**Computer Systems Technology**

British Columbia Institute of Technology

COMP 8006 - Assignment3- Testing

Albert Huang &

Aiyan Ma

Feb 20, 2018

# Table of Contents

[Table of Contents 2](#_Toc1963849649)

[1. Monitor Testing 3](#_Toc413256480)

[1.1 Test Outline 3](#_Toc1994814797)

[1.2 Test Case Descriptions 6](#_Toc2067130505)

[1.2.1 Test 1 6](#_Toc1579398804)

[1.2.2 Test 2 7](#_Toc476197695)

[1.2.3 Test 3 8](#_Toc964156295)

[1.2.4 Test 4 9](#_Toc613581265)

[1.2.5 Test 5 9](#_Toc962794077)

[1.2.6 Test 6 10](#_Toc416379932)

[1.2.7 Test 7 10](#_Toc1272846028)

[1.2.8 Test 8 11](#_Toc1995542986)

[1.2.9 Test 9 11](#_Toc1905873675)

[1.2.10 Test 10 11](#_Toc1057762710)

[1.2.11 Test 11 12](#_Toc897360974)

[1.2.12 Test 12 12](#_Toc841284663)

[1.2.13 Test 13 13](#_Toc1128490880)

[1.2.14 Test 14 14](#_Toc521240778)

[1.2.15 Test 15 15](#_Toc1170661056)

[1.2.16 Test 16 16](#_Toc1405203251)

[1.2.17 Test 17 17](#_Toc1907134773)

[1.2.18 Test 18 18](#_Toc1390879778)

[1.2.19 Test 19 19](#_Toc1924653775)

# Monitor Testing

## 1.1 Test Outline

| Rule # | Test Description | Tool Used | Expected Results | Pass/Fail |
| --- | --- | --- | --- | --- |
| 1 | Successfully use command line interface start monitor | Bash/vim | Monitor start from command line interface, and record a password attempt in blacklist | Pass. Detailed results are attached. |
| 2 | Successfully use GUI start the monitor program | Bash | Monitor start by using GUI | Pass. Detailed results are attached. |
| 3 | Successfully use regular expression tester to test user defined pattern | Bash | Try regular expression tester many times to confirm user defined pattern works | Pass. Detailed results are attached. |
| SSH Service | | | | |
| 4 | Successfully login SSH server | SSH client | The application won’t match anything  (command: #ssh server\_ip) | Pass. Detailed results are attached. |
| 5 | Fail to input the correct password and matched by default method (awk) | SSH client | The application would match the record from log file, and add or update the record to the blacklist  (command: #ssh server\_ip) | Pass. Detailed results are attached. |
| 6 | Fail to input the correct password and matched by regular expression | SSH client | The application would match the record from log file, and add or update the record to the blacklist  (command: #ssh server\_ip) | Pass. Detailed results are attached. |
| 7 | More than one visitor fail to input the correct password | SSH client | The application would match the record from log file, and add the new record into the blacklist  (command: #ssh server\_ip) | Pass. Detailed results are attached. |
| 8 | One visitor try many times and then get blocked through iptables | SSH client & iptables | The application would match the record from log file, and delete the record from the blacklist  The iptables block the visitor.  (command: #ssh server\_ip)  (command: #iptables -L) | Pass. Detailed results are attached. |
| 9 | More than one visitors try many times and then get blocked through iptables | SSH client & iptables | The application would match the record from log file, and delete the record from the blacklist  The iptables add a rules of blocking visitors.  (command: #ssh server\_ip)  (command: #iptables -L) | Pass. Detailed results are attached. |
| 10 | Crontab task activate the application | crontab | The application would activate through crontab  (\* \* \* \* \* /app path/monitor.sh timelimit $target\_ip $port) | Pass. Detailed results are attached. |
| 11 | After blocking period the visitor can access SSH service again | SSH client & iptables | After the blocking period, the blocking rule will be deleted, the visitor could access the ssh server again  (command: #iptables -L)  (command: #ssh server\_ip) | Pass. Detailed results are attached. |
| FTP Service | | | | |
| 12 | Successfully login FTP server | FTP client | The application won’t match anything  (command: #ftp server\_ip) | Pass. Detailed results are attached. |
| 13 | Fail to input the correct password and matched by regular expression | FTP client | The application would match the record from log file, and add or update the record to the blacklist  (command: #ftp server\_ip) | Pass. Detailed results are attached. |
| 14 | Fail to input the correct password the 2nd time and matched by regular expression | FTP client | The application would match the record from log file, and add or update the record to the blacklist  (command: #ftp server\_ip) | Pass. Detailed results are attached. |
| 15 | More than one visitor fail to input the correct password | FTP client | The application would match the record from log file, and add the new record into the blacklist  (command: #ftp server\_ip) | Pass. Detailed results are attached. |
| 16 | One visitor try many times and then get blocked through iptables | FTP client& iptables | The application would match the record from log file, and delete the record from the blacklist  The iptables add a rule of blocking visitor.  (command: #iptables -L) | Pass. Detailed results are attached. |
| 17 | More than one visitors try many times and then get blocked through iptables | FTP client & iptables | The application would match the record from log file, and delete the record from the blacklist  The iptables add a rules of blocking visitors.  (command: #ftp server\_ip)  (command: #iptables -L) | Pass. Detailed results are attached. |
| 18 | Crontab task activate the application | crontab | The application would activate through crontab  (grep CRON /var/log/syslog)  (\* \* \* \* \* /app path/monitor.sh timelimit $target\_ip $port) | Pass. Detailed results are attached. |
| 19 | After blocking period the visitor can access FTP service again | FTP client& iptables | After the blocking period, the blocking rule will be deleted, the visitor could access the ssh server again  (command: #iptables -L)  (command: #ftp server\_ip) | Pass. Detailed results are attached. |

