Measuring Inconsistency in Network Intrusion Rules

Kevin McAreavey, Weiru Liu, Paul Miller
School of Electronics, Electrical Engineering and Computer Science
Queen's University Belfast
Belfast, BT7 1NN, Northern Ireland
{kmcareavey01, w.liu, p.miller}@qub.ac.uk

***Abstract*—In this preliminary case study, we investigate how inconsistency in a network intrusion detection rule set can be measured. To achieve this, we first examine the structure of these rules which incorporate regular expression (Regex) pattern matching. We then identify primitive elements in these rules in order to translate the rules into their (equivalent) logical forms and to establish connections between them. Additional rules from background knowledge are also introduced to make the correlations among rules more explicit. Finally, we measure the degree of inconsistency in formulae of such a rule set (using the Scoring function, Shapley inconsistency values and Blame measure for prioritized knowledge) and compare the informativeness of these measures. We conclude that such measures are useful for the network intrusion domain assuming that incorporating domain knowledge for correlation of rules is feasible.**

***Index Terms*—Network intrusion detection, inconsistency measures.**

I. Introduction

Research into intrusion detection within the network security domain has accumulated a considerable amount of rules for detecting various intrusions and security threats. For instance, the size of Snort (a well known open source Network Intrusion Detection System (NIDS) and Network Intrusion Prevention System (NIPS)) has expanded rapidly in recent years [1]. Such intrusion detection systems (IDSs) have provided advantages over traditional security measures in terms of accuracy and reliability [2]. Furthermore, efficient implementation of the rules used by these IDSs has proven that it is possible to integrate these systems into modern high speed networks. One such method has been to execute regular expression (Regex) pattern matching on network traffic using high performance computing facilities.

However, the current IDS rules have their limitations as identified in [3]. The most significant limitation is that for every rule there is one alert, and there is little consideration for subsequent correlation of individually detected primitive alerts. Therefore high level network intrusion events are not adequately detected and reported. At the same time new rules are continuously being added, in Snort for example [1], so it is crucial to validate any new rules against existing rules to identify and remove inconsistencies.

In this paper, we do not consider how network intrusion rules are obtained or whether each rule is correct. Rather, we assume that given a set of such rules (called Snort2Regex rules) created using the Snort2Regex translator [4], our task is to deploy formal AI approaches for analyzing these rules and to discover any inconsistencies (the occurrence of conflicting information) among them. Such attempts for inconsistency detection have not been carried out within intrusion detection research. The implication for intrusion detection systems is that the detection of intrusion attempts will not be reliable when conflicting alerts are reported. By measuring the degree of inconsistency for each formula we can identify those formulae which are more inconsistent than others in order to begin resolving conflicts.

Recent developments in inconsistency handling [5], [6], [7] have demonstrated that to effectively handle inconsistencies, it is necessary to propose measures to quantify the degree of inconsistency that individual formulae have contributed, instead of considering the inconsistency of the knowledge base as a whole. These formula-level inconsistency values, such as the Scoring function in [6] and the I_{MI} Shapley value in [7], can be considered a degree of blame assigned to each formula in terms of its contribution to the inconsistency of the overall base. The advantage of these methods is that they identify an ordering for the degree of inconsistency that each formula brings. This means that inconsistency can be resolved by removing or modifying formulae which contribute more to inconsistency.

In many real word applications, such as in Requirements Engineering (RE) [8], some knowledge will be more important than others. Rather than assuming that all formulae are equally important, knowledge of this nature is often organized into some kind of prioritized or stratified form to differentiate more important vs. less important knowledge. Accordingly inconsistency measures for flat knowledge bases need to be adapted for prioritized bases. The approach described in [9] demonstrates the significance of applying inconsistency measures to prioritized knowledge bases where the advantage of this approach is that the priority of a formula can be considered when measuring inconsistency.

In this preliminary study, we examine the structure of rules in the Snort2Regex rule set and then break them down into their elementary units in order to reflect their primary structure. We then apply some existing approaches to measuring inconsistency for rules in the Snort2Regex format. This study reveals that the I_{LP_m} Shapley value is most useful for non-prioritized knowledge while the $Blame_v$ prioritized measure potentially provides more useful paths for resolving inconsistency. The investigation will lead to proposals as to how inconsistencies can be most effectively resolved.

II. Preliminaries

Let $\mathcal{L}$ denote the propositional language obtained from a finite set of propositional atoms $\mathcal{P} = \{\alpha, \beta, \gamma, ...\}$, using logical connectives $\{\vee, \wedge, \neg, \rightarrow\}$. An interpretation w is a total function from $\mathcal{P}$ to $\{0, 1\}$ and let $\mathcal{W}$ denote the set of all interpretations. Interpretation w is a model of formula a, denoted $w \models a$, iff a is true in the usual truth-functional way. We denote set inclusion (resp. strict) by $\subseteq$ (resp. $\subset$). Let $\perp$ denote an inconsistent formula.

Definition 2.1 ([6]) Let $\mathcal{D}$ be the set of databases formed from $\mathcal{L}$, where $\mathcal{D} = \wp(\mathcal{L})$. Let $\mathbb{N}$ be the set of natural numbers. For $n \in \mathbb{N}$, $\mathcal{D}^n$ is the set of databases of size n where $\mathcal{D}^n = \{\Gamma \in \mathcal{D} \mid |\Gamma| = n\}$

Definition 2.2 (MI and MC [6]) Let $\Delta \in \mathcal{D}$, $Con(\Delta) = \{\Gamma \subseteq \Delta \mid \Gamma \nvdash \perp\}$, and $Incon(\Delta) = \{\Gamma \subseteq \Delta \mid \Gamma \vdash \perp\}$.

$MC(\Delta) = \{\Phi \in Con(\Delta) \mid \forall \Psi \in Con(\Delta), \Phi \not\subset \Psi\}$

$MI(\Delta) = \{\Phi \in Incon(\Delta) \mid \forall \Psi \in Incon(\Delta), \Psi \not\subset \Phi\}$

$MI(\Delta)$ is the set of minimal inconsistent subsets (MISs) of Δ, and $MC(\Delta)$ the set of maximal consistent subsets of Δ.

