

Cost-Benefit Analysis of Confidentiality Policies for Advanced Knowledge Management Systems

Deborah W. May

March 2003

U.S. Department of Energy

Lawrence
Livermore
National
Laboratory

DISCLAIMER

This document was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the University of California, and shall not be used for advertising or product endorsement purposes.

This work was performed under the auspices of the U. S. Department of Energy by the University of California, Lawrence Livermore National Laboratory under Contract No. W-7405-Eng-48.

This report has been reproduced directly from the best available copy.

Available electronically at <http://www.doc.gov/bridge>

Available for a processing fee to U.S. Department of Energy
And its contractors in paper from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831-0062
Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-mail: reports@adonis.osti.gov

Available for the sale to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Road
Springfield, VA 22161
Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-mail: orders@ntis.fedworld.gov
Online ordering: <http://www.ntis.gov/ordering.htm>

OR

Lawrence Livermore National Laboratory
Technical Information Department's Digital Library
<http://www.llnl.gov/tid/Library.html>

Cost-Benefit Analysis of Confidentiality Policies for Advanced Knowledge Management Systems

Deborah W. May

Lawrence Livermore National Laboratory

Information Operations and Assurance Center

may14@llnl.gov

1 Problem Statement

Knowledge Discovery (KD) processes can create new information within a Knowledge Management (KM) system. In many domains, including government, this new information must be secured against unauthorized disclosure. Applying an appropriate confidentiality policy achieves this. However, it is not evident which confidentiality policy to apply, especially when the goals of sharing and disseminating knowledge have to be balanced with the requirements to secure knowledge. This work proposes to solve this problem by developing a cost-benefit analysis technique for examining the tradeoffs between securing and sharing discovered knowledge.

1.1 Complex Confidentiality Requirements

One reason providing confidentiality for discovered knowledge is particularly difficult is the complexity of the confidentiality requirements of modern information systems. The difficulty arises in many application domains, rang-

ing from the healthcare industry, to financial institutions to government. Throughout this paper we draw on hypothetical examples from government information systems (GIS), especially from the intelligence and law enforcement communities. However, the ideas presented here can be extended to other domains in a straightforward manner.

Traditionally, a GIS labels data and documents with strictly increasing security levels, such as **unclassified**, **secret**, or **top secret**. These labels represent the level of confidentiality that is required for that data. Generally, people or processes have clearance levels that corresponds to data confidentiality levels, and cannot access data that is labelled higher than the clearances they possess. A system that must protect this type of data is called a multi-level secure (MLS) system. Government data is also often compartmentalized, or labelled, based on need-to-know groups. For example, military data and human resource data may be compartmentalized separately. Compartments do not represent a hierarchy of confidentiality requirements; the requirements are different but not strictly increasing. Systems that are required to protect this type of data are called multi-lateral secure. A modern GIS will probably have to provide both multi-level and multi-lateral security; this paper will refer to such a system as a ML/LS system.

Controlling access to data is a complex process involving policies authentication, authorization, integrity, and confidentiality. A ML/LS system has to authenticate users' credentials and enforce authorization rules that state which credentials are required to perform different operations within the system. The authentication and information flow policies both assume that data is already labelled appropriately and that rules can be applied to determine who can have different types of access. Confidentiality and integrity

policies determine how data is labelled. Integrity policies ensure that labels reflect the degree of trust that can be placed in a data item based on the clearance level of the person or process that has written it. Confidentiality policies ensure that labels reflect the clearance level required for a person or process to read the data. Of these four parts of an overall access control policy, confidentiality policies are the focus of this paper.

Although it is not part of the traditional government labelling scheme, it may be helpful to think of three different types of confidentiality. First, is the need for the *existence* of data to be held in confidence. Second, is the need for the *content* of data to be held in confidence. Finally, is the need for the *context* of data to be held in confidence. Consider data about a new secret weapon that a U.S. enemy is developing. The mere existence of this data in a GIS may be highly confidential since it might raise questions about how the U.S. came to know about the weapon in the first place. This is probably not the case for data about a U.S. fighter plane, though in this case content details such as fuel efficiency or weapon precision may be highly confidential. Two types of context information exist. The first is relationship data. Data about a port overseas may not be confidential, and neither may data about a large submarine fleet. However, the fact that the port is the destination of the fleet may be highly confidential, since it reveals mission details. The second type of context information is reporting data. Data about a company reported by a business journal may not be confidential, but the fact that the CIA also reports information about this company may well be confidential. Although this work does not seek to expand the current U.S. government data labelling mechanism, it can still be useful to keep these different types of confidentiality in mind. The knowledge representation scheme presented in Section 1.4 and assumed

throughout this paper automatically accounts for context confidentiality, but existence and content confidentiality play an important role when evaluating security property requirements, particularly when differentiating between nondeducibility and noninterference (see Section 2.3 for more details).

1.2 Capabilities of Advanced KM Systems

Confidentiality labels are often assigned to data or documents by a human expert who understands the information domain. Alternatively, one common way to label data automatically is to assign labels at “system-high,” meaning that every data item is labelled with the highest label the automated system is authorized to use. A large scale KM system, intended for use across multiple organizations, would obviously not operate at system-high. In fact, system-high does not even apply to multi-lateral secure systems. ML/LS solutions have been proposed that do allow data to be secured at different levels and compartments of confidentiality (Section 2.4 describes some of these). However, ML/LS systems still assume that every data item has a label assigned to it at the time it is published. Presumably this label is assigned by a human prior to publication, or the label is inherited from the data processing system that published the data.

Today’s advanced KM systems are creating their own information internally to supplement knowledge that is explicitly published. Advances in knowledge discovery techniques are making such systems possible and popular. The question that arises is how to label internally discovered information such that an acceptable level of confidentiality is maintained. Agat and Sands [1] as well as Hale and Shenoit [24] note that the threat of security leaks brought about by automated discovery is constantly increasing

due to advances in algorithms and computing power. This general problem is compounded in systems where algorithms operate over data with different labels, because there is no single correct way to label the output of the algorithm. Advanced KD capabilities make legacy ML/LS solutions obsolete; new solutions are needed that can address advances in KD algorithms.

1.3 Identifying the Costs and Benefits of Confidentiality Policies

One of the primary goals of a KM system is to share knowledge; this goal is obviously in direct conflict with the need to secure knowledge. One way to label discovered knowledge would be to assign a system-high label. This would certainly prevent unauthorized disclosure, but the price to be paid would be very high, since only a very limited user base would have access to the knowledge. Providing a technique, or framework, that enables decision makers to understand the tradeoffs between sharing and securing knowledge is the goal of this work.

Before developing a technique for doing this, we must first quantitatively define *cost* and *benefit* in the context of a KM system. There are several possibilities here, but we have chosen to define cost as the reduction in knowledge sharing capabilities. This is defined quantitatively throughout Sections 2.1, 3.1, and 3.4. Two other possible costs are a reduction in knowledge quality as a result of applying confidentiality policies and the computational cost of enforcing a confidentiality policy. For the purposes of this paper, the assumption is made that all KD processes operating on the information in the KM system are owned by the KM system and have all confidentiality levels and compartments available to them. Thus, degrada-

tion of quality due to KD is not an issue—this only becomes an issue when KD processes occur outside of the KM system or when system-owned KD algorithms have limited access to data. The performance cost of enforcing a confidentiality policy is likely to be significant. However, this cost is highly dependent on the details of the KM system architecture and the security mechanisms being enforced. Since this paper is focusing on policy rather than mechanism and seeks to apply to a large class of KM systems, this type of cost can not be accurately accounted for.

Defining benefit for a confidentiality policy is not as elusive as defining cost. Benefit is simply the degree of confidentiality provided by the policy. Benefit is defined quantitatively in Sections 2.3, 3.2, 3.3, and 3.4. The quantitative definition, however, is not numerical. Rather, benefit is defined in terms of formal security properties. Thus, directly comparing cost to benefit is not possible. The cost-benefit technique proposed in Section 3.4 takes this into account, but it does require that the decision maker understand the security properties that define benefit, or at least know which property or properties are required for the KM system of interest.

1.4 Motivation by Example

To illustrate the relevance and practicality of this work it is useful to put the problem into the context of a realistic application domain and KM system. It is not the intent to limit the applicability of this work to the domain and system discussed; it is certainly relevant to other domains that have similarly complex confidentiality requirements and advanced knowledge discovery capabilities.

Changes in the structure of the U.S. government and its agencies, partic-

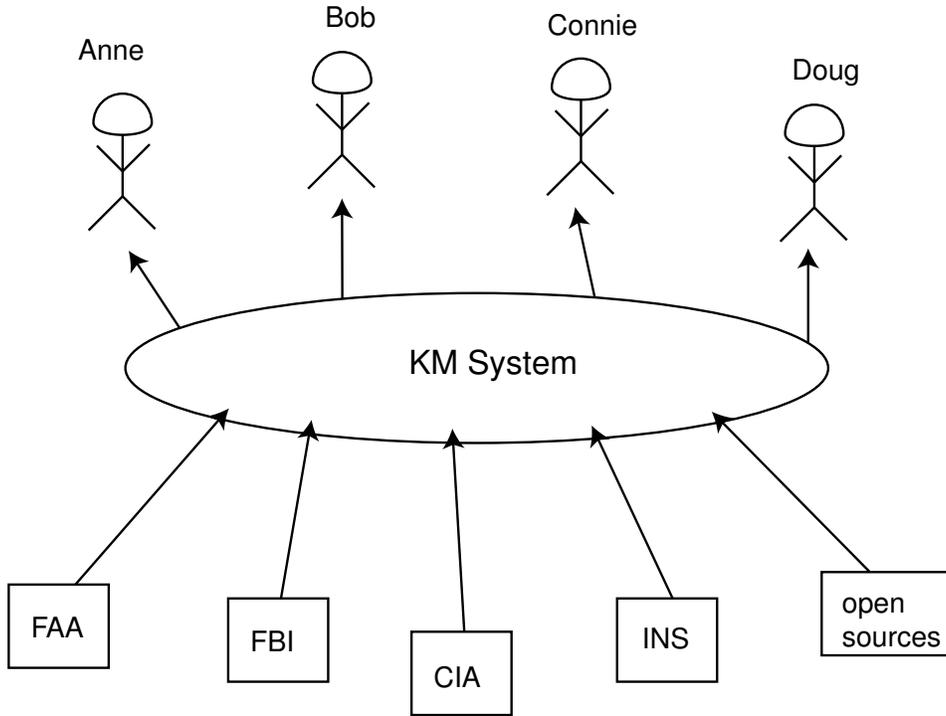


Figure 1: KM Environment

ularly the introduction of the Department of Homeland Security, have highlighted the need to more effectively and efficiently share knowledge across agencies. A shared KM environment is one way to accomplish this. A notional KM environment is shown in Figure 1. Here, several users, all with different jobs and security credentials, utilize the KM system to gain insight to questions pertinent to their jobs. Also, several agencies publish information to the KM system. The published information varies widely in subject area and security requirements. Tables 1 and 2 provide more details about the users and publishers of the KM system. The open sources publisher really refers to a number of possible sources of publicly available information, rather than any specific agency or information system. Compartment

Name	Job	Clearance Level	Compartments
Anne	immigration specialist	U	travel, visas, customs
Bob	intelligence analyst	TS	Russia, Iraq, weapons, travel
Connie	military adviser	S/TS	Iraq, military, /weapons
Doug	detective	S	U.S. Persons, customs, S. America

Table 1: KM System Users: U=Unclassified, S=Secret, TS= Top Secret. Connie has additional privileges (Top Secret and weapons) depending on the current national security conditions.

credentials are only shown for the users, to illustrate the complexity of including multi-lateral security into the system. Throughout the rest of this example, compartments are not used; only classification levels are shown.

It is important to point out that the KM system will in some way combine the information published, so that the resulting knowledge is more than the sum of its parts. A users should be able to get more questions answered by the KM system than could easily be accomplished by using agency-specific information systems.

