Aragorn: Eliciting and Maintaining Secure Service Policies

Nirav Ajmeri, Chung-Wei Hang, Simon Parsons, Munindar P. Singh

Abstract

Services today are configured through policies that capture expected behaviors. However, because of subtle and changing stakeholder requirements, producing and maintaining policies is nontrivial. Policy errors are surprisingly common and cause avoidable security vulnerabilities.

We propose Aragorn, an approach that applies formal argumentation to produce policies that balance stakeholder concerns. We demonstrate empirically that, compared to the traditional approach, Aragorn performs (1) better on coverage, correctness, and quality; (2) equally well on learnability and effort÷coverage and difficulty; and (3) slightly worse on time and effort needed. In this way, Aragorn demonstrate the potential for argumentation-based approaches to specify policies.

Introduction

Service engagements and organizational computing involve interactions of autonomous parties. To provide effective and secure service delivery, organizations specify policies that capture various aspects of their service interactions.

Policies promise the benefits of being explicit and inspectable. However, creating and maintaining policies is nontrivial: Human errors in system configuration are a significant threat to security. Specifically, defining and maintaining policies *correctly* is nontrivial because correctness depends upon stakeholder requirements, which can conflict and change. Policy errors can cause avoidable failure in service provisioning and introduce security vulnerabilities [15].

Approach

We propose Aragorn, a general-purpose method for policy creation and maintenance based on argumentation. Aragorn incorporates domain-specific argumentation schemes and provides an evidential basis for balancing competing concerns that might reflect (inconsistent) stakeholder requirements. In essence, Aragorn models each policy specification as a decision and represents arguments pro and con for that specification. This network of arguments captures the design rationale for each policy, highlighting the dependencies and conflicts between the policies. The arguments serve as guidance for updating the policies in light of changing requirements and changing premises---specifically, in determining which policies to drop, insert, or modify in light of changes to stakeholder requirements.

Aragorn comprises two main parts. The first is an evidence-based framework that identifies actions on policies (such as activating or deactivating a policy). Successful argumentation relies upon applying one or more *argumentation schemes* that identify critical questions that an argument can raise. Aragorn incorporates argumentation schemes geared toward service security policies, and provides a simple approach to reasoning about the strength of belief in an argument. Aragorn's second part is a methodology that guides developing security policies that are the most defensible given the premises and arguments.

Successful service delivery involves security and associated policies at multiple levels. For concreteness, we consider cloud services and adopt firewall policies to demonstrate Aragorn. Firewalls are a well understood setting in which to refine and evaluate methods for creating and maintaining service policies. Firewalls must be updated in light of new evidence about security threats and vulnerabilities [10].

Contributions and Findings

The novelty of this paper arises from its (1) treatment of a policy-specification methodology and (2) focus on the human aspects of service delivery and policy specification.

We demonstrate Aragorn via a firewall policy modeling scenario. We empirically evaluate Aragorn in a human-subject study focused on defining and maintaining firewall policies. We find that Aragorn yields improvements over the traditional approach in measures (defined below) of maintainability, coverage, correctness, effort, and effort÷coverage. Participants find Aragorn easy to learn and not difficult to apply though it requires additional effort. We conjecture that aggregated gains in quality compensate for the additional time though we expect that effective tool support would mitigate this shortcoming.

Firewall as an Exemplar Domain

Firewalls and firewall policies are central to the security infrastructure of an enterprise. They serve as the first line of defense against attacks and unauthorized traffic. Dynamically emerging threats and requirements require a system administrator to continually refine the firewall policies. As for any security policy, creating and maintaining firewall policies is nontrivial, and a lack of efficient mechanisms and tools to analyze firewall policies results in policy errors. For instance, Wool [15] found that despite significant effort by enterprises, firewalls are frequently wrongly configured. The complexity and challenges involved in defining and maintaining firewall policies makes firewalls an appropriate choice as the exemplar domain of application for Aragorn.

A firewall controls information flow from and to a computer network. A firewall policy is an ordered set of rules [2]. Each rule specifies if an incoming or outgoing packet is allowed or blocked, depending upon its protocol (TCP or UDP), source IP, source port, destination IP, and destination port. Given a packet, a firewall considers the rules in sequence and the takes action of the first rule that applies. If that action is not desired, it might result in a breach. Therefore, it is crucial to order the rules correctly. Table 1 shows a partial firewall policy, and lists the requirements (R), and the facts (F) for the firewall policy.

[Table 1 about here.]