In logics, inconsistency is the occurrence of conflicting information. e.g. both α and $\neg\alpha$ can be proven. However, for the rule set we are considering, we need to determine if inconsistent information will ever be produced, e.g. $\{\alpha \rightarrow \beta, \beta \rightarrow \neg\alpha\}$ is classically consistent but if we later learn $\{\alpha\}$ then $\{\alpha, \alpha \rightarrow \beta, \beta \rightarrow \neg\alpha\}$ is classically inconsistent.

Definition 2.3: A knowledge base K is called preemptively-inconsistent if $K \nvdash \perp$ and $\exists \Theta \subset \mathcal{L}$, s.t. $K \cup \Theta \vdash \perp$ and $K \cup \Theta' \nvdash \perp$ where $\forall \Theta' \subset \Theta$. $MPS(K|\Theta) = \{\Phi \setminus \Theta | \Phi \in MI(K \cup \Theta)\}$. $MPS(K|\Theta)$ is the set of minimal preemptively-inconsistent subsets of K w.r.t. Θ.

Θ can be taken as a kind of constraint for a given K. In the application we consider, Θ is in fact a subset of atoms which is a very small proportion of alerts generated, e.g. the most important alerts. $MPS(K|\Theta)$ is Θ sensitive, since given a K, there may be more than one Θ satisfying Def 3.2. Due to space limitation, we do not consider how this can be addressed [10], instead, we are interested in K as long as there exist a Θ making $K \cup \Theta$ preemptively-inconsistent.

III. Network Intrusion Rules

We currently have 8500 rules from [4], which have been translated into Snort2Regex format using the Snort2Regex translator. Regex pattern matching is then executed using high-speed hardware for intrusion detection on raw packet data. Rules in the Snort2Regex format are difficult to read and analyze in terms of inconsistency so we first translate them into a logical format and then measure their inconsistency.

A. Logical representation of rules

Definition 3.1: Let K be a unique rule identifier, m and n integers, A a set of regular expressions for packet header matching, B a set of regular expressions for packet content matching, γ a message describing the consequence of the rule and L a reference for identifying a source. A rule Π in Snort2Regex format is defined as:

$$\Pi = \left\{ \begin{array}{c} K \\ headers : m \\ A = \{A_1, \ldots, A_m\} \\ content : n \\ B = \{B_1, \ldots, B_n\} \\ msg : \gamma \\ \{reference : L;\} \end{array} \right\}$$

Snort2Regex rule Π says that from any of the given headers coupled with all the contents, we can infer the message. Π can be translated into a unique equivalent propositional formula denoted as $\mathcal{P}(\Pi) = (\alpha_1 \vee \ldots \vee \alpha_m) \wedge (\beta_1 \wedge \ldots \wedge \beta_n) \rightarrow \gamma$, where each regular expression in A (resp. B) is represented by an atom α_i (resp. β_i).

Example 3.1 Given two intrusion detection rules $\mathbb{A}$ and $\mathbb{B}$ of form Π

$$\mathbb{A} = \left\{ \begin{array}{c} 2 \\ \textit{headers: 1} \\ \textit{\^.\{1\}.\{1\}.\{2\}.\{2\}.\{2\}.\{1\}} \\ \textit{\x02.\{2\}.\{4\}.\{4\}.\{2\}.\{2\}} \\ \textit{content: 0} \\ \textit{msg:"EXPLOIT IGMP IGAP} \\ \textbf{account overflow attempt"} \\ \textit{reference:bugtraq,9952;} \\ \textit{reference:cve,2004-0176;} \\ \textit{reference:cve,2004-0367} \end{array} \right\}$$

$$\mathbb{B} = \left\{ \begin{array}{c} 3 \\ \textit{headers: 1} \\ \textit{\^.\{1\}.\{1\}.\{2\}.\{2\}.\{2\}.\{1\}} \\ \textit{\x02.\{2\}.\{4\}.\{4\}.\{2\}.\{2\}} \\ \textit{content: 0} \\ \textit{msg:"EXPLOIT IGMP IGAP} \\ \textbf{message overflow attempt"} \\ \textit{reference:bugtraq,9952;} \\ \textit{reference:cve,2004-0176;} \\ \textit{reference:cve,2004-0367} \end{array} \right\}$$

for an IGMP/IGAP account overflow attempt exploit.

Both rules contain one regular expression for packet header matching but no regular expressions for packet content matching; they produce different messages for intrusion alerts; and originate from the same sources, i.e. bugtraq,9952, cve,2004-0176 and cve,2004-0367.

Let A_i (resp. B_i) be a unique regular expression, i.e.

\^.{1}.{1}.{2}.{2}.{2}.{1}\x02.{2}.{4}.{4}.{2}.{2}

from $\mathbb{A}$ (resp. $\mathbb{B}$) matched to packet contents. Let ξ be the message "EXPLOIT IGMP IGAP account overflow attempt" from $\mathbb{A}$ and let χ be the message "EXPLOIT IGMP IGAP message overflow attempt" from $\mathbb{B}$, then we have logical rules $\mathcal{P}(\mathbb{A}) = \alpha_i \rightarrow \xi$ and $\mathcal{P}(\mathbb{B}) = \alpha_i \rightarrow \chi$. □

B. Extending the rule set

Snort2Regex treats $\mathbb{A}$ and $\mathbb{B}$ independent from each other, suggesting

ξ = "EXPLOIT IGMP IGAP account overflow attempt"

χ = *"EXPLOIT IGMP IGAP message overflow attempt"* have no connection. However there are obvious connections, e.g. both refer to the IGMP protocol, EXPLOIT attempts, etc, which are ignored by Snort2Regex. To establish connections among rules, we examine the structure of these rules and establish connections where possible for inconsistency analysis.

