[11] O b i t k o M., *Introduction to Genetic Algorithms*. http://cs.felk.cvut.cz/ xobitko/ga, 1998.

CU PRIVIRE LA OBŢINEREA UNEI CATEGORIZĂRI OPTIMIZATE ÎN SPAŢIILE CONCEPTUALE UTILIZÂND ALGORITMI GENETICI

(Rezumat)

Se propune o nouă abordare privind minimizarea sau maximizarea distanţei dintre prototipurile unor clase pentru a obţine o diferenţiere categorială eficientă, sau o generalizare bazată pe valorile comune ale domeniilor de cunoştinţe. Folosirea algoritmilor genetici se dovedeşte utilă in special in cazul domeniilor multidimensionale neechilibrate. Rezultatele obţinute sunt in concordanţă cu aşteptările teoretice şi cu rezultatele furnizate de alt algoritm de ponderare a trăsăturilor.

BULETINUL INSTITUTULUI POLITEHNIC DIN IAŞI
Tomul XLIX (LIII), Fasc. 1-4, 2003
AUTOMATICĂ şi CALCULATOARE

FIREWALL GENERATOR (I)

BY
EUGEN PETAC and BOGDAN MUŞAT

Abstract

Linking of a computer to the Internet imples, generally, the use of the Linux operating system and of the TCP/IP protocol suite. These components have their own security problems. Access to the Internet also implies the use of a set of dozens of services, programs, with various security problems, either due to software errors or due to mis-incorporation of some appropriate security facilities. Iptables is used to configure, administer and inspect the packet filtering rules tables from the Linux kernel. Each of the defined tables can contain a number of predefined chains but it can also contain user-defined chains. The novelty that the *Firewall Generator* application (built in PHP language) brings is the possibility of dynamic manipulation of firewall rules through a web interface. The *tcpServer* program written in C listens to requests at address 127.0.0.1, port 1 500.

Keywords: Linux operating system, TCP/IP protocol suite, Iptables, tcpServer.

1. Introduction

A firewall is a system or a group of systems that imposes an access control policy between two networks. The means through which this imposing is accomplished varies pretty much, but, basically, the firewall can be thought as a pair of mechanisms: one that blocks the traffic and one that allow traffic. Some firewalls put great accent on blocking the traffic, while others put accent on allowing traffic. Probably the most important thing in a firewall is to implement an access control policy. If it is not known precisely what type of access must be permitted or forbidden, a firewall won't help anybody. It is important to acknowledge that a firewall configuration, because it represents a mechanism for strengtheing security policy, imposes its policy on everything that is behind him.

Firewall administrators that protect a large number of hosts have therefore a large responsibility.

1.1. Understanding Firewalls

There are firewalls for the majority of the operating systems (Unix, Linux, Windows, etc.), both free (Firewall Builder http://www.fwbuilder.org/), and commercial (Norton Personal Firewall – http://www.symantec.com/sabu/nis/npf/). But we shall speak only of those for Linux (iptables).

The first firewall computer was UNIX host with two connections to two different networks and didn't do traffic routing. A network card was connected to the Internet and the other to a private LAN. To connect to the Internet from the private network you had to log to the UNIX server on which the firewall was. Then the resources of the server were used to access the Internet. For example, it could be used X-Windows to execute the Netscape browser on the firewall system and the display was done on a workstation.

The browser, running on the firewall system, has access to both networks. This type of system with two network connections is very good if you TRUST ALL users. A Linux system can be easily set, creating on it user accounts for anyone who wishes Internet access.

With this configuration, the only computer from the private network that can communicate with the exterior is the firewall. None can download in the personal workstations. They must first download the files on the firewall system and then download the files from the firewall in the workstations.

Attention: 99% of attacks begin by gaining access to a user account in the attacked system. That's why this type of firewall is not recommended. It is also very limited.

1.2. Firewall Policies

There are two firewall policy principles:
Everything that is not explicitly permitted is forbidden. (DROP policy)
Everything that is not explicitly forbidden is allowed. (ACCEPT policy)
Firewall are used with two purposes:
1. To keep the people away (worms, hackers).
2. To let the people (employees, children).

To create a security policy the followings steps have to be considered:
1. Describe what services you have to offer.
2. Describe the group of people to whom you must offer the services.
3. Describe each service each group must access.
4. For each service of each group describe how the service should be secured.
5. Write a statement through which all other forms of access are forbidden.

