Defence Research and Development Canada
Recherche et développement pour la défense Canada

MulVAL extensions II

Michael Froh and Glen Henderson

Defence R&D Canada – Ottawa

Canada

Contract Report
DRDC Ottawa CR 2009-132
August 2009

MulVAL extensions II

Michael Froh
Bell Security Solutions Inc.

Glen Henderson
Bell Security Solutions Inc.

Prepared by:

Michael Froh
Bell Security Solutions Inc.

Project Manager: Joanne Treurniet
Contract Number: W7714-6-3311
Contract Scientific Authority: Craig Burrell

Defence R&D Canada – Ottawa

Contract Report

DRDC Ottawa CR 2009-132

August 2009

Scientific Authority

Original signed by Craig Burrell

Craig Burrell

Approved by

Original signed by Julie Lefebvre

Julie Lefebvre
Head/NIO Section

Approved for release by

Original signed by Brian Eatock

Brian Eatock
Head/Document Review Panel

Abstract

MulVAL: A Logic-based Network Security Analyzer is a general networked system tool that reasons on multi-stage, multi-host attack paths. This work proposes extensions to MulVAL that are needed in order to have MulVAL reason on *exposed critical resources* in support of computer network defence situational awareness. Extensions are proposed to model high-level mission centric IT Services. These services are then mapped onto MulVAL using a simple dependency model. The concept of safeguard effectiveness and safeguard vulnerability is introduced, which will model the additional work effort required for an attacker to circumvent the safeguard. A proposed risk approximation for MulVAL is defined as a function of the variables *WorkFactor*, *AttackConsequence*, and *AssetValue*. This function and its values will need to be developed heuristically. The calculation of risk will allow the ranking of MulVAL generated attack paths to provide better reasoning on exposed critical resources.

Résumé

MulVAL : Un analyseur logique de la sécurité des réseaux est un outil de système en réseau général qui applique des raisonnements à des chemins d'attaque à utilisateurs multiples et à étapes multiples. Dans ce document, on propose les extensions MulVAL requises pour que MulVAL puisse appliquer des raisonnements à des *ressources essentielles à risque*, à l'appui de la connaissance de la situation de la défense des réseaux informatiques. Les extensions proposées visent à modéliser des services de TI généraux axés sur les missions. Les services modélisés sont mappés dans MulVAL à l'aide d'un modèle de dépendance simple, puis le concept de l'efficacité et de la vulnérabilité des mécanismes de protection qui servira à modéliser les efforts additionnels qu'un pirate devra déployer pour contourner ces mécanismes est introduit. Une approximation du risque proposée pour MulVAL est établie en tant que fonction des variables *WorkFactor* (facteur de travail), *AttackConsequence* (conséquence d'attaque), et *AssetValue* (valeur de biens). Cette fonction et ses valeurs devront être développées heuristiquement. Le fait de calculer le risque permet de classer les chemins d'attaque générés par MulVAL pour appliquer de meilleurs raisonnements aux ressources essentielles à risque.

This page intentionally left blank.

Executive summary

MulVAL extensions II

Michael Froh, Glen Henderson; DRDC Ottawa CR 2009-132; Defence R&D Canada – Ottawa; August 2009.

Background: Computer network defence is typically done via a static threat and risk model, and does not take into account which hosts are operationally critical. In reality, risks to a network are dynamic. In a military context, lives may be at stake if functionality is lost. Current technology addresses neither of these issues. Defence R&D Canada – Ottawa has been researching useful tools to model *dynamic asset protection* of military networks. A research project at Princeton entitled *MulVAL: A Logic-based Network Security Analyzer* was identified as a promising modeling tool for determining network attack paths. MulVAL can be applied to model computer network defence situational awareness (CNDSA). However, MulVAL does not adequately model: high-level IT Service definitions, mapping IT Services onto IT system elements, safeguards, or attack path ranking. This work proposes extensions to the current MulVAL model that are needed in order to have MulVAL reason on *exposed critical resources* in support of CNDSA.

Principal results: IT Services have been defined as: a $itServiceDefinition/6$, which defines the confidentiality, integrity and availability sensitivity of the data provided by the service; and a set of $allow/3$ MulVAL predicates, which define the authorized access of system users to the IT Service.

Research on various IT dependency models concluded that well-developed models, such as the Common Information Model (CIM), are too complex for inclusion in MulVAL; and that none provide useful automated collection of dependency information at this time. Therefore, MulVAL was extended to provide a simple dependency model by manually mapping $itServiceDefinition/6$ onto $networkService/5$. This mapping is then used to derive where IT Service data is mapped onto IT elements including: servers, clients, and networks.

A general framework was developed for positioning the development of MulVAL safeguard extensions. MulVAL currently models some access control safeguards (for example, $hacl/4$ defines authorized network communications). This approach means that the presence of safeguards disallow possible attack paths. The concept of modeling safeguards was extended to include the possibility of safeguard effectiveness and safeguards vulnerabilities. In this new approach, safeguards will not necessarily eliminate particular attack paths, but rather will increase the work factor involved in exploiting the attack path.

MulVAL currently reports sets of possible attack paths; however, there is no ranking of these attack paths from a degree of risk perspective. MulVAL does not model Threat Agents, their capability or intent; therefore it is not possible to calculate attack path likelihood. However, the risk associated with each attack path can be approximated using a $f(WorkFactor, AttackConsequence, AssetValue)$. We take the strategic approach of ranking outside MulVAL since Prolog does not easily lend itself to risk calculation. The exact definition of $WorkFactor$, $AttackCosequence$, and $AssetValue$ is not known at this time, nor is the f needed to combine these values to approximate risk. Therefore, a heuristic approach will be needed by attempting value sets and functions in meaningful model networks and considering the MulVAL risk ranking.

A strategy for handling MulVAL data uncertainty was proposed, including the handling of inaccurate data, as well as uncertainty in the the existing MulVAL concepts.

Significance of results: The results presented in this work provide enough extensions to the current MulVAL model that it is now possible to implement an experimental model with an aim to: validate the model extensions, develop further extensions (such as, additional safeguards), and to start heuristic validation of the proposed values and functions in approximating risk so that attack paths can be ranked.

Future work: The main focus of future work should be in implementing an experimental model of the proposed extensions to gain heuristic knowledge on how an extended MulVAL handles exposed critical resources.

Sommaire

MulVAL extensions II

Michael Froh, Glen Henderson ; DRDC Ottawa CR 2009-132 ; R & D pour la défense Canada – Ottawa ; août 2009.

Contexte : La défense d'un réseau informatique est généralement assurée au moyen d'un modèle statique de la menace et des risques, sans faire état des hôtes essentiels sur le plan opérationnel. En fait, les risques auxquels est exposé un réseau sont dynamiques. Dans un contexte militaire, une perte de fonctionnalité peut mettre des vies en jeu. Or, la technologie actuelle ne traite aucune de ces questions. R & D pour la Défense Canada – Ottawa a mené des recherches afin de trouver des outils utiles pour modéliser la protection dynamique des biens dans les réseaux militaires. Un projet de recherche, réalisé à Princeton et appelé *M*ulVAL : A Logic-based Network Security Analyzer, a permis d'identifier un outil de modélisation prometteur pour déterminer les chemins d'attaque de réseaux. MulVAL peut être appliqué pour modéliser la connaissance de la situation de la défense des réseaux informatiques (CSDRI), mais ne permet pas d'établir des modèles adéquats : définition des services de TI généraux, mappage de ces services vers des éléments de système de TI, mécanismes de protection et classement des chemins d'attaque. Dans le document, on propose les extensions au modèle MulVAL qui sont requises pour que MulVAL puisse appliquer des raisonnements à des *r*essources essentielles à risque, à l'appui de la CSDRI.

Principaux résultats : Les services de TI ont été définis comme suit : une définition de $itserviceDefinition/6$ qui porte sur l'importance de la confidentialité, de l'intégrité et la disponibilité des données fournies par les services, ainsi qu'un ensemble de prédicats $allow/3$ MulVAL qui déterminent l'accès autorisé des utilisateurs de système aux services de TI.

Les recherches effectuées à l'aide de divers modèles de dépendance de la TI ont permis de conclure que les modèles bien développés, comme le modèle d'information commun (CIM), sont trop complexes pour qu'il soit possible de les intégrer à MulVAL et qu'aucun de ces modèles ne permet pour l'instant une collecte automatisée utile de données de dépendance. On a donc élargi MulVAL de manière à procurer un modèle de dépendance simple en mappant manuellement $itServiceDefinition/6$ vers $networkService/5$. Le mappage obtenu peut alors être utilisé pour établir vers quels éléments de TI (notamment les serveurs, les clients et les réseaux) les données sur les services de TI doivent être mappées.

On a élaboré un cadre général pour positionner le développement des extensions de protection MulVAL. MulVAL sert actuellement à modéliser des mécanismes de

contrôle d'accès (par exemple, $hacl/4$ permet de définir les communications réseau autorisées). Selon cette approche, la présence de mécanismes de protection bloque les chemins d'accès possibles. Le concept de la modélisation de mécanismes de protection a été élargi afin d'englober l'efficacité et les vulnérabilités possibles des mécanismes de protection. Selon cette nouvelle approche, les mécanismes de protection n'élimineront pas forcément des chemins d'attaque particuliers, mais ils augmenteront le facteur de travail requis pour exploiter un chemin d'attaque.

Dans l'état actuel des choses, MulVAL signale des ensembles de chemins d'attaque possibles, sans toutefois classer ces chemins d'attaque en fonction de leur degré de risque. Comme MulVAL ne permet de modéliser ni les agents de menace ni leur capacité et leur but, il est impossible d'établir la probabilité d'un chemin d'attaque. Cependant, le risque associé à un chemin d'attaque peut être déterminé par approximation au moyen d'une fonction $f(WorkFactor, AttackConsequence, AssetValue)$. On adopte l'approche stratégique du classement à l'extérieur de MulVAL, étant donné que Prolog ne se prête pas facilement au calcul des risques. La définition exacte des valeurs $WorkFactor$, $AttackConsequence$ et $AssetValue$ n'est pas connue à l'heure actuelle pas plus qu'il n'est nécessaire d'avoir la fonction f pour combiner ces valeurs et faire une approximation du risque. Il faudra donc avoir recours à une approche heuristique en testant des ensembles de valeurs et des fonctions dans des réseaux de modèles significatifs et en examinant le classement des risques générés par MulVAL.

On a proposé une stratégie pour gérer l'incertitude des données MulVAL, notamment le traitement des données inexactes, de même que l'incertitude liée aux concepts MulVAL existants.

Signification des résultats : Dans les résultats qui sont présentés dans le document, le modèle MulVAL actuel est suffisamment élargi pour qu'il soit dès lors possible de mettre en œuvre un modèle expérimental permettant de valider les extensions du modèle, de développer d'autres extensions (par exemple, d'autres mécanismes de protection) et d'entreprendre la validation heuristique des valeurs et des fonctions proposées dans le cadre de l'approximation du risque afin de pouvoir classer les chemins d'attaque.

Travaux futurs : Les travaux futurs devraient porter principalement sur la mise en œuvre d'un modèle expérimental des extensions proposées, lequel servirait à acquérir une connaissance heuristique de la façon dont un modèle MulVAL élargi gère les ressources essentielles à risque.

Table of contents

List of figures

List of tables

1 Introduction

1.1 Background

Computer network defence is typically done via a static threat and risk model, and does not take into account which hosts are operationally critical. In reality, risks to a network are dynamic. In a military context, lives may be at stake if functionality is lost. Current technology addresses neither of these issues.

In 2005, Defence Research and Development Canada (DRDC) initiated an effort to develop a feasible abstraction in the area of defensive posture technology. This effort resulted in the creation of a white paper entitled: *Dynamic Asset Protection & Risk Management Abstraction Study* [1]. Concurrently, work by Beaudoin et al. [2] was carried out resulting in a model for computer network defence situational awareness (CNDSA), which consists of a series of information fusion stages which result in a final determination of the impact of an event on operations. A fusion element of this model brings together Operations, ITI, Safeguards, Vulnerabilities and Exploits to determine the defensive posture of the network, or in other terms, the exposed critical resources on the network.

A research project at Princeton entitled *MulVAL: A Logic-based Network Security Analyzer* [3, 4] was identified in the *Abstraction Study* as a promising area of research. MulVAL is an end-to-end framework and reasoning system that conducts multihost, multistage vulnerability analysis on a network. The output is a set of attack vectors specific to the network in question. Extensions to MulVAL that would be beneficial in furthering the work at DRDC on dynamic network defensive posture were identified in the report [1], and subsequent work began to address these issues [5]. Further work is required to extend the functionality of MulVAL to allow for a determination of the exposed critical resources.

1.2 Objective

The work that has been performed to date on MulVAL extensions has shown that MulVAL and the Open Vulnerability and Assessment Language (OVAL) are promising technologies for evaluating the exposed critical resources on a network. This work will address, or attempt to address, the functionalities that MulVAL will need to have in order to perform this task.

1.3 Report Structure

This report contains the following major sections:

1. Section 2 provides discussion on each of the major research areas in determining exposed critical resources;
2. Section 3 describes how high-level *IT Service Definitions* can be defined;
3. Section 4 discusses how system dependency models might be used to trickle-down *IT Service Definitions* onto system components. This section describes a proposed MulVAL model extension to derive $dataBind/3$ definitions based on a simple dependency model;
4. Section 5 describes a general approach to modeling safeguards within MulVAL;
5. Section 6 describes a strategy for determining attack path ranking on MulVAL reasoned attack paths;
6. Section 7 discusses how data uncertainty is present in the extended MulVAL model;
7. Section 8 provides ideas for further research;
8. Appendix A provides related thinking on a method of classifying safeguard modeling;
9. Appendix B provides a concise description of the proposed MulVAL Extensions; and
10. Appendix C provides a concise description of the current MulVAL model for completeness.

2 Exposed Critical Resources: A Roadmap

The objective of this work is to address those areas within MulVAL needed to provide a complete picture of *exposed critical resources*. As shown in Figure 1 exposed critical resources was defined in [2] as being derived from a number of inputs into a number of processes.

1. *Critical resources*, which are those resources needed to support the operations capability requirements, or IT Services. Critical resources are the Information Technology Infrastructure (ITI) elements that are combined to provide the required IT Services (triangle 1);
2. *System description*, which is a description of the ITI resource model including its safeguards (positive attributes) and vulnerabilities as negative system attributes (triangle 2);

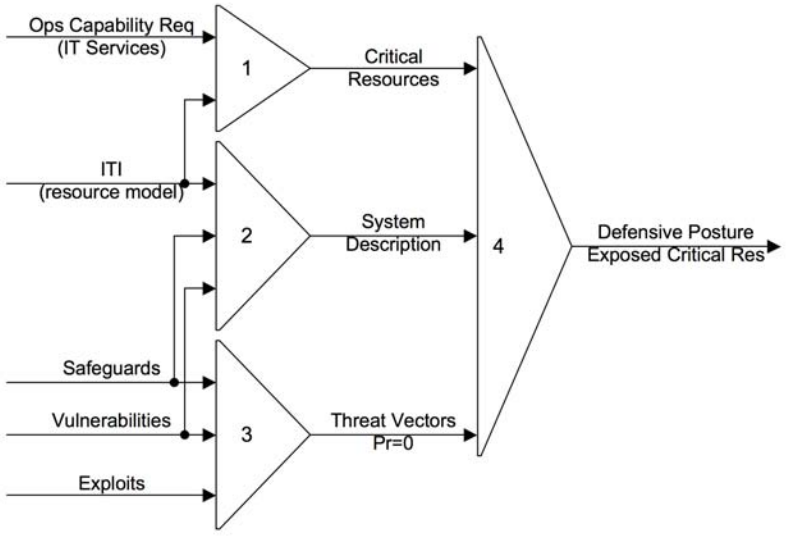

Figure 1: *Exposed Critical Resources*

3. *Threat vectors*, which are the known body of knowledge on how vulnerabilities, exploits, and safeguards interact (triangle 3); and

4. *Exposed critical resources* are a combination of *critical resources*, *system description*, and *threat vectors* (triangle 4). This provides a defensive posture to the networked system.

Given the current MulVAL model (see Appendix C), Figure 1 can be re-drawn from a MulVAL modeling perspective. The modified figure of exposed critical resources is shown in Figure 2 and described below.

