# A Comprehensive Sanitization Approach for Workflow Provenance Graphs

Noha Nagy Mohy
Information Systems Department,
Faculty of Computers and Information,
Cairo University, Egypt
n.nagy@fci-cu.edu.eg

Hoda M. O. Mokhtar
Information Systems Department,
Faculty of Computers and Information,
Cairo University, Egypt
h.mokhtar@fci-cu.edu.eg

Mohamed E. El-Sharkawi
Information Systems Department,
Faculty of Computers and Information,
Cairo University, Egypt
m.elsharkawi@fci-cu.edu.eg

## ABSTRACT

As the number of provenance aware organizations increases, particularly in workflow scientific domains, sharing provenance data becomes a necessity. Meanwhile, scientists wish to share their scientific results without sacrificing privacy, neither directly through illegal authorizations nor indirectly through illegal inferences. Nevertheless, current work in workflow provenance sanitizing approaches do not address the disclosure problem of sensitive information through inferences. In this paper, we propose a comprehensive workflow provenance sanitization approach called ProvS that maximize both graph utility and privacy with respect to the influence of various workflow constraints. Experimental results show the effectiveness of ProvS through testing it on a graph-based system implementation.

## Keywords

Graph anonymity; Graph privacy; Secure provenance graph.

## 1. INTRODUCTION

Securing provenance particularly in workflow domain is a significant challenge that is still open for research [1], [4], [6], [7], [15], [22]. As provenance stores the history, in some cases provenance are becoming more precious than traditional data. Hence, it can give a chance for adversary users to employ it for breaking security and attacking privacy. Sharing provenance is needed, particularly in scientific workflows. While scientists may wish to share their experiments and results with others, they may have privacy concerns about their scientific results. Also, releasing provenance query graph results should meet strict security rules to prevent disclosure of sensitive information.

Securing provenance data has been studied in recent years. Hiding, anonymization, and grouping are well known sanitization approaches that are used to preserve privacy of sensitive provenance graph components[nodes/edges] [3], [6], [8], [9], [12]–[14], [18], [19], [25]. In fact, these approaches vary in terms of graph utility, privacy and conformance to provenance policies that guarantee the completeness and correctness of the provenance graph [3], [14]. Hiding approach removes the required sensitive graph components which cause dangling nodes and edges [11]. Anonymization approach only hides the identification attributes. This approach gave anonymization approach the privilege of satisfying provenance graph policies and increasing the graph utility at the same time [14], [29]. However, they do not guarantee the privacy of sensitive information as the attacker can re-identify the anonymized graph components. Grouping approach provides an abstract graph view that preserves the privacy of sensitive graph components, their major drawback is decreasing the graph utility [5], [19]. In addition, they require checking the resulting graph validity according to provenance policies and handling invalid graphs either by grouping more nodes and edges or inventing new dummy nodes to satisfy the provenance graph policies [13].

On the other hand, in the application domain, workflow systems are impacted by industrial laws and regulations that control the workflow execution to ensure the compliance of business rules and regulations that prevent business failures [31]. There are multiple types of workflow constraints that specify the control flow between business processes, define restrictions on the resources or define exception handling procedures. Actually, these workflow constraints are good seeds for attacking privacy particularly in a provenance area as provenance stores a complete history of the workflow execution.

This paper focuses on the problem of attacking graph privacy by re-identifying sanitized graph components through using domain knowledge. The domain knowledge that we address is the workflow constraints. We study the main factors that affect the provenance graph sanitization privacy, utility, and provenance graph policy. The paper introduces a sanitization approach called ProvS that utilizes anonymization, grouping and workflow constraints to produce a set of sanitization actions. These sanitization actions need to be applied to the workflow provenance graph to preserve its privacy while considering the graph utility and provenance graph policies. The key contributions of this paper are:

- Proposing a comprehensive sanitization approach tailored to preserve privacy of workflow provenance graphs.
- Handling the privacy problem in anonymization approach
- Increasing the graph utility of the grouping approach
- Automatically decides which properties (nodes/edges) of the graph need to be grouped and which properties need to be anonymized without user intervention.

The rest of the paper is organized as follows: Section 2 presents a brief background information about provenance graphs in addition to the various types of workflow constraints. Related work is reviewed in Section 3. A motivating example is presented in Section 4. Section 5 introduces the proposed approach ProvS. Finally, Section 6 concludes the paper and presents some points for future work.

## 2. BACKGROUND

This section provides a brief introduction to the most common keywords that will be used along this paper.

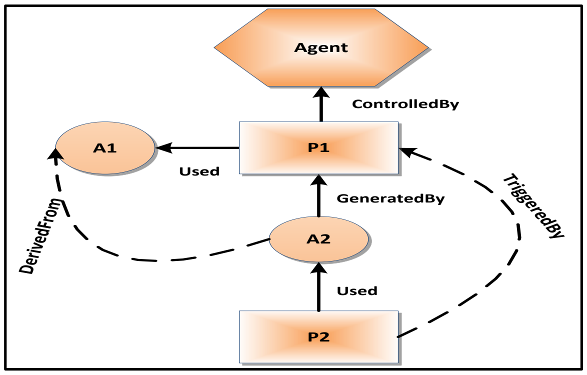

**Figure 1. OPM [20]**

### 2.1 Workflow Provenance Graph

Workflow provenance records the history of workflow executions [17]. There are several tools for capturing workflow provenance [2], [23], [29]. These tools capture information about the sequence of workflow process executions used to produce a data item, as well as the intermediate data passed between these processes. Most of these tools use the Open Provenance Model (OPM) that was proposed in [20] as their standard model for representing provenance data. OPM in Figure 1 models provenance as a Directed Acyclic Graph (DAG) which consists of three types of nodes: artifacts represent the data used, processes represent actions performed on or caused by artifacts, and resulting in new artifacts, and agents that represent actors executing the process. The edges of the OPM graph represent a relationship between two nodes. Provenance graphs are captured automatically by the workflow system. Provenance graph can be formally defined as G=(V,E,L,F) where V is the set of vertices, $E \subseteq V \times V$ is a set of edges, L is a set of labels, and F is a labeling function $F:E \rightarrow L$ that assigns each edge a label.