Defining and maintaining a firewall policy involves complex decision making. There is a potential for conflicts between policies resulting in *anomalies* [1]. A rule L *generalizes* over a rule M provided L has higher priority than M, every packet that satisfies L also satisfies M, and both rules specify different actions. For example, in Table 1, Rule 12 generalizes over Rule 9 [2]. Additional firewall anomalies include *shadowing*, *correlation*, and *redundancy* [1]. Some firewall rules are never executed because of such anomalies.

Anomalies in a firewall policy may result in legitimate packets being blocked, and unwanted packets being allowed. An administrator should resolve any anomalies. However, detecting and resolving anomalies is tedious and error-prone. Common errors include failing to validate key assumptions; omitting important evidence; overlooking connections across the available evidence; and improperly prioritizing conflicting evidence and justifications.

Argumentation and Argumentation Schemes

An argument is constructed and consists of three parts: a conclusion, a set of premises, and an inference from the premises to the conclusion. Each elementary argument captures a reasoning step from premises to conclusion. An argument can be supported or attacked by other arguments. Argumentation involves constructing a network of arguments, where the conclusion of one is a premise of another [4]. Formal argumentation theory models arguments as first-class entities. An argument may *undermine* (contradict the premises of) or *rebut* (contradict the conclusions of) another [4].

An argumentation scheme is a pattern for constructing arguments [13] that represents the inference structure of an argument and provides *critical questions* for evaluating if the argument holds. A critical question captures critical thinking needed to engage in an argument. Specifically, a decision-maker raises critical questions to find arguments that support or attack another argument.

A decision-maker follows an argumentation scheme to iteratively collect evidence and infer the veracity of its conclusion until a decision (i.e., whether to accept or reject the conclusion) can be made. We write a scheme in the following template: $\langle \text{Premise}, \text{Conclusions}, \text{Questions} \rangle$.

The practical reasoning argumentation scheme provides a template for a situation where a decision-maker asserts that action A should be carried out because A is a means to realize requirement R [13]. We write this scheme as follows:

- Premise (Major) h_1: R is a requirement.
- Premise (Minor) h_2: Action A is a means to realize R.
- Conclusion c: A should be carried out.

This scheme corresponds to these critical questions:

CQ1 What requirements should we consider that might conflict with R?

CQ2 Are there alternative means available for carrying out R?

CQ3 Is A more efficient than these alternative means?

CQ4 Is it practically possible to bring about A?

CQ5 What consequences of bringing about A should we take into account?

CQ6 Are other actions, in addition to A, required to bring about R?

These critical questions help identify supporting or conflicting requirements and alternative and efficient means to realize a requirement. We adopt additional argumentation schemes, such as *arguments by consequence*, and *arguments from alternatives and opposites* to answer these questions.

Aragorn: Evidence-Based Argumentation

The decision-maker maintains pieces of evidence, each with a mass value, defining its importance. The decision-maker's knowledge base consists of premises and inference rules that are supported or opposed by some pieces of evidence. A belief measure of a premise or inference rule is calculated based on the mass values of the evidence that supports or opposes it. The decision-maker applies argumentation schemes to construct an argument and calculates the belief measure of its conclusion by combining the belief measures.

We adapt a probability-certainty representation of belief [14]. The probability represents the likelihood that the conclusion is true given the evidence, whereas the certainty measures the amount of nonconflicting evidence. A sufficiently high certainty indicates the completeness of the argument.

We define the belief measure of a premise or inference rule as in Tang et al. [12].

Definition 1 (Belief Measure Combination) *Let h_A and h_B be premises in Σ with belief measures $\langle b_A, d_A, u_A\rangle$, $\langle b_B, d_B, u_B\rangle$, respectively. Let h_C be another premise and $\delta = \frac{h_A}{h_C}$ be an inference rule with belief measure $\langle b_\delta, d_\delta, u_\delta\rangle$. Assume h_A and h_B are independent. Then,*

$$\begin{aligned}
b_{A\wedge B} &= b_A b_B \\
d_{A\wedge B} &= d_A d_B + d_A u_B + u_A d_B \\
b_{A\vee B} &= (b_A + b_B)/2 \\
d_{A\vee B} &= (d_A + d_B)/2 \\
b_C &= b_\delta b_A \\
d_C &= b_\delta d_A
\end{aligned}$$

By answering a critical question about h_i, the decision-maker collects (or updates) pieces of evidence $e \in E$ and their mass values corresponding to h_i. That is, answering a critical question about h_i either produces new evidence e_i, or increases the mass value of existing e_i, where $e_i \in \{E_{b(h_i)}, E_{d(h_i)}\}$. Some mass values of $e_j \in E_{u(h_i)}$ decrease because the mass values should sum to one. The evidence (and its mass value) can come from a new argument.