1.3. Firewall Types

a) Filtering firewalls – which block the selected packets.
b) Proxy servers (sometimes called firewalls) – which make a connection for a user.

a) *Packet filtering firewalls*

Filtering the packets is the type of firewall incorporated in the Linux kernel. A filtering firewall works at the network level. The data will leave the system only if the firewall rules allow it. As the packets arrive, they are filtered by type, source address, destination address and port, this information being contained in each packet.

Many network routes can accomplish a firewall function. Filtering firewalls can be thought as a sort of router. That's why, to work with a router a detailed study of an IP packet is required.

Because very few data are analysed and recorded, these types of firewalls consume lesser CPU resources and create a smaller latency in the network.

Filtering firewalls don't offer password-based control. User cannot identify. The only identity a user has is the IP address assigned to it's own workstation. This can be a problem if DHCP (dynamically assigned IP addresses) is used because the rules are based on IP addresses and will have to be adjusted if the IP addresses change.

Filtering firewalls are more transparent to user. The user doesn't have to set rules in his applications in order to use the Internet. For most proxy servers this is not true.

b) *Proxy Servers*

Proxy servers are usually used to control or monitor the traffic that goes out. Some application-level proxies also do caching to the data requested by the user (stores the data). This caching minimizes the bandwidth request and diminishes the access to the same data for the next user. It also keeps an indisputable record of everything that was transferred. There are two types of proxy servers:
b_1) Application-level proxies – that do the job for the user.
b_2) SOCKS proxies – that crosses out (interchanges) the ports.

b_1) *Application-level Proxies*

The best example is a person that telnets to another computer and from there telnets outside. With a proxy application-level server this process is automated. When a user wants to telnet outside, the application sends him first to the proxy, then the proxy connects to the requested server (outside) and returns the data to the user.

Because proxy servers manipulate all communications, they can log everything the user does. The HTTP (web) proxies can record any visited URL and the FTP proxies can log any transferred file. They can even filter improper words from the visited sites or they can scan for viruses.

Application-level proxy servers can authenticate users. Before doing a connection outside, the server can request a user to login first.

b_2) *SOCKS Proxies*

A SOCKS server is like a switch. It purely commutes the connection through a system to another outside connection. Many SOCKS servers work only with TCP connections. Like filtering firewalls they cannot authenticate the user. But they can record where the user connected.

2. Iptables

Iptables (http://www.iptables.org) are used to configure, administer and inspect the packet filtering rules tables from the Linux kernel. Various tables can be defined. Each table contains a number of predefined chains but it can also contain user-defined chains.

Each chain is a list of rules that can match a set of packets. Each rule specifies what to do with a matching packet. This is called "a target", which can be a jump to a user-defined chain in the same table.

a) *Tables*

Today there are three independent tables (tables that are present at a certain moment depend on the kernel configuring options and on which modules are present).

The "-t" or "-table" specifies the table in which the command will operate. If the kernel is configured to automatically load the modules, a load of the module according to the table will be attempted, if it isn't already loaded. The tables are the following:

α) *Filter*; this is the default table. It contains the predefined INPUT chains (for packets coming in the Linux machine itself), FORWARD (for packets that are routed through the Linux machine) and OUTPUT (for the local generated packets, coming out of the Linux machine).

β) *Mat*; this table is consulted when a packet that creates a new connection is encountered. It consists of three predefined chains: PREROUTING (for altering packets as soon as they com in), OUTPUT (for altering the local generated packets before they come out) and POSTROUTING (for altering the packets before they come out).

γ) *Mangle*; this table is used for special alteration of packets. It has two predefined chains: PREROUTING (for altering the packets that come in before routing) and OUTPUT (for altering the packets before routing).

b) *Targets*

A firewall rule specifies the criteria for a packet and a target. If the packet does not match the rule, the next rule in the chain is examined. If it matches, then the rule is specified by the value of the target, that can be the name of a user-defined chain or one of the special values: ACCEPT, DROP, QUEUE or RETURN.

1° ACCEPT means the packet will pass;

2° DROP means the packet will be ignored;

3° QUEUE means the packet will be placed in the user space (if supported by the hernel);

4° RETURN means stopping the cross of the current chain and resuming from the next rule from the previous (parent) chain.

If the end of a predefined chain was reached or if there is a rule in a predefined chain that has RETURN as target, then the target specified by the chain policy will determine the fate of the packet.

c) *Extensions of targets*

Iptables can use target extension modules: the following extensions are included in the standard distribution.