Although this paper seeks to be as general as possible, and does not intend to propose a KM system design, it is difficult to illustrate important concepts without having a knowledge representation (KR) scheme in mind. The one presented here is overly simplified, but should be convenient for illustration purposes. The KR scheme chosen has two elements, nodes and

Publisher	Subject	Highest Classification Level
FAA	airline travel	U
FBI	U.S. law enforcement and investigations	TS
CIA	foreign intelligence	TS
INS	immigration and border control	U
open sources	news, maps, web pages, public data	U

Table 2: KM System Publishers

links. Nodes represent subjects or objects and links represent relationships. A subject-relationship-object triple is a fact or statement. Combining facts creates a graph of arbitrary size and topology. Since links represent a relationship, they have a declarative meaning signifying that relationship. Such links are called *semantic links* and a graph with semantic links is called a *semantic graph*. Since this system is assumed to create knowledge, both by the way it combines heterogeneous information and by applying knowledge discovery processes, we will refer to the graph-represented information as the Semantic Knowledge Graph (SKG).

Nodes in the SKG have several fields, shown in Table 3. The *ID* is used only to uniquely identify the node. *Type* indicates the class of node (person, building, or university, for example). The *content* is the value of the node. This could be a report, an image file, a list of attribute values, or just a single value. Our example uses single values only. The remaining fields are reporting fields. Since several agencies may publish information about the same subjects, multiple sets of reporting fields are allowed.

Multiplicity	Field
1	type
1	ID
1	content
1 or more	classification level [optional] compartments publisher [optional] confidence or quality rating

Table 3: Node Template

Links in the SKG have a different set of fields, shown in Table 4. *Source ID* and *destination ID* uniquely identify the node and indicate which two nodes it connects. *Type* is the semantic meaning of the link. The reporting data is the same as the node reporting data fields.

A SKG cannot evolve completely arbitrarily if it is to be predictable enough to query effectively. There must be some rules that indicate what types of nodes and links can combine to form facts, and how to format different fields. An ontology serves this purpose. An ontology is a specification, or schema, that states these rules using a formal specification language. A sample ontology is not presented here, but it is important to keep in mind that a certain structure is imposed on knowledge by the ontology. This structure must be carefully selected, so as not to limit the expressiveness of the SKG too much. The knowledge metrics that will ultimately be used to define cost in this work (described in Sections 2.1.1, 3.1, and 3.4), may not be accurate and effective metrics if the expressiveness of the ontology is too limited.

Multiplicity	Field
1	type
1	source ID
1	destination ID
1 or more	classification level [optional] compartments publisher [optional] confidence or quality rating

Table 4: Link Template

There are many other design issues involved in developing a KM system that are outside the scope of this work. The KM system and KR scheme presented above are intentionally simplified to provide a useful vehicle to motivate this work. Since the focus of this work is on the KD and confidentiality aspects of the KM system, aspects not related to these topics have not been discussed. The remainder of this section assumes the system features presented above. A small subgraph of a SKG is shown in Figure 2. Figure 3 shows the full details of the nodes in the SKG, while Figure 4 illustrates that a user with limited security credentials can view only a portion of those details.

A KD algorithm is then applied to the SKG subgraph. It works as follows:

Task: Identify new employer-employee relationships.

Background information:

- **When people receive money, 90% of the time it is from their**

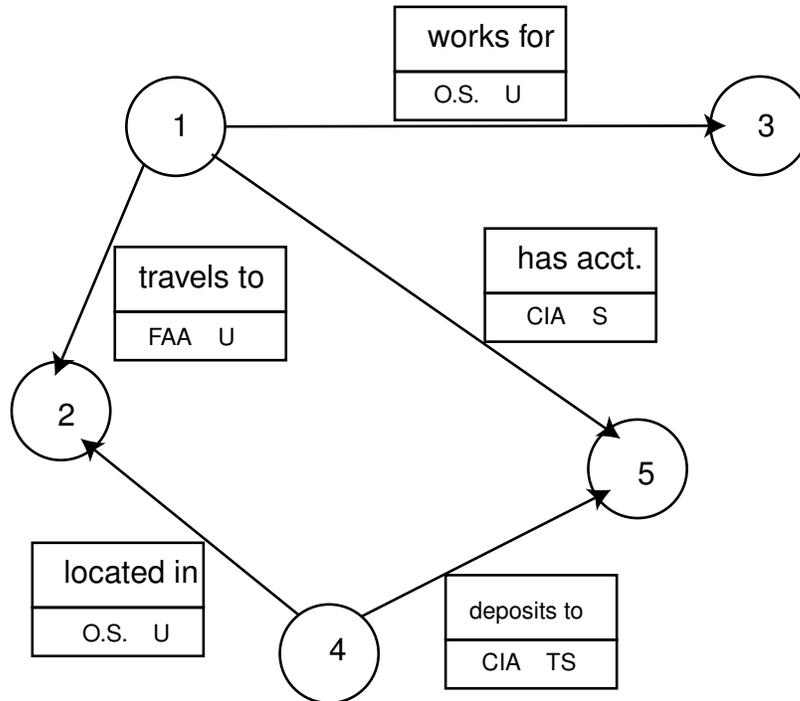


Figure 2: Sample portion of the Semantic Knowledge Graph. Nodes are labelled using the IDs from Figure 3. Link types are shown, while source and destination ID are implicit in the graph drawing. Link compartment and confidence information is not included for simplicity.

employer. 10% of the time it is from a non-employer.

- **When people travel, 60% of the time it is business-related travel. 40% of the time it is personal travel.**

It is possible that the algorithm will discover that Joe Smith works for the KGB (possibly with some likelihood or confidence level). This discovery is based on data at all three classification levels, plus background data, so it is not obvious what classification label the new link should have. Figure 5

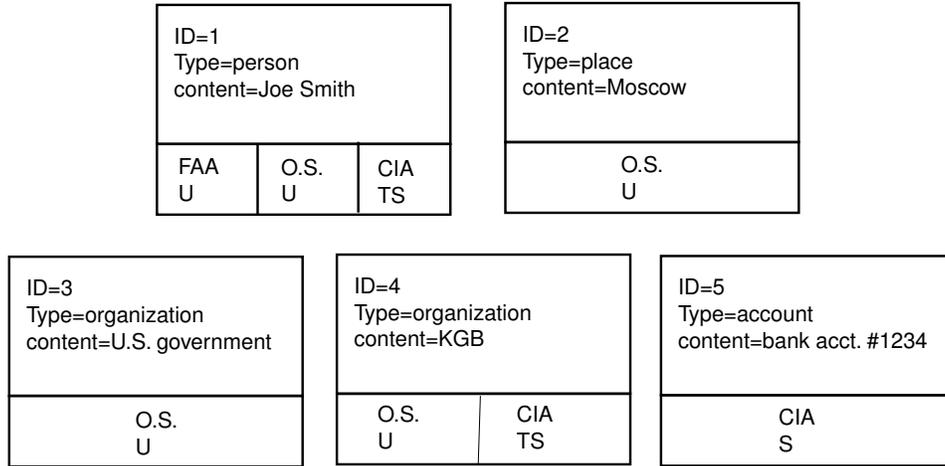


Figure 3: Example nodes published in the Semantic Knowledge Graph. OS=open sources. Compartment and confidence information is not included for simplicity.

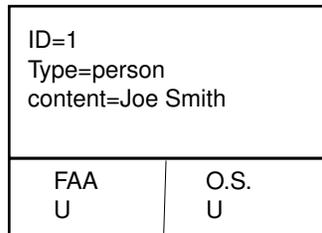


Figure 4: Anne's and Doug's view of Node ID=1. They do not have access to the fact that the CIA publishes this same data at the Top Secret classification level.

shows the newly discovered link.

To determine the appropriate label for the new link, several issues must be considered, including:

- How does the KD algorithm work?

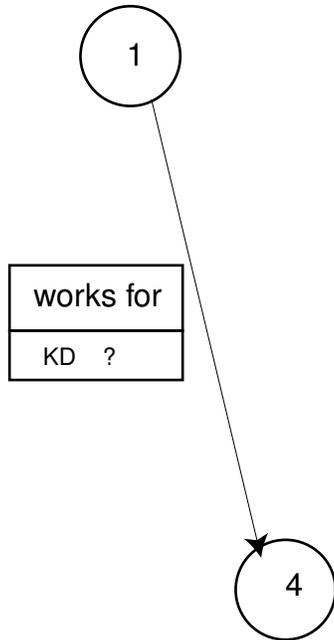


Figure 5: A KD algorithm has published this new link. (KD=knowledge discovery algorithm.) The classification label has not yet been designated. Link compartment and confidence information is not included for simplicity.

- Do users know how the KD algorithm works?
- What is the classification of the background information?
- Do users have access to background information?
- What relative impact did each SKG element have on the outcome?
- Can users correlate input data to output data?
- What level of confidentiality of input data needs to be achieved in the KM environment?

- What is being given up by achieving higher confidentiality.

The answers to all these questions are needed to appropriately label the new link. To understand some of the risks involved in labelling the new link, consider the following two examples.

Assume the new link is labelled Unclassified. Anne will then have the view of the SKG shown in Figure 6.

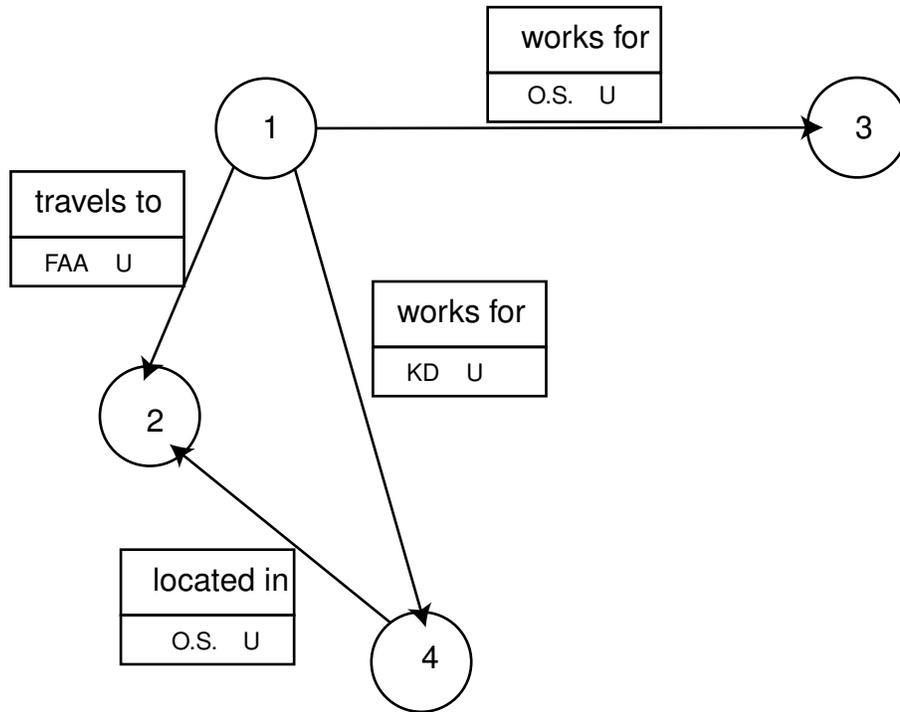


Figure 6: Anne's view of the Semantic Knowledge Graph if the new link is labelled Unclassified.

Intuitively, it seems Anne should not have access to the knowledge that Joe Smith works for the KGB. But even if a human expert classifier deems this information Unclassified, there is another issue to consider. The fact that this link was published by a KD algorithm could cause Anne to wonder

on what information this discovery was based. If she knows only what is shown in Figure 6, it is unlikely that she can accurately deduce the bank account and deposit information (Secret and Top Secret respectively). However, if she knows the KD algorithm and the background information, the likelihood of her figuring this out increases. If Anne can guess that Secret and Top Secret data with a high rate of accuracy, this could be considered a breach of confidentiality.

