IJREAT International Journal of Research in Engineering & Advanced Technology, Volume 1, Issue 3, June-July, 2013
**ISSN: 2320 - 8791**
**www.ijreat.org**

# Enhanced Network Intrusion Detection System

**Sanu Skaria[1], Robin Abraham[2] , Ajith Kurian Issac[3]**

[1]**Electronics and Communication Department, MG University, Muvattupuzha, Kerala, India**

[2] **Electronics and Communication Department, MG University, Muvattupuzha, Kerala, India**

[3]**Wipro Technologies,VLSI Division, Kochi, Kerala, India**

**Abstract**

Network intrusion detection systems plays an important role in today's networking because of the increase in network traffic and due to the growing number of network attacks. Many software and hardware approaches are proposed to implement network intrusion detection system. State traversal algorithm based on memory architecture has attracted a lot of attention because of its easy re-configurability and scalability. But the state traversal algorithm doesn't provide any solution for similar states present in certain patterns. It is often referred to as loop back problem. This paper proposes a method which successfully eliminates the loop back problem and achieves the very high memory efficiency.

***Keywords:*** *State Traversal Algorithm, Pattern, Finite State Machine, Loop Back Problem*

## 1. Introduction

Network intrusion detection systems mostly place at strategic points in a network, so that it can monitor the traffic travelling to or from different devices on that network. An efficient intrusion detection system should successfully eliminate threats with minimal hardware requirements. State traversal algorithm based on pattern matching technique successfully eliminates the network attacks. Also it offers very high memory efficiency compared to its predecessors like AC algorithm [2]. As the amount of attack patterns increasing day by day, the system need to accommodate all of them. This creates a huge memory overhead for storing and processing these patterns. So it is required to adopt a memory efficient practice. The only exception to this in state traversal algorithm is the loop back problem while merging pseudo equivalent states [1].

In this paper we propose a simple and efficient way to resolve the loop back problem existing in the state traversal algorithm [1]. The modified hardware architecture will resolve the loop back problem existing in the state traversal algorithm. We successfully implemented the modified architecture and found to be effective in resolving the loop back problem. With this modification the memory efficiency can further increased which results in even less memory usage than the state traversal algorithm. The simulation experiments results shows reduction in memory usage.

## 2. Overview and Problem Description

Network intrusion detection systems are generally implemented using finite state machines. In State traversal algorithm, state transition graph for the attack patterns are developed initially. To reduce the memory usage, similar states are merged to form a single state. For example consider fig 1 which is the state diagram for matching two string patterns "pxyz" and "jxyk". Here states corresponding to "x and y" ie states 2, 6 and states 3, 7 are similar states because they have identical input transitions, identical failure transition, identical if_Final bit and different next states [1].

After merging similar states the new state diagram will look like fig 2. Here for the new state 26 and 37 the pa_Vec and if_Final bits are updated by taking the union of the corresponding states pa_Vec and if_Final [1].Considering the large number of virus patterns going to be implemented in the system, this method of merging similar states achieves high memory efficiency by reducing the number of states to implement the state machine. In addition, the concept of pre_Reg ensures that the state machine will detect only correct patterns [1].

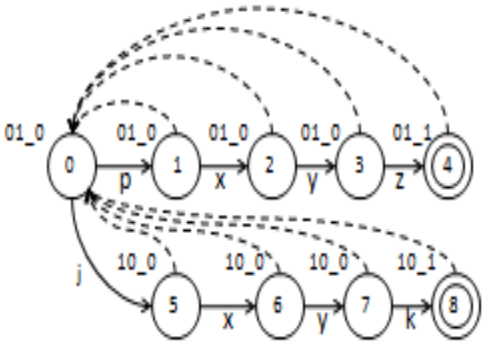



Fig. 1 State diagram of "pxyz" and "jxyk"

## 2.1 Loop Back Problem

If two patterns have more than one set of similar pattern subsets between them, the state machine may produce incorrect decisions due to loop back problem. In such a situation merging cannot be done between states. Consider two patterns "pbcdek" and "jdebcs" and state diagram for them is shown in fig3. Here states corresponding to "bc" and "de" are similar. So after merging similar states the state machine will look like fig 4.

