Prioritizing Intrusion Analysis Using Dempster-Shafer Theory

Loai Zomlot, Sathya Chandran Sundaramurthy, Kui Luo, Xinming Ou
Kansas State University
Manhattan, KS, USA

S. Raj Rajagopalan
HP Labs
Princeton, NJ, USA

ABSTRACT

Intrusion analysis and incident management remains a difficult problem in practical network security defense. The root cause of this problem is the large rate of false positives in the sensors used by Intrusion Detection System (IDS) systems, reducing the value of the alerts to an administrator. Standard Bayesian theory has not been effective in this regard because of the lack of good prior knowledge. This paper presents an approach to handling such uncertainty without the need for prior information, through the Dempster-Shafer (DS) theory. We address a number of practical but fundamental issues in applying DS to intrusion analysis, including how to model sensors' trustworthiness, where to obtain such parameters, and how to address the lack of independence among alerts. We present an efficient algorithm for carrying out DS belief calculation on an IDS alert correlation graph, so that one can compute a belief score for a given hypothesis, *e.g.* a specific machine is compromised. The belief strength can be used to sort incident-related hypotheses and prioritize further analysis by a human analyst of the hypotheses and the associated evidence. We have implemented our approach for the open-source IDS system Snort and evaluated its effectiveness on a number of data sets as well as a production network. The resulting belief scores were verified through both anecdotal experience on the production system as well as by comparing the belief rankings of hypotheses with the ground truths provided by the data sets we used in evaluation, showing thereby that belief scores can be effective in mitigating the high false positive rate problem in intrusion analysis.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General; K.6.5 [**Managing of Computing and Information Systems**]: Security and Protection

General Terms

Experimentation, Security, Theory

Keywords

Dempster-Shafer Theory, Enterprise Network Security, Intrusion Detection and Analysis, Reason about uncertainty

AISec'11, October 21, 2011, Chicago, Illinois, USA.
Copyright 2011 ACM 978-1-4503-1003-1/11/10 ...$10.00.

1. INTRODUCTION

Intrusion analysis is the process of examining real-time events such as IDS alerts and audit logs to identify and confirm successful attacks and attack attempts into computer systems. The IDS sensors that we have to rely on for this purpose often suffer from a large false positive rate. For example, we run the well-known open-source IDS system Snort on our departmental network containing just a couple of hundreds machines and Snort produces hundreds of thousands of alerts every day, most of which happen to be false alarms. The reason for this are well-known: to prevent false negatives, *i.e.* detection misses from overly specific attack signatures, the signatures that are loaded in the IDS are often as general as possible, so that an activity with even a remote possibility of indicating an attack will trigger an alert. It then becomes the responsibility of a human analyst monitoring the IDS system to distinguish the true alarms from the enormous number of false ones. How to deal with the overwhelming prevalence of false positives is the primary challenge in making IDS sensors useful, as pointed out by Axelsson [3] more than 10 years ago.

Due to the lack of effective techniques to handle the false-positive problem, it is common among practitioners to altogether disable IDS signatures that tend to trigger large number of false positives. At one site we visited, the security analysts did not use the standard Snort rule sets at all, but rather resorted to secret, i.e. unpublished, attack signatures that are highly specific to their experience and environment and with known (small) false negative rates. We were told by the security analysts that secret signatures can only help capture some "low-hanging fruit", and that many attacks are likely missed due to the disabled signatures. Turning off IDS signatures is like turning a blind eye to attack possibilities, which we believe is a drastic consequence of the lack of effective solution techniques to *prioritize* investigations of alerts from IDS and audit logs. But, lacking any other significant distinguishing feature between the alerts, practitioners see no alternative.

1.1 Quantifying Uncertainty

Current IDS systems do not distinguish nor help distinguish the alarms that are highly likely to be true from those that have only a small chance of being true. By treating each suspected or imputed attack as has been suggested in earlier literature (see, e.g. [5] and references therein), merely as a hypothesis whose validity needs to be established, an effective approach to dealing with false positives can be formulated. The task then is to quantify the uncertainty in the hypotheses ascribed to IDS alerts by correlating multiple-point observations that are relevant to each alert. Given a list of intrusion hypotheses sorted by confidence and annotated by the evidential support for each hypothesis, it would be much easier for a human analyst to decide which hypotheses deserve further investigation. Since most network intrusions involve multiple actions, if we can relate observations from multiple events, a true successful attack will likely have multiple pieces of corroborating evidence, thus increasing the certainty of the attack hypothesis. Correspondingly, a false positive in one sensor is likely to have less corroborating evidence, thus the particular attack hypothesis will have a low score and be ignored. The key question then is *how to calculate a hypothesis's possibility of being true based on both the reasoning structure in which it is derived and the quality of the evidence that supports it.*

There have been past attempts [34, 36] at achieving this. Bayesian analysis [14] has been the standard and there have been some approaches using alternative theories such as Dempster-Shafer theory [23]. However, a number of **fundamental issues** in applying these mathematical theories to intrusion analysis remain to be addressed. For Bayesian analysis, it seems difficult to establish adequate prior probabilities such as the probability of a specific attack occurring in the environment or determine the conditional probabilities between system events in a robust manner. For Dempster-Shafer theory, it is not clear how to model sensor quality, where to obtain such parameters, and how to handle non-independent sources of evidence.

1.2 Our Contributions

Dempster-Shafer theory has unique advantages in handling uncertainty in intrusion analysis, namely, the ability to deal with the lack of prior probabilities for all (singleton) events and the ability to combine beliefs from multiple sources of evidence [6, 7, 34]. In this paper we present an extended Dempster-Shafer model that addresses the fundamental issues in applying DS in intrusion analysis, as mentioned in 1.1. We have implemented our method on top of our IDS alert correlation tool SnIPS [15, 21], so that one can calculate a numeric confidence score for each derived hypothesis and prioritize the results based on the scores. Our contributions are:

Using "unknown" to capture sensor quality.

(see section 3.2). Dempster-Shafer theory allows specifying a weight on "unknown" (or ”to be determined") rather than specifying precise probabilities for every possible event in the space. We use this ability to represent lack of knowledge to capture the intuitive notion of IDS sensor quality (which usually turns out to be imprecisely described), without suffering the non-intuitive effects of aggregation or forced classification that have been observed by researchers [34].

Accounting for lack of independence among alerts.

(see section 3.3). A long-standing assumption in DS theory is that multiple pieces of evidence are independent, which is a property that is hard to confirm in practice. This is especially a problem in IDS alerts since many alerts are triggered by the same or similar signatures. In combining these alerts to derive the overall belief on the attack status, it is important that such non-independence be appropriately accounted for so that the result is not skewed by over-counting. To the best of our knowledge, our method is the first in applying sound non-independent DS belief combination in IDS alerts.

Efficient calculation.

(see section B). A direct application of DS formulas can result in exponential (in the number of hypotheses – in our case, IP addresses) blow-up of belief combinations. We adopt a "*translate-then-combine*" approach so that beliefs are propagated in a correlation graph and only combined at join points in the graph. This produces an efficient algorithm with worst-case running time quadratic in the number of IP addresses in the input alerts.

Linking to practical IDS tools.

(see section 4) We have implemented our approach for the open-source IDS system Snort, and applied it on a production network and a number of data sets. We rigorously evaluate our method by separating the tuning phase from the testing phase, so that we do not fit our tool's parameters to work well with just one particular data set. Our evaluation suggests that the scores computed from our algorithm provide a useful ranking for the correlated alerts based on the correlations' trustworthiness. We have validated the results both anecdotally as well as with data set ground truths whenever available.

Robustness of solution.