Definition 3.2 Let Π *be a rule in Snort2Regex format and* γ *be its message. Assume that* γ *can be expressed as* $\gamma = \{\gamma_1, ..., \gamma_t\}$ *where each* γ_i *is a meaningful expression for intrusion detection, and is treated as an atom, then* γ *can be expressed as a conjunctive form of atoms obtained from* γ_i*, denoted as* $\mathcal{P}(\gamma) = \gamma_1 \wedge ... \wedge \gamma_t$.

For instance, given message ξ in $\mathbb{A}$ where
$\xi = \{\mathsf{EXPLOIT}, \mathsf{IGMP}, \mathsf{IGAP}, \mathsf{account}, \mathsf{overflowAttempt}\}$,
its logical form is $\mathcal{P}(\xi) = \phi \wedge \nu \wedge \tau \wedge \rho \wedge \sigma$, where ϕ denotes atom EXPLOIT, and υ denotes IGMP, etc. Similarly, the logical form for χ in $\mathbb{B}$ is $\mathcal{P}(\chi) = \phi \wedge \nu \wedge \tau \wedge \pi \wedge \sigma$. Therefore ξ and χ share some common atoms, e.g., EXPLOIT, etc.

The importance of this transformation is that it allows correlations between rules to be identified and, along with underlying knowledge constraints, makes it easier to maintain the rule set by identifying inconsistencies.

Example 3.2 Given a new Snort2Regex rule suggesting $\mathsf{IGAP} \rightarrow \neg\mathsf{EXPLOIT}$ denoted as $\tau \rightarrow \neg\phi$ then we have a new rule set $K = \{\xi \wedge \chi \rightarrow \phi \wedge \nu \wedge \tau \wedge \rho \wedge \pi \wedge \sigma, \tau \rightarrow \neg\phi\}$ where $MPS(K|\{\xi, \chi\}) = \{K\}$.

The preemptive-inconsistency in K occurs because we know the IGAP variant will not be involved in EXPLOIT attempts but we have previous IGAP related EXPLOIT rules. The main advantage of identifying formulae at this elementary level is that it allows connections among rules to be identified.

IV. Approaches to Inconsistency Detection

Some more explanation about the motivation for Definition 2.3 is in order. Although a subset of rules K translated from rules in the SnortsRegex format may be consistent in the sense of classical logic, when an event (denoted as α) is reported, $K \cup \{\alpha\}$ could be inconsistent. There are potentially millions of events which may be generated by these Snort2Regex rules. However, if do not detect inconsistencies in the rules until events are detected then actual intrusion detection will be unreliable. Therefore, subsets of rules for detecting intrusion attempts should be consistent given any eventuality.

Consider this simple example: we have a blacklist of IP addresses from which exploit attempts are known to originate; we have a rule stating that if the source address is blacklisted then an exploit attempt is detected; we have another rule stating that if the packet uses the IGMP protocol and the source address is blacklisted then an exploit attempt is not detected. These rules are classically consistent but if we later receive an IGMP packet originating from a blacklisted IP address then they will become classically inconsistent. However this inconsistency detection can only be achieved at runtime, and since we need to ensure that inconsistencies will never occur, preemptive-inconsistency is needed.

Let us look at this example, given $K_1 = \{\alpha \rightarrow \beta, \beta \rightarrow \neg\alpha\}$ then $K_1 \cup \{\alpha\}$ is classically inconsistent. However we cannot simply add formula to create an inconsistency and apply inconsistency measures in the traditional way to the result (such as $K_1 \cup \{\alpha\}$) since there are potentially a number of variations which could create an inconsistency. Given $K_2 = \{\alpha \vee \beta \wedge \gamma \rightarrow \delta, \delta \rightarrow \neg\alpha \wedge \neg\beta\}$ there are two alternatives which create an inconsistency, i.e. $K_2 \cup \{\alpha\}$ and $K_2 \cup \{\beta, \gamma\}$ where $|K_2 \cup \{\alpha\}| = 3$ and $|K_2 \cup \{\beta, \gamma\}| = 4$. These will produce different inconsistency measures because the number of formulae in the inconsistency will vary. Therefore we consider K_2 as a preemptively-inconsistent subset where $|K_2| = 2$. We are only interested in K_2, not $K_2 \cup \{\alpha\}$ or $K_2 \cup \{\beta, \gamma\}$.

Example 4.1 Regular expressions in Snort2Regex rules are implemented as a whole in a high-performance environment, individual primitive elements are not considered at the regular expression or message level. However, individual primitive elements in regular expressions and in messages do have delicate connections where domain experts have this extra knowledge to make the links between what is being detected and how it is being detected.

Let E be the rule set K from Example 3.2 with this domain knowledge applied to atoms in the set:

$$E = \left\{ \begin{array}{c} \alpha \rightarrow \nu, \\ \beta \rightarrow \rho \wedge \sigma, \\ \gamma \rightarrow \pi \wedge \sigma, \\ \nu \wedge \rho \wedge \neg\pi \wedge \sigma \rightarrow \phi, \\ \nu \wedge \psi \rightarrow \neg\delta \end{array} \right\}$$

Let Z and H be rule sets built from the same language:

$$Z = \left\{ \begin{array}{c} \alpha \rightarrow \neg\phi, \\ \psi \rightarrow \delta \wedge \sigma, \\ \phi \rightarrow \neg\alpha, \\ \mu \vee \tau \rightarrow \theta, \\ \neg\omega \rightarrow \phi \end{array} \right\} \quad H = \left\{ \begin{array}{c} \sigma \rightarrow \neg\beta \wedge \neg\gamma, \\ \alpha \rightarrow \phi, \\ \mu \rightarrow \neg\epsilon, \\ \kappa \rightarrow \lambda \end{array} \right\}$$