Defining Policies via Aragorn

Besides general argumentation schemes [13], Aragorn uses domain-specific schemes (e.g., the firewall argumentation scheme), associated critical questions, and related pieces of evidence to synthesize arguments.

Constructing Arguments

Table 1 lists the requirements underlying a firewall policy, for instance, Example Inc. needs to enable file transfer. And, one known fact is that FTP and SFTP enable file transfer. To construct arguments for these requirements and premises, we adopt the practical-reasoning argumentation scheme introduced above. From the file-transfer requirement and the fact, we conclude that FTP should be enabled. Further, on answering critical questions associated with the scheme, we find that FTP needs port 20. Iterating over the scheme again we conclude that we need to allow port 20 to support file-transfer requirement.

A useful firewall-specific scheme may be expressed as follows [6].

- Evidence E
- Premise h_1: Port P on destination H is required by application A
- Premise (Assumption) h_2: H is patched to handle vulnerabilities
- Premise (Exception) h_3: Vulnerability V on port P
- Premise (Exception) h_4: Source S is malicious
- Conclusion c: Allow packet to port P on destination H from S
- Inference Rule δ_1: $\frac{\neg h_3}{h_2}$
- Inference Rule δ_2: $\frac{\neg h_4}{h_2}$
- Inference Rule δ_3: $\frac{h_1 \wedge h_2}{c}$

Inference rules δ_1, δ_2, and δ_3 are general knowledge required by a computational system; each may be reasoned about by other argumentation schemes, which calculate the corresponding triple $\langle b(\delta), d(\delta), u(\delta)\rangle$.

The following critical questions are associated with the firewall scheme.

CQ1. Does any requirement rely upon opening port P?

CQ2. Are there any known security vulnerabilities V?

CQ3. Is there any evidence that host H avoids vulnerability V?

CQ4. Is there any evidence that source S is malicious?

Associating Evidence and Belief Measure

Consider the file-transfer requirement in Table 1, and the scenario where the administrator needs to determine if file transfer from the source locX.example.com should be allowed. By *argument by practical reasoning*, we see that FTP is a means to file transfer, and conclude that to enable file transfer via FTP, packets to port 20 should be allowed. Also, using a similar argument for SFTP, we see that SFTP is a means to file transfer, and conclude that to enable file transfer via SFTP, packets to port 21 should be allowed. Next, as described above, we apply the firewall scheme and associate the pieces of evidence listed in Figure 1c to determine if packets with destinations of ports 20 and 21 should be allowed.

Considering the pieces of evidence and their mass values in Figure 1c, we calculate the belief measure of the conclusions of arguments $A1$ "Allow packets to port 20" and $A2$ "Allow packets to port 21" as follows.

Argument $A1$: Allow packets to port 20

$$\begin{array}{rll}
h_1 : & \langle 0.70, 0.20, 0.10 \rangle & e_1 \\
h_2 : & (\neg h_3) \vee (\neg h_4) = \langle 0.30, 0.02, 0.68 \rangle & e_3, e_5, e_7, e_8, \delta_1, \delta_2 \\
c : & h_1 \wedge h_2 = \langle 0.20, 0.133, 0.667 \rangle & \delta_3 \\
p(c) : & 0.60 & \\
c(c) : & 0.333 &
\end{array}$$

Argument $A2$: Allow packets to port 21

$$\begin{array}{rll}
h_1 : & \langle 0.90, 0.05, 0.05 \rangle & e_2 \\
h_2 : & (\neg h_3) \vee (\neg h_4) & e_4, e_6, e_7, e_8, \delta_1, \delta_2 \\
 & = \langle 0.68, 0.002, 0.318 \rangle & \\
c : & h_1 \wedge h_2 & \delta_3 \\
 & = \langle 0.581, 0.015, 0.404 \rangle & \\
p(c) : & 0.975 & \\
c(c) : & 0.596 &
\end{array}$$

[Figure 1 about here.]

Figure 1 shows the belief measure calculation for arguments $A1$ and $A2$ to determine if packets being sent to ports 20 and 21 should be allowed. We observe that argument $A1$ yields low probability and certainty values whereas $A2$ yields high probability with reasonable certainty. Therefore, the decision-maker can allow packets to port 21.