There are some provenance policies that ensure the correctness and completeness of the provenance graph which discussed in [14].

**Table 1. Provenance graph policy [14]**

| Provenance Policy | Description |
|---|---|
| No Write Conflict (NWC) | A data artifact can be written by only one process |
| No Cyclic Dependency (NCD) | There is no cycle between any two nodes |
| No-Type Error | Two nodes with a direct dependency are of different types. |
| No-False Dependency (NFD) | Two nodes are dependent in the resulted graph only if they are dependent in the original graph |
| No-False Independency (NFI) | Two nodes are independent in the resulted graph only if they are independent in the original graph |

### 2.2 Workflow Constraints

The executions of workflow systems are always governed by a set of constraints that guarantee the correctness of the execution [24]. These constraints can be defined by the user or can be captured from the Business Process Model (BPM) that acts as a schema for the workflow system. Processes and its related constraints are the key elements of workflow constraints. There are different types of workflow constraints: some constraints based on data values where the end result of the process determines the following workflow. Constraints that control the flow of the processes, constraints based on time while other constraints based on roles that determine who is responsible for what. A brief description of workflow constraints that are considered in this paper are outlined in Table 2.

**Table 2. Workflow constraints**

| | Constraint Name | Description |
|---|---|---|
| **Actor Constraints** [28] [31] | Separation of Duty (SOD) | Mutual exclusive processes must be executed by different persons or roles |
| | Binding of Duty (BOD) | Multiple processes must be executed by the same person or role. |
| | Role Constraint (RC) | Process must be executed by a specific role(s) |
| **Process Constraints** [26][32] 15] | Sequential Processes (SP) | A process output is used as an input to another process |
| | Parallel Processes (PP) | Processes are working concurrently, they start at the same time |
| | Multiple Merge Pattern (MMP) | The process is taking its input from multiple processes |
| | Same Input Processes (SI) | Multiple processes are using the same input data |
| | Exclusive Processes (EP) | Only one of the exclusive processes is executed |
| | Time Constraints (TC) | The execution time of a process depends on the execution time of another process |
| **Data Constraints** | Certain Process (CP) | If the condition on a process output data is true this output should be used by a certain process |
| | Different workflow (DWF) | If the condition on a process output data is true this output must be used by a specific process certain number of times |

### 2.3 Graph Privacy and Utility

Workflow provenance graphs contain sensitive information. For example, in healthcare, activities including patient diagnoses, treatments, and processes performed by health care professionals are considered sensitive information. A major goal of preserving privacy is to assure that sensitive information is properly protected. Hence, we need to define what is meant by sensitive information. *Sensitive information* is the information that needs to be hidden from unauthorized users. In this paper, sensitive information can be a provenance graph node (process, data, or actor) or a provenance graph edge (used, generated by, controlledBy).

We formalize two privacy goals in workflow provenance graphs.

- ***Node Privacy:*** node privacy concerns with hiding the identity or attribute values of a node. Let G be the original graph and V be a node belonging to G. Let G' be the sanitized graph view of G. The privacy of V means if an adversary is given G' and extra domain knowledge information then he should not reveal the identity of E in the original graph G.

- ***Edge Privacy:*** edge privacy concerns with hiding the relationships between two of the graph nodes. Let G be the original graph and E be an edge belonging to G. Let G' be the sanitized graph view of G. The privacy of E means if an adversary is given G' and extra domain knowledge information then he should not reveal the existence of E in the original graph G. In this paper, Edge privacy can be specified in terms of node privacy as it requires hiding the identity of the two connected nodes. Note that, we do not consider removing edges in our proposed approach.

Inspired by the fact that a process uses input data to produce output data, the following is a subset of the inference rules presented in [30] to prevent disclosure of sensitive information that will be considered in our approach.
**R1:** Revealing the input and output would identify the process
**R2:** Revealing the process and output would reveal the input and revealing the input and process would reveal the output

Utility is concerned with the amount of information presented in the sanitized graphs to be useful for sharing. Utility of the graph is measured by the utility of its nodes and utility of its connections (edges). There are different utility measures for graph. In this paper, we use the utility measure that introduced in [3] where the utility of nodes is measured by the average percentage of nodes in the original graph that are retained the sanitized graph. The utility of edges is measured by the average percentage of edges in the original graph that are retained in the sanitized graph.

## 3. RELATED WORK

Provenance sanitization via provenance graph transformation is discussed by several researchers [32], [33], [18], [13], [34]. The authors in [35] proposed ZOOM which controls access to sensitive provenance data by driving provenance views from workflow views. The main advantage of ZOOM it allows provenance to be derived at different levels of granularity. However, it cannot be used for complex scientific workflows. The authors in [11] proposed an anonymization approach named Surrogate Parenthood that derives a protected graph G' from the original graph G that preserves the provenance policies. In Surrogate Parenthood each path in G' exists in G. The main advantage of this work is the fact that it preserves the utility of the original graph. The authors in [12] presented a framework called ProPub to publish provenance data based on the data log which stores user privileges, portions of the graph that need to be abstracted, deleted, or retained, as well as graph policies. The main advantage of this framework is that it detects conflicts between sanitization policy and provenance policies.
The authors in [19] proposed a model known as Provenance Abstraction Model (PAM) and implemented a tool called ProvAbs which uses a grouping approach with a defined clearance level of both graph components and users to get a secured graph view. The main advantage of this model is that the resulting graph preserves the confidentiality of the provenance graph. On the other hand, it may require multiple grouping to preserve to provenance policies. The authors in [6] proposed a graph grammar rewriting rules that generate a safe provenance graph. Their proposed rewriting rules involve some graph operations such as vertex contraction, path contraction, edge contraction, and node relabeling. The main limitation of this work is its negligence of provenance policies as well as they do not study the disclosure threat.