Given the rule set $\Gamma = E \cup Z \cup H$ (where $|\Gamma| = 14$) and using preemptive inconsistency, the MPSs of Γ are:

$$\begin{array}{l} MPS(\Gamma|\{\alpha, \rho, \neg\pi, \sigma\}) = \\ \quad \{\{\alpha \rightarrow \nu, \nu \wedge \rho \wedge \neg\pi \wedge \sigma \rightarrow \phi, \alpha \rightarrow \neg\phi\} = M_1, \\ \quad \{\alpha \rightarrow \nu, \nu \wedge \rho \wedge \neg\pi \wedge \sigma \rightarrow \phi, \phi \rightarrow \neg\alpha\} = M_3\} \\ MPS(\Gamma|\{\nu, \psi\}) = \{\nu \wedge \psi \rightarrow \neg\delta, \psi \rightarrow \delta \wedge \sigma\} = M_2 \\ MPS(\Gamma|\{\beta\}) = \{\beta \rightarrow \rho \wedge \sigma, \sigma \rightarrow \neg\beta \wedge \neg\gamma\} = M_4, \\ MPS(\Gamma|\{\gamma\}) = \{\gamma \rightarrow \pi \wedge \sigma, \sigma \rightarrow \neg\beta \wedge \neg\gamma\} = M_5 \\ MPS(\Gamma|\{\alpha\}) = \{\{\phi \rightarrow \neg\alpha, \alpha \rightarrow \phi\} = M_6, \\ \quad \{\alpha \rightarrow \neg\phi, \alpha \rightarrow \phi\} = M_7\} \end{array}$$

A. Scoring function

As described in [6] the scoring function assigns a value to each subset in terms of its contribution to the overall inconsistency of a rule set based on how many inconsistencies would be resolved if the subset were removed. This provides an intuitive ordering to inconsistent subsets where the greater the score the greater the inconsistency and a value of 0 indicates the subset is entirely consistent.

Definition 4.1 (Scoring function [6]) Let $\Delta \in \mathcal{D}$ *and* S *be the scoring function for* Δ *defined as follows, where* $S : \wp(\Delta) \mapsto \mathbb{N}$ *and* $\Gamma \in \wp(\Delta)$*, then* $S(\Gamma) = |MI(\Delta)| - |MI(\Delta - \Gamma)|$.

Example 4.2 The result of applying the scoring function on formulae in preemptively-inconsistent subsets in Γ w.r.t. $\Theta = \{\alpha, \rho, \neg\pi, \sigma, \nu, \psi, \beta, \gamma\}$ is:

$S(\{\alpha \to \nu\}) = 2$ $S(\{\beta \to \rho \wedge \sigma\}) = 1$
$S(\{\gamma \to \pi \wedge \sigma\}) = 1$ $S(\{\nu \wedge \rho \wedge \neg\pi$
$S(\{\nu \wedge \psi \to \neg\delta\}) = 1$ $\wedge\sigma \to \phi\}) = 2$
$S(\{\alpha \to \neg\phi\}) = 2$ $S(\{\psi \to \delta \wedge \sigma\}) = 1$
$S(\{\phi \to \neg\alpha\}) = 2$ $S(\{\sigma \to \neg\beta \wedge \neg\gamma\}) = 2$
$S(\{\alpha \to \phi\}) = 2$

From these results we can identify an inconsistency order based on the scoring value, i.e. those formulae with a score of 2 would be considered more inconsistent than those with a score of 1 because their removal would resolve more inconsistencies. However it is not sufficiently discriminatory because variations in inconsistency scores are limited (e.g. 0, 1, 2). This means identifying inconsistency ordering will also be limited because the scoring function does not assign each formula a proportion of blame for the overall inconsistency.

The combined scores for subsets E, Z and H would be $S(E) = 6$, $S(Z) = 5$ and $S(H) = 4$. The overall inconsistency score for the rule set Γ is $S(\Gamma) = 7$. $\square$

B. Shapley inconsistency value

Two methods were proposed in [7] to address the issue of identifying the proportion of blame associated with each formula in terms of their contribution to the overall inconsistency of a base. Both methods take an inconsistency measure as a payoff function in coalitional form and, using the Shapley value from coalitional game theory, determine the proportional inconsistency for each formula in a base called the Shapley Inconsistency Value (SIV). We first present the definition for the SIV, since it will be used by the I_{MI} and I_{LP_m} inconsistency measures presented in this section.

Definition 4.2 (Shapley Inconsistency Value [7]) Let I be a basic inconsistency measure. We define the corresponding Shapley Inconsistency Value (SIV), noted S_I, as the Shapley value of the coalitional game defined by the function I, i.e. let $a \in K$:

$$S_I^K(a) = \sum_{C \subseteq K} \frac{(c-1)!(n-c)!}{n!}(I(C) - I(C \backslash \{a\}))$$

where n is the cardinality of K and c is the cardinality of C.

Definition 4.3 (SIV Vector) $S_I(K|\Theta)$ denotes the vector of the corresponding SIV for each formula of the base K w.r.t. Θ where $a \in K$, i.e. $S_I(K|\Theta) = (S_I^{K \cup \Theta}(a_1), \ldots, S_I^{K \cup \Theta}(a_n))$.

Definition 4.4 (Max SIV) Let K be a belief base, $\hat{S}_I(K|\Theta) = max\ S_I^{K \cup \Theta}(a)$, $a \in K$.

We now define I_{MI} measure [7], which takes into account the extent to which a formula contributes to an inconsistency.

Definition 4.5 (MI [7]) The MI inconsistency measure is defined as the number of minimal inconsistent subsets of K, i.e.: $I_{MI}(K) = |MI(K)|$.

The $S_{I_{MI}}$ measure can be summarized as the sum of the proportion of all MISs of which a formula is a member.