(see section 4.4) We emphasize that our final goal is to sort alerts by confidence, hence we are interested in the relative order of hypotheses by confidence, not in establishing absolute certainties about attacks. Our application of DS requires the assignment of numeric values (constants) to certainty levels {unlikely, possible, likely, probable} but there is no help in the theory itself as to the manner of assignment. In a standard application of DS, the numeric scores may affect the final conclusions. However, since we are interested only in *relative* belief strengths assigned to the hypotheses, our approach is robust to small changes in these constants. Given any two related hypotheses, the absolute belief values are irrelevant as long as the relative strengths of belief remain unchanged when we slightly vary the numeric constants. Our experimental analysis shows that this is indeed true; the classifier's operating characteristic does not change when the constants' values are varied within a small range.

2. BACKGROUND ON DEMPSTER-SHAFER THEORY

A common example to illustrate the difference between probability theory and Dempster-Shafer theory is that if we toss a coin with an unknown bias, probability theory will still assign 50% for Heads and 50% for Tails by the principle of indifference ([11], p. 167), which states that all states of unknown probability should be assigned equal probability. Dempster-Shafer theory, on the other hand, handles this by assigning 0% belief to $\{Head\}$ and $\{Tail\}$ and assigning 100% belief to the *set* $\{Head, Tail\}$, meaning "either Head or Tail". By allowing us to assign 100% belief to $\{Head, Tail\}$ if warranted, DS does not force us to pick a probability when there is no basis to assign it. More generally, the DS approach allows for three kinds of answers: *Yes, No, or Don't know.* The last option of allowing ignorance makes a big difference in evidential reasoning. See [13], Ch. 2 for a discussion of the relative merits of DS belief theory. In DS theory, a set of disjoint hypotheses of interest, *e.g.,* {*attack, no-attack*}, is called a *frame of discernment.* The *basic probability assignment*, (*bpa*) function, also called the *mass* *distribution function* (m_θ), distributes the belief over the *power set* of the frame of discernment and is defined as:

$$m_\theta : 2^\theta \rightarrow [0,1] \tag{1}$$

$$m_\theta(\{\}) = 0 \quad \text{and} \quad \sum_{x \subseteq \theta} m_\theta(x) = 1 \tag{2}$$

Definition 1. *Let θ be a frame of discernment and m_θ a bpa function. The belief function is defined as*

$$Bel(x) = \sum_{y \subseteq x} m_\theta(y), for\ x \subseteq \theta \tag{3}$$

2.1 Dempster's Rule of Combination

The goal of combination is to fuse the evidences for a hypothesis from multiple *independent* sources and calculate an overall belief for the hypothesis [22]. Figure 1 illustrates this idea, where $alert_2$, $alert_3$ are two alerts triggered by independent IDS sensors. Independence means that knowing whether one sensor is trustworthy or not will not influence the likelihood for the other being trustworthy or not. A common assumption is that if two sensors are independent if they operate on completely unrelated features to determine attack possibilities. Both alerts could indicate that machine ip_2 is doing malicious probing of ip_3. The question is how we combine the beliefs from the two evidence sources. In general we have the following rule for fusing known as the Dempster's rule of combination.

$$m_{1,2}(h) = \frac{1}{1-K} \cdot \sum_{h_1 \cap h_2 = h} m_1(h_1) \cdot m_2(h_2) \tag{4}$$

$$K = \sum_{h_1 \cap h_2 = \{\}} m_1(h_1) \cdot m_2(h_2) \tag{5}$$

Where h_i's and h's are subsets of H, hypotheses in the frame of discernment. K is a normalization factor that is a measure of the *conflict* between the two sources of evidence, which is equivalent to the measure of the cases of empty intersection between the h_i's. The combined mass function must be normalized by $1 - K$ when conflict exists [22, 23].

The multiplication in formula 4 is only valid when the two evidence sources are independent [22]. This is often not the case in practice and especially so in IDS alerts since many alerts are generated by the same or related signatures. In the next section we introduce our extension of the DS model to account for non-independent evidence sources, so that the DS model can be correctly applied in intrusion analysis.

3. APPLYING DEMPSTER SHAFER ON INTRUSION ANALYSIS

3.1 The SnIPS Framework

We have built our DS-based hypothesis prioritization model on top of the SnIPS [15, 21] intrusion analysis system from our past work. SnIPS can work with the open-source Snort IDS system. It maps a triggered IDS alert to a hypothesis such as "machine compromised." It also maps the trustworthiness of the hypothesis to a discrete tag such as "possible" and "likely." Our past work [21] showed that building the mappings does not require much additional work, since the information already exists in an ad-hoc manner in the Snort rule repository. We have developed a heuristic algorithm to automatically infer the mappings by analyzing the Snort rules' documentation. After the alerts are mapped to hypotheses, the hypotheses are reasoned about efficiently based on a succinct internal reasoning model, and an alert correlation graph is built that shows the possible links among the hypotheses and alerts [29].

Example of SnIPS Output.

Figure 1 shows a sample segment of alert correlation graph automatically generated by SnIPS. "*compromised*", "*sendExploit*", and "*probeOtherMachine*" are predicates used to describe various attack hypotheses. The arrows' direction in the graph is aligned with inference. Five groups of alerts $alert_1 - alert_5$, are triggered by four different sensors. The notion of a sensor in our model is a bit different than other previous works. In our model we are not using the notion of physical snort sensor (i.e. Network card), or IDS in general. Instead, we are using each snort signature as a virtual sensor supporting the correlation graph. This is under the assumption that snort alerts will be triggered independently. For example in Figure 1 $sensor_1$ could be snort rule *1:1390.* This rule is usually trigged when an attempt is made to execute shellcode on a host[1]. The sensor nodes (the ones in dotted squares) are not part of the graph and are added here for clarity. $alert_1$ is mapped to the fact that host ip_1 sent an exploit to ip_2; both $alert_2$ and $alert_3$ are mapped to the fact that ip_2 did malicious probing to ip_3, and so on. The rationale for this correlation graph is that after ip_1 sends an exploit to ip_2, ip_2 may be compromised (node 9). Once the attacker has compromised ip_2, he can send malicious probing from there. Thus these alerts are all potentially correlated in the same underlying attack sequence. For representational simplicity, time information is not shown in the example graph (but is part of the reasoning process). In this example, $alert_2 - alert_5$ happened after $alert_1$. The arrow of the arcs indicate that all of $alert_1 - alert_5$ support the hypothesis that ip_2 was compromised.

3.2 Metrics for Sensor Quality

False positive and negative rates have been the standard metrics for characterizing an IDS sensor's quality. In this work we do *not* subscribe to such probabilistic metrics. Rather, we will use the "unknown feature" provided by DS theory to capture the case when we do not trust a sensor. The nature of *unknown* matches naturally with how humans interpret IDS alerts. When an alert is fired, we will have some degree (say 10%) of belief that an attack is going on. If we were to use a probability interpretation, we would have to say that we have 90% belief that the attack is *not* going on. One may find it counter-intuitive to positively assert that an attack is not going on based on seeing an alert. In probability theory, this is addressed by comparing the probability of an attack before and after seeing an alert. However, this would require the specification of the prior probability of attacks, which is hard if not impossible to obtain. By using DS, we can assign 0.1 belief to "attack"($\{true\}$), 0 belief to "no-attack" ($\{false\}$), and 0.9 belief to "Don't know" ($\{true, false\}$). This is a more intuitive quantitative interpretation of what an IDS alert provides: it gives some (small) belief that there is an attack but it does *not* give us any belief for "no-attack." Just because the sensor is not trustworthy, does not mean an attack is not going on. There may still be attack that is completely outside the scope of the sensor's

