Journal of Cloud Computing
a SpringerOpen Journal

RESEARCH Open Access

A multi-level security model for partitioning workflows over federated clouds

Paul Watson

Abstract

Cloud computing has the potential to provide low-cost, scalable computing, but cloud security is a major area of concern. Many organizations are therefore considering using a combination of a secure internal cloud, along with (what they perceive to be) less secure public clouds. However, this raises the issue of how to partition applications across a set of clouds, while meeting security requirements. Currently, this is usually done on an ad-hoc basis, which is potentially error-prone, or for simplicity the whole application is deployed on a single cloud, so removing the possible performance and availability benefits of exploiting multiple clouds within a single application. This paper describes an alternative to ad-hoc approaches – a method that determines all ways in which applications structured as workflows can be partitioned over the set of available clouds such that security requirements are met. The approach is based on a Multi-Level Security model that extends Bell-LaPadula to encompass cloud computing. This includes introducing workflow transformations that are needed where data is communicated between clouds. In specific cases these transformations can result in security breaches, but the paper describes how these can be detected. Once a set of valid options has been generated, a cost model is used to rank them. The method has been implemented in a tool, which is described in the paper.

Introduction

Cloud computing is of growing interest due to its potential for delivering cheap, scalable storage and processing. However, cloud security is a major area of concern that is restricting its use for certain applications: "Data Confidentiality and Auditability" is cited as one of the top ten obstacles to the adoption of cloud computing in the influential Berkeley report [1]. While security concerns are preventing some organizations from adopting cloud computing at all, others are considering using a combination of a secure internal "private" cloud, along with (what they perceive to be) less secure "public" clouds. Sensitive applications can then be deployed on a private cloud, while those without security concerns can be deployed externally on a public cloud. However, there are problems with this approach. Currently, the allocation of applications to clouds is usually done on an ad-hoc, per-application basis, which is not ideal as it lacks rigour and auditability. Further, decisions are often made at the level of granularity of the whole application, which is allocated entirely to either a public or private cloud based on a judgment of its overall sensitivity. This eliminates the potential benefits for partitioning an application across a set of clouds, while still meeting its overall security requirements. For example, consider a medical research application in which data from a set of patients' heart rate monitors is analyzed. A workflow used to analyze the data from each patient is shown in Figure 1. The input data is a file with a header identifying the patient, followed by a set of heart rate measurements recorded over a period of time. A service (*Anonymize*) strips off the header, leaving only the measurements (this application is concerned with the overall results from a cohort of patients, not with individuals). A second service (*Analyze*) then analyzes the measurements, producing a summary.

Analyzing the heart rate data is computationally expensive, and would benefit from the cheap, scalable resources that are available on public clouds. However, most organizations would be unlikely to consider storing medical records on a public cloud for confidentiality and, in some cases, legal reasons. Therefore, one solution is to deploy the whole workflow on a secure private cloud. However, this may overload the finite resources of the private cloud,

Correspondence: Paul.Watson@ncl.ac.uk
School of Computing Science, Newcastle University, Newcastle-upon-Tyne, NE1 7RU, UK

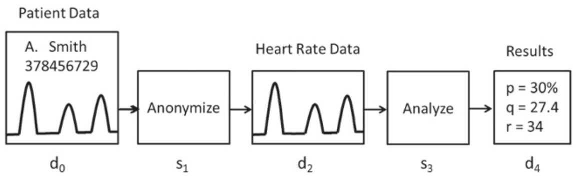

Figure 1 An example medical data analysis workflow.

resulting in poor performance, and potentially a negative impact on other applications.

An alternative solution is to partition the application between the private cloud and an external public cloud in order to exploit the strengths of both. This could be attempted in an ad-hoc fashion by a security expert but, as this paper describes, there are challenges in working out the set of partitioning options that still preserve the required security of data and services. This paper therefore describes an alternative to ad-hoc solutions – a method that takes an application consisting of a set of services and data connected in a workflow, and determines the valid set of deployments over a set of clouds, ensuring that security requirements are met. Although the paper is focused on workflows in which services communicate through passing data, the method can be applied to other types of distributed system that are composed of a set of communicating components. The method is based on Multi-Level Security models [2], specifically Bell-LaPadula [3]. The result of the method is the complete set of options that meet the organization's security requirements for the application. The method introduces transformations that need to be performed on the workflows where data is communicated between clouds; the paper identifies the security issues that can be raised as a result, and the extra security checks that need to be performed to address this. When the method results in more than one valid partitioning option, there is the issue of how to choose the best. The paper shows how a cost model can be introduced to rank the valid options; a model based on price is defined, and applied to the running medical workflow example. The full method, including the cost model, has been implemented in a tool that has been built to automate and explore its application.

The paper is structured as follows. The **Method** section gives a brief introduction to Multi-Level Security models and Bell-LaPadula. It then describes how the Bell-LaPadula rules can be applied to ensure that a workflow meets the security requirements of its constituent services and data. The method is then extended to cloud computing by assigning security levels to clouds, and building on Bell-LaPadula to define a method for determining if security requirements are met in a particular deployment of the constituent parts of a workflow onto a set of clouds. The **Calculating valid deployment options** section then defines a method for enumerating all valid options for deploying a workflow over a set of clouds so as to meet security requirements. It highlights the issues raised when data must flow between clouds, and shows the workflow transformations and security checks that must be included in the method if security is to be guaranteed. The result is a set of valid options; the **Selecting a deployment option with a cost model** section then introduces a model that can be used to select the best option. The method is then applied to a second, more complex example (in the **A more complex example** section). A tool has been designed and built to implement the method. As described in the **Tooling** Section, it is structured as a set of rules, transforms and a cost model, allowing it to be enhanced to meet other non-functional requirements, including dependability. Following a review of related work, the paper draws conclusions and outlines further work.

Method

This section describes how the Bell-LaPadula security model can be applied to workflows, and can then be extended to the deployment of workflows on clouds. Through this section, a workflow is modeled as a directed graph in which services and data are represented as nodes. Services consume zero or more data items and generate one or more data items; the edges in the graph represent the data dependencies.

Representing security requirements

The Bell-LaPadula multi-level access control model [3] is adopted, with services modeled as the subjects (S), and data as the objects (O) [4]. The security model therefore consists of the following:

- a set of actions (A) that subjects (S) can carry out on objects (O). In the case of services operating on data in a workflow, the actions are limited to read and write. Therefore, the set of actions (A) is: $A = \{r, w\}$
- a poset of security levels: L
- a permissions matrix: $M : S \times O \rightarrow A$ (the contents of the matrix are determined by the workflow design; i.e. if service s_1 reads datum d_0 then there will be an entry in the matrix: $s_1 \times d_0 \rightarrow r$; similarly, if service

s_1 writes datum d_2 then there will be an entry in the matrix: $s_1 \times d_2 \rightarrow w$)

- an access matrix: $B : S \times O \rightarrow A$ (this is determined by the execution of the workflow: if there are no choice points then it will equal the permissions matrix, however, if there are choice points then it will equal a subset of the permissions matrix corresponding to the path taken through the workflow when it is executed.
- a clearance map: $C : S \rightarrow L$ (this represents the maximum security level at which each service can operate)
- a location map: $l : S + O \rightarrow L$ (this represents the security level of each service and datum in the workflow)

In a typical Multi-Level Security scenario, the system moves through a set of states, and the model can have different values for permissions, access, clearance and location in each state. However, here the execution of a workflow is modeled as taking place within a single state. Normally a service would be expected to have a clearance that is constant across all uses of that service in workflows, however the location can be chosen specifically for each workflow, or even (though less likely) for each invocation of a workflow. However, the model itself is general, and makes no assumptions on this.