A. LOG

Activates the kernel's option of recording the packets that match in the logs. When this option is set for a rule, the Linux kernel will display some info for all the packets that match (such as the fields in the IP header, etc.) in the kernel log (from where they can be read with *dmesg* or *syslogd*). It has the following options:

a) *log-level level*: the level of recordings in the log; it can be one of the seven levels;

b) *log-prefix prefix*: prefixes the messages from the log with the specified prefix, up to 29 letters length, being useful for distinguishing the messages in the logs;

c) *log-tcp-sequence*: writes in the log the TCP sequence numbers; this fact is a risk to security if users can read the log;

d) *log-tcp-options*: writes in the log the options from the TCP packets header;

e) *log-ip-options*: writes in the log the options from the IP packets header.

B. MARK

Thsi target is used to set the value of the netfilter marking associated to the packet. It is valid only in the *mangle* table: *set-mark mark*.

C. REJECT

It is used to send back a packet with an error message as reply to the packet that matches, otherwise it is equivalent to DROP. This target is valid only in the INPUT, FORWARD chains and in the user-defined chains that are called from these three predefined chains. Fen options control the nature of the error message sent back: *reject-with type*. The type can be: *icmp-net-unreachable*, *icmp-host-unreachable*, *icmp-port-unreachable*, *icmp-proto-unreachable*, *icmp-net-prohibited*, *icmp-host-prohibited*, *tcp-reset* which will return the corresponding ICMP error message (*port-unreachable* is the default). The *echo-reply* option is also permitted. It can be used only for the rules that specify a ping ICMP packet, and generates a ping reply. Finally, the *tcp-reset* option can be used for the rules that match the TCP protocol; this option sends back a TCO RST packet. It is used mainly to block the ident probings that appear frequently when an e-mail is sent to the hacked mail hosts (which otherwise wouldn't accept the e-mail).

D. TOS

This target is used to set the 8 bit Type of Service field from the IP header. It is valid only in the *mangle* table, *set-tos tos*, where *tos* can be:

a) Maximize-Delay 16 (0×10).

b) Maximize-Throughput 8 (0×08).

c) Maximize-Reliability 4 (0×04).

d) Minimize-Cost 2 (0×02).

e) Normal-Service 0 (0×00).

E. MIRROR

This is a target that switches the source and destination in the IP header and retransmits the packet.

It is valid only in the INPUT, FORWARD, PREROUTING chains and in the user-defined chains that are called from these three predefined chains. To be noticed that the packets that come out are not seen by any chain for filtering of packets, nor by the chacking of the NAT connection, to avoid the loops and other problems. Few options control the nature of the error message sent back.

F. SNAT

This target is valid only in the NAT table in the PREROUTING chain and it will modify the source address of the packet (and all other packets from the same connection will be modified), and future rules won't be examined anymore. It has an option: *to-source ip-addr-ipaddr port:port*, which can specify a single source IP address, a series of IP addresses and optionally, a series of ports (which is valid only if the rule also specifies the "-p tcp" or "-p udp" protocol).

If no series of ports is specified, than the source ports under 512 will be mapped towards other ports under 512; those between 512 and 1023 will be mapped towards ports under 1024, and the other ports will be mapped towards 1024 and upper. Where is possible, no intervention will be made to alter the port.

G. DNAT

This target is valid only in the NAT table in the PREROUTING and OUTPUT chains and in the user-defined chains that are called only from the two chains. The destination address of the packet will be modified (and all other packets from the same connection will be modified), and future rule won't be examined anymore. It has an option: *to-destination ip-addr-ipaddr port:port*, which can specify a single IP destination address, a series of IP addresses and optionally, a series of ports (which is valid only if the rule also specifies the "-p tcp" or "-p udp" protocol).

If no series of ports is specified, then the destination address won't be modified.

H. MASQUERADE

This target is valid only in the NAT table in the POSTROUTING chain. It is used only with the dynamically assigned IP addresses in dial-up connections: if there is a static IP address, the SNAT target should be used. Masquerading is equivalent to specifying a mapping of an IP address to the interface through which the packet comes out; it also has the effect that all connections are forgotten when the interface stops. This is the correct behavior when the next dial-up connection is probably not to have the same interface address (thus any connections already established are lost anyway). It has an option: *to-ports port-port*, which specifies a series of source ports that will be used bypassing the default heuristic selections of source ports of the SNAT target. This option is valid only if the rule also specifies the "-p tcp" or "-p udp" protocol.

I. REDIRECT

This target is only valid in the NAT table in the PREROUTING and OUTPUT chains and in the user defined chains that are called only from the two chains. It alters the destination IP address to send the packet to the machine itself (local generated packets are mapped at address 127.0.0.1). It has an option: *to-ports port-port*, that

specifies a destination port or a series of ports. Without this option, the destination port is never altered. The rule is valid only if it also specifies the "-p tcp" or "-p udp" protocol.

