ON THE CORRECTNESS OF FIREWALL POLICY DEPLOYMENT

M. EL MARRAKI, A. KARTIT

Department of Computer Sciences, Faculty of Sciences, University of Mohamed V
Rabat, Morocco

ABSTRACT

Firewall policies can contain several thousand rules due to the large size and complex structure of modern networks. The size and complexity of these policies require automated tools providing a user-friendly environment to specify, configure and safely deploy a target policy. In this paper, we show that naïve deployment approaches can easily create a temporary security hole by permitting illegal traffic or interrupt service by rejecting legal traffic during the deployment. We make some contributions to the correctness of firewall policy deployments and we show that the category of type I policy editing is wrong and could lead to security vulnerabilities. We then provide a correct algorithm for publishing political class type I. Our algorithm can be used even for the deployment of policies whose size is very important.

Keywords: *Policy Deployment (PD), Firewall Policy Management (FPM), Network Security (NS).*

1. INTRODUCTION

A firewall is a system that protects a computer or computers network intrusions from another network (such as internet). The firewall system contains rules that essentially permit [1]: (i) to authorize the connection (enable), (ii) to block the connection (deny). All of these rules are the policy that underlies the functioning of the firewall. Due to the large size and complexity of networks today, the politics become large (about 600000 rules) and complicated [2]. Also, the design of a policy is a very complex task for an administrator. Indeed, we should take into consideration a large number of cases to avoid cases of access.

In addition, an administrator may want to configure in real time an active policy to replace it with a new policy. This configuration is still problematic because it must reconcile the continued service and avoid security breaches. The ordered list of operations to be applied to achieve a new configuration is particularly sensitive. As a result, these policies require automatic tools for providing a right environment to specify, configure and deploy security policy target. Much research has dealt with the specification {[3], [4], [5]} policies, conflict detection {[6], [7], [8]} and the optimization problem {[9], [10]}, but very few studies have interested in the deployment of policies. That is why we have tried to focus on problems associated with the deployment of policies to make it easier for network administrators.

The deployment of a firewall policy should have the following characteristics [2]: Correctness, confidentiality, security and speed. Only recently, some researchers have proposed deployment strategies for two important categories of publishing policies [2]. Our work is focused on language editing policy type I. We will demonstrate that the algorithm ”Scanning Deployment” already proposed is wrong and we propose another version of the algorithm which is correct and will allow us to replace a source policy with a target policy.

2. POLICY DEPLOYMENT

A firewall policy deployment should have following characteristics [2]: correctness, confidentiality, safety, and speed.

Correctness: A deployment is correct if it successfully implements the target policy on the firewall. After a correct deployment the target policy becomes the running policy. Correctness is an essential requirement for any deployment.

Confidentiality: Confidentiality refers to securing the communication between a management tool and a firewall. Due to the sensitive nature of information transmitted during a deployment, the communication between management tool and firewall should be confidential.

Safety: A deployment is safe if no legal packet is rejected and no illegal packet is accepted during the deployment.

Speed: A deployment should be done in the shortest time, so that the desired state of affairs is achieved as quickly as possible. A deployment algorithm should have a good running time, so that it is applicable even for large policies.

Different firewalls support different policy editing commands. The set of policy editing commands that a firewall supports is called its policy editing language.

In [2], the authors classify policy editing languages into two representative classes, Type I and Type II, and provide deployment algorithms for both types of languages. Type I editing supports only two commands, append and delete. Command (app r) appends a rule r at the end of the running policy R, unless r is already in R, in which case the command fails. Command (del r) deletes r from R, if it is present. As Type I editing can transform any running policy into any target policy [2], therefore it is complete. Most older firewalls and some recent firewalls, such as FWSM 2.x and JUNOSe 7.x, only support Type I editing.

Indeed, the deployment algorithm type I used is called "*Scanning Deployment*".

