documentação pyFirewall

JUAN CARLOS BINDEZ

Versão v1.1.2

Sexta, 26 de Agosto de 2022

Versão Atual pyFirewall v1.1.2

"This project is licensed under the MIT License."

"Este projeto está licenciado nos termos da licença MIT."

Detalhes de lançamento

v1.1.2:

reestruturação de código
correções de bugs

v1.1.1:

adiconado regras de bloqueios e liberação de iPs especificos
correção de bugs

v1.0.4:

alterações na interface do usuario

v1.0.3:

correção de bugs
reestruturação de código
adicionado recursos de melhorias de navegação pelo menu.

v1.0.2:

adicionado recurso para salvar as alterações do firewall
deletar regras

v1.0.1:

reestruturação de código
correção de falhas

v1.0.0:

versão inicial

Objetivo do software:

facilitar a configuração de firewall (iptables)
diminuir a quantidade de comandos digitados
facilitar a visualização das regras diminur o tempo de configuração de firewall

o que é o pyFirewall:

O pyFirewall é um software escrito em Python na versão 3.10.4, que visa manipular os comandos do iptables (https://g.co/kgs/9ZJDYt), este programa pode te ajudar a entender as regras de firewall e facilitar as configurações.

Como usar?

Faça um git clone:

```
git clone https://github.com/JuanBindez/pyFirewall-v1.1.2
```

Acesse a pasta:

```
cd pyFirewall-v1.1.2/
```

Agora é só rodar o software:

```
python3 pyfirewall.py
```

Índice dos namespaces

Lista de namespaces

Lista dos namespaces com uma breve descrição:

Índice dos componentes

Lista de componentes

Lista de classes, estruturas, uniões e interfaces com uma breve descrição:

Índice dos ficheiros

Lista de ficheiros

Lista de todos os ficheiros com uma breve descrição:

Documentação dos namespaces

Referência ao namespace banner

Funções

def **header_banner** ()

Descrição detalhada

```
Aqui esta o banner da interface do usuario.
```

Documentação das funções

def banner.header_banner ()

```
13 def header_banner():
14       print(Color.AMARELO +
15                     '''
16
17
18                                                          ,,
,,    ,,
19                                      `7MM"""YMM    db
`7MM   7MM
20                                        MM     `7
MM    MM
21           `7MMpdMAo. `7M'    `MF'   MM    d    `7MM   `7Mb,od8   .gP"Ya   `7M'
,A     `MF' ,6"Yb.    MM    MM
22             MM    `Wb    VA    ,V     MM""MM       MM     MM' "' ,M'    Yb    VA
,VAA    ,V   8)    MM    MM    MM
23             MM     M8     VA ,V       MM    Y       MM     MM       8M"""""""
VA ,V   VA ,V      ,pm9MM    MM    MM
24             MM    ,AP       VVV        MM            MM     MM       YM.     ,
VVV     VVV     8M    MM    MM    MM
25             MMbmmd'        ,V       .JMML.       .JMML..JMML.     `Mbmmd'
W        W       `Moo9^Yo.JMML.JMML. v1.1.2
26             MM             ,V
27           .JMML.       OOb"
28
29                                              Copyright (c) 2022 Juan Carlos
Bindez
30
31                                                *[Ctrl + C] Para Sair do
Programa
32                     '''
33       + Color.RESET)
34
```

Referência ao namespace colors

Componentes

class **Color**

Descrição detalhada

```
    classe para colorir a interface do usuario.
```

Referência ao namespace ipv4

Namespaces

logica_ipv4

Descrição detalhada

```
Copyright (c) 2022 Juan Carlos Bindez
"This project is licensed under the MIT License."
```

Referência ao namespace ipv4.logica_ipv4

Componentes

class **LogicasMenu1**
class **RegrasList**
class **DeleteRegra**
class **SaveTable**
class **IpRegras**

Descrição detalhada

```
    Classes usadas para armazenar comandos do iptables,
    e metodos que executam os comandos.
```

Referência ao namespace ipv6

Descrição detalhada

```
ipv6 ainda não implementado.
```

Referência ao namespace pyfirewall

Funções

def **Ver_regras_firewall** ()
INCIO DO BLOCO DE MENU IPV4 ####.