On the other extreme, assume that the new link is labelled Top Secret. Doug's view of the SKG is would look like Figure 7.

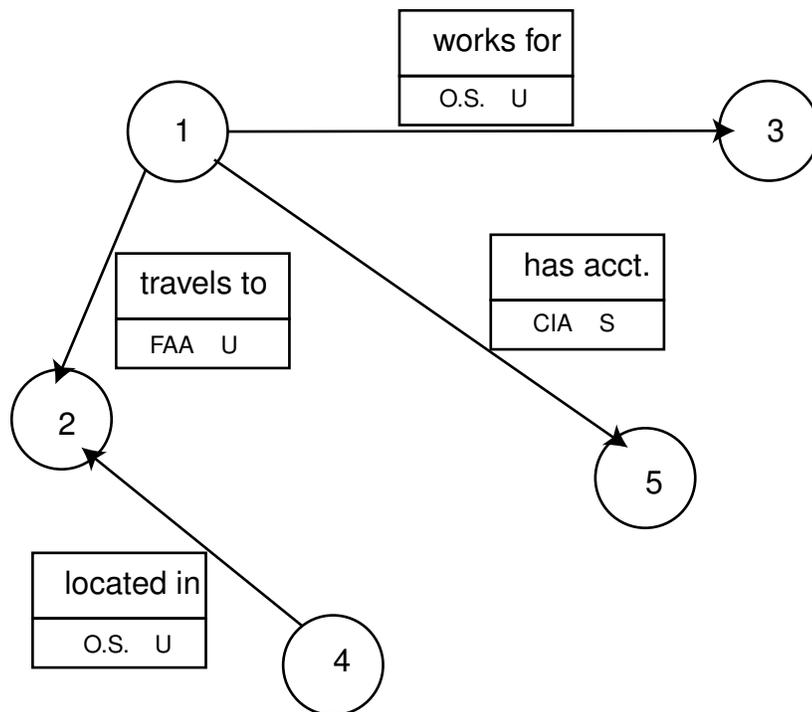


Figure 7: Doug's view of the Semantic Knowledge Graph if the new link is labelled Top Secret.

Let's supposed Doug is investigating Russian mafia activity within the

United States. Doug's view of another portion of the SKG is shown in Figure 8.

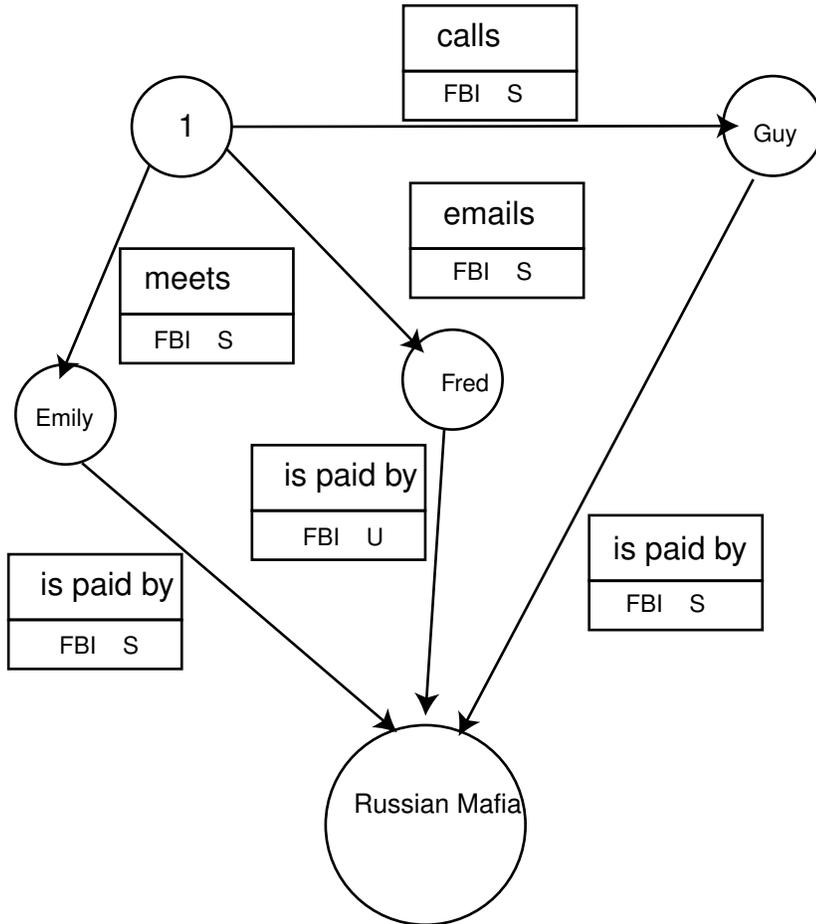


Figure 8: Another portion of the Semantic Knowledge Graph, as viewed by Doug. This portion is connected to the portion shown in Figure 2 by node ID=1.

Given Doug's information he may conclude not to suspect that Joe has ties to the Russian mafia. He may even conclude that Joe is a federal undercover agent setting up a sting operation. However, if Doug also knew

that Joe works for the KGB he would probably draw different conclusions. This single fact could be the breakthrough to help Doug solve his case—if only he had access to that link. Is withholding that information worth the cost? Would labelling the link Secret enable Doug to accurately guess the single piece of Top Secret data in the SKG?

These two examples illustrate the importance of labelling discovered knowledge in such a way as to balance the security requirements of the system with the need to share knowledge effectively. Section 3 proposes a way to accomplish this.

2 Related Work

The approach taken to achieve the goals of this work draws on four different areas: complex network characterization, KD algorithm analysis, security property analysis, and security model development. A solid understanding of these areas is fundamental, as they will be heavily utilized and extended to complete this work. The background that was required to develop the approach detailed in Section 3, is described in the next four subsections.

2.1 Characterization of Complex Networks

The first step in understanding how confidentiality policies may affect a knowledge management system is to quantitatively characterize the structure of the knowledge in the system. It is primarily the topological structure of a graph that will ultimately determine how easily it can be used to navigate knowledge. Although almost no work has been done in this area specifically, there is a substantial body of work emerging in the area of complex networks. The networks of interest range from social networks to biological networks to physical computer networks. It is likely that research in complex networks will provide insight into methods for measuring a knowledge graph's structure quantitatively. Keeping in mind the semantic graph model described in Section 1.4, it becomes obvious how work in the area of complex networks may be applicable to knowledge metrics.

2.1.1 Features of Complex Networks

There are several features of complex networks that can be measured and used to describe various characteristics of a network.

- *Degree distribution:* Degree is the number of links associated with a

node. For directed graphs, degree can be separated into in-degree (links directed into a node) and out-degree (links directed away from a node). A graph's degree distribution describes the number of nodes that have a certain degree, k , for all possible values of k . For an existing graph, obtaining the degree distribution is as simple as counting the number of nodes that have each possible degree k , and plotting k versus the number of nodes with that degree. For stochastic graphs (those that can be generated by a rule-set that is probabilistic), degree is described as the probability, $P(k)$, that a vertex is connected to k other vertices [8]. In this case degree distribution is plotted as $P(k)$ versus k . There are two distributions that are of interest in the field of complex networks. Each is described below, followed by a discussion of what these distributions indicate about a graph.

A graph with degree distribution that follows a Poisson distribution is homogenous, meaning that each node has approximately the same degree. When plotted, this distribution of k or $P(k)$ peaks at some average $\langle k \rangle$, and then decays exponentially for large values of k . More precisely,

$$P(k) = \frac{e^{-\lambda} \lambda^k}{k!}, \quad (1)$$

where n is the number of nodes, p is the probability that any pair of nodes is connected, and λ is $(n - 1)p$, or the expected node degree. Thus, the existence of nodes with very high degrees is highly unlikely [4, 41]. A graph with this degree distribution may be sparse or highly-connected depending on the value of $\langle k \rangle$. Additionally, studies have shown that the navigability of graphs with this type of distribution

is hindered as a result of both random and targeted link failures or removals [4].

A graph with degree distribution that follows a power-law distribution is inhomogeneous. In this type of distribution, k or $P(k)$ decays as a power-law. More precisely,

$$P(k) \sim k^{-\gamma} \tag{2}$$

Here, γ is known as the degree exponent. Studies of several real world networks, such as the world-wide web and paper citation networks, have revealed degree exponents between 2 and 3, though the existence of other values has not been ruled out [2, 11, 10, 16]. These graphs are often referred to as scale-free graphs, since the degree exponent is independent of the size of the graph. With this distribution, unlike a Poisson distribution, finding nodes with very large degree is statistically likely. The navigability of such a graph is not hampered by random link removals, but is extremely vulnerable to targeted link removals [2, 27].

- *Diameter*: The diameter, d , of a graph is usually the average distance between two randomly selected nodes [8], or the average shortest distance between all pairs of vertices $\langle d \rangle$ [2]. However, sometimes diameter can refer to the maximum shortest path, $\max(d)$, between all pairs of vertices [27, 38]. Depending on how the diameter measurement is to be used, $\max(d)$ may only apply to pairs of vertices for which a path exists [16]. This can also be a useful metric, but throughout this paper, diameter will refer to the average or expected distance,

$\langle d \rangle$, unless otherwise specified. Diameter may also be calculated on subgraphs, such as connected components. In general, diameter is a measure of interconnectedness (small d indicates high connectivity), though when taken alone, it may be misleading. It may be useful to examine diameter as a distribution rather than a single value, or to use it along with other graph metrics.

- *Clustering properties:* The clustering coefficient C_i of a node is defined as

$$C_i = \frac{2n_i}{k_i(k_i - 1)} \quad (3)$$

[39, 10], where n_i is the number of links between the k_i neighbors of node i . C_i can be thought of as a metric describing the extent to which node i is part of a fully connected cluster or neighborhood of nodes. This information is not provided by the degree distribution.

Several interesting observations have been made about the nature of the clustering coefficient. First, a study of several real-world networks [39] has shown that the average clustering coefficient $\langle C \rangle$ is much higher for real-world networks than is predicted by most network models. Most of the real-world networks studied were also found to follow a power-law degree distribution. In these cases, $\langle C \rangle$ was found to be independent of the number of nodes, N . This finding agrees with earlier findings [20] indicating that for deterministic scale-free networks, a node with k links has an average clustering coefficient that follows a scaling law such that

$$C(k) \sim k^{-1}. \tag{4}$$

This is not the case for networks with a Poisson degree distribution, where $C(N)$ decreases as N^{-1} with the number of nodes in the network. It has also proven not to be true for power-law networks where nodes and their relationships represent some geographical organization [39].

In general, this all implies that in scale-free networks, nodes with larger degrees have smaller clustering coefficients. This is extremely important in understanding the role of hubs to a scale-free network. Hubs are the nodes with high degree and small C_i that act as bridges between clusters. Their existence is crucial to a graph's navigability [41, 39].

Two other clustering metrics that can be useful are S , the size of the largest cluster (here, a cluster refers to an isolated subgraph), and $\langle s \rangle$, the average size of all isolated clusters (i.e. all clusters other than the largest). Albert et al. [4] use these to characterize the fragmentation process that occurs when a graph has a fraction of its nodes removed. This can be used to illustrate the robustness or fragility of a graph's topology.

- *Connected components:* A graph's macroscopic structure may prove useful in understanding its connectivity. One study, examining the world-wide web as a directed graph, identified six important components or pieces of the graph. First is the giant core, or Strongly Connected Component (SCC). Any node in the SCC is reachable from any other node in the SCC. Then there are the IN and OUT components.

All nodes in OUT can be reached from the SCC while all nodes in the SCC can be reached from IN. There are also Tendrils. Tendrils, or subgraphs, hanging off the IN component can be reached from IN, but are effectively dead ends, since there is no return path to any other component. The OUT component can be reached from Tendrils hanging off of it, but there is no return path to these Tendrils. Tubes are Tendrils that hang off of IN, but then feed into OUT. Finally, there are disconnected components which do not reach the rest of the graph and are not reachable from it [16]. Figure 9 illustrates these graph components.

