Arguing About Firewall Policy

Andy Applebaum [a], Karl Levitt [a], Jeff Rowe [a], and Simon Parsons [b]
[a] *Dept. of Computer Science, University of California, Davis,*
applebau@ucdavis.edu
[b] *Dept. Comp. and Info. Science, Brooklyn College, City University of New York.*

Abstract. In this paper, we present a new framework to analyze firewall policy by using argumentation. At the core of this new idea is extending firewall rules with the concept of "reasons" and arguing about the reasons, not the rules. Depending on how the reasons are designed, the resulting framework can be useful in a number of ways: new anomalies in a firewall policy can be identified while, at the same time, stronger recommendations can be given to resolve those anomalies that are detected.

Keywords. Argumentation, value-based framework, firewall, anomaly

1. Introduction

A firewall is a program that controls the flow of information in to and out of a computer or network. Traditionally, most firewalls act as a barrier between a local network or personal user and the Internet; when a file (referred to as a "packet") from the Internet is sent to the protected computer, the firewall determines if that file should be allowed or blocked. Due to the widespread amount of malicious activity on the Internet, it's essential that a user's firewall be configured properly — allowing too many packets in subjects the user to potential harm, while blocking too many packets would prevent the user from doing anything useful. As a result, much work has been done (see section 4) to analyze firewall policy to ensure that a given policy behaves as desired. In this paper, we propose a new framework to analyze firewall policy that combines traditional firewall anomaly detection with argumentation by extending the standard firewall model to include "reasons".

2. Firewalls

2.1. Design and Operation

Firewalls are typically configured as a set of rules that describe how to behave when given a specific packet. Behavior is almost always either "allow" or "block", and while there are many parameters of a packet that a rule can check, in our model we only consider the following three: protocol (TCP or UDP), source IP

order	action	protocol	source IP	port
1	allow	*	*	20
2	allow	*	*	80
3	block	*	123.456.78.90	*
4	allow	*	*	21
5	block	*	*	53
6	allow	TCP	123.456.78.11	23
7	block	*	123.456.78.*	*
8	allow	UDP	123.456.78.11	5027
9	allow	UDP	*	*
10	block	*	*	6969
11	allow	*	75.75.75.75	53
12	block	*	*	*

Table 1. An example firewall policy.

(sender), and port (usually the intended purpose). As multiple rules may apply to the same packet, there is almost always a well-ordering of the rules to resolve conflicts. During operation, when a packet comes in to a firewall, it is first checked against rule 1. If that rule applies to the packet, whatever action the rule specifies is taken and execution stops. If rule 1 does not apply, then the packet is checked against rule 2 and acted on accordingly. This process is repeated until either a rule matches the packet, or all the rules are exhausted and a default rule applies. The following is a more precise specification of a rule:

$$rule ::= order : action,\ protocol,\ sourceIP,\ port$$

with each part is defined as follows ($*$ being a keyword for "any", and $\mathbb{Z}$ standing for the set of positive integers.):

$$\begin{array}{llll} order & \in \mathbb{Z} & action & \in \{allow, block\} \\ protocol & \in \{TCP, UDP, *\} & sourceIP & ::= x.x.x.x \\ x & \in [0, ..., 255] \cup \{*\} & port & \in \mathbb{Z} \cup \{*\} \end{array}$$

A "firewall policy" is simply a set of rules as specified above, where each rule has a unique value for "order" such that the lower-ordered rules have higher priority. Table 1 presents an example firewall policy, and Table 2 outlines the execution of that policy on some hypothetical packets — when packet A comes in, it's compared to rule 1: since A's port is 20, rule 1 applies, so it's allowed. When packet B comes in, rules 1 and 2 are skipped (as B's port is 23), and then rule 3 is applied. Packet C ignores rules 1–4, and then 5 is chosen. Packet D fits the specification for rule 5, so it's blocked. Packet E does not match any rule, so the "catch-all" of 12 is chosen. Finally, packet F is allowed from rule 9.

2.2. Anomalies

The motivation for applying argumentation in this domain is the idea of an "anomaly" in a firewall policy. This idea is taken from Al-Shaer et. al in [1,2]. In

name	protocol	source IP	port	resulting action	specified rule
A	TCP	1.1.1.1	20	allow	1
B	UDP	123.456.78.90	23	block	3
C	TCP	75.75.75.75	53	block	5
D	UDP	123.456.78.56	15	block	7
E	TCP	1.1.1.1	99	block	12
F	UDP	1.1.1.1	6969	allow	9

Table 2. Hypothetical packets and the appropriate actions.