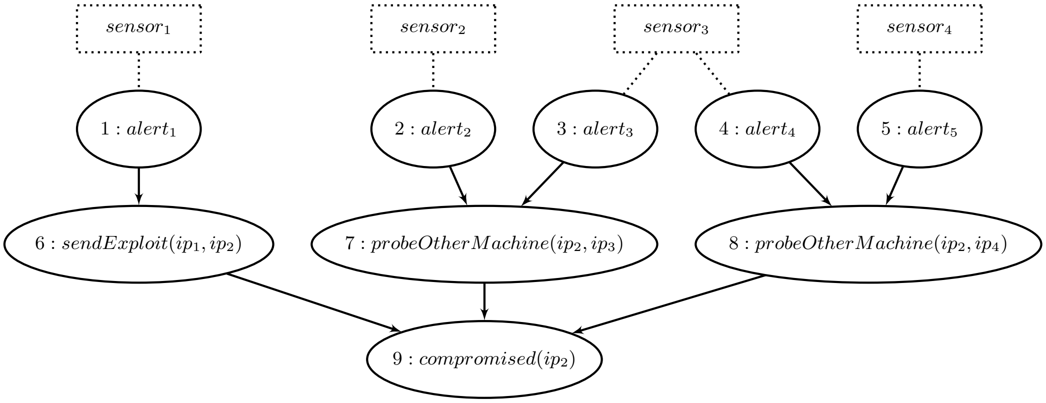


Figure 1: Automatically generated correlation graph segment from SnIPS

detection. Assigning the remaining weight to the "unknown" state indicates that we acknowledge the open-ended nature of attacks, which captures the reality of cyber security.

Using this method, we only need a single metric δ to characterize a sensor's quality. δ corresponds to the bpa of $\{true\}$ for the corresponding hypothesis when the sensor fires. Then $1-\delta$ will be assigned to $\{true, false\}$ (denoted θ thereafter). In the example of Figure 1, if we have $\delta = 0.1$ as $sensor_1$'s trustworthiness, $alert_1$ will translate to 0.1 mass distribution for $sendExploit(ip_1, ip_2)$ being *true*. 0.9 weight will be distributed to $sendExploit(ip_1, ip_2)$ being θ ("unknown"). We view δ as a metric solely dependent on the sensor's trustworthiness. We also assume for simplicity that shared IDS sensors only give us positive correlation, i.e. the triggering of one alert cannot cause us to decrease our belief in another correlated alert but only to increase it or stay the same. IDS signatures often come with ad-hoc natural-language descriptions that indicate the quality of the signature in terms of how likely the triggered alerts will be false positives, using qualitative terms such as "possible" and "likely." SnIPS extracts such terms from the Snort rule documentation and assigns a corresponding "certainty tag" for alerts generated by the rule [21]. In practice such tags can be provided easily by the rule writer if they are standardized, since they are already used in an informal way. We use the SnIPS certainty tags to map to the quantitative quality metrics for alerts generated by the various Snort rules (sensors), in the scheme shown in Table 1.

Table 1: Mapping discrete certainty tags to quantitative sensor quality metrics

Measures		Metrics	Measures		Metrics
unlikely	→	0.01	possible	→	0.33
likely	→	0.66	probable	→	0.99

The intuition is that humans typically cannot distinguish small differences in numerical parameters, thus a few discrete levels are sufficient to express the various beliefs one can ascribe to an alert. Through our analysis of the Snort rules' documentation, we found that four levels are sufficient to differentiate the various belief levels reflected by the rule writers about an alert's trustworthiness [21]. There is a low belief 0.01 and a high belief 0.99. The other two levels are evenly divided in the middle space. Another consequence of this model of sensor quality is that there will be no conflict among alerts. When we do not trust an alert, we just say "Don't know" whether the hypothesis is true, rather than assert that the hypothesis is false. This will not contradict the fact that we may trust another alert which derives the same hypothesis to be true. Thus we always have $K = 0$ in the combination formula (4).

3.3 Extending DS Combination Rule for Non-independent Evidence

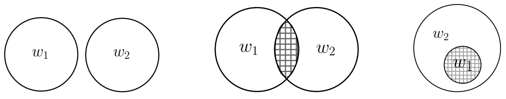
Correlated alerts could provide falsely elevated belief that an attack is going on, since multiple pieces of evidence point to the same conclusion. A key question is whether these multiple pieces come from independent sources. Through our research we discovered that we cannot ignore or avoid the overlapping nature of evidence. Often times we see multiple alerts in correlation supporting a hypothesis, but these alerts are triggered by the same or similar IDS signatures leading to an unjustifiably high level of confidence if we apply the standard Dempster rule of combination. In reality these multiple alerts should not significantly increase our belief in the hypothesis.

There may also be "partial non-independence" between two sources of evidence. In Figure 1, the main hypothesis is node 9: "whether machine ip_2 is compromised." This hypothesis is supported by the alert node 1 - 5. Node 1, an alert triggered by $sensor_1$, has evidence supporting node 6. Node 2 and 3 have evidence supporting node 7, so we combine the belief in 2 and 3 into node 7. Similarly, the belief in 4 and 5 will be combined into 8. Then we need to combine the belief in 6, 7, and 8 to answer the final question in node 9. Now we cannot ignore the fact that these nodes have overlapping evidence. Specifically, both node 7 and 8 partially rely upon alerts triggered by $sensor_3$. As a result, node 7 and 8 are not completely independent and we cannot simply apply the Dempster rule of combination (section 2.1).

There are a number of approaches in the DS literature to account for such dependence [10, 24, 25, 26]. We adopt an idea proposed originally by Shafer [26] which interprets combined bpa's as joint probabilities. Based on this, we develop a set of customized combination formulas to correctly account for the dependence in evidence when combining beliefs in the alert correlation graph.

3.3.1 The Customized Combination Formula

The reason Dempster's rule of combination has to assume evidence sources are independent is that joint mass function is calculated through multiplication (formula 4). For non-independent evidence, multiplication of bpa's from two



(a) Independent (b) Non independent (c) dependent

Figure 2: Venn diagram illustration of evidence dependency

sources is no longer valid [26]. Instead of $m_1(h_1) \cdot m_2(h_2)$, we use $\psi[h_1, h_2]$ to denote the joint bpa of the two sources. We obtain the following new formula for combining possibly non-independent evidence.

$$m_{1,2}(h) = \sum_{h_1 \cap h_2 = h} \psi[h_1, h_2] \tag{6}$$

One implication that arises from the application domain, namely intrusion analysis, is that the only h_i's of interest are $\{true\}$ (referred to as t hereafter) and $\{true, false\}$ (referred to as θ hereafter). The following equations specify $\psi[h_1, h_2]$,

$$\psi[t,t] = r_1 \cdot m_1(t) + (1 - r_1) \cdot m_1(t) \cdot m_2(t) \tag{7}$$
$$\psi[t,\theta] = (1 - r_1) \cdot m_1(t) \cdot m_2(\theta) \tag{8}$$
$$\psi[\theta,t] = (1 - r_2) \cdot m_1(\theta) \cdot m_2(t) \tag{9}$$
$$\psi[\theta,\theta] = r_1 \cdot m_2(\theta) + (1 - r_1) \cdot m_1(\theta) \cdot m_2(\theta) \tag{10}$$

The values r_1 and r_2 are *overlapping factors* which measure the amount of overlapping in the evidence from the two sources. Intuitively, r_1 is the portion of $m_1(t)$ that relies upon overlapping evidence from $m_2(t)$. Figure 2 shows three cases of overlapping between evidence sources. w_i is the statement that $h_i = t$ and the circles represent the IDS signature (sensor) set that triggered the alerts in support of this statement. For example, in Figure 2(a), the sensors that support w_1 are completely disjoint from those supporting w_2, giving the independent case, where we have $r_1 = r_2 = 0$ and equation (7 – 10) become the classical Dempster rule of combination. Figure 2(c) illustrates another extreme case where the evidence supporting w_1 completely falls inside the evidence supporting w_2. In this case we have $r_1 = 1$ meaning that when we believe w_1, we will believe w_2 as well. Thus $\psi[t,t] = m_1(t)$ and $\psi[t,\theta] = 0$. In this case if we do not trust any evidence for w_2 we will not trust any evidence in w_1 either. Thus we have $\psi[\theta,\theta] = m_2(\theta)$. In general r_1 will be between 0 and 1 as illustrated in Figure 2(b). The assumption is that the amount of overlapping between two pieces of evidence will affect their inter-dependence.