def **deletar_regras_firewall** ()
escolha 2 ####

def **regras_de_ports_firewall** ()
escolha 3 ####

def **salva_regras_firewall** ()
escolha 4 ####

def **netfilter_install** ()
escolha 5 ####

def **exclui_tab_firewall** ()
escolha 6 ####

def **ip_regras** ()
escolha 7 ####

def **menu_main_ipv4** ()
fim do menu de escolha 7 ####

Variáveis

ver_regras = **LogicasMenu1**("sudo iptables -L --line-numbers")
delete = **LogicasMenu1**("sudo iptables -D INPUT")

Descrição detalhada

```
Todas as funções desse script buscam comandos embutidos
nas classes do arquivo logica_ipv4.py e os executam,
conforme solicitado na interface do usuario.
```

Documentação das funções

def pyfirewall.deletar_regras_firewall ()

escolha 2 ####

```
41      def deletar_regras_firewall():
42           os.system("clear")
43           ver_regras.start_command()
44           header_banner()
45           # deleta id de regra no firewall
46           print(Color.AMARELO +
47
48                     '''
49                                                  Deletar de qual tabela?
50
51                                         *[0]Voltar
52                                         *[1]INPUT
53                                         *[2]FORWARD
54                                         *[3]OUTPUT
55
56                     '''
57           + Color.RESET)
58
59           choice_delete = str(input(">>"))
60
61           if choice_delete == "0":
62                os.system("clear")
63                menu_main_ipv4()
64
65           elif choice_delete == "1":
66                DeleteRegra.delete_INPUT.delete_id()
67                os.system("clear")
68                ver_regras.start_command()
69                menu_main_ipv4()
70
71           elif choice_delete == "2":
72                DeleteRegra.delete_FORWARD.delete_id()
73                os.system("clear")
74                ver_regras.start_command()
75                menu_main_ipv4()
76
77           elif choice_delete == "3":
78                DeleteRegra.delete_OUTPUT.delete_id()
79                os.system("clear")
80                ver_regras.start_command()
81                menu_main_ipv4()
82
83           else:
84                os.system("clear")
85                print("ops, digite apenas os numeros listados!")
86                menu_main_ipv4()
87
88           os.system("clear")
89           ver_regras.start_command()
90           delete.delete_id()
91           menu_main_ipv4()
92
93
```

def pyfirewall.exclui_tab_firewall ()

escolha 6 ####

```
235       def exclui_tab_firewall():
236            os.system("clear")
237            ver_regras.start_command()
238            header_banner()
239
240            print(Color.AMARELO +
241
242                      '''
243                                                      Escolha a Tabela a ser
Excluída
244
245                                             *[0]Voltar
246                                             *[1]INPUT
247                                             *[2]FORWARD
248                                             *[3]OUTPUT
249                                             *[4]Todas as tabelas
250
251                      '''
252            + Color.RESET)
253
254            escolha = str(input(">>"))
255
256            if escolha == "0":
257                 os.system("clear")
258                 menu_main_ipv4()
259
260            elif escolha == "1":
261                 os.system("sudo iptables -F INPUT")
262                 os.system("clear")
263                 ver_regras.start_command()
264                 menu_main_ipv4()
265
266            elif escolha == "2":
267                 os.system("sudo iptables -F FORWARD")
268                 os.system("clear")
269                 ver_regras.start_command()
270                 menu_main_ipv4()
271
272            elif escolha == "3":
273                 os.system("sudo iptables -F OUTPUT")
274                 os.system("clear")
275                 ver_regras.start_command()
276                 menu_main_ipv4()
277
278            elif escolha == "4":
279                 os.system("sudo iptables -F")
280                 os.system("clear")
281                 ver_regras.start_command()
282                 menu_main_ipv4()
283
284            else:
285                 os.system("clear")
286                 print("ops, digite apenas os numeros listados!")
287                 menu_main_ipv4()
288
289
290
```

def pyfirewall.ip_regras ()