The Bell-LaPadula model states that a system is secure with respect to the above model if the following conditions are satisfied $\forall \text{subjects} u \in S$ and $\forall \text{objects} i \in O$

$$\text{authorization:} B_{ui} \subseteq M_{ui} \tag{1}$$

$$\text{clearance:} l(u) \leq c(u) \tag{2}$$

$$\text{no-read-up:} r \in B_{ui} \Rightarrow c(u) \geq l(i) \tag{3}$$

$$\text{no-write-down:} w \in B_{ui} \Rightarrow l(u) \leq l(i) \tag{4}$$

For workflows, the implications of these conditions are:

(1) all actions carried out by services must conform to the permissions granted to those services
(2) a service can only operate at a security level (location) that is less than or equal to its clearance
(3) a service cannot read data that is at a higher security level than its own clearance
(4) a service cannot write data to a lower security level than its own location.

For example, consider a service s_1 which consumes datum d_0 and produces datum d_2:

$$\boxed{d_0} \rightarrow \boxed{s_1} \rightarrow \boxed{d_2}$$

(in these diagrams, the $\rightarrow$ is used to show data dependency, and each block – service or datum – is uniquely identified by the subscript). The following rules must be met:

by (3)

$$c(s_1) \geq l(d_0) \tag{5}$$

and by (4)

$$l(d_2) \geq l(s_1) \tag{6}$$

The relationship between security levels is captured in Figure 2. Arrows represent $\geq$ relationships.

Whilst assigning a security level to a datum in a workflow is directly analogous to assigning a level to an object (e.g. a document) in the standard Bell-LaPadula model, assigning a security level to a service may be less intuitive. The justification is that an organization may have differing levels of confidence in the set of services they wish to use. For example, they may be very confident that a service written in-house, or provided by a trusted supplier, will not reveal the data it consumes and produces to a third party either deliberately or through poor design; in contrast, there is a risk that a service downloaded from the Internet, of unknown provenance, may do just that. Therefore, the organization can assign a high security level to the former service, and a low level to the latter.

For a specific workflow, when an organization's security experts are assigning locations to services, they may in some cases chose to set the location below that of the clearance level in order to allow a service to create data that is at a lower level than its clearance level; i.e. so that the no write down rule (4) is not violated. This may, for example, take place when the expert knows that the output data will not be sensitive, given the specific data that the service will consume as input in this specific workflow. A concrete example would be a service that summarizes textual data. This has been written to a high standard, and the security expert is confident that it will not leak data to a third party. Therefore, its clearance is high. However, in one particular workflow it is known that this service will only be used to summarise public data downloaded from the World Wide Web, which is also where its output will be published. Therefore, the security expert would set the

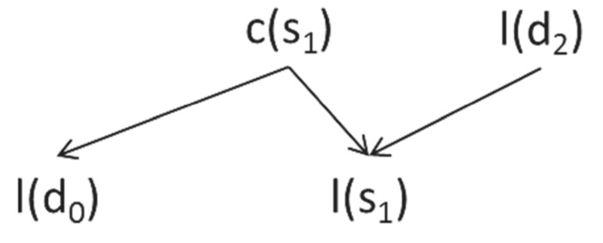

Figure 2 The relationship between security levels for a service that consumes and produces data.

service's location to an appropriately low level so that the write down rule was not violated.

Cloud security

This section describes how the Bell-LaPadula model, as applied to workflows, can be extended to encompass cloud computing.

Let us say that an organization wishes to run a particular workflow. As more than one cloud is available, a decision must be made as to where the data and services should be placed. In current practice, it is typical that a security expert or system administrator would just take a considered view on the overall security level of the workflow, and that of the clouds on which it could be deployed. For example, let us say that there are two clouds, one a highly trusted private cloud contained within the intranet of the organization, and the other a less trusted public cloud. It may seem obvious in this case that a workflow that operates on sensitive medical data should run only on the internal cloud. Similarly, a workflow that summarises public data could be deployed on the public cloud. However, there are two problems with this approach. Firstly, it is informal, being based on an expert's judgment; a systematic approach is preferable as it will give more consistent, defendable results. Secondly, the approach deploys the whole of a workflow on a single cloud. This rules out other options that may:

- reduce cost: for example by running less sensitive, but computationally intensive, sub-parts of the workflow on a public cloud if that avoids the need to purchase expensive new servers so that the internal cloud can handle the extra load
- increase reliability: for example by having the option to run on a public cloud if the private cloud has an outage
- increase performance: for example by taking advantage of the greater processing capacity of the public cloud for the computationally intensive services in a workflow

Therefore, the rest of this section extends the security model introduced earlier in order to allow systematic decisions to be taken on where the services and data within a workflow may be deployed to ensure security requirements are met.

To do this, the location map is extended to include clouds which we denote by P (to avoid confusion with the C conventionally used to denote the clearance map):

- location map: $l : S + O + P \rightarrow L$

Also, H is added to represent the mapping from each service and datum to a cloud:

$$H : S + O \rightarrow P]$$

We then add a rule that any block (service or datum) must be deployed on a cloud that is at a location that is greater than or equal to that of the block, e.g. for a block x on cloud y:

$$l(p_y) \geq l(b_x) \tag{7}$$

Returning to the example service introduced in the previous section:

$$\boxed{d_0} \rightarrow \boxed{s_1} \rightarrow \boxed{d_2}$$

if, in H, d_0 is on cloud p_a, s_1 on p_b and d_2 on cloud p_c then the following must be true:

$$l(p_a) \geq l(d_0) \tag{8}$$

$$l(p_b) \geq l(s_1) \tag{9}$$

$$l(p_c) \geq l(d_2) \tag{10}$$

This allows us to extend (6) to:

$$l(p_c) \geq l(d_2) \geq l(s_1) \tag{11}$$

The complete relationship between security levels for blocks and clouds is captured in Figure 3.

Calculating valid deployment options

Using the above model and rules, it is now possible to automatically enumerate all the valid deployment options for a workflow. These are generated in two stages. Firstly, given the following:

- the set of clouds P
- the set of services S
- the set of data O
- the map of security locations l

we can define the valid mappings of services and data onto clouds, using rule (7):

$$V : S + O \rightarrow P$$

$$V = \{b \rightarrow p | b \in S + O, p \in P, l(b) \leq l(p)\}$$

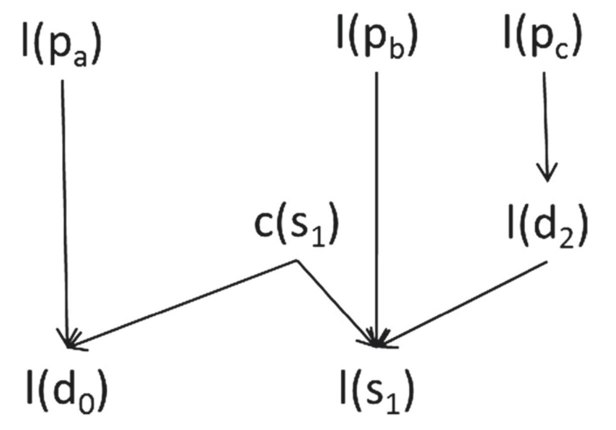

Figure 3 The relationship between cloud and workflow block security levels.

To illustrate this, we use the medical workflow of Figure 1, with two clouds. This has two services connected in a pipeline, each with one datum as input and one as output:

$$\boxed{d_0} \rightarrow \boxed{s_1} \rightarrow \boxed{d_2} \rightarrow \boxed{s_3} \rightarrow \boxed{d_4}$$

Table 1 shows an example location and clearance table (while the scheme is general, this example uses only two security levels: 0 and 1). Here, c_1 is a private cloud, which is at a higher security level than the public cloud c_0. The patient data (d_0) is at the highest security level, while the other data is at the lower level as it is not confidential. Service s_1 is cleared to access confidential data at level 1, but its location has been set to 0 in this workflow so that it can produce non-confidential output at level 0 without violating the Bell-LaPadula "no-write-down" rule (4).

Based on this mapping of blocks and clouds to locations, Table 2 then shows the possible valid placement of each block onto the two clouds.

Having determined all valid mappings of services and data to clouds, the set of all valid workflow deployments is given by:

$$W : (S + O \rightarrow P) \rightarrow \{(S + O \rightarrow P)\} =$$

$$\{w \in ||V||, \forall b \in S{+}O.\exists p \in P.b \rightarrow p \in w, |w| = |S{+}O|\}$$

Where $||V||$ is the power set of V and $|w|$ is the cardinality of w. Algorithmically, in the implementation of the method, W is computed by forming the cross-product of the block-to-cloud mappings contained in V.