3.3.2 Estimation of The Overlapping Factors

We provide semantics for the overlapping factors using the probability theory. The detailed formulation can be found in Appendix A. The definition of r_i requires knowing certain conditional probabilities ($Pr[w_{i\pm1}|w_i]$ in the Appendix), which is not available. Thus we need to estimate r_i just as we need to estimate the bpa's for the sensors. In SnIPS each alert node is associated with a set of IDS signatures that triggered it. We view these signatures as different sensors (Figure 1). In our analysis the identities of the sensors that triggered an alert are propagated to the hypotheses it supports and further along the graph to other hypotheses it implies. Thus each hypothesis such as h_1 or h_2 is associated with the set of sensors whose alerts support it. Each sensor s has a quality metric δ_s as discussed in section 3.2. Let R_1 and R_2 be the two sensor sets associated with the hypothesis h_1 and h_2 to be combined using formula 6, and $R = R_1 \cap R_2$. We use formula 11 or 12 to estimate the overlapping between h_1 and h_2.

$$r_1 = \frac{\sum_{s \in R} \delta_s}{\sum_{s \in R_1} \delta_s} \quad , \quad r_2 = r_1 \cdot \alpha \quad , \quad \alpha \leq 1, \tag{11}$$

$$r_2 = \frac{\sum_{s \in R} \delta_s}{\sum_{s \in R_2} \delta_s} \quad , \quad r_1 = r_2 \cdot \alpha^{-1}, \quad \alpha > 1, \tag{12}$$

where α is defined in (A.1) and can be computed as:

$$\alpha = \frac{m_1(t) \cdot (1 - m_2(t))}{m_2(t) \cdot (1 - m_1(t))} \tag{13}$$

That is, we gauge the overlapping between the two sources by dividing the weight of the overlapping part by the total weight of each source, where the weight is calculated as the sum of the sensor quality metrics. Depending on the value of α, we estimate one of r_1, r_2 and compute the other using α. The above estimation ensures that both r_1 and r_2 are within $[0, 1]$. This is an intuitive estimation, and for both extreme cases in Figure 2 the estimation gives the accurate result (0 for independence and 1 for complete dependence). Appendix A gives further intuition behind the overlapping factor.

3.4 Belief Calculation Algorithm

Typically the alert correlation graph returned by SnIPS is not fully connected but contains a number of correlation graph segments like the one shown in Figure 1. The algorithm in general takes a set of correlation graph segments and calculates the belief value for each node on each graph. The graph segments are then sorted in descending order based on the maximum belief values for the sink nodes. To calculate the belief of the sink node, the algorithm propagates the quality metrics of each alert in the graph. The propagation will use the translation relation between the semantics of nodes. The algorithm applies the extended combination rule when there are multiple arcs flowing into one node like node 9 in Figure 1. IDS signature identifications are propagated throughout the graph to be used in estimating the overlapping factor using formula 11 or 12. The complexity of this algorithm is *linear* in the size of the graph. In the worst case the SnIPS-generated graph is quadratic in the number of IP addresses in the alerts [29]. Appendix B has the formal algorithm with its details.

3.5 An Illustrative Example

We use the example in Figure 1 to show the belief calculation process. It starts by computing the belief values for the source nodes alerts (node 1 – 5), each of which is associated with the sensor (IDS signature) that triggered it as in section 3.2. Then the belief values will be propagated through the graph using the semantics of the source node to the destination node using a set of predefined translation (compatibility) tables. Combination will be needed when multiple derivation paths lead to a single node. Let us take node 9 as an example, which has three pieces of evidence flowing into it from node 6,7,8. All the parent nodes 6,7,8's belief values based on their perspective semantics are translated into the bpa on node 9's semantics ($compromised(ip_2)$).

The algorithm sorts the three branches based on the translated belief values and combines the highest belief pair. In the similar manner the combined branches are further combined with the rest branches. Let us assume that node 7 and 8 are the first pair to combine. Node 7's belief value after translation is $m_1(t)$=0.68 and node 8 's value is $m_2(t)$=0.6. First we need to estimate the correlation factors r_1 and r_2 using formulas (11) or (12). Let R1 and R2 be the two sensor sets for node 7 and 8. $R_1 = \{sensor_2, sensor_3\}$ and $R_2 = \{sensor_3, sensor_4\}$. The quality metrics for the sensors are $\delta_{sensor_2} = 0.2$, $\delta_{sensor_3} = 0.6$, and $\delta_{sensor_4} = 0.01$. Using formula 13, we have $\alpha = 1.42$. After using formula 12, since $\alpha > 1$, we have $r_2 = 0.98, r_1 = 0.69$. Then after applying rules 7-10, we get table 2.

Table 2: Combination example

(h_1,h_2)	$\psi(h_1, h_2)$	h	(h_1,h_2)	$\psi(h_1, h_2)$	h
(t,t)	0.596	{t}	(t, θ)	0.084	{t}
(θ, t)	0.004	{t}	(θ, θ)	0.316	{θ}

Finally, $Bel(\{true\}) = 0.68$, calculated by summing up all the subsets of $\{true\}$ values. The final step is to combine this result with the belief from node 6, which will be in a similar manner. The sensor set associated with the combined belief will be the union of the sensor sets from all branches.

3.6 A Note on Methodology

Absence of Evidence.

Our current model just counts for the supporting evidence when they are present. On the other hand, DS can handle the absence of evidence as negative evidence by assigning weight to false in the computation table. Below we discuss the reason why we did not include this functionality.

Besides the well-known poor priors problem in Bayesian inference, there is a second challenge in real-world intrusion detection systems that needs to be addressed as well, which is that we typically have a very poor understanding of all the ways in which an attack occurs. IDS systems such as Snort are signature-oriented in that they are designed to detect the occurrence of a specific event or series of events that serve as an sensor for an underlying attack. By Bayesian methodology we should be able to measure and use in predictive analysis both the true positive and true negative rates of detection, when available. However, because these rates are measured in the laboratory under different conditions than the real environment traffic the claimed rates tend to be estimates of the real rates whose quality is undetermined. From systems administrators' experience we have learned that for signature-based systems true positive rates (i.e. $Prob(attack\ has\ occurred | alert\ has\ fired)$) are usually close to accurate whereas false negative rates (i.e. ($Prob(no\ attack\ has\ occurred | alert\ has\ not\ fired)$ are less so. The positive case is intuitive – the specificity of the signature in an alert leads us to believe that the attack under question may have occurred. (Yet, true positive rates are never 1 because multiple system behaviors, some of them unknown to us, may satisfy the same signature leading to false positives.) For the negative case, when a (possibly expected) signature event is not seen that may be either because the attack has not occurred (a true negative) or because the attack has occurred in an undetectable manner (a false negative). This is not symmetric with the positive case because in the negative case we are modeling attack behavior rather than system behavior. In keeping with our approach of making minimal assumptions about attacks, our belief strength is currently built on the true positive rate alone. In future work we can consider the use of the true negative rate for specific sensors that can reliably detect the absence of an attack. DS theory can handle this type of negative evidence with the corresponding compatability relations among positive and negative evidence defined. The computation formula will also need to be extended to handle mixed positive/negative evidence.