2.1. The Algorithm "Scanning Deployment"

Algorithm 1: Scanning Deployment (already existed) [2]

```
Scanning_Deployment (I, T) {
 /* An algorithm using only app
and del to transform policy I into
policy T */

 S ← empty stack
 H← empty hash table
/* Phase 1: add rules */
 i ← 1
 for t ← 1 to SizeOf(T) do
 while i ≤ SizeOf(I) and I[i]<>
T[t] do
 /* I[i] needs to be deleted */
 S. PUSH (I[i])
 H.ADD (I[i])
 i ← i + 1
 if i > SizeOf(I) then
 if H.Contains(T[t]) then
 H.Remove(T[t])
 IssueCommand( del T[t])
 IssueCommand( app T[t])
 /* Phase 2: clean up */
 for j ← SizeOf(I) down to i do
 IssueCommand( del I[j])
 while not S.IsEmpty() do
 r ← S.POP()
 if H.Contains(r) then
 IssueCommand( del r)
 }
```

I[i]: is the i^{th} rule of the original policy. In the real case can be replaced for example by "permit TCP 200.168.1.1 12.3.4.0/24 23".

Shortcoming: Phase 2 of the algorithm does not give good results.

We will show this through a sample run.

See Figure1 and Figure2.

We completed the first phase with i = 9 and t = 13, Sizeof(I) = 8, so we will never run the loop:

```
for j ← SizeOf(I) down to i do
IssueCommand (del I[j])
```

After running phase 2, the algorithm gives the following result: *See Figure2*

It is therefore clear that H is different from T. Therefore, the algorithm is not correct.

2.2 Our Contribution

We start by giving a simple deployment algorithm for an initial policy I and target policy T that will allow us to correct the algorithm "scanning deployement". I and T are coded as arrays of characters, so that I[i] refers to the i^{th} rule of I. Initially, the running policy H equals I. In phase1, the algorithm appends to the end of H every rule r in T, starting from r = T [1]. If r is already in I, then it removes r from H before appending it back. In phase 2, it removes from H every rule r that is in I but not T. the new algorithm is called: "Enhanced_Scanning_Deployment"

Algorithm 1: Scanning Deployment (new release)

```
Enhanced_Scanning_Deployment (I,T)
{
/* an algorithm using only app and
del to transform policy I into
policy T */
H← empty hash table
/* Phase 1: add rules */
 i←1
 for t←1 to SizeOf(T) do
 while ((i<=SizeOf(I)) AND
(I[i]<>T[t])) do
 /* I[i] needs to be deleted */
 H. ADD(I[i])
 i ← i + 1
```

```
end while
 if (i>SizeOf(I)) then
      if (H.Contains(T[t])) then
      H.Remove(T[t])
      IssueCommand(del T[t])
     end if
      IssueCommand(app T[t])
end if
end for

/* Phase 2: clean up */
K←sizeof(I)+sizeof(T)-sizeof(I∩T);
j←1
While (k>sizeof(T)) do
 t←1, trouve←false
While ((t<=sizeof(T)) AND
(trouve=false)) do
If (H(j)=T(t)) then
j←j+1
trouve←true
else
t←t+1
end if
end while
If (trouve=false) then
  Issuecommand(del(H(j))
  K←k-1
end if
end while
}
```

This algorithm gives good results whatever the size of the original and target policy.

We will show this through the previous example. *See Figure3, Figure4, Figure5, Figure6, Figure7 and Figure8.*

Having finished the execution of the algorithm ”Enhanced_Scanning_Deployment”, policy being implemented is identical to the policy target. Therefore, we can say that this new version of the Algorithm is correct.

3. CONCLUSION

In this paper, we showed how unsophisticated approaches to the deployment of policies may temporarily accept unwanted traffic and prohibit trafficking desirable. Up to now, approaches to unsafe deployment is still practiced by management tools firewall. We showed, through examples, that the policy language edition type I is not accurate but we could make it correct through the changes we have made on the algorithm “Scanning Deployment”. We will be soon working on language editing Type II policies to make deployment very effective, safe and fast.

REFRENCES:

[1] S. Karen and H. Paul, “Guidelines on Firewalls and Firewall Policy”, NIST Recommendations, SP 800-41, July, 2008.

[2] C. C. Zhang, M. Winslett, and C. A. Gunter, **“**On the Safety and Efficiency of Firewall Policy Deployment”, In SP ’07: Proceedings of the 2007 IEEE Symposium on Security and Privacy, pages 33-50,Washington, DC, USA, 2007.