Shadowing	Correlation		Generalization		Redundancy
(5, 11)	(1, 3)	(1, 7)	(1, 12)	(2, 12)	
(7, 8)	(2, 3)	(2, 7)	(4, 12)	(6, 7)	
	(3, 4)	(3, 9)	(6, 12)	(8, 12)	
	(4, 7)	(5, 9)	(9, 12)	(11, 12)	
	(7, 9)	(9, 10)			

Table 3. All anomalies in the example policy. Each pair (x, y) is an anomaly.

these papers, the authors define four anomalies as relations between rules — we summarize them here:[1]

Shadowing: Rule a is said to shadow rule b if a has higher-priority than b, a and b specify different actions, and every packet that satisfies b also satisfies a. In the example policy, rule 5 shadows rule 11 and rule 7 shadows rule 8.

Correlation: Rules a and b are correlated if a and b specify different actions and some packets that satisfy a also satisfy b and vice-versa. There are many correlation anomalies in the policy above, some examples being between rules 1 and 3 and between rules 4 and 7.

Generalization: Rule a is said to generalize rule b if b has higher-priority than a, a and b specify different actions, and every packet that satisfies b also satisfies a. In our policy, rule 12 generalizes rule 9.

Redundancy: Rules a and b are redundant if a and b specify the same action, rule a is higher priority than rule b, and every packet that satisfies b also satisfies a. There are no examples of redundancy in our example.

In the context of [1], anomalies are presented to the administrator for resolution — the anomalies defined above outline potential "problems" in the policy that need to be addressed. Take, for example, the shadowing anomaly between rule 5 and rule 11: rule 5 blocks all packets coming in through port 53, while rule 11 intends to allow these packets if and only if they come from IP 75.75.75.75. During the execution of this policy, rule 11 will never be enforced as the packets that it specifies correspond to the packets specified by higher-priority rule 5. The administrator, however, most likely added rule 11 for some reason, so an appropriate way to handle this anomaly is to notify the administrator to make sure that the over-arching security policy is still being adhered to. Table 3 identifies all anomalies in the example policy.

[1] Our descriptions differ slightly from those in the literature; namely, our definition of redundancy is simpler.

3. Incorporating Argumentation

3.1. Goals

Argumentation applied directly to firewall rules is unlikely to yield any interesting results. The reason for this is straightforward: since there is an order over the rules, any conflicts that arise in rule selection can always be resolved by analyzing the order of the rules. However, a very natural extension of the way that firewalls are specified is to supplement the rules with the reasons for setting them. Such an extension then permits argumentation to be applied to resolve rules on the basis of these reasons, and that is what we investigate here. We will consider three scenarios: a policy where each rule specifies its order value as well as its reason, a policy that only specifies the reasons for each rule, and a policy that only specifies the ordering of the rules. We present a few brief details regarding these scenarios:

Rules with an ordering and reasons: In this scenario, we are given a policy where each rule has a distinct order and is supplemented with the reasons that the rule exists. Our goal in this scenario is to perform anomaly analysis and use the reasons to arrive at strong recommendations for anomaly resolution as well as to identify new anomalies that lie in the reasons themselves. For this paper, we will be primarily working with this scenario.

Rules with reasons but no ordering: Here, our goal is to identify anomalies while treating each rule as having the same priority. Using the reasons, we attempt to arrive at an ordering that minimizes the number of conflicts and anomalies.

Rules with an ordering but no reasons: Here we want to look at the rules and abstract an overall security policy from how they are ordered. To do this, we need to require that "reasons" be defined somewhere and any rule can be compared to this definition — we ultimately guess what the reason may be for each rule and then use the ordering of the rules themselves to arrive at a relative ordering of the reasons, which we treat as a "security policy".

All of these scenarios are realistic from the security point of view but we will only cover the first in this paper. We do this partially in the interest of brevity, but also because a solution to the first scenario can be abstracted out to the second two. This can be done by setting all rules to equal weight for the second scenario and performing anomaly resolution from there (the recommendations are the ordering), or treating the ordering of the rules as "law" in the third scenario. In any case, it seems the critical test of this system will be to ensure its validity for the first scenario.