Empirical Evaluation

We empirically evaluated the comparative effectiveness of Aragorn and the traditional rule-based approach (henceforth Trad). We considered following hypotheses.

$\mathbf{H_1}$. Aragorn yields policies of higher coverage than Trad.

$\mathbf{H_2}$. Aragorn yields policies of greater correctness than Trad.

$\mathbf{H_3}$. Modelers expend less time and effort in defining policies using Aragorn than those using Trad.

We conducted a human-subject study to evaluate these hypotheses. Our study was approved by our university's Institutional Review Board (IRB). We collected an informed consent from each participant and provided a payment of 20 USD to each participant completing the study.

Study Design

We selected 24 computer science (21 graduate, and three undergraduate) students. Each participant had more than three years of programming and software development experience, and was familiar with conceptual modeling, network security, and firewalls. Since network administration is a technology task performed by computer network engineers, our participants are close surrogates for firewall administrators. Of the participants, 19 had academic or industry experience with network security and firewalls and 16 had academic or industry experience with conceptual modeling.

Our study included three phases and applied the one-factor (approach) design with two alternatives (Trad and Aragorn).

Phase 1: Learn. Participants in each group learned the respective approach by specifying a firewall policy for a hypothetical academic scenario. The academic scenario had eight requirements that participants took into account when specifying a firewall policy.

Phase 2: Design. Each participant specified a firewall policy for a hypothetical enterprise scenario using the approach learned in Phase 1. The scenario had twelve requirements.

Phase 3: Maintain. We provided participants an incomplete solution to the scenario of Phase 2, and asked them to modify that solution to accommodate five additional requirements.

We mitigated two main threats to our study. Specifically, to handle skill differences between participants, we surveyed participants' about their educational backgrounds, their prior experience with conceptual modeling and network security, and their familiarity with defining and maintaining firewall policies. We balanced the groups based on the survey. To handle participants' failure to report information, participants completed the difficulty survey after each phase, while it was fresh in their minds, and recorded the completion time.

We split participants into two groups produced appropriate artifacts. The Trad group defined firewall packet filtering rules based on the requirements and the evidence described in the scenario. The Aragorn group used argumentation schemes and critical questions to create an argumentation network consisting multiple arguments (with premises and claims) and to associate evidence with arguments and assign a strength value (a decimal value between 0 and 1) based on their intuition.

Metrics

We analyzed the artifacts by each participant to measure the following:

Learnability Time in minutes to learn and design the solution. Lower is better.

Maintainability Time in minutes to make changes to an existing solution. Lower is better.

Coverage Ratio of the number of requirements satisfied to the total number of requirements in the design and the maintenance phases. Higher is better.

Correctness Ratio of the number of requirements satisfied to the number of requirements attempted. Higher is better.

Quality Product of *coverage* and *correctness*. Higher is better.

Difficulty in learning Rating by participant on a scale of 1--5 interpreted as very easy, easy, neutral, difficult, and very difficult. Lower is better.

Difficulty in applying Rating by participant on a scale of 1--5 interpreted as very easy, easy, neutral, difficult, and very difficult. Lower is better.

Effort Product of time in minutes to design the solution, and *difficulty in applying*. Lower is better.

Effort÷Coverage ratio *Effort* divided by *coverage*. Lower is better.

Results and Discussion

We analyzed the solutions designed by each participant and computed the study parameters. We applied Wilcoxon's ranksum-test to compare the difference in the medians ($\tilde{x}$). Wilcoxon's ranksum-test is stricter than the t-test, and does not assume normality. Table 2 lists the computed values.

[Table 2 about here.]

Learnability and Maintainability The average time to learn the approaches and design the solution for the security settings in Phases 1 and 2 was lower for Aragorn (84 minutes) than for Trad (88.6 minutes). However, the time taken to modify an existing solution in Phase 3 was greater for Aragorn (29.5 minutes) than for Trad (20.08 minutes), thus the null hypothesis is not rejected. The result was not surprising because participants in the Aragorn group were required to meticulously answer each critical question in the argumentation scheme as they made changes to the existing arguments. We can potentially overcome this challenge with improved tool support.