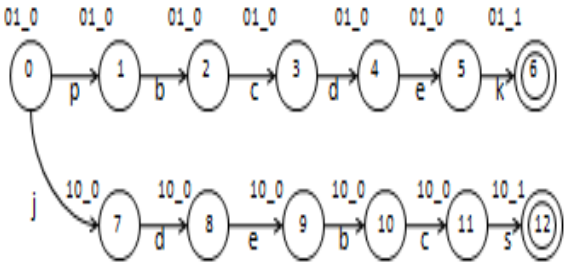
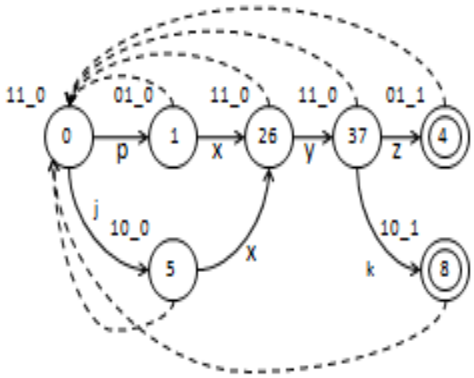



Fig. 3 State diagram of "pbcdek" and "jdebcs"



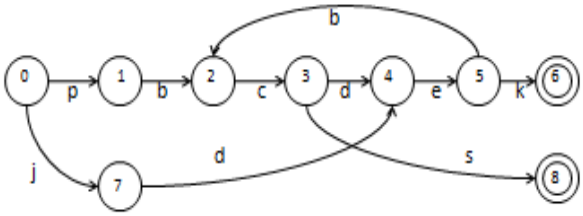




Fig 4.New state diagram after merging similar states

Fig. 2 State diagram after merging similar states

State machine in fig 4 may produce false positive results due to loop back problem. If the input pattern is "pbcdebcdek", then the state machine reaches state 6 which is the final state of pattern "pbcdek". This is because the above merging has caused looped transitions in the state diagram. That means a non virus pattern is mistaken as a virus pattern. In this case merging is not possible.

## 3. Problem Solution and System Architecture

Fig 5 is the modified hardware architecture for solving loopback problem in state traversal algorithm. The proposed hardware architecture will eliminate the loopback problem by introducing a new circuitry for the if_Final generation logic. A pattern is said to be detected in state traversal algorithm when the if_Final bit goes high. In the proposed modified hardware architecture we have altered the way the module raises its if_Final bit. As long as if_Final bit is low a pattern is said to be never detected. In the

new method we have introduced additional logic so as to make sure a wrong pattern is never detected and it also allows efficient merging of all the pattern types. Consider the above example for loop back problem. In fig 6 input string "pbcdebcdek" will result in a false detection as explained earlier. In order to fix this we have included a new block called n_valid detector and counter circuit. The circuit will detect assertion of n_valid signal and the counter continues to count it. The n_valid signal acts as an enable for the counter. The counter will be reset to zero whenever the n_valid is deasserted or when count reaches its maximum count value or when system reset is applied. After reaching the final state, the module will check whether the n_valid count