3.2. System Design

In a nutshell, the system we propose to analyze firewall rules is a multi-domain (or hierarchical) set of value-based argumentation frameworks ([6]). In a rough sense, the ground level has an argument for each rule where arguments attack each other if the corresponding rules share any packets and specify different actions. Each rule/argument has a value associated with it that specifies a domain that that

rule	attacks	rule	attacks
1	3, 7, 12	7	1, 2, 4, 6, 8, 9
2	3, 7, 12	8	7, 12
3	1, 2, 4, 9	9	3, 5, 7, 10, 12
4	3, 7, 12	10	9
5	9, 11	11	5, 12
6	7, 12	12	1, 2, 4, 6, 8, 9, 11

Table 4. All overlaps within the example policy. The rule in the left-hand column attacks the rule(s) in the right-hand column.

rule's reason appeals to. This then leads to higher-level domains which contain arguments to analyze the strength of a specific rule in the previous domain; i.e., if a rule exists "because a sender is trustworthy", then we will have a domain to argue about the trustworthiness of senders where that rule will need to establish its claim. Attacks at the ground domain may represent attacks in a higher domain (such as two rules disagreeing about the trustworthiness of a sender) or an attack between values (one rule appealing to trustworthiness the other to vulnerability). This system is repeated until the domain of reasons runs out.

3.3. Standard Domains

The above description of our system is very high-level and not particularly specific. In order to better describe how this system operates, we work through modeling the example policy in Table 1 by first presenting a few standard domains. At the ground level, we have that each rule attacks each other rule that shares a packet and specifies a different action. We model this in Table 4, where the left hand column is the rule name and the right hand column is the set of rules that the rule in the left hand column attacks.

It's certainly possible to take the above attacks and use argumentation alone to yield some analysis. Such a method, though, is lacking in that the existence of a conflict isn't nearly as important as why that conflict exists — the shadowing of rule 5 over rule 11 is a far different conflict than the correlation of rules 1 and 12. To identify this, we introduce reasons. At the ground level, we have the following reasons: accessibility, prophylaxis, legitimate/malicious sender, enable/disable a protocol, and enable/disable a program. The first two are catch-alls, the former for allowing and the latter for blocking, while the other three yield higher-level domains. We now modify Table 4 to to include the "reason" (or value) of each argument/rule and obtain Table 5.

In a traditional value-based framework [6], there exists an ordering of the values of the arguments that can be used to resolve conflicts. If such an ordering is already in place, then the above work-through is technically sufficient, and any anomalies detected can be resolved by using the preference of values (for example, if "legitimate sender" is more important than "disable protocol", 11 should be put above 5). However, if an ordering is absent/incomplete or we want more in-depth analysis, we need to analyze higher-level domains. The first such domain is the sender domain where we reason about the trustworthiness of senders. This domain can be defined in many ways (such as utilizing trust-computing measures to gauge each sender), but for the purpose of this paper, we treat this domain as being

rule	values	attacks
1	enable protocol	3, 7, 12
2	enable protocol	3, 7, 12
3	malicious sender	1, 2, 4, 9
4	enable protocol	3, 7, 12
5	disable protocol	9, 11
6	legitimate sender, enable program	7, 12
7	malicious sender	1, 2, 4, 6, 8, 9
8	legitimate sender, enable program	7, 12
9	accessibility	3, 5, 7, 10, 12
10	disable protocol	9
11	legitimate sender, enable protocol	5, 12
12	prophylaxis	1, 2, 4, 6, 8, 9, 11

Table 5. Overlap of rules in example policy. Here the left-hand column gives the rule name, the center column gives the reason behind the rule, and the right-hand column gives the rule(s) attacked by the rule in the left-hand column.

very basic — if a rule says that a sender is trustworthy or malicious, we take that rule at its word, with conflicts being resolved by some preferential relationship. The sender conflicts are presented in Table 6(a), with the intra-domain conflicts specified in the right hand column.

The next domain to look at is the protocol domain. Here, we replace the "enable/disable protocol" labels with the actual protocol being acted on — Table 6(b) displays the transformation.