To conclude, previous attempts proposed different approaches to preserve the privacy of the OPM provenance graph but sacrifice either the graph privacy or the graph utility. Grouping approaches preserve the privacy of the OPM graph, but it sacrifices utility in addition, it requires handling conflicts between the resulted graph and the provenance policies. On the other hand, anonymization approaches increase the utility of the sanitized OPM graph (as it just changes the nodes to less sensitive ones and do not change the structure of the graph) but they do not guarantee its privacy. Moreover, anonymization and grouping depend on the user specification which is very hard specifically for large OPM graphs. Nevertheless, previous efforts ignored the importance of preserving graph privacy and provenance policies while preserving an acceptable level the graph utility. In the remaining of this paper, we present a novel approach for protecting provenance graphs from disclosing sensitive information that caused by workflow constraints without sacrificing neither graph privacy nor its utility.

## 4. MOTIVATING EXAMPLE

In this Section, we show how the knowledge of workflow constraints may allow adversary users to infer sensitive information from a sanitized workflow provenance graph. In the following discussion, Pa denotes an anonymized view of a process node. Da denotes an anonymized view of a data node. While Pg and Dg denote a grouped process node and a grouped data node respectively.

Figure 2 represents graph G which is a fragment of a workflow provenance graph. Consider the following set of constraints that govern the workflow execution where P1, P2, P4, P5 are workflow processes.

C1: P1 and P2 have the same input data
C2: P4 must be executed by a manager
C3: P2 and P5 must be executed by the same actor
C4: P2 and P4 are sequential processes
C5: If the output data of P4 is greater than 5 then it must be processed by P5

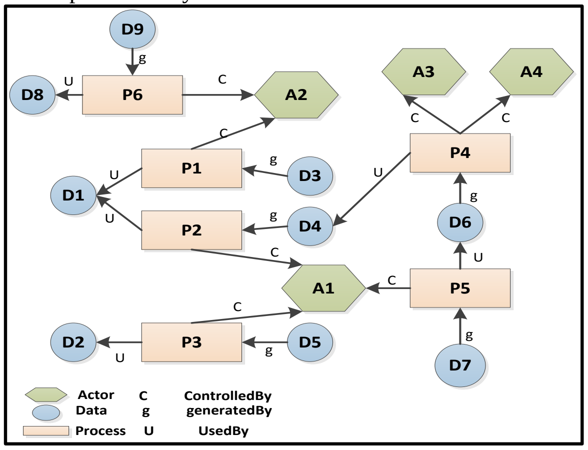

**Figure 2. Workflow provenance graph G**

In the following discussion, we discuss two scenarios with different types of sensitive information and the effectiveness of different sanitization approaches mainly anonymization and grouping to prevent disclosure of sensitive information.

**Scenario 1:** Sensitive information is a process node P2. Figure 3 represents different sanitized views of graph G that are displayed in Figure 2. Figure 3(a) represents G1, which is produced using anonymization. P2 is anonymized to Pa and its output D2 is

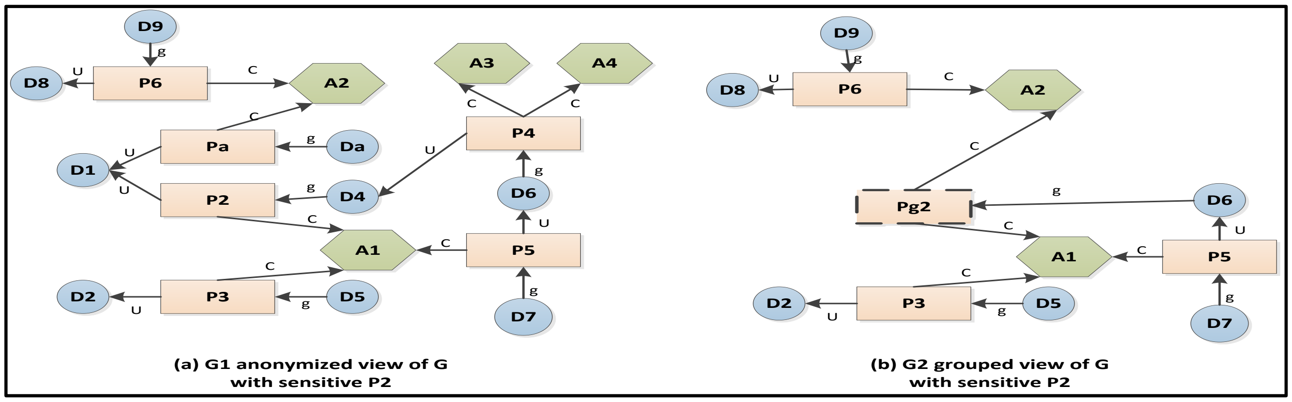
(a) G1 anonymized view of G with sensitive P2
(b) G2 grouped view of G with sensitive P2
D9
D8
P6
A2
A3
A4
Pa
Da
D1
P2
D4
P4
D6
A1
P5
D2
P3
D5
D7
Pg2
U
g
C