Coverage and Correctness The mean coverage (74.54%) and mean correctness (82.93%) for Aragorn was found to be significantly higher than the mean coverage (35.76%) and mean correctness (39.99%) obtained for Trad. Also, the p values for coverage and correctness are significant at the 5% level. This result can be attributed to Aragorn participants following a systematic approach guided by argumentation schemes to define general arguments from requirements, and further refine these general arguments into packet-level arguments to decide which packets to allow or deny. Figures 2a and 2b show the coverage and correctness box plots based on the numbers of requirements satisfied in Phases 2 and 3.

Difficulty Median perceived difficulty to learn was found to be the same for the two groups. Difficulty in applying was higher for Aragorn. This can be attributed to the participants lacking prior experience of working with formal argumentation. Figures 2c and 2d show plots for difficulty perceived by the participants. The p values indicate there is no significant difference in difficulty.

Effort and Effort÷Coverage ratio Participants using Aragorn spent more effort on the task. Figure 2e shows the boxplot for effort expended by the participants. However, as Figure 2f shows, the effort÷coverage ratio was lower for Aragorn (291.99) than for Trad (466.09). We do not include time and effort expended in Phase 1 when computing these parameters. The p values indicate there is no significant difference in the associated effort÷coverage measure.

[Figure 2 about here.]

Related Work

Relevant work includes works that combine argumentation and security requirements.

Walton, Reed and Macagno [13] provide a foundation for using general argumentation schemes. Several frameworks, models, and tools supporting argumentation have been proposed [6, 8, 9, 11], but their effectiveness has not been thoroughly empirically evaluated.

Bentahar et al. [3] propose an argumentation-driven approach for communities of web services. Their approach enables web services to negotiate and persuade peers to join communities, and reason about their prior commitments. Aragorn is novel in incorporating argumentation schemes and combining evidence and beliefs to capture the design rationale in service policies.

Franqueira et al. [5] propose an approach for modeling security requirements through argumentation that helps assess risks associated with security arguments. Similarly, Ionita et al.'s [7] risk analysis method is based on an argumentation game, and follows a tabular approach in which arguments are listed sequentially in their order of introduction. This method appears no more usable or scalable than the traditional firewall policy representation. These works cannot handle uncertainty and incompleteness pf information. Also, these works have not been empirically evaluated.

Conclusions

Our empirical evaluation of Aragorn indicates that Aragorn performs significantly better than the traditional approach. Irrespective of the time and effort expended, the measures of coverage, correctness, and quality are higher for participants using Aragorn. Aragorn performs on par with the traditional approach on learnability, difficulty, and effort÷coverage.

Aragorn addresses a general problem in specifying service security policies, namely, to respect the requirements of the various stakeholders despite any conflicts among those requirements. Aragorn provides a way to incorporate evidence where available and to ask critical questions to help orient a search for additional evidence.

An important future direction is to automatically extract evidence supporting and opposing policy arguments from a security policy corpus. Another direction is to adapt existing context-aware requirements elicitation approaches to systematically define, maintain and reason about contextual goals and plans.

Acknowledgments

We thank the US Department of Defense for support through the Science of Security Lablet at NC State University.

References

[1] Ehab Al-Shaer and Hazem Hamed. Discovery of policy anomalies in distributed firewalls. Proc. *Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, pages 2605--2616, 2004. IEEE Computer Society.

[2] Andy Applebaum, Karl Levitt, Jeff Rowe, and Simon Parsons. Arguing about firewall policy. Proc. *International Conference on Computational Models of Argument*, pages 91--102, 2012.

[3] Jamal Bentahar, Zakaria Maamar, Wei Wan, Djamal Benslimane, Philippe Thiran, and Sattanathan Subramanian. Agent-based communities of web services: An argumentation-driven approach. *Service Oriented Computing and Applications*, 2(4):219--238, 2008.

[4] Philippe Besnard and Anthony Hunter. *Elements of Argumentation*. MIT Press, 2008.

[5] Virginia N.L. Franqueira, Thein Than Tun, Yijun Yu, Roel Wieringa, and Bashar Nuseibeh. Risk and argument: A risk-based argumentation method for practical security. Proc. *IEEE International Requirements Engineering Conference*, pages 239--248, 2011.

[6] Thomas Gordon, Henry Prakken, and Douglas Walton. The Carneades model of argument and burden of proof. *Artificial Intelligence*, 171(10--15):875--896, 2007.

[7] Dan Ionita, Jan-Willem Bullee, and Roel Wieringa. Argumentation-based security requirements elicitation. Proc. *IEEE Workshop on Evolving Security and Privacy Requirements Engineering*, pages 7--12, 2014.