escolha 7 ####

```
292       def ip_regras():
293
294            def header_escolha7():
295                 os.system("clear")
296                 header_banner()
297
298                 print(Color.AMARELO +
299                           '''
300                                                *[0]Voltar
301                                                *[1]ACCEPT
302                                                *[2]DROP
303
304
305                           '''
306                 + Color.RESET)
307
308
309            def ip_regra_INPUT_ACCEPT():
310                 IpRegras.ip_ACCEPT_tab_INPUT.ip_func_regra()
311                 os.system("clear")
312                 ver_regras.start_command()
313                 menu_main_ipv4()
314
315
316            def ip_regra_FORWARD_ACCEPT():
317                 IpRegras.ip_ACCEPT_tab_FORWARD.ip_func_regra()
318                 os.system("clear")
319                 ver_regras.start_command()
320                 menu_main_ipv4()
321
322
323
324            def ip_regra_OUTPUT_ACCEPT():
325                 IpRegras.ip_ACCEPT_tab_OUTPUT.ip_func_regra()
326                 os.system("clear")
327                 ver_regras.start_command()
328                 menu_main_ipv4()
329
330
331
332            def ip_regra_INPUT_DROP():
333                 IpRegras.ip_DROP_tab_INPUT.ip_func_regra()
334                 os.system("clear")
335                 ver_regras.start_command()
336                 menu_main_ipv4()
337
338
339            def ip_regra_FORWARD_DROP():
340                 IpRegras.ip_DROP_tab_FORWARD.ip_func_regra()
341                 os.system("clear")
342                 ver_regras.start_command()
343                 menu_main_ipv4()
344
345
346            def ip_regra_OUTPUT_DROP():
347                 IpRegras.ip_DROP_tab_OUTPUT.ip_func_regra()
348                 os.system("clear")
349                 ver_regras.start_command()
350                 menu_main_ipv4()
351
352
353            os.system("clear")
354            header_banner()
355            print(Color.AMARELO +
356                      '''
```

```
357
358                                                  Escolha a Tabela
359
360                                          *[0]Voltar
361                                          *[1]INPUT
362                                          *[2]FORWARD
363                                          *[3]OUTPUT
364                    '''
365          + Color.RESET)
366
367          escolha7 = str(input(">>"))
368
369          if escolha7 == "0":
370               os.system("clear")
371               menu_main_ipv4()
372
373          elif escolha7 == "1":
374               header_escolha7()
375               escolha = str(input(">>"))
376
377               if escolha == "0":
378                    os.system("clear")
379                    menu_main_ipv4()
380
381               elif escolha == "1":
382                    ip_regra_INPUT_ACCEPT()
383
384               elif escolha == "2":
385                    ip_regra_INPUT_DROP()
386
387
388          elif escolha7 == "2":
389               header_escolha7()
390               escolha = str(input(">>"))
391
392               if escolha == "0":
393                    os.system("clear")
394                    menu_main_ipv4()
395
396               elif escolha == "1":
397                    ip_regra_FORWARD_ACCEPT()
398
399               elif escolha == "2":
400                    ip_regra_FORWARD_DROP()
401
402          elif escolha7 == "3":
403               header_escolha7()
404               escolha = str(input(">>"))
405
406               if escolha == "0":
407                    os.system("clear")
408                    menu_main_ipv4()
409
410               elif escolha == "1":
411                    ip_regra_OUTPUT_ACCEPT()
412
413               elif escolha == "2":
414                    ip_regra_OUTPUT_DROP()
415
416          else:
417               os.system("clear")
418               print("Ops, Digite apenas os numeros listados!")
419               ip_regras()
```

def pyfirewall.menu_main_ipv4 ()

fim do menu de escolha 7 ####

escolha 8 ### fim do menu de escolha 8 ###

MENU INICIAL PRINCIPAL IPV4 ######

```
429       def menu_main_ipv4():
430            header_banner()
431            print(Color.AMARELO +
432                       '''
433                                                  *[1]Ver regras
434                                                  *[2]Delete regra
435                                                  *[3]Ports
436                                                  *[4]Salvar
437                                                  *[5]Instalar o netfilter-
persistent.service
438                                                  *[6]Excluir tabelas
439                                                  *[7]Ip (regras para IPs
especificos)
440
441                       '''
442            + Color.RESET)
443
444            choice = str(input(">>"))
445
446            if choice == "1":
447                 Ver_regras_firewall()
448
449            elif choice == "2":
450                 deletar_regras_firewall()
451
452            elif choice == "3":
453                 regras_de_ports_firewall()
454
455            elif choice == "4":
456                 salva_regras_firewall()
457
458            elif choice == "5":
459                 netfilter_install()
460
461            elif choice == "6":
462                 exclui_tab_firewall()
463
464            elif choice == "7":
465                 ip_regras()
466
467            else:
468                 os.system("clear")
469                 print("Digite Apenas os Números Listados!")
470                 menu_main_ipv4()
```