All possible valid workflow deployments – as defined by W – for the running medical workflow example are shown in Figure 4. The cloud on which a datum or service is deployed is indicated as a superscript; e.g. d_j^a is datum j deployed on cloud a.

Transferring data between clouds

There is still an important issue to be addressed: the approach makes assumptions that are unrealistic for a practical distributed workflow system. It assumes that:

Table 1 Locations and clearances for the medical analysis example

	Location (l)	Clearance (c)
d_0	1	
s_1	0	1
d_2	0	
s_3	0	0
d_4	0	
c_0	0	
c_1	1	

Table 2 Valid mappings of blocks to clouds

Block	Cloud c_0	Cloud c_1
d_0		•
s_1	•	•
d_2	•	•
s_3	•	•
d_4	•	•

1. a service can generate as its output a datum directly on another cloud, without that item being first stored on the same cloud as the service
2. a service can consume as its input a datum directly from another cloud, without that item ever being stored on the same cloud as the service

This problem is solved in two stages. Firstly, a new type of service is introduced – sxfer – which will transfer data from one cloud to another (this is analogous to the exchange operator used in distributed query processing [5]). It would be implemented with sub-components running on the source and destination clouds. The sxfer service takes a datum on one cloud and creates a copy on another. All the workflows generated by W are then transformed to insert the transfer nodes whenever there is a inter-cloud edge in the workflow graph. There are four graph transformations:

$$\boxed{d_j^a} \rightarrow \boxed{s_i^a} \Rightarrow \boxed{d_j^a} \rightarrow \boxed{s_i^a} \tag{12}$$

$$\boxed{d_j^a} \rightarrow \boxed{s_i^b} \Rightarrow \boxed{d_j^a} \rightarrow \boxed{sxfer} \rightarrow \boxed{d_j^b} \rightarrow \boxed{s_i^b} \tag{13}$$

$$\boxed{s_i^a} \rightarrow \boxed{d_j^a} \Rightarrow \boxed{s_i^a} \rightarrow \boxed{d_j^a} \tag{14}$$

$$\boxed{s_i^a} \rightarrow \boxed{d_j^b} \Rightarrow \boxed{s_i^a} \rightarrow \boxed{d_j^a} \rightarrow \boxed{sxfer} \rightarrow \boxed{d_j^b} \tag{15}$$

Transforms (12) and (14) reflect the fact that if both nodes are deployed on the same cloud then no change is needed. In contrast, (13) and (15) introduce new $\boxed{\text{sxfer}}$ nodes to transfer data between clouds.

Unfortunately, the creation of new copies of data through transforms (13) and (15) introduces potential security problems. When transform (13) is applied, there is the need to check that cloud b has a sufficiently high security level to store the copy of d_j that would be created on it (the copy inherits the security level of the original). The following rule must therefore be checked to ensure this is true:

$$l(p_b) \geq l(d_j) \tag{16}$$

Similarly, for transform 15:

$$l(p_a) \geq l(d_j) \tag{17}$$

$$\boxed{d_0^1} \rightarrow \boxed{s_1^0} \rightarrow \boxed{d_2^0} \rightarrow \boxed{s_3^0} \rightarrow \boxed{d_4^0}$$

$$\boxed{d_0^1} \rightarrow \boxed{s_1^0} \rightarrow \boxed{d_2^0} \rightarrow \boxed{s_3^0} \rightarrow \boxed{d_4^1}$$

$$\boxed{d_0^1} \rightarrow \boxed{s_1^0} \rightarrow \boxed{d_2^1} \rightarrow \boxed{s_3^0} \rightarrow \boxed{d_4^0}$$

$$\boxed{d_0^1} \rightarrow \boxed{s_1^0} \rightarrow \boxed{d_2^1} \rightarrow \boxed{s_3^0} \rightarrow \boxed{d_4^1}$$

$$\boxed{d_0^1} \rightarrow \boxed{s_1^0} \rightarrow \boxed{d_2^0} \rightarrow \boxed{s_3^1} \rightarrow \boxed{d_4^0}$$

$$\boxed{d_0^1} \rightarrow \boxed{s_1^0} \rightarrow \boxed{d_2^0} \rightarrow \boxed{s_3^1} \rightarrow \boxed{d_4^1}$$

$$\boxed{d_0^1} \rightarrow \boxed{s_1^0} \rightarrow \boxed{d_2^1} \rightarrow \boxed{s_3^1} \rightarrow \boxed{d_4^0}$$

$$\boxed{d_0^1} \rightarrow \boxed{s_1^0} \rightarrow \boxed{d_2^1} \rightarrow \boxed{s_3^1} \rightarrow \boxed{d_4^1}$$

$$\boxed{d_0^1} \rightarrow \boxed{s_1^1} \rightarrow \boxed{d_2^0} \rightarrow \boxed{s_3^0} \rightarrow \boxed{d_4^0}$$

$$\boxed{d_0^1} \rightarrow \boxed{s_1^1} \rightarrow \boxed{d_2^0} \rightarrow \boxed{s_3^0} \rightarrow \boxed{d_4^1}$$

$$\boxed{d_0^1} \rightarrow \boxed{s_1^1} \rightarrow \boxed{d_2^1} \rightarrow \boxed{s_3^0} \rightarrow \boxed{d_4^0}$$

$$\boxed{d_0^1} \rightarrow \boxed{s_1^1} \rightarrow \boxed{d_2^1} \rightarrow \boxed{s_3^0} \rightarrow \boxed{d_4^1}$$

$$\boxed{d_0^1} \rightarrow \boxed{s_1^1} \rightarrow \boxed{d_2^0} \rightarrow \boxed{s_3^1} \rightarrow \boxed{d_4^0}$$

$$\boxed{d_0^1} \rightarrow \boxed{s_1^1} \rightarrow \boxed{d_2^0} \rightarrow \boxed{s_3^1} \rightarrow \boxed{d_4^1}$$

$$\boxed{d_0^1} \rightarrow \boxed{s_1^1} \rightarrow \boxed{d_2^1} \rightarrow \boxed{s_3^1} \rightarrow \boxed{d_4^0}$$

$$\boxed{d_0^1} \rightarrow \boxed{s_1^1} \rightarrow \boxed{d_2^1} \rightarrow \boxed{s_3^1} \rightarrow \boxed{d_4^1}$$

Figure 4 All valid workflows produced by mapping blocks to clouds.

If either is violated then the workflow does not meet the security requirements, and so should be removed from the set W of valid mappings of services and data to clouds. Proof that this violation can only occur in two specific cases now follows.

Firstly, consider (16). By rule (2) we have:

$$c(s_i^b) \geq l(s_i^b) \tag{18}$$

First consider the case where:

$$c(s_i^b) = l(s_i^b) \tag{19}$$

i.e. the clearance of the object is equal to its location.

Rules (3) and (4) give

$$c(s_i^b) \geq l(d_j) \tag{20}$$

and

$$l(p_b) \geq l(s_i^b) \tag{21}$$

then, by (19)

$$l(p_b) \geq l(s_i^b) \geq l(d_j) \Rightarrow l(p_b) \geq l(d_j) \tag{22}$$

and rule (16) is satisfied. Therefore, in this case there are no violations.

However, if:

$$c(s_i^b) > l(s_i^b) \tag{23}$$

i.e. the clearance of the service is strictly greater than its location then combining (23) with (3) and (4) in a similar way to the above, we get:

$$l(s_i^b) < c(s_i^b) \geq l(d_j) \tag{24}$$

and

$$l(p^b) \leq l(s_i^b) < c(s_i^b) \tag{25}$$

so it is possible that

$$l(p^b) < l(d_j) \tag{26}$$

in which case rule (16) is violated and so that particular workflow deployment does not meet the security requirements.