[8] Chris Reed and Glenn Rowe. Araucaria: Software for argument analysis, diagramming and representation. *International Journal on Artificial Intelligence Tools*, 13(04):961--979, 2004.

[9] Jordan Salvit, Zimi Li, Senni Perumal, Holly Wall, Jennifer Mangels, Simon Parsons, and Elizabeth Sklar. Employing argumentation to support human decision making. Proc. *International Workshop on Argumentation in Multi-Agent Systems*, 2014.

[10] Thomas Shinder. Security considerations for platform as a service (PaaS). http://social.technet.microsoft.com/wiki/contents/articles/3809.security-considerations-for-platform-as-a-service-paas.aspx, November 2013.

[11] Mark Snaith and Chris Reed. TOAST: Online ASPIC$^+$ implementation. Proc. *International Conference on Computational Models of Argument*, pages 509--510, 2012.

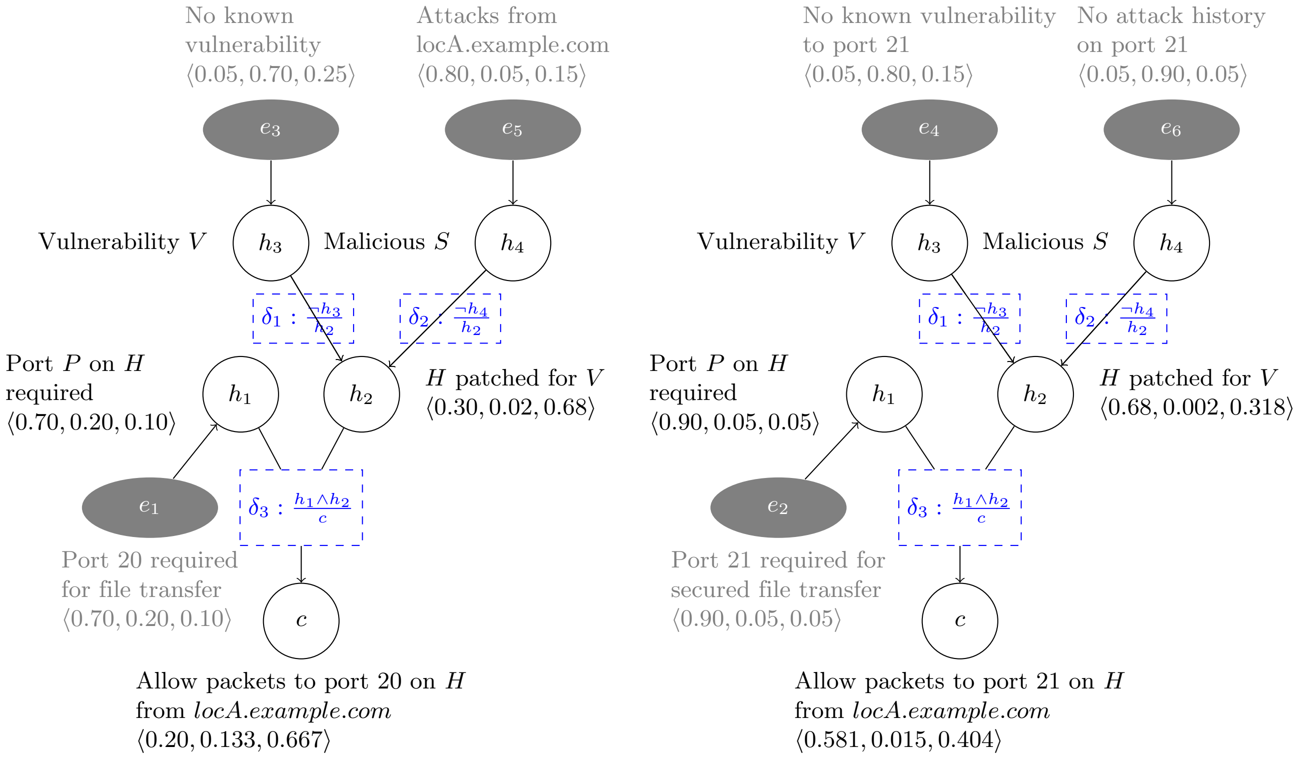
[12] Yuqing Tang, Chung-Wei Hang, Simon Parsons, and Munindar Singh. Towards argumentation with symbolic Dempster-Shafer evidence. Proc. *International Conference on Computational Models of Argument*, pages 462--469, 2012.