## 1.2 Test Case Descriptions

### 1.2.1 Test 1

This was a simple test for how to use command line interface start monitor:

Set variables by hard code:

[screen shot]

Set general variable by command line:

[screen shot]

Capture one line from log:

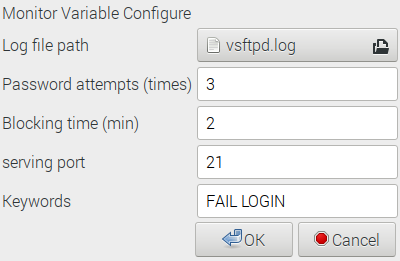
[screen shot]

### 

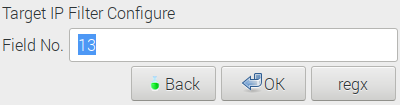
### 1.2.2 Test 2

This was a simple test for how to use GUI run the monitor program.

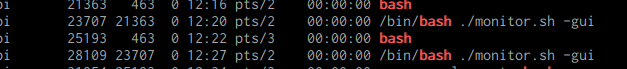
Run the script: ./monitor.sh -gui



Set general variables then click OK button :



After settling variables then check: ps -ef | grep bash

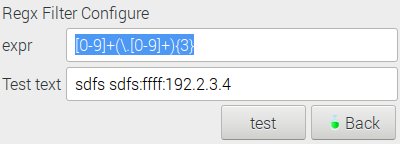


Test passes, we can see that the program is running at process no. 28109

### 1.2.3 Test 3

Use regular expression tester to test user defined pattern.