Turning now to the data produced by services, rule (17) can be violated by transform (15) in the case where the service s_0 writes up data (4) to a level such that:

$$l(p_a) < l(d_j)$$

The effect of the transformations is to modify the security lattice of Figure 2 to that of Figure 5. The arc from $l(p_b)$ to $l(d_0)$ is introduced by transform (13) which adds a copy of d_0 into the workflow, while the arc from $l(p_b)$ to $l(d_2)$ is introduced by transform (15) which adds a copy of d_2.

Applying the transformations to each workflow in Figure 4, followed by rules (16) and (17) removes half of

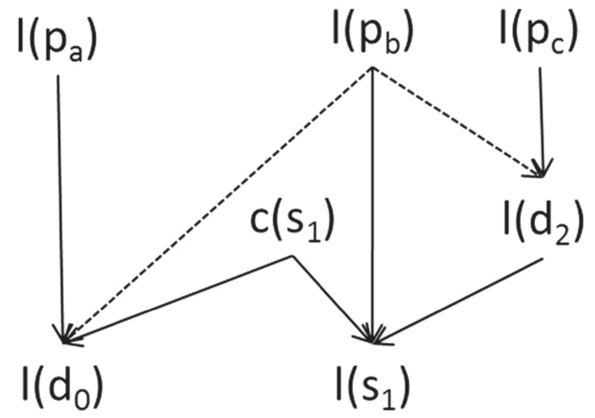

Figure 5 **The relationship between security levels after transformation for inter-cloud data transfer.**

the possible deployment options. Removing two duplicates created by the transformations leaves the six valid options shown in Figure 6. Another view of the remaining options is shown in Figure 7. As services can have multiple inputs and outputs, the arcs in the diagrams are labelled with the input / output number. These diagrams were generated automatically by the tool we have built to implement the methods described in this paper. The aim is to provide a security expert with an easy to understand view of the possible options.

Whilst a simple, linear workflow has been used here to illustrate the method, it is applicable to all workflows that can be represented by a directed graph, whatever their structure. The discussion so far does however still leave open the issue of how to choose between these valid options? The next section therefore describes how a cost model (also implemented in the tool) can be used to select the best option based on the charges made by the cloud providers.

Selecting a deployment option with a cost model

Once all valid options for allocating services and data to clouds have been determined, one must be selected, and used to enact the workflow. This decision could be made by a deployment expert, but this section describes how it can be achieved automatically through the use of a cost model. Different criteria may be important for different applications (e.g. dependability, performance), but this section illustrates the approach by describing a model that minimizes price.

Cloud pricing is measured using the metrics by which cloud providers allocate charges. For a cloud (p) this is represented as:

- volume of data transferred into a cloud: e^p_{dxi}
- volume of data transferred out of a cloud: e^p_{dxo}
- volume of data stored, per unit of time for which it is stored: e^p_{ds}

$$\boxed{d_0^1} \rightarrow \boxed{s_1^1} \rightarrow \boxed{d_2^1} \rightarrow \boxed{sxfer} \rightarrow \boxed{d_2^0} \rightarrow \boxed{s_3^0} \rightarrow \boxed{d_4^0}$$

$$\boxed{d_0^1} \rightarrow \boxed{s_1^1} \rightarrow \boxed{d_2^1} \rightarrow \boxed{sxfer} \rightarrow \boxed{d_2^0} \rightarrow \boxed{s_3^0} \rightarrow \boxed{d_4^0} \rightarrow \boxed{sxfer} \rightarrow \boxed{d_4^1}$$

$$\boxed{d_0^1} \rightarrow \boxed{s_1^1} \rightarrow \boxed{d_2^1} \rightarrow \boxed{sxfer} \rightarrow \boxed{d_2^0} \rightarrow \boxed{sxfer} \rightarrow \boxed{d_2^1} \rightarrow \boxed{s_3^1} \rightarrow \boxed{d_4^1} \rightarrow \boxed{sxfer} \rightarrow \boxed{d_4^0}$$

$$\boxed{d_0^1} \rightarrow \boxed{s_1^1} \rightarrow \boxed{d_2^1} \rightarrow \boxed{sxfer} \rightarrow \boxed{d_2^0} \rightarrow \boxed{sxfer} \rightarrow \boxed{d_2^1} \rightarrow \boxed{s_3^1} \rightarrow \boxed{d_4^1}$$

$$\boxed{d_0^1} \rightarrow \boxed{s_1^1} \rightarrow \boxed{d_2^1} \rightarrow \boxed{s_3^1} \rightarrow \boxed{d_4^1} \rightarrow \boxed{sxfer} \rightarrow \boxed{d_4^0}$$

$$\boxed{d_0^1} \rightarrow \boxed{s_1^1} \rightarrow \boxed{d_2^1} \rightarrow \boxed{s_3^1} \rightarrow \boxed{d_4^1}$$

Figure 6 **The Workflows that remain valid after Transfer Blocks are Added.**

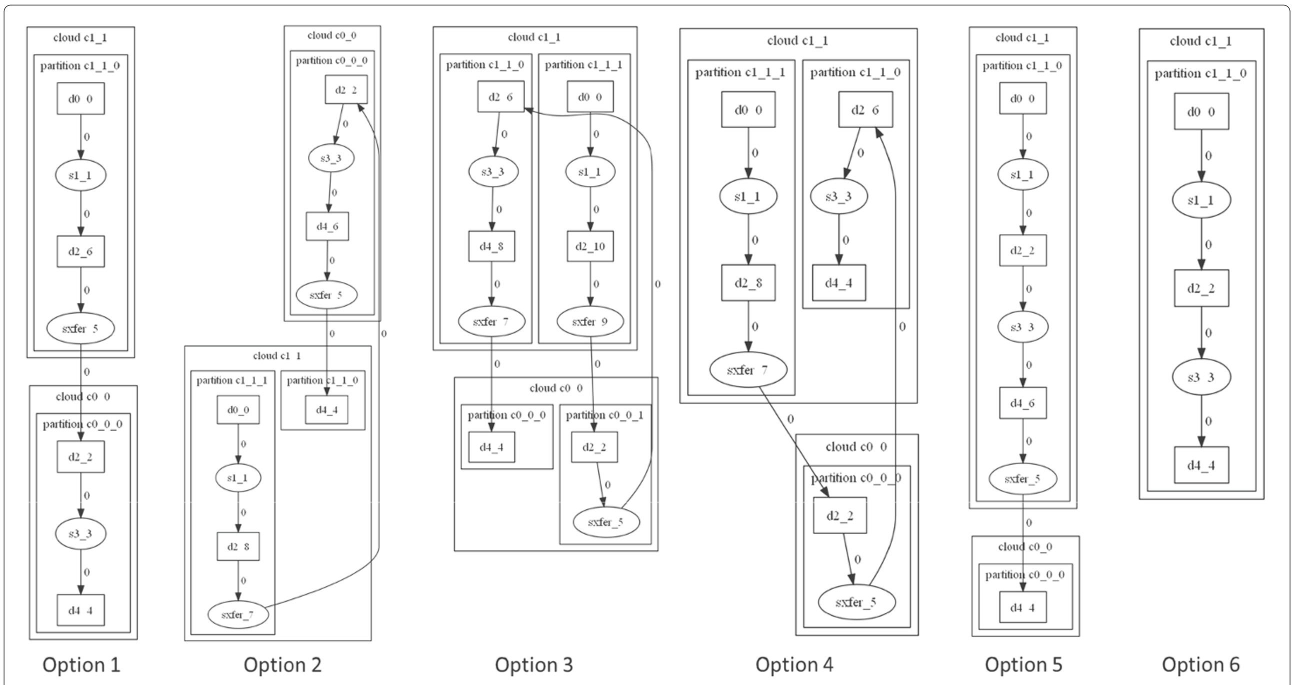

Figure 7 The six valid cloud mappings.