The resulting pruned table is not particularly interesting from an argumentation perspective — protocols will rarely attack each other as they typically specify different ports. The only real exception to this is if one rule specifies "enable protocol" while another does "disable protocol", as in 5 and 11. Conflicts like this would need to appeal to a different level for resolution. One potential method of analysis at the protocol level is to argue that a certain protocol "is typically used for some purpose". As an example, BitTorrent is frequently used for illegal file sharing, while DNS is frequently used for critical network communications. Depending on the context, one might argue that HTTP is used for research, or that HTTP is used for unwarranted recreation. If the administrator pre-defines protocol use like this beforehand, analysis can be done regarding the security of

(a) Analysis of senders.

rule	sender	attacks
3	123.456.78.90	
6	123.456.78.11	7
7	123.456.78.*	6, 8
8	123.456.78.11	7
11	75.75.75.75	

(b) Analysis of protocol.

rule	protocol	attacks
1	FTP data	
2	HTTP	
4	FTP control	
5	DNS	11
10	BitTorrent	
11	legitimate sender, DNS	5

Table 6. Analysis of rules in terms of the senders (a) and the protocol (b) that they refer to. In (b), the protocols were obtained by comparing the ports specified by the rules to the list of registered ports.

rule name	program	rule attacks
6	gaming client	
8	gaming client	

Table 7. Program analysis. While 6, 7 and 8 are in conflict, since 7 isn't a program rule, the analysis is done at a different level.

the ports themselves, abstracting the rules to a higher domain — most likely a value-based framework with the protocols as arguments and typical use as values. This idea of "typically used" can also be applied to higher domains, which we touch on below.

The program domain is a bit more interesting than protocols as a program can require multiple rules. In our example, rules 6 and 8 are facilitating some program, and in order for that program to work, both rules must be present. This highlights our first new anomaly: rules 6, 7 and 8 are out of order. The reason for this is that rule 7 segments rules 6 and 8, which are really part of a larger meta-rule. More specifically, rules 6 and 8 allow a specific sender access via specific ports to allow for the running of a program; however, since rule 7 shadows rule 8, that trustworthy sender will never be able to interact via rule 8, and thus the program won't run. Thus, the correct ordering would be to either completely remove 6 and 8, or put 7 after 8 to allow unmitigated access for sender 123.456.78.11. Note that a resolution of this conflict also resolves the conflicts that were present in the sender domain.

3.4. Abstract Domains:

The aforementioned domains provide a strong basis for firewall policy analysis. The next step is to define the higher level domains, which is going to be more context based as defined by the administrator — instead of trying to outline each possible domain, we present two hypothetical domains that could be potentially useful. The first is the "suite domain" — a suite is some sort of functionality that requires certain properties in the rules. Suites are specified in the form of "if I want a, then I must have $b_0, b_1, ..., b_n$". In the context of our example:

If I want **web browsing**, then I need **HTTP** and **DNS**.
If I want **FTP**, then I need **FTP data** and **FTP control**.

(a)

rule	desired suite	attacks
1	FTP	
2	web browsing	
4	FTP	
6	game client	
8	game client	

(b)

rule	instance of	attacks
1	file transfer	
2	recreation, research	
4	file transfer	
5	network tool	11
6	recreation	
8	recreation	
10	file transfer	
11	network tool	5

Table 8. (a) Rules and their suites. Since no suites are explicitly disabled, there are no direct conflicts on the suite level. (b) Rules related to the instance-of domain.

These two suites present two new anomalies in the policy. The first is that while rule 2 explicitly allows HTTP, rule 5 blocks DNS, thereby disabling the web browsing suite. Working in this domain, then, we can see a new conflict between rule 2 and rule 5 — since rule 5 explicitly blocks DNS, rule 2's desire for web browsing will never be realized. The other new anomaly is between rules 1, 3 and 4: this is similar to the anomaly between 6, 7 and 8. More specifically, the IP blocked by rule 3 will have access to some FTP data but not FTP control, disabling the FTP suite for it. While this may not be a huge violation, it seems that a proper ordering would have rule 3 before rules 1 and 4. Table 8(a) shows the rules with their suites.

The last domain we will address is the "instance of" domain. This domain maps rules, protocols, programs and suites to "ideas" that describe the general purpose of the lower-level objects. For example, the FTP suite and BitTorrent protocol are instances of "file transfer", web browsing is an instance of "recreation", and DNS is an instance of a "network tool". One could argue that network tools should be enabled for the stable operation of a local network, while recreation should be disabled as it unnecessarily slows down more important tasks. Additionally, file transfer can be viewed as important for telecommunication/productivity, while at the same time, it opens the administrator up to liability if things are downloaded illegally or vulnerability if malicious files are installed. Table 8(b) maps protocols and suites to instance of categories. Lastly, the instance of domain can be seen as a larger version of the "typically used" domain that was defined in section 3.3 (the instance of domain applying to programs, protocols, suites, etc., while in 3.3, typically used only referred to protocols).