[3] E. Al-Shaer and H. Hamed, ”Modeling and Management of Firewall Policies”, Network and Service Management, IEEE Transactions on, 1(1):2-10, April 2004.

[4] Y. Bartal, A. J. Mayer, K. Nissim, and A.Wool. Firmato, “A Novel Firewall Management Toolkit”, In IEEE Symposium on Security and Privacy, pages 17-31, 1999.

[5] M. G. Gouda and A. X. Liu, “Firewall Design: Consistency, Completeness, and Compactness”, In ICDCS, pages 320-327, 2004.

[6] F. Baboescu and G. Varghese, “Fast and Scalable Conflict Detection for Packet Classifiers”, In ICNP, pages 270-279, 2002.

[7] Z. Fu, S. F.Wu, H. Huang, K. Loh, F. Gong, I. Baldine, and C. Xu, “IPSec/VPN Security Policy: Correctness, Conflict Detection, and Resolution”, In POLICY, pages 39-56, 2001.

[8] A. X. Liu, “Change-impact analysis of firewall policies”, In ESORICS, pages 155-170, 2007.

[9] H. Hamed and E. Al-Shaer, “Dynamic rule-ordering optimization for highspeed firewall filtering”, In ASIACCS, pages 332-342, 2006.

[10] J. Qian, “ACLA: A framework for Access Control List (ACL) Analysis and Optimization”, booktitle = Proceedings of the IFIP TC6/TC11 International Conference on Communications and Multimedia Security Issues of the New Century. page 4, Deventer, The Netherlands, The Netherlands,2001.