- time units of cpu consumed in the execution of a service: e^{p}_{cpu}

Cost metrics are characterised for a datum (d) as:

- data size: size(d)
- data longevity – the length of time the datum is stored: longevity(d)

Finally, the cost metric for a service (s) is characterised as:

- time units of cpu consumed in the execution of a service: cpu(s)

The cost model for a workflow execution can then be defined as:

$$\text{cost} = \sum_{d=0}^{d=k-1} e^{p}_{ds}.\text{size}(d).\text{longevity}(d) + \sum_{s=0}^{s=m-1} e^{p}_{cpu}.\text{cpu}(s) + \sum_{x=0}^{x=q-1} (e^{ps}_{dxo} + e^{pd}_{dxi}).\text{size}(d)$$

where k is the number of data items in the workflow, and m is the number of services, while q is the number of inter-cloud data transfers. In the third term that calculates data transfer costs, ps represents the source cloud and pd the destination cloud for the transfer.

Using the cost model requires estimates of data sizes and cpu costs. This is realistic for many workflows, and producing these estimates is made easier if performance and capacity are logged for each run, so allowing statistical analysis to generate predictions. This is, for example, done by the e-Science Central cloud platform [6] which logs data on all data sizes, and service execution times.

Two examples now highlight the use of the model. Consider the valid mapping options shown in Figure 7 for the running medical workflow example. In the simplest case, if the performance and cost of both clouds are equal (as in Table 3), then the cost difference between options is dependent only on the number of inter-cloud communications. Table 4 gives example values for the blocks in the workflow. The size of d_0 will be known as it is the input to the workflow, while that for d_2 and d_4 are estimates, perhaps based on the results of previous runs. To set the longevity values, it is assumed that the input (d_0)

Table 3 Cloud costs: Example 1

Cloud	Storage (GB / Month)	Transfer in (/GB)	Transfer out (/GB)	CPU (/s)
c_0	10	10	10	10
c_1	10	10	10	10

Table 4 Block info

Block	Size (GB)	Longevity (months)	CPU (s)
d_0	10	12	
s_1			100
d_2	5	0	
s_3			50
d_4	1	12	

and output data (d_4) is stored for a year, while intermediate data (d_2) (along with any intermediate copies of data created by transforms) is immediately discarded once it has been consumed by a service: in this case s_2.

Table 5 shows the results when the cost model is applied. Each row represents the cost of an option in Figure 6. The final column of the table gives the order of the options (from lowest to highest cost). This confirms that the cheapest is option 6, in which all the blocks are deployed on the same cloud, and so there are no inter-cloud transfer costs.

While it may seem that an option in which all services and data are deployed on a single cloud will always be the cheapest, if CPU costs vary between clouds, then inter-cloud transfers may be worthwhile. Table 6 shows clouds with a different set of cost parameters. Here, a private cloud (c_1) has higher security, but higher CPU and data costs, compared to a public cloud (c_0). The effect of plugging these values into the cost model is shown in Table 7. The result is that the best option is now the one that allocates as much work as possible to the public cloud, which has lower CPU costs.

A more complex example

The medical example used to date consists to a purely linear workflow: each service reads and writes only one data item. However, the method supports arbitrary workflows, and this is now demonstrated through the example of a more complex workflow,which is based on that introduced in [4]. It is shown in Figure 8.

Table 5 Workflow deployment options costs: Example 1

Option	Storage	Transfer	CPU	Total	Order
1	1320	100	1500	2920	3
2	1320	120	1500	2940	4
3	1320	220	1500	3040	6
4	1320	200	1500	3020	5
5	1320	20	1500	2840	2
6	1320	0	1500	2820	1

Table 6 Cloud costs: Example 2

Cloud	Storage (GB / Month)	Transfer in (/GB)	Transfer out (/GB)	CPU (/s)
c_0	5	5	5	5
c_1	10	5	5	10

With security settings shown in Table 8, the workflow meets the Bell-LaPadula criteria, and produces the three workflow partitionings shown in Figure 9.

With block costs from Table 9 and cloud costs from Table 6 (as in the previous example), the costs of the three partitioning options are shown in Table 10. This example illustrates the importance of the use of the cost model in allowing an expert or automatic deployment system to filter out valid but inefficient workflow partitionings such as option 2.

Tooling

The method described in this paper has been implemented in a tool which takes the workflow and security requirements as input, and generates as output the set of valid partitions with costs. The tool is implemented in the functional language Haskell [7], with workflows represented as directed graphs. This section explains how the tool implements the multi-level security method that is the focus of this paper, and then goes on to explain how it can be used to process other classes of non-functional requirements.

The tool is structured so that three types of functions are used to process the initial workflow:

1. **Transformation** functions take a workflow as input and generate a set of workflows derived from it. In the multi-level security method they are used for two purposes: to generate the candidate workflow partitionings from the initial workflow (Figure 4); and to insert the inter-cloud transfers according to (13) and (15).
2. **Rules** are implemented as filter functions that take a workflow as input and return a boolean indicating whether the workflow meets the rule. Workflows that do not meet a rule are removed from the set of

Table 7 Workflow deployment options costs: Example 2

Option	Storage	Transfer	CPU	Total	Order
1	1260	75	1250	2585	1
2	1320	90	1250	2660	2
3	1260	165	1500	2925	5
4	1320	150	1500	2970	6
5	1260	15	1500	2775	3
6	1320	0	1500	2820	4

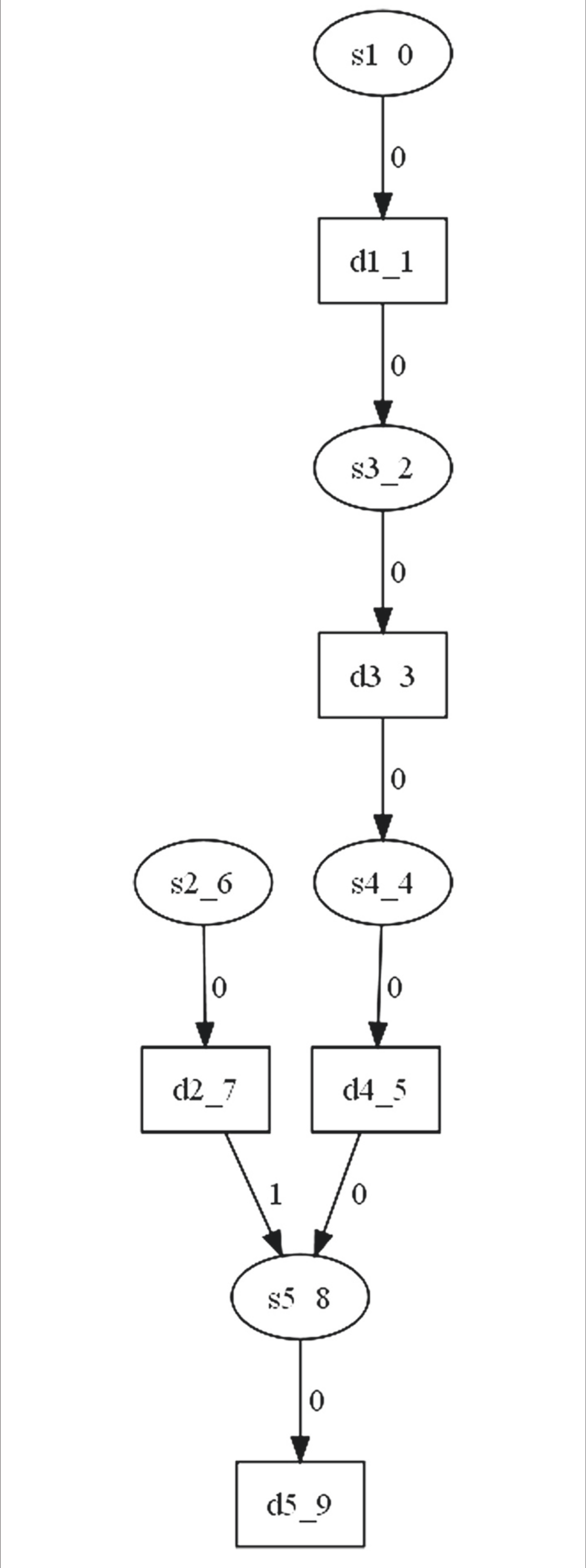

Figure 8 A more complex workflow.