1. *System description* remains a combination of ITI, safeguard and vulnerability description. This is currently modeled within MulVAL using automated OVAL scanners on all hosts within a network. The existence of vulnerabilities is taken from a a known source of vulnerability information based on software product name and version. The output of the OVAL scanner is a set of MulVAL primitive predicates which define the system in MulVAL (triangle 1);

2. *Critical resources* remains the same but a *trickle-down* modeling technique is needed to take as input high level IT Service requirements and map them onto the MulVAL system description (triangle 2). The input to this process is modified to be the MulVAL model of the system and that aspect of the ITI resource model that identifies resource dependencies;

3. *Vulnerability information* is taken from NVD, which includes CVSS vulnerability information. Note that exploits are not explicitly contained in NVD but the

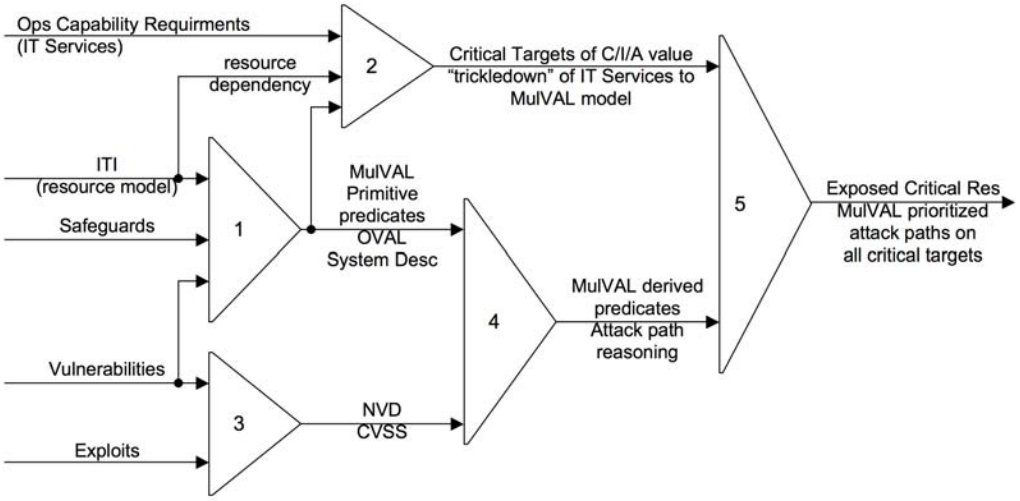

Figure 2: *Exposed Critical Resources from a MulVAL Perspective*

CVSS information does give an indication of the complexity and requirements needed to exploit the vulnerability. MulVAL assumes that exploits exist for all vulnerabilities in NVD (triangle 3); and

4. *General attack path reasoning* is a new element in the diagram that describes the existing MulVAL model reasoning on attack paths. This is defined as the derived MulVAL predicates which provide general logic reasoning on attacks on the system (triangle 4); and

5. *Critical target attack path reasoning* is the adoption of the existing MulVAL model to accommodate ITI component value (criticality). This process includes the ability to run multiple iterations of the general MulVAL reasoning in meaningful ways to establish a prioritized list of attack paths against the most critical targets (triangle 5).

The process described in Figure 2 can be expressed in an iterative manner that relates to a military observe, orient, decide, act (OODA) loop. Figure 3 shows how the Commander provides a mission objective to their headquarters staff, who then determine the combination of various functions to achieve that plan. In the case of J6, the headquarters defines the IT Services required by all other force elements to achieve the mission. Note that the bulk of the mission planning is based on doctrine of how the joint force elements interact. Figure 4 shows how a Commander's mission objective goes through a process of Mission Planning which leads to a set of IT Service Definitions. Figure 4 then shows an OODA loop involving MulVAL as an Observation and Orientation tool.

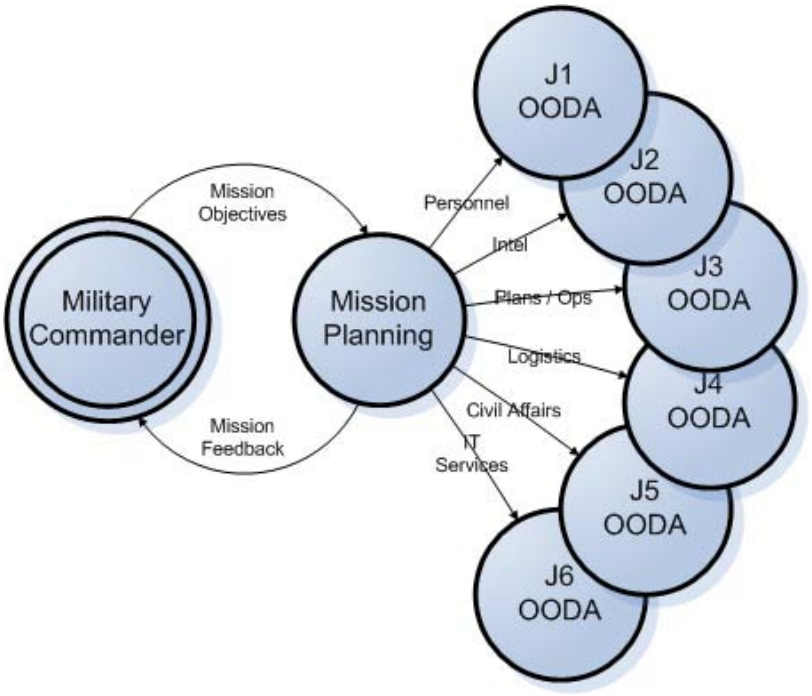

Figure 3: *General Mission Planning*

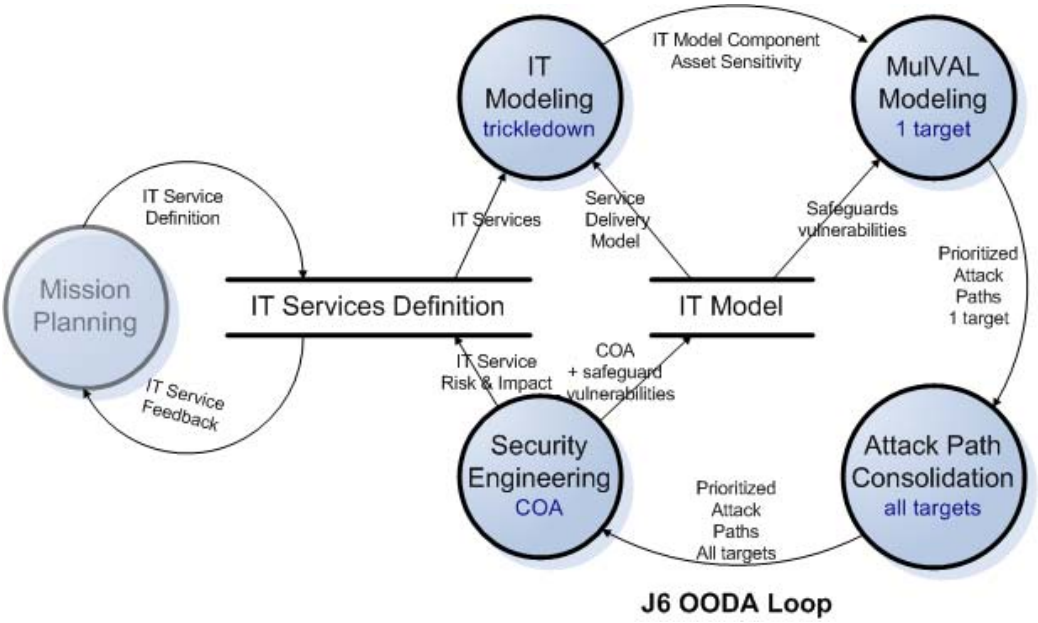

Figure 4: *High Level Process Flow Diagram*

1. *Observation*, from the perspective of MulVAL modeling exposed critical resources, is the collection of system description primitives from the OVAL scanners as well as mapping IT Security Services onto this network model;

2. *Orientation* is deriving attack paths within MulVAL to determine risk to exposed critical resources as shown in the MulVAL Modeling and Attack Path Consolidation activities;

3. *Decision* is outside the scope of the current MulVAL model and the current work but is shown in the figure as the security engineering exercise of adding safeguards or reducing vulnerabilities (the two basic courses of action available to the J6 commander); and

4. *Action* is outside the scope of the current MulVAL model and the current work.

This work will focus on the following areas:

1. Developing a standard means to describe IT Service Requirements. This is needed to identify the critical resources "from the top down";

2. Extending the MulVAL IT Modeling to include a "trickle-down" model for deriving IT Model component asset values based on broad IT Service Definitions above;

3. Extending the MulVAL system description (IT model) and attack path modeling to better describe safeguards protecting assets;

4. Developing a scheme for modeling safeguard effectiveness;

5. Further develop a method for ranking MulVAL attack paths both for a single target and for all critical resources (many targets); and

6. Develop a strategy for handling incomplete data in the MulVAL model.

3 Describing IT Service Requirements

Beaudoin et al [2] provides a starting set of requirements for describing IT Service requirements, which are assessed within the context of this work:

1. *Define a set of high-level IT Services.* This concept is considered in scope since it will define the primary business function of the network modeled within MulVAL. As such, it contributes to the definition of critical resources. The notion here is that operational capability is expressed as a set of specific information processing services. For example, a satellite imagery feed, real-time communications, and an air targeting application are all IT services which provide some operational capability. Note that the notion of operational capability runs counter to traditional threat and risk assessments (TRA) methodologies which focus mainly on information and not operational capability;

2. *Define the confidentiality, integrity, and availability (C/I/A) Quality of Protection (QoP) for each IT Service.* This concept is considered in scope since it defines fundamental asset value within MulVAL. As described in [5] the properties of confidentiality and integrity apply mainly to the information an IT service processes, while availability applies to the service itself[1];

3. *Define the required QoP over time.* This concept is considered out of scope since MulVAL does not yet model temporal changes in its modeling, so a MulVAL model is considered a snap-shot in time of the system being modeled. Temporal changes over time are required to build a complete IT Service definition but can be left to future work on enhancing MulVAL to model temporal changes in general;

4. *Define the confidentiality, integrity, and availability of IT Services relative to each other.* This concept is considered in scope since it helps deal with fundamental questions in ranking MulVAL attack path output. It addresses whether any one of C/I/A are biased more than the others;

5. *Define an IT Service relative to other IT Services in this network mission.* This concept is considered in scope since the modeled system will have a number of IT Services and the MulVAL attack path output must be ranked amongst all the provided services. Note that this requirement implies that at least a relative scale of C/I/A QoP values be defined in order to compare one critical service to another;

6. *Define an IT Service relative to other IT Services in other network missions.* This concept is considered out of scope since it deals with more complex strategic environments supporting multiple missions. At this time it is important to get a basic framework for defining IT Service Definitions. This requirement is seen as future work; and

7. *Define an IT Service relative to environmental events.* For example, access to port maps and harbour master notices becomes highly critical as ships are coming into port (the environmental event). This concept is considered out of scope since it deals with fundamentally temporal events. The MulVAL model first needs to be able to reason on a static snap-shot in time of the networked system. This requirement addresses a larger issue of near real-time iteration over the MulVAL model to provide a cyclic OODA processing environment.

[1] [5] did note that integrity also applies to processes, or services. However, for this initial work, we will focus on service availability

3.1 Define a set of high-level IT Services

As shown in Figure 3, the headquarters mission planning results in a number of mission sub-goals in each of the major force areas. In the case of J6 (Communications and IT Services), the mission sub-goal comes in the form of a set of IT Service definitions, which are composed of two categories of information:

1. The value of the data provided by the IT Service; and
2. The required authorized access to the IT Service.

We can define the data value in the following primitive predicate:

$$/******DefineITServices******/$$
$$itServiceDefinition(Data, ConfValue, IntValue, AvailValue, MttrValue, Bias). \quad (1)$$

where:

1. $Data$ refers to the data in the current primitive $dataBind/3$;
2. $ConfValue$ is the confidentiality value of the data and is defined in Section 3.2.1;
3. $IntValue$ is the integrity value of the data and is defined in Section 3.2.2;
4. $AvailValue$ and $MttrValue$ are the availability and mean time to repair (MTTR) values as defined in Section 3.2.3; and
5. $Bias$ is a value that specifies the inter-relationship between the $ConfValue$, $IntValue$, $AvailValue$, and $MttrValue$ and is defined in Section 3.3;

Predicate 1 defines the IT Service in a global sense. In order to relate this to a specific distributed user community to support the mission, the service must be made available to a set of principals by defining a set of appropriate $allow/3$ predicates:

$$\exists \; C_x = \{y | y \; is \; the \; set \; of \; Principals \; allowed \; Access \; to \; Data \; for \; itService_x\} \quad (2)$$

We have to be mindful of managing the $Data$ namespace since $allow/3$ can take only three values for $Access$: $read$, $write$, and $exec$, which represent the Unix file

permissions defined in the current MulVAL model. As such, a complex IT Service that defines many classes of *Data* may require multiple $itServiceDefinition/6$ predicates to properly express the desired service characteristics. For example, an intelligence service might might have two different classes of data, which requires different access for the same *Principal* or class of *Principal*. This might be expressed using the following predicates:

$$\begin{array}{l} itServiceDefinition(int\_Radar\_image, c, i, a, mttr, normal) \\ itServiceDefinition(int\_UAV\_image, c, i, a, mttr, normal) \\ allow(air\_Operator, write, int\_Radar\_image) \\ allow(air\_Operator, read, int\_UAV\_image) \end{array} \quad (3)$$

In this example, the fact that *air_Operator* needed *write* access to *int_Radar_image* but only *read* access to *int_UAV_image* forces the need to define two $itServiceDefinition/6$ predicates even though the remaining sensitivity values are identical and they relate to a single intelligence application.

3.2 Define the confidentiality, integrity, and availability (C/I/A) Quality of Protection (QoP) for each IT Service

3.2.1 Ranking Confidentiality

Ranking of Department of National Defence (DND) assets is governed by the Government of Canada's (GOC) Government Security Policy (GSP) [6], of which Section 10.6 states:

> **Confidentiality**
>
> Departments must identify information and other assets when their unauthorized disclosure, with reference to specific provisions of the Access to Information Act and the Privacy Act, could reasonably be expected to cause injury to:
>
> 1. the national interest. Such information is classified. It must be categorized and marked based on the degree of potential injury (injury: "Confidential"; serious injury: "Secret"; exceptionally grave injury: "Top Secret").
> 2. private and other non-national interests. Such information is protected. It must be categorized and marked based on the degree of potential injury (low: "Protected A"; medium: "Protected B", high: "Protected C").

Availability, Integrity and Value

Departments must identify and categorize assets, especially critical services, based on the degree of injury (low, medium, high) that could reasonably be expected to result from compromise to their availability or integrity. They must consider the value (e.g., monetary, heritage) of assets in determining injury. In order to indicate the level of safeguarding, departments should consider marking for availability and integrity purposes.

The policy from a confidentiality perspective requires the examination of injury (or impact) to the national interest or to non-national interests of disclosure of assets. Although the GSP suggests ranking on a high / medium / low scale, it was noted in [5] that ranking asset valuation using a small range of values on an absolute scale can be problematic. For example, using (High, Medium, Low) values is not sufficient to model the Government of Canada security policy for both classified and designated confidential assets. The classified (Top Secret, Secret, Confidential) and designated (Protected A, B, and C) scales are based on injury to the national interest and individuals/corporations, respectively.

However, when one combines the scales in a single absolute value range, highly classified national interest material (for example, Top Secret Special Access) typically eclipse all other values. For example, a viable absolute scale of asset value for various information classification is shown in Table 1. This skewing towards highly classified assets can then bury reasonably likely attacks to lower valued assets that should be addressed. Using a relative asset value scale that is specific to the system being modeled may be necessary. In these cases, MulVAL models of systems with differing assets are not comparable from an asset valuation or attack path injury perspective.

Table 1: *Hypothetical Asset Confidentiality Values*

Classification	Ranking Value
Top Secret	1.0
Secret	0.6
Protected C	0.4
Confidential	0.3
Protected B	0.2
Protected A	0.1
Unclassified	0.01

The values proposed in Table 1 are hypothetical. Assessing the correct values for asset confidentiality will require a heuristic approach of experimental development to ensure that appropriate values are set, especially in light of assessing attack path

ranking within MulVAL. These heuristics are needed in order to determine the ranking of impact to different critical assets with different confidentiality values.

3.2.2 Ranking Integrity

The GSP suggests that integrity be given a high / medium / low ranking scale. If we apply the same weighting spread used by CVSS on its vulnerability weightings we get a ranking value as shown in Table 2.

Table 2: *Hypothetical Asset Integrity Values*

Asset Description	Ranking Value	GSP rank
Mission Critical Data	1.0	High
Regular Business Data	0.7	Medium
Crap Data	0	Low

The values proposed in Table 2 are hypothetical. Assessing the correct values for asset integrity will require a heuristic approach of experimental development to ensure that appropriate values are set, especially in light of assessing attack path ranking within MulVAL. These heuristics are needed in order to determine the ranking of impact to different critical assets with different integrity values.

3.2.3 Ranking Availability

Availability is often calculated based on mean time to repair (MTTR)[2] and mean time between failure (MTBF) as shown in Equation 4 as noted in [7].

$$Availability = \frac{MTBF}{(MTBF + MTTR)} \tag{4}$$

Hussain [7] also notes that the term defects per million (DPM) can also be used and is directly correlatable to the 9's rating as $DPM = availability * 10^6$. DPM might be expressed as unsuccessful web page hits or Voice Over IP (VOIP) calls.

Note that availability in Equation 4 is an average value and may not provide sufficient definition of operational requirements since many combinations of MTBF and MTTR can provide the same availability. In a tactical military context, the commander is often interested in knowing an operational capability that is unavailable and its time to restoration. Therefore, MTTR from Equation 4 is also an important operational parameter.

[2]It might be useful to consider expressing a target MTTR with an allowable variance.

If both availability and MTTR are specified then MTBF can be calculated as an average value.

Availability within the telecoms domain uses the concept of *number of 9's* to define availability requirements. Table 3 shows the basic definition of each of the 9's ratings. Although any period could be chosen to determine the amount of allowed downtime, it seems appropriate to select one day, or 24 hours, when considering tactical military systems.

Table 3: *Telcom 9's Availability Ratings*

	Daily Downtime			
9's Rating	**Hours**	**Min**	**Sec**	**DPM**
90%	2.4	144	8640	100000
99%		14	864	10000
99.9%		1	86	1000
99.99%			8	100
99.999%			1	10

Note that adding each additional 9 provides an order of magnitude reduction in allowable downtime of a system for some period of time. Each step typically requires a significant amount of work to achieve the additional 9 availability.

Note also that for lower availability ratings, the amount of downtime could be composed of several different combinations of MTBF and MTTR. Higher availability ratings reduce the combinations of MTBF and MTTR to a much smaller set.