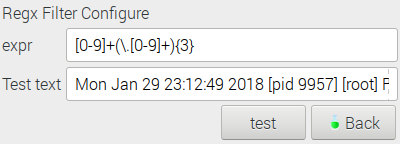
Run the script: ./monitor.sh -gui step 3 after click regx button



Test this pattern is works for the default testing text or not by click test button:



Test succeed! Retry another one by click retry button:



This time I try the line from vsftpd log:



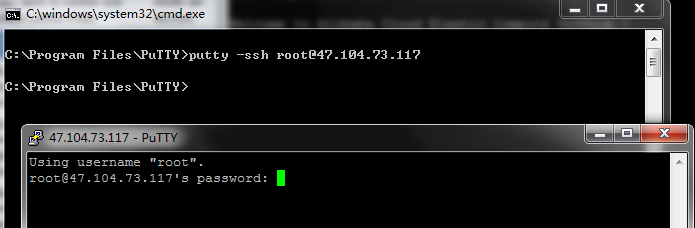
Test succeed!

Test passes.

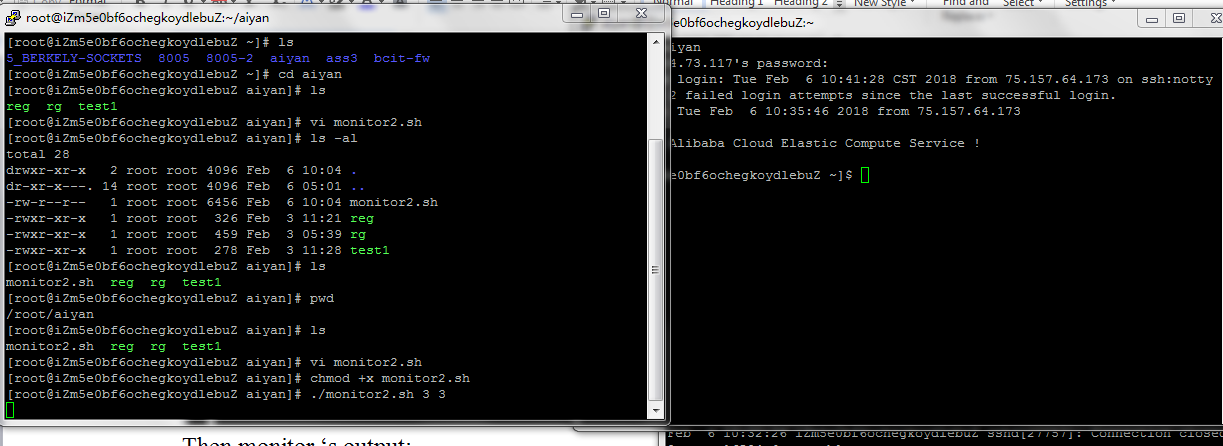
### 1.2.4 Test 4

This test case is just test if the monitor program would take any action about visitor successfully login SSH server

Run the command in client side: putty -ssh root@ssh\_server\_ip



Then monitor ‘s output:



[do twice by using different terminals or ip]

### 1.2.5 Test 5

This test case test if one visitor fail to input the correct password for the first time then how the monitor program react when using default ip filter (awk).

**Fail to get the IP**

The IP is not correctly retrieved, it should be 75.157.65.60

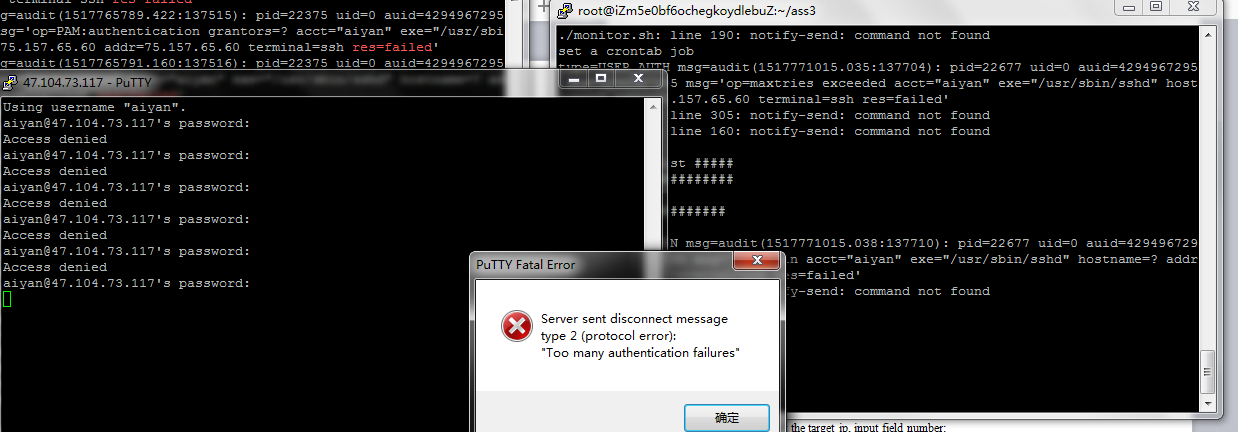
But the iptable rules is:

Chain INPUT (policy ACCEPT)

target prot opt source destination

DROP tcp -- 13.0.0.0 anywhere tcp dpt:ssh

type=USER\_AUTH msg=audit(1517773338.007:137835): pid=23675 uid=0 auid=4294967295 ses=4294967295 msg='op=PAM:authentication grantors=? acct="aiyan" exe="/usr/sbin/sshd" hostname=75.157.65.60 addr=75.157.65.60 terminal=ssh res=failed'

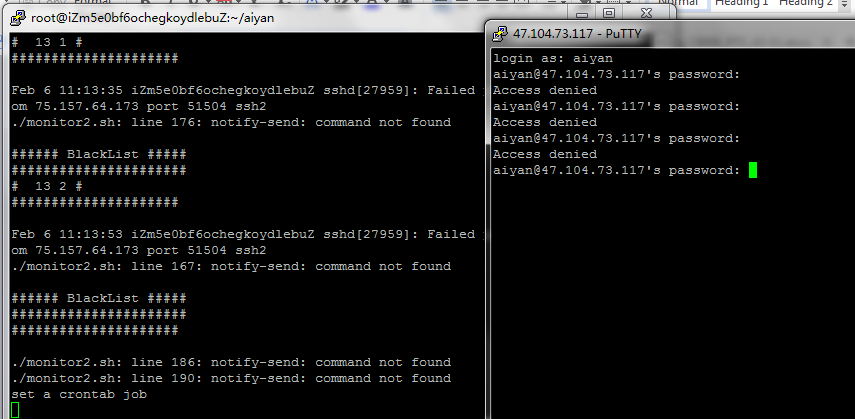


It proves that capture ip just by location is not that reliable. While it can works in fedora’s secure file due to the log format.

[test under fedora OS]

After changing the code of monitor.sh, re-test at Centos as

1. turn off the regex ( check and make sure regx=0 by default);
2. Sshing the host by 3 times wrongly passwd:



The record of log is:

Feb 6 11:13:26 iZm5e0bf6ochegkoydlebuZ sshd[27959]: pam\_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=ssh ruser= rhost=75.157.64.173 user=aiyan

Feb 6 11:13:29 iZm5e0bf6ochegkoydlebuZ sshd[27959]: Failed password for aiyan from 75.157.64.173 port 51504 ssh2

Feb 6 11:13:35 iZm5e0bf6ochegkoydlebuZ sshd[27959]: Failed password for aiyan from 75.157.64.173 port 51504 ssh2

Feb 6 11:13:53 iZm5e0bf6ochegkoydlebuZ sshd[27959]: Failed password for aiyan from 75.157.64.173 port 51504 ssh2