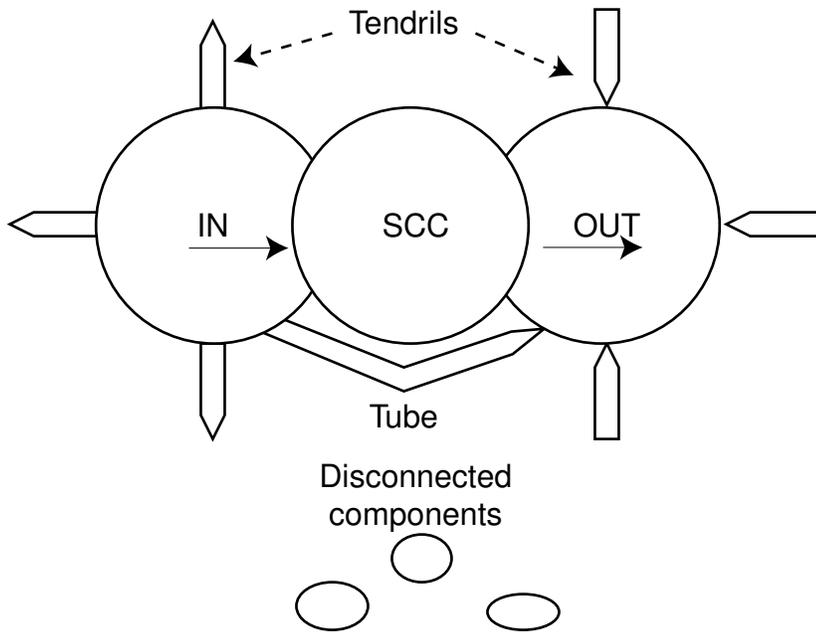


Figure 9: Connected component structure of a directed graph. Adapted from [16].

Schwartz et al. [40] also refers to the Giant Weakly Connected Compo-

ment (GWCC) in which each node is reachable from every other node if the links are treated as bidirectional.

Identifying these types of components in a graph could provide insight into its robustness and navigability. Other graph metrics, such as diameter, clustering coefficient, degree distribution, and relative size, could all be applied to specific graph components. For example, Schwartz et al. [40] examined each of these components separately to understand the percolation critical exponents and their dependence on correlated degree exponents in directed scale-free networks. Likewise, findings in Broder et al. [16] based on the macro-component structure of the world-wide web revealed many statistics about web path traversals that could not be easily calculated by brute force. Their results also showed the resilience of web searches to the removal of a significant number of nodes.

- *Spectral properties:* The graph metrics discussed thus far primarily describe structural features of a graph. Farkas et al. [22] suggest that in addition to examining structural properties of graphs, examining spectral properties can be a useful tool. Specifically, they analyze spectral properties to classify real graphs into the network models described below in 2.1.2.

They represent a graph by its adjacency matrix and take the set of eigenvalues, λ , of this matrix. λ is the spectrum of the graph. From this, they obtain the spectral density, $\rho(\lambda)$, of the graph as,

$$\rho(\lambda) := \frac{1}{N} \sum_{j=1}^N \delta(\lambda - \lambda_j), \quad (5)$$

which is the density of the eigenvalues. The spectral density provides useful information about the topological characteristics of a graph. For example, D_k , the number of directed loops of length k , can be calculated. Additionally, spectral density can provide clustering information that can be used to identify scale-free and small-world graphs. The specific mathematical methods that Farkas et al. use require that only a small subgraph be measured, which could be beneficial in many cases. However, if their methods are intended for graphs that are sufficiently sparse, so it is not clear that they would be applicable in all cases.

2.1.2 Stochastic Complex Network Models

Several network models have been proposed, and it is important to understand each of them, as they each have different implications for network characteristics of interest. Of primary interest are stochastic network models, since their topological evolutions are governed by probabilistic rules. Each of the stochastic models presented below is described in terms of the rules governing network generation or growth. Any real-world networks that fit the models are discussed. Finally, the model's effects on the network features described above will be presented.

- *The Classical Random Network Model:* The Classical Random Network model was proposed by Erdos and Renyi [21], and represented a major breakthrough in graph theory. This model constructs a graph by first defining N nodes and then connecting each pair with probability p . This results in a graph with $pN(N - 1)/2$ links distributed randomly. The degree distribution peaks at $\langle k \rangle$, and decays exponen-

tially for $k \gg \langle k \rangle$. One notable feature of this model is that the graph is static; the size is predetermined and neither nodes nor links are added after initial construction.

Although this model was used for nearly fifty years to represent networks encountered in nature, more current research has not been able to identify any real-world networks that fit this model better than one of the others described below [10, 4]. In terms of the network features described above, graphs in this class have a Poisson degree distribution, a diameter that grows monotonically with N , clustering coefficients that follow a Poisson distribution, and $C(N) \sim N^{-1}$. Additionally, as a fraction of nodes, f , are removed from the graph, S and $\langle s \rangle$ show interesting behavior. For small f , $\langle s \rangle \simeq 1$ (i.e. only single nodes break off from the main cluster) and S decreases. However, at a threshold value of f , $S \simeq 0$ and $\langle s \rangle$ peaks (i.e. the main cluster breaks apart into many isolated clusters). For high f , even these isolated clusters break apart and $S \simeq 0$ and $\langle s \rangle \rightarrow 1$ [4]. For a sufficiently large p , a graph in this class would consist primarily of a large GWCC, evolving into an SCC as p continues to increase. For small p , the tendrils would be larger than the GWCC, and almost no SCC would be present. An analysis of the spectral density by Farkas et al. [22], shows that the topology is tree-like, with some shortcuts. Their conclusion is based on the fact that $\rho(\lambda)$ is symmetric, which indicates that the number of isolated clusters grows linearly with N and all loops with an odd length disappear. So, as $N \rightarrow \infty$ the topology evolves into a tree where a loop involves traversing any link exactly twice, thus the emergence of loops with only even length.

- *The Watts-Strogatz Network Model:* Watts and Strogatz presented another stochastic network model to represent real-world networks that exhibit high clustering and regular topology with some random disorder [42]. In other words, they wanted to model networks that exhibited the small-world phenomenon, the phenomenon that even in very large networks, the distance between two randomly selected nodes is quite small. They originally named their model the Small-World model. However, since then, other models have been proposed that also model the small-world phenomenon, so this model is now commonly referred to as the Watts-Strogatz (WS) Model. This model starts with a ring of N nodes, with each node being connected to its k nearest neighbors. This regular graph is then rewired by selecting a link with probability p , and reconnecting it to another randomly selected node [42].

As originally intended, the WS Model accurately represents the small-world phenomenon and helped to fill the gap between network models that were completely regular and the Classical Random Network Model. However, many of the real-world networks that it is was intended to model are more accurately modelled by one of the scale-free network models discussed below. Like the Classical Random Network Model, the WS Model generates a graph with a Poisson degree distribution, with an even steeper exponential drop off. However, the WS Model graph's diameter does not change monotonically with N . In fact, the diameter is quite small compared to N and is highly resistant to changes in N . As would be expected, the clustering coefficient for any node is quite high. After all, small diameter and large C_i are the defining features of the small world phenomenon. The connected com-

ponent structure shows a very large SCC and almost no Tendrils or disconnected components. The GWCC is similar in size to the SCC. The spectral density, $\rho(\lambda)$, shows a high number of triangles, even when rewiring is increased greatly [22]. This is indicative of a high average clustering coefficient.

- *The Ultrametric Network Model:* In an effort to formulate a more realistic model where node degree was more widely distributed, Hogg proposed the Ultrametric Model [25]. This model logically groups nodes of a graph into an unbalanced binary tree (where nodes are leaves only) and then calculates the ultrametric distance, u , between each pair of nodes, which is the distance up the tree to the first common ancestor. Each pair of nodes is linked with probability p^u . This model results in a broader degree distribution than the other models discussed so far and more closely resembles the degree distribution seen in real-world networks since nodes of high and low degree are present in statistically significant quantities. However, it does not model the small-world phenomenon so commonly seen in real-world networks since the average clustering coefficient is low [41].
- *The BA Network Model:* Barabasi and Albert recently proposed yet another complex network model, the BA Model, in an attempt to address shortcomings in other models. In particular they hoped to more accurately capture the degree distributions seen in nature and account for those distributions in their model. Two important aspects of the BA Model are *growth* and *preferential attachment*. Barabasi and Albert recognized that real-world networks are not static entities; they grow and evolve over time. Thus, the BA Model begins with a small

number of initial nodes, m_o . New nodes are added individually and linked to $m \leq m_o$ nodes already present in the graph. To determine the which m nodes to link to, the BA Model introduces the concept of preferential attachment, which means that new nodes are more likely to link to existing nodes with high degree i.e. the rich-get-richer phenomenon. Mathematically, the probability, Π , that a new node will be connected to node i is

$$\Pi(k_i) = \frac{k_i}{\sum_j k_j}, \quad (6)$$

where k_i is the degree of node i , and j is the total number of nodes in the graph when the new node is added [2, 9, 26, 10, 8, 7].

Some tangential studies offer modifications and extensions to the basic BA Model. One notable feature missing from all the models discussed above is the concept of weighted links. Yook et al. [44] address this and supplement the BA Model by exploring ways to generate weights, augment preferential attachment to account for weights, and analyze the effect on the resulting network topologies. Schwartz et al. [40] studied the effects of in-degree and out-degree correlation on navigability and the macro-structure of connected components. Their results indicate that it is important to differentiate between correlated and uncorrelated degrees in the BA Model. Farkas et al. [22] further define preferential attachment by describing two possible forms: new node attachment and internal attachment. They recognize that links in real-world networks are added not only when new nodes are introduced, but may appear between two older nodes. Their data suggests that the probability of internal attachment, $\Pi(k_1, k_2)$, increases

as either k_1 or k_2 increases and that it is linear in k_1k_2 . The basic BA Model describes only new node attachment probability, which is dependent only on the degree of the old node.

Several real-world networks studied fit the BA Model very closely. The world-wide web topology has been studied extensively and has been found to fit the BA model whether a small sample of the web is used or a very large sample [2, 16, 9, 8]. Other social networks that exhibit a similar structure include author collaboration networks, actor collaboration networks, and paper citation networks [26]. Other networks, such as biochemical reaction networks, food chain networks, or physical computer networks may fit the BA Model as well.

A graph that fits the BA model will have a power-law degree distribution, rather than the exponential distribution predicted by most other models. This distribution causes the diameter to be quite small and only logarithmically dependent on N . Such a network can be thought of as a hub-based network, or a network whose connectivity is highly dependent on the nodes with very high degree. Even so, the clustering coefficient does not follow the scaling law (eq.4), rather, $C(N) \sim N^{-0.75}$. This leads to questions about whether or not the small world phenomenon is adequately represented by the BA Model [39]. Broder et al. [16], in their study of the macro-structure of the world-wide web raise some interesting related points. Their data on the navigability-related features of the connected components indicate that the web's connectivity is not as dependent on hubs as was previously thought. Also, their study indicates that the web's macro-structure is more complex than can be explained by the small-

world phenomenon and that the Zipf distribution may more accurately describe the degree distribution than the power-law distribution, especially for in-degree. The Zipf distribution is an inverse polynomial function of *ranks* of degree rather than *magnitude* of degree. The BA Model is still new and under development. Phenomena that are not explained well by this model are still being uncovered and discussed, so it is not yet clear how well the BA Model really does represent real-world networks.