Example 4.3 Given the rule set Γ in Example 4.1, the result of applying the MI Shapley inconsistency measure on formulae in preemptively-inconsistent subsets of Γ w.r.t. $\Theta = \{\alpha, \rho, \neg\pi, \sigma, \nu, \psi, \beta, \gamma\}$ is (to reduce notation in the calculations below, we simply write Γ instead of $\Gamma \cup \Theta$, this applies to the remaining examples in which Θ is defined):

$S^\Gamma_{I_{MI}}(\{\alpha \to \nu\}) = \frac{2}{3}$ $S^\Gamma_{I_{MI}}(\{\beta \to \rho \wedge \sigma\}) = \frac{1}{2}$
$S^\Gamma_{I_{MI}}(\{\gamma \to \pi \wedge \sigma\}) = \frac{1}{2}$ $S^\Gamma_{I_{MI}}(\{\nu \wedge \rho \wedge \neg\pi$
$S^\Gamma_{I_{MI}}(\{\nu \wedge \psi \to \neg\delta\}) = \frac{1}{2}$ $\wedge\sigma) \to \phi\}) = \frac{2}{3}$
$S^\Gamma_{I_{MI}}(\{\alpha \to \neg\phi\}) = \frac{5}{6}$ $S^\Gamma_{I_{MI}}(\{\psi \to \delta \wedge \sigma\}) = \frac{1}{2}$
$S^\Gamma_{I_{MI}}(\{\phi \to \neg\alpha\}) = \frac{5}{6}$ $S^\Gamma_{I_{MI}}(\{\sigma \to \neg\beta \wedge \neg\gamma\}) = 1$
$S^\Gamma_{I_{MI}}(\{\alpha \to \phi\}) = 1$

Giving a vector value

$$S_{I_{MI}}(\Gamma) = (\tfrac{2}{3}, \tfrac{1}{2}, \tfrac{1}{2}, \tfrac{2}{3}, \tfrac{1}{2}, \tfrac{5}{6}, \tfrac{1}{2}, \tfrac{5}{6}, 0, 0, 1, 1, 0, 0)$$

The Max SIV for subsets E, Z and H would be: $\hat{S}^\Gamma_{I_{MI}}(E) = \frac{2}{3}$, $\hat{S}^\Gamma_{I_{MI}}(Z) = \frac{5}{6}$, and $\hat{S}^\Gamma_{I_{MI}}(H) = 1$. And a Max SIV for the rule set $\hat{S}^\Gamma_{I_{MI}}(\Gamma) = 1$. $\square$

This function produces more useful results than the Scoring function since it has a greater degree of discrimination in the inconsistency measures (e.g. $0, \frac{1}{2}, \frac{2}{3}, \frac{5}{6}, 1$) which allows us to identify more significant inconsistent formulae. The MI measure also produces a notably different inconsistency ordering, e.g. $\{\sigma \to \neg\beta \wedge \neg\gamma\}$ has the maximum inconsistency value by MI but is less significant by the Scoring function.

Next we introduce the I_{LP_m} measure [7], which takes into account the number of atoms, as well as of formulae, contributing to an inconsistency.

Definition 4.6 (I_{LP_m} [7]) Let $\mathcal{P}$ be the set of atoms in a language and K be a belief base from the language. Let ω be a total function from $\mathcal{P}$ to $\{T, F, B\}$, where B indicates an interpretation "both true and false". "Truth values" are ordered as $F <_t B <_t T$. Then

$$I_{LP_m} = \frac{min_{\omega \in Mod_{LP}(K)}\{|\omega!|\}}{|\mathcal{P}|}.$$

Here

$$\omega! = \{x \in \mathcal{P} | \omega(x) = B\}$$

which is the set of "inconsistent" variables in an interpretation. Also

$$Mod_{LP}(\varphi) = \{\omega \in 3^{\mathcal{P}} | \omega(\varphi) \in \{T, B\}\}$$

which is the set of models of formula φ.

$$min(Mod_{LP}(\varphi)) = \{\omega \in Mod_{LP}(\varphi) | \nexists \omega' \in Mod_{LP}(\varphi) \text{ s.t. } \omega'! \subset \omega!\}$$

The minimum models of a formula are the "most classical" ones. The LP_m consequence relation is then defined by:

$$K \models_{LP_m} \varphi \text{ iff } min(Mod_{LP}(K)) \subseteq Mod_{LP}(\varphi)$$

So φ is a consequence of K if all the "most classical" models of K are models of φ.

The $S_{I_{LP_m}}$ measure can be summarized as extending the $S_{I_{MI}}$ to include the number of atoms in an inconsistent formula as a proportion of the overall language.

Example 4.4 Given the rule set Γ, the result of applying the I_{LP_m} Shapley inconsistency measure on formulae in preemptively-inconsistent subsets of Γ w.r.t. $\Theta = \{\alpha, \rho, \neg\pi, \sigma, \nu, \psi, \beta, \gamma\}$ is:

$S^\Gamma_{I_{LP_m}}(\{\alpha \to \nu\}) = \frac{4}{51}$ $S^\Gamma_{I_{LP_m}}(\{\beta \to \rho \wedge \sigma\}) = \frac{3}{34}$
$S^\Gamma_{I_{LP_m}}(\{\gamma \to \pi \wedge \sigma\}) = \frac{3}{34}$ $S^\Gamma_{I_{LP_m}}(\{\nu \wedge \rho \wedge \neg\pi$
$S^\Gamma_{I_{LP_m}}(\{\nu \wedge \psi \to \neg\delta\}) = \frac{3}{34}$ $\wedge\sigma) \to \phi\}) = \frac{10}{51}$
$S^\Gamma_{I_{LP_m}}(\{\alpha \to \neg\phi\}) = \frac{5}{51}$ $S^\Gamma_{I_{LP_m}}(\{\psi \to \delta \wedge \sigma\}) = \frac{3}{34}$
$S^\Gamma_{I_{LP_m}}(\{\phi \to \neg\alpha\}) = \frac{5}{51}$ $S^\Gamma_{I_{LP_m}}(\{\sigma \to \neg\beta \wedge \neg\gamma\}) = \frac{3}{17}$
$S^\Gamma_{I_{LP_m}}(\{\alpha \to \phi\}) = \frac{2}{17}$

Giving a vector value

$$S^\Gamma_{I_{LP_m}}(\Gamma) = (\tfrac{4}{51}, \tfrac{3}{34}, \tfrac{3}{34}, \tfrac{10}{51}, \tfrac{3}{34}, \tfrac{5}{51}, \tfrac{3}{34}, \tfrac{5}{51}, 0, 0, \tfrac{3}{17}, \tfrac{2}{17}, 0, 0)$$

