Firewall Rule Anomaly Detection and Resolution using Particle Swarm Optimization Algorithm

John Kingsley Arthur
Valley View University
Computer Science Dept.
Accra-Ghana

Edward Kwadwo Boahen
Jiangsu University
Comp. Sci and Tel. Engineering
Zhenjiang, China

Ronky Francis Doh
Valley View University
Computer Science Dept.
Accra- Ghana

Eric Appiah Mantey
Jiangsu University
Comp. Sci and Tel. Engineering
Zhenjiang, China

ABSTRACT
The firewall ensures the protection of a network by falling on some number of defined rules set by the administrator of the computer network. Managing these rules to be optimum without errors is very difficult and sometimes leads to the formation of anomalies such as redundant, correlation, and shadowing rules. This defined problem has received the attention of both the academic and industry players in finding a pragmatic solution. A lot of reasonable attempts has been made by researchers of which many resorted to the automation of the firewall rule management process. The automation is to aid determine and resolve the conflicting rules and also to reduce the load that will be on the network administrator, which almost always leads to the creation of contradictory rules. The existing literature has not focused much on the amount of time it takes to determine and resolve these anomalies. Most of the conflicting rules are as a result of the wrongful position[index] a rule may occupy in the rule list. The research proposes a contextual design of an improved firewall framework, that rest on the heuristic approach of the Particle Swarm Optimization (PSO) Algorithm to determine and assign the best position [index] to a rule and thereby improving the search and resolution of identified anomalies in a firewall rules list. Three (3) lightweight algorithms are designed for anomaly detection and resolution using PSO as the backbone.

General Terms
Firewall, Conflicting rules, Particle Swarm Optimization

Keywords
Firewall rule management, Network Security, Particle Swarm Optimization Algorithm

1. INTRODUCTION
Firewalls are essential in the provision of security to a network. Companies and individuals who value the information that resides on their network invest hugely into keeping their data safe. Firewalls have been widely implemented in defending suspicious network packet traffic [1] and unauthorized access to Internet-based enterprises. The firewall does reside in-between the public and private network regulating the packets that are shared across the two(2) networks. The firewall contains rules that make it possible to filter incoming and outgoing packets and to take a decision to accept, deny, or discard a packet [2]. The ability of the firewall to perform excellently well depends on how well these rules are set by the administrator of the network. Keeping these rules tidy and optimized all time is tedious and cumbersome work, especially when done manually. It becomes more difficult, even as the size of the network grows. This brings up instances of oversight of the administrator that leads to some unexpected rule anomalies and network performance flaws such as rule conflicts or overlapping, improper rule alignment, and rule generalization, including redundant rules. By these rule inconsistencies, legitimate packets that need to go through the network successfully may be denied. This undoubtedly means the illegitimate data packets may be instead allowed into the network that is to be protected. Giving this background there is, therefore, an imminent need to critically pay attention to how firewall rules are created and maintained in the firewall by the network administrator.

These inconsistencies in rule management have caught the attention of many researchers and have, therefore come up with prepositions and recommendation of theories and practices on how to detect and resolve the said anomalies. Majority of the literature [1]-[6],[8]-[11], and [13] considered the detection and resolution by automating of rule management processes. Very few, such as [12] studied the visualization of rules and their interrelations. [4] and [5] considered purely rule relations and how it influences the reordering and total performance of the firewall. [7] and [8] made an enormous effort by improving the performance of the firewall rule, but the emphasis was on the reordering of rules, which is not enough. Very few have considered the time involved in detecting and resolving the anomalies. The time taken to detect and correct an anomaly is very critical to the total efficiency of the firewall and its working functionality. A more effective method must be chosen to successfully identify and classify rules into their respective anomalies and subsequently resolved within the shortest possible time. The arrangement of the rules in the rule list is sequential and likened to a database's order of rows. The execution of the rules then becomes sequential. The topmost rule is executed before the next. If frequently accessed rules are put at the button, this will perceptibly elongate the time it will take the control to that index of rule for execution. The position a rule is slotted very important. Some rules may never be executed because of the position they occupy in the rest list. Given this, a rule must occupy the best position for it to be shortly identified for execution. A technique to do this will suggest improvement on time. The Particle Swarm Optimization Algorithm is proven to be very fast to reach its optimum solution as against the genetic algorithm and therefore, will be used in this case to improve the outcome of the existing methods.