- *The Hierarchical Network Model:* The Hierarchical Model is based on the BA Model, but seeks to model a network where C is independent of N , following the scaling law (eq.4) and thus revealing a hierarchical topology that is not described by other models but that appears in real-world networks [39]. The Hierarchical Model modifies the BA Model by restricting the preferential attachment mechanism such that each highly connected cluster has a strict upper limit clustering coefficient and average degree. The model begins with a small cluster of n densely connected nodes. In the next step, $n - 1$ copies of the cluster are generated. A fraction, p , of the new nodes are connected independently to nodes in the original cluster following preferential attachment. That is, a selected node will connect to node i of the original cluster with probability $\Pi = k_i / \sum_j k_j$, where k_i is the degree of node i and the summation is over the j nodes of the original cluster. The next step generates $n - 1$ copies of the new n^2 size cluster, but only p^2 fraction of the new nodes are selected for connection to the n^2 size cluster. This model also has a deterministic version, which is described in [11, 10]. This results in a network that obeys the scaling law (eq.4) while still

following a power law degree distribution. Additionally, nodes in this model always follow the law that higher degree nodes have smaller clustering coefficients. Several real-world networks have been shown to have characteristics modelled by the Hierarchical Model, including metabolic and protein interaction networks, as well as the world-wide web and actor collaboration networks. However, networks that represent a geographical organization do not fit this model well [10]. Ravasz et al. do note, however, that this is not the only model that produces these results. In particular, the model of Klemm and Eguiluz, as described in [39], propose keeping p constant for each iteration, but deactivating nodes at each time step with probability $P_i \sim k^{-1}$. In effect, this ages off less connected nodes and causes a densely connected central core to emerge. Although this model bypasses preferential attachment, the resulting network topology is equivalent. Preferential attachment is one of the most controversial aspects of the BA Model, as many do not feel it accurately describes the motivations behind the evolution of real-world networks. This Klemm-Eguiluz modification may offer a more realistic picture.

2.2 Knowledge Discovery and Creation

Most KD algorithms take as input some prior background information in addition to the knowledge currently represented by the knowledge graph. They then process this combination of information to produce new knowledge as output. This knowledge may be presented to a user, fed back into the graph in some predetermined fashion, or utilized by other graph processing algorithms.

There are many different KD algorithms, and new ones are being developed all the time. These algorithms can be loosely categorized by their primary task [23]. Though the categories described by Fayyad et al. are not necessarily exhaustive and are not the only way to classify KD algorithms, they do provide a convenient way to organize the algorithms so that general statements can be made about their characteristics.

Understanding the processing that is performed by advanced KD algorithms is an important aspect of this work. It will be important to know to what extent the *input* to the algorithm can be deduced from the *output*. This information will have a significant impact on the security labelling requirements imposed on the output. Section 3.2 will investigate approaches for analyzing this aspect of KD algorithms.

Each KD task category below is briefly described, and examples are given that elucidate how this task might be applied in a government KM system. These specific examples are for illustration only. They do not represent any actual data sets; the level of knowledge granularity is higher than would be expected in reality and the specific KD task achieved is trivialized to simplify the explanation of each task.

2.2.1 Classification

KD algorithms that learn functions that map data into predefined classes are known as classification algorithms. It is important to differentiate this from an algorithm that simply assigns classes to objects based on predefined rules. Such an algorithm is not performing a learning or discovery task, whereas a classification algorithm is learning the rule set for assigning classes to objects.

Task: Learn how to classify foreign military activities into the classes **routine**, **cause for close observation**, **cause for diplomatic action**, and **cause for immediate military response**.

Background information: Examples of foreign military activities that fall into each predefined class.

Example: Consider the following military activities have been observed and recorded in a graph:

- **5 new encampments have been established along a contentious border.**
- **Existing encampments have increased in activity level by 15%.**
- **No suspicious information has been reported from undercover agents.**
- **Diplomatic relations are good.**
- **A large government payment was made to a known supplier of missiles.**

Given a sufficient set of background information, the algorithm might classify this situation as **cause for close observation**. This assessment is new knowledge discovered by the system.

Not just activities and trends, but objects of interest can also be classi-

fied. For example, based on content, satellite images could be classified as military installations, civilian towns, factories, or naval fleets.

2.2.2 Regression

KD algorithms that learn functions that map data to real-valued variables are known as regression algorithms.

Task: To predict the quantity of illegal drug production of some geographical region known to produce drugs.

Background information: Example data sets where the drug production quantity is known and other facts about the regions are described.

Example: Consider the following data is available on a drug-producing region:

- **Region is 1127 acres.**
- **Climate is tropical.**
- **Terrain is mountainous.**
- **There are 3 buildings that serve as warehouses.**
- **Each day 5 trucks on average pick up loads in this area.**
- **There are 2 churches in this region.**

Based on this data and the background information, the regression algorithm might predict that 40 tons of marijuana can be produced weekly by this region. This fact is new knowledge discovered by the system.

Another way to use regression is to estimate a probability. For example, based on remotely sensed data, a regression algorithm could predict the probability that a specific military target was hit by a missile. Again, this class of KD algorithms should not be confused with an algorithm that simply computes an output value based on input variable values. A KD algorithm has to learn what input variables to use and what form the function should take. The final variable above, the number of churches in a region, is unlikely to be highly correlated to drug production. The KD algorithm must make this determination based on the background information.

2.2.3 Clustering

KD algorithms that identify categories based on data are known as clustering algorithms. Clusters may be mutually exclusive or overlapping and may or may not be exhaustive. Clustering algorithms do not assign meaning to the clusters they identify; that task is left to a human user or other algorithm.

Task: Identify the cluster or clusters exhibited in the data set of Figure 10.

Given the data set in Figure 10, which plots bank account balance versus number of cell phone calls, a clustering algorithm might identify the four clusters shown and define them mathematically. These mathematical functions are new knowledge. Given these clusters, a human expert may then assign meaning to clusters (and thus data points, as shown in Figure 10).

In this example, the expert determined that two clusters probably represent drug dealers while the other two clusters do not.

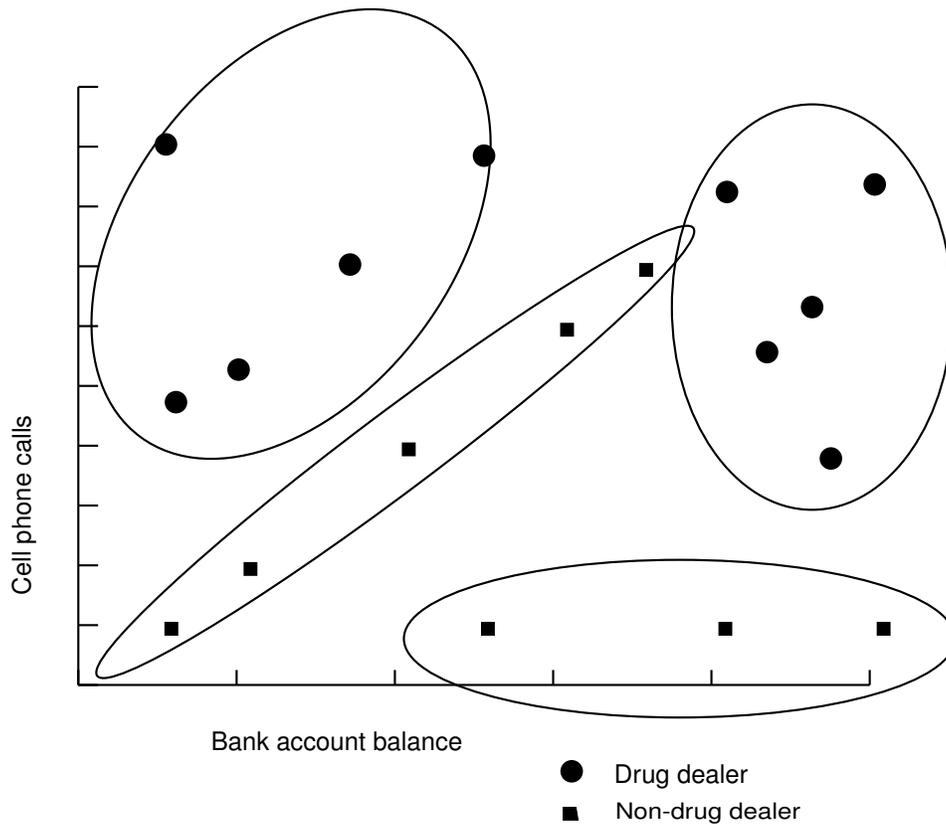


Figure 10: Clustering of data into four sets. The algorithm identified the clusters, but a human expert assigned meaning to the clusters.

It is not likely that this type of discovered knowledge would be fed back into the graph. However, it could be used to support additional graph processing, be used by an expert system or decision support system, or be used in advanced query processing. Much more advanced clustering algorithms would define clusters based on a multitude of variables and could work with both continuous and discrete variables.

2.2.4 Summarization

Summarization algorithms utilize a variety of methods to provide a compact description of data. Advanced summary algorithms might derive summary rules or discover functional relationships between objects.

Task: Identify companies that may be ordering components that can be used to manufacture weapons.

Background information: Data describing which components can be used to manufacture weapons.

Example: Consider the following data available in the graph:

- **AllChem sells chemical A and chemical Z.**
- **ACME Electrical sells electrical part X.**
- **Willie's Widgets sells mechanical part Q.**
- **Sam's Fertilizer Company, Inc. orders from All Chem and Willie's Widgets.**

Supposing the background information revealed that chemical Z is commonly used in chemical weapons, then the algorithm would report that Sam's Fertilizer Company, Inc. may be purchasing components that can be used to manufacture weapons. Additional information such as how much of AllChem's sales are chemical Z, or what specific chemical Sam's is purchasing from AllChem may alter the results, or provide some probability that Sam's is actually purchasing chemical Z.

More advanced summarization algorithms may perform more indirect inferencing. For example, background information may only include examples of other companies purchasing weapons components, rather than background information that explicitly states which components can be used to produce weapons. In that case, the algorithm must determine for itself which components may be used to manufacture weapons. A simpler summarization algorithm that lends itself well to query processing for graph-modelling information is one that can add shortcut links. For example, if Mr. X works at Company A, and Mr. Z owns Company A, then an algorithm could create a new link from Mr. Z to Mr. X labelled “is boss of”.

2.2.5 Dependency Modelling

A KD algorithm that finds a model that describes dependencies between variables is known as a dependency modelling algorithm. Dependency models may just describe the existence of a dependency, or they may assign a weight to a dependency indicating its strength or probability.

Task: Develop a model that describes the travel behavior of someone who is likely to blow up an airplane.

Background information: Data describing the travel behavior of several individuals, distinguished by whether or not they have blown up airplanes.

After processing the background information, the algorithm may determine that the following variables describe the behavior of someone who is likely

to blow up an airplane:

- $18 \leq \text{age} \leq 30$
- **checked baggage at airport = no**
- **purchased one-way ticket = yes**
- **gender = male**

The algorithms may have determined that many other available variables were not meaningful in modelling the behavior of interest, such as height, airline, and date travelling. This model is not based on information in the graph and does not feed back into the graph itself. However, the resulting model may be used by other algorithms to process the graph.

2.2.6 Change and Deviation Detection

A KD algorithm that discovers significant changes from previous or normal values performs the task of change or deviation detection.

Task: Report significant changes in naval port traffic.

Background information: Possibly none, though data describing how much of a change is considered significant could be utilized.

Example: Consider the following graph data at time t_0 :

- **Port P has 27 ships docked.**
- **10 ships are en route to Port P.**

- **4 ships are departing Port P.**

Later, at time t_n , the graph data indicates:

- **Port P has 116 ships docked.**
- **56 ships are en route to Port P.**
- **0 ships are departing Port P.**

This would likely cause the algorithm to report that significant changes in port traffic at Port P are occurring.

2.3 Security Properties

Any confidentiality policy has a specific set of security properties that describe the extent or type of security provided by the policy. This section describes several of the most important properties.