def pyfirewall.netfilter_install ()

escolha 5 ####

```
227       def netfilter_install():
228            os.system("sudo apt-get install netfilter-persistent.service")
229            os.system("sudo apt-get install iptables-persistent")
230            time.sleep(2)
231            os.system("clear")
232            menu_main_ipv4()
233
```

def pyfirewall.regras_de_ports_firewall ()

escolha 3 ####

```
95       def regras_de_ports_firewall():
96
97
98            def regra_port_INPUT():
99                 header_regra_port()
100                choice_regra = str(input(">>"))
101
102                if choice_regra == "0":
103                     os.system("clear")
104                     menu_main_ipv4()
105
106                elif choice_regra == "1":
107                     RegrasList.ports_tab_input_accept.port_change()
108                     os.system("clear")
109                     ver_regras.start_command()
110                     menu_main_ipv4()
111
112                elif choice_regra == "2":
113                     RegrasList.ports_tab_input_drop.port_change()
114                     os.system("clear")
115                     ver_regras.start_command()
116                     menu_main_ipv4()
117
118                else:
119                     os.system("clear")
120                     print("ops, digite apenas os numeros listados!")
121                     menu_main_ipv4()
122
123
124           def header_regra_port():
125                os.system("clear")
126                header_banner()
127                print(Color.AMARELO +
128                     '''
129                                          *[0]Voltar
130                                          *[1]ACCEPT
131                                          *[2]DROP
132
133                     '''
134                + Color.RESET)
135
136
137
```

```
138           def regra_port_FORWARD():
139                header_regra_port()
140                choice_regra = str(input(">>"))
141
142                if choice_regra == "1":
143                     RegrasList.ports_tab_forward_accept.port_change()
144                     os.system("clear")
145                     ver_regras.start_command()
146                     menu_main_ipv4()
147
148                elif choice_regra == "2":
149                     RegrasList.ports_tab_forward_drop.port_change()
150                     os.system("clear")
151                     ver_regras.start_command()
152                     menu_main_ipv4()
153
154                else:
155                     os.system("clear")
156                     print("ops, digite apenas os numeros listados!")
157                     menu_main_ipv4()
158
159
160           def regra_port_OUTPUT():
161                header_regra_port()
162                choice_regra = str(input(">>"))
163
164                if choice_regra == "1":
165                     RegrasList.ports_tab_output_accept.port_change()
166                     os.system("clear")
167                     ver_regras.start_command()
168                     menu_main_ipv4()
169
170                elif choice_regra == "2":
171                     RegrasList.ports_tab_output_drop.port_change()
172                     os.system("clear")
173                     ver_regras.start_command()
174                     menu_main_ipv4()
175
176                else:
177                     os.system("clear")
178                     print("ops, digite apenas os numeros listados!")
179                     menu_main_ipv4()
180
181
182
183           os.system("clear")
184           ver_regras.start_command()
185           header_banner()
186           print(Color.AMARELO +
187                     '''
188                                                     Escolha a Tabela
189
190                                          *[0]Voltar
191                                          *[1]INPUT
192                                          *[2]FORWARD
193                                          *[3]OUTPUT
194
195                     '''
196           + Color.RESET)
197
198           choice_tab = str(input(">>"))
199
200           if choice_tab == "0":
201                os.system("clear")
202                menu_main_ipv4()
203
204           elif choice_tab == "1":
205                regra_port_INPUT()
206
207           elif choice_tab == "2":
208                regra_port_FORWARD()
209
210           elif choice_tab == "3":
211                regra_port_OUTPUT()
212
```

def pyfirewall.salva_regras_firewall ()

escolha 4 ####

```
217       def salva_regras_firewall():
218            os.system("sudo service netfilter-persistent save")
219            time.sleep(2)
220            os.system("sudo systemctl restart netfilter-persistent.service")
221            time.sleep(2)
222            os.system("sudo systemctl status netfilter-persistent.service")
223            os.system("clear")
224            menu_main_ipv4()
225
```

def pyfirewall.Ver_regras_firewall ()