**Figure 3. Sanitized views of G in case of P2 is the sensitive information**

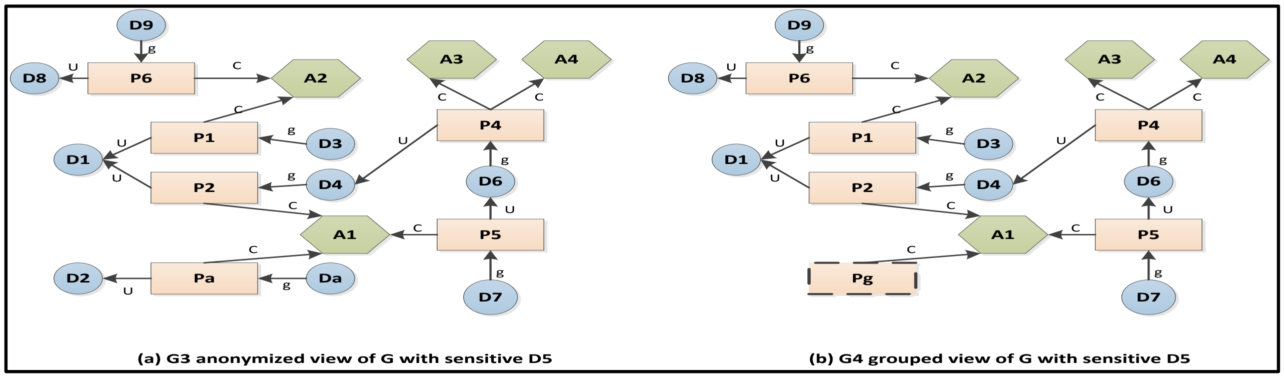
(a) G3 anonymized view of G with sensitive D5
(b) G4 grouped view of G with sensitive D5
D9
D8
P6
A2
A3
A4
P1
D3
D1
P2
D4
P4
D6
A1
P5
D2
Pa
Da
D7
Pg
U
g
C

**Figure 4.Sanitized views of G in case of D5 is the sensitive information**

anonymized to Da to ensure that Pa cannot be detected from its input and output data values. G1 prevents direct attacks to P2. However, an attacker can infer that Pa is P2 via constraint C1. The utility of graph G1 is [node utility=19/19 and edge utility=19/19].

Using the grouping approach, P1, P2 and their output data D3 and D4 are grouped to process Pg1. This grouping will preserve the privacy of P2, however, it violates the provenance policies as two nodes from the same type Pg1 and P4 are connected. Hence, we need to group more nodes to satisfy all the provenance policies. Figure 3 (b) represents graph G2 that is produced via grouping P4 and its actors with Pg1 and D1 to Pg2.we had to group D1 to handle NFD policy (Table 1). G2 preserves the privacy of sensitive process P2 and preserves the provenance policies. However, it affects the graph utility (node utility=12/19, edge utility=12/19) as it groups many of the graph components.

**Scenario 2:** Sensitive information is a data node D5. Figure 4(a) represents G3 that anonymizes D5 to Da and anonymizes P3 to Pa. the utility of G3 is [the node utility=19/19 and edge utility=19/19]. Figure 4(b) represents G4 that groups P3, D5, and D2 to Pg. Both G3 and G4 preserve the privacy of D5, but the utility of G3 has been better than G4 utility [node utility= 16/19 and edge utility=16/19].

This example proves that one approach will not fit in all cases. The choice of the optimal sanitization approach depends on the defined workflow constraints.

## 5. ProvS Architecture

In the following discussion, we state some general assumptions to ensure the effectiveness of our proposed approach. Those assumptions basically define the general workflow structure to distinguish other structures that can be a result of firing workflow constraints. First, workflow processes are in sequential order. Second, the cardinalities of the relationships between the provenance graph components are unknown. Third, different processes are executed by different actors (SOD). Fourth, the input provenance graph is valid according to provenance graph policy (see Table 1). We used the graph utility measure that proposed in [18]. Finally, ProvS produces the set of sanitization actions that need to be applied to a provenance graph to be secured. We are not concerned with how the nodes will be anonymized as there are many approaches exits in the literature for anonymization [27] [16]. The architecture of ProvS is portrayed in Figure 5. The architecture encapsulates two main phases which will be described in the following discussion.

### 5.1 The Design Phase

The design phase is concerned with collecting and refinement workflow constraints by the workflow experts to be used for controlling the workflow execution. This phase is performed only once in offline. It may be repeated only when workflow constraints are updated. Actually, the main incentive behind studying workflow constraints are twofold:

- Sanitization will work under supervision of a Workflow Management System (WFMS) that enforces these constraints during workflow runs. Therefore, the workflow constraints will play an integral part in securing workflow provenance graphs.
- Determining what nodes need to be anonymized and what nodes need to be grouped in order to preserve the graph privacy with respect to the knowledge of workflow constraints.