2.3.1 Nondeducibility

To understand *nondeducibility*, think of the system of interest as a state machine that has both high and low inputs and outputs. In the context of this work, a knowledge discovery algorithm is the system of interest, and the input data and output knowledge are the items of concern. A confidentiality policy for the KD algorithm will state how to apply confidentiality labels to the output knowledge. Here, only high and low levels are considered, but the extension to multiple levels is trivial. A policy is deducibly secure if a set of low inputs and low outputs cannot be used to deduce information about high inputs or high outputs [14]. There are other ways of describing nondeducibility. Bishop [14] and Anderson [6] each define nondeducibility

as a property of a system in which for every low trace (set of low inputs and low outputs) a low user cannot distinguish which high trace, from the set of all possible high traces, may have accompanied it. However, the existence of a high trace may be evident. Nondeducibility is regarded as a rather weak security property because a system is considered deducibly secure even if high information can be deduced, but with less than 100% accuracy.

An important quality of security properties is *composability* since it indicates whether a sequence of systems with a specific security property can be combined to form a new system that still possesses that property. Systems which have nondeducibility as their strongest security property are not composable [14, 30].

2.3.2 Noninterference

There are four variations on noninterference discussed in this section: deterministic noninterference, strong noninterference, generalized noninterference, and restrictiveness. The first two are applicable only to deterministic systems (stateless systems). The latter two are applicable to nondeterministic systems which are stateful. Generally speaking, noninterference is a property that extends and strengthens nondeducibility. Any system that is noninterference-secure is deducibly secure, but the opposite is not necessarily true [30].

Deterministic noninterference is a property of systems that prevents high inputs from affecting low outputs whatsoever. *Strong noninterference* provides additional security by not allowing high outputs to occur unless they are triggered by high inputs. Requiring strong noninterference may be too restrictive since it disallows some systems that are obviously secure [14].

Both deterministic noninterference and strong noninterference are considered composable. However, several rather restrictive assumptions about the system must be made for this to be the case [30].

To address a system that is nondeterministic, or has asynchronous inputs and commands, McCullough introduced *generalized noninterference* [30]. This property states that altering a high input in a sequence of inputs produces low outputs that could have been produced given the original sequence of inputs. This property is only composable if no feedback is allowed between systems.

The *restrictiveness* property, also introduced by McCullough [30], extends and strengthens generalized noninterference. Restrictiveness is basically the nondeterministic form of strong noninterference. Consider an input sequence of both high and low inputs and a system that uses them to produce both high and low outputs. If the system is restrictive, then any changes to the high inputs, including their order, does not affect the low state of the system, in addition to not affecting the immediate low outputs. In other words, the high inputs can not have an effect on any state information that will affect future low outputs. Also, a restrictive system does not allow the low level state to be affected by high level outputs. Differentiating between the states of a system, and thus the future outputs, and the immediate outputs alone results in a security property that is composable [14].

2.3.3 Inference Controls

Logical inference is the ability to infer high data from low data. What distinguishes this from deducibility is that logical inference is not confined

to inferring high input data from low output data of a specific system or algorithm. It is more general and applies to high and low data items that are not necessarily functionally related.

One type of inference capability arises due to relationships between pieces of data, rather than from the data pieces themselves. For example, a person's name may not be confidential and a number representing a salary may not be either. However, the relationship that correlates that person to that salary may be confidential.

Several methods have been proposed for constraining relational databases so that this type of inference is not possible [29, 24, 37]. However, when the knowledge representation model proposed in this work is used, this type of inference is not an issue, since links represent semantic relationships between nodes, and links have their own classification labels.

Another type of inference capability arises when a collection of low data allows one to infer information that is high. Census data provides an example of this type of inference problem. The primary solution to this problem involves suppressing some of the low information—just enough so that the high information can no longer be inferred [6]. There is a fine line between this type of inference and aggregation, another security problem discussed in Section 2.3.4.

This work will only be concerned with logical inference capabilities resulting from knowledge discovery processes that occur within the system. Logical inference that is made possible by a user's domain knowledge or any source of information external to the system will not be considered. Given this constraint, logical inference is essentially the same as deducibility for the purposes of this work. Since nondeducibility and noninterference are more formal and precisely defined security properties, it is likely that they will be

used to describe security models developed as part of this work. However, some of the models developed to address logical inference may be useful.

2.3.4 Aggregation Constraints

Another notable confidentiality problem is the aggregation problem. This problem arises when two or more low data items, when taken together, have a high classification. This differs from the logical inference problem in that aggregation does not imply that the low data allows one to infer high data, but rather, that the collection of low data is itself classified high [29]. One example of the aggregation problem is the NSA phonebook. Though no single phone number is high, the collection is. This differs subtly from the census data problem in that the census data collection is not high, but the personal information that be inferred from it is. This distinction is not always easy to make and two problems often overlap and are even addressed in similar ways [6].

Several systems and security models have been developed that claim to address the aggregation problem [36]. However, the solutions are impractical for several reasons. Meadows' solution assumes that information can easily be divided into datasets such that the classification of any possible combination of datasets is known in advance. For a dynamically evolving KM environment that contains knowledge from a variety of domains, it is unlikely that devising such datasets and classification rules will be possible. Furthermore, a history must be kept that tracks a user's or system's accesses to prevent illegal combinations of accesses over time. Having to store and validate against these history logs could require an extensive amount of storage and also hinder query performance considerably. The fact that a

user or system may have different clearance levels or roles in different locations or situations makes maintaining and utilizing this history information even more complex. Finally, even a perfect solution can only extend to the boundary of the system. The system cannot detect or prevent a user from obtaining low data through other legal means. In consideration of these issues, the security policies developed as part of this work will not attempt to incorporate aggregation constraints or solve the aggregation problem.

2.4 Security Models

A number of security models have been proposed that address confidentiality in the context of ML/LS systems. This section presents some of the more influential models. Although many of these models address both integrity and confidentiality, the focus here will be on the portions of the models that address confidentiality.

2.4.1 The Bell-LaPadula Model

The Bell-LaPadula (BLP) Model [13] [12] was originally designed to address confidentiality and information flow concerns in military information systems, particularly operating systems. BLP addresses multi-level security and its main goal is to prevent a subject (user or process) from reading objects at a classification level higher than the subject's clearance level. Both mandatory access controls (MAC) and discretionary access controls (DAC) are utilized. Two key features of BLP include the *Simple Security Condition* and the **-Property*. The Simple Security Condition stipulates that no process may read data at a higher level than that process's clearance level. This is also known as no-read-up (NRU). The **-Property* states that no process

may write data to a lower level than that process's clearance level. This is known as no-write-down (NWD). The *-Property is very limiting, but a loophole is provided by allowing processes to work at either their maximum security level or a current security level, which may be lower. A complete and formal description of BLP is provided by Bishop in [14].

BLP has been criticized extensively for a variety of weaknesses, mostly related to theoretical aspects of the model and the basic assumptions it makes about security models in general. See [31], [32], [14], and [6] for discussions about BLP shortcomings and proposed solutions.

2.4.2 The Chinese Wall Model

Brewer and Nash [15] formulated the Chinese Wall (CW) Model to address conflict of interest issues in the financial and legal sectors. Whereas BLP is ideally suited for multi-level secure systems, CW is targeted to multi-lateral secure systems. The CW Model consists of objects (data items) that are grouped into classes. Classes are then grouped into conflict of interest classes (COI). If classes are in the same COI then they are in competition or may have a conflict of interest. A user or process can only access a data item if another data item that maps to the same COI class has not been previously accessed by that user. Thus, the model has a temporal aspect. Two key features of CW are the *CW-Simple Security Condition* and the *CW-* Property*. The first of these stipulates that a subject can only read an object from class C' and COI class COI' if 1)the subject has already read objects that belong to C' , 2)all of the objects read previously by the subject belong to COI classes other than COI' , or 3) the object has been sanitized. The *CW-* Property* states that a subject may only write an object if 1)

the *CW*-* *Simple Security Condition* would permit the subject to read that object or 2) All unsanitized objects that can be read by the subject belong to the same class as the object to be written [14] [6]. Extensions to the CW model allow it to address both multi-level and multi-lateral security requirements [36].

2.4.3 A Model for Medical Information Systems

The need to protect patients' privacy while providing medical practitioners with the information needed to diagnose and treat patients effectively has prompted the the development of a security model targeted specifically to medical information systems (MIS). Anderson's British Medical Association (BMA) Model is one of the more influential security models for MIS [14] [5] [6]. The model consists of nine principles:

- Access control lists indicate which groups or individuals have read access, write access, or both.
- Record opening controls prevent unauthorized clinicians from making referrals. In other words, the referring clinician must be on the access control list as must be the new clinician to whom the patient was referred.
- Responsibility for the access control list of a given patient lies with a single clinician.
- Patients are notified of all changes to the access control list, and their consent is required (except in emergencies).
- No data is deleted so as to facilitate auditing.

- All accesses are logged to assist auditing.
- Information flow controls prevent data from one record being written to another record unless the second record's access control list is contained in the original record's access control list.
- Measures are in place to prevent large amounts of data being made accessible to a single person or group.
- A trusted computing base shall be used to enforce the other principles and will be evaluated by an independent authority.

2.4.4 The Lattice Model

The Lattice Model was formalized by Denning [17] and is very similar to BLP, but it also addresses multi-lateral security issues. A lattice of security labels is shown in Figure 11. The features of BLP can then be applied by following the arrows down the lattice. The Simple Security Condition now means NRU and also no-read-across. In other words, only data with a label set that can be reached by following the arrows can be read. The *-Property now means NWD and also no-write-across. Implementing this model can be difficult even though it is simple in theory. The possible combinations of labels explodes in size, making enforcing a lattice-based policy very expensive.

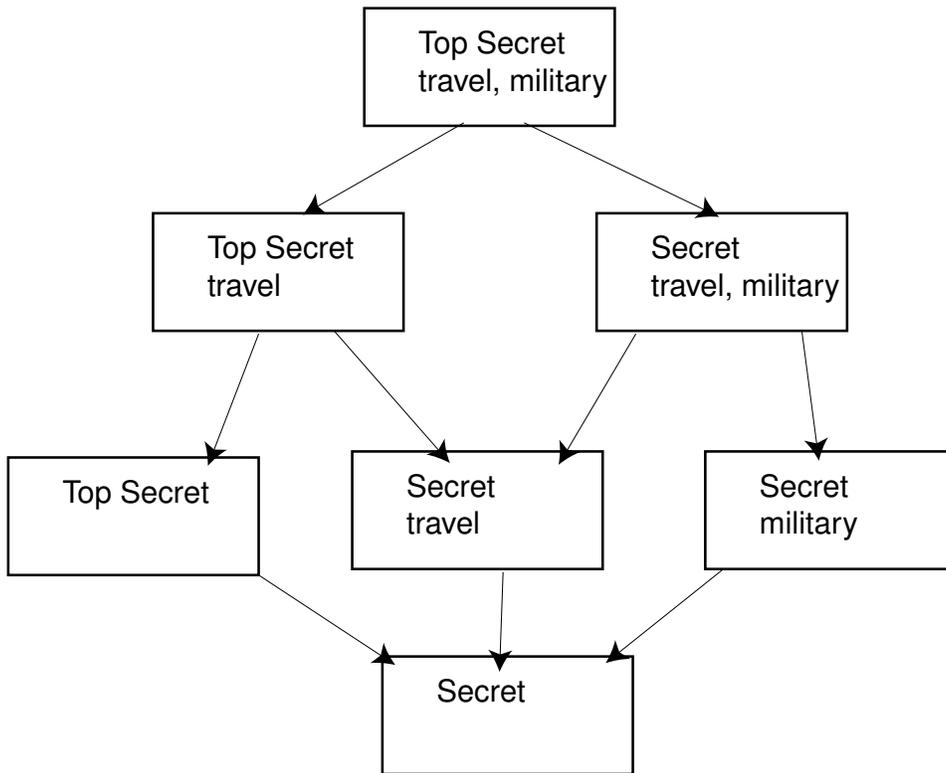


Figure 11: A lattice of security labels. Both classification levels and compartments are used.

3 Approach

This section describes the four high level tasks that compose this work: developing knowledge metrics, analyzing KD algorithms, developing security policies, and developing a cost-benefit analysis technique. The general approach that will be taken to accomplish these four tasks is described. Each task is then recast as a very specific step or set of steps that can be objectively evaluated for completion at the conclusion of this work.