INCIO DO BLOCO DE MENU IPV4 ####.

escolha 1 ####

```
34       def Ver_regras_firewall():
35            # ve regras existentes no firewall
36            os.system("clear")
37            ver_regras.start_command()
38            menu_main_ipv4()
39
```

Documentação das variáveis

pyfirewall.delete = LogicasMenu1("sudo iptables -D INPUT")

pyfirewall.ver_regras = LogicasMenu1("sudo iptables -L --line-numbers")

Documentação da classe

Referência à classe colors.Color

Atributos Públicos Estáticos

string **VERDE** = '\033[92m'
string **VERDE_CLARO** = '\033[1;92m'
string **VERMELHO** = '\033[91m'
string **AMARELO** = '\033[93m'
string **AZUL** = '\033[1;34m'
string **MAGENTA** = '\033[1;35m'
string **NEGRITO** = '\033[;1m'
string **CYANO** = '\033[1;36m'
string **CYANO_CLARO** = '\033[1;96m'
string **CINZA_CLARO** = '\033[1;37m'
string **CINZA_ESCURO** = '\033[1;90m'
string **PRETO** = '\033[1;30m'
string **BRANCO** = '\033[1;97m'
string **INVERTE** = '\033[;7m'
string **RESET** = '\033[0m'

Documentação dos dados membro

string colors.Color.AMARELO = '\033[93m' `[static]`

string colors.Color.AZUL = '\033[1;34m' `[static]`

string colors.Color.BRANCO = '\033[1;97m' `[static]`

string colors.Color.CINZA_CLARO = '\033[1;37m' `[static]`

string colors.Color.CINZA_ESCURO = '\033[1;90m' `[static]`

string colors.Color.CYANO = '\033[1;36m' `[static]`

string colors.Color.CYANO_CLARO = '\033[1;96m' `[static]`

string colors.Color.INVERTE = '\033[;7m' `[static]`

string colors.Color.MAGENTA = '\033[1;35m' `[static]`

string colors.Color.NEGRITO = '\033[;1m' `[static]`

string colors.Color.PRETO = '\033[1;30m' `[static]`

string colors.Color.RESET = '\033[0m' `[static]`

string colors.Color.VERDE = '\033[92m' `[static]`

string colors.Color.VERDE_CLARO = '\033[1;92m' `[static]`

string colors.Color.VERMELHO = '\033[91m' `[static]`

A documentação para esta classe foi gerada a partir do seguinte ficheiro:

pyFirewall-v1.1.2/**colors.py**

Referência à classe ipv4.logica_ipv4.DeleteRegra

Atributos Públicos Estáticos

delete_INPUT = **LogicasMenu1**("sudo iptables -D INPUT {}")
delete_FORWARD = **LogicasMenu1**("sudo iptables -D FORWARD {}")
delete_OUTPUT = **LogicasMenu1**("sudo iptables -D OUTPUT {}")

Descrição detalhada

```
Aqui os comandos do iptables para deletar regras.
```

Documentação dos dados membro

ipv4.logica_ipv4.DeleteRegra.delete_FORWARD = LogicasMenu1("sudo iptables -D FORWARD {}")[static]

ipv4.logica_ipv4.DeleteRegra.delete_INPUT = LogicasMenu1("sudo iptables -D INPUT {}")[static]

ipv4.logica_ipv4.DeleteRegra.delete_OUTPUT = LogicasMenu1("sudo iptables -D OUTPUT {}")[static]

A documentação para esta classe foi gerada a partir do seguinte ficheiro:

pyFirewall-v1.1.2/ipv4/**logica_ipv4.py**

Referência à classe ipv4.logica_ipv4.IpRegras

Atributos Públicos Estáticos

ip_ACCEPT_tab_INPUT = **LogicasMenu1**("sudo iptables -A INPUT -s {} -j ACCEPT")
ip_DROP_tab_INPUT = **LogicasMenu1**("sudo iptables -A INPUT -s {} -j DROP")
ip_ACCEPT_tab_FORWARD = **LogicasMenu1**("sudo iptables -A FORWARD -s {} -j ACCEPT")
ip_DROP_tab_FORWARD = **LogicasMenu1**("sudo iptables -A FORWARD -s {} -j DROP")
ip_ACCEPT_tab_OUTPUT = **LogicasMenu1**("sudo iptables -A OUTPUT -s {} -j ACCEPT")
ip_DROP_tab_OUTPUT = **LogicasMenu1**("sudo iptables -A OUTPUT -s {} -j DROP")