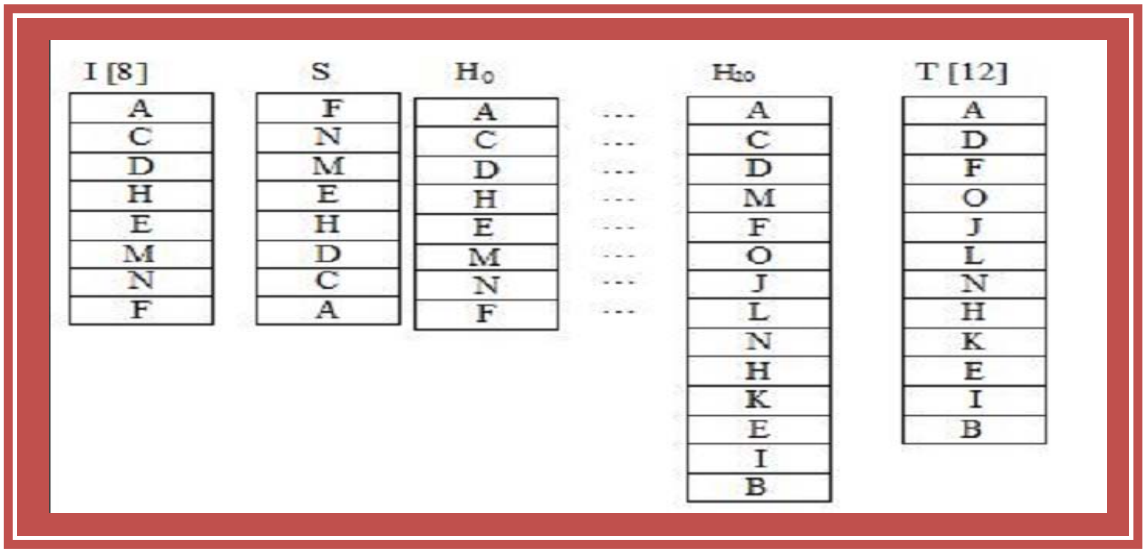

FIGURE 1: SCANNING_DEPLOYMENT PHASE 1 RUNNING EXAMPLE

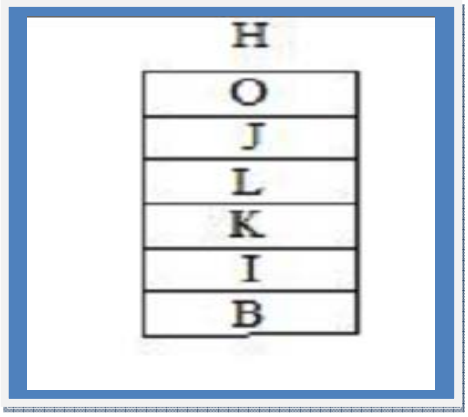

FIGURE 2: SCANNING_DEPLOYMENT PHASE 2 RUNNING EXAMPLE

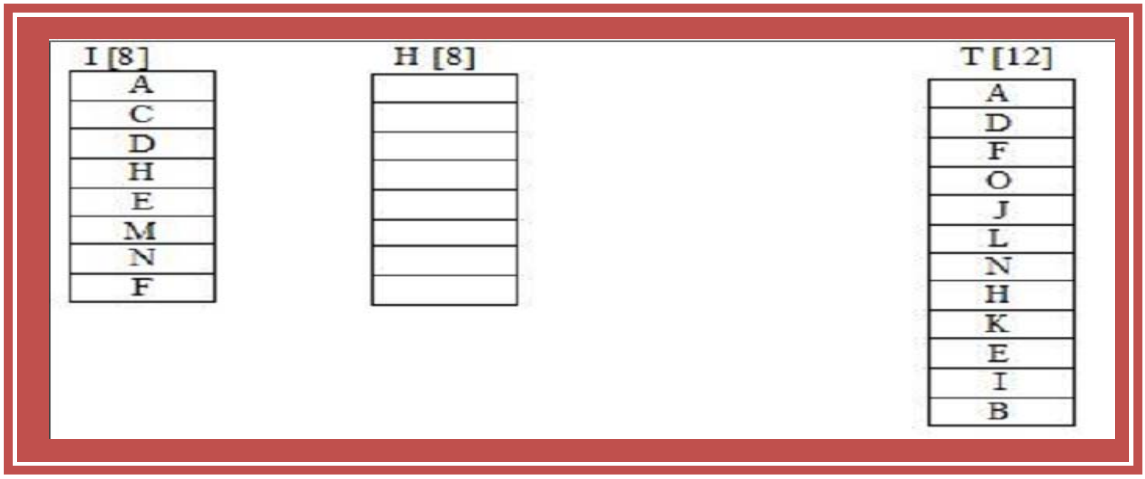

FIGURE 3: ENHANCED_SCANNING_DEPLOYMENT PHASE 1 RUNNING EXAMPLE (1)

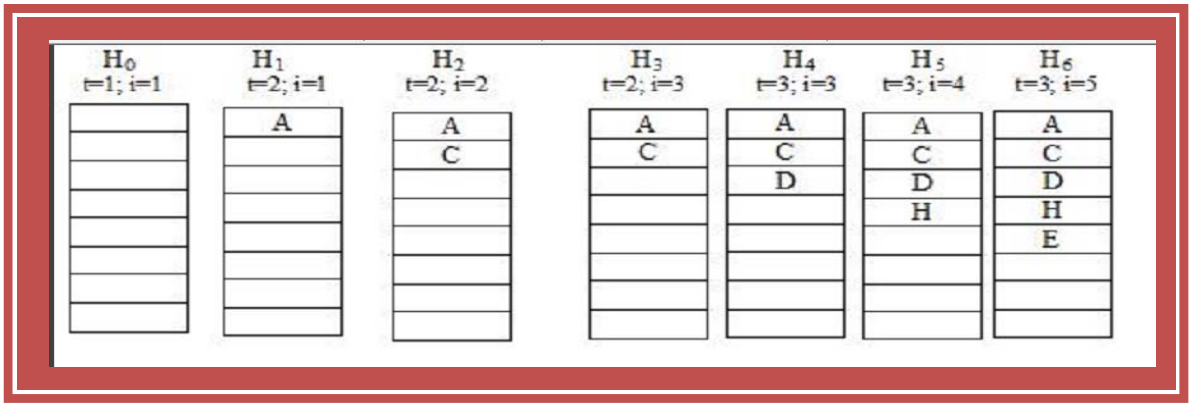

FIGURE 4: ENHANCED_SCANNING_DEPLOYMENT PHASE 1 RUNNING EXAMPLE (2)