The research proposes a contextual design of an improved firewall framework, that rest on the heuristic approach of the Particle Swarm Optimization (PSO) Algorithm to determine and assign the best position [index] to a rule and thereby improving the search and resolution of identified anomalies in a firewall rules list. Also, three (3) algorithms are designed to

perform detection and classification of anomalies into redundant, shadowing, and correlated rules. The algorithms will run simultaneously. Furthermore, the research work proposes a robust theoretical framework for the implementation of the PSO firewall anomaly detection and resolution system. The classification of rules will mean focus will be on only the cluster where rule anomaly is assigned and not the entire anomaly rule set.

The subsequent part of the manuscript is fragmented into six (6) sections. Section 2, 3, 4, 5, 6, and 7. Where section 2, the existing works are discussed, and also, the gaps are constructively stated. In section 3, Firewalls and firewall rule anomalies are defined using a visual example. In section 4, the theoretical basis of PSO is explained. In section 5, the application of PSO in the detection and resolution of the anomalies are shown, including three (3) proposed algorithms. In section 6, the detail of how system concept can be implemented is highlighted. Section 7 is the very last part where the recommendation for next improvement of work is stated, including the conclusion.

2. RELATED LITERATURE

Firewalls play a crucial role in keeping safe a network from an unsafe network. Several research works have been done on this subject matter to recommend new and improved approaches to detect and resolve the anomalies. In the research work of [2], authors sought to resolve conflicting anomalies by proposing a new paradigm of the firewall design consisting of two parts; the Single Domain Decision firewall (SSD) and Binary Tree firewall(BTF). The SSD is a new firewall rule management policy that ensures that rules certainly do not conflict. On the other hand, the BTF is a data structure and an algorithm to fast check the firewall rules for the presence of an anomaly.

The genetic algorithm has also been used as an alternative method in optimizing the performance of the anomaly detection. In the research work of [14], the authors used the Genetic algorithm to find the optimal rule order that minimizes the average number of rule comparisons while maintaining integrity. The Genetic algorithm goes through several iterations at stages of the selection, crossover, and mutations before population optimization is realized. The higher the number of iterations, the higher the amount of time it takes for a population to be optimized. In other words, the number of iterations will influence the time to identify and resolve the anomalies.

The works of [1],[10] and [13] all focused on detecting and resolving anomalies by using the rule-based segmentation technique. [13] presents a grid representation of the rules and also identifies their overlapping associations. However, it does not define how the identified overlapping packet set can be resolved. No technique as well is deployed by their research to measure the amount of time it takes for the segmentation to and associations to be established amongst the rules. An improvement of [13] is done by [1]. In the work of [1], the major focus was to propose a framework that will detect and resolve conflicting and redundant firewall rules. In their work, they used the technique of a segmentation rule management framework, which depends heavily on the Apriori algorithm to build the associations between the rules. However, according to [15] the Apriori algorithm takes excessive time to formulate associations between rules. The weakness of the Apriori algorithm will affect the time to optimize the rules in the rule set, especially when there are more rules in the firewalls database. [10] used the same segmentation technique but focused only on redundant rule detection and resolution. It was their work that introduced redundancy removal software called Firewall Anomaly Management Environment (FAME). In FAME, a straightforward approach is used, where various components of the rules are compared and if found same, then it is declared as redundant else classified as some other anomaly. The research work does not consider how to identify the other anomaly cluster and how to remove or resolve them. This makes the outcome of FAME not wholly devoid of anomalies.

In the research work of [4], they sought to propose an algorithm to get the best case for solving the conflict and shadowing rules. In their proposed range algorithm, IP address is divided into the specified range, then rules are compared with the divided IP from the first stage, and then a table is generated from the result. Rules from the tables are now compared with other rules to detect the existence of an intersection between a source and destination address between rules with same decisions. Rules are extracted when there is an intersection, else added since they are disjoint. In an attempt to make sure all the rules are independent of each, all redundant and shadowing rules are removed. Some error logs are sent to the network administrators console for resolution.