3. Description of the PHP Firewall Generator Application

The novelty that the *Firewall Generator* application brings, built in PHP language, is the possibility of dynamic manipulation of firewall rules through a web interface.

The application addresses to network administrators that know the iptables program pretty well, but also to those that wish to learn it. Before the beginning the use of the application the careful reading of the manual pages ("man iptables") and of the annex documentation is strongly recommended.

Such an application, running on the Internet, presents security problems, which will be analysed one at a time.

The following question rises: "How can a command like "iptables –A INPUT –p tcp –sport 1.2.3.4 –j REJECT" be executed? The answer sounds simple: "by using the system("command"); function from PHP. We observe on running that nothing happens. Why? Because PHP executes itself as user *nobody*, while *iptables* requires *root* user for execution. What do we do then? We shall write a program in C that will listen to requests at address 127.0.0.1, port 1500. If the program is alunched with *root* rights it is obvious that, he too will be able to launch programs that require *root* rights.

Such a program can be dangerous from a security point of view, because any command from the system can be executed. That's why several security measures are in order.

First, the program will have the *root* owner and the *root* group. Then the access rights will be *read, write, execute* for owner and nothing for the rest. Here is how it should look the result of a "ls –lsa tcpServer" command:

```
20 -rwxr-xr-x  1 root   root 17147 Jun 13 04:02 tcpServer
```

In the following we shall alude the program as tcpServer. TcpServer will listen only at address 127.0.0.1 so that no one else, but the Linux machine, can connect. The port 1500 has been chosen so that it does not interfere with other standard ports. Encrypting the traffic between the PHP page and *tcpServer* is not necessary because in a Linux machine, the traffic cannot be listened by users. In order for a user not to be able to launch commands as root using tcpServer, a simple authentication mechanism based on a password has been inserted: the client (PHP) sends a password to *tcpServer* and if the password is ok, *tcpServer* will execute the command, otherwise refusing the execution and writing in log "Unauthorized access".

For the users not to be able to read the contents of the PHP sources it is recommended to change the owner and the group of all PHP source files to apache.apache having chmod 600.

The Internet being an open environment, the traffic can be easily listened. That's why we encrypted all traffic, using the HTTPS protocol. If the accessing of the pages through the HTTP protocol is attempted, the mesage "*Page visualization ... is not allowed without a secure connection*".

At the accessing of the main page, the traffic will be automatically redirected to a secure connection through the HTTPS protocol.

The starting page shows a login window where the user name and password are requested.

When the password is written, nothing will be shown on the screen, as there is the danger of someone following the number of characters of the password, watching from behind the administrator. This masking was done by setting the font from the form to white colour, identical to that of the background.

If the login was successful, it goes on, otherwise displaying the error message "*Incorrect login*". After correct login, the screen is divided in three frames. In the topside is the application title: "*PHP Firewall Generator*", on the left is a menu and in the right the option selected from the menu is displayed.

Here's what the menu contains:

a) I n t e r f a c e: all system interfaces are shown (eth, ppp). Practically, it is the result of the "*ifconfig*" command. All network adapters, all ppp interfaces, with IP addresses, default routes, etc., are shown here.

b) P r o t o c o l s: a list of the protocols from the system is presented, found in the /etc/protocols file.

c) S e r v i c e s: a list of all services in the system is displayed (in the /etc/services file). The services are specified also by name and by number (for instance: ssh:22, http:80).

d) O p e n p o r t s: all ports listening to applications are shown. These informations are useful in finding the ports that need to be protected.

FILTER

There are three options for the filter table:

a) change the INPUT, FORWARD or OUTPUT chain policy to ACCEPT or DROP;

b) resets the counters of the INPUT, OUTPUT FORWARD chains or of the user defined chains;

c) creates a new user defined chain.

The INPUT chain is for the packets that enter the Linux machine; the FORWARD chain is for packets that convey through the Linux machine; the OUTPUT chain is for packets that exit the LINUX machine.

Visualize the FILTER, NAT or MANGLE Table

In the part second of these paper we will present, step by step, the chains and the possible user defined chains. In each table the name of chain is written, followed by the number of packets conveyed through the chain, as well as the traffic (in Mbytes) done in that chain. A blue table follows with three columns and multiple lines, on each line being a rule. In the first column there is the current number of the rule,

in the second column there is the rule itself and in the third column there is a check box and an icon for its editing. At the pressing of a rule's icon, the application will display a page for editing this rule.