3.1 Develop and Apply Knowledge Metrics

The graph metrics described in Section 2.1, such as degree distribution, diameter, clustering information, component structures, etc., can all be applied to a SKG. However, before utilizing these metrics in the context on a knowledge network, two extensions to these metrics are needed.

One extension is to differentiate between the varying semantic meanings of links. Most of the work done in the area of complex networks assumes that all links are equivalent in meaning. This is also true for the specific real-world networks discussed throughout the complex networks literature, such as the various collaboration networks, where all the links meant “collaborated with.” A large and complex KM system is not so limited in scope, so there will be many different meanings among links. This extension may be as simple as enumerating each different link type to prevent links of different types from being collapsed into a single link, even if they connect the same two nodes.

The second extension involves assigning weights to links. Links may have different weights for a variety of reasons. Some links may be more useful than others for semantically traversing through the knowledge. Links may also

have different quality ratings or confidence ratings if their existence has an element of uncertainty. The work of Yook et al. [44] could be applied directly or adapted, depending on the other knowledge network characteristics that are discovered. (Their work applies directly to only two specific network models.) They present a method for assigning weights to links as a network evolves and extend the concept of preferential attachment to account for weighted links.

Two methods for assigning link weights in a knowledge graph will be considered. Usage statistics could prove helpful in understanding which types of links actual users prefer to traverse. Such usage statistics might include the number of times a link is traversed or the number of times a node represents a source or destination (rather than an intermediate step). This data could be very difficult to gather and at best may be based on the preferences of a small sample of users with a limited domain focus. However, the general method of gathering usage statistics and utilizing that data should be general enough to prove beneficial in describing one way to assign link weights. Another method is to base link weights on domain expertise that is embedded in a knowledge graph's ontology. Given the knowledge representation method chosen for this study, the ontology could be supplemented such that each link type definition includes a weight that rates its potential usefulness. If links are added based on uncertain knowledge, by a human user, by a data collection system, or by a knowledge discovery engine, then a weight could be assigned to a link to indicate just how certain that knowledge is. It is likely that a combination of both these factors will play a role in assigning link weights.

The first step in this work is to extend the metrics in 2.1 to account for weighted with diverse semantic meanings.

One example of how this could be done applies to the diameter of a graph. Consider links with weights assigned ranging from 1 to 10, where 1 indicates a highly useful link type and 10 indicates a link type that is so common that it may not be very useful. The diameter calculation would then incorporate link weights, rather than mere link existence. Thus, a graph with more links of useful link types would have a smaller diameter than a graph with less useful link types.

Another possible extension applies to the degree distribution and is very straightforward. Since links have meaning, there may be several links connecting the same two nodes, if those nodes have multiple relationships with each other. The method of calculating degree distribution does not need to be modified, but the way it is interpreted does. Semantic links may cause degrees to be much higher than in graphs where only a single link is allowed between two nodes. This fact needs to be considered when evaluating the degree distribution and interpreting how that distribution may affect graph navigability and robustness.

Next, the new metrics will be applied to a real-world SKG. The graph will consist of knowledge about people, places, organizations, suppliers, travel, and international incidents. Knowledge sources may include web-based news groups, multiple web-based news reports, and other information sources. The metrics will be applied both before and after access control policies are enforced. Although it would be useful to apply these metrics multiple times during graph evolution, it is unlikely that there will be adequate access to any production knowledge system that could provide real-world knowledge evolution in addition to real-world knowledge results. Thus, the metrics will be applied only after the complete graph is generated.

The second step in this work is to apply the extended metrics to real-world

graphs.

The purpose of collecting this data is to identify a network model that closely fits real-world knowledge networks. If the real-world network fits one of the models described in 2.1.2, then many important graph features will be identified. For example, the evolution of the knowledge graph will be characterized, and a method of computationally generating a topologically equivalent (though perhaps larger) graph will be available. Most importantly, insight into the effects of applying access control policies will be gained. This information could assist decision makers when they are performing the cost-benefit analysis technique described in Section 3.4. In particular, it could provide a shortcut for assigning a value to the cost C .

The third step in this work is to analyze the results of applying the metrics to identify which network model(s) most closely fits this class of real-world graphs.

3.2 Analyze KD Algorithms and Formalize the Analysis Process

As briefly discussed in Section 2.2, the extent to which input of a KD algorithm can be deduced from the output will be investigated as part of this work. This is important because ultimately, some security labelling rules will need to be applied to the output. If the input can be deduced, given the output, then the output will have to be labelled and secured appropriately. If not, then more relaxed security labelling may be acceptable (and even advantageous). In this Section, the term *deduce* is used loosely. Ultimately, the goal is to identify which, if any, of the security properties described in Section 2.3 hold for a given KD algorithm when specific labelling rules are

applied to the output. This section uses the term deducibility to refer collectively to the formal security vulnerabilities of Section 2.3: deducibility, inference, and interference (and any others that are found to be useful).

For many KD algorithms, there are two types of input, the background information, and the data over which knowledge discovery is to occur (i.e. the data in the graph). Some KD algorithms utilize only one of these two types of input. For clarity, the remainder of this section will refer to the first type as background data, the second type as graph data, and the collection of them both as input data. The deducibility of both types of input is of concern.

The remainder of this section refers repeatedly to *accessibility*. Rather than going into details about classification labelling and authorizations at this point, the term data *accessibility* will be used to refer to a single users ability to view data. It is assumed that if users are allowed to view one piece of data at their level of accessibility, then they can view all data available at that level.

Deducibility (again, this term is used loosely) depends on four conditions of the system:

- The amount of background information, if any, accessible by the user
- the amount of graph data, if any, accessible by the user
- the ability of the user to identify input data and match it to corresponding output data
- the extent to which the user is familiar with the KD algorithm(s) being used within the system

The conditions of the system, the KD algorithm, and the portion of the output that is accessible, will determine whether or not the input is deducible.

It is likely that all the algorithms belonging to a single KD category, as organized in Section 2.2, will have the same deducibility conditions. Thus, analysis will occur on the category level rather than the algorithm level. If this approach is found to lead to inaccurate results, then a few commonly used algorithms will be analyzed instead. In either case, the analysis process will be described.

The fourth step in this work is to analyze either at least three categories of KD algorithms from Section 2.2, or at least three KD algorithms, and for each, to indicate what portion of the output data needs to be made inaccessible to ensure that inaccessible input data is secure. The analysis will be done for each possible set of system conditions. The analysis process will be detailed so that it can be repeated for additional categories or algorithms.

These results could be used to assign a classification labelling policy based on the expected system conditions, or control the system conditions to maximize the amount of output data that can be made accessible.

To understand how this analysis might be accomplished, consider a simplified example, based on the KD algorithm from Section 1.4 that discovered the new link between Joe Smith and the KGB. Assume this algorithm is very simplistic and operates as follows:

```
FOR node_ID=1 to N DO
  IF node_type=person AND node has_account THEN
    IF node of node_type=(company OR organization)
      deposits_to that account
    THEN add link link_type=works_for,
```

```
source node=node_ID,  
destination node=company  
or organization node_ID,  
...  
ELSE add no links
```

This algorithm does not even consider travel information. Referring back to Anne's view of the graph in Figure 6 it is obvious that if she does not know the background information on which the algorithm is based she cannot accurately deduce the existence or content of the Secret and Top Secret information. This new link could have just as easily been added based on the fact that Joe Smith has travelled to Moscow and the KGB is located there. This conclusion makes several assumptions, such as Anne not having access to other examples of this algorithm discovering new links that might lead her to accurately infer the background information. This example is not rigorous in its analysis, but it illustrates some of the issues that will have to be considered when a more rigorous analysis is performed. Investigating exactly how to perform such an analysis is one of the most challenging aspects of this work.

3.3 Develop Confidentiality Models

As stated in Section 3.2, appropriate labelling rules will need to be applied to the output of KD algorithms. Confidentiality models, such as those discussed in Section 2.4, specify labelling rules. The application of different models will result in different security properties. Understanding how different models affect security is a key component of this work.

The models in Section 2.4 provide a good starting point for developing new models that will be useful in a KM system with complex confidentiality requirements. It is possible that some of those models may even directly apply to such a system. However, it seems more likely that these models will have to be extended so that they address labelling of discovered knowledge. This is simply because in the environment of interest, neither the KD process or its inputs can easily be identified as high or low. Rather, they are both high and low, and the current models do not address this situation. The example in Section 1.4 shows how assigning a label to discovered knowledge based on the highest level of input data could be unnecessary and costly. This is the approach that most existing security models would follow. The same example also shows that assigning the label based on the label of the majority of input data could provide insufficient security. Extensions to existing security models need address these complex issues.

The fifth step in this work is to extend existing security models to address a KM system with complex confidentiality requirements. Ideally, two or three extended models will be sufficient to provide a range of security properties, though establishing whether or not this is the case is part of the work to be accomplished. It may turn out that KD algorithms impact the range of security properties that can be achieved more do than the security models.

The existing security models considered for extension, will not necessarily be confined to those discussed in Section 2.4.

Though most of this discussion has been on models, ultimately a policy will need to be applied. A policy is simply an instantiation of a model, which is an abstraction. For the purposes of this work, the policies developed will be simple instances of the models. This paper often uses the two terms interchangeably to reflect the straightforward mapping we hope to achieve.

3.4 Develop a Cost-Benefit Analysis Technique

So far, this work had proposed the development and application of knowledge metrics, the analysis of KD algorithms, and the extension and application of confidentiality models. However, it is not the intent to prescribe a single confidentiality model as the solution for all KM systems. Rather, it is the intent to develop a technique that will allow KM system designers and policy makers to understand the tradeoffs between confidentiality and knowledge sharing in a quantitative fashion so that informed decisions can be made about security policies for KM systems.

To accomplish this task, we will develop a technique that identifies and correlates the costs and benefits of various confidentiality policy options. Cost is defined as the change in knowledge metric values that occurs as a result of applying a confidentiality policy. The “user” of this technique will have the option of weighting each metric by its relative importance, since they may not be of equal importance depending on the intended use of the system. The ability to weight the metrics allows them to be applicable in different environments. Then the weighted values will be combined into a single value to represent the navigability, N , of the knowledge graph. Intuitively, navigability is a measure of how easy it is for a user to explore the data managed by the KM system.

The sixth step in this work is to develop a function for combining weighted knowledge metrics into a single meaningful value, N , representing navigability.

Cost, C , can then be defined as

$$C = N_0 - N_P \tag{7}$$

where N_0 is the navigability prior to applying a confidentiality policy (all data resulting from a KD algorithm is marked low, or available to anyone), and N_P is the navigability after applying a confidentiality policy, P to this output.

The benefit, B , of a confidentiality policy will be described in terms of security properties. B can be a tuple (p_1, p_2, \dots, p_n) where each element is one of the security properties determined to be relevant for KM system confidentiality. Each element could be a percentage or a boolean value. Traditionally, the security properties in Section 2.3 would be boolean valued, indicating whether the property is present or not. However, to compensate for the weakness of nondeducibility, that property may be real valued, indicating to what extent input is nondeducible. In other words, if the user could deduce inaccessible input information with 85% accuracy, then the element of the tuple representing nondeducibility would be 0.85. Alternatively, if all of the properties are monotonically increasing in security strength, then B could simply be the strongest security property provided. For example, assume that there are three properties in the tuple: nondeducibility, generalized noninterference, and restrictiveness. Since restrictiveness implies generalized noninterference and generalized noninterference implies nondeducibility, it would be sufficient to indicate only the strongest property provided.

B is dependent on three aspects of the system: the confidentiality policy P , the KD algorithm or algorithm class Alg , and the system conditions Sys . P is a formally defined policy, but not a value. Alg is a formally defined algorithm or class of algorithms, which may be enumerable, but is not a value. Sys is a tuple (w, x, y, z) where w and x are real values representing the percentage of background information and graph information, respec-

tively, accessible to the user. Whether the user can correlate input values to output values is represented by the boolean value y . Whether the user knows which KD algorithm is being used is represented by the boolean value z .