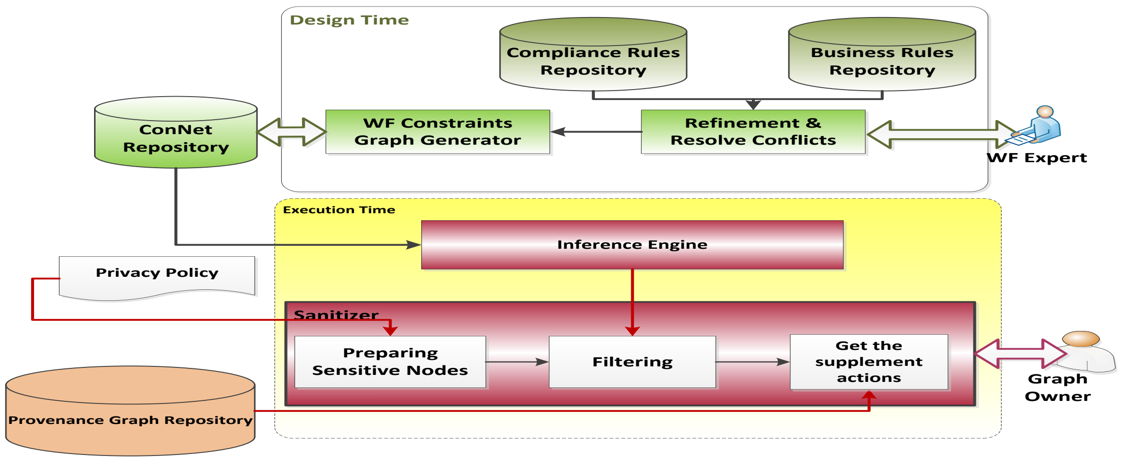
Design Time
Compliance Rules Repository
Business Rules Repository
ConNet Repository
WF Constraints Graph Generator
Refinement & Resolve Conflicts
WF Expert
Execution Time
Inference Engine
Privacy Policy
Sanitizer
Preparing Sensitive Nodes
Filtering
Get the supplement actions
Graph Owner
Provenance Graph Repository

**Figure 5. ProvS architecture**

Workflow constraints specify either the values of nodes' attributes or the relationships between processes. For illustration, BOD constraint defines a link structure as multiple process nodes are connected to the same agent node through controlledBy relationship. MMP constraint defines a link structure as multiple data output nodes are connected to the same process node. SP constraint defines a link structure as a process output node is connected to the other process node. PP constraint defines a value of two process nodes, it does not specify any link structure between them. Similarly, PEP constraint defines a value of a process node.

From the previous discussion and our motivation example we can conclude some remarks, First, constraints that affect the identity disclosure, attribute disclosure or define the general workflow structure (discussed earlier in this section) can be secured using anonymization approach. On the other hand, constraints which specify a link structure different from the general workflow structure cannot be secured against illegal inferences using an anonymization approach as the structure of the graph is the key player and anonymization does not hide this structure. Therefore, it should be sanitized using a grouping approach to hide the graph structure. A complete assessment of workflow constraints using anonymization and grouping against privacy is discussed in [21].

Subsequently, we classify the workflow constraints (Table 2), according to their influence on graph privacy as follows:

- ***Configuration constraints*** are the types of constraints that define link structure different from the general workflow structure.
- ***Identity constraints*** are the types of constraints that define a node identity or attribute values such that relate actor or role to a process or define relationships between processes and/or data.

Orthogonal to this classification, Binding of duty, different workflow and same input constraints are considered configuration constraints while the other constraints are considered identity constraints.

**Table 3. Workflow constraints classification**

| Classification | Workflow Constraints |
|---|---|
| Identity constraints | CP, SOD, SP, PP, EP, TC, RC |
| Configuration constraints | BOD, MMP, SI, DWF |

Table 3 classifies the workflow constraints presented in Table 2 according to our constraints classification approach. As we mentioned, SP is the general graph structure therefore, it will not be considered as a configuration constraint.

Driven by the following facts: workflow constraints may have different influence on the privacy of OPM graphs. 1) Some constraints might not affect the privacy of OPM graph, 2) One constraint may be used to reveal sensitive information; 3) Multiple constraints can be combined to reveal sensitive information. We propose a constraint network graph (ConNet) which offers a means of identifying potential relationships between workflow components.

ConNet is an undirected graph that represents the workflow constraints exist among different workflow components. A ConNet node can be a process, data, or actor that exists in a workflow constraint. The edges represent constraints between the connected nodes. These edges are labelled with the workflow constraint name that governs the execution of the two connecting nodes. The main incentive behind this graph is to discover all the paths that an attacker can go through to infer sensitive information. ConNet is constructed as follows:

```
For each constraint in the workflow constraints
    1. Draw a node for each OPM component in the constraint
    2. Label these nodes with the component's name in the constraint
    3. Connect the nodes in each constraint with an edge and label this edge with the constraint name (specified in Table 2).
```

Continuing with our motivating example discussed in Section 4, the ConNet is illustrated in Figure 6 ConNet graph is stored in ConNet repository to be used later in the execution phase.

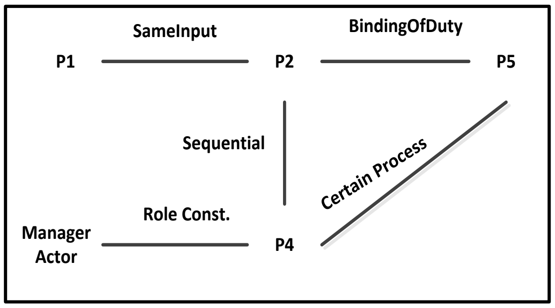

**Figure 6. ConNet graph for the motivating example**

## 5.2 The Execution Phase

This phase is concerned with generating the sanitization actions that need to be applied to a workflow provenance graph based on privacy policy and present these sanitization actions to the graph owner. The main components of this phase are:

***Privacy Policy:*** the privacy policy stores the privacy preferences. It contains the sensitive graph components [nodes/edges] that need to be hidden.