Table 8 Block locations and clearances

	Location (*l*)	Clearance (*c*)
$s1_0$	0	1
$s2_6$	0	0
$s3_2$	1	1
$s4_4$	0	1
$s5_8$	1	1
$d1_1$	1	
$d2_7$	0	
$d3_3$	1	
$d4_5$	1	
$d5_9$	1	

valid workflows. In the methods described in this paper, they are used to check that the initial workflow conforms to Bell-LaPadula (Figure 2), and that the candidate workflow partitionings conform to Figure 3.

3. **Cost Model** functions take in a candidate workflow and assign a cost. They are used to implement the model described in Section “Selecting a deployment option with a cost model”.

Overall, the tool takes the initial workflow (e.g. that of Figure 1) and security requirements, applies the set of transformations and rules, and then uses the cost model function to rank the valid workflows.

The tool also automatically generates the diagrams that are shown in Figure 7 using the GraphViz software library [8]; these visualisations have proved to be a useful way to review the available options. It can also generate an html report for a security review, containing all the information generated by the method, including the diagrams, security tables and ranked cost tables. Finally, it can automatically generate LaTeXtables. Therefore all the tables and diagrams used in this paper have been automatically generated by the tool. Automatically generating diagrams and tables eliminates the risk of transcription errors [9] when conveying results through reports and papers. These reports can be used by system administrators to configure the partitions manually onto the clouds, but we are also currently developing a tool to do this automatically, as described in the next section.

Generalising the approach

While this paper focuses on meeting security requirements, the structure of the tool,with its three types of functions described above, allows it to be utilised as a general system for generating partitions of workflows over clouds that meet non-functional requirements. As well as security, those requirements can include dependability

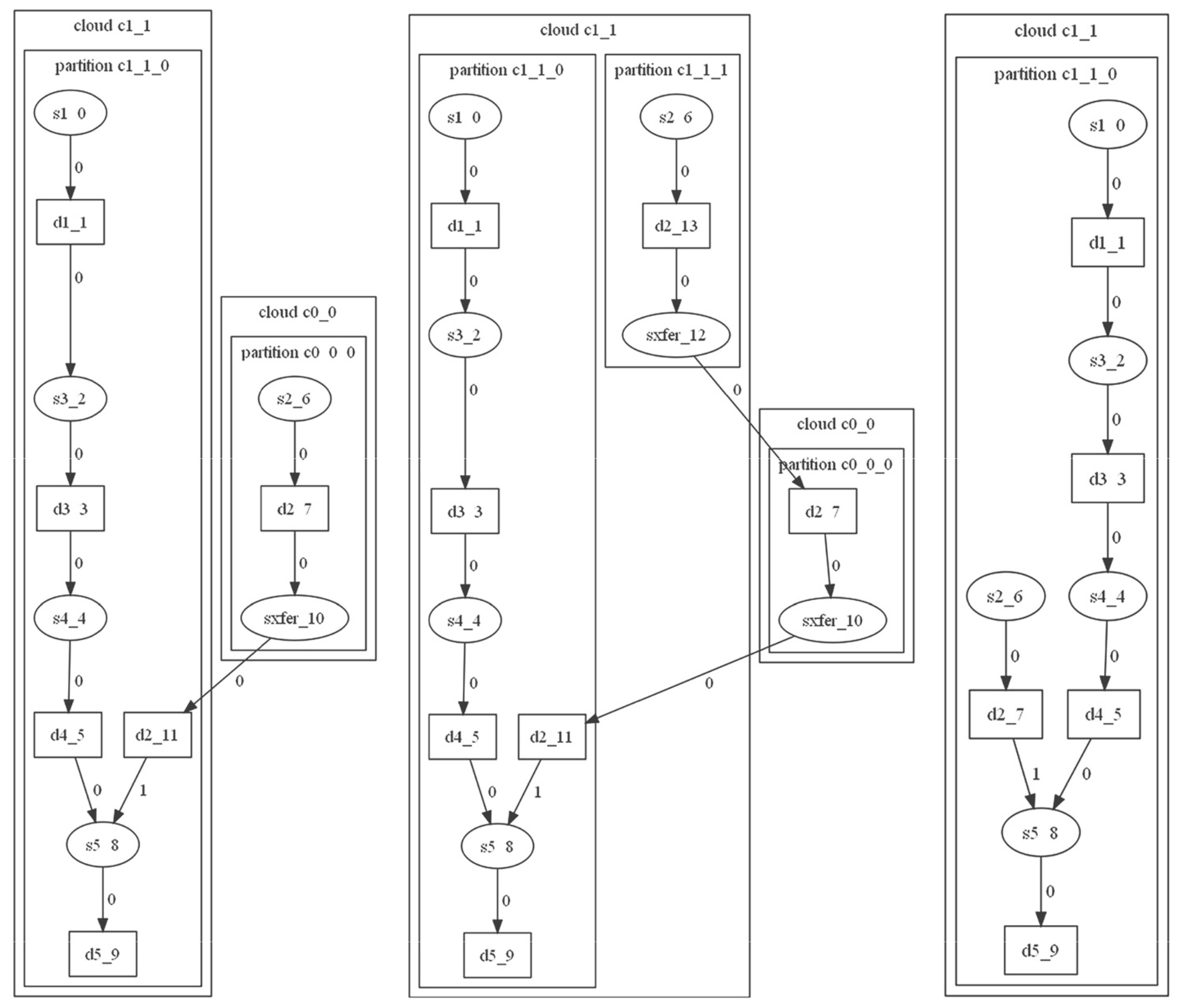

Figure 9 Valid workflow partitionings for the second example.

and performance. For example, the tool is currently being extended to encompass dependability requirements. One example requirement came out of discussions with the designer of an application in which the key workflow was very similar to that of Figure 1, but with the additional constraint that the input and output data for the workflow as a whole (e.g. d_0 and d_4 in Figure 1) must **not** be stored on the same cloud. We were able to meet this requirement simply by adding one extra rule into the tool. This is a function *apart* that takes three arguments:

- the set of blocks that need to be kept apart on different clouds
- the mapping of blocks onto clouds
- the workflow to be checked against the rule

The result of the function is true if every block contained in the first argument is deployed on a different cloud. One subtlety is that the function must take into account the fact that transformations may create one or more copies of the blocks that have been specified by the user as needing to be kept apart. For example, transforms (13) and (15) can create such copies. It is important that those copies are also included in the set of blocks that need to be kept apart. This is is achieved by exploiting the fact that the tool assigns blocks a name and identifier pair, e.g. '(*xfer*,14)'. While the identifier of each block is unique, the name can be shared by multiple blocks. When a block is copied by a transformation, it is given the same name but a new, unique identifier (similarly, separate deployments of the same service share the same name,

Table 9 Block Costs

	Size	Longevity	CPU
*d*1	10	12	
*d*2	5	12	
*d*3	1	0	
*d*4	20	0	
*d*5	20	12	
*s*1			100
*s*2			50
*s*3			10
*s*4			40
*s*5			60

but have a different unique identifier), e.g. a transform copying data item (d,14) could generate (d,27) where 27 is a unique identifier, not shared with another block. The *apart* function therefore works by comparing whether two blocks with the same name are stored on the same cloud, irrespective of their identifiers. For the running medical workflow example, adding the rule that d_0 and d_4 in Figure 1 must be deployed on different clouds results in only allowing the single partitioning option shown in Figure 10.

Figure 11 shows the architecture of the generalised tool. The Deployment Manager takes in the workflow and the user's non-functional requirements. It uses the rules and transformations to generate a set of valid partitionings over the available clouds, and then applies a cost model to rank them. The 'best' workflow is then executed across the set of clouds.

To achieve this, each cloud must run software that can store data and execute workflows. In this work the e-Science Central cloud platform [6] is used. This has the advantage of providing a portable platform that can be deployed on a range of clouds including public clouds (Amazon and Windows Azure) but also private clouds. Figure 12 shows the e-Science Central Architecture.