Benefit, B , is not a function in the strict sense, since it is dependent on variables that are not numerical values. It is more useful to think of benefit as a three-dimensional matrix, with each cell value representing the benefit for the P , Alg , and Sys corresponding to that cell. Again, it is not the intent of this work to fill in every possible cell of this matrix. Rather, this work will define the steps that have to be completed so that decision makers can fill in their own matrix values for the policies, algorithms, and system conditions that are possible in their KM system.

The seventh step in this work is to outline the steps required for this cost-benefit analysis technique.

The benefit matrix and the values of C for the same policies used to define B , enable decision makers to do several things. First, given a fixed policy, algorithm, and system conditions, they can make informed statements about the security provided and the cost of providing that level of security. Second, they can tune the cost, benefit, or both, by altering the policy, algorithm, and/or system conditions. The following example provides a notional illustration of how this might work.

Assume a KM system designer must choose a confidentiality policy from among three candidate policies, $P1$, $P2$, and $P3$. The system conditions are fixed, so he only has to examine two dimensions of the benefit matrix. Assume B is a tuple (degree of nondeducibility, generalized noninterference, restrictiveness), where the first element is real valued and the latter two are boolean. Of the two different KD algorithms, he must assume both will be

	P1	P2	P3
Alg1	(100,0,0)	(98,0,0)	(100,1,0)
Alg2	(100,1,0)	(100,0,0)	(100,1,1)

Table 5: An example benefit matrix, with the system condition dimension excluded.

used in the system. The benefit matrix is shown in Figure 5.

Suppose the cost information is as follows: $C(P1) = .25$, $C(P2) = .02$, and $C(P3) = .6$. Since cost is the reduction in navigability caused by the security policy, a small cost is desirable. The costs shown here assume that $N = 1$.

The KM system designer could conceivably choose any of the policies. *P2* appears to be the best deal, since the cost so low. However, if the security requirements of the system stipulate that 100% nondeducibility is required in all cases, then he must choose one of the other policies. Given the considerable cost of *P3*, he should only choose that policy if generalized noninterference is also a strict requirement of the system. If he could choose both a policy and one of the algorithms, he could safely select *P2* and *Alg2* even if 100% nondeducibility is required.

3.5 Completion Criteria

Once the seven steps (shown in italics throughout Section 3) have been completed, this work will be considered complete. Modifications to the specific approaches taken to accomplish the goals described in these seven steps may be made as necessary, so long as the goals are achieved.

4 Schedule

Best Case: March 31, 2004 draft dissertation to committee. May 31, 2004 completion of all requirements for Ph.D. unless earlier date is required for Spring 2004 graduation.

Realistic Case: October 1, 2004 draft to dissertation committee. December 1, 2004 completion of all requirements for Ph.D. unless earlier date is required for Fall 2004 graduation.

- *Step 1:* March 2003 – June 2003.
- *Steps 2 and 3:* June 2003 – August 2003. Dependent on Step 1 and access to appropriate data.
- *Step 4:* June 2003 – December 2003. This is the most challenging component of this work.
- *Step 5:* October 2003 – February 2004. Partially dependent on Step 4.
- *Step 6:* 1 month, anytime between August 2003 and March 2004. Dependent on Steps 1, 2, and 3.
- *Step 7:* March 2004– April 2004.
- *Writing:* Throughout, and continuing through June 2004.

Helpful Resources

In addition to the many works explicitly cited throughout this paper, a handful of other works proved invaluable to the completion of this proposal.

Specifically, Albert et al. [3] provided an extensive and mathematically rigorous overview of nearly every aspect of complex networks discussed in this paper. Barabasi [7] and Watts [43] were also useful and are highly recommended for their lighter treatment of the same subject matter. Additional sources of insightful discussions about the BLP Model and security modelling in general include three papers by McLean [34] [33] [35]. Security model papers not discussed in Section 2.4, but that have been useful in understanding security requirements for ML/LS systems include Denning et al.'s two related papers on view-based security modelling [18] [19] and Landwehr and Lubbes paper that provides insight into the Orange Book security criteria [28].

Acknowledgement

This work was performed under the auspices of the U.S. Department of Energy by University of California Lawrence Livermore National Laboratory under contract number W-7405-Eng-48.

References

- [1] AGAT, J., AND SANDS, D. On confidentiality and algorithms. *IEEE Symposium of Security and Privacy* (2001), 64–77.
- [2] ALBERT, R., AND BARABASI, A.-L. Diameter of the world-wide web. *Nature 401* (September 9, 1999), 130–131.
- [3] ALBERT, R., AND BARABASI, A.-L. Statistical mechanics of complex networks. *Reviews of Modern Physics 74*, 1 (January, 2002), 47–97.

- [4] ALBERT, R., JEONG, H., AND BARABASI, A.-L. Error and attack tolerance of complex networks. *Nature* 406 (July 27, 2000), 378–381.
- [5] ANDERSON, R. A security policy model for clinical information systems. *IEEE Symposium on Security and Privacy* (1996), 30–43.
- [6] ANDERSON, R. *Security Engineering: A Guide to Building Dependable Distributed Systems*. John Wiley & Sons, New York, 2001.
- [7] BARABASI, A.-L. *Linked: The New Science of Networks*. Perseus Publishing, Cambridge, MA, 2002.
- [8] BARABASI, A.-L., ALBERT, R., AND JEONG, H. Scale-free characteristics of random networks: The topology of the world-wide web. *Physica A* 281 (2000), 69–77.
- [9] BARABASI, A.-L., ALBERT, R., JEONG, H., AND BIANCONI, G. Power-law distribution of the world wide web: Response. *Science* 287 (March 24, 2000).
- [10] BARABASI, A. L., DEZSO, Z., RAVASZ, E., YOOK, S.-H., AND OLTVAI, Z. Scale-free and hierarchical structures in complex networks. In *To appear in Sitges Proceedings on Complex Networks* (2004).
- [11] BARABASI, A.-L., RAVASZ, E., AND VICSEK, T. Deterministic scale-free networks. *Physica A* 299 (October 15, 2001), 559–564.
- [12] BELL, D., AND LAPADULA, L. Secure Computer System: Unified Exposition and Multics Interpretation. Technical Report MTR-2547 Vol.I, MITRE Corporation, 1973.

- [13] BELL, D., AND LAPADULA, L. Secure Computer Systems: Mathematical Foundations. Technical Report MTR-2997 Rev.I, MITRE Corporation, 1975.
- [14] BISHOP, M. *Computer Security: Art and Science*. Addison-Wesley, Boston, 2002.
- [15] BREWER, D., AND NASH, M. The Chinese Wall Security Policy. In *Proceedings of the 1989 Symposium on Security and Privacy* (May, 1989), pp. 206–214.
- [16] BRODER, A., KUMAR, R., MAGHOUL, F., RAGHAVAN, P., RAJAGOPALAN, S., STATA, R., TOMKINS, A., AND WEINER, J. Graph structure in the web. *Computer Networks: The International Journal of Distributed Informatique* 33 (June, 2000), 309–320.
- [17] DENNING, D. A lattice model for secure information flow. *Communications of the ACM* 19, 5 (May, 1976), 236–243.
- [18] DENNING, D., AKL, S., HECKMAN, M., LUNT, T., MORGENSTERN, M., NEUMANN, P., AND SCHELL, R. Views for multilevel database security. *IEEE Transactions on Software Engineering* 13, 2 (February, 1987), 129–140.
- [19] DENNING, D., AND LUNT, T. The SeaView Security Model. In *Proceedings of the 1988 Symposium on Security and Privacy* (May, 1988), pp. 218–233.
- [20] DOROGOVTSSEV, S., GOLTSEV, A., AND MENDES, J. Pseudofractal scale-free web. *Physics Review E* 65 (December, 2001), 1–4.

- [21] ERDOS, P., AND RENYI, A. On random graphs. *Publ. Math Debrecen* 6 (1959), 290–297.
- [22] FARKAS, I., DERENYI, I., JEONG, H., NEDA, Z., OLTVAI, Z., RAVASZ, E., SCHUBERT, A., BARABASI, A.-L., AND VICSEK, T. Networks in life: Scaling properties and eigenvalue spectra. *Physica A* 314 (2002), 25–34.
- [23] FAYYAD, U., PIATETSKY-SHAPIRO, G., SMYTH, P., AND UTHURUSAMY, R., Eds. *Advances in Knowledge Discovery and Data Mining*. AAAI Press and MIT Press, Menlo Park, CA and Cambridge, MA, 1996.
- [24] HALE, J., AND SHENOI, S. Catalytic Inference Analysis: Detecting Inference Threats Due to Knowledge Discovery. *IEEE Symposium on Security and Privacy* (1997), 188–199.
- [25] HOGG, T. Refining the phase transition in combinatorial search. *Artificial Intelligence* 81 (1996), 127–154.
- [26] JEONG, H., ALBERT, R., AND BARABASI, A.-L. Measuring preferential attachment for evolving networks. *Europhysics Letters* (April, 2001).
- [27] KIM, B. J., YOON, C. N., HAN, S. K., AND JEONG, H. Path finding strategies in scale-free networks. *Physical Review E* 65 (January 23, 2002).
- [28] LANDWEHR, C., AND LUBBES, H. Determining Security Requirements for Complex Systems with the Orange Book. *Proceedings of the Eighth National Computer Security Conference* (October, 1985).

- [29] LUNT, T. Aggregation and Inference: Facts and Fallacies. *IEEE Symposium on Security and Privacy* (1989), 102–109.
- [30] MCCULLOUGH, D. Noninterference and the composability of security properties. *IEEE Symposium on Security and Privacy* (1988), 177–186.
- [31] MCLEAN, J. A Comment on the 'Basic Security Theorem' of Bell and LaPadula. *Information Processing Letters* 20, 2 (1985), 67–70.
- [32] MCLEAN, J. Reasoning about security models. *IEEE Symposium on Security and Privacy* (1987), 123–131.
- [33] MCLEAN, J. Security models and information flow. *IEEE Symposium on Security and Privacy* (July 27, 1990), 180–189.
- [34] MCLEAN, J. The specification and modeling of computer security. *IEEE Computer* 23, 1 (1990), 9–16.
- [35] MCLEAN, J. Security models. In *Encyclopedia of Software Engineering*, J. Marciniak, Ed. Wiley Press, 1994.
- [36] MEADOWS, C. Extending the brewer-nash model to a multilevel context. *IEEE Symposium on Security and Privacy* (1990), 95–102.
- [37] MORGENSTERN, M. Controlling logical inference in multilevel database systems. *IEEE Symposium on Security and Privacy* (1988), 245–255.
- [38] PHILIPS, T., TOWNSLEY, D., AND WOLF, J. On the diameter of a class of random graphs. *IEEE Transaction on Information Theory* 36, 2 (March, 1990), 285–288.
- [39] RAVASZ, E., AND BARABASI, A.-L. Hierarchical organization in complex networks. *Europhysics Letters* (September, 2002).

- [40] SCHWARTZ, N., COHEN, R., BEN AVRAHAM, D., BARABASI, A.-L., AND HAVLIN, S. Percolation in directed scale-free networks. *Physical Review E* 66 (2002).
- [41] WALSH, T. Search on high degree graphs. In *Proceedings of the CP'01 Workshop on Modelling and Problem Formulation* (2001).
- [42] WATTS, D., AND STROGATZ, S. Collective dynamics of 'small-world' networks. *Nature* 393 (June 4, 1998), 440–442.
- [43] WATTS, D. J. *Small Worlds: The Dynamics of Networks Between Order and Randomness*. Princeton University Press, Princeton, NJ, 1999.
- [44] YOOK, S., AND BARABASI, A.-L. Weighted evolving networks. *Physical Review Letters* 86 (2001), 5835–5838.