Table 4: *Hypothetical Asset Availability Values*

Availability	**Ranking Value**
99.999%	1.0
99.99%	0.8
99.9%	0.4
99%	0.2
90%	0.1

The MTTR value has a direct correlation to Availability. If $MTTR > 1min = 60sec$ then $avail <= 99.9\%$ by definition in that both have $\tilde{1}$ min of downtime considering a 24 hour counting period. We need to correlate the minimum MTTR to other operational capabilities. For example, if a downed IT Service implies I cannot shoot a weapon, then maybe 1 min is too long. From a critical business perspective 1 min seems pretty good.

Table 5: *Hypothetical Asset Availability MTTR Values*

MTTR			
min	**hr**	**day**	**Ranking Value**
1	–	–	1.0
60	1	–	0.8
240	4	–	0.4
1440	24	1	0.2
10080	168	7	0.1

The values proposed in Tables 4 and 5 are hypothetical. Assessing the correct values for asset availability will require a heuristic approach of experimental development to ensure that appropriate values are set, especially in light of assessing attack path ranking within MulVAL. These heuristics are needed in order to determine the ranking of impact to different critical assets with different availability values.

3.3 Define the confidentiality, integrity, and availability of IT Services relative to each other

The concept of defining relative ranking amongst the C/I/A values for a single IT Service is needed in order to provide ranking of the MulVAL attack paths that will impact the IT service. Table 6 shows all possible combinations of relative ranking amongst C/I/A. CVSS provides an impact bias parameter in their base score that provides a specific weighting to relative C/I/A impacts. These values are also shown in the table in column 2 along with the assigned weightings in column 3. The table also includes some suggestions for classes of information or IT Service that would have the specified C/I/A ranking.

Note that the ranking provided in Table 6 is relative only and does not imply some absolute weighting amongst the C/I/A values. The CVSS bias parameter does use the actual weighted values defined in column 3. The CVSS weightings are simple and self-consistent; however, it is not clear whether these values are sufficient to model C/I/A ranking for a single service. Our approach is to leave the definition of actual values to later research based on heuristic analysis of MulVAL attack path rankings.

3.4 Define an IT Service relative to other IT Services in this network mission

The C/I/A value ranges defined in Tables 1, 2, 4, and 5 are all relative with some hypothetical weights assigned. The current strategy is to carry all of these values for-

Table 6: *Asset C/I/A Relative Ranking Classes*

Ranking	CVSS C/I/A		Possible IT Services / Data
	Bias	**Weights**	
$C = I = A$	Normal	0.33/0.33/0.33	Secure VOIP Phone Service
$C > I = A$	Confidentiality	0.5/0.25/0.25	Raw Intelligence Data
$C > I > A$	–	–	Corroborated Intelligence Data
$C > A > I$	–	–	STU-III Phone Service
$I > C = A$	Integrity	0.25/0.5/0.25	Geographic Information Services (GIS)
$I > C > A$	–	–	Unmanned Aerial Vehicle (UAV) Imagery
$I > A > C$	–	–	smart bomb imagery
$A > C = I$	Availability	0.25/0.25/0.5	General Business Email
$A > C > I$	–	–	Combat Net Radio (CNR)
$A > I > C$	–	–	DNS Servers

ward to the MulVAL attack path ranking implementation. This will allow a heuristic approach to defining appropriate values based on the results of attack path ranking for assets of differing values.

4 Modeling Infrastructure

In this section we explore potential extensions to the MulVAL model to derive IT model component asset values from high-level IT Service definitions. We are looking for a method to Ótrickle-downÓ high-level IT Services onto those IT components that make up the service. In Figure 2, this model extension is shown as triangle 2. This implies the following of this model:

1. The inputs to this model extension are: IT Service Definitions, the ITI resource dependency model, and the MulVAL system model (that is, all the predicates describing hosts, services, principals, data, etc.). Ideally, the ITI resource dependency model should be provided using an automated data collection process;

2. The outputs of this model extension should be derived predicates in the MulVAL model based on the inputs defined above;

3. The model must capture how MulVAL model elements constitute a functioning service as defined in Section 3.1. The description of the resource dependency may be a combination of both primitive and derived predicates;

4. The model must trickle-down the IT Service's C/I/A values in an appropriate manner as defined in Sections 3.2 and 3.3. This will result in some derived

output predicates, which must be tied to the IT components captured in the MulVAL system description (that is, the OVAL scanner output);

4.1 Network Dependency Models

Ensel [8] provides a good review of the state of the art in dependency modeling. He states, in general, that dependency models are not currently used in IT management since automated models have two major problems: the lack of direct dependency information, and scalability issues. Ensel [8] categorizes three basic methods of building dependency models: Environmental Models, Abstract Models, and Object Models. Each model type has its advantages and disadvantages. Ensel goes on to define an automated method of building real-time dependency models using neural nets. This approach to automating dependency models may be of use in a real environment and work alongside MulVAL. Further research is needed to determine if this automated approach is suited to MulVAL.

Common Information Model (CIM)[3] [9] is an abstract model. CIM is a very complex model which attempts to model all aspects of IT systems from a systems management perspective. The CIM model is heavily weighted towards low level system information collection and correlation; however, it does model high-level IT Services as one of its upper-most modeling concepts. From general reading it appears that the bulk of the work in automating CIM model population is at the lower, more detailed CIM model levels. This was confirmed by Ensel [8]. The authors have a significant concern with taking the relatively simplistic MulVAL model and binding it to the overly complex CIM model for the purposes of defining an ITI dependency model. To this point, a development principle for MulVAL has been to keep things as simple as possible.

Keller and Ensel [10] present an approach to managing service dependencies at a high level using the Resource Description Framework (RDF). This research provides useful information for building tools to manage IT Service dependencies. The work does not directly address how high-level dependency information is collected in any automated fashion; however it does provide a concise view of the approaches for automated dependency collection. The following text is quoted from [10] and the references are Keller and Ensel's original references:

> "**2.2.2 Acquiring Dependency Information from Managed Resources**
>
> The question where the dependency information on the managed resources comes from is another crucial issue, although this is not fully within the

[3]CIM is a information model schema developed by the Distributed Management Task Force (DTMF). The exchange of CIM operations over HTTP allows CIM implementations to interoperate in an open, standardized manner in support of the Web Based Enterprise Management (WBEM) initiative

scope of this paper. For the sake of completeness, we will briefly mention some of the more common approaches:

- The most straightforward way is to provide appropriate instrumentation within the applications and services themselves; the problem is that none of today's applications is able to provide this kind of information at an acceptable granularity.
- Another approach consists in instrumenting the communication protocol stack and/or some shared libraries of the host system to intercept the communication between different parties in order to infer potential dependencies. The resulting information could be either provided by a specific 'dependency agent' or given out as flat files. [28] describes a CORBA-based approach for message reflection based on portable interceptors to gather information regarding the messages exchanged between the objects of a distributed application.
- [15] describes an approach that makes use of information stored in system configuration repositories for generating appropriate service dependency information.
- A commonly used technique in system and protocol design, which has only recently been applied to service and application management [2] is the active perturbation of components within a system (i.e., injecting faults in a controlled manner and observing the behavior of the components) while run- ning synthetic transactions against it. This technique can be used to obtain the required dependency information; however, great care has to be taken if it is used on production systems.
- Other approaches come from the area of Artificial Intelligence. [7] uses Neural Networks to interpret low-level information like the activity of pairs of services over time. This allows to derive dependencies even in heterogeneous environments.
- The OSI General Relationship Model (GRM) [14] defines a powerful generic model for defining relationships between managed objects and provides a mechanism for qualifying these relationships by means of attributes. In addition, the GRM specifies extensible operations that can be invoked on the managed relationships. While its functionality is needed in any distributed system, the GRM is tightly coupled with the OSI Structure of Management Information and CMISE and, thus, has not been used outside of TMN environments. The work described in [6] is a precursor to the GRM and addresses similar issues; the architecture and implementation of a GRM based management system is the subject of [23].

- Finally, a CIM Object Manager (CIMOM), as proposed by the Distributed Management Task Force (DMTF) could be used to expose the necessary information. The CIM Core Model [3] provides an association class CIM Dependency, from which many subclasses are derived.

Every one of the aforementioned approaches for generating dependency models has its specific advantages and drawbacks. Given the fact that dependencies cross system and organizational boundaries, it is likely that a combination of some of these approaches is needed to yield the most comprehensive amount of dependency information."

The use of package managers such as the RPM Package Manager (RPM) can be used to define software dependencies in system configurations. The dependencies defined are usually relate to library or environmental parameters. Therefore, RPM structures are very useful for populating *dependsOn*/3 MulVAL primitive predicates. The use of RPM, or similar package managers, could provide an automated source for deriving a software and program dependency model, but do not provide any sense of mapping high-level IT Services onto ITI elements.

Keller et al [11] extend the CIM Metrics Model to deal with metric aggregation. They note that *Service Level Management* is based on Service Level Agreements (SLA) that are negotiated between a customer and service provider. SLAs often contain precise definitions on how lower-level resource data is aggregated into higher-level service parameters. This concept is a similar concept to our high-level IT Service definition. The work may provide a useful framework for providing aggregation functions on impact modeled after their performance aggregation model.

4.2 A Simple MulVAL Dependency Model

Given the lack of credible dependency models with automated collection capabilities, we propose a strategy for developing this capability within MulVAL. An abstracted layered approach to defining IT Service resource dependencies will be taken as shown in Figure 5. With abstraction we can leave the network extension complexity from the model extension and focus solely on the provision of IT services. In this way the *hacl*/4 rules become the transition between the two abstract layers in Figure 5. That is, from an application perspective, the details of how *hacl*/4 predicates are derived are hidden from the application model, it just knows whether a specific communication is allowed or not. Our approach to using abstraction is:

1. First, we attempt to define a dependency model only at the application service layer. Lower layer network management is more mature in mapping and modeling low level network elements. It is the higher level dependencies amongst application services that is now manually derived, if at all;

2. Next, we can build simple MulVAL predicates to derive the aggregate value of other MulVAL elements such as: hosts, directories, nfs servers, etc.;

3. As future research, the modeling of network elements can be considered to model network element dependency.

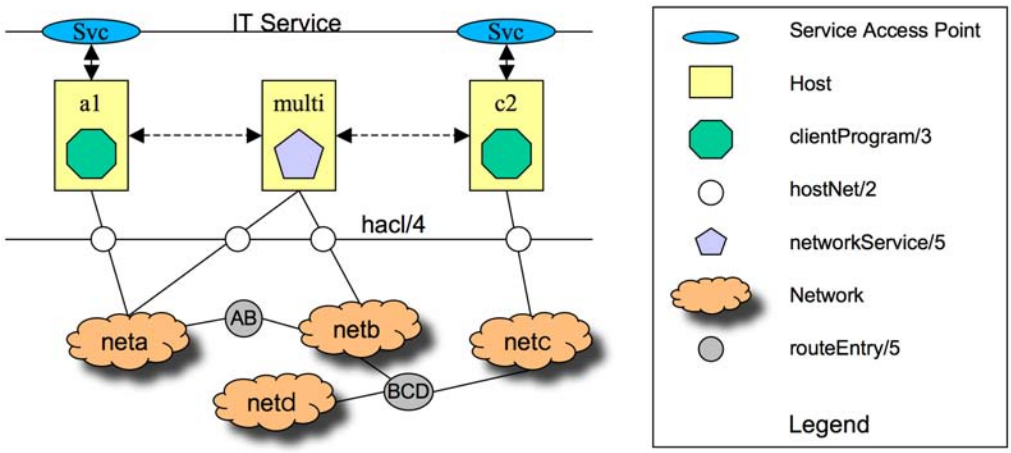

Figure 5: *Layer Abstraction*

As shown in Figure 5, we can generally build the provision of high level IT Services within the current MulVAL model as follows flowing from the IT Service definition down to users accessing the service:

1. The IT Service is defined to have a specified C/I/A value as a collective whole (new primitive predicate);

2. In light of no feasible automated dependency model, we must define which network services are providing the IT Service (new primitive predicate);

3. The service runs under a specific set of privileges or account on the server host ($networkService/5$);

4. The service account has certain file access privileges on the server host ($accessFile/4$);

5. Now that the IT Service is tied to network service(s) on server host(s) with account and file access, we need to derive a binding of IT Service values to the data being served by the network service(s) and the network service(s) themselves (new derived $dataBind/3$ for data, and new derived predicate for services);

6. The service is associated with software that provides the service ($networkService/5$);

7. The server host must have a service running on a specified port ($networkService/5$);

8. The client host must have a valid network path using the required protocol to the correct port on the server host ($hacl/4$);

9. The client host must be attached to a network ($hostNet/2$);

10. The client host must have a client program capable of accessing the service ($clientProgram/2$ which needs to be extended to tie to service or protocol);

11. Users need an account on the client host ($hasAccount/3$);

12. Users require physical access to a client host (not modeled in MulVAL);

The following text is an attempt to extend the MulVAL model with the generalized trickledown approach described above. We first define the IT Service using Predicate 1 and 2 as shown in Section 3.1. For example, an intelligence database might be defined as being Top Secret, of normal Business Data, needing 99% availability with a one day MTTR, and a confidentiality bias:

$$itServiceDefinition(intDB, topsecret, busdata, two9, oneday, confidentiality). \tag{5}$$

We now need to map this service onto one or more network services within an existing MulVAL system description. That is, OVAL scanners have already populated all of the hosts, network services, accounts, and file access on the network. We map which hosts, programs and accounts are deemed to provide the IT service using the following primitive predicate:

There are several options for defining the mapping of IT Services onto network elements including:

1. Directly map the $Data$ defined in $itServiceDefinition/6$ onto $Hosts$ and $Path$ using $dataBind/3$ primitive predicates. Then use $accessFile/4$ to derive what $networkService/5$ can access and serve the $Data$. This approach assumes that the number of $dataBind/3$ primitives to manually enter is manageable;

2. Define a new primitive that binds the $Data$ in $itServiceDefinition/6$ to a $networkService/5$, then use the $accessFile/4$ derived predicate to derive $dataBind/3$ for all network paths. Duplicating the entire $networkService/5$ predicate in a new primitive predicate seems like carrying too much irrelevant information since the protocol and port are more the mechanics of how data is moved and

not the fundamental definition of what the network service provides. It seems more appropriate to carry only the salient items from $networkService/5$, which includes the Host, Program, and Account elements. This allows for the same program to be running multiple times on a host and supporting different IT Services[4]. This strategy is not as tightly defined as networkService/5 but may be a good generalization.

$$serviceBinding(Service, Host, Program, Account). \tag{6}$$

In general, there can be a $m : n$ relationship between $itServiceDefinition/6$ and $networkService/5$. That is, an $itServiceDefinition/6$ might be mapped to multiple $networkService/5$ and a $networkServices/5$ can support many $itServiceDefinition/6$. This approach assumes that there is less work to manually create $serviceBinding/4$ than to define $dataBind/3$ predicates by assuming that since the $number\ of\ networkService/5 << number\ of\ dataBind/3$; and

3. Define the dependency model using the $dependsOn/3$ primitive predicate to have a Service depend on a Program on a specific Host. The simplicity of this approach is appealing; however, this is probably too loose a coupling and would not provide a realistic mapping since it assumes that all programs of the same name on the same host will provide the service. Using this approach would require careful management of the $Program$ namespace within Prolog.

$$dependsOn(Host, Service, Program). \tag{7}$$

Based on the assumption that the number of path statements produced from an OVAL scan of a network will far exceed the number of network services, the proposed approach is to use the $serviceBinding/4$ and then derive $dataBind/3$ predicates.

Continuing our example, the intelligence database might be specified as running on two servers:

$$\begin{array}{l} serviceBinding(intDB, intsvr2, mysql\_5.0, mysql). \\ serviceBinding(intDB, intsvr32, postgresql\_4.7, postgres). \end{array} \tag{8}$$

As noted in [5], a new $dataBind/5$ predicate is introduced that includes a confidentiality and integrity value to the existing $dataBind/3$ primitive predicate.