The Max SIV for subsets E, Z and H would be $\hat{S}^{\Gamma}_{I_{LP_m}}(E) = \frac{10}{51}$, $\hat{S}^{\Gamma}_{I_{LP_m}}(Z) = \frac{5}{51}$, and $\hat{S}^{\Gamma}_{I_{LP_m}}(H) = \frac{3}{17}$. And a Max SIV for the rule set $\hat{S}^{\Gamma}_{I_{LP_m}}(\Gamma) = \frac{10}{51}$. □

We can see that this function introduces the concept that the more atoms of a language affected by inconsistency, the more inconsistent the base, e.g. $\{\nu \wedge \rho \wedge \neg\pi \wedge \sigma \rightarrow \phi\}$ now has the maximum inconsistency value. It also produces more variation in inconsistency measures than both the Scoring function and MI Shapley measure (e.g. $0, \frac{4}{51}, \frac{3}{34}, \frac{5}{51}, \frac{1}{17}, \frac{2}{17}, \frac{3}{17}, \frac{10}{51}$).

C. $Blame_v$ prioritized measure

There are two types of prioritized knowledge bases described in [9], namely the Type-I and the Type-II prioritized knowledge base, which provide an intuitive ordering for formulae within a base. The approach of Type-I knowledge bases is to assign a numerical significance value to each formula. Its limitation is that it is difficult to determine a significance value for a formula in relation to other formulae in the base. Instead of assigning numerical values to formulae, Type-II prioritized knowledge bases divide formulae into subsets based on their priority, and a knowledge base is regarded as a collection of subsets with the first subset containing the most important formulae etc.

Definition 4.7 Let T be a Type-II prioritized knowledge base [9] with k priority levels. Let $T = \langle A_1, \ldots, A_k \rangle$ containing subsets of formulae assigned to priority levels $1, \ldots, k$ respectively, where A_1 has the most important formulae, and A_k the least.

Example 4.5 Let Γ be a Type-II prioritized knowledge base and E, Z and H be the sets of formulae from *Example 4.1*. Let E be most significant and H be least significant, i.e. $\Gamma = \langle E, Z, H \rangle$. The set of MPSs of Γ from *Example 4.1* w.r.t. $\Theta = \{\alpha, \rho, \neg\pi, \sigma, \nu, \psi, \beta, \gamma\}$ are now:

$$
\begin{aligned}
M_1 &= \langle \{\alpha \rightarrow \nu, \nu \wedge \rho \wedge \neg\pi \wedge \sigma \rightarrow \phi\}, \{\alpha \rightarrow \neg\phi\}, \emptyset \rangle \\
M_2 &= \langle \{\nu \wedge \psi \rightarrow \neg\delta\}, \{\psi \rightarrow \delta \wedge \sigma\}, \emptyset \rangle \\
M_3 &= \langle \{\alpha \rightarrow \nu, \nu \wedge \rho \wedge \neg\pi \wedge \sigma \rightarrow \phi\}, \{\phi \rightarrow \neg\alpha\}, \emptyset \rangle \\
M_4 &= \langle \{\beta \rightarrow \rho \wedge \sigma\}, \emptyset, \{\sigma \rightarrow \neg\beta \wedge \neg\gamma\} \rangle \\
M_5 &= \langle \{\gamma \rightarrow \pi \wedge \sigma\}, \emptyset, \{\sigma \rightarrow \neg\beta \wedge \neg\gamma\} \rangle \\
M_6 &= \langle \emptyset, \{\phi \rightarrow \neg\alpha\}, \{\alpha \rightarrow \phi\} \rangle \\
M_7 &= \langle \emptyset, \{\alpha \rightarrow \neg\phi\}, \{\alpha \rightarrow \phi\} \rangle \quad \square
\end{aligned}
$$

To begin defining the $Blame_v$ measure from [9] we must define the concept of Opposed formulae. This is the set of remaining formulae from a MIS resulting from the removal of a single formula from the MIS at each priority level. The cardinality of this set is used in determining a degree of blame for the formulae in an inconsistency.

Definition 4.8 (Opposed formulas [9]) Let M be a minimal inconsistent prioritized knowledge base and α^P a formula being attached with a priority level. Then the set of opposed formulae to α^P w.r.t. M, denoted $Opp(M, \alpha^P)$, is defined as:

$$
Opp(M, \alpha^P) = \begin{cases} \{\alpha^P\}, & \text{if } M = \{\alpha^P\}, \\ M - \{Opp(M, \alpha^P)\}, & \text{if } \{\alpha^P\} \subset M, \\ \emptyset, & \text{if } \alpha^P \notin M. \end{cases}
$$

Example 4.6 Given the MPS M_1 from $MPS(\Gamma | \{\alpha, \rho, \neg\pi, \sigma\})$, the result of applying the $Opp(M_1, m)$ for $m \in M_1$ is:

$$
\begin{aligned}
&Opp(M_1, \alpha \rightarrow \nu) = \langle \{nu \wedge \rho \wedge \neg\pi \wedge \sigma \rightarrow \phi\}, \{\alpha \rightarrow \neg\phi\}, \emptyset \rangle \\
&Opp(M_1, \nu \wedge \rho \wedge \neg\pi \wedge \sigma \rightarrow \phi) = \langle \{\alpha \rightarrow \nu\}, \{\alpha \rightarrow \neg\phi\}, \emptyset \rangle \\
&Opp(M_1, \alpha \rightarrow \neg\phi) = \langle \{\alpha \rightarrow \nu, \nu \wedge \rho \wedge \neg\pi \wedge \sigma \rightarrow \phi\}, \emptyset, \emptyset \rangle \quad \square
\end{aligned}
$$

We continue defining the structure of the $Blame_v$ measure by introducing the Inc_v measure from [9]. This, along with the Opposed formula, determines the $Blame_v$ value of a formula.