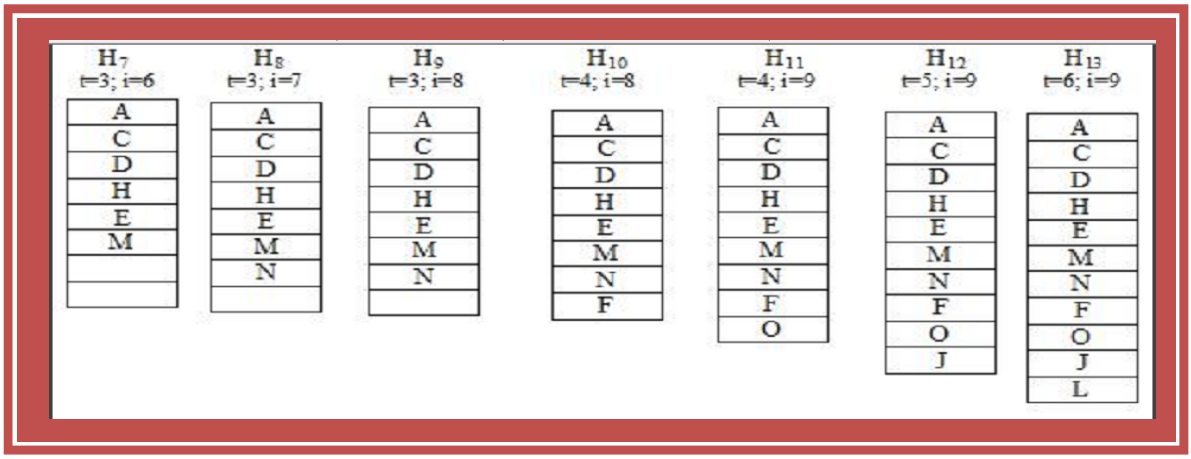

FIGURE 5: ENHANCED_SCANNING_DEPLOYMENT PHASE 1 RUNNING EXAMPLE (3)

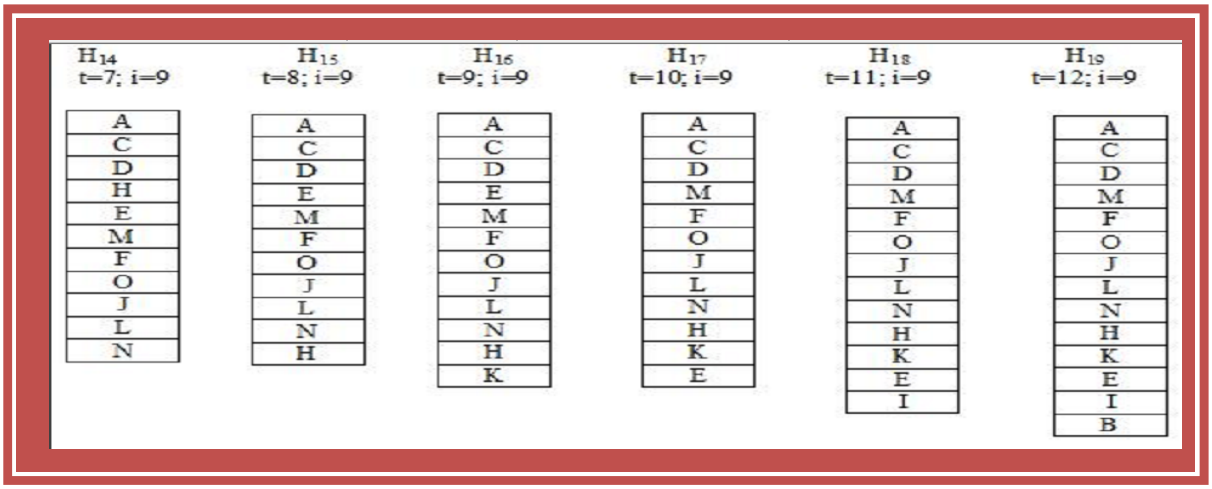

FIGURE 6: ENHANCED_SCANNING_DEPLOYMENT PHASE 1 RUNNING EXAMPLE (4)