***Provenance Graph Repository:*** stores the workflow provenance graphs that need to be sanitized for sharing.

***The Inference Engine Module:*** The main purpose of this module is to find the set of provenance graph nodes that are related to the sensitive information using ConNet graph. The InferenceEngine procedure in Figure 7 generates two sets, one for the anonymized nodes AA set which is composed of individual nodes that are required to be anonymized, while the other is for the grouped nodes GA set which is a set of nodes set which are required to be grouped together. The procedure first checks if the sensitive node N exists in the ConNet (line 2) then, for all the edges that exist in the paths connected to N, it gets the edge label to determine its classification type (line 4-8). Then, based on the edge classification the connected nodes of this edge it is added either to AA or GA (line 9-11).

***The Sanitizer Module:*** the sanitizer has three main functions (i) preparing the sensitive node set, (ii) Filtering; (iii) get the supplement actions. The sanitizer algorithm is presented in Figure 9. Preparing sensitive nodes set by acquiring the sensitive nodes from the privacy policy and for each sensitive edge it get its connected nodes and add them to the sensitive nodes set. Next, the sanitizer sends these sensitive set to the InferenceEngine module (line 5). Further, it filters the AA set and GA set generated from the InferenceEngine. For multiple sensitive nodes which have different sanitization actions the sanitizer merge the anonymization sets and grouping sets. In addition, it uses the following rules in the filtering step.

***Rule 1:*** *If a process exists in both AA set and GA set then remove it from AA* (as grouping is more restricted than anonymization).

***Rule2.*** *If a process exists in multiple groups in GA then merge these groups to one group* (as a node cannot exist in more than one group).

Finally, the sanitizer gets the supplement actions (Line 8 and 221) to handles the limitations of anonymization and grouping. For grouping set it uses Rule 3 (Line 8-12).

```
Procedure: InferenceEngine (Node N, Graph ConNet, AA output, GA output)
1.  Define set AA=$\phi$, GA=$\phi$
2.  IF (N exists in ConNet) Then
3.  Begin
4.    For each path connected to N
5.      For each edge in the path
6.        (N1, N2)= Get the connected nodes
7.        C= Get label of the edge
8.        IF((C=BOD)$\vee$(C=MMP)$\vee$(C=SIP)$\vee$(C=DNC)) Then
9.          GA= {{N1, N2}}
10.       Else
11.         AA= {N1, N2}
12.       End IF
13. End
```

**Figure 7. InferenceEngine procedure**

```
Procedure: RelatedActions (Node N, Graph G, AA output)
1. Begin
2. IF(N is a data node) Then
3.    AA=AA$\cup${process used N}$\cup${process generated N}
4. Else IF (N is a process node) Then
5.    AA=AA$\cup${data generated from N as v}$\cup${processes used v}
6. End IF
7. End
```

**Figure 8. RelatedActions procedure**

***Rule 3:*** *For each grouped process set, group the data outputs nodes of the grouped processes into a data group node* (to prevent NFD and NWC problem in Table 1).

For the anonymization set. It computes the related actions using procedure RelatedActions in Figure 8. Which uses the following rules to generate related actions that need to be added to AA set.

***Rule 4:*** *for each anonymized data node, anonymize its generated process and used process* (to prevent disclosure of this data using R1 and R2 in Section 2.3).

***Rule 5:*** *for each anonymized process in AA anonymize its output data and the process that used this output data* (to prevent disclosure of the anonymized process using R1 and R2).

The sanitizer output the final anonymization set and grouping set to the graph owner.

Continuing with our motivating example discussed in Section 4, for the first scenario, the anonymization set AA= {P4, D6, A3, A4} and the grouping set GA= {{P1, P2, P5}, {D3, D4, D7}}. The resulted graph utility will be [nodes utility=15/19 and the edge utility= 15/19].

**Table 4. Features comparison for sanitization approaches**

| | | **ZOOM** [10] | **Surrogates** [3] | **ProPub** [12] | **ProvAbs** [18] | **ProvS** |
|---|---|---|---|---|---|---|
| **Sanitization Method** | **Anonymization** | X | √ | X | X | √ |
| | **Grouping** | √ | X | √ | √ | √ |
| **Conflict Detection and resolution** | | X | - | √ | √ | - |
| **Study the utility of graph** | | X | √ | X | √ | √ |
| **Preserve Graph Privacy** | | X | X | X | X | √ |

```
The Sanitizer Algorithm
Input: OPM graph G(V,E), Sensitive Set SS, ConNet C(V,E)
Output: Anonymization set AA and Grouping set GA
Method:
1.   Define Anonymization Actions Sets AA
2.   Define Grouping Actions Set GA
3.   Define Output Data Set OD
4.   For each S  in SS
5.     InferenceEngine (S, ConNet, AA,GA)
6.   IF $(AA \neq \phi \vee GA \neq \phi)$ Then
7.      Begin
8.         For each group in GA
9.          Begin
10.              OD=set of all generated data from this group
11.              $GA=GA \cup OD$
12.          End
13.        IF (an element exists in different pairs in GA) Then
14.              union these pairs to be one group
15.            For each  element e in AA
16.                IF$(e \in h$ where $h \in GA)$ Then
17.                    AA=AA-{e}
18.              For each element e in  AA
19.                  RelatedActions(e, G, AA output)
20.       End IF
21.  Else// S is not in the ConNet
22.    Begin
23.        $AA=AA \cup \{S\}$
24.        RelatedActions(S, G, AA output)
25.     End
26.  End
```

Figure 9.The sanitizer algorithm

In the second scenario, the anonymization set AA= {P3, D5, P5} and the grouping set GA=ϕ. The resulted graph utility will be [node utility= 19/19 and edge utility=19/19].