[4]It is not clear if this is a common practice in IT systems administration

This predicate can be derived under the conservative assumption that any files accessed by the network services providing the IT Service would store the IT Service *Data* at those file locations. Note that this predicate does not yet account for $accessFile/4$ from a network file service such as nfs. Therefore, $dataBind/5$ can be derived as shown below:

$$
\begin{array}{l}
/******dataBind/5Derivation******/ \\
dataBind(Service, Host, Path, C, I) :- \\
\quad itServiceDefinition(Service, C, I, A, MTTR, Bias), \\
\quad serviceBinding(Service, Host, Program, Account), \\
\quad networkService(Host, Program, \_protocol, \_port, Account), \\
\quad accessFile(Account, Host, _{a}ccess, Path).
\end{array}
\tag{9}
$$

In our intDB example we would derive the following $dataBind/5$ statements assuming the $networkService/5$ and $accessFile/4$ predicates shown:

$$
\begin{array}{l}
dataBind(intDB, intsvr2, /var/mysql/data, topsecret, busdata) :- \\
\quad itServiceDefinition(intDB, topsecret, busdata, two9, oneday, confidentiality), \\
\quad serviceBinding(intDB, intsvr2, mysql\_5.0, mysql), \\
\quad networkService(intsvr2, mysql\_5.0, \_protocol, \_port, mysql), \\
\quad accessFile(mysql, intsvr2, \_access, /var/mysql/data).
\end{array}
\tag{10}
$$

$$
\begin{array}{l}
dataBind(intDB, intsvr32, /usr/local/pg/data, topsecret, busdata) :- \\
\quad itServiceDefinition(intDB, topsecret, busdata, two9, oneday, confidentiality), \\
\quad serviceBinding(intDB, intsvr32, postgresql\_4.7, postgres), \\
\quad networkService(intsvr32, postgresql\_4.7, \_protocol, \_port, postgres), \\
\quad accessFile(postgres, intsvr32, \_access, /usr/local/pg/data).
\end{array}
\tag{11}
$$

As suggested in [5], a new $networkService/7$ predicate is introduced that adds availability information to the existing $networkService/5$ primitive predicate. Given the multiple factor definition of availability the new predicate needs to be $networkService/7$ to accommodate both availabilty and MTTR values.

$$
\begin{aligned}
&/ * * * * * * networkService/7 Derivation * * * * * * / \\
&networkService(Host, Program, Protocol, Port, Priv, A, MTTR) :- \\
&\quad itServiceDefinition(Service, C, I, A, MTTR, Bias), \\
&\quad networkService(Host, Program, Protocol, Port, Priv).
\end{aligned} \tag{12}
$$

Similarly we can extend the $networkService/7$ availability to any libraries that the service depends on:

$$
\begin{aligned}
&/ * * * * * * DeriveLibraryAvailValue * * * * * * / \\
&libraryAvailValue(Library, A, MTTR) :- \\
&\quad networkService(Host, Program, \_protocol, \_port, \_priv, A, MTTR), \\
&\quad dependsOn(Host, Program, Library).
\end{aligned} \tag{13}
$$

We can further derive ITI element values for server hosts by using the following derived predicates:

$$
\begin{aligned}
&/ * * * * * * DeriveHostSensitivityForServers * * * * * * / \\
&hostSensitivity(ServerHost, C, I, A, MTTR) :- \\
&\quad itServiceDefinition(Data, C, I, A, MTTR, \_bias), \\
&\quad serviceBinding(Data, ServerHost, \_program, \_account).
\end{aligned} \tag{14}
$$

Similarly we could derive ITI element values for client hosts by extending the value of data on servers to the client hosts that principals use to access that data by using the following derived predicate. Note that these predicates assume the extended use of the $access/3$ predicate defined in [12]:

$$
\begin{aligned}
&/ * * * * * * DeriveHostSensitivityForClients * * * * * * / \\
&hostSensitivity(ClientHost, C, I, A, MTTR) :- \\
&\quad itServiceDefinition(Data, C, I, A, MTTR, \_bias), \\
&\quad access(Principal, \_access, Data), \\
&\quad hasAccount(Principal, ClientHost, \_account).
\end{aligned} \tag{15}
$$

We can extend the value derivation to networks by assessing the value of the hosts attached to the network:

$$
\begin{aligned}
&/ * * * * * * DeriveNetSensitivity * * * * * * / \\
&netSensitivity(Net, C, I, A, MTTR) : - \\
&\quad hostSensitivity(Host, C, I, A, MTTR), \\
&\quad hostNet(Host, Net).
\end{aligned}
\tag{16}
$$

When deriving ITI element asset values, MulVAL will create separate logic relations according to the derived predicates. One must be careful in crafting queries since there is no relational logic associated with multiple relations for a single ITI element. That is, some software may have multiple MTTR values assigned to it but there is no logic to define how multiple values are aggregated. For example, if a library has two MTTR values (1min and 4hours), querying $libraryAvailValue(libraryX, A, 4hours)$ will return $TRUE$ and may give a false impression of the derived aggregate availability requirement of the library. Whereas a query of $libraryAvailValue(libraryX, A, MTTR)$ will return all A and $MTTR$ values. It would then be up to some external routine to determine an aggregate value for the component.

The further extension of Sensitivity into the network layer abstraction is left for future research.

The simple dependency model derived here assumes a client/server application architecture. Consideration for other application models is left for future research (for example, publish/subscribe, or peer to peer).

5 Modeling Safeguards

The MulVAL model, given its Datalog implementation, performs analysis based on the expression of facts and rules about the modeled environment. The implemented rules identify violations of policy due to:

1. Improper configuration, for example, a user that can access a data set when the security policy should not support this access; and

2. The presence and exploitability of software vulnerabilities that would allow a malicious user to access or modify information in violation of policy (or render that information unavailable).

The latter scenario is the core function of MulVAL, that is, using public domain information to identify exploitable vulnerabilities in the modeled environment. It is significant to note that the triggering event in MulVAL for generating this list of attack paths is the matching process. A vulnerable piece of software must be found in the modeled environment in order for an attack path to be identified. Conversely,

the absence of vulnerabilities presents the modeled environment as Ôless at riskÕ, presenting safeguards as an absence of vulnerabilities.

This view of safeguards as being modeled as the absence of vulnerabilities is supported by many of the facts and rules that are used to describe the environment. For example, the *hacl*/4 rule shows which machine to machine connections are possible. Alternatively, any machine-to-machine connection that is not presented in a *hacl*/4 rule is not possible, that is, access control rules completely prevent this activity. The MulVAL model represents what operations are possible. Given the matching process in Datalog, if the capability to perform an illegal operation is not expressly stated, that operation will not match in the reasoning rules and will never be presented as a vector for an attack. Safeguards, in this case, are implicitly modeled into the environment.

The difficulty of this implicit representation is that many safeguards are themselves prone to attack. Very few safeguards provide absolute protection of the information they are positioned to protect. In reality, the majority of safeguards provide a measure of resistance against an attack. This resistance can be presented in terms of, among others:

- The prevention of attacks;
- The detection/deterrence of attacks; and
- The containment of the impact of attacks.

The challenge for MulVAL is, then, to represent accurately the role of safeguards and to provide a measure of the protection these safeguards bring to the modeled environment.

The original research into the development of a Dynamic Asset Protection Abstraction [1] collapsed a set of IT infrastructure and process layers into the following domain aggregation:

- Social
- Logical
- Physical

Similarly, a review of the modeled MulVAL facts identifies additional granularity of the logical domain, providing constructs for information at the follow logical levels:

- The network level (location);
- The system level (host) and

- The application level (program).

These five domains, therefore, provide a breadth of scope for the identification of safeguards that will have an effect on the security posture of an IT environment. Within these domains, many types of safeguards can be deployed to counter the vulnerabilities that may be present within each domain. Given that DRDCÕs interest in MulVAL is as a dynamic risk-modeling tool, it is a useful exercise to examine the Threat and Risk Assessment (TRA) methodologies for classifying safeguards. There is no definitive approach for safeguard classification. Risk management specialists draw upon many sources of risk modeling theory and practices to ensure that a sufficiently broad scope of investigation is set for identifying and evaluating safeguards. Universally, safeguards are described as belonging to one of the following control types:

- Physical controls: physical restriction of access, physical hardening, and electronic monitoring;
- Administrative controls: policies, procedures, guidelines and processes such as employee screening; and
- Logical controls: technical controls such as firewalls, IDS, I&A models and cryptography.

The BSSi risk management practice has drawn upon NIST [13], GoC [14] and ISO [15] standards to synthesize a comprehensive list for the categorization of safeguards. In the context of this paper, the intersection point between this dual view of safeguards, that is, what the safeguard protects (the type) and where within the IT organization the safeguard applies (the DAP layer, or domain) is presented as a safeguard class in this paper. The following table describes the valid safeguard classes and where there is an existing MulVAL fact or rule that captures the nature of a specific safeguard class.

Please refer to *Annex A: Safeguard Type Definitions* for a description of the safeguard types presented in this table.

The cells in this table, representing safeguard classes, fall into one of four categories:

1. There is no intersection, the type of safeguard described does not have a significance for the domain.
2. The MulVAL model already represents this safeguard class.
3. The MulVAL model can be extended to include this safeguard class.
4. The MulVAL model does not (currently) have the ability to represent this safeguard class.

Safeguard		DAP Layer				
		Social	Logical			Physical
Type	Protection Mechanism		Application	System	Network	
Security Policy			allow	dataBind		
Organizational Security						
Risk Management			MulVAL	MulVAL	MulVAL	
Personnel		incompetent				
Physical						
System Acquisition, Development, Operations & Maintenance			vulExists (absent)	vulExists (absent)		
System Integrity & Security Configuration	Information Disclosure					
	Input Validation					
	DoS					
Identification & Authentication	local ID			has Account inGroup		
	Enterprise ID					
Access Control				fileOwner fileGroupOwner fileAttr	hacl hostNet routeEntry	
Cryptographic Services						
Audit & Accountability						
Incident Management						
Business Continuity Management						
Compliance						

Figure 6: *Full List of Safeguard Classes*

The shading in the table represents those types of safeguards, or DAP layers that are not within the scope of this report, and will not be examined.

It is significant to note that the delineation between the 3rd and 4th categories is not clear at this point. It is safe to say that certain safeguards can easily be included into the existing MulVAL model, whereas other safeguards will require substantial rework of the model before they can be included in the analysis.

5.1 Incompatible Safeguard Classes

The are currently no relevant mechanisms in MulVAL for representing the safeguards classes discussed in this section.

Physical Controls and the Physical Domain. The NVD, the primary source of vulnerability information, recognizes locally and remotely exploitable vulnerabilities. This criterion applies to vulnerabilities that require interaction through an interactive session with the target system and vulnerabilities that can be exploited through network connectivity to the target system, respectively. To include within the NVD those vulnerabilities that can be exploited through physical (but not necessarily requiring an interactive session), a new category of access complexity would be needed to include physical connectivity. For example, a method to bypass authentication and access control mechanisms using a terminal attached to a local port on the target system would be a physical vulnerability, since the only requirement is physical access to the system. This is a common vulnerability in network appliance devices.

In the absence of NVD support for physical attacks, physical vulnerabilities would require the creation of an entire new type of vulnerability. For example, if an attacker gains physical access to the data center, a broad range of DoS attacks can be initiated including, effecting power outages, theft of systems, destruction of systems and network disruption. Modeling physical controls would require the introduction of many new concepts, including physical separation of zones, physical access control mechanisms, building layout and construction materials. There are a sufficient number of extensions and rework to integrate these concepts in MulVAL to state that a comprehensive inclusion of physical controls is not compatible with the current model. It is also important to note that physical security is as much concerned with infrastructure resiliency in the face of natural and man-made disasters as with malicious attack. Fire suppression and HVAC are all within the realm of physical security and must be considered in the context of modeling these controls.

Personnel Controls and the Social Domain. There is a minimal representation of personnel security in MulVAL in the incompetent user predicate. This is an important factor in determining the success of what ICAT considers ÔuserÕ attacks, where an unsuspecting user must be duped into unwittingly participating in the attack (for example, running a piece of malware or participating in a phishing attack). Personnel security in this context must be a function of the level of trust (or alternatively competency) in an employee. Factors that will increase the organizationÕs level of trust in an employee would include:

- Security awareness training
- Long periods of time between security events

- Number of security incidents where the user was at fault
- Vouching from other trusted employees
- Background checks

However, the majority of personnel controls are procedural based and are not modeled in MulVAL. In general ÔsoftÕ security safeguards such as policies, procedures, guidelines and standards, both their definition and execution, will be difficult to measure in an objective and relevant fashion. No two organizations will have the same policy and it is not clear how 2 distinct security policies can be compared. This is very much the challenge of the TRA process where very often the analyst is called upon to use judgment as to the sufficiency of the soft security practices. It may be possible to state whether or not a needed policy document or practice exists; it is very difficult to assess how effective these safeguards are in achieving a robust security posture or mitigating attacks.

Additional Soft Security Practices. Similarly to the safeguard classes discussed above, the following safeguards will not have a relevancy in the current MulVAL model, given their origin in the statement and enforcement of policy, practices and procedures:

- Organizational security;
- Incident Management; and
- Compliance.

Returning to the original categorization of control types, therefore, it is clear the MulVAL has a bias towards the support of technical controls. Physical and administrative controls will have a minor role in the development of MulVAL modeling given their subjective nature and the challenge in developing specific vulnerabilities that are mitigated through these controls. This modeled challenge may be reduced over time as the MulVAL model evolves to include new concepts, domains and enhanced scope.

5.2 Current MulVAL Safeguards

Currently, MulVAL model represents the following safeguards.

Security Policy. The *allow*/3 predicate specifies what is present in the organizationÕs security policy as allowable data access. It is presented in terms of unique principals, data constructs and the privileges that user should have on the data in question. This is a highly granular access control specification, but well within the intent of system-specific security policy definitions.

The *dataBind*/3 predicate creates a logical construct for the application of policy rules. A set of data elements are aggregated under a single symbol that identifies the data as belonging to a single logical group with the expectation that a common set of policy rules should apply to that data

Risk Management. Without the specification of risk related predicates, MulVAL itself can be seen as a safeguard within the Risk Management context. MulVAL is a tool that can be used to assess the current risk level within an IT environment, precisely the type of safeguard that would be placed within this category. The primary domain of MulVAL, however, is the logical domain. Social (e.g. personnel) components of the model are minimally present and physical controls are entirely absent.

Personnel. A single predicate *incompetent* is used to identify users that may inadvertently participate in an attack path. Security awareness training will decrease the likelihood that a user will perform poor security practices. Therefore, the absence of this predicate is an indication of the implementation of personal security safeguards.

System Acquisition, Development, Operations and Maintenance. When good security practices are in place, no software vulnerabilities should be present in the IT environment. As such the absence of the *vulExists*/3 predicate confirms that these safeguards are in place and effective.

Identification & Authentication. The *hasAccount*/3 predicate links a principal, a user, to an electronic identity on a specific system. This implies that the user will perform an authentication function in order to bind to that identity, however, the authentication mechanism and process is not modeled in MulVAL currently. A userÕs group membership is mapped using the inGroup/2 predicate

Access Control. Of all the safeguard types, MulVAL models access control safeguards most thoroughly. At the system level, file permissions safeguards are modeled in terms of file user ownership $fileOwner/3$, file group ownership $fileGroupOwner/3$, and file attributes $fileAttr/11$.

Network access controls, from system to system over a specific port and protocol, are mapped through *hacl*/4. Similarly, as stated in the document MulVAL extensions, host network location is specified through $hostNet/2$ and routes through network devices though $routeEntry/5$.

5.3 Modeling Additional Safeguards

In preparation for a discussion on how safeguards can be modeled in MulVAL, Figure 7 has been updated to include some sample safeguards that could be included in each safeguard class.

Safeguard		Logical		
Type	Protection Mechanism	Application	System	Network
Security Policy		allow	dataBind	
Risk Management		MulVAL	MulVAL	MulVAL
System Acquisition, Development, Operations & Maintenance		vulExists	vulExists	
System Integrity & Security Configuration	Information Disclosure	Application hardening	**System Hardening**	Network Address Translation
	Input Validation	Application proxies	Stack monitoring	Intrusion Prevention Systems (malformed packets)
	DoS	Application throttling	TCP/IP stack security configuration	Intrusion Prevention Systems (flooding attacks)
Identification & Authentication	local ID	**cvssAccess cvssAuthentication**	hasAccount inGroup	
	Enterprise ID			Network level accounts (LDAP / NIS)
Access Control			fileOwner fileGroupOwner fileAttr	hacl hostNet routeEntry
Cryptographic Services		Object cryptography, Session Security	File system cryptography	**Communication cryptogrphy**
Audit & Accountability		Application logs	Critical file integrity checking, systems logs	IDS
Business Continuity Management		HA configuration	HA configuration RAID	Redundant paths

Figure 7: *Potential Safeguards to be Modeled*

Taking a specific safeguard class from this table, Identification and Authentication in the Logical-Application domain, a sample safeguard implemented in this class would be to enable Basic authentication on a web server in the target environment. The NVD database includes a classification attribute for vulnerabilities that indicates whether account access is required to exploit the vulnerability. In this scenario, a web server that requires authenticated account access would not be immediately vulnerable to attack if the authentication safeguard were included in the model. In short, the inclusion of this safeguard to the model eliminates a set of attack paths that would previously have been valid.

Primitives:

$$hasAccount(Principal, Host, Application, Account) \tag{17}$$

$$applicationAccount(Host, Application, Mechanism, Account) \tag{18}$$

Derived Rules: Attack requiring application authentication

$$\begin{aligned} &execCode(Attacker, Host, Perm) :- \\ &\quad malicious(Attacker), \\ &\quad vulExists(Host, Software, remoteExploit, privilegeEscalation, authRequired), \\ &\quad networkServiceInfo(Host, Software, Protocol, Port, Perm), \\ &\quad netAccess(Attacker, Host, Protocol, Port), \\ &\quad hasAccount(Attacker, Host, Software, Account), \\ &\quad applicationAccount(Host, Software, Mechanism, Account). \end{aligned} \tag{19}$$

As another example, the ability to infiltrate a system is a function of the degree that system has been hardened. Certain operating systems contain vulnerabilities that allow disruption to that systemÕs network service, so long as the attack can access the host. If, however, the system has been properly hardened, the network stack will have been configured to make DoS attacks less likely to succeed (e.g. close half open sessions). The addition of a safeguard class of the type System Integrity & Security Configuration (Denial of Service) at the Logical System domain will reflect this improved security posture at the system network stack.

Primitives:

$$systemHardened(Host, Level) \tag{20}$$

Derived Rules: A flooding attack

$$\begin{aligned} &canFlood(Attacker, Host, Level) :- \\ &\quad malicious(Attacker), \\ &\quad vulExists(Host, Software, remoteExploit, dos), \\ &\quad networkServiceInfo(Host, Software, Protocol, Port, Perm), \\ &\quad netAccess(Attacker, Host, Protocol, Port), \\ &\quad systemHardened(Host, Level). \end{aligned} \tag{21}$$

Additionally, communication security through cryptographic sessions would be an instantiation of a Communication Services safeguard at the Network domain. Attacks to sniff data off the wire would be thwarted if this safeguard were present. This last example is of particular interest since it brings forward an important fact. Whereas the previous 2 examples build upon the existing MulVAL concept of attacks being expressions of the exploitation of vulnerabilities, there is no corresponding vulnerability in MulVAL to describe information loss through eavesdropping [12].