3.5. Anomaly Resolution

The previous sections dealt almost exclusively with the concept of anomaly identification. While our framework does yield new anomaly discoveries, argumentation does not play a significant role in their identification. Instead, we primarily want to use argumentation to recommend actions regarding how to resolve the conflicts. In this section, we look at how this can be done by examining the new anomalies and then the standard anomalies. The new anomalies that were mentioned in sections 3.3 and 3.4 are reproduced below in Table 9. We present a few potential ways to argue for and against specific paths of resolution.

The "gaming client" anomaly has three potential resolutions: place 6 and 8 above 7, place 7 above 6 and 8, or ignore it. To help the administrator pick the correct solution, we present him or her with the following questions:

- Is the sender 123.456.78.11 more trustworthy than the group of senders 123.456.78.*? (resolving the sender-domain conflict)

desired effect	rules in favor	rules against	domain of attack
enable gaming client	6, 8	7	ground (sender (7) vs. program (6, 8)), sender
allow web browsing	2, 11	5	suite (2, 5), protocol (5, 11)
allow FTP	1, 4	3	ground (sender (3) vs. suite (1, 4))

Table 9. New anomalies.

order	value name
1	allow programs
2	block malicious senders
3	allow legitimate senders
4	block protocols

order	value name
5	block programs
6	allow protocols
7	prophylaxis
8	accessibility

Table 10. A potential ordering of the ground-based values with lower order meaning higher priority.

- Is the program specified by rules 6 and 8 more important than blocking the group of senders 123.456.78.*? (resolving the value-based conflict)

If the answer to either of these questions is a strong yes, then we re-order 7 below 6 and 8, otherwise we may as well remove 6 and 8. In order to assist in answering these questions, we present arguments from the relevent domains: on the sender domain, the argument against the trustworthiness of 123.456.78.* might be that a few bad packets were sent by 123.456.78.90, so the entire range is being blocked, while the argument for allowing 123.456.78.11 could be as simple as "that sender is a trusted relative". In this case, the answer to the questions would be that yes, the specific sender is more trustworthy than the range, so we should prioritize 6 and 8 before 7. On the flip side, we could note that "game client" is used for recreation, and if the administrator previously claimed "reducing vulnerability is always more important than recreation", then we would have evidence to remove 6 and 8. Ultimately, the administrator will need to make the decision, and if the arguments for both sides are presented in a clear manner, we can be more confident in the security of the final outcome. As a last remark, the shadowing anomaly between rules 7 and 8 will be resolved pending the resolution to the "game client" anomaly — here, by extending the firewall model to include these reasons, we've introduced an argumentation-based scheme that helps to resolve the traditional anomalies.

There are three main ways to resolve the web browsing anomaly: ignore it, remove rule 5, or place rule 11 before rule 5. Resolution here depends on the following questions: (1) Is blocking DNS more important than allowing web browsing? (the suite-domain conflict), and (2) Is blocking DNS more important than allowing the legitimate sender 75.75.75.75? (the value-based conflict).

If the answer to both these questions is no, then we should either remove rule 5 or order rule 11 above rule 5. If the answers are both yes, then we should remove rule 2 and rule 11. If the answers are a mixture, then we'd need more information: if 75.75.75.75 is in fact shown to be malicious in the sender domain, and we know that 75.75.75.75 is the only possible DNS server, then DNS should remain blocked and rules 2 and 11 should be removed. In the case that web browsing is more important than blocking DNS and 75.75.75.75 is a legitimate sender, then rule 11 should go above rule 5. When presented to the administrator, arguments can appeal to ideas such as "DNS (as a protocol) is usually secure and should be allowed" or "network tools are usually secure" or "recreational activities should never take priority over security" or "75.75.75.75 is a trusted and secure DNS server". When the administrator chooses how to fix this problem, having a web of relationships and "big picture" ideas can help him or her best conform the firewall