### 1.2.6 Test 6

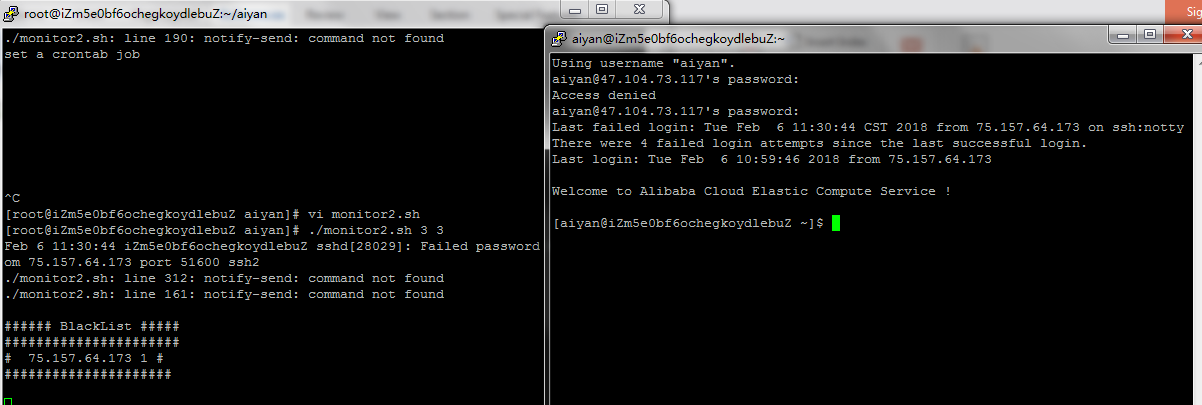
This test case test if one visitor fail to input the correct password then how the monitor program react and this time we are using regular expression to match keywords.

Set the regx=1 to turn on the regx

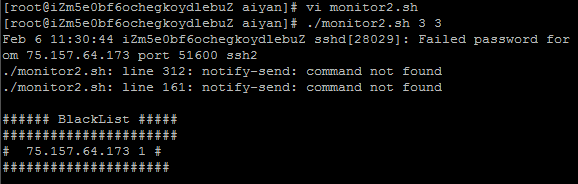
Ssh to the host by *putty aiyan@47.104.73.117*

One time wrong password, then input the correct password and can login

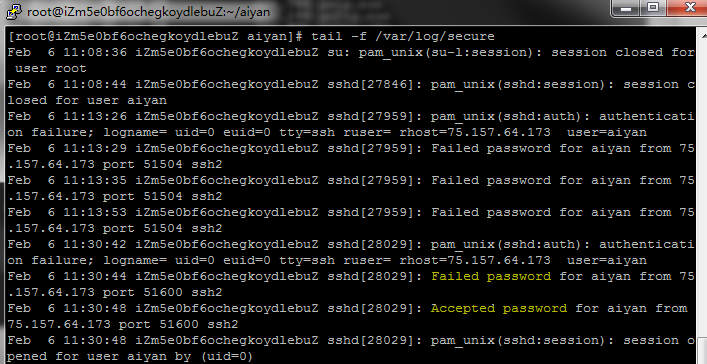
[screen shot but not match the blocking trigger]



[program output screen shot: blacklist]



[ssh log screen shot: tail the file]



### 1.2.7 Test 7

This test case test if more than one visitor fail to input the correct password what the monitor would respond for it.