[13] Douglas Walton, Chris Reed, and Fabrizio Macagno. *Argumentation Schemes*. Cambridge University Press, 2008.

[14] Yonghong Wang, Chung-Wei Hang, and Munindar Singh. A probabilistic approach for maintaining trust based on evidence. *Journal of Artificial Intelligence Research*, 40:221--267, 2011.

[15] Avishai Wool. Trends in firewall configuration errors: Measuring the holes in Swiss cheese. *IEEE Internet Computing*, 14(4):58--65, 2010.

Figures



No known vulnerability ⟨0.05, 0.70, 0.25⟩
Attacks from locA.example.com ⟨0.80, 0.05, 0.15⟩
e_3
e_5
Vulnerability V
h_3
Malicious S
h_4
$\delta_1 : \frac{\neg h_3}{h_2}$
$\delta_2 : \frac{\neg h_4}{h_2}$
Port P on H required ⟨0.70, 0.20, 0.10⟩
h_1
h_2
H patched for V ⟨0.30, 0.02, 0.68⟩
e_1
$\delta_3 : \frac{h_1 \wedge h_2}{c}$
Port 20 required for file transfer ⟨0.70, 0.20, 0.10⟩
c
Allow packets to port 20 on H from *locA.example.com* ⟨0.20, 0.133, 0.667⟩
No known vulnerability to port 21 ⟨0.05, 0.80, 0.15⟩
No attack history on port 21 ⟨0.05, 0.90, 0.05⟩
e_4
e_6
Vulnerability V
h_3
Malicious S
h_4
$\delta_1 : \frac{\neg h_3}{h_2}$
$\delta_2 : \frac{\neg h_4}{h_2}$
Port P on H required ⟨0.90, 0.05, 0.05⟩
h_1
h_2
H patched for V ⟨0.68, 0.002, 0.318⟩
e_2
$\delta_3 : \frac{h_1 \wedge h_2}{c}$
Port 21 required for secured file transfer ⟨0.90, 0.05, 0.05⟩
c
Allow packets to port 21 on H from *locA.example.com* ⟨0.581, 0.015, 0.404⟩

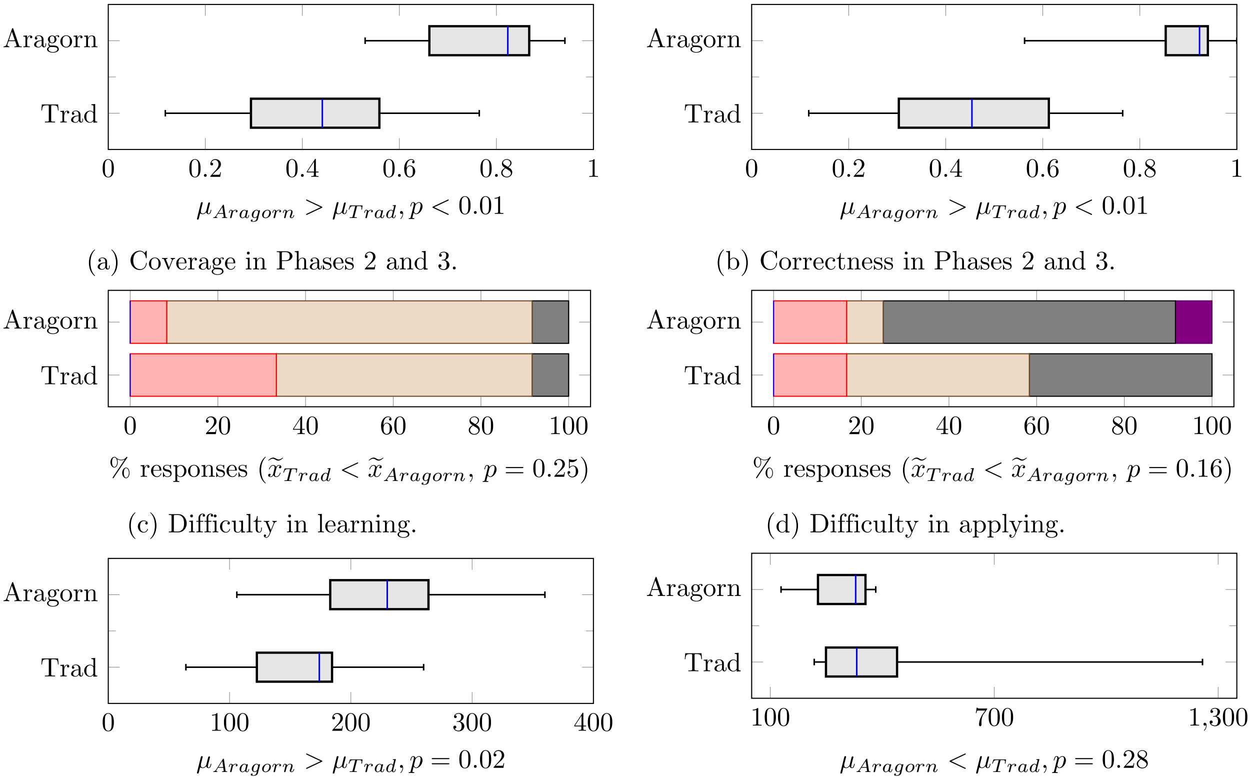
(a) Argument $A1$: Allow packets to port 20.