Descrição detalhada

```
Aqui as regras especificas para ip
```

Documentação dos dados membro

ipv4.logica_ipv4.IpRegras.ip_ACCEPT_tab_FORWARD = LogicasMenu1("sudo iptables -A FORWARD -s {} -j ACCEPT")[static]

ipv4.logica_ipv4.IpRegras.ip_ACCEPT_tab_INPUT = LogicasMenu1("sudo iptables -A INPUT -s {} -j ACCEPT")[static]

ipv4.logica_ipv4.IpRegras.ip_ACCEPT_tab_OUTPUT = LogicasMenu1("sudo iptables -A OUTPUT -s {} -j ACCEPT")[static]

ipv4.logica_ipv4.IpRegras.ip_DROP_tab_FORWARD = LogicasMenu1("sudo iptables -A FORWARD -s {} -j DROP")[static]

ipv4.logica_ipv4.IpRegras.ip_DROP_tab_INPUT = LogicasMenu1("sudo iptables -A INPUT -s {} -j DROP")[static]

ipv4.logica_ipv4.IpRegras.ip_DROP_tab_OUTPUT = LogicasMenu1("sudo iptables -A OUTPUT -s {} -j DROP")[static]

A documentação para esta classe foi gerada a partir do seguinte ficheiro:

pyFirewall-v1.1.2/ipv4/**logica_ipv4.py**

Referência à classe ipv4.logica_ipv4.LogicasMenu1

Membros públicos

def **__init__** (self, **command**)
def **start_command** (self)
def **delete_id** (self)
def **port_change** (self)
def **ip_func_regra** (self)

Atributos Públicos

command

Descrição detalhada

```
Estes metodos executam comandos do os.system().
```

Documentação dos Construtores & Destrutor

def ipv4.logica_ipv4.LogicasMenu1.__init__ (*self*, *command*)

```
21      def __init__(self, command):
22           self.command = command
23
24
```

Documentação dos métodos

def ipv4.logica_ipv4.LogicasMenu1.delete_id (*self*)

```
31      def delete_id(self):
32           id = int(input(Color.VERMELHO + " digite numero da regra a ser
deletada \n>>" + Color.RESET))
33           os.system(self.command.format(id))
34
35
```

def ipv4.logica_ipv4.LogicasMenu1.ip_func_regra (*self*)

```
43      def ip_func_regra(self):
44           ip = str(input(Color.VERMELHO + "Digite o ip Escolhido \n IP >> " +
Color.RESET))
45           os.system(self.command.format(ip))
46
```

def ipv4.logica_ipv4.LogicasMenu1.port_change (*self*)

```
37      def port_change(self):
38           port = str(input(Color.VERMELHO + "Digite a Porta Escolhida \n PORT
>> " + Color.RESET))
39           os.system(self.command.format(port))
40
41
```

def ipv4.logica_ipv4.LogicasMenu1.start_command (*self*)

```
26      def start_command(self):
27           os.system(self.command)
28
29
```

Documentação dos dados membro

ipv4.logica_ipv4.LogicasMenu1.command

A documentação para esta classe foi gerada a partir do seguinte ficheiro:

pyFirewall-v1.1.2/ipv4/**logica_ipv4.py**

Referência à classe ipv4.logica_ipv4.RegrasList

Atributos Públicos Estáticos

ports_tab_input_accept = **LogicasMenu1**("sudo iptables -A INPUT -p tcp --dport {} -j ACCEPT")
ports_tab_forward_accept = **LogicasMenu1**("sudo iptables -A FORWARD -p tcp --dport {} -j ACCEPT")
ports_tab_output_accept = **LogicasMenu1**("sudo iptables -A OUTPUT -p tcp --dport {} -j ACCEPT")
ports_tab_input_drop = **LogicasMenu1**("sudo iptables -A INPUT -p tcp --dport {} -j DROP")
ports_tab_forward_drop = **LogicasMenu1**("sudo iptables -A FORWARD -p tcp --dport {} -j DROP")
ports_tab_output_drop = **LogicasMenu1**("sudo iptables -A OUTPUT -p tcp --dport {} -j DROP")