4. EXPERIMENTAL RESULTS

We implemented the algorithms in Java, and have been applying the system on our departmental network with about 200 servers and workstations (including Windows, Linux, and Mac OS X). The Snort alert collection, correlation, and DS algorithm application were all carried out on a Ubuntu server running a Linux kernel version 2.6.32 with 16GB of RAM on an eight-core Intel Xeon processor of CPU speed 3.16GHz. So far we have not encountered any performance bottleneck in our algorithm.

4.1 Evaluation Methodology

The objective of our evaluation is to examine whether the belief values calculated from our DS algorithm can help a security analyst to prioritize further investigation. To that end, we assigned to an IDS alert the belief value which is the highest belief of the hypotheses that supports. This can be easily calculated from the alert correlation graph through linear traversal. If IDS alerts with high belief values turn to be more likely true alerts than those with low belief values, it is an indication of the effectiveness of our approach.

Moreover, to show that it is indeed the application of Dempster-Shafer theory helps in alert prioritization, we compared the performance of our DS algorithm against that of the following alternative methods:

1. Using sensor quality metrics only. In this method, we simply use the sensor quality metrics assigned to each alert as described in Section 3.2 as an alert's belief value.
2. Using the maximum sensor-quality metric in a correlation graph as the belief value for all alerts in the graph.
3. Using the belief values calculated from the standard DS rule of combination, instead of from our customized DS.

All these methods assign a belief value to an IDS alert. A threshold value was chosen. Alerts with belief values above the threshold will be classified as true alerts, and those below the threshold will be classified as false alerts.

We used the truth files that included in the data set to determine which alerts are actually true alerts and which are actually false alerts. Then we compared this against the classification provided by the belief values. The key metrics in the classifier's performance are precision, recall (true positive), and false positive. As the belief-value threshold is changed, the classifier will obtain different operating points in terms of true positive and false positive. We draw ROC curves for the four methods and compared their performance.

$$\text{precision} = \frac{\#\text{ true alerts above threshold}}{\#\text{ alerts above threshold}}$$

$$\text{recall (true positive)} = \frac{\#\text{ true alerts above threshold}}{\#\text{ total true alerts}}$$

$$\text{false positive} = \frac{\#\text{ false alerts above threshold}}{\#\text{ total false alerts}}$$

4.2 The Rationale for Using Lincoln Lab Dataset

The first source of data we used in our evaluation is Snort alerts from the CIS departmental network at Kansas State University. Due to the lack of ground truth in such data from the production network, we provide anecdotal experiences on the effectiveness of our algorithm. In addition, we tested our prototype on the MIT Lincoln Lab DARPA intrusion detection evaluation data set. Although the Lincoln Lab data set has been criticized in the literature [16, 17], it still one of small number of usable publicly available data sets for IDS research. This is due to its well-documented ground truth and the existence of both background and attack traffic. We believe the limitation of the (LL) dataset will not significantly affect the validity of our evaluation for the following reasons.

1. Most of the identified problems in the (LL) dataset would affect anomaly-based detection [16] where one needs to use the data for both training and testing purposes. These defects will not affect as much signature-based IDS such as Snort, which we use as the underlying alert source.
2. Our reasoning model is built a priori from existing Snort rule repositories, and calibrated on our departmental network, completely unrelated to the (LL) data.
3. The problem in (LL) dataset's background traffic [17] makes it hard to make claims on the performance of the tested system on real networks. This is especially the case since it is a very old data set now. For this reason we will mainly use the dataset to compare performance. The relative performance of the various methods is likely not affected as much as the absolute performance, since they may all benefit or suffer from the specific features of the data set.

4.3 Lincoln Lab DARPA Data-set Results

4.3.1 DARPA 1998 and 1999 Training Data

We obtained the training data[1] in packet capture (pcap) format for both the 1998 and 1999 DARPA Intrusion Detection Evaluation program. We ran Snort on the packet capture data, ran SnIPS on the alerts triggered by Snort, and ran our DS calculation algorithm as well as the other three methods mentioned in Section 4.1 on the generated alerts and correlation graphs. We created ground truth about alerts using the truth files provided at the data set website. Each day has attacks targeted at specific machines, as given in the truth files. We carefully went through each attack described and checked against the alert database to pick out those alerts that can be verified as true alerts. The rest of the alerts are false alerts. This ground truth allows us to calculate the true positive and false positive of the various classifiers and plot their corresponding ROC curves.

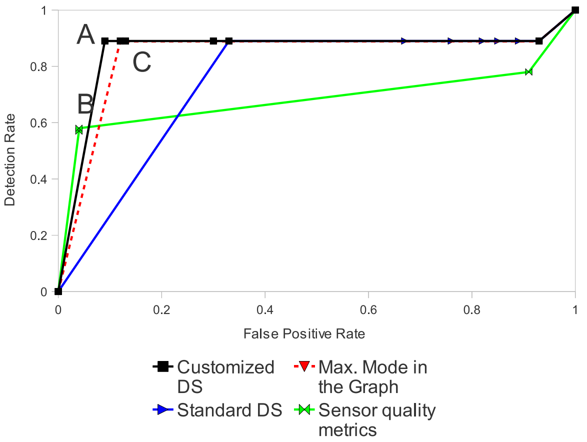


Figure 3: Lincoln Lab 1998 ROC curves

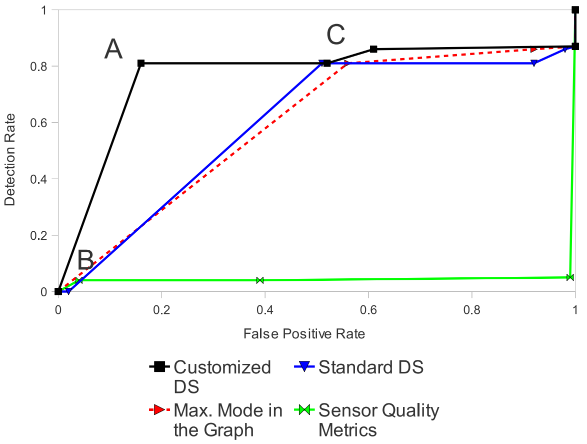


Figure 4: Lincoln Lab 1999 ROC curves

The Receiver Operating Characteristic (ROC) curve is a standard way to compare performance of IDS systems [3]. It shows the relationship between the detection rate (true positive) and false positive rate of a classifier. In general the steeper and closer to the left-up corner, the better the classifier. A comparison of the ROC curves generated for both data sets is shown in Figure 3 and 4. From the curves it is clear that our customized DS algorithm outperforms the other three alternative methods. Some operating points of the other three methods come close to the customized DS algorithm for the (LL 98) data, *e.g.* point B and C. But these points become much more inferior for the (LL 99) data. Whereas our DS algorithm produces the most optimal operating point consistently for both graphs (point A, corresponding to belief threshold 0.9).

4.4 Sensitivity Analysis

We also did experiments to test how the variation in the choice of sensor quality metric values for the certainty tags affect our algorithm's performance. We varied the default mapping shown in Table 1 in four different ways, each of which perturbs the numeric value by about 10%, *e.g.* from 0.33 to 0.3. We compared the results from all the four cases along with the default case in the ROC curves for the (LL 99) data (Figure 5). One can find that the five curves exactly

[1] Only training data's truth file is publicly available.