This is a cloud Platform-as-a-Service that provides users with the ability to store, share and analyse data in the cloud. Data can be uploaded and tagged with structured or unstructured metadata. Tools are provided to allow services, written in a variety of languages, to be packaged and loaded into the platform. Users can then create workflows from those services and execute them in the

Table 10 Workflow partitioning option costs

Option	Storage	Transfer	CPU	Total	Rank
1	4500	50	2350	6900	2
2	5100	100	2600	7800	3
3	4200	0	2600	6800	1

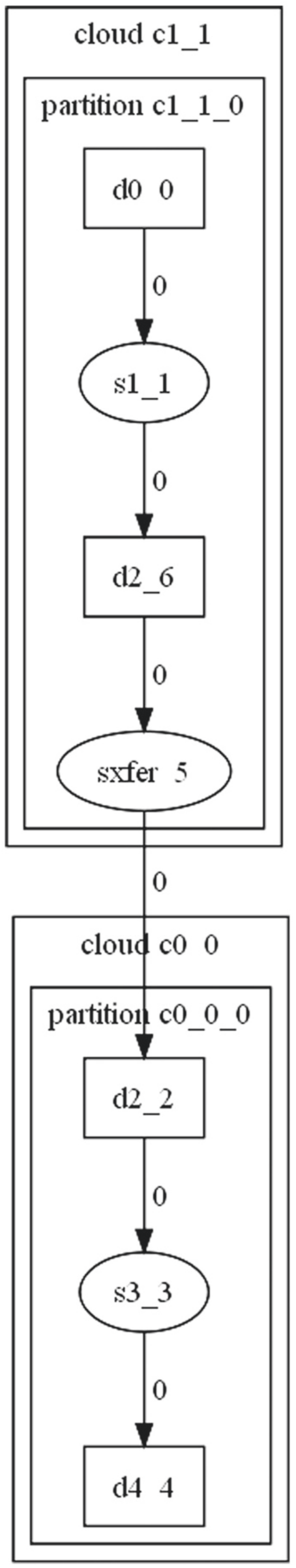

Figure 10 The only valid workflow partitioning if d_0 and d_4 must not be deployed on the same cloud.

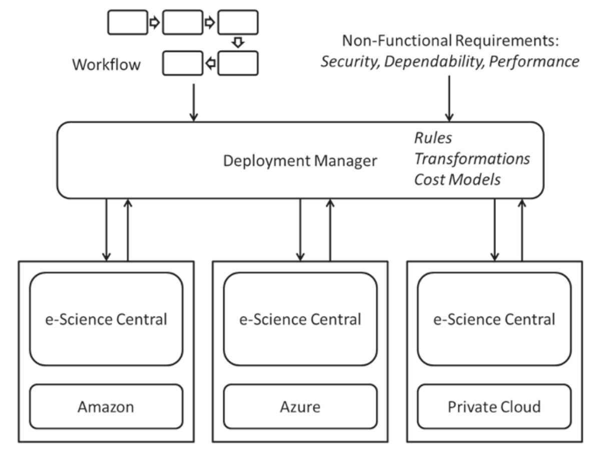

Figure 11 The Architecture of the General Cloud Workflow Partitioning Tool.

cloud; a graphical, in-browser editor is provided to allow users to create new workflows. e-Science Central provides scalable cloud computing as it can distribute the execution of a set of workflows across a set of nodes. This is mainly used to scale throughput (to increase the number of workflow executions per second), but it can also be used to reduce response time where there are opportunities to execute sub-workflows in parallel. The system has achieved scalability at over 90% efficiency, when running on up to 190 cloud nodes, each with two cores.

All accesses to e-Science Central system are policed by a Security service. This ensures that access to all data, services and workflows can be controlled by users. The basis for this is a social networking system that allows users to connect to each other, as well as to create and join groups. Users can chose to keep data, services and workflows private, or can share them with other users, or with groups to which they are connected.

All actions carried out in e-Science Central are recorded in an audit/provenance log. Users can view and query the subset of this log that the security system permits them to see. This allows users to determine exactly how each data item was created (e.g. the graph of service/workflow executions) and how it has been used.

e-Science Central has a Software-as-a-Service interface that allows users to perform all actions through a web browser. In addition, there is an API so that programs can drive the system. This includes all the actions needed to deploy and execute partitioned workflows over federated clouds: storing data, creating and executing workflows. The portability of e-Science Central means that the 'Deployment Manager' in Figure 11 can create and trigger the execution of the workflow partitions in exactly the same way, irrespective of the underlying cloud on which the partition is deployed and executed.

Figure 11 shows two way communication between the 'Deployment Manager' and 'e-Science Central'. The above description of the tool describes the flow of information

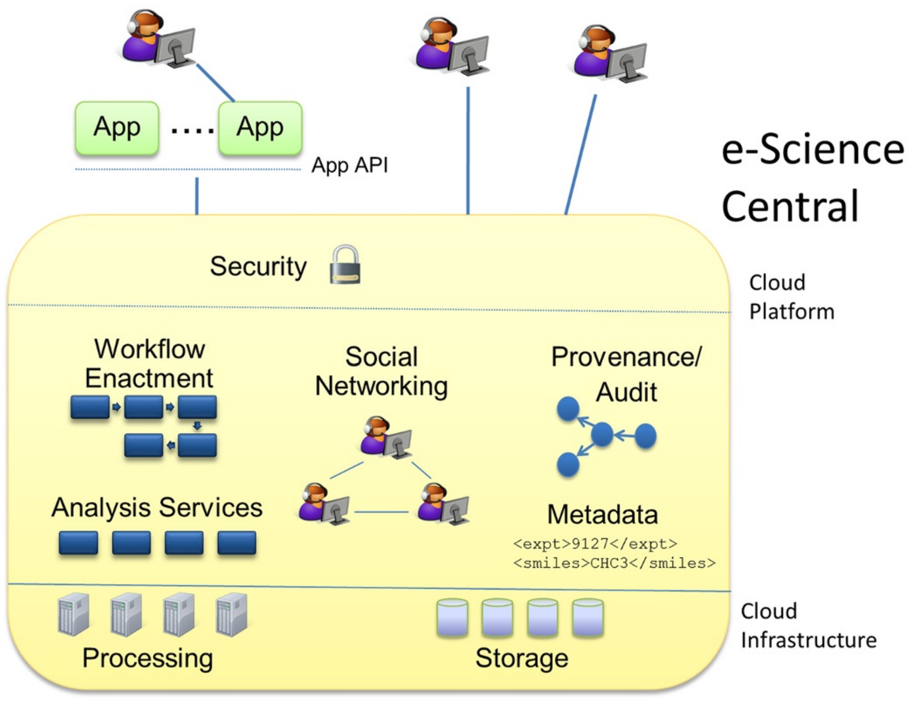

Figure 12 The Architecture of the portable e-Science Central Cloud Platform.

from the 'Deployment Manager' to the 'e-Science Central' instances in the form of workflow partitions to be executed. Information flowing in the reverse direction can include cost updates. If cloud providers start to vary their pricing models frequently, and provide APIs to access it, this information could also be fed back into the Deployment Manager so that the cost models can be kept up-to-date. e-Science Central also has a sophisticated provenance capture and analysis system [10]; this includes collecting information on data sizes and service execution times that can be used to improve the accuracy of the estimated block costs that feed into the cost models.

The Deployment Manager could also monitor the availability of the clouds, for example by sending regular "ping" requests to the e-Science Central deployments, or by including time-outs in the calls to execute a workflow partition on a cloud. This would then allow it to select dynamically between possible workflow partitionings based on which clouds are currently available. Of course, there is a danger that all valid options depend on the availability of one particular cloud (as in the running medical workflow example). Rather than discover this at run-time, it is possible to determine this statically using the tool. This can be done by running the tool multiple times, each time omitting one cloud. This will determine whether the execution of a workflow is critically dependent on a single cloud (in this case the tool will generate no valid options). Having done this, an organization for whom the workflow is business-critical could ensure that the cloud in question has sufficiently high levels of availability, or identify (or create) a second cloud with a sufficiently high security clearance that could also be used by the workflow if the other failed.