Descrição detalhada

```
Aqui esta os comandos de iptables para regra de firewall,
para desbloqueio de portas.
```

Documentação dos dados membro

ipv4.logica_ipv4.RegrasList.ports_tab_forward_accept = LogicasMenu1("sudo iptables -A FORWARD -p tcp --dport {} -j ACCEPT")[static]

ipv4.logica_ipv4.RegrasList.ports_tab_forward_drop = LogicasMenu1("sudo iptables -A FORWARD -p tcp --dport {} -j DROP")[static]

ipv4.logica_ipv4.RegrasList.ports_tab_input_accept = LogicasMenu1("sudo iptables -A INPUT -p tcp --dport {} -j ACCEPT")[static]

ipv4.logica_ipv4.RegrasList.ports_tab_input_drop = LogicasMenu1("sudo iptables -A INPUT -p tcp --dport {} -j DROP")[static]

ipv4.logica_ipv4.RegrasList.ports_tab_output_accept = LogicasMenu1("sudo iptables -A OUTPUT -p tcp --dport {} -j ACCEPT")[static]

ipv4.logica_ipv4.RegrasList.ports_tab_output_drop = LogicasMenu1("sudo iptables -A OUTPUT -p tcp --dport {} -j DROP")[static]

A documentação para esta classe foi gerada a partir do seguinte ficheiro:

pyFirewall-v1.1.2/ipv4/**logica_ipv4.py**

Referência à classe ipv4.logica_ipv4.SaveTable

Atributos Públicos Estáticos

status_service = **LogicasMenu1**("sudo systemctl status netfilter-persistent.service")

Descrição detalhada

```
Aqui os comandos iptables para ver o status do serviço iptables,
restartar o serviço e iniciar o serviço.
```

Documentação dos dados membro

ipv4.logica_ipv4.SaveTable.status_service = LogicasMenu1("sudo systemctl status netfilter-persistent.service")`[static]`

A documentação para esta classe foi gerada a partir do seguinte ficheiro:

pyFirewall-v1.1.2/ipv4/**logica_ipv4.py**

Documentação do ficheiro

Referência ao ficheiro pyFirewall-v1.1.2/banner.py

Namespaces

banner

Funções

def **banner.header_banner** ()

Referência ao ficheiro pyFirewall-v1.1.2/colors.py

Componentes

class **colors.Color**

Namespaces

colors

Referência ao ficheiro pyFirewall-v1.1.2/ipv4/__init__.py

Namespaces

ipv4

Referência ao ficheiro pyFirewall-v1.1.2/ipv6/__init__.py

Namespaces

ipv6

Referência ao ficheiro pyFirewall-v1.1.2/ipv4/logica_ipv4.py

Componentes

class **ipv4.logica_ipv4.LogicasMenu1**
class **ipv4.logica_ipv4.RegrasList**
class **ipv4.logica_ipv4.DeleteRegra**
class **ipv4.logica_ipv4.SaveTable**
class **ipv4.logica_ipv4.IpRegras**

Namespaces

ipv4.logica_ipv4

Referência ao ficheiro pyFirewall-v1.1.2/pyfirewall.py

Namespaces

pyfirewall

Funções

def **pyfirewall.Ver_regras_firewall** ()
INCIO DO BLOCO DE MENU IPV4 ####.

def **pyfirewall.deletar_regras_firewall** ()
escolha 2 ####

def **pyfirewall.regras_de_ports_firewall** ()
escolha 3 ####

def **pyfirewall.salva_regras_firewall** ()
escolha 4 ####

def **pyfirewall.netfilter_install** ()
escolha 5 ####

def **pyfirewall.exclui_tab_firewall** ()
escolha 6 ####

def **pyfirewall.ip_regras** ()
escolha 7 ####

def **pyfirewall.menu_main_ipv4** ()
fim do menu de escolha 7 ####

Variáveis

pyfirewall.ver_regras = LogicasMenu1("sudo iptables -L --line-numbers")
pyfirewall.delete = LogicasMenu1("sudo iptables -D INPUT")

Referência ao ficheiro pyFirewall-v1.1.2/README.md