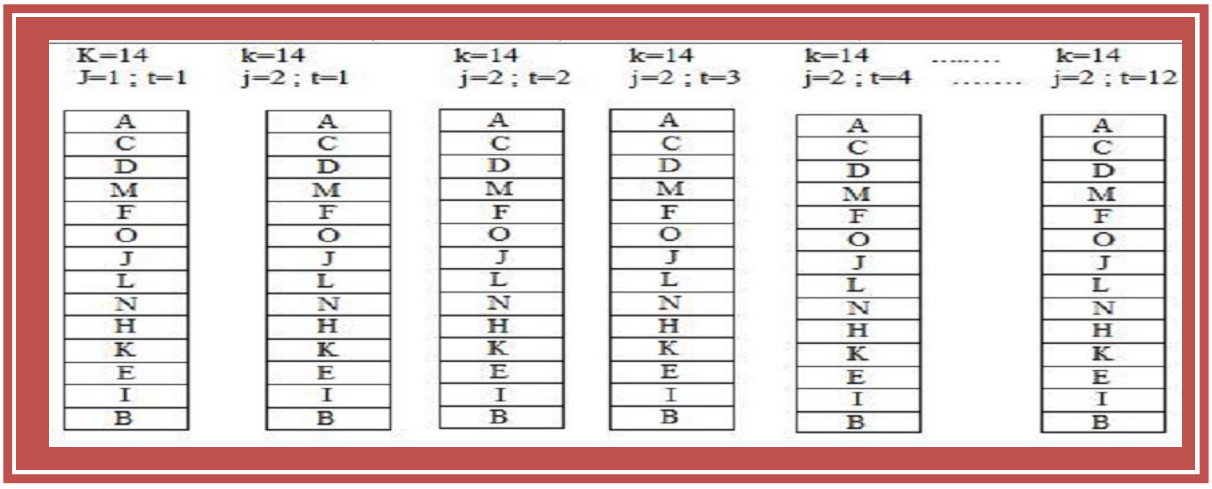

FIGURE 7: ENHANCED_SCANNING_DEPLOYMENT PHASE 2 RUNNING EXAMPLE (1)

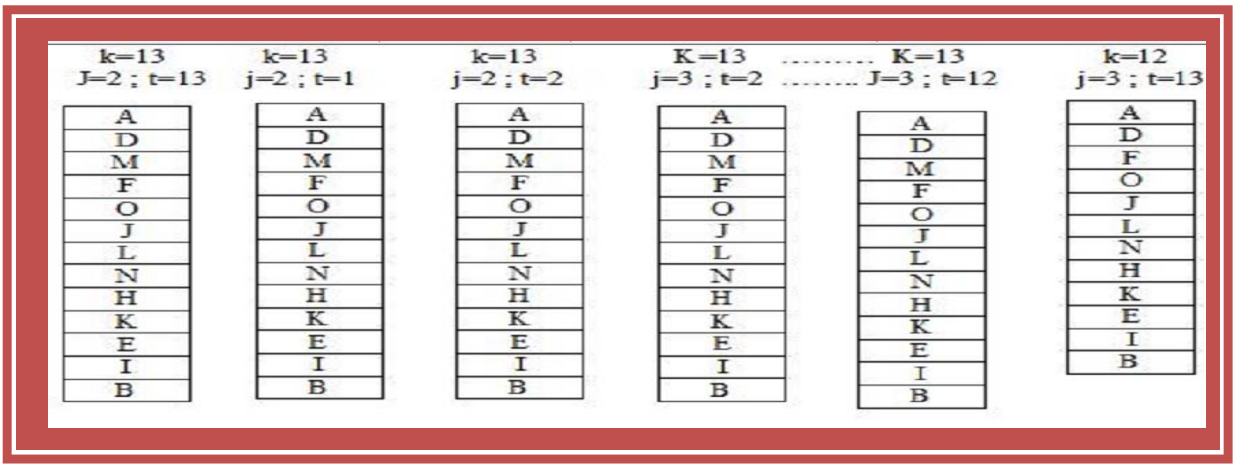

FIGURE 8: ENHANCED_SCANNING_DEPLOYMENT PHASE 2 RUNNING EXAMPLE (2)