Other safeguards categories can be modeled as well, however, the extent of the effort to include the safeguard and its supporting characteristics will vary. As an example of a complex safeguard modeling effort, we can examine the safeguards that would be deployed to audit the network, specifically, an Intrusion Detection System (IDS). The following is a list of issues that would have to be addressed in order to model this safeguard.

User intent. Currently malicious users are modeled as having a goal to utilize system vulnerabilities to violate the security policy. There is no indication within this model as to whether or not the user is concerned with being detected. In reality, however, the need to operate stealthily is a significant sub-goal of the majority of professional hackers, particularly in the military arena. When detected, the theft of tactical data will cause a disruption to operating since specific battle plans will have to be abandoned once their confidentiality is compromised. If undetected, however, the impact of the theft is magnified since a successful counter-intelligence effort can significantly impact military operations. Thus, modeling the stealth factor will need to be taken into account when reworking the MulVAL model to include an IDS.

Detection thresholds. With stealthiness as the userÕs intent, the associated vulnerability for the IT environment is that the attacker is not detected. It is not clear at this time what metric would be needed to measure how well hidden illicit traffic is within a normal network operations. A possible approach is to set thresholds for activity for the IDS safeguard and then model traffic needed to successfully execute an attack path.

Take, as an example, the following scenario.

1. A brute force application authentication attack requires 4 data exchanges for each attempt (get resource, received challenge, send credentials, get response)
2. There are 650,000 potential credentials to try with the expectation of success at the 50% attempted mark.
3. The IDS is configured to alert when more than 20 login attempts per second are attempted against the resource.

The attacker can expect success if they have 5 hours in which to spread out the attack and will not be detected since the attack averages only 18 attempts per second. Alternatively, if the attacker has only 1 hour in which to execute the attack, in order to avoid detection, the user will only be able to cover 20% of the credential space.

In short, the effort needed to enhance the MulVAL model to include the use of IDS and detection tools is substantial since new goals and vulnerabilities, not to mention the need for a mechanism for representing the stealth factor (e.g. time or throughput) would need to be added to the model.

5.4 Modeling Many Paths

The most important distinction between the current MulVAL model and the model that incorporates safeguards is the number of attack paths that will be generated. While some safeguards, most significantly the ones that are present in the current MulVAL model, express their effectiveness as a binary value. That is, if they are present they provide absolute protection against the attacks they are designed to mitigate. For example, the hacl rules that are expressed in the model are the only valid paths between systems between networks.

New safeguards that are to be introduced to the model will not have this absolute quality. In particular, the safeguards that present ÔresistanceÕ to attack, such as cryptography or system hardening will express their effectiveness, not as a binary value, but as some range value, indicating that the safeguard protects the data, but a determined and/or capable attacker will be able to by pass these safeguards.

In order to show this aspect of the model, that some attack paths are possible, but only after an amount of effort is made to bypass the safeguards, the model must be able to match on the successful attack conditions. Recalling that the Datalog nature of MulVAL will only show valid attack paths, the attacks that are mitigated through non-absolute safeguards must still be presented in terms of valid attack paths. This theme is revisited in the proceeding section on safeguard effectiveness, but it suffices to say at the moment that it is not possible to show partially effective safeguards unless the complete attack path can be matched in Datalog.

To illustrate this point, if we wanted to show a partially effective network access control safeguard, for example a firewall, it would be necessary to first identify the possible attack paths that could be reached if that safeguard were bypassed. Without this mechanism, it is not possible to see the effect on the environment if partially effective safeguards were bypassed. In order to assess the impact of a bypassed firewall safeguard, MulVAL must be able to determine what the hacl rules would be if the firewall was not enforcing any access controls.

The channeling, then, for modeling safeguards is that, prior to introducing these control mechanisms, it must first be possible to represent the full scope of the vulnerability if the safeguards were not present. Once this comprehensive impact is in the model, safeguards can be introduced to show how the risk posed by these vulnerabilities are mitigated. However, in terms of the MulVAL model itself, showing the result of no safeguards in the environment will significantly increase the number of attack paths that are possible. For example, if we want to model a compromised firewall, we must consider all possible connections that originate on one side of the firewall and terminate on the other side.

Some approaches to dealing with this potential explosion of possible attack paths are presented below.

Inside MulVAL

Predicates can be modeled to exclude a class of attack, unless specific conditions are present. For example, using encrypted network sessions, where the vulnerability related to information disclosure on the wire, this approach can be described as follows:

> Encrypted sessions are invulnerable unless the communication program or algorithm has been compromised. In this case, only communication sessions that use that protocol are considered suspect.

If we determine that an algorithm like AES-192 is vulnerable, for example, the algorithm is broken, we can add a $vulCrypto/1$ predicate to indicate this fact and state that: All paths that were previously protected by this algorithm should now show up as attack paths. Note that the amount of effort required to exploit a cryptographic vulnerability can be modeled within the $vulCrypto/1$ predicate in the form of a work factor value. This work factor can be used in an attack path ranking context to identity and flag those attack paths that utilize a compromised encryption algorithm.

However, if there is no $vulCrypto/1$ entry for a particular algorithm, the communication paths that are protected with that algorithm are considered safe and will not show up as attack paths where the communications path is compromised. Thus fewer attack paths will be presented by MulVAL. Note this is in sync with the MulVAL modeling approach where a vulnerability is needed to trigger an exploit and an entry on the attack path.

Inside Datalog Via Cuts

The cut predicate is similar to a switch/break construct in the C programming language. It tells Datalog that once it has satisfied one version of a predicate, it does not need to look for others that match the predicate. The cut predicate tells Datalog

not to pass back through this point when looking for alternative solutions. In other words, all choices that have been made up to the cut point are considered to be the only choices.

XSB supports cuts; but with some limitations. Specifically, XSB allows cuts to be set within, but not across, tabled predicates. As the XSB implementation builds table relations on demand, that is, one at a time, the effect of the cut to to prevent the creation of relations once a certain criteria has been satisfied. With fewer relations built, the model is simplified and there will computational savings gained. In short, If we can cut away more challenging safeguards in favour of easier ones, we eliminate less likely attack paths from the subsequent analysis.

As described in [16], the use of cuts remains controversial; just as with the C switch/case statement. Cuts are most frequently used as a model optimization technique, rather than a modeling tool [5]. Therefore, it is recommended that other methods for reducing the number of attack paths be examined more closely in the initial modeling stage for MulVAL safeguards and the use of the cut predicate can be reviewed as part of an optimization process.

Outside MulVAL

A final approach to dealing with the large number of attack paths that will be generated when partially effective safeguards are modeled is to allow Datalog to generate these paths, but then prune the attack tree based on the level of effort that would be required to successfully execute the attack. This will require a more detailed description of safeguards effectiveness, which is provided in the next section.

5.5 Safeguard Effectiveness

The Handbook of Computer Security defines security effectiveness in terms of the degree to which a safeguard may be characterized as effectively mitigating a vulnerability and reducing associated loss risks. This can be viewed in one of two ways:

- does the safeguard impede an attacker from exploiting the vulnerability; or
- does the safeguard reduce the exposure of the vulnerable portion of the attack surface (for example, a safeguard may completely block TCP exploits, but not UDP exploits of the same vulnerability).

This paper is primarily concerned with the first type of effectiveness, namely, safeguards as a function of making the environment resistant to attack. This resistance is seen as the reduction of the likelihood of a successful exploitation of a vulnerability, as the safeguard will:

[5] However, [16] provides a description of how to model negation using cuts.

- increase the needed tools/capabilities on the part of the attacker; and/or
- increase the work factor associated with executing an attack (for example, more time is required to effect an attack)

When viewed in this context, safeguards are seen as a mechanism by which attack paths can be reduced (or eliminated) from consideration as a source of risk to the environment. This is analogous to the risk management process of ensuring that the security protections that are in place in the IT environment reduce all threat scenarios to a low likelihood of occurrence. This is a direct interpretation of the attack path ranking problem discussed subsequently in this document, but it suffices to say that a high work factor may position the risk posed by the attack down below more promising (or threatening) attacks paths.

But what is this mechanism? The most convenient expression of this resistance is work factor, that is, the amount of effort that is needed to successfully exploit a vulnerability in spite of the safeguards that are in place to prevent this activity. This work factor is a cumulative property as each additional safeguard that is in place between the attack and the target along an attack path adds to the total work of executing the exploit. Therefore, each safeguard in the solution space will have an associated work factor.

The challenge remains to define and apply work factor in a consistent way for all safeguard throughout the modeled environment. Two different safeguard classes may not have enough common ground to establish a basis for comparing work factor values. Even within the same safeguard class, while some safeguards will lend themselves to an expression of effectiveness, the work factor for others will only be possible in terms of "better than" relationships.

For example, the hardening systems against attack is better than using the base system install security configuration. A range of work factor value is possible, as seen in the following table:

Table 7: *System Hardening Work Factor*

System Hardening State	Work Factor Value
No hardening (base system security installation)	0
Security configuration review by an IT administrator	1
Security configuration review by an security officer	2
Application of vendor supplied hardening scripts	3
Use of organizationally approved hardening scripts	4
NIST hardening procedures applied	5
Internal VA	6
External (3rd party) VA	7
External penetration test (more than 12 months ago)	8
External penetration test (less than 12 months ago)	9

Often, however, there will be jumps in the work factor value, for example, the use of evaluated products as a safeguard.

Table 8: *Evaluated Products Work Factor*

System Acquisition	Work Factor Value
General Purpose System and OS	0
Appliance Device	5
EAL 4 evaluated product	7
EAL 4+ evaluated product	9

Again, the following table shows the work factor value for cryptographic protections for information in motion. Under the GSP, CSE is responsible for provides cryptography processes for sensitive information. This table lists the approved cryptographic algorithms within the GoC and the associated cryptoperiod[6].

What is notable in this table is that it is possible to define different protection mechanisms that, while different in implementation, are equal in terms of work factor. In this case, CAST5-128, Triple-DES and AES-128 are all considered to have a work factor of 7 and are, therefore, equal in terms of the difficulty in breaking the algorithm and bypassing the safeguard. This demonstrates how the work factor values can be derived from external evaluation criteria. Where available these criteria will be valuable tools to create an accurate assessment of the work factor, however, in many cases the work factor values will only be determined through subjective opinion and experimentation.

[6]The *cryptoperiod* is the length of time that a cryptographic key may be used under some key management policy. The intent is to limit the amount of ciphertext encrypted under a single key that is available for cryptanalysis. Note that the $cryptoperiod << TimeToBreakKey$.

Table 9: *Cryptographic Algorithm Work Factor*

Encryption Algorithm	Key Size	Cryptoperiod (days)	Work Factor
None	0	0	0
SKIPJACK		1	4
CAST5	80	1	4
CAST5	128	7	7
Triple-DES	112	7	7
AES	128	7	7
AES	192	7	8
AES	256	7	9

The work factor values presented in these tables are provided as a demonstration of the intra-safeguards comparison approach only and do not reflect the true relative strength of each safeguard implementation.

5.6 Work Factor in the Model

The work factor value associated with safeguards must be present in the model. When attack paths are identified, the work factors associated with exploiting vulnerabilities or defeating safeguards can be extracted from MulVAL for inclusion in attack path ranking algorithms that determination of the actual level of effort for an attacker to pursue any particular attack path.

For example, if a cryptographic algorithm has a certain work factor, it may be desirable to express that work factor in actual terms of length of time required to compromise the data. This will be a function of many elements outside the model such as the speed and number of systems used to break the cryptographic protections.

Within MulVAL, the ability to assign an effectiveness factor to each safeguard allows the model to accurately reflect the security configuration of the IT environment. However, outside the model, the tools used to rank attack paths and visualize the security environment will not require this level of granularity. In actual fact, post-MulVAL analysis will act on the analysis results of the model as a whole (i.e. all detected attack paths), rather than an in depth examination of each particular safeguard. It would then be useful to be able to set a threshold when performing this analysis such that high work factor safeguards are eliminated from consideration (i.e. the likelihood of exploitation is low).

In order to be able to use a single threshold to drive this analysis, the work factor values across all safeguard must be harmonized. That is, two different safeguards that have the same work factor value should require roughly the same order of effort to

bypass. Using the previously described safeguard work factor values, we can compare these safeguards side-by-side, as seen in figure 8.

Figure 8: *Harmonized Work Factor*

If a risk analyst sets the minimum acceptable safeguard level to a work factor of 5, then any attack paths that are mitigated through the use of safeguards with a work factor of less than 5 are considered at risk and should be included in the risk analysis and attack path ranking. This would include:

- All general purpose hardware and operating systems;
- All systems that have had their security configuration set using the organization's hardening scripts (or less effective measures); and
- All network sessions that are protected using the SKIPJACK or 80-bit CAST algorithm.

The challenge of achieving harmonized work factor values across all safeguards is the difficulty in equating safeguards that capture divergent aspects of IT security. When trying to equate two safeguards, many issues, including the following, must be considered:

1. Is the type of work reflected in the work factor similar? For example, one safeguard may represent a high degree of human computer interaction, whereas other safeguards may be bypassed through purely computational effort, such as breaking a password file.
2. Is the work factor a single value characteristic (such as the evaluation level of a system) or a composite of characteristics, such as algorithmic complexity, where key size, algorithmic lineage and known weaknesses all may be used to assess the work factor for breaking a cryptographic algorithm.
3. Are there known methodologies for evaluating the effectiveness of both safeguards and are there common threads between these methodologies?

4. Are there security organizations that have performed research into methods for evaluating these safeguards and does this information, coming from a common source, use terms and methods that would allow safeguard effectiveness can be compared?

In the absence of any industry guidance for comparing safeguards, the risk analyst will be forced to establish these work factors as a manual practice. With expertise from the security field (e.g. professional penetration testers) and a period of calibration with results analysis, it is believed that the harmonization of safeguard work factors can be achieved.

6 Attack Path Ranking

A primary objective of Defence R&D Canada – Ottawa is to provide the ability to rank, or sort, MulVAL produced attack paths into a coherent set to determine suitable courses of action as noted in the J6 OODA loop identified within Figure 4.

6.1 Defining Attack Path Risk

As noted in [1], in order to rank MulVAL attack paths, we must have some indication of the risk associated with each path. With associated risk, one can sort the attack paths so that those that have the greatest potential risk can be deal with first as part of a risk management process. A noted in Figure 9 from [1]:

$$Risk = f(AttackLikelihood, Impact) \tag{22}$$

$$AttackLikelihood = f(SuccessLikelihood, ThreatAgentMotivated) \tag{23}$$

We are not modeling any aspect of Threat Agents within MulVAL, therefore, we cannot reason according to Equation 23. Therefore, we can only approximate risk by approximating the $AttackLikelihood$. Our risk approximation assumes that $ThreatAgentMotivated$ is always true, so Equation 23 becomes:

$$AttackLikelihood = f(SuccessLikelihood) \tag{24}$$

Similarly we are not modeling Threat Agent *capability* so our $SuccessLikelihood$ becomes a function of strictly the difficulty in exploiting vulnerabilities within the

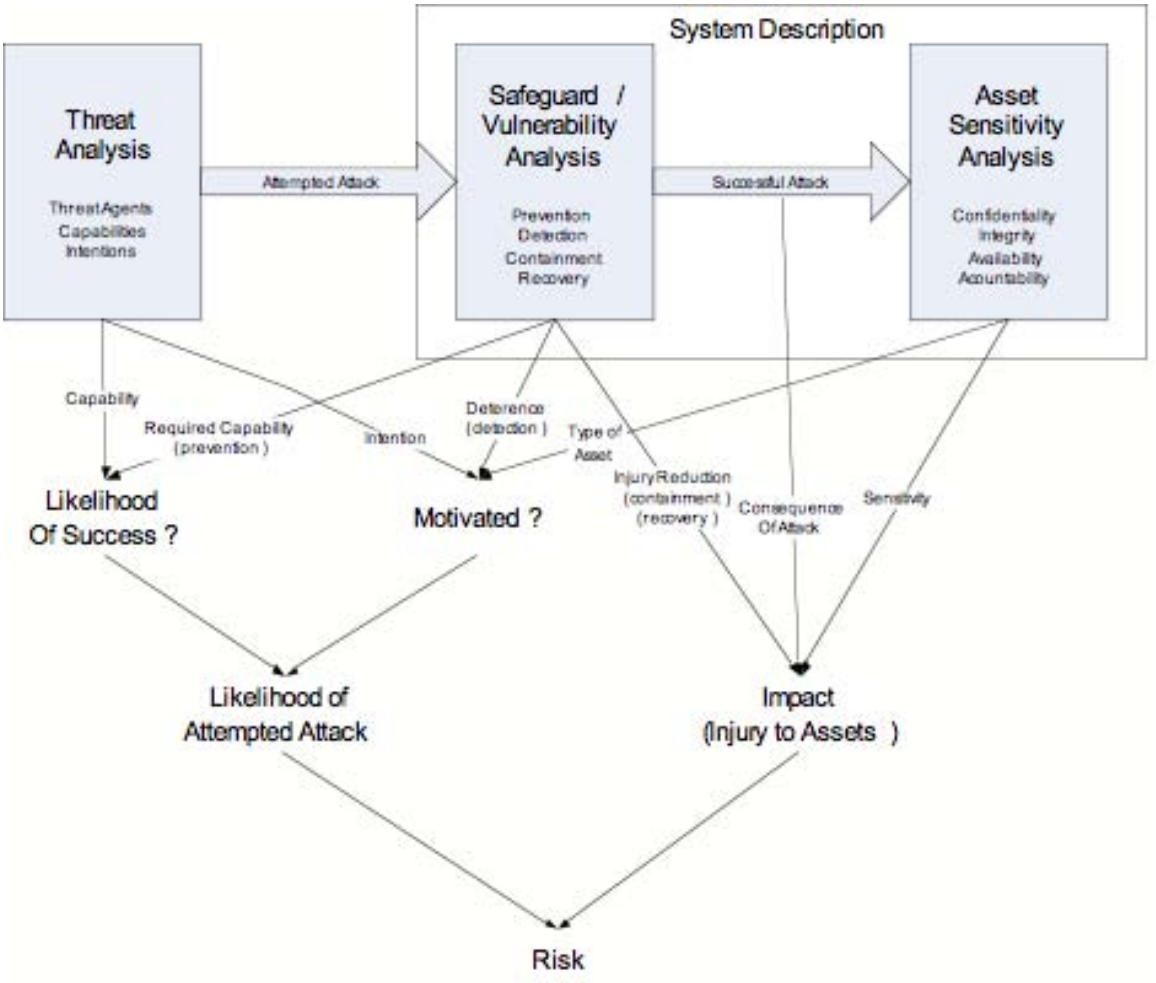

Figure 9: *A General Risk Management Model*

system or defeating preventative safeguards that we model. We call this $WorkFactor$ so Equation 24 becomes:

$$AttackLikelihood = f(WorkFactor) \tag{25}$$

And our risk Equation 22 becomes a risk approximation Equation:

$$Risk \approx f(WorkFactor, Impact) \tag{26}$$

As shown in Figure 9, $Impact$ is a function of injury reduction safeguards, consequences of an attack, and the sensitivity of the asset. However, we do not yet model injury reduction safeguards in MulVAL so $Impact$ is:

$$Impact = f(AttackConsequence, AssetValue) \tag{27}$$

Therefore, our risk approximation Equation 26 becomes:

$$Risk \approx f(WorkFactor, AttackConsequence, AssetValue) \tag{28}$$

In order to use our risk approximation in helping to rank MulVAL attack paths we need the following elements:

1. $AssetValue$ has been defined in Section 3 as a set of valued IT Service Definitions. This includes a set of hypothetical values for confidentiality (Table 1), integrity (Table 2), availability (Table 4), and MTTR (Table 5). These values are considered hypothetical since it is not yet clear how the values will interact with the mixing functions defined below to produce our risk approximation;
2. $AttackConsequence$ has been defined in [12] as those data elements in the National Vulnerability Database (NVD) which identify confidentiality, integrity, or availability impacts;
3. A $WorkFactor\ Function$ is needed to combine all of the $WorkFactor$ elements in a multi-stage attack path in Equation 28; and
4. An $Impact\ Function$ is needed to combine the $AttackConsequence$ and $AssetValue$ in Equation 27;
5. A $Risk\ Function$ is needed to combine $WorkFactor$, $AttackConsequences$ and $AssetValue$ in Equation 28.

It is not clear at this point in time, what the correct values are for $AssetValue$ or $AttackConsequence$. Nor is it clear what the correct $Risk\ Function$ should be. As a result, it will require a heuristic approach to try differing values and functions in order to see their affect on attack path ranking when compared to current best practices in risk management and common sense.

6.2 Fundamental Ranking Strategies

There are two fundamental strategies for assessing attack path ranking:

1. *Rank within MulVAL* – MulVAL currently only provides a binary indication of whether an attack path is possible within the context of attacks modeled. In this strategy the MulVAL model is extended to calculate our risk approximation by calculating an attack paths work factor as MulVAL is reasoning on the attack path; and

2. *Rank outside MulVAL* – In this strategy, MulVAL is used to reason on plausible attack paths with no attempt to rank the multiple outcomes. MulVAL will be required to provide attack path information to an external process which will perform the ranking.

Given the uncertainty in the $AssetValues$, $AttackConsequences$, and $Risk\ Function$, our strategic approach is to rank outside MulVAL. This provides us with several advantages at this stage in MulVAL model development:

1. Exporting all plausible attack paths from MulVAL ensures that attack paths deemed of low risk are not pruned within the model. In this way, all attack paths can be examined and their ranking scores to heuristically determine the validity of the $AssetValues$, $AttackConsequences$, and $Risk\ Function$;

2. There are other, more effective, computing languages for computing risk scores than Prolog;

3. Keeping the $AssetValues$, $AttackConsequences$, and $Risk\ Function$ outside MulVAL means that these can be changed with immediate feedback on the ranking without re-running MulVAL analyses;

4. We need a MulVAL controlling function. Although independent of the ranking strategy, it should be a minor extension to include the attack path ranking.

Note that Figure 4 already reflects the *rank outside MulVAL* strategy by showing a separate process to determine attack paths to a single target, and then a follow-on process to rank all attack paths for all targets.

The criteria for when attack path ranking needs to move from outside to inside MulVAL will be when the number of attack paths becomes too large. This will likely be exhibited by excessive MulVAL analysis times for large network topologies. At this point, moving the ranking into MulVAL will allow Prolog to optimize its analysis run time by pruning any attack paths that do not meet a minimum risk threshold. Note that it will likely be easier to validate the $AssetValues$, $AttackConsequences$, and $Risk\ Function$ outside of MulVAL. Therefore, having validated values and functions would be another criteria for moving the attack path ranking inside MulVAL.

6.3 Attack Path Approach

Figure 10 is based on Figure 4 and shows our attack path approach to define a *MulVAL controlling process* which will perform the following functions:

1. Query MulVAL for the list of $AssetValue$s within specified criteria (for example, all Secret data);

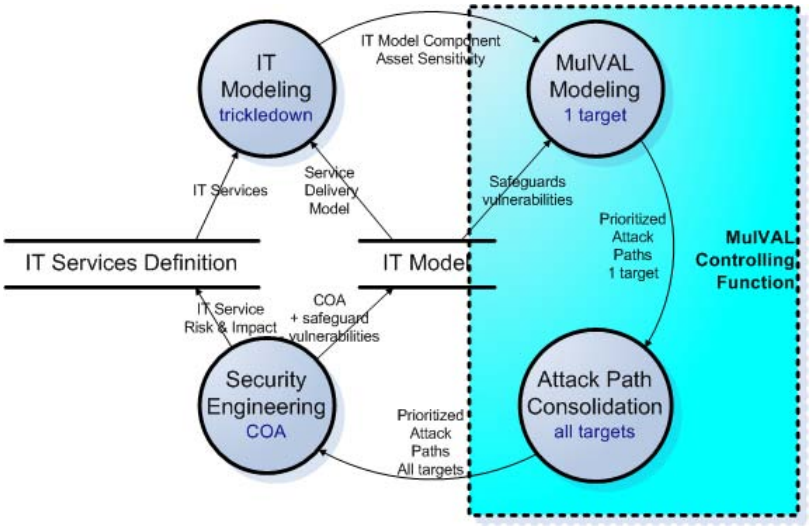

Figure 10: *MulVAL Controlling Function*

2. For each asset of value in the $AssetValue$ list, perform the following:
   (a) Execute a MulVAL query using $policyViolation(Attacker, Access, asset)$. This should derive all attack paths that could potentially impact confidentiality and availability on the specific *asset*,
   (b) Execute a MulVAL query using $serviceImpacted(Principal, Access, asset)$. This should derive a list of all *Principal*s who are potentially impacted by the lack of availability of the specified *asset*,
   (c) Extract the attack paths using the function defined in Section 6.4 for each of these two queries,
   (d) Correlate related NVD information from an NVD data store, and
   (e) Invoke the *Risk Function* algorithm on the resulting set of attack paths to derive a ranked set of attack paths for the specified *asset*. Note that this correlates to the *MulVAL Modeling – 1 target* process in Figure 4; and
3. Apply the *Risk Function* to the combined set of all attack paths on all targets to create a single combined attack path ranking. Note that this correlates to the *Attack Path Consolidation – all target* process in Figure 4.

Note that the two MulVAL queries listed $policyViolation/3$ and $serviceImpacted/3$ were identified in [12] as the appropriate queries to determine potential impact.

6.4 MulVAL Function to Extract Ranking Data

A new MulVAL function is need which will output the data required for ranking attack paths. This function will need to output the following information for each

attack step:

1. the specific vulnerability, if any, exploited during this attack step. Several reports [1, 5, 12] have identified NVD data parameters that could contribute to defining a work factor for a particular attack path step. All off these data parameters are indexed using the CVE Identifier[7];

2. the *Host*; and

3. the *Data* associated with the final step in the attack; and

4. the *consequence* associated with the final step in the attack. This will be used to determine what *AssetValue* should be applied in the *Risk Function*.

Simon Ou [3, 4, 17] has discussed the possibility of developing a generalized MulVAL XML output function, which could then be converted to different output formats using XSLT translations. This generalized output would replace the existing MulVAL *visualize* function with its *text* and *html* output formats, which would become XSLT translations. If such a function existed, then achieving the required output function would require the writing of an XSLT translation.

7 Strategy for Handling Inaccurate Data

The value of the information obtained through the analytic aspects of MulVAL and its ancillary modeling processes (e.g. attack path ranking) is directly related to the value of the data that is used to establish the model and perform the analysis. Where inaccurate data is used to create and populate the model constructs, the confidence level that the analysts have in the result will decrease. This becomes a significant issue when the analysis is used to drive COA activities since inaccurate analysis results may lead to ineffective, or possibly counterproductive, security actions that may, in fact, degrade the security posture presented by the IT environment.

In this context, inaccurate data can be thought of in three ways.

1. Incomplete data: elements in the actual environment that do not have an analogue in the model.

2. Uncertain data: elements in the model that have been added though some form of assumptions or guesswork, without the ability to confirm these data elements with the actual environment.

[7]Note that the NVD range and consequence (especially privilege escalation) are specifically modeled in MulVAL so that it can reason on multi-stage attacks.

3. Incorrect data: elements added to the model that are not in accord with the real world environment.

Note there are relationships between these categories that can redirect the nature of the inaccuracy, but will not effectively address the problem of inaccurate data in the model. For example, if it is not desirable to have an incomplete model, information can be added into the model to fill in gaps of missing information, but the confidence level in this assumption-based data will have a high degree of uncertainty. If it is not desired to have uncertainty in the model, the confidence level attributed to the assumption-based data can be raised, but this will lead to incorrect information and results.

Ultimately, there are two questions that must be addressed in order to eliminate inaccuracy in the model:

- Model validity: How well does the model represent the actual environment?
- Model verification: How well does the data in the model represent the actual environment?

Note that the dynamic risk model, that is MulVAL and its ancillary processes, has several independant sources of inaccuracy. Figure 11 identifies these sources of inaccuracy within the model.

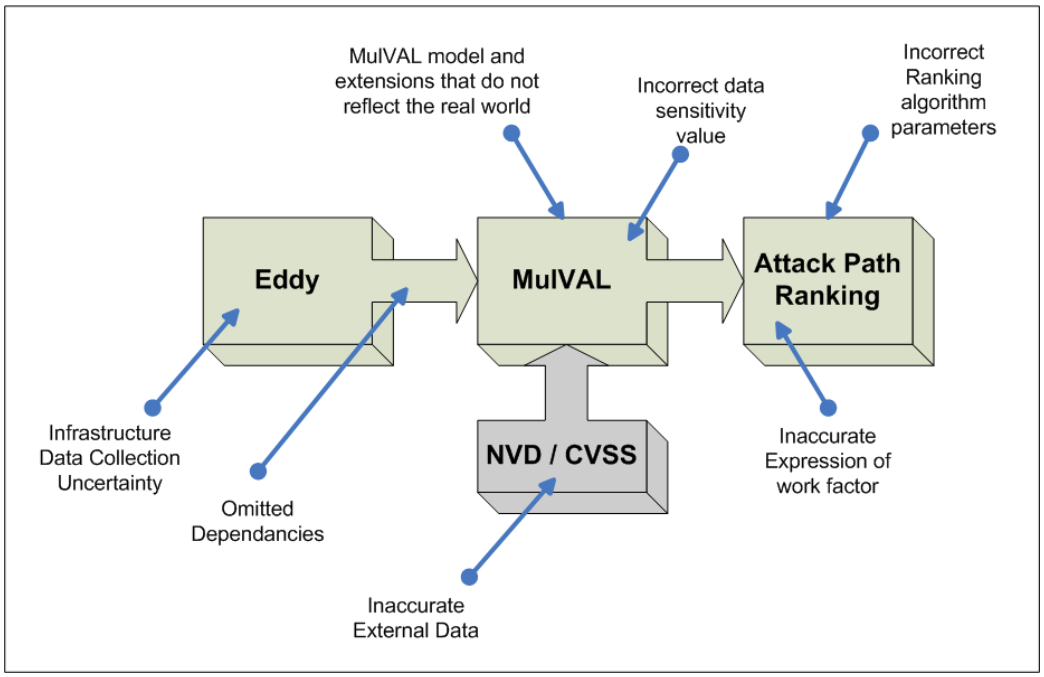

Figure 11: *Sources of Model Uncertainty*

The nature of the sources of inaccuracy differ greatly, and therefore, no one strategy for handling this inaccuracy can be used across the dynamic risk analysis problem space. Each source of inaccuracy must be examined separately.

7.1 Inaccuracy in Existing MulVAL Concepts

In the existing model, the effects of inaccuracy can be seen in the representation of the infrastructure, the model definition itself and the external information sources. Each potential source of inaccuracy is examined in the following sections.

7.1.1 IT Infrastructure

So long as the MulVAL model constructs are considered to be accurate, the challenge with modeling the IT infrastructure is to ensure that the data that is populated in the model is correct. The question was raised as an issue in the DAP prototyping effort [18]. If we have complete information about our own infrastructure, then it is possible to map this data into the model. The use of tools that have a full range of visibility in the environment (e.g. OVAL scanners, tools that can extract and process firewall rules) can greatly minimize the amount of guesswork needed to fully populate the model.

Where uncertainty must be reflected in the model, predicates can be extended to include a confidence value associated with the Datalog fact.

$$networkService(Host, Software, Protocol, Port, Permission, Confidence) \quad (29)$$

For example, an OVAL scanner will generate a fact as follows:

$$networkService(system1, apache, tcp, 80, www, 100) \quad (30)$$

But if we detect this information from the network (for example, via a Remote Vulnerability Scan (RVS)), we have to guess at the software and permission values. Therefore, our ÔconfidenceÕ in the fact is reduced:

$$networkService(system1, apache, tcp, 80, www, 75) \quad (31)$$

The manner by which we set the confidence value will likely have subjective and objective elements. For example, there will be a base level of confidence for each Datalog fact, given that some elements of each fact will can be ascertained with high confidence. In the example given above, host, protocol and port can be determined through remote scanning. When, then a remote service is found, and immediate base fact can be set indicating this information:

$$networkService(system1, ?, tcp, 80, ?, 50) \tag{32}$$

How the confidence level is allocated based on information obtained through guesswork will be a more subjective analysis, including such issues as:

1. How are similar systems configured? If the IT administrator tends to run all services under application accounts, the risk analyst would tend to assume that is what has been done to this system as well. This would reflect the fact that good security practices are being followed in the environment.

2. Is there corroborating evidence? Can we obtain a banner from the service to determine the software version?

3. Do all assumed facts contribute equally to the setting of the confidence level for the Datalog fact as a whole?

This approach to modeling uncertainly will require effort to fine-tune these parameters to achieve an accurate reflection of how uncertainty impacts the model.

Note that this approach places the uncertainty value in the fact itself. This is due to the fact that we will want MulVAL to export confidence levels associated with a given attack path to aid in the ranking effort. This may not be an absolute requirement since the attack path ranking process, if not performed through MulVAL, could extract this information from a separate data store. If information about the IT environment is placed in a database, MulVAL predicates could be built from a subset of this data to product the attack paths. The attack path ranking process could then return to the database to extract the needed confidence data for the ranking function.

Note that this idea of returning to the database to retrieve extended attributes of a particular Datalog fact could also be applied to the work factor values.

One advantage of using a database to store confidence values is that it would be easier to assign separate confidence value to each parameter in the Datalog facts. In the example above, we could have a confidence value for the software being used and a confidence value for the permissions value. This would give more granularity in the

uncertainly associated with a particular fact which would be of additional benefit to the ranking and exploration functions in the dynamic risk model.

The use of confidence values in the ranking process will depend on the intent of the analysis. One view would be that attack paths that we are certain of are of prime concern since they represent a clear danger to the environment. An alternate view is that attack paths with a high uncertainly are of more concern since it is more difficult to assess the impact of a successful exploitation of the attack. The main factor that will determine which analysis approach is more important to an analyst is how conservative the assumptions are with respect to 'guessed' information. For example, if the analyst assumes a worst-case scenario (e.g. that a vulnerability is exploitable as root), then the uncertainty in the data can only reduce the impact of the exploit.

7.1.2 The MulVAL Model Itself

Examining the MulVAL model itself, that is, how the model represents objects and interconnections in an IT environment, the question becomes: does this model represent reality? There is a high confidence that what is modeled in MulVAL represents an accurate reflection of the real world. However, due to the fact that MulVAL is an abstraction, there is a limit to the granularity of information that MulVAL can currently process. This leaves some reasoning abilities beyond of the scope of the model.

