Posture - public document, http://security.utexas.edu/policies/data_classification.
[24] University of Southern California - ISIS Security Posture - public document, http://itservices.usc.edu/securityservices.
[25] J. Chase, V. Thummala, "A guided tour of SAFE GENI", *Duke University Department of Computer Science Tech Report, CS-2014-002*, 2014.
[26] T. Yu, Y. Zhang, K. Lin, "Efficient Algorithms for Web Services Selection with End-to-End QoS Constraints", *ACM Transactions on the Web*, 2017.
[27] R. Hand, M. Ton, E. Keller, "Active Security", *Proc. of ACM HotNets*, 2013.
[28] T. Wood, K.K. Ramakrishnan, Jinho Hwang, G. Liu, Wei Zhang, "Toward a Software-Based Network: Integrating Software Defined Networking and Network Function Virtualization", *IEEE Network*, 2015.
[29] A. Zaalouk, R. Khondoker, R. Marx, and K. Bayarou, "OrchSec: An Orchestrator-Based Architecture For Enhancing Network Monitoring and SDN Control Functions", *Proc. of IEEE NOMS*, 2014.
[30] Internet2 InCommon - https://www.incommon.org.
[31] OpenID: Standard and Decentralized Authentication Protocol - http://openid.net.
[32] X.509 Specification - http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0.pdf.
[33] R. Ananthakrishnan, J. Bryan, K. Chard, I. Foster, T. Howe, M. Lidman, S. Tuecke, "Globus Nexus: An Identity, Profile, and Group Management Platform for Science Gateways and other Collaborative Science Applications", *Proc. of IEEE CLUSTER*, 2013.
[34] B. Baker, K. Borne, T. Handley, J. Kantor, J. Hughes, R. Lambert, C. Lee, H. Larrieu, R. Plante, "LSST Data Management Cybersecurity Draft Plan", 2015.
[35] R. Pordes, D. Petravick, B. Kramer, D. Olson, M. Livny, A. Roy, P. Avery, K. Blackburn, T. Wenaus, F. Wurthwein, I. Foster, R. Gardner, M. Wilde, A. Blatecky, J. McGee, R. Quick, "The Open Science Grid", *Journal of Physics: Conference Series*, 2007.
[36] E. Deelman, G. Singh, M. Su, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, K. Vahi, G. Berriman, J. Good, A. Laity, J. Jacob, D. Katz, "Pegasus: A Framework for Mapping Complex Scientific Workflows onto Distributed Systems", *Scientific Programming*, Vol. 13, No. 3, pp. 219-237, 2005.
[37] Extreme Science and Engineering Discovery Environment - https://www.xsede.org.
[38] Information Science Institute - http://www.isi.edu.
[39] Texas Advanced Computing Center - https://www.tacc.utexas.edu.
[40] Integrated Rule-Oriented Data System (iRODS) - http://irods.org.
[41] Y. Zhang, S. Debroy, P. Calyam, "Network-wide Anomaly Event Detection and Diagnosis in perfSONAR", *IEEE Trans. on Network and Service Management*, 2016.
[42] T. Ostwald S. Hernan, S. Lambert, A. Shostack, "Uncover Security Design Flaws Using The Stride approach", *MSDN Magazine*, 2006.
[43] M. Berman, J. Chase, L. Landweber, A. Nakao, M. Ott, D. Raychaudhuri, R. Ricci, I. Seskar, "GENI: A Federated Testbed for Innovative Network Experiments", *Elsevier Computer Networks*, Vol. 61, No. 14, pp. 5-23, 2014.
[44] "Minimum Security Requirements for Federal Information and Information Systems", *NIST Technical Report*, 2006.
[45] M. Masse, "REST API Design Rulebook", *O'Reilly Media ISBN: 978-1-4493-1050-9*, 2011.
[46] R. Morgan, S. Cantor, S. Carmody, W. Hoehn, K. Klingenstein, "Federated Security: The Shibboleth Approach", *EDUCAUSE Quarterly*, 2004.
[47] Amazon Web Services - http://aws.amazon.com.
[48] IMOD: image processing, modeling and display programs - http://bio3d.colorado.edu/imod.

Matthew Dickinson received his M.S. degree in Computer Science from University of Missouri-Columbia in 2009. He is currently pursuing his Ph.D. degree in Computer Science at University of Missouri-Columbia. His current research interests include Cloud Computing, Big Data Networking, and Cloud Security.

Saptarshi Debroy received his PhD degree in Computer Engineering from University of Central Florida in 2014, MTech degree from Jadavpur University, India in 2008, and BTech degree from West Bengal University of Technology, India in 2006. He is currently a Post Doctoral Fellow at University of Missouri-Columbia. His current research interests include Cloud Computing, Big Data Networking, and Cognitive Radio Networks.

Prasad Calyam received his MS and PhD degrees from the Department of Electrical and Computer Engineering at The Ohio State University in 2002 and 2007, respectively. He is currently an Assistant Professor in the Department of Computer Science at University of Missouri-Columbia. His current research interests include distributed and cloud computing, computer networking, and cyber security.

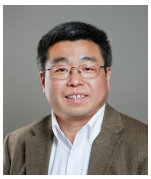
Samaikya Valluripally is currently a Computer Science MS student in University of Missouri Columbia. She received her Bachelor of Technology degree in Computer Science from Jawahrlal Nehru Technological University, India in the year 2014. Her current research interests include Cloud Computing, Big Data Analytics, Data Mining, Entrepreneurship.

Yuanxun Zhang received his BE degree from Southwest Jiaotong University, China, in 2006. He is currently pursuing his PhD degree in University of Missouri-Columbia. His research interests include network performance monitoring, software-defined networking, and big data analytics.

Ronny Bazan Antequera received his MS degree in Computer Science from University of Missouri-Columbia in 2014. He received his BS degree in Computer Science from San Andres University, Bolivia in 2005. He is currently pursuing his Ph.D. degree in Computer Science at University of Missouri-Columbia. His current research interests include hybrid cloud computing and software-defined networking.

Trupti Joshi received her PhD from the University of Missouri-Columbia in 2013. She received her MS from University of Tennessee-Knoxville in 2003. She is currently a Director, Translational Bioinformatics at University of Missouri-Columbia. Her current research interests include bioinformatics, computational systems biology and genomics.

Tommi White received her Ph.D. and BS degrees in Biochemistry from the University of Missouri in 2007 and 2000 respectively. She is currently the Associate Director of the University of Missouri Electron Microscopy Core Facility and is also an Assistant Research Professor in Biochemistry. Her research interests include biological specimen preparation methods to preserve ultrastructure, and to visualize purified proteins in a frozen hydrated state.



Dong Xu received his PhD from University of Illinois at Urbana-Champaign in 1995. He received his MS and BS from Peking University. He is currently the James C. Dowell Professor and Chair of Computer Science department at University of Missouri-Columbia. His current research interests include bioinformatics, protein structure prediction and computational systems biology.