## 5.3 Evaluation

In Table 4, we compare the main features of different existing sanitization approaches against ProvS. It is clear from the table that both Surrogates and ProvS do not cause conflicts in the resulting graph while other approaches detect and solve conflicts. The reason is that anonymization preserves the graph structure and ProvS uses homogenous grouping which handles this issue. For the utility issue surrogates provides a utility measure for paths and nodes to measure the informative of the surrogates. ProvAbs annotates each node in the graph with a utility value which defines the nodes to be retained in the resulted graph. ProvS uses utility for paths and nodes to measure the informative of the resulted graph. For the privacy feature, all the approaches that use grouping preserve the graph privacy.

We perform experiments to ensure the effectiveness of ProvS. Experiments were conducted on a HP PC with 2.53GHz Intel Core i5 CPU, 4GB RAM and 160 disk space running MS Windows 7. Neo4j 2.2.5 graph database is used to store OPM provenance graphs and constraints network (ConNet). Java is used as the main programming language for writing the logic of the code and we used Cypher queries to retrieve data from the provenance graph and ConNet.A set of workflow constraints was generated along with a series of synthetic provenance graphs that satisfy the generated workflow constraints. The experiments were conducted on 3 different sets of workflow constraints. The first a set of type configuration constraints which were sanitized via grouping, the second a set of type identity constraints which were sanitized via an anonymization approach. The third set is mixed half of them from identity constraint type and the other half from the configuration constraint type which was sanitized using a combination of anonymization and grouping. The results of the experiments are shown in Figure 10 , which shows clearly that ProvS increases the graph utility even if the workflow constraints are configuration constraints. This is due to homogenous grouping that decreases the number of graph properties which are needed to be grouped.

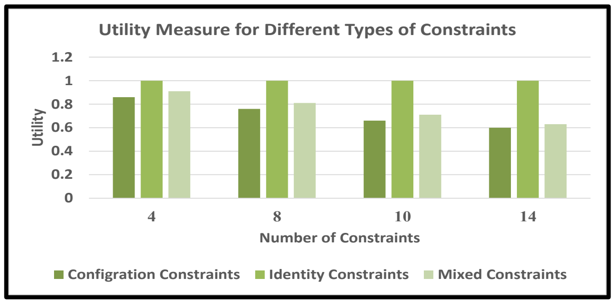

**Figure 10. Utility measure for different types of constraints**

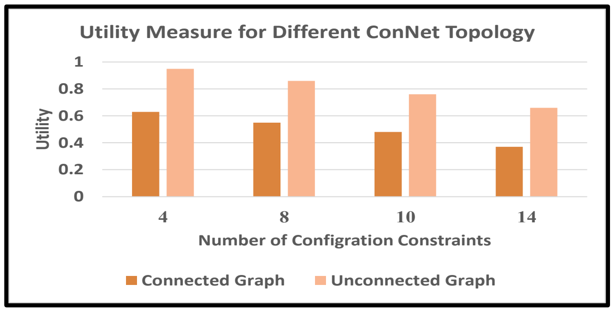

**Figure 11. Utility measure for different ConNet topology**

Additionally, an additional series of tests to determine the effect of ConNet topology on the provenance graph utility in case of configuration constraints. Figure 11 shows a comparison of the ConNet topology versus graph utility while keeping the sensitive information, provenance graph size and ConNet size constant. This comparison shows that unconnected ConNet graphs increase the utility of the graph as the groups in the grouping set will not be overlapping and hence, will not be joined. On the other hand, highly connected ConNet graphs decrease the resulted graph utility as many nodes will be in different groups and hence these groups will be joint to be one group.

## 6. CONCLUSION AND FUTURE WORK

Traditional sanitization approaches which are tailored to secure provenance graphs are no longer sufficient as they ignore an important source of security, namely graph privacy attacks due to the use of workflow constraints. In this paper, we first illustrated the main problems of anonymization and grouping approaches. Then, we highlighted different types of workflow constraints that could be used in disclosing sensitive information. Further, we classified these constraints according to their influence on graph privacy. Consequently, we introduced ProvS that ensures secure and valid provenance graphs. ProvS combines anonymization and grouping for sanitizing sensitive provenance graph. Experimental results show the effectiveness of ProvS through testing it on a graph-based system implementation.

For future work, we plan to enhance the performance of ProvS in the presence of large provenance graphs and large number of workflow constraints. In addition, applying our approach to different case studies.

## 7. REFERENCES

[1] Alsiyami, A., "A Policy Language Definition for Provence in Pervaisve Computing," *Ph.D.Thesis. Univ. Sussex*.

[2] Altintas, I., O. Barney, and E. Jaeger-frank, "Provenance Collection Support in the Kepler Scientific Workflow System," in *Proceedings of the international conference on Provenance and Annotation of Data*, 2006, vol. 4145, pp. 118–132.

[3] Blaustein, B., A. Chapman, L. Seligman, M. D. Allen, and A. Rosenthal, "Surrogate Parenthood: Protected and Informative Graphs," in *PVLDB*, 2011, vol. 4, no. 8, pp. 518–527.

[4] Cadenhead, T., M. Kantarcioglu, and B. Thuraisingham, "A framework for policies over provenance," *Proc. 3rd USENIX Work. Theory Pract. Proven. (TaPP '11)*, 2011.

[5] Cadenhead, T., M. Kantarcioglu, and B. Thuraisingham, "An Evaluation of Privacy, Risks and Utility with Provenance," *Secur. Knowl. Manag. Work. (SKM), Novemb.*, 2010.

[6] Cadenhead, T., V. Khadilkar, M. Kantarcioglu, and B. Thuraisingham, "Transforming provenance using redaction," *Proc. 16th ACM Symp. Access Control Model. Technol. - SACMAT '11*, p. 93, 2011.