As an example, we present the currently unmodeled concept of software service dependencies, or alternatively, software composition. This is a common application configuration seen in web and multi-tier applications. A web server can be seen as providing a network service to deliver static HTML data. An equally valid view of web services is that they provide an environment into which web applications can be deployed. For example, a web server can support collaboration tools, like MediaWIKI or act as a front-end for database applications. Where vulnerabilities are found in the web server software, MulVAL will identify potential attack paths where these vulnerabilities are exploited. The challenge, however, is that web application vulnerabilities are also present in the NVD and there is no mechanism by which these vulnerabilities can be modeled in the current MulVAL model.

As an example, the following MulVAL rule shows a privilege escalation attack through a software vulnerability:

$$
\begin{aligned}
&execCode(Attacker, H, Perm) :- \\
&\quad malicious(Attacker), \\
&\quad vulExists(H, Software, remoteExploit, privEscalation), \\
&\quad networkService(H, Software, Protocol, Port, Perm), \\
&\quad netAccess(Attacker, H, Protocol, Port).
\end{aligned} \tag{33}
$$

To detect the presence of an exploitable vulnerability, MulVAL matches the software hosted as a network service to the software identified in the NVD. When a vulnerability is present in a software component, that is, software that is not directly hosted as a network service but is hosted by another network service, MulVAL will not detect the presence of this vulnerability in the environment. The link between the network service program and the vulnerable program cannot be made.

This is an example of incompleteness in the MulVAL model that can lead to an inaccurate analysis. Specifically, there is a category of software vulnerabilities that:

1. cannot be modeled;
2. may, in fact, be present in the environment ; and
3. may lead to attack paths that could compromise data and services in the environment.

These attack paths will, therefore, not be found in the current model.

It is expected that, as the MulVAL model is further refined, it will more completely cover the range of possible attack paths and more accurately represent real world environments.

For example, to extend the MulVAL model to include vulnerabilities in software service dependencies, we must first define a predicate to show the dependency. There is an existing *dependsOn*/3 predicate that could be used for this purpose, however, since this predicate is used to model library dependencies, the link is from a program to a required library. We then suggest a new predicate: *reliesOn*/3 to reverse this relationship to show that a subordinate software package requires a hosting or data service. For example:

$$
reliesOn(system, Software, Service) \tag{34}
$$

This predicate could be instantiated with the following facts:

$$\begin{array}{l} reliesOn(system, mediawiki, apache) \\ reliesOn(system, mediawiki, mysql) \end{array} \tag{35}$$

These rules would demonstrate that, for the given system, the MediaWIKI software package is present and relies upon Apache and MySQL to deliver its data services. Expressing this fact as part of a privilege escalation attack, we can derive the following rule:

$$\begin{array}{l} execCode(Attacker, Host, User) :- \\ \quad malicious(Attacker), \\ \quad reliesOn(Host, Software, Service), \\ \quad vulExists(Host, Software, remoteExploit, privilegeEscalation), \\ \quad networkService(Host, Service, Protocol, Port, User), \\ \quad netAccess(Attacker, Host, Protocol, Port). \end{array} \tag{36}$$

For example,

$$\begin{array}{l} execCode(Attacker, wikihost, \_user) :- \\ \quad malicious(Attacker), \\ \quad reliesOn(wikihost, mediawiki, apache), \\ \quad vulExists(wikihost, mediawiki, remoteExploit, privilegeEscalation), \\ \quad networkService(wikihost, apache, http, 80, www), \\ \quad netAccess(Attacker, wikihost, http, 80). \end{array} \tag{37}$$

It is significant to note that the establishment of this new category of software dependency attacks would require the automated discovery and population of these dependencies into the $reliesOn/3$ predicate. While these extensions to MulVAL may make the model more accurate in representing the types of attacks that are possible, if the data in the model is not properly populated we have moved the source of inaccuracy from the model itself to the data in the model.

7.1.3 External Sources

Refer to the research project [12] for a discussion of the ICAT/NVD methods of modeling attack characteristics and impact and how these models apply to the MulVAL interpretation of impact.

Note that the data feeds provided by this publicly available vulnerability database are outside the control of MulVAL analysts. That is, other than ensuring that the most up to date feed is used to evaluate software vulnerabilities, the accuracy of the data from the source is solely under the control of the NVD team. Greater accuracy may be possible through combining information from multiple vulnerability databases, for example, the Open Source Vulnerability Database (OSVDB). Combining information from multiple sources can ensure that the information used by MulVAL is more complete and correct and, therefore, more accurate.

7.2 Inaccuracy in Proposed MulVAL Concepts

In the proposed model extensions, as specified in [12] and this document, the effects of of inaccuracy can be seen in the representation of asset value of data, the dependancy model, work factor for safeguards and the attack path ranking algorithm. Each potential source of inaccuracy is examined in the following sections.

7.2.1 Data Asset Value

Data asset valuation modeling was presented as an extension to the MulVAL model in [5]. The use of existing GoC mechanisms for classifying data confidentiality and integrity for data elements and availability for information services would seem to be the most appropriate language for the representation of data asset value in the MulVAL model. Data asset management, however, remains a complex problem in the security field as it is very challenging to ensure that C/I/A values for data are assigned appropriately and that the data resides in protected processing zones appropriate for the level of sensitivity for the data.

Within the MulVAL modeling environment, there is then an added layer of uncertainty since the real world sensitivity level for data assets must be correctly mapped to the model. It is not clear, at this time, what tools are available that can be used to automatically synchronize the sensitivity level of data assets in the model with their counterparts in the real world. An investigation into the current methods for automatically assessing data sensitivity is recommended with a view to how this information can be used by the MulVAL data acquisition process.

7.2.2 Dependency Modeling

The mechanism by which inter-dependencies within an IT environment are presented in the model has its basis in well known/documented relationships. For example the manner in which DND maps missions to services, the definition of common service architectures and tools to define software dependencies are well known entities. The paper has presented a mechanism by which trickle-down impact is a function of these

dependencies. As this mechanism is further researched it will become more clear as to how well this dependency modeling reflect reality. This may only be achieved through interviews with DND strategic, tactical and operations staff.

The danger of inaccurate dependancy modeling is that it will not be possible to trickle down the effect of a particular exploit to all data elements that are, in actuality, affected by the C/I/A impact of the exploit. Similarly, it will not be possible to identify all critical components on which data or services depend in the IT environment context.

7.2.3 Work Factor

As described previously in this document, the ranking of attacks paths is a method through which a risk analyst can determine the relative importance of attack paths. This method of comparing two attack paths to determine which one represents a more pressing risk to the environment is based most significantly on the consequences of a successful instantiation of the attack (i.e. the impact) and the likelihood of the attack. Likelihood has been previous described in terms of the amount of effort that is required to stage, launch and successfully execute the attack. This amount of effort has been presented as the work factor. This work factor, which has been presented as an attribute or characteristic of the safeguards in the system, must be determined for each stage of the attack in order to determine the relative importance of the attack from the perspective of presenting risk to the environment.

To attach a work factor to a safeguard, the analyst must fist determine the best criteria to use to measure this work factor. For some safeguards this will be a simple process since the criterion that is used is an intrinsic property of the safeguard. For example, the work factor associated with system hardening lends itself to a simple expression of *this system is more hardened that that one*, given the type of security configuration enhancements that have been used to secure the system. In other cases, it will be more difficult to select a single criterion to use to express the work factor value. Cryptographic protections, for example, can use a set of criteria to set the work factor, including:

- the key size;
- the status of the algorithm's approval status by security organizations;
- the presence of known algorithmic weaknesses; and
- the amount of industry vetting the algorithm has undergone.

For this type of safeguard, the only way to express work factor is to synthesize these factors into a general statement of relative work factor. Security organizations that

make algorithm robustness a study perform this synthesis when they set recommended security practices for the use of cryptographic protections. This reliance on security organizations for best practices associated with the use of safeguards may ultimately be the best option for selecting the work factor value.

Once the criterion, or aggregated criteria, is chosen, the assignment of the specific work factor values can be done. Once again, this will either be set as an intrinsic property of the safeguard, or will have to be set using expertise from the IT security industry. The concern with inaccurately setting the work factor criteria or value is that potential attacks that present a more significant danger to the organization may not be ranked sufficiently high to catch the attention of the IT risk analysts. Alternatively, less critical safeguards may consume time and effort where the attention to security could be more beneficially spent elsewhere. In short, inaccurate work factor values will incorrectly drive COA options.

7.2.4 Attack Path Ranking

Similarly, the algorithms that have been established to rank the relative importance of attack paths (from a threat perspective) may not be produce the correct real-world ranking of the actual risk to the organization. This may be due to the fact that not all factors that influence the ranking have been considered or have not been assigned the correct weights. Again, the risk to the organization is that an incorrect ranked list of attack paths will incorrectly direct COA actions.

Unfortunately, as there is no industry practices that can aid in modeling this ranking algorithm, the only option available to the risk analyst to calibrate the algorithm is to use trial scenarios and then evaluate the results in consultation with experienced penetration testing experts working in this field.

8 Future Research

Given the proposed MulVAL extensions to model *exposed critical resources*, the main thrust of any future work should be the implementation of an experimental model with an aim to:

1. validate the proposed model extensions;
2. develop further extensions (such as, additional safeguards); and
3. to start heuristic validation of the proposed values and functions in approximating risk so that attack paths can be ranked.

In addition, the following specific research topics were identified:

1. Define the required QoP over time for IT Service Definitions. This touches on a larger issue of trying to address temporal issues within the MulVAL model. Other temporal concepts might be: time to execute high work factor attacks, such as brute force key searches or password cracking;

2. Define an IT Service relative to other IT Services in other network missions;

3. Define an IT Service relative to environmental events. This touches on a larger issue of trying to address rapid OODA cycles with MulVAL as the orientation model. Rapid OODA cycles could be based on IT Service Definition changes, environmental events, missions changes, IT sensor feeds, etc.;

4. Consider the work of Ensel [8] in automated dependency model building using neural networks to determine if this technique is suitable for integration with MulVAL; and

5. Consider the work of Keller et al [11] in CIM performance aggregation to determine if better impact aggregation functions can be developed for MulVAL, or whether this work provides insights into the inverse of IT Service Level decomposition onto ITI elements.

References

[1] Henderson, Glen, Bacic, Eugen, and Froh, Michael (2005), Dynamic Asset Protection & Risk Management Abstraction Study, (DRDC Ottawa CR 2005-205) Cinnabar, Inc., Ottawa, Ontario.

[2] Lefebvre, Julie H., Gregoire, Marc, Beaudoin, Luc, and Froh, Michael (2005), Computer Network Defence Situational Awareness: Information Requirements, (DRDC Ottawa TM 2005-254) Defence R&D Canada – Ottawa.

[3] Ou, Xinming, Govindavajhala, Sudhakar, and Appel, Andrew W. (2005), MulVAL: A Logic-based Network Security Analyzer, In *Proceedings of the 14th USENIX Security Symposium*, pp. 113–128, Baltimore, Maryland.

[4] Ou, Xinming (2005), A Logic-Programming Approach to Network Security Analysis, Ph.D. thesis, Princeton University.

[5] Bacic, Eugen, Froh, Michael, and Henderson, Glen (2006), MulVAL Extensions for Dynamic Asset Protection, (DRDC Ottawa CR 2006-251) BSSI Inc., Ottawa, Ontario.

[6] Treasuary Board Secretariat (2002), Government Security Policy (online), http://www.tbs-sct.gc.ca/pubs_pol/gospubs/tbm_12a/gsp-psg_e.asp (Access Date: Oct 2006).

[7] Hussain, Iftekhar (2004), Fault-Tolerant IP and MPLS Networks, Ch. I-1: Understanding High Availability of IP and MPLS Networks, Cisco Press.

[8] Ensel, Christian (2001), A Scalable Approach to Automated Service Dependency Modeling in Heterogeneous Environments, In *EDOC '01: Proceedings of the 5th IEEE International Conference on Enterprise Distributed Object Computing*, p. 128, Washington, DC, USA: IEEE Computer Society.

[9] Distributed Management Task Force, Common Information Management (CIM) Tutorial (online), http://www.wbemsolutions.com/tutorials/CIM/cimtutorial.pdf (Access Date: October 2006).

[10] Keller, Alexander and Ensel, Christian (2002), An Approach for Managing Service Dependencies with XML and the Resource Description Framework, *Journal of Network and Systems Management*, 10(2), 147–170.

[11] Keller, Alexander, Benke, Oliver, Debusmann, Markus, Koppel, Andreas, Kreger, Heather, Maier, Andreas, and Schopmeyer, Karl (2004), The CIM Metrics Model: Introducing flexible data collection and aggregation for

performance management in CIM, *IEEE Transactions on Network and Service Management*, 1(2), 59–71.

[12] Froh, Michael, Henderson, Glen, Sawilla, Reg, and Burrell, Craig (in preparation), MulVAL Requirements on NVD, (DRDC Ottawa CR 2009-xxx) BSSI Inc., Ottawa, Ontario.

[13] Ross, Ron, Katzke, Stu, Johnson, Arnold, Swanson, Marianne, Stoneburner, Gary, Rogers, George, and Lee, Annabelle (2005), Recommended Security Controls for Federal Information Systems.

[14] Treasury Board Secretariat (2005), Management of Information Technology Security – Operational Security Standard.

[15] JTC 1/SC 27, ISO/IEC 17799: Code of practice for information security management (online), http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=39612 (Access Date: May 2009).

[16] Sagonas, Konstantinos, Swift, Terrance, and Warren, David S. (1994), XSB as an efficient deductive database engine, *SIGMOD Record*, 23(2), 442–453.

[17] Ou, Xinming, Boyer, Wayne F., and McQueen, Miles A. (2006), A Scalable Approach to Attack Graph Generation, In *Proceedings of 13th ACM Conference on Computer and Communications Security (CCS 2006)*, Alexandria, Virginia, USA: Association of Computing Machinery.

[18] Henderson, Glen and Tremblay, Larry (2006), Dynamic Asset Protection – Prototype Software Development Report, (DRDC Ottawa CR 2006-247) Cinnabar Inc., Ottawa, Ontario. Distribution limited to Department of National Defence, Canadian Forces and their qualified defence contractors.

This page intentionally left blank.

Annex A: Safeguard Type Definitions

This section takes the safeguard types that were presented in the *Modeling Safeguards* section and provides a description of the intent and role of these safeguards in an IT environment. This information is derived from a synthesis of concepts taken from the following three best practices sources:

1. NIST Special Publication 800-53 (Recommended Security Controls for Federal Information Systems)
2. ISO/IEC 17799 (Code of practice for Information Security Management)
3. Treasury Board Management of Information Technology Security (MITS) Operational Security Standard.

Security Policy: In its general sense, a security policy is a statement produced by senior personnel that dictates the role that security will play within an organization. At this level, security policies often deal with the legal, regulatory, and liability issue that have a relevancy in the IT environment. High-level security policies are often drilled down to more specific system-based (or issue-based) policy statements of what is allowed and what is against policy in terms of user access to services and data. As a high-level directive, the security policy depends upon more detailed security practice document to indicate how the policy is to be enforced.

Organizational Security: This category takes an organizational view of the security practices to ensure that there is a culture of security present in the organization. This security culture must be supported by management and distributed throughout the organization to ensure that roles, responsibilities, accountability, processes, and expertise are all present and effective to support the security policy.

Security Risk Management: There must be an ongoing effort to monitor the risk level within the organization, given that many of the variables in the risk management equation will change over time. These variable will include: changes to data sensitivity, discovery of new vulnerabilities in the organizationÕs security posture, and general changes to the organizationÕs IT infrastructure.

Personnel Security: In spite of advances in the area of IT security and risk management, people remain the single largest source of risk to an organization. Threat scenarios presented in a risk management context usually take one of one of two forms: malicious internal users that frequently have significant privileges with in the IT environment and loyal users that participate in questionable practices due to lack of knowledge of security issues.

Physical & Environment Security: This is the traditional view of security, that is, locked doors and armed guards used to protect assets. Physical controls are almost always considered frequently considered their own class of control.

System Acquisition, Development, Operations and Maintenance: What controls are placed over the selection or development of a new hardware/software component, prior to the introduction of that component to the IT environment. For example, are evaluated products required for use in the processing facility? What development methodologies are used for the development of new software practices? Are there separate development / testing / production environments in place to allow for integration testing and impact analysis? Is there a patch management process in place?

System Integrity and Security Configuration: How are components configured so as to ensure that the integrity of the environment is maintained? This is the primary area of technical controls that will prevent systems from being vulnerable to attack. Penetration testers will examine many areas of the security configuration (network, system and application) to discover poor configuration and detected vulnerabilities. Similarly, hacking methodologies usually lead a prospective attacker through a set of activities to walk through the security configuration and leverage information found at each level, including system enumeration, footprinting, detecting and exploiting vulnerabilities, gaining privileged access and, ultimately, compromising system integrity through rootkits and Trojans.

Identification and Authentication: All safeguards related to the questions of: attributing an electronic identity to a specific individual and verifying the electronic identity of a user on the system, would fall into this category.

Access Control: All safeguards related to restricting access to data and services to a specific set of users would fall into this category. This would include the use of access control models and techniques, such as MAC/DAC and RBAC. Within this paper, the access control category is limited to the technical safeguards; however, in practice, access control can be executed through physical and administrative controls as well.

Cryptographic Services: All safeguards related to the use of cryptographic protections would fall into this category. These controls provide protection, as data exists in various states within the environment. That is, data will exist, at different times, within an object store, on a file system or in transit within a network and cryptographic controls can be used at each point along the data path to ensure data protection. This category can include all infrastructure component needed to make cryptographic services work, for example PKI components.