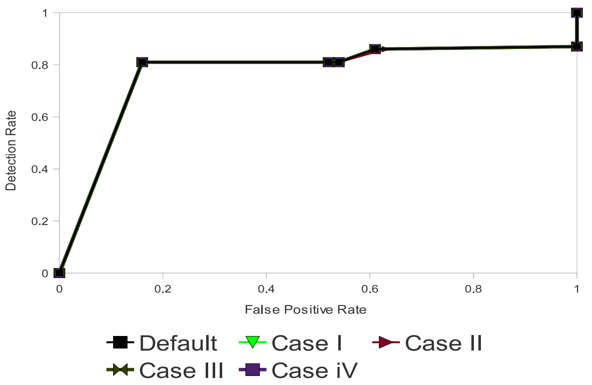


Figure 5: Lincoln Lab 1999 sensitivity analysis's ROC curves

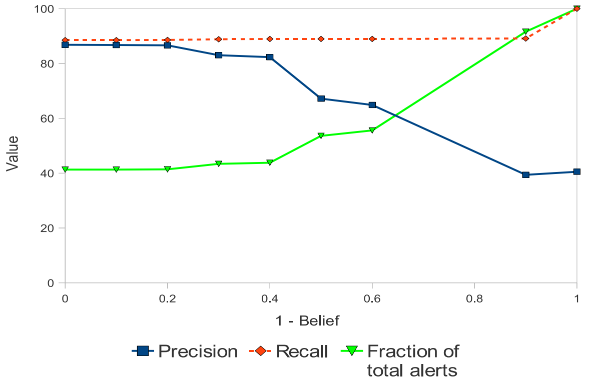


Figure 6: Prioritizing effect in Lincoln Lab 1998

overlap with each other indicating that small perturbation in the values for the certainty tags has virtually no effect at all on the performance of the classifier. We did the same experiment for (LL 98) data and also obtained five overlapping curves.

4.5 Prioritization Effect

Our main objective of applying Dempster-Shafer theory is to use the relative belief values to prioritize intrusion analysis. Figure 6 and 7 show how the precision and recall changes when the threshold decreases from 1 to 0 (note that 0 in the X axis corresponds to belief 1, and 1 corresponds to belief 0). When one starts with alerts with high beliefs, the precision is high meaning more of the effort is devoted to useful tasks. When the threshold decreases, the cumulative precision decreases as well. This is a strong indication that the calculated belief values can be used effectively to prioritize further investigation.

At the highest belief range (0 point at the X axis) the percentage of total alerts captured is about 40%, and the recall is about 80%. This means that if one only analyzes alerts with the highest belief (e.g. >0.9), it only includes 40% of all alerts whereas covers 80% of all the true alerts. The recall curve is very flat meaning that most of the attacks can be captured using a high threshold value. This is certainly only the case for these two specific data sets, but nevertheless it indicates the effectiveness of prioritization provided by the DS method. Without it, one would have to look at twice as many alerts to achieve the same coverage.

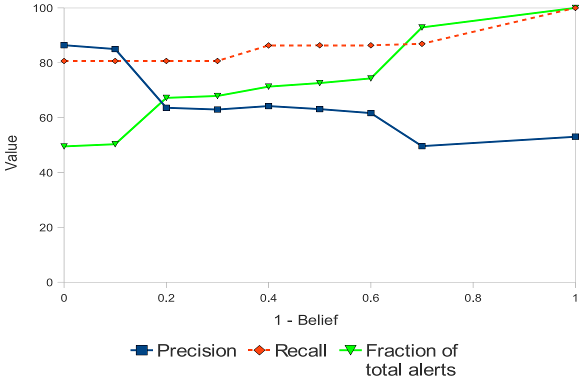


Figure 7: Prioritizing effect in Lincoln Lab 1999

4.6 The Production Network Results

In our evaluation we had permission from the University IT Security team to monitor the Snort alerts from the Computing and Information Sciences departmental network. This process had little privacy concern because the links between the machines' *IPs* and the users were not given to us and Snort only reveals limited payload data. All researchers involved in the experiments had signed the University IT Personnel Ethics agreement, consistent with the University policy.

Since it is hard to get ground truth in live systems, we presented the results to the system administrator of our departmental network to get his feedback on the tool's effectiveness. Regarding the quantitative belief calculation, the system administrator found that although the numbers themselves were hard to interpret intuitively, the ranking would be useful in prioritizing further analysis. He agreed that the higher-ranked correlations are indeed what he would like to investigate further, compared with the lower-ranked ones. In most cases the investigation indicated false positive or turned out inconclusive. But the ranking reduces the search scope for the system administrator which in real terms may translate to many man-hours of intrusion analysis oversight by a human. Certainly such anecdotal experiences cannot serve as validation of the method's effectiveness but real-world feedback is valuable in judging whether the tool is likely to be useful in the future.

5. RELATED WORK

Chen, et al. [7] described the general approach of applying standard DS theory to combine multiple sensors' reports for intrusion detection in ad-hoc networks. *Yu, et al.* [34] extended Dempster-Shafer theory for alert fusion in the HPCN IDS alert correlation systems [33]. They observed that direct application of Dempster-Shafer theory in IDS alert fusion provides non-intuitive results and extended DS to weight alerts based on their quality. Our approach is different in that we directly capture sensor quality by assigning the remaining bpa to the unknown case ($\{true, false\}$), instead of the $\{false\}$ case. We feel that our approach better captures the intuitive semantics provided by an IDS alert (section 3.2). Neither *Chen* nor *Yu* addresses non-independence among evidence sources, which we believe is an important issue and have designed a customized DS combination rule to handle (section 3.3).

There have also been approaches for alert fusion and prioritization based on decision theories. *Barreno, et al.* [4]

introduce an optimal approach for combining binary classifiers using the Neyman-Pearson lemma. *Guofei, et al.* [12] propose an alert fusion technique based on likelihood ratio test (LRT). We would like to investigate the possibility that these techniques could be applied in an IDS alert correlation framework and compare the result with that of our DS-based approach.

Ou, et al. proposed an empirical approach to handling uncertainty in intrusion analysis [21]. They proposed using discrete tags to capture alert uncertainty in correlation analysis and a "proof-strengthening" technique to elevate confidence in a hypothesis where there are multiple derivation paths pointing to the same conclusion. The proof-strengthening rule is based on empirical experience and the authors did not provide the rationale behind it. Our approach takes discrete input metrics, but uses a quantitative combination method which provides a finer-grained result that can be used to rank hypotheses. Our quantitative approach has a well-established theoretical foundation, and can potentially provide better prioritization.

There have also been work on using Bayesian Network in intrusion detection [2] and IDS alert correlation [18, 36]. The advantage of applying DS as opposed to Bayesian theory is that one does not need to know all the prior probabilities of events which are often unavailable. Indeed, DS is one of the various so-called non-traditional theories for uncertainty that generalize specific probabilities to an interval of probability, which also include Belief Theory, Subjective Logic, and Possibility Theory. Some of these other approaches have been proposed in IDS alert fusion [30]. According to [22] the DS Belief Function theory is superior to the other theories because of

- the relatively high degree of theoretical development in DS theory,
- the aspect of Dempster-Shafer theory as a generalization of traditional probability theory, namely, where probabilities are assigned to sets of events as opposed to mutually exclusive singletons events,
- the versatility of DS theory in combining different types of evidence from multiple sources, and
- the large number of applications of DS theory in engineering in the past ten years.

IDS alert correlation has been extensively studied in the past ten years [8, 9, 19, 20, 27, 31, 32, 33, 35, 37]. However, just because a correlation exists does not automatically mean the associated alerts are high confidence. The correlation itself are often "false correlations". From our conversation with system administrators, it is highly desirable that alert correlation tools prioritize their output based on the likelihood of true attacks. Our work provides one possible approach to this prioritization.