rules in conflict	anomaly name	recommendation	justification
(5, 11)	shadowing	place 11 before 5	allow sender > block protocol
(7, 8)	shadowing	place 8 before 7	allow program > block sender
(1, 3), (1, 7)	correlation	place 3, 7 before 1	block sender > allow protocol
(2, 3), (2, 7)	correlation	place 3, 7 before 2	block sender > allow protocol
(3, 4)	correlation	ignore	block sender > allow protocol
(3, 9)	correlation	ignore	block sender > accessibility
(4, 7)	correlation	place 7 before 4	block sender > allow protocol
(5, 9)	correlation	ignore	block protocol > accessibility
(7, 9)	correlation	ignore	block sender > accessibility
(9, 10)	correlation	place 10 before 9	block protocol > accessibility
(1, 12), (2,12), (4, 12)	generalization	ignore	allow protocol > prophylaxis
(6, 7)	generalization	ignore	allow program > block sender
(6, 12), (8, 12)	generalization	ignore	allow program > prophylaxis
(9, 12)	generalization	remove 9	prophylaxis > accessibility
(11, 12)	generalization	ignore	allow sender > prophylaxis

Table 11. Anomalies and their corresponding recommendation based on the ordering in Table 10.

policy to the overall security policy. Lastly, like the above anomaly, resolution here would also resolve the shadowing between 5 and 11.

We will not perform an in-depth analysis for resolution of the FTP anomaly, instead only remarking on a few arguments that can be made in favor of prioritizing 3 before 1 and 4: "file transfer is easily exploitable to send malicious data", "malicious senders should not be given access to exploitable systems", "123.456.78.90 has been known to exploit the FTP control protocol", etc. It might also be the case that the administrator is actually managing a file-sharing site, so allowing everyone access to FTP is crucial. Again, as with the other anomalies, we rely on the administrator to make the final decision, but ensure that he or she has enough access to the background information to make an informed decision.

Resolution of the standard anomalies can be done in a manner similar to the above. However, if the policy being analyzed is very large, then such a process might be tedious for the administrator. Instead, we note that resolution of these conflicts tends to revolve around questions of the form "is a preferred to b?" — with this in mind, our goal for standard anomalies might be to recommend general policy rules that remove or ignore the detected conflicts. As an example, we see in Table 1 that rule 12 generalizes all "allow" rules; if prophylaxis were a number-one priority, it's possible that that rule should be shadowing all of the allowed ones. If, conversely, prophylaxis isn't particularly important, the rules are correct as they are. Resolution of standard anomalies like this should correspond to resolution of unresolved conflicts between values in each domain. In Table 10 we present a possible ordering of ground-level values, and in Table 11 we present a recommendation for each anomaly and the reason for that recommendation.

4. Related Work

Due to the widespread deployment of firewalls across all fields, it's no surprise that there's a large amount of literature on the subject of firewall policy analysis.

Many of these systems use some sort of analysis on the rules themselves: [14] uses static analysis, treating the rules as a program, while [9] uses a data mining technique. Policy analysis, in this regard, is frequently viewed in the "anomaly" sense similar to the Al-Shaer definitions; in [7], they introduce a variation that includes multi-rule anomalies. Another extension of policy analysis is role-based systems, which can be seen in [10].

More related to our framework are systems that utilize some form of high-level security policy that low-level rules are related to. Common themes among these systems are either generation of a high-level policy from low-level rules (as in [12]), or generation of low-level rules from a high-level policy (as in [5]). Perhaps most noticeable is [4], where the authors seek to use argumentation to help generate the low-level rules from high-level (and possibly conflicting) policies. In fact, the authors of [4] have another work, [3], where they use argumentation in the context of anomalies and policy analysis — our framework differentiates itself from these last two through the introduction of "reasons", which is intended as a somewhat hierarchical and generic expression of high-level policy.

In the context of argumentation, our paper does not provide any particularly new concepts, taking from the general idea of frameworks in [8], and more directly, the value-based framework in [6]. In fact, reasoning about higher-level policy via the preferencing of values is an idea that can be seen in [11], where they provide meta-arguments to reason about preference levels. Additionally, the idea of multiple domains of argumentation is an adaptation of the hierarchical model in [13] — ours differs slightly in the relationship of the domains, however, this idea can most likely still be modeled in their framework.