Related work

The motivation for this paper came from the author's experience of cloud applications with security constraints (e.g. healthcare applications in the 'Social Inclusion through the Digital Economy' (SiDE) project [11]). However, the general concern that security was a barrier to use of the cloud for many organizations and applications has been widely discussed [1]. The general issues associated with security and clouds are discussed in [12]. A high-level approach to deciding where an application could be deployed is discussed in [13]. Another approach to eliciting and exploiting information on the security and other properties of clouds is described in [14]. These methodologies could be valuable in assigning security levels to clouds, services and data: something which is orthogonal to the scheme described in this paper.

In [4], Bell-LaPadula is also applied to workflow security. Petri Nets are used to model the workflow, rather than the approach taken in this paper. However, the key difference is that its scope does **not** extend to considering the deployment of blocks within a workflow across a set of computational resources, as this paper does. It also differs in including *clearance* but not *location* in its embodiment of Bell-LaPadula.

There has been a large body of work on using cost models to predict execution times in order to select between options for deploying workflows over grids and clouds [15,16]. However, perhaps due to the relatively recent introduction of pay-as-you-go cloud computing, there is much less work on using price-based cost models. In [17], both execution time and price-based models are used to compare a set of options for allocating a workflow over local resources and a public cloud. The work in [18] uses non-linear programming to generate options for using clouds to execute a workflow. Security is not a consideration in any of these papers.

Once the partitioning of a workflow over a set of clouds has been decided, a distributed workflow enactment engine is needed to actually run the workflow. The issues around this are discussed in [19] and a solution is proposed.

Conclusions

This paper has described a new method for automatically determining valid options for partitioning a workflow of services and data across a set of clouds based on security requirements. A cost model is then used to choose between the available options. The main contribution is to show how multi-level security, which has previously been applied to workflows, can be extended to encompass the allocation of workflow services and data to clouds. This has demonstrated that the need for inter-cloud data transfers raises interesting potential security violations that need to be addressed; in the running medical workflow example, this ruled out over half of the possible partitioning options. Although the paper focuses on workflows, the method can be applied to other distributed systems whose components are to be partitioned over a set of clouds.

The tool developed to implement the method has proved invaluable in two ways. Firstly, it removes the chance of human error in applying the various stages of the method to the input workflow. To reinforce this advantage, it can automatically generate both html and LATEXtables, diagrams and reports (the LATEXtables were used in this paper). Secondly, developing the tool forced us to think about how best to structure the implementation of the method, which resulted in a very general system that operates on rules, transforms and cost models. Whilst the focus of this paper is on the multi-level security rules and transforms, as described in sub-section "Generalising the approach", we have been exploring the extension of the approach to other non-functional requirements such as dependability. Whilst this work is ongoing, even in its current form the method can illuminate dependability issues; for example, analysing the set of valid options can highlight the dependency of the workflow on a specific cloud (as in the running medical workflow example). This will allow an organization which is dependent on the workflow to ensure that this cloud has sufficiently high levels of availability, or encourage it to identify a second cloud with a sufficiently high security clearance that could also be used by the workflow.

Overall, the hope is that the approach described in this paper can move the process of partitioning workflows over federated clouds from one in which a human administrator makes an informed but ad-hoc choice, to one in which a tool, such as the one built to implement this method, can determine the valid options based on a rigorous underlying set of rules, and then suggest which is the best, based on a cost model. The approach therefore has the advantage that it can reduce both security violations and execution costs.

Competing interests
The author declare that he have no competing interests.

Acknowledgements
The author would like to thank Leo Freitas, John Mace, Paolo Missier, Chunyan Mu, Sophie Watson, Feng Hao, Zhenyu Wen, Simon Woodman and Hugo Hiden for their comments and suggestions. This work was funded by the Research Councils UK 'Social Inclusion through the Digital Economy' project EP/G066019/1.

Received: 2 February 2012 Accepted: 5 June 2012
Published: 28 July 2012

References
1. Armbrust M, Fox A, Griffith R, Joseph AD, Katz RH, Konwinski A, Lee G, Patterson DA, Rabkin A, Stoica I, Zaharia M (2009) Above the clouds: a berkeley View of Cloud Computing, Technical Report UCB/EECS-2009-28, EECS Department, University of California, Berkeley. http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.html
2. Landwehr CE (1981) Formal models for computer security. ACM Comput Surveys 13: 247–278
3. Bell DE, LaPadula LJ (1973) Secure Computer Systems: Mathematical Foundations. Tech. rep., MITRE Corporation
4. Knorr K (2001) Multilevel Security and Information Flow in Petri Net Workflows. In: Proceedings of the 11th Conference on Advanced Information Systems Engineering
5. Graefe G (1990) Encapsulation of parallelism in the Volcano query processing system. In: Proceedings of the 1990 ACM SIGMOD international conference on, Management of data, SIGMOD '90. ACM New York, pp 102–111. http://doi.acm.org/10.1145/93597.98720
6. Watson P, Hiden H, Woodman S (2010) e-Science Central for CARMEN: science as a service. Concurr Comput: Pract Exper 22: 2369–2380. http://dx.doi.org/10.1002/cpe.v22:17
7. Jones SP (2003) Haskell 98 language and libraries: the Revised Report. Cambridge University Press. ISBN 0521826144
8. Ellson J, Gansner E, Koutsofios L, North SC, Woodhull G (2002) Graphviz – open source graph drawing tools. Graph Drawing 2265: 483–484. http://www.springerlink.com/index/bvdkvy7uj1plml8n.pdf
9. Kelly MC (1989) The Works of Charles Babbage. William Pickering, London
10. Woodman S, Hiden H, Watson P, Missier P (2011) Achieving reproducibility by combining provenance with service and workflow versioning. In: 6th Workshop on Workflows in Support of Large-Scale Science
11. Social Inclusion through the Digital Economy. www.side.ac.uk
12. Mace J, van Moorsel A, Watson P (2011) The case for dynamic security solutions in public cloud workflow deployments, dsnw. IEEE/IFIP 41st International Conference on Dependable Systems and Networks Workshops. pp 111–116. http://csdl2.computer.org/csdl/proceedings/dsnw/2011/0374/00/05958795-abs.html
13. Capgemeni (2010) Putting Cloud security in perspective. Tech. rep., Capgemeni
14. Pearson S, Sander T (2010) A mechanism for policy-driven selection of service providers in SOA and cloud environments. In: New Technologies of Distributed Systems (NOTERE), 2010 10th Annual International Conference on. pp 333–338
15. Yu J, Buyya R, Tham CK (2005) Cost-based scheduling of scientific workflow applications on utility grids. In: First International Conference on e-Science and Grid Computing (e-Science'05). pp 140–147
16. Singh G, Kesselman C, Deelman E (2007) A provisioning model and its comparison with best-effort for performance-cost optimization in grids. In: Proceedings of the 16th international symposium on High performance distributed computing - HPDC '07. ACM Press, New York, pp 117–126
17. Deelman E, Singh G, Livny M, Berriman B, Good J (2008) The cost of doing science on the cloud: the Montage example. In: Proceedings of the 2008 ACM/IEEE conference on Supercomputing, SC '08. IEEE Press, Piscataway, pp 50:1–50:12. http://dl.acm.org/citation.cfm?id=1413370.1413421
18. Pandey S, Gupta K, Barker A, Buyya R (2009) Minimizing Cost when Using Globally Distributed Cloud Services: A Case Study in Analysis of Intrusion Detection Workflow Application. Tech. rep., Cloud Computing and Distributed Systems Laboratory, The University of Melbourne, Australia, Melbourne, Australia
19. Woodman S (2008) A programming system for process coordination in virtual organisations. PhD thesis, School of Computing Science, Newcastle University

doi:10.1186/2192-113X-1-15
Cite this article as: Watson: A multi-level security model for partitioning workflows over federated clouds. *Journal of Cloud Computing: Advances, Systems and Applications* 2012 **1**:15.