Reordering is one of the approaches used in firewall optimization. In the work of [7], the Heuristic Approximation Algorithm is used to optimize the rule list in combination with the Rule-based segmentation for firewall anomaly identification. Anomalies are identified by the rule-based segmentation technique; however, the research is silence on how the redundant anomaly is going to be corrected. The optimization is well elaborated to improve the systems total performance.

3. FIREWALL AND FIREWALL ANOMALIES

Firewall rules can be described as the translation of firewall policies into the actual configuration. More often, firewall rules are organized of a condition and action based on the condition. Network packets that arrive at the firewall are matched and tested against the condition of the rules which results in either an acceptance or denial to or from the network. Anytime, there is a match; the subsequent rules are automatically skipped. Hence the order in which the firewall rules are arranged is essential. If a rule is not well situated can lead to severe anomalies. Policies, on the other hand, are defined as an abstract, high-level definition of traffics that is to be allowed through a network and those that are not to be allowed. Firewall policies are mostly specified as a sequence of the rule referred to as Access Control List (ACL) which can contain inconsistencies. These inconsistencies are known as anomalies. Firewall rule anomalies refer to any inconsistency in firewall rules that results in either allowing unwanted traffic to enter or leave the network or deny passage to legitimate traffic. The firewall rule most of the time, has the syntactical structure defined as:

<Ruleid><Protocol><Source_ip><Source_port>
<Destination_ip><Destination_port><Action>..... (i)

Each attribute can be defined as a range of values, which can be represented and analyzed as sets.

With a given rule list, R, each policy will be expressed as

R={r1,r2,r3,…rn} ……..…….. (ii)

where each element of r, is a tuple having the structure as defined in (i) above.

3.1 Identification of Anomalies

Assumption(s)

We assume a generic name of a rule called v1.

The current position of a rule is cp.

R= Rule, **P**=Protocol, **SP**=Source port, **D_ip**= Destination IP, **D**=Destination Port, and **Act**=Action as used in the headings of tables 1 and 2.

Table 1. Table captions should be placed above the table

R	P	Source ip	sp	D_ip	D	Act
R1	TCP	30.1.1. *	80	100.168.1. *	40	Allow
R2	UDP	20.1. *.*	40	120.168.1. *	50	Allow
R3	UDP	20.1.4. *	40	120.168.1.4	50	Deny
R4	TCP	20.1.1.21	*	100.168.1. *	*	Allow
R5	TCP	30.1.1. *	80	100.168.1. *	40	Allow
R6	TCP	20.1.1.21	*	100.168.1. *	50	Deny

3.1.1 Shadowing Anomaly

For a rule **r** to be shadowed by another rule v1, the index value of r in the rule list is greater than that of the v1 such that:

r.cp>v1.cp……..... (ii)

and also

iff r U v1= v1 or v1 U r = v1 ...……..…(iii)

This situation, as indicated in (ii) and (iii) above, will result in the fact that r is never activated. A typical situation is depicted in table 1 above. where r2 shadows r3. This is because r2 allows every UDP packet that is coming from port 40 of IP, 20.1.*.* to the port 50 of IP, 120.168.1.* for which is to be denied or discarded by r3.

3.1.2 Correlation Anomaly

In the perspective of rule correlation, a rule, r, correlates with v1 iff $r \cap v1 = v1'$ and/or $r \cap v1 = r'$ but r.action $\neq$ v1.action…….. (iv)

This implies that some network packets (v1′) may be allowed or denied through the network. Similarly, r′ may be allowed or denied based on its position. In other words, the intersection of r and v1 are either allowed or denied by preceding rule. From the above table, r4 correlates with r6, and all TCP packets coming from any port of IP 20.1.1.21 to port 50 of IP 100.168.1.* matches the intersection of these rules. Since r4 is a preceding rule of r6 in the rule list, every packet within the intersection of these rules is denied by r4. However, if their positions are exchanged, the same packets will be allowed.

3.1.3 Redundant Anomaly

The rule, r, is said to be redundant if and only

if r = v1 ……. (v)

where all members of **r** are the same values as v1 such even if either r or v1 is removed, it will not affect the protection of the network. For example, r1 is redundant concerning r2 in Table 1, since all TCP packets coming from port 80 of 10.1.2.* to the port 40 of 100.168.1.* matched with r1 can match r5 as well with the same action.