5. Conclusions and Future Plans

The above framework is still some way from implementation. More work needs to be done to better outline and diagram the actual domains: the sender domain needs a concrete metric, the protocol domain should be pre-defined, syntaxes are needed for suites and programs, and mapping into the instance of domain should be straightforward. Additionally, languages are needed to quickly view arguments that attack protocols (the "typically used" idea) or instances (i.e., attacking file transfer via vulnerability). Once this is done, there are other scenarios that we would like to apply our framework to, including generating high-level policy from the rules, and generating rules from high-level policy. We've also been considering models that involve multiple agents arguing about a semi-distributed firewall policy, using a hierarchical model for the agents and arguments. Still, even incomplete, the framework provides new insight and strong potential for a new application of argumentation. With more work, a tool that realizes this system can be designed to help system administrators securely govern their networks; such a tool would be essential in an environment that has high turnaround of IT personnel to help standardize and learn security policy. In any case, this design should at the very least serve to strengthen the idea that argumentation has a wide range of applicable scenarios, firewall policy analysis among them.

Acknowledgements: Research was partially funded by the National Science Foundation (CNS 1117761) and the Army Research Laboratory and Cooperative Agreement Number W911NF-09-2-0053. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory, the National Science Foundation, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

[1] E. Al-Shaer and H. Hamed. Firewall policy advisor for anomaly discovery and rule editing. In *IFIP/IEEE Eighth International Symposium on Integrated Network Management*, pages 17–30, 2003.

[2] E. Al-Shaer, H. Hamed, R. Boutaba, and M. Hasan. Conflict classification and analysis of distributed firewall policies. *Selected Areas in Communications, IEEE Journal on*, 23(10):2069 – 2084, oct. 2005.

[3] A. K. Bandara, A. Kakas, E. C. Lupu, and A. Russo. Using argumentation logic for firewall policy specification and analysis. In *Proceedings of the 17th IFIP/IEEE international conference on Distributed Systems: operations and management*, DSOM'06, pages 185–196, Berlin, Heidelberg, 2006. Springer-Verlag.

[4] A. K. Bandara, A.C Kakas, E. C. Lupu, and A. Russo. Using argumentation logic for firewall configuration management. In *Proceedings of the 11th IFIP/IEEE international conference on Symposium on Integrated Network Management*, IM'09, pages 180–187, Piscataway, NJ, USA, 2009. IEEE Press.

[5] Y. Bartal, A. Mayer, K. Nissim, and A. Wool. Firmato: a novel firewall management toolkit. In *Security and Privacy, 1999. Proceedings of the 1999 IEEE Symposium on*, pages 17 –31, 1999.

[6] T. J. M. Bench-Capon. Persuasion in practical argument using value-based argumentation frameworks. *Journal of Logic and Computation*, 13(3):429–448, 2003.

[7] T. Chomsiri and C. Pornavalai. Firewall rules analysis. In *Proc. of The 2006 International Conference on Security and Management*, June 2006.

[8] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77:321 – 357, 1995.

[9] K. Golnabi, R.K. Min, L. Khan, and E. Al-Shaer. Analysis of firewall policy rules using data mining techniques. In *Network Operations and Management Symposium, 2006. NOMS 2006. 10th IEEE/IFIP*, pages 305 –315, april 2006.

[10] J. D. Guttman. Filtering postures: Local enforcement for global policies. In *Proceedings, 1997 IEEE Symposium on Security and Privacy*, pages 120–129. IEEE Computer Society Press, 1997.

[11] S. Mogdil and T. Bench-Capon. Integrating object and meta-level value based argumentation. In *Proceedings of the 2008 Conference on Computational Models of Argument: Proceedings of COMMA 2008*, 2008.

[12] A. Tongaonkar, N. Inamdar, and R. Sekar. Inferring higher level policies from firewall rules. In *Proceedings of the 21st conference on Large Installation System Administration Conference*, pages 2:1–2:10, Berkeley, CA, USA, 2007. USENIX Association.

[13] M. Wooldridge, P. McBurney, and S. Parsons. On the meta-logic of arguments. In *Proceedings of the fourth International Conference on Autonomous agents and Multiagent systems*, AAMAS '05, 2005.

[14] L. Yuan, H. Chen, J. Mai, C. Chuah, Z. Su, and P. Mohapatra. Fireman: a toolkit for firewall modeling and analysis. In *Security and Privacy, 2006 IEEE Symposium on*, pages 199. –213, may 2006.