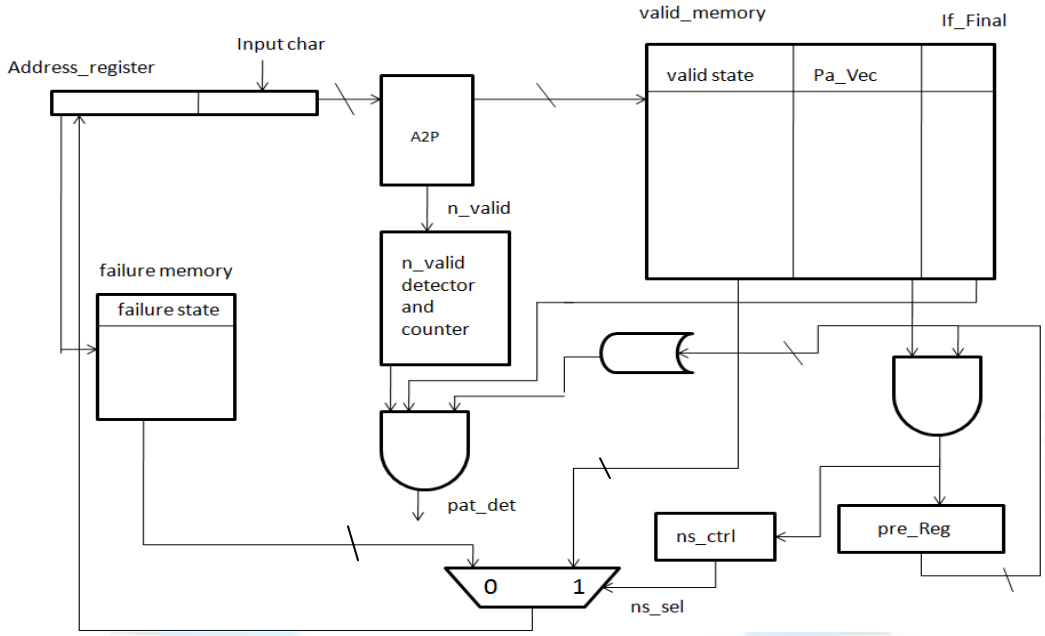



Fig. 5 Hardware architecture

value is same as the pattern length that is 6 in this case. If yes, the if_final is ANDed with the output of the counter module to generate a new signal pat_det. If that is high the pattern is detected. In the above example the pattern length is 6 but for the pattern "pbcdebcdek" the pattern length is 10. So after reaching final state module will check for the count value against the pattern length that is 6 which is not equal to pattern length, the pat_det is not asserted and it is not detected as a pattern .The AND gate also has a bit wise ORfed output of the pre_Reg as one of the input. This will make sure that at the end of the pattern, the pre_Reg contents are also nonzero. This ensures that only valid patterns are detected.

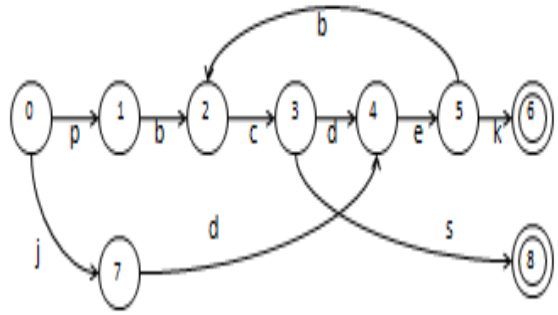



Fig. 6 The merged state diagram for the following patterns"pbcdek" and "jdebcs" with loop back

## 4. System Description

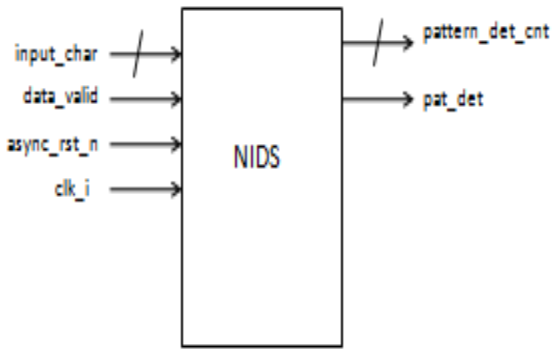



Fig. 7 Block Diagram

### 4.1Clock and Reset for the Module

The design is working on a periodic clock with 50% duty cycle and is fed through a module pin called clk_i. The module uses an asynchronous reset. Upon reset the module registers will reset to its initial state

### 4.3A2P (Address to Position)

This unit of the module will take the output of the address register and based on the value of the register it generates the corresponding memory address in which the next state transition is stored.

### 4.4 Valid Memory

The module has a memory block which stores the state information on particular address. The output address from A2P will point to a particular location in the memory block. The contents are then used to extract valid state, pa_Vec and if_Final bits based on the incoming patterns.

### 4.5 Pre_Reg

The output from the memory consists of pa_Vec along with next state and "if_Final". The pa_ Vec is taken from the memory and is stored in a register called "pre_Reg". The value in the pre_Reg is then anded with the previous value during the transition.