4. THE THEORETICAL CONCEPT OF PARTICLE SWARM OPTIMIZATION

The Particle Swarm Optimization (PSO) algorithm is a population-based search algorithm based on the simulation of the social behavior of birds within a flock [16]. The original purpose of the Particle Swarm concept was to visually simulate the beautiful and uncalculable movement of a bird flock, with the intent of determining patterns that guide the ability of birds to fly together and to suddenly change direction with regrouping in an optimal formation. From this initial aim, the concept evolved into an easy and methodical optimization algorithm.

In PSO, people additionally alluded to as particles, are "flown" through hyperdimensional search space. Changes to the situation of particles inside the pursuit space depend on the social-mental inclination of people to mimic the accomplishment of different people. The progressions to a particle inside the swarm are along these lines influenced by the experience, or information, of its neighbors. The search characteristic of a particle is thus influenced by that of other particles within the swarm (PSO is, therefore, a kind of symbiotic cooperative algorithm). The effect of modeling this social behavior is that the search process is such that particles stochastically move toward previously successful positions in the search space.

The PSO method keeps up a swarm of particles, where every particle represents a potential solution. In relationship with evolutionary computation models, a swarm is like a populace, while a particle is like a person. In straightforward terms, the particles are "flown" through a multidimensional search space, where the situation of every molecule is balanced by its understanding and that of its neighbors. Let xi(t) represent the position of particle i in the search space at time step t; unless otherwise stated, t denotes discrete time steps. The position of the particle is changed by adding a velocity, vi(t), to the current position, i.e.

xi(t + 1) = xi(t) + vi(t + 1)……..…….. (vi)

with xi(0) ~ U(xmin,xmax).

It is the velocity vector that controls the optimization mechanism and reflects both the experiential knowledge of the particle and socially exchanged information from the particle's neighborhood. The experiential learning of a particle is by, and large alluded to as the subjective component, which is corresponding to the separation of the particle from its very own best position (alluded to as the particles' personal best position) found since the first run. The socially traded data is alluded to as the social part of the velocity equation.

For the global best PSO, or gbest PSO, the neighborhood for each particle is the entire swarm. The social network employed by the gbest PSO reflects the star topology. For the star neighborhood topology, the social component of the particle velocity update reflects information obtained from all the particles in the swarm. In this case, the social information is the best position found by the swarm, referred to as yˆ(t).

For gbest PSO, the velocity of particle i is calculated as

vij(t + 1) = vij(t) + c1r1j(t)[yij(t) − xij(t)] + c2r2j(t)[yˆj(t) − xij(t)] ……. (vii)

where vij (t) is the velocity of particle i in dimension j = 1, . . . , nx at time step t, xij(t) is the position of particle i in dimension j at time step t, c1 and c2 are positive acceleration constants used to scale the contribution of the cognitive and

social components respectively, and r1j (t), r2j (t) ~ U (0, 1) are random values in the range [0,1], sampled from a uniform distribution. These random values introduce a stochastic element to the algorithm.

The personal best position, yi, associated with particle i is the best position the particle has visited since the first time step. Considering minimization problems, the personal best position at the next time step, t + 1, is calculated as

$$y_i(t+1) = \begin{cases} y_i(t) \text{ if } f(x_i(t+1)) \geq f(y_i(t)) \\ x_i(t+1) \text{ if } f(x_i(t+1)) < f(y_i(t)) \end{cases} \quad \ldots(viii)$$

where f : Rnx → R is the fitness function. As with EAs, the fitness function measures how close the corresponding solution is to the optimum, i.e. the fitness function quantifies the performance, or quality, of a particle (or solution).

The global best position, yˆ(t), at time step t, is defined as

$$\hat{y}(t) \in \{y0(t),\ldots,yns(t)\} | f(\hat{y}(t)) = \min\{f(y0(t)),\ldots,f(yns(t))\} \ldots(ix)$$

where ns is the total number of particles in the swarm. The global best position can also be selected from the particles of the current swarm, in which case

$$\hat{y}(t) = \min\{f(x0(t)),\ldots,f(xns(t))\} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (x)$$

5. APPLICATION OF THE PSO IN THE DETECTION AND RESOLUTION OF THE ANOMALIES

5.1 Representation of a Particle as a Rule

Each particle is represented as a one-dimensional matrix with seven (7) attributes representing the rules id, protocol, source ip, destination ip, source port, destination port, and action as shown in the table below.