Under the blue table there are three commands accessible through buttons: delete all rules from the chain, erase the chain (only if it is a user defined chain and the chain is empty), and rename the chain (only if it is user defined chain). At the end of the page there is an "Erase" button that deletes all checked rules.

4. Conclusions

Because Linux is by definition an operating system for computer networks, its network possibilities are countless, TCP/IP, TCP/IP v6, IPX/SPX, Apple Talk, WAN, X.25, Frame Relay, ISDN, PPP, SLIP, PLIP, Radio Amateur, ATM, file sharing, email, www, FTP, NEWS, DNS, DHCP, NIS, LDAP, Telnet, X, VNC, Router, Multicasting, Bridge, IP Masquerade, IP Accounting, IP Aliasing, Traffic Shaping, Firewall, Port Forwarding, Load Balancing, EQL, Proxy, Diald, Tunneling, IP Mobil, VPN, SNMP, RAID.

Linking of a computer to the Internet implies, generally, the use of the Linux operating system and of the TCP/IP protocol suite. These components have their own security problems. Access to the Internet also implies the use of a set of dozens of services, programs, with various security problems, either due to software errors or due to mis-incorporation of some appropriate security facilities. In general, in order for a user to take the appropriate security measures at network connection, he must understand the way in which the LINUX operating system works with the Internet.

Our Firewall Generator is a very useful tool designed for LINUX system administrators. The Web interface designed in PHP and JavaScript allows the easily composure of firewall rules, as well as their easily manipulation. With few mouse clicks any rule from the firewall can be written, edited or deleted.

Received, November 20, 2003 *„Ovidius" University, Constanţa, Romania*

R E F E R E N C E S

[1] B u r a g a S., C i o b a n u G., *Computer Networks Programming Workshop* (in Romanian). Ed. Polirom, Bucureşti, 2001.

[2] J u r c a I., *Programming Computer Networks* (in Romanian). Ed. de Vest, Timişoara, 2000.

[3] O g l e t r e e T.W., *Firewalls - Protection of Networks Connected to the Internet* (in Romanian, transl. from English). Ed. Teora, Bucureşti, 2001.

[4] P e t a c E., P e t a c D., *Principles and Techniques of Data Protection in Computer Network* (in Romanian). Ed. MatrixRom, Bucureşti, 1998.

[5] S c h n e i e r B., *Applied Cryptography* J. Wiley a. Sons, New York, 1994.

[6] T a n e n b a u m A.S., *Computer Networks* (in Romanian, transl. from English). Ed. Agora. Tg. Mureş, 1988.

[7] Z i e g l e r R.L., *Firewalls - Protecting Linux Systems* (in Romanian, transl. from English). Ed. Teora, Bucureşti, 2001.

[8] http://www.seifried.org/lasg Linux Administrators Security Guide.

[9] http://www.redhat.com RedHat Network.

[10] http://www.iptables.org IPTables Home Page.
[11] http://netfilter.samba.org/ NetFilter.
[12] http://linuxfocus.org/ Abaout Linux.
[13] http://www.linux.org/ About Linux.
[14] http://securityfocus.com About computer security.
[15] http://www.bugtraq.com/ About exploits.
[16] http://www.freshmeat.net/ Software for Linux.

GENERATOR DE FIREWALL (I)

(Rezumat)

Conectarea unui calculator la Internet presupune, in general, folosirea sistemului de operare LINUX şi a suitei de protocoale TCP/IP. Aceste componente au propriile lor probleme de securitate. Accesul la Internet presupune şi folosirea unui set de cateva zeci de servicii, programe, cu numeroase probleme de securitate, fie datorită unor erori in software, fie datorită neincorporării unor facilităţi de securitate potrivite. Iptables este folosit pentru a configura, administra şi inspecta tabelele de reguli de filtrare a pachetelor din kernel-ul Linux-ului. Fiecare din tabelele definite poate conţine un număr de lanţuri predefinite dar poate conţine şi lanţuri definite de utilizator. Noutatea cu care vine aplicaţia *Generator de firewall*, realizat in limbajul PHP, constă in posibilitatea manipulării dinamice a regulilor de firewall printr-o interfaţă web. Programul *tcpServer*, scris in limbajul C, „ascultă" cereri la adresa 127.0.0.1, portul 1500.

Data apariţiei: februarie 2005
Comanda nr.52 Tiraj: 175
Coli tipar: 8,25

UNIVERSITATEA TEHNICĂ "GH.ASACHI" IAŞI
TIPOGRAFIA

Str. Lascăr Catargi nr.38
Tel./Fax: 0232/21 59 49