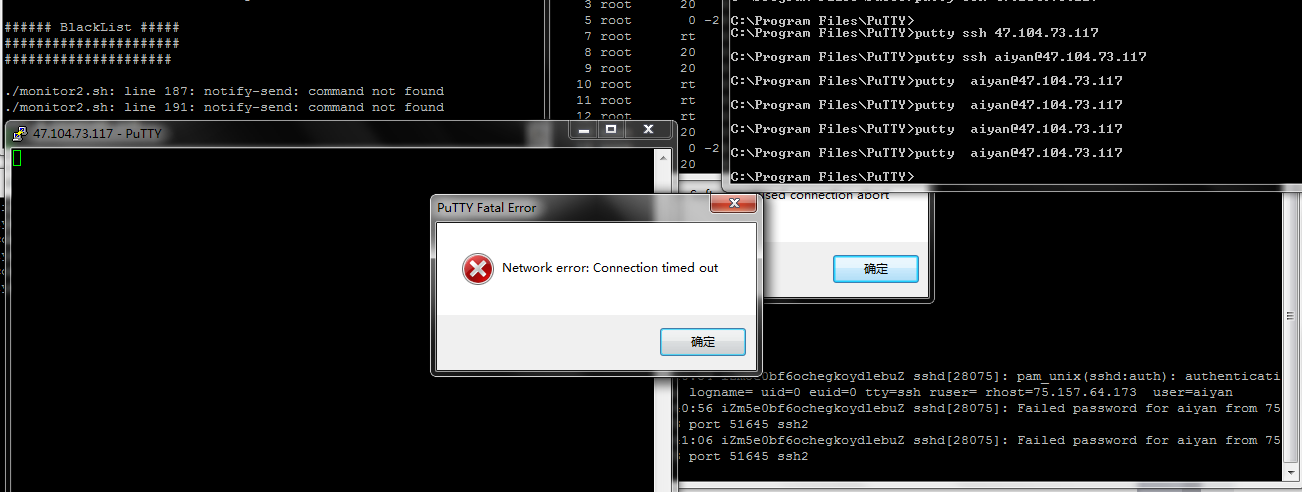
[screen shots as above mentioned]

### 1.2.8 Test 8

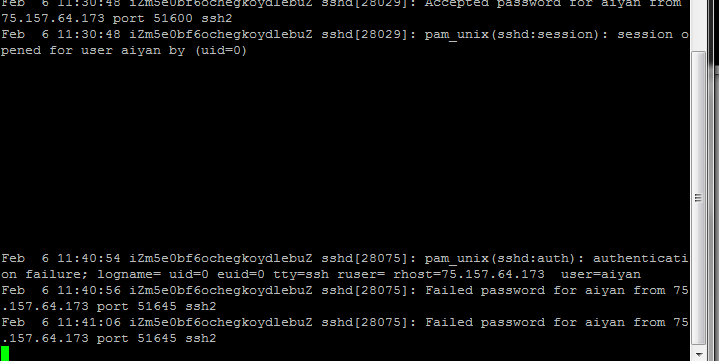
This test case test if one visitor try many times that over the threshold then what the monitor would respond for it.

After 3 times wrong password, the connection is reset and no more new connection is allowed.

Functional Pass:



[screen shot for ssh log : tail ]



[screen shot for program output : blacklist]

[screen shot for iptables ]

[screen shot for crontab]

### 1.2.9 Test 9

This test case test if more than one visitors try many times that over the threshold then what the monitor would respond for it.

[screen shots as above mentioned]

### 1.2.10 Test 10

This test case test how crontab task activate the application after monitor insert a rule into crontab.

When the task start time pass, the crontab log record:

Command: grep CRON /var/log/syslog

[screen shots for cron log: grep CRON /var/log/syslog]

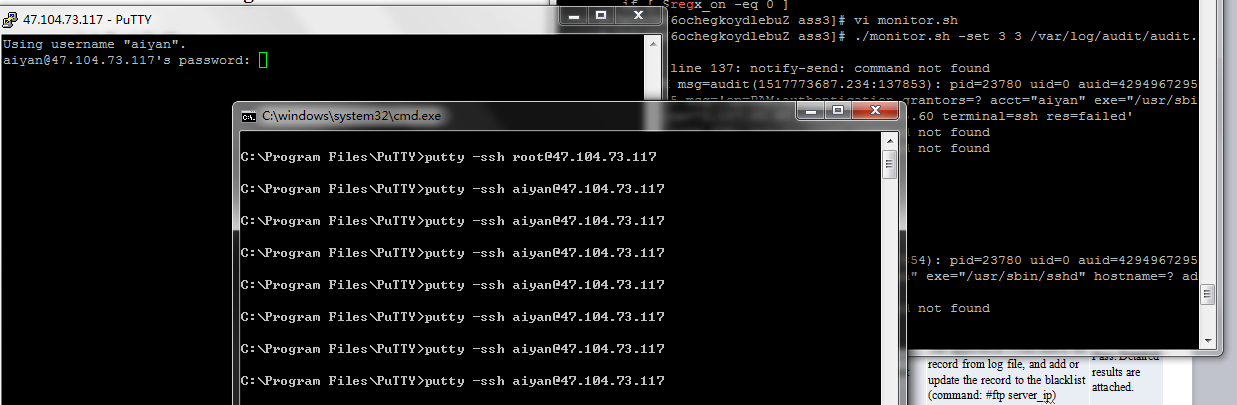
[screen shots for iptables: iptables -L]