Table 2. Representation of a Particle

R	P	Source ip	SP	D_ip	D	Act

5.2 Initial Solution

The initial solution for Anomaly Detection is obtained by random initial rule attributes (id, protocol,ip_source,source_port,destination_ip,destination_port, action) of each rule; a matrix is employed in recording the attributes and the anomaly detection information of a rule, and the determination of flow of initial solution is as follows.Set up the number of firewall rule, and generate an empty matrix for the initial attributes of the rule.Based on randomly and evenly distributed manner, generate the relation between rules, into the matrix.Call the Anomaly detection function; determine the anomaly for each rule, into the matrix.Determine the correctness, the fitness value of the initial particle.

5.3 Anomaly Detection Algorithm

The Anomaly Detection Algorithm is made up of three (3) different subroutines. The redundancy detection algorithm, the correlation detection algorithm, and the shadowing detection algorithm.

5.4 Formulation of Algorithms

The algorithm maintains two lists of rules, previous_list, and current_list. Previous_list contains original firewall rules with its configurations whiles the current_list comprises the newly anomaly free firewall rules.

Rule r refers to rules yet to be inserted into current_list
Rule s refers to rules already inserted into current_list
R_action refers to the action constraint of rule r.
S_action refers to the action constraint of rule s.

5.4.1 Redundancy Anomaly Detection

If Prevoius_list = ϕ

current_list=current_list+r

If r ==s && r_action== s_action (Determination of

the fitness)

Update the individual optimal solution and the

global optimal solution using (x)

Update the speed vector by using (vi)

Update the speed vector by using (vii)

else If r!=s

current_list=current_list+r

endif

endif

5.4.2 Correlation Anomaly Detection

If Prevoius_list = ϕ
current_list=current_list+r

else if r ∩ s==true && r.action ≠ s.action (Determination of fitness)

Update the individual optimal solution and the

global optimal solution using (x)

Update the speed vector by using (vi)

Update the speed vector by using (vii)

else

current_list=current_list+r

endif

endif

5.4.3 Shadowing Anomaly Detection

If Prevoius_list = ϕ

current_list=current_list+r

If r.cp>s.cp && r U s= s or s U r = s (Determination

of fitness)

Update the individual optimal solution and the

global optimal solution using (x)

Update the speed vector by using (vi)

Update the speed vector by using (vii)

else

current_list=current_list+r

endif

endif

5.4.4 The Reordering Function

Reordering is essential when focusing on improving the total performance of the rule list. The order in which the rule is placed will influence how much time it will take to be fetched for matching against an incoming or outgoing network packet.

To reorder, each rule, r, is given a weight based on the number instances it has been accessed. The rule with the highest value moves to the very top and in that order follows subsequently to the bottom that which has the least.

5.4.5 Integration of the Algorithms

empty. When the firewall rule list is empty; any rule created will be devoid of an anomaly. The anomalies begin to show only when there is an existing rule. In the subsequent sections; a rule can be added to the existing rules; if and only if it does not correlate, shadow nor become redundant to an existing rule

An optimal value of the population of the rules(particles) is determined. The check and classification, including the resolution of anomalies, are done while the population has not gotten to its optimal solution.In this section how the three(3) algorithms by design are to work together is shown.

function get_rule()
{

Get rule, rule_list, previous_list, current_list

}

function get_Optimumlvalue()

{

optimum_solution;

}

while (population!=get_OptimumValue())

{

get_rule()

if Prevoius_list = ϕ

current_list=current_list+r

print rule_list

else

{

if(r ==s && r_action== s_action)!

{

if(r ∩ s==true && r.action ≠ s.action)!

{

if(r.cp>s.cp && r U s= s or s U r = s)!