Denceux's work [10] explicitly raises and addresses the question of non-independent sources in DS theory. They point out that lack of independence in evidence is a valid concern in many applications and propose a new rule of combination called the "cautious rule" to handle this case. The cautious rule is designed to be as general as possible and is hence very complex and unintuitive. Our combination rule follows the general idea proposed by Shafer [26] and is based on a simple probabilistic semantics. It could be that our rule can be considered a highly specialized case of the general cautious rule, appropriate to our application.

Sun et al. [28] present an application of DS theory to the risk analysis of information systems security. They present an evidential reasoning approach that provides a rigorous, structured model to incorporate relevant risk factors, related counter measures and their interrelationships when estimating information system risk. *Chen et al.* [6] present an application of DS to the detection of anomalies in a variety of systems such as worm detection in email and learning in biological data. They show that by combining multiple (independent) signal sources using belief values and the Dempster combination rule, it is possible to achieve better results (characterized by rate of classification error) than by using a single signal. They point out that the advantage of using DS theory over Bayesian is that no *a priori* knowledge is required, making it potentially suitable for anomaly detection of previously unseen information whereas Bayesian inference requires *a priori* knowledge and does not allow allocating probability to ignorance.

6. FUTURE WORK

We will continue to apply our system on more production systems for extended periods of time, and gather data to further analyze its performance on real systems nowadays. There are more types of information than IDS alerts that could be incorporated into intrusion analysis; and Dempster-Shafer theory could be useful to reason about a much wider variety of dependency among various types of sensors, including non-monotonic dependencies. There are also other aspects such as temporal relationship that could affect the dependency. We plan to investigate along these directions when we gain more empirical experience of the method's effectiveness on production systems.

7. CONCLUSION

In this paper we presented a practical approach to prioritizing intrusion analysis using an extended Dempster-Shafer theory. The proposed DS application can correctly combine non-independent evidence commonly found in correlated IDS alerts. We proposed a DS model for capturing sensor quality that corresponds to the intuitive interpretation, and designed an algorithm for calculating belief values for hypotheses on an alert correlation graph. The main goal of this work is to reduce the workload on the system administrator by picking out those intrusion alerts that are most likely to be true and hence worthy of further investigation. We conducted rigorous evaluation of our approach on both a production network and two additional data sets. The results of evaluation strongly indicate that the ranking provided by the DS belief value gives good and robust prioritization on correlated alerts based on their likelihood of being true attacks. We believe our proposed approach will provide valuable practical tools to assist security analysts.

8. ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their valuable feedback and Alvaro Cardenas, the shepherd of our paper, in preparing for the final version. This material is based upon work supported by U.S. National Science Foundation under grant no. 1038366 and 1018703, AFOSR under Award No. FA9550-09-1-0138, and HP Labs Innovation Research Program. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the

National Science Foundation, AFOSR, or Hewlett-Packard Development Company, L.P.

9. REFERENCES

[1] Snort rules documentation:http://www.snort.org.

[2] Magnus Almgren, Ulf Lindqvist, and Erland Jonsson. A multi-sensor model to improve automated attack detection. In *11th International Symposium on Recent Advances in Intrusion Detection (RAID 2008)*. RAID, September 2008.

[3] Stefan Axelsson. The base-rate fallacy and the difficulty of intrusion detection. *ACM Trans. Inf. Syst. Secur.*, 3(3):186–205, 2000.

[4] Marco Barreno, Alvaro A. Cárdenas, and J. D. Tygar. Optimal ROC curve for a combination of classifiers. In *Advances in Neural Information Processing Systems (NIPS*, page 2008, 2007.

[5] Brian Carrier. A hypothesis-based approach to digital forensic investigations. Technical report, Center for Education and Research in Information Assurance and Security (CERIAS), Purdue University, 2006.

[6] Qi Chen and Uwe Aickelin. Anomaly detection using the Dempster-Shafer method. In *International Conference on Data Mining (DMIN2006)*, 2006.

[7] Thomas M. Chen and Varadharajan Venkataramanan. Dempster-Shafer theory for intrusion detection in ad hoc networks. *IEEE Internet Computing*, 2005.

[8] Steven Cheung, Ulf Lindqvist, and Martin W Fong. Modeling multistep cyber attacks for scenario recognition. In *DARPA Information Survivability Conference and Exposition (DISCEX III)*, pages 284–292, Washington, D.C., 2003.

[9] Frédéric Cuppens and Alexandre Miège. Alert correlation in a cooperative intrusion detection framework. In *IEEE Symposium on Security and Privacy*, 2002.

[10] Thierry Denceux. The cautious rule of combination for belief functions and some extensions. In *9th International Conference on Information Fusion*, 2006.

[11] T. L. Fine. *Theories of Probability*. Academic Press, New York, 1973.

[12] Guofei Gu, Alvaro A. Cárdenas, and Wenke Lee. Principled reasoning and practical applications of alert fusion in intrusion detection systems. In *Proceedings of the 2008 ACM symposium on Information, computer and communications security*, ASIACCS '08, pages 136–147, New York, NY, USA, 2008. ACM.

[13] Joseph Y. Halpern. *Reasoning about uncertainty*. The MIT Press, London,England, 2005.

[14] Finn V. Jensen and Thomas D. Nielsen. *Bayesian Networks and Decision Graphs*. Springer Verlag, 2007.

[15] Argus Lab. Snort intrusion analysis using proof strengthening (SnIPS). http://people.cis.ksu.edu/~xou/argus/software/snips/.

[16] M.V. Mahoney and P.K. Chan. An analysis of the 1999 DARPA/Lincoln Laboratory evaluation data for network anomaly detection. In *Proceedings of the Sixth International Symposium on Recent Advances in Intrusion Detection (RAID)*, 2003.

[17] John McHugh. Testing intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory. *ACM Trans. Inf. Syst. Secur. (TISSEC)*, 3(4):262–294, 2000.

[18] Gaspar Modelo-Howard, Saurabh Bagchi, and Guy Lebanon. Determining placement of intrusion detectors for a distributed application through bayesian network modeling. In *11th International Symposium on Recent Advances in Intrusion Detection (RAID 2008)*. RAID, September 2008.

[19] Peng Ning, Yun Cui, Douglas Reeves, and Dingbang Xu. Tools and techniques for analyzing intrusion alerts. *ACM Transactions on Information and System Security*, 7(2):273–318, May 2004.

[20] Steven Noel, Eric Robertson, and Sushil Jajodia. Correlating Intrusion Events and Building Attack Scenarios Through Attack Graph Distances. In *20th Annual Computer Security Applications Conference (ACSAC 2004)*, pages 350– 359, 2004.

[21] Xinming Ou, S. Raj Rajagopalan, and Sakthiyuvaraja Sakthivelmurugan. An empirical approach to modeling uncertainty in intrusion analysis. In *Annual Computer Security Applications Conference (ACSAC)*, Dec 2009.

[22] K. Sentz and S. Ferson. Combination of evidence in Dempster-Shafer theory. Technical report, Sandia National Laboratories, Albuquerque, New Mexico., 2002.

[23] G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, 1976.

[24] Glenn Shafer. The problem of dependent evidence. Technical report, University of Kansas, 1984.

[25] Glenn Shafer. Belief functions and possibility measures. *The Analysis of Fuzzy Information*, 1986.

[26] Glenn Shafer. Probability judgment in artificial intelligence and expert systems. *Statistical Science*, 2(1), 1987.

[27] R. Smith, N. Japkowicz, M. Dondo, and P. Mason. Using unsupervised learning for network alert correlation. *Advances in Artificial Intelligence*, pages 308–319, 2008.