Audit & Accountability: All safeguards used to track activity within an IT environment would be considered auditing controls. Auditing is used to detect activity that is not in compliance with established policies and identify suspicious behavior. Again, auditing safeguards are technical controls that serve to identify violations of the security policy and hold users accountable.

Incident Management: Incident management is the performance of reactive and proactive services to help prevent and handle computer security events and incidents. It includes identifying and minimizing the impact of technical vulnerabilities in software or hardware that may expose computing infrastructures to attacks or compromise, thereby causing incidents.

Business Continuity Management: All safeguards related to the continuation or restoration of service in the face of network, system or application outages.

Compliance: Safeguards imposed due by legal or regulatory agencies or achieved though security programs provided by regulatory agencies.

This page intentionally left blank.

Annex B: MulVAL Extensions

This appendix contains a concise description of the extensions proposed in this report to the current MulVAL model described in Appendix C. This appendix will only list changed predicates. The reader is referred to Appendix C for the remaining MulVAL model predicate descriptions. This appendix is organized into separate sections for each model change. Each section contains a description of the changes made and any changed primitive and derived predicates. None of the changed predicates have been modeled in XSB yet.

B.1 Modeling IT Services

Section 3 proposed one new primitive predicate $itServiceDefinition/6$, which defines the asset value of data provided by a high-level IT Service.

Primitive Predicate Changes

```
/****** Define IT Services ******/
itServiceDefinition(Data, ConfValue, IntValue, AvailValue, MttrValue, Bias).
```

Derived Predicate Changes

No derived predicates were changed or defined.

B.2 Modeling Infrastructure

Section 4 proposed several new predicates in order to define a simple dependency model:

1. A new primitive predicate $servicebinding/4$, which is a manually generated mapping of IT Service to network service;
2. A new derived predicate $dataBind/5$, which replaces the existing $dataBind/3$ predicate with derived data values;
3. A new derived predicate $networkService/7$, which augments the existing primitive $networkService/5$ found by the OVAL scanners. The new derived predicate adds availability value to the network service;
4. A new derived predicate $libraryAvailValue/3$, which adds availability value to libraries supporting valued network services;

5. Two new derived predicates for *hostSensitivity*/5, which defines the C/I/A value of client and server hosts; and

6. A new derived predicate for *netSensitivity*/5, which defines the C/I/A value of a network.

Primitive Predicate Changes

```
serviceBinding(Service, Host, Program, Account).
```

Derived Predicate Changes

```
/****** dataBind/5 Derivation ******/
dataBind(Service, Host, Path, C, I) :-
    itServiceDefinition(Service, C, I, A, MTTR, Bias),
    serviceBinding(Service, Host, Program, Account),
    networkService(Host, Program, _protocol, _port, Account),
    accessFile(Account, Host, _access, Path).

/****** networkService/7 Derivation ******/
networkService(Host, Program, Protocol, Port, Priv, A, MTTR) :-
    itServiceDefinition(Service, C, I, A, MTTR, Bias),
    networkService(Host, Program, Protocol, Port, Priv).

/****** Derive Library Avail Value ******/
libraryAvailValue(Library, A, MTTR):-
    networkService(Host, Program, _protocol, _port, _priv, A, MTTR),
    dependsOn(Host, Program, Library).

/****** Derive Host Sensitivity For Servers ******/
hostSensitivity(ServerHost, C, I, A, MTTR) :-
    itServiceDefinition(Data, C, I, A, MTTR, _bias),
    serviceBinding(Data, ServerHost, _program, _account).

/****** Derive Host Sensitivity For Clients ******/
hostSensitivity(ClientHost, C, I, A, MTTR) :-
    itServiceDefinition(Data, C, I, A, MTTR, _bias),
    access(Principal, _access, Data),
    hasAccount(Principal, ClientHost, _account).

/****** Derive Net Sensitivity ******/
netSensitivity(Net, C, I, A, MTTR) :-
    hostSensitivity(Host, C, I, A, MTTR),
    hostNet(Host, Net).
\end{predicate}
```

B.3 Modeling Safeguards

Section 5 proposed two new predicates that demonstrate modeling safeguards and safeguard effectiveness:

1. A new primitive predicate *systemHardened*/2, which describes the level of hardening of a system (that is, a safeguard effectiveness); and
2. A new derived predicate *canFlood*/3, which uses the safeguard effectivess to derive the difficulty in successfully flooding a targeted host.

Primitive Predicate Changes

```
systemHardened(Host, Level)
```

Derived Predicate Changes

```
canFlood(Attacker, Host, Level) :-
    malicious(Attacker),
    vulExists(Host, Software, remoteExploit, dos),
    networkServiceInfo(Host, Software, Protocol, Port, Perm),
    netAccess(Attacker, Host, Protocol, Port),
    systemHardened(Host, Level).
```

B.4 Modeling Uncertainty

Section 7 proposed several new predicates in order to handle a class of application not covered in the existing model. LAMP (Linux, Apache, MySQL, and Perl — PHP — Python) is a term used to describe web-based applications which are readily available. The NVD database is starting to see a large number of vulnerabilities listed against LAMP open-source projects described as products.

1. A new primitive predicate *reliesOn*/3, which defines is similar to the existing primitive *dependsOn*/3, but conveys a peer-to-peer application software relationship such as used in LAMP applications. It may be possible to automatically populate some *reliesOn*/3 data using package manager dependency information; and
2. A new derivation of the existing *execCode*/3, which finds remote escalation attack steps when a LAMP software *reliesOn*/3 a network service accessible via a network.

Primitive Predicate Changes

```
reliesOn(system, Software, Service)
```

Derived Predicate Changes

```
execCode(Attacker, Host, User) :-
    malicious(Attacker),
    reliesOn(Host, Software, Service),
    vulExists(Host, Software, remoteExploit, privilegeEscalation),
    networkService(Host, Service, Protocol, Port, User),
    netAccess(Attacker, Host, Protocol, Port).
```

Annex C: Current MulVAL Model

The current MulVAL model is taken from Ou [4] with an extended network model from [5]. The network extension generates `hacl()` rules from a number of new primitives. The following MulVAL model changes were made:

1. a new `hostNet/2` primitive predicate was added to describe which networks hosts were attached to;
2. a new `routeEntry/5` primitive predicate was added to describe the authorized routing table entries in firewalls and filtering routers;
3. a new `route/4` derived predicate was added to calculate routes within the network; and
4. the `hacl/4` primitive predicate was changed from a primitive to a derived predicate, and derived from valid routes.

C.1 Primitive Predicates for Unix-family Platform

```
/****** Section Base MulVAL *******/
vulExists(Host, ID, Program).
vulProperty(ID, ExploitRange, ExploitConsequence).
bugHyp(Host, Program, Range, Consequence).
dependsOn(Host, Program, Library).
networkService(Host, Program, Protocol, Port, User).
setuidProgram(Host, Program, Perm).
clientProgram(Host, Program).
malicious(Principal).
incompetent(Principal).
fileAttr(Host, Path, R1, W1, X1, R2, W2, X2, R3, W3, X3).
fileOwner(Host, Path, User).
fileGroupOwner(Host, Path, Group).
inGroup(User, Group).
nfsExportInfo(Server, Path, Client, Access, RootSquash, Security).
nfsUserMap(UserClient, UserServer, RootSquash).
allow(Principal, Access, Data).
hasAccount(Principal, Host, Account).
located(Principal, Zone).
dataBind(Data, Host, Path).


/****** Section Network Extension *******/
hostNet(Host, Network).
routeEntry(Router, InitNet, DestNet, Protocol, Port).
```

C.2 Interaction Rules for Unix-family Platform

Each interaction rule is introduced by an explain rule clause. The first argument of explain rule is a plain-text explanation of the interaction rule. The second argument is the rule itself. There are also fact clauses, and they are introduced by the load clause statement. The interaction clauses will be loaded dynamically into the Prolog database so that the meta-interpreter introduced in Appendix B can use them.

```
/****** Section Network Routing *******/

explain_rule(
    Õdirect route between subnets through an intermediate routerÕ,
    (route(InitNet, DestNet, Protocol, Port) :-
        routeEntry(Router, InitNet, DestNet, Protocol, Port),
        hostNet(Router, InitiNet),
        hostNet(Router, DestNet))
    ).

explain_rule(
    Õtransitive routing through an intermediate networkÕ,
    (route(InitNet, DestNet, Protocol, Port) :-
        route(InitNet, TransitNet, Protocol, Port)
        route(TransitNet, DestNet, Protocol, Port))
    ).

/****** HACL Section *******/

explain_rule(
    'Hosts can only communicate between networks through a valid route',
    (hacl(InitHost, TargetHost, Protocol, Port) :-
        hostNet(InitHost, InitNet),
        hostNet(TargetHost, TargetNet),
        InitNet \= TargetNet,
        route(InitNet, TargetNet, Protocol, Port)),
    ).

explain_rule(
    'hosts on same network have no communication restrictions',
    (hacl(InitHost, TargetHost, _, _) :-
        hostNet(InitHost, CommonNet),
        hostNet(TargetHost, CommonNet)),
    ).


/****** Section execCode *******/

explain_rule(
    Õaccount user can execute arbitrary codeÕ,
    (execCode(P, H, Perm) :-
```

```
        hasAccount(P, H, Perm))
    ).

explain_rule(
    ÕWhen a principal is compromised, any machine he has
    an account on will also be compromisedÕ,
    (execCode(P1, Host, Perm) :-
        principalCompromised(P2, P1),
        hasAccount(P2, Host, Perm),
        canAccessHost(P1, Host))
    ).

explain_rule(
    Õlocal exploitÕ,
    (execCode(P, Host, Perm) :-
        malicious(P),
        execCode(P, Host, Perm2),
        vulExists(Host, Software, localExploit, privEscalation),
        setuidProgram(Host, Software, Perm))
    ).

explain_rule(
    Õremote exploit of a server programÕ,
    (execCode(Attacker, H, Perm) :-
        malicious(Attacker),
        vulExists(H, Software, remoteExploit, privEscalation),
        networkService(H, Software, Protocol, Port, Perm),
        netAccess(Attacker, _, H, Protocol, Port))
    ).

explain_rule(
    Õremote exploit for a client programÕ,
    (execCode(Attacker, H, Perm) :-
        malicious(Attacker),
        vulExists(H, Software, remoteExploit, privEscalation),
        clientProgram(H, Software),
        incompetent(P),
        hasAccount(P, H, Perm))
    ).

explain_rule(
    ÕTrojan horse installationÕ,
    (execCode(Attacker, H, root) :-
        malicious(Attacker),
        accessFile(Attacker, H, write, Path))
    ).


/******** Section netAccess ********/

explain_rule(
```

```
    Õmulti-hop accessÕ,
    (netAccess(P, H2, Protocol, Port) :-
        execCode(P, H1, Perm), /* Any permission level */
        hacl(H1, H2, Protocol, Port))
    ).

explain_rule(
    Õdirect accessÕ,
    (netAccess(P, H, Protocol, Port) :-
        located(P, Zone),
        hacl(Zone, H, Protocol, Port))
    ).


/****** Section canAccessHost ******/

explain_rule(
    Õdirect access to hostsÕ,
    (canAccessHost(P, H) :-
        execCode(P, H, Perm))
    ).

explain_rule(
    Õaccess a host through a log in serviceÕ,
    (canAccessHost(P, H) :-
        logInService(H, Protocol, Port),
        netAccess(P, _, H, Protocol, Port))
    ).


/******** Section accessFile ********/

explain_rule(
    ÕexecCode implies file accessÕ,
    (accessFile(P, H, Access, Path) :-
        execCode(P, H, Usr),
        localFileProtection(H, Usr, Access, Path))
    ).


/****** Section principalCompromised ******/

explain_rule(
    Õpassword sniffingÕ,
    (principalCompromised(Victim, Attacker) :-
        hasAccount(Victim, H, Perm),
        execCode(Attacker, H, root),
        malicious(Attacker))
    ).

explain_rule(
```

```
    Õincompetent userÕ,
    (principalCompromised(P1, P2) :-
        incompetent(P1),
        malicious(P2))
    ).


/***************** Section ssh *********************/

explain_rule(
    Õssh is a log in serviceÕ,
    (logInService(H, Protocol, Port) :-
        networkService(H, sshd, Protocol, Port, _))
    ).


/*************** Section nfs ****************/
/* Principal P can access files on a NFS server if the files
   on the server are mounted at a client and he can access the
   files on the client side */
explain_rule(
    ÕNFS semanticsÕ,
    (accessFile(P, Server, Access, ServerPath) :-
        nfsMounted(Client, ClientPath, Server, ServerPath, Access),
        accessFile(P, Client, Access, ClientPath))
    ).

/* Principal P can access files on a NFS client if the files
   on the server are mounted at the client and he can access the
   files on the server side */
explain_rule(
    ÕNFS semanticsÕ,
    (accessFile(P, Client, Access, ClientPath) :-
        nfsMounted(Client, ClientPath, Server, ServerPath, read),
        accessFile(P, Server, Access, ServerPath))
    ).

explain_rule(
    ÕNFS shellÕ,
    (accessFile(P, Server, Access, Path) :-
        malicious(P),
        netAccess(P, Client, Server, rpc, 100003),
        nfsExportInfo(Server, Path, Access, Client).
    ).


/***************** Section misc ******************/

explain_rule(
    Õroot has arbitrary accessÕ,
    (localFileProtection(H, root, Access, Path))
```

```
    ).

/* Kernel is both a network service and a setuid program */
   load_clause(setuidProgram(Host, kernel, root)).
   load_clause(networkService(Host, kernel, _, _, root)).
explain_rule(
    ÕScanner reports security bugÕ,
    (vulExists(H, ID, Sw, Range, Consequence):-
        vulExists(H, ID, Sw),
        vulProperty(ID, Range, Consequence))
    ).

explain_rule(

    ÕIntroducing hypothetical bugÕ,
    (vulExists(H, ID, Sw, Range, Consequence):-
        bugHyp(H, Sw, Range, Consequence))
    ).

explain_rule(
    ÕLibrary bugÕ,
    (vulExists(H, ID, Sw, Range, Consequence):-
        vulExists(H, ID, Library, Range, Consequence),
        dependsOn(H, Sw, Library))
    ).

explain_rule(
    'data access by principals',
    (access(P, Access, Data) :-
        dataBind(Data, H, Path),
        accessFile(P, H, Access, Path))
    ).

explain_rule(
    'policy violation',
    (policyViolation(P, Access, Data) :-
        access(P, Access, Data),
        not allow(P, Access, Data))
    ).
```

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when document is classified)

1. ORIGINATOR (The name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Centre sponsoring a contractor's report, or tasking agency, are entered in section 8.)

Michael Froh
Bell Security Solutions Inc.

2. SECURITY CLASSIFICATION (Overall security classification of the document including special warning terms if applicable.)

UNCLASSIFIED

3. TITLE (The complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S, C or U) in parentheses after the title.)

MulVAL extensions II

4. AUTHORS (Last name, followed by initials – ranks, titles, etc. not to be used.)

Froh, M.; Henderson, G.

5. DATE OF PUBLICATION (Month and year of publication of document.)	6a. NO. OF PAGES (Total containing information. Include Annexes, Appendices, etc.)	6b. NO. OF REFS (Total cited in document.)
August 2009	86	18

7. DESCRIPTIVE NOTES (The category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.)

Contract Report

8. SPONSORING ACTIVITY (The name of the department project office or laboratory sponsoring the research and development – include address.)

Defence R&D Canada – Ottawa
3701 Carling Avenue, Ottawa ON K1A 0Z4, Canada

9a. PROJECT NO. (The applicable research and development project number under which the document was written. Please specify whether project or grant.)	9b. GRANT OR CONTRACT NO. (If appropriate, the applicable number under which the document was written.)
15bo03	W7714-6-3311

10a. ORIGINATOR'S DOCUMENT NUMBER (The official document number by which the document is identified by the originating activity. This number must be unique to this document.)	10b. OTHER DOCUMENT NO(s). (Any other numbers which may be assigned this document either by the originator or by the sponsor.)
DRDC Ottawa CR 2009-132	

11. DOCUMENT AVAILABILITY (Any limitations on further dissemination of the document, other than those imposed by security classification.)

(X) Unlimited distribution
( ) Defence departments and defence contractors; further distribution only as approved
( ) Defence departments and Canadian defence contractors; further distribution only as approved
( ) Government departments and agencies; further distribution only as approved
( ) Defence departments; further distribution only as approved
( ) Other (please specify):

12. DOCUMENT ANNOUNCEMENT (Any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in (11)) is possible, a wider announcement audience may be selected.)

Unlimited

13. ABSTRACT (A brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual.)

MulVAL: A Logic-based Network Security Analyzer is a general networked system tool that reasons on multi-stage, multi-host attack paths. This work proposes extensions to MulVAL that are needed in order to have MulVAL reason on *exposed critical resources* in support of computer network defence situational awareness. Extensions are proposed to model high-level mission centric IT Services. These services are then mapped onto MulVAL using a simple dependency model. The concept of safeguard effectiveness and safeguard vulnerability is introduced, which will model the additional work effort required for an attacker to circumvent the safeguard. A proposed risk approximation for MulVAL is defined as a function of the variables *WorkFactor*, *AttackConsequence*, and *AssetValue*. This function and its values will need to be developed heuristically. The calculation of risk will allow the ranking of MulVAL generated attack paths to provide better reasoning on exposed critical resources.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (Technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus. e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

network security
MulVAL
attack path
computer network defence
CND
situational awareness

Defence R&D Canada

Canada's leader in Defence
and National Security
Science and Technology

R & D pour la défense Canada

Chef de file au Canada en matière
de science et de technologie pour
la défense et la sécurité nationale

www.drdc-rddc.gc.ca