{

current_list=current_list+r

Update the individual optimal solution and

the global optimal solution using (x)

Update the speed vector by using (vi)

Update the speed vector by using (vii)

get_Optimumvalue()

Print rule_list}}}

}}

5.4.6 Flowchart of Integrated Algorithm

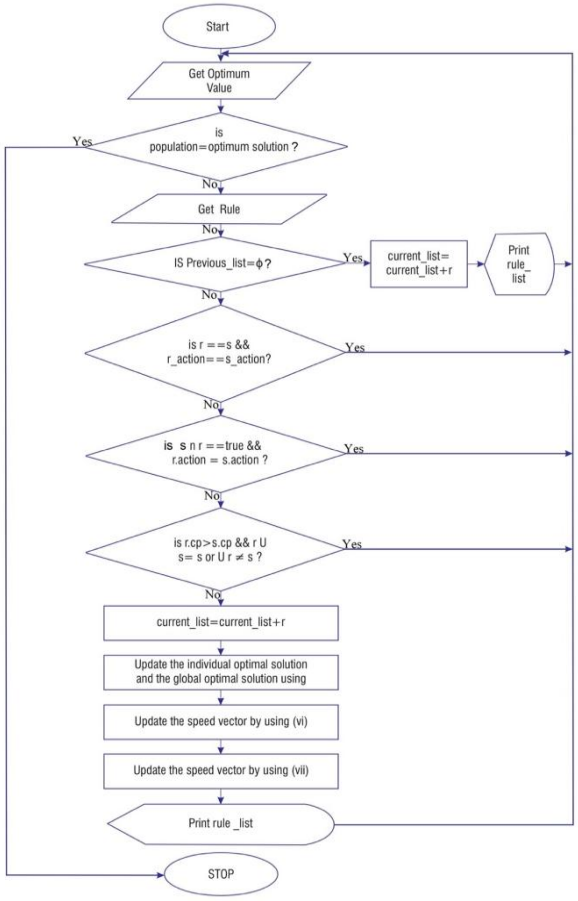

Fig 1: Flowchart of Integrated Algorithm

6. THEARCHITECTURAL FRAMEWORK FOR IMPLEMENTATION

6.1 Phase 1

In the first phase of the system implementation, the network administrator creates the rules for the firewall. All the rules are submitted to the Rule Optimizer residing on the server as shown in fig 2 below. The rules that created are based on available resources of the private network and the function the network must achieve. Network users are anticipated to have access to the resources on the network or outside of the network based on the policies that will be set.

6.2 Phase 2

In this phase, the rule optimizer[integrated algorithms] performs an analysis on all the rules to identify anomalies. The anomaly detection is done by invoking the Redundancy, Shadowing, and Correlation Anomaly Detection Functions as described in Section V

6.3 Phase 3

Firewall log files are submitted to the Optimizer to determine which of the rules are frequently used. This is to help reorder

the rules in the rule set. The frequently used rules are brought to the very top.

6.4 Phase 4

In the very final phase, the network administrator receives a report of the refined rules.

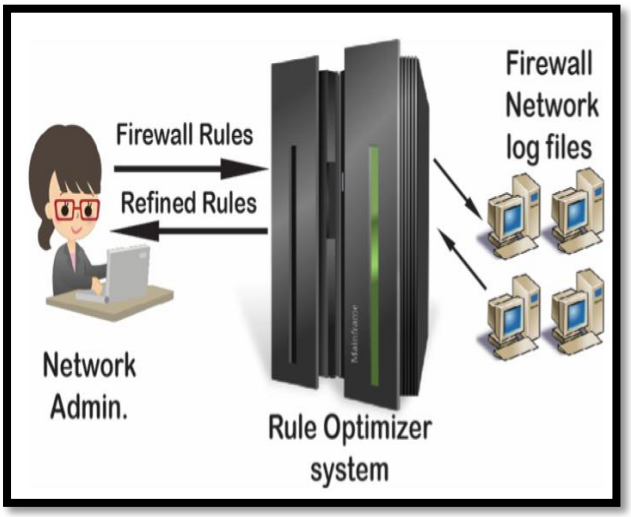

Fig 2: Architectural Framework for System Implementation

7. CONCLUSION AND DIRECTION FOR RECOMMENDATION FOR FUTURE WORKS

In this research work, the gaps in the existing literature regarding rule anomaly detection and resolution is identified. Most literature has concentrated on literally automating the rule management processes. However, this research uses a heuristic approach in an attempt of reducing the amount of time taken to determine and resolve firewall anomalies.