(b) Argument $A2$: Allow packets to port 21.

Evidence	Premise	Belief Measure	Description	Argument
e_1	h_1	$\langle 0.70, 0.20, 0.10\rangle$	Port 20 is required for file transfer via FTP	$A1$
e_2	h_1	$\langle 0.90, 0.05, 0.05\rangle$	Port 21 is required for file transfer via SFTP	$A2$
e_3	h_3	$\langle 0.05, 0.70, 0.25\rangle$	No known vulnerability to port 20	$A1$
e_4	h_3	$\langle 0.05, 0.80, 0.15\rangle$	No known vulnerability to port 21	$A2$
e_5	h_4	$\langle 0.80, 0.05, 0.15\rangle$	Attacks from locA.example.com, port 20	$A1$
e_6	h_4	$\langle 0.05, 0.90, 0.05\rangle$	No attack history on port 21	$A2$
e_7	δ_1	$\langle 0.80, 0.05, 0.15\rangle$	Decision-maker's experience in the rule	$A1$, $A2$
e_8	δ_2	$\langle 0.80, 0.05, 0.15\rangle$	Decision-maker's experience in the rule	$A1$, $A2$
e_9	δ_3	$\langle 0.95, 0.02, 0.03\rangle$	Decision-maker's experience in the rule	$A1$, $A2$

(c) Pieces of evidence corresponding to the file transfer requirement in Table 1.

Figure 1: Example arguments and pieces of evidence.



(a) Coverage in Phases 2 and 3.

(b) Correctness in Phases 2 and 3.

(c) Difficulty in learning.

(d) Difficulty in applying.

(e) Effort: time taken × subjective difficulty.

(f) Effort÷coverage ratio for Phases 2 and 3.

Figure 2: Results.

Tables

Table 1: An example firewall policy [2], and associated requirements (R) and facts (F)

#	Action	Protocol	Source	Port
1	Allow	*	*	20
2	Allow	*	*	80
3	Block	*	locA.example.com	20
4	Allow	*	*	21
5	Block	*	*	53
6	Allow	TCP	locB.example.com	23
7	Block	*	*.example.com	*
8	Allow	UDP	locB.example.com	5027
9	Allow	UDP	*	*
10	Block	*	*	6889
11	Allow	*	example.net	53
12	Block	*	*	*

#	Description
R	Example Inc. requires file transfer
F	FTP (port 20), SFTP (port 21) enable file transfer
R	Public website, made available via port 80
R	Prevent known attacks
F	Attacks from locA.example.com, port 20
R	Example Inc. requires telnet access
F	Telnet requires access to port 23
R	Application to access UDP port 5027
R	Prevent access to torrent ports
F	Torrent uses port 6889
R	Enable DNS for example.net
F	DNS requires access to port 53

Table 2: Empirical results on the effectiveness of Aragorn vis à vis the traditional approach. (Bold is better.)

	Statistic	Trad	Aragorn	*p*
Learnability (in minutes)	Mean	88.66	**84**	0.45
Maintainability (in minutes)	Mean	**20.08**	29.5	**0.03**
Coverage (in %)	Mean	35.76	**74.54**	$<$ **0.01**
Correctness (in %)	Mean	39.99	**82.93**	$<$ **0.01**
Quality (in %)	Mean	14.30	**61.82**	$<$ **0.01**
Learning difficulty (1--5)	Median	3	3	0.25
Applying difficulty (1--5)	Median	**3**	4	0.16
Effort	Mean	**158.33**	225.42	**0.02**
Effort÷coverage	Mean	466.09	**291.99**	0.28