Definition 4.9 (Inc_v inconsistency measure [9]) Let $K = \langle K(1), \ldots, K(n) \rangle$ be a Type-II prioritized knowledge base. The inconsistency measure for K, denoted as $Inc_v(K)$, is defined as:

$$
Inc_v(K) = \sum_{M \in MI(K)} Inc_v(M),
$$

where $Inc_v(K) = \frac{Sig_v(M)}{|M|} \times Inc_c(M^*)$ for each $M \in MIS(K)$.

Especially, we call $Inc_v(K) = \frac{Sig_v(M)}{|M|} \times Inc_c(M^*)$ the i-th level inconsistency amount of M, and abbreviate it as $Inc_v^{(i)}(M)$.

Example 4.7 Given the Type-II prioritized knowledge base Γ w.r.t. $\Theta = \{\alpha, \rho, \neg\pi, \sigma, \nu, \psi, \beta, \gamma\}$ and the MPS M_1, the result of applying the Inc_v inconsistency measure is:

$$
Inc_v(M_1) = (\tfrac{2}{3^2}, \tfrac{1}{3^2}, \tfrac{0}{3^2}) = (\tfrac{2}{9}, \tfrac{1}{9}, 0)
$$

The Inc_v result for Γ is $Inc_v(\Gamma) = (\frac{45}{36}, \frac{35}{36}, 1)$. □

We now present the full definition of the $Blame_v$ measure from [9].

Definition 4.10 (The Blame of each formula for the k-th level inconsistency [9]) Let $K = \langle K(1), \ldots, K(n) \rangle$ be a Type-II prioritized knowledge base. Then for each $1 \leq k \leq n$, the blame of each formula of K for the k-th level inconsistency of K, denoted $Blame_v^{(k)}$, is defined as follows:

$$
\forall \alpha \in K, Blame_b^{(k)}(K, \alpha) = \sum_{M \in MI(K)} Blame_b^{(k)}(M, \alpha).
$$

where

$$
Blame_v^{(k)}(M, \alpha) = \begin{cases} \frac{Sig_v^{(k)}(Opp(M,\alpha))}{\sum_{\beta \in M} Sig_v^{(k)}(Opp(M,\beta))} \times Inc_v^{(k)}(M), & \text{if } |M(k)| > 0, \\ 0, & \text{if } |M(k)| = 0. \end{cases}
$$

for each $M \in MI(K)$.

Example 4.8 Given the results of $Opp(M_1, m)$ for $m \in M_1$ and $Inc_v(M_1)$, the result of $Blame_v^{(k)}$ for the formula $\alpha \rightarrow \nu$ in each priority level is:

$$
\begin{aligned}
Blame_v^{(1)}(M_1, \alpha \rightarrow \nu) &= \frac{|\{\nu \wedge \rho \wedge \neg\pi \wedge \sigma \rightarrow \phi\}|}{|\{\nu \wedge \rho \wedge \neg\pi \wedge \sigma \rightarrow \phi\}| + |\{\alpha \rightarrow \nu\}| + |\{\alpha \rightarrow \nu, \nu \wedge \rho \wedge \neg\pi \wedge \sigma \rightarrow \phi\}|} \times \tfrac{2}{9} = \tfrac{1}{18} \\
Blame_v^{(2)}(M_1, \alpha \rightarrow \nu) &= \frac{|\{\alpha \rightarrow \neg\phi\}|}{|\{\alpha \rightarrow \neg\phi\}| + |\{\alpha \rightarrow \neg\phi\}| + |\emptyset|} \times \tfrac{1}{9} = \tfrac{1}{18} \\
Blame_v^{(3)}(M_1, \alpha \rightarrow \nu) &= \frac{|\emptyset|}{|\emptyset| + |\emptyset| + |\emptyset|} \times \tfrac{2}{9} = 0
\end{aligned}
$$

Giving an overall $Blame_v$ value for $\alpha \rightarrow \nu$ in M_1:

$$
Blame_v(M_1, \alpha \rightarrow \nu) = (\tfrac{1}{18}, \tfrac{1}{18}, 0). \quad \square
$$

Definition 4.11 (The Blame of each Formula for Inconsistency [9]) Let $K = \langle K(1), \ldots, K(n) \rangle$ be a Type-II prioritized knowledge base. The blame of each formula of K for the inconsistency of K, denoted $Blame_v$, is defined as follows:

$$\forall \alpha \in K, Blame_v(K, \alpha) = (Blame_v^{(1)}(K, \alpha), \ldots, Blame_v^{(n)}(K, \alpha),$$

where $(Blame_v^{(1)}(K, \alpha)$ *is the blame of* α *to the* k*-th level inconsistency of* K *for each* $1 \leq k \leq n$.

Definition 4.12 (The relation of less blameful than, $\leq_B$ *[9]) Let* K *be a Type-II prioritized knowledge base. A binary relation on* K*, denoted as* $\leq_B$*, is defined as follows:* $\alpha \leq_B \beta$ *if and only if* $Blame_v(K, \alpha) \preceq Blame_v(K, \beta)$.

Further, $\alpha <_B \beta$ *if* $\alpha \preceq_B \beta$ *and* $\beta \not\preceq_B \alpha$. $\alpha \simeq_B \beta$ *if* $\alpha \leq_B \beta$ *and* $\beta \leq_B \alpha$. *We say that* α *is less blameful for the inconsistency in* K *than* β *if* $\alpha <_B \beta$.