[screen shots for crontab: crontab -l]

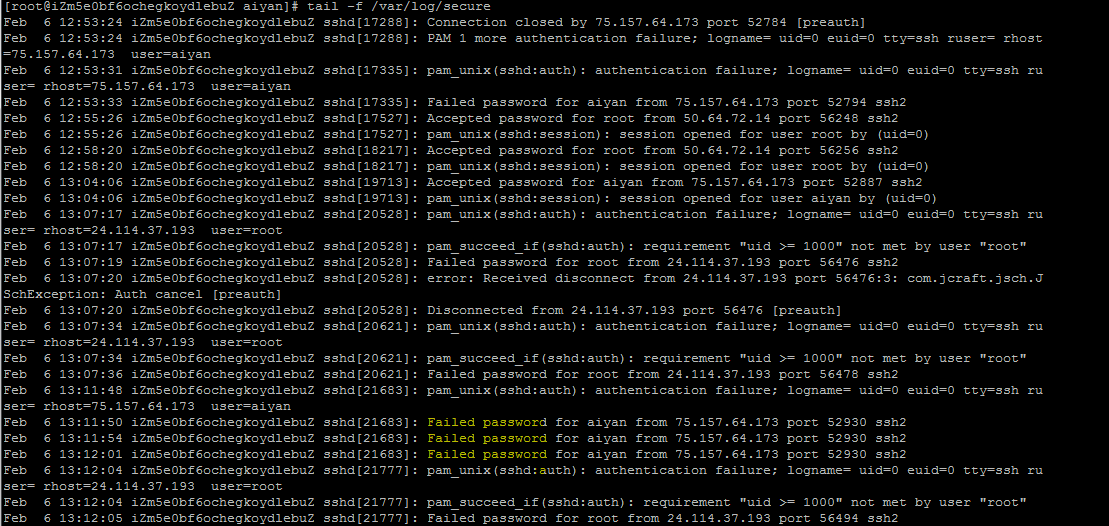
### 1.2.11 Test 11

This test case test if after blocking period the visitor can access SSH service again

After the blocking time, visitor try to log in the ssh again:



[screen shot for ssh log : tail ]

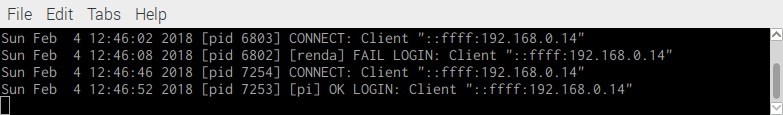


### 

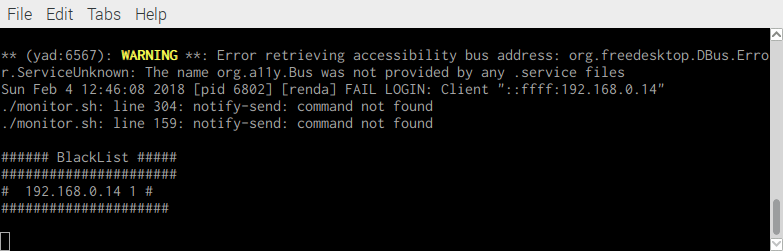
### 1.2.12 Test 12

This test case is just test if the monitor program would take any action about visitor successfully login FTP server

Run the command in client side: ftp $ftp\_server\_ip