[28] Lili Sun, Rajendra P. Srivastava, and Theodore J. Mock. An information systems security risk assessment model under Dempster-Shafer theory of belief functions. *Journal of Management Information Systems*, 22, 2006.

[29] Sathya Chandran Sundaramurthy, Loai Zomlot, and Xinming Ou. Practical IDS alert correlation in the face of dynamic threats. In *the 2011 International Conference on Security and Management (SAM'11)*, Las Vegas, USA, July 2011.

[30] Helen Svensson and Audun Jϕsang. Correlation of intrusion alarms with subjective logic. In *sixth Nordic Workshop on Secure IT systems (NordSec)*, 2001.

[31] F. Valeur. *Real-Time Intrusion Detection Alert Correlation*. PhD thesis, University of California, Santa Barbara, May 2006.

[32] Fredrik Valeur, Giovanni Vigna, Christopher Kruegel, and Richard A. Kemmerer. A Comprehensive Approach to Intrusion Detection Alert Correlation. *IEEE Transactions on Dependable and Secure Computing*, 1(3):146–169, 2004.

[33] Dong Yu and Deborah Frincke. A novel framework for alert correlation and understanding. In *International Conference on Applied Cryptography and Network Security (ACNS)*, 2004.

[34] Dong Yu and Deborah Frincke. Alert confidence fusion in intrusion detection systems with extended Dempster-Shafer theory. In *43rd ACM Southeast Conference*, Kennesaw, GA, USA, 2005.

[35] Y. Zhai, P. Ning, and J. Xu. Integrating IDS alert correlation and OS-level dependency tracking. *Intelligence and Security Informatics*, pages 272–284, 2006.

[36] Yan Zhai, Peng Ning, Purush Iyer, and Douglas S. Reeves. Reasoning about complementary intrusion evidence. In *Proceedings of 20th Annual Computer Security Applications Conference (ACSAC)*, pages 39–48, December 2004.

[37] J. Zhou, M. Heckman, B. Reynolds, A. Carlson, and M. Bishop. Modeling network intrusion detection alerts for correlation. *ACM Transactions on Information and System Security (TISSEC)*, 10(1):4, 2007.

APPENDIX

A. SEMANTICS OF THE OVERLAPPING FACTORS

Since we only have two non-zero bpa subsets: t and θ, in each hypothesis's frame of discernment, we use w_i to denote the fact that we trust h_i ($h_i = t$) and $\bar{w_i}$ (negation of w_i) to denote the fact that we do not trust h_i ($h_i = \theta$). One may find it strange that w_i and $\bar{w_i}$ appear to be not mutually exclusive, since θ includes both t and f. This is exactly the unique way in which DS expresses disbelief in a hypothesis – it differentiates clearly between not believing a hypothesis and believing the negation of that hypothesis. When we trust a hypothesis, we believe its state is t and when we do not trust a hypothesis, we do not know what its state is, hence θ. Interested readers are referred to Shafer's discussion on how to handle non-independent evidence using this interpretation [26]. The semantics of overlapping factor can be defined as:

$$r_1 = \frac{Pr[w_2|w_1] - Pr[w_2]}{Pr[\bar{w_2}]}, r_2 = \frac{Pr[w_1|w_2] - Pr[w_1]}{Pr[\bar{w_1}]}$$

Let us take r_1 as an example to explain the semantics. If we condition on trusting hypothesis h_1, the probability that we also trust h_2 is greater than or equal to its absolute probability since shared IDS sensors only give us positive correlation. The bigger the difference, the stronger influence trusting h_1 has on trusting h_2. The extreme case is when $Pr[w_2|w_1] = 1$, which gives $r_1 = 1$. Both r_1 and r_2 measure the dependence between w_1 and w_2, but from different directions.

THEOREM A.1.

$$r_2 = \alpha \cdot r_1, \ \ \textit{where}\ \alpha = \frac{Pr[w_1] \cdot Pr[\bar{w_2}]}{Pr[w_2] \cdot Pr[\bar{w_1}]} \tag{14}$$

PROOF.

$$r_1 \cdot Pr[\bar{w_2}] \cdot Pr[w_1] = Pr[w_1, w_2] - Pr[w_1] \cdot Pr[w_2]$$
$$r_2 \cdot Pr[\bar{w_1}] \cdot Pr[w_2] = Pr[w_1, w_2] - Pr[w_1] \cdot Pr[w_2]$$

We then have

$$r_1 \cdot Pr[\bar{w_2}] \cdot Pr[w_1] = r_2 \cdot Pr[\bar{w_1}] \cdot Pr[w_2]$$

□

THEOREM A.2.

$$\psi[h_1, h_2] = Pr[w_1, w_2]$$

PROOF. Let us substitute r_i's into formulas (7) – (10). Let us also substitute the following definitions:

$$m_i(t) = Pr[w_i] \quad m_i(\theta) = Pr[\bar{w_i}]$$

knowing that:

$$Pr[w_2|w_1] = \frac{Pr[w_1, w_2]}{Pr[w_1]}$$

then substitute the above into the definition of r_1, we get

$$r_1 \cdot Pr[\bar{w_2}] \cdot Pr[w_1] = Pr[w_1, w_2] - Pr[w_1] \cdot Pr[w_2]$$

knowing that $Pr[\bar{w_2}] = 1 - Pr[w_2]$, then:

$$\begin{aligned} &Pr[w_1, w_2] = r_1 \cdot Pr[w_1] + (1 - r_1) \cdot Pr[w_1] \cdot Pr[w_2] \\ &= \psi[t, t] \end{aligned}$$

□

The importance of this theorem is that our way of calculating the joint bpa $\psi[h_1, h_2]$ is sound in that it gives a generalization of the joint probability distribution of the trustworthiness of two (potentially) dependent sources. This also follows Shafer's general guide on how to handle non-independent evidence sources in DS [26], although Shafer did not provide the specific formulations.

B. BELIEF CALCULATION ALGORITHM

The main algorithm is **DsCorr** (Algorithm 1). This function takes *GraphSet* which is a set of correlation graph segments. It iterates on each graph, and returns a set of the graph segments sorted by the belief of the sink node (or highest sink node for multiple sinks) in descending order.

Algorithm 1 Rank graph segments by belief

```
function DsCorr(GraphSet)
    for each Graph in GraphSet do
        MakeAcyclic(Graph)
        ProcessingQueue ← all the source nodes
        while (ProcessingQueue is not empty) do
            Node←ProcessingQueue.RemoveHead
            ComputeNodeBelief(Node)
            Node.visited ← true
            for each c in Node.Children do
                if all c's parents are marked visited
                    AND c is not visited then
                    add c into ProcessingQueue
                end if
            end for
        end while
        record the highest belief value of sink nodes.
    end for
    return SortGraphSetbyBelief(GraphSet)
end function
```

Algorithm **ComputeNodeBelief** (algorithm 2) takes a node and returns the belief value of it. There are three cases to consider for the node: 1) it is a source node; 2) it has only one parent node, 3) it has multiple parents. In the first case **AssignBpaValues** is called to compute the basic probability assignment based on the method in Section 3.2. This case applies to the alert nodes, *e.g.*, node 1-5 in Figure 1. In the second case the node has only one parent so the translation function is called. The third case for combination is done by first translating implicitly into the node and then combine.

Algorithm 2 Compute the belief of a node

```
function ComputeNodeBelief(Node)
    if Node has no parents then
        AssignBpaValues(Node)
    else if Node has one parent p then
        Node.belief ← Translate(p)
        Node.sigSet ← p.sigSet
    else if Node has multiple parents ps then
        Node.belief ← Combine(ps)
        Node.sigSet ← union of ps.sigSet
    end if
end function
```