[7] Capitani Di Vimercati, S. De, S. Foresti, and P. Samarati, "Access Control Policies and languages," *Int',l J. Comput. Sci. Eng.*, vol. 3, no. 2, pp. 94–102, 2007.

[8] Chebotko, A., S. Lu, S. Chang, S. Member, S. Chang, and F. Fotouhi, "Secure abstraction views for scientific workflow provenance querying," *IEEE Trans. Serv. Comput.*, vol. 3, no. 4, pp. 322–337, Oct. 2010.

[9] Cheney, J. and R. Perera, "An Analytical Survey of Provenance Sanitization," in *IPAW*, 2014, pp. 113–126.

[10] Cohen-Boulakia, S., O. Biton, S. Cohen, and S. Davidson, "Addressing the provenance challenge using ZOOM," *Concurr. Comput. Pract. Exp.*, vol. 20, no. 5, pp. 497–506, 2008.

[11] Davidson, S. B., Z. Bao, and S. Roy, "Hiding Data and Structure in Workflow Provenance," *DNIS*, pp. 41–48, 2011.

[12] Dey, S. C., D. Zinn, and B. Ludäscher, "ProPub: Towards a declarative approach for publishing customized, policy-aware provenance," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 6809 LNCS, pp. 225–243, 2011.

[13] Dey, S. and B. Ludascher, "A declarative approach to customize workflow provenance," *Proc. Jt. EDBT/ICDT 2013 Work. - EDBT '13*, p. 9, 2013.

[14] Dey, S., D. Zinn, and B. Ludáscher, "Repairing provenance policy violations by inventing non-functional nodes," in *CEUR Workshop Proceedings*, 2011, vol. 737, pp. 109–122.

[15] Jurnawan, W. and U. Röhm, "Data provenance support in relational databases for stored procedures," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2009, vol. 5667 LNCS, pp. 97–111.

[16] Kifer, D., "L-Diversity : Privacy Beyond k -Anonymity," *Proc. 22nd Int. Conf. Data Eng.*, vol. 1, pp. 1–36, 2006.

[17] Kumar, M. and M. K. Anand, "Managing Scientific Workflow Provenance," Unversiy od California, 2009.

[18] Missier, P., J. Bryans, C. Gamble, V. Curcin, and R. Danger, "ProvAbs: model, policy, and tooling for abstracting PROV graphs," *Procs. IPAW 2014 (Provenance Annot.*, pp. 3–15, 2014.

[19] Missier, P., J. Bryans, C. Gamble, V. Curcin, and R. Danger, "Provenance Graph Abstraction by Node Grouping," *Sch. Comput. Sci. Tech. Rep. Ser.*, no. August, 2013.

[20] Moreau, L., J. Freire, J. Futrelle, R. E. Mcgrath, J. Myers, and P. Paulson, "The Open Provenance Model : An Overview," *IPAW*, no. August 2007, pp. 323–326, 2008.

[21] Nagy, N., "Assessment of Workflow Constraints and Sanitization Approaches on Provenance Graph Privacy and Utility," *Tech. report, Cairo Univ.*, 2016.

[22] Nguyen, D., J. Park, R. Sandhu, D. Nguyen, and R. Sandhu, "A Provenance-based Access Control Model," *2012 Tenth Annu. Int. Conf. Privacy, Secur. Trust*, pp. 137–144, 2012.

[23] Oinn, T., M. Addis, J. Ferris, D. Marvin, M. Senger, M. Greenwood, T. Carver, K. Glover, M. R. Pocock, A. Wipat, and P. Li, "Taverna: A tool for the composition and enactment of bioinformatics workflows," *Bioinformatics*, vol. 20, no. 17, pp. 3045–3054, 2004.

[24] Pesic, M., "Constraint-Based Workflow Management Systems: Shifting Controls to Users," 2008.

[25] Rachapalli, J., V. Khadilkar, M. Kantarcioglu, and B. Thuraisingham, "REDACT: A Framework for Sanitizing RDF Data," in *Proceedings of the 22nd international conference on World Wide Web companion*, 2013, pp. 157–158.

[26] Runte, W. and M. El Kharbili, "Constraint Checking for Business Process Management," in *GI Jahrestagung*, 2009, pp. 4093–4103.

[27] Samarati, P., "Protecting respondents' identities in microdata release," *IEEE Trans. Knowl. Data Eng.*, vol. 13, no. 6, pp. 1010–1027, 2001.

[28] Schefer, S., M. Strembeck, J. Mendling, and A. Baumgrass, "Detecting and Resolving Conflicts of Mutual-Exclusion and Binding Constraints in a Business Process Context," *Move to Meaningful Internet Syst. OTM 2011*, vol. 7044, no. October, pp. 1–18, 2011.

[29] Silva, C. T., E. Anderson, E. Santos, and J. Freire, "Using VisTrails and provenance for teaching scientific visualization," *Comput. Graph. Forum*, 2011.

[30] Thuraisingham, B., T. Cadenhead, M. Kantarcioglu, and V. Khadilkar, *Secure Data Provenance and Inference Control with Semantic Web*. Auerbach Publications, 2014.

[31] Turetken, O., A. ELGammal, Van Den Heuvel, and M. Papazoglou, "Enforcing Compliance on Business Processes Through the Use of Patterns," in *19th European Conference on Information Systems (ECIS 2011), Finland*, 2011.

[32] White, S. a and I. B. M. Corp, "Process Modeling Notations and Workflow Patterns," *Bus. 21*, vol. 21, pp. 1–25, 2004.