Based on the Particle Swarm Optimization Algorithm, three (3) lightweight algorithms are designed to determine and resolve firewall rule anomalies (such as redundant rules, shadowing rules, and correlation rules). Also, some implementation issues are discussed to guide how the system could be implemented. The proposed algorithm is yet to be implemented where the real values of its efficiency (big O Notation) will be determined. Subsequent work will consider making use of a live firewall rule dataset to perform analytics to see how the concept will perform in real time.

8. REFERENCES

[1] Darade, R. V. and Kumbharkar, P.B. "Firewall policy anomaly detection and resolution", An International Journal of Advanced Computer technology, (June, 2014). Volume III, Issue VI in COPUSOFT

[2] Khummanee, S., Khumseela, A., Puangpronpitag, S. 2013. Towards a New Design of Firewall: Anomaly Elimination and Fast Verifying of Firewall Rules, in 10th International Joint Conference on Computer Science and Software Engineering (JCSSE).

[3] Abedin, M., Nessa, S., Khan, L., and Thuraisingham, B., "Detection and Resolution of Anomalies in Firewall Policy Rules", in Damiani E., Liu. P(eds) Data and Applications Security XX. DBSec 2006. Lecture Notes in Computer Science, vol 4127. Springer, Berlin, Heidelberg.

[4] Farouk, A., Agiza, H. N., and Radwan, E. "Detecting inconsistent firewall configuration rules using range algorithm" in International Conference on Machine Learning and Computing IPCSIT vol.3, IACSIT Press, Singapore.

[5] Swamy, D. K., Narender, T. "Improvements in Firewall Policy Rules to Identifying and Resolving Anomalies" in International Journal Of Advanced Research and Innovations, Vol.1, Issue .9

[6] Chandre, P. R., Surve, R.R., Badhan, S. R., Surve, A. B., Mane, V. T. "Anomalies of Firewall Policy Detection and Resolution" in International Journal of Engineering Research and Applications, ISSN: 2248-9622, Vol. 4, Issue 3(Version 1), March 2014, pp. 696-701

[7] Kachare, S. S., Deshmukh, P.K. "Firewall Policy Anomaly Management with Optimizing Rule Order", in International Journal of Application or Innovation in Engineering & Management (IJAIEM), ISSN 2319 – 4847, Volume 4, Issue 2, February 2015.

[8] Jitha, C. K., Namboodiri, S. "Firewall Policy Anomalies-Detection and Resolution", in International Journal of Computer Trends and Technology (IJCTT), Volume 4, Issue 7, July 2013

[9] D. Hemkumar, M. Chugh, "Methods for Firewall Policy Detection and Prevention", in International Journal of Science, Engineering and Technology Research(IJSETR), Volume 3, Issue 7, July 2014.

[10] Sethuram, J., and Sankareeswari, G. "Redundancy Management a nd Anomaly Detection o n Firewall Ruleset u sing Fame", in International Journal of Science Technology & Engineering, Volume 1, Issue 10, April 2015. ISSN(online): 2349-784X.

[11] Prasath, A. Y., and Revithi, N.,"Dynamic Rule based Interfirewall Optimization using Redundancy Removal Algorithm", in International Journal of Computer Applications. Volume 92, No. 6, April 2014

[12] Hongxin, H., A. Gail-Joon, A., and K. Kulkarni, "Detecting and Resolving Firewall Policy Anomalies", in IEEE Transactions On Dependable And Secure Computing, Vol. 9, NO. 3, MAY/JUNE 2012

[13] Anbarasan, A., Balasubramani, G., Madhan, C., Naveenkumar, P., and N.S. Nithya, "Detecting and Resolving Firewall Policy Anomalies Using Rule-Based Segmentation", in International Journal of Computer Science and Mobile Computing(IJCSMC), Vol. 2, Issue. 4, April 2013, pg.134 – 137.

[14] El-Alfy, E. M. "A Heuristic Approach for firewall Policy Optimization", in ICACT 2007.

[15] Al-Maolegi, M. and Arkok, B. "An Improved Apriori Algorithm For Association Rules", in International Journal on Natural Language Computing (IJNLC). Vol. 3, No.1, February 2014.

[16] Engelbrecht, A. P. "Computational Intelligence: An Introduction", in John Wiley & Sons Ltd. 2nd Edition. 2007.