### 4.6 Ns_Cntrl Block

The ns_cntrl block is used to select between failure state and valid state based on the "n_valid" and the output from "pre_Reg". This determines what should be the value in the address register depending on whether it is valid state transition or invalid transition.

### 4.7 Failure State

When an input pattern comes and it is not producing a valid transition, then the address register should be updated with failure state. For that whenever failure transition occurs, the failure state is updated. This is taken care by the failure state memory.

### 4.8 N_valid Detector and Counter Circuit

Whenever a valid state transition occurs the n_valid bit goes low. The detector circuit will detect whenever the n_valid goes low and generates the enable for the counter circuit. The counter circuit counts the number of clock cycles the n_valid remains low. The counter will be reset when the count reaches its maximum value. Whenever the last state is reached the the circuit will check whether the count value is equal to the pattern length, then only it will assert the pat_det signal.

## 5. Result

The hardware architecture is coded in verilog and simulated using modelsim. The observed waveform result is shown in fig 8.

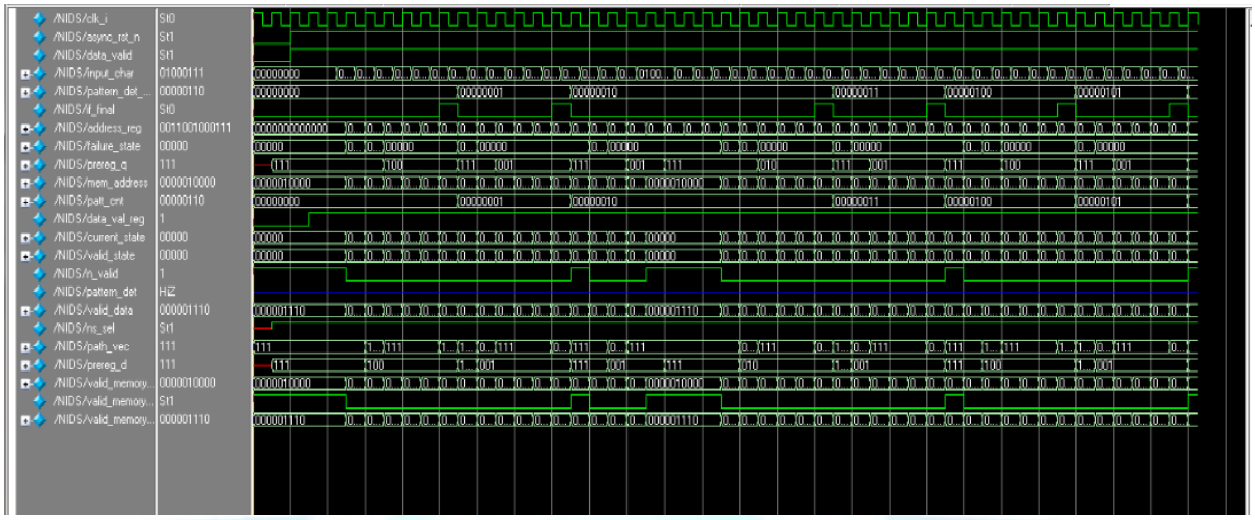



Fig.8 Simulation result showing loop back problem is completely eliminated.

## 6. Conclusion

In this paper, we presented a new approach for overcoming loop back problem. The proposed method achieves high memory efficiency with minimum hardware complexity.

## References

[1]Cheng-Hung Lin, and Shih Cheih,"Efficient pattern matching algorithm for memory architecture",IEEE transactions on very large scale integration systems (VLSI) ,vol.19, No.1, January 2011.

[2] Vassilis Dimopoulos, Ioannis Papaefstathiou, and Dionisios Pnevmatikatost,"A Memory-Efficient Reconfigurable Aho-Corasick FSM Implementation for Intrusion Detection Systems", International Conferenceon DigitalObjectIdentifier: 10.1109/ICSA MOS.2007.4285750,Publication Year:2007 , Page(s): 186 – 193.

[3] Brodie,R Cytron, and D. Taylor, "A scalable architecture for high-throughput regular-expression pattern matching," in Proc. 33rd Int. Symp. Comput. Arch.(ISCA),2006,pp.191–122.