Example 4.9 Given the Type-II prioritized rule set Γ, the result of applying the $Blame_v$ measure on formulae in preemptively-inconsistent subsets of Γ w.r.t. $\Theta = \{\alpha, \rho, \neg\pi, \sigma, \nu, \psi, \beta, \gamma\}$ would be:

$$\begin{array}{ll}
Blame_v(\Gamma, \alpha \to \nu) & = (\frac{1}{9}, \frac{1}{9}, 0) \\
Blame_v(\Gamma, \beta \to \rho \wedge \sigma) & = (0, 0, \frac{1}{4}) \\
Blame_v(\Gamma, \gamma \to \pi \wedge \sigma) & = (0, 0, \frac{1}{4}) \\
Blame_v(\Gamma, \nu \wedge \rho \wedge \neg\pi \wedge \sigma \to \phi) & = (\frac{1}{9}, \frac{1}{9}, 0) \\
Blame_v(\Gamma, \nu \wedge \psi \to \neg\delta) & = (0, \frac{1}{4}, 0) \\
Blame_v(\Gamma, \alpha \to \neg\phi) & = (\frac{1}{9}, 0, \frac{1}{4}) \\
Blame_v(\Gamma, \psi \to \delta \wedge \sigma) & = (\frac{1}{4}, 0, 0) \\
Blame_v(\Gamma, \phi \to \neg\alpha) & = (\frac{1}{9}, 0, \frac{1}{4}) \\
Blame_v(\Gamma, \sigma \to \neg\beta \wedge \neg\gamma) & = (\frac{1}{2}, 0, 0) \\
Blame_v(\Gamma, \alpha \to \phi) & = (0, \frac{1}{2}, 0)
\end{array}$$

This produces an ordering for the distribution of blame between inconsistent formulae of Γ, i.e.

$$(\{\beta \to \rho \wedge \sigma\} \simeq_B \{\gamma \to \pi \wedge \sigma\}) <_B \{\nu \wedge \psi \to \neg\delta\} <_B \{\alpha \to \phi\} <_B (\{\alpha \to \neg\phi\} \simeq_B \{\phi \to \neg\alpha\}) <_B (\{\alpha \to \nu\} \simeq_B \{\nu \wedge \rho \wedge \neg\pi \wedge \sigma \to \phi\}) <_B \{\psi \to \delta \wedge \sigma\} <_B \{\sigma \to \neg\beta \wedge \neg\gamma\}$$

where $\sigma \to \neg\beta \wedge \neg\gamma$ is considered the most inconsistent. □

The informativeness of the measures applied in this section can be summarized as:

- the I_{MI} Shapley value is more representative than the Scoring function because it considers the proportion that a formula contributes to the base;
- the I_{LP_m} Shapley value is an extension of the I_{MI} Shapley value because it also considers the proportion of a language affected by inconsistency;
- when formulae are prioritized the $Blame_v$ measure, unlike the I_{MI} Shapley value, can take into account this ordering. However it cannot represent the proportion of language affected by inconsistency.

V. Conclusion

Effective management of inconsistencies in intrusion detection rules is essential for reliable detection of intrusion attempts, especially with a large set of rules. The ability to not only identify these inconsistencies, but to distribute the degree of blame among contributing rules is important in the process of resolving inconsistencies.

In this preliminary study, we investigated how Snort2Regex rules can be translated into logical formulae. We began by identifying elementary units in the rules because in the current Snort2Regex system, each rule is looked at as a whole, so it is difficult to correlate them. We then extended this rule set to include some domain knowledge to more explicitly correlate rules. Finally, we applied a number of approaches for measuring inconsistency in this extended rule set. Our results can be summarized as:

- measuring inconsistencies in intrusion detection rules is potentially very useful for this domain since ensuring accurate and reliable intrusion detections is dependent on a consistent detection system;
- the inconsistency measures presented here are effective for quantifying the inconsistency of formula in an extended, knowledge-based, system. Using these values they can determine an inconsistency ordering for the degree of blame associated with each formula in order to begin resolving inconsistency;
- the major problem identified by this paper is that current rules used for intrusion detection are vulnerable to inconsistencies, but because they do not incorporate any domain specific knowledge, it is difficult to correlate rules in order to identify these inconsistencies.

From this investigation, we also discovered that integrating both the I_{LP_m} Shapley value and inconsistency measures for Type-II prioritized knowledge bases could give some distinct advantages. It would be interesting to see how these two measures can bet integrated to produce a new measure on inconsistency.

Further work will aim at proposing strategies for removing minimal numbers of Snort2Regex rules in order to resolve inconsistency.

Acknowledgement

We would like to thank Kedian Mu for his very helpful comments, and thank Sakir Sezer and Antonio Munoz for the Snort2Regex rule set.

References

[1] B. Kim, S. Yoon, J. Oh. Multihash based Pattern Matching Mechanism for High-Performance Intrusion Detection. *International Journal of Computers*, Issue 1, Volume 3. 2009.

[2] R. Agarwal, M. Joshi. PNrule: A New Framework for Learning Classifier Models in Data Mining. *Technical Report TR 00-015*. 2000.

[3] S. Beg, U. Naru, M. Ashraf, S. Mohsin. Feasibility of Intrusion Detection System with High Performance Computing: A Survey. *International Journal for Advances in Computer Science*, Volume 1, Issue 1. 2010.

[4] A. Munoz, S. Sezer, D. Burns, G. Douglas. An Approach for Unifying Rule Based Deep Packet Inspection. *IEEE ICC*. 2011.

[5] A. Hunter, S. Konieczny. Approaches to Measuring Inconsistent Information. In *Inconsistency Tolerance*, volume 3300 of LNCS. Springer. 189–234. 2004.

[6] A. Hunter. Logical Comparison of inconsistent perspectives using scoring functions. *Knowledge and Information Systems* 6(5) 528-543. 2003.

[7] A. Hunter, S. Konieczny. Shapley Inconsistency Values. In *Proceedings of the 10th International Conference on Knowledge Representation (KR-06)*, 249–259. AAAI Press. 2006.

[8] K. Mu, Z. Jin, R. Lu, W. Liu. Measuring inconsistency in requirements specifications. Proceedings of the *Eighth European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty* (ECSQARU05):440-451. LNAI 3571, Springer. 2005.

[9] K. Mu, W. Liu, Z. Jin. Measuring the Blame of each Formula for Inconsistent Prioritized Knowledge Bases. *Journal of Logic and Computation* (doi: 10.1093/logcom/exr002). 2011.

[10] M-O. Cordier, S. Loiseau. Validation of First-Order Rule-Based Systems. *Computational Intelligence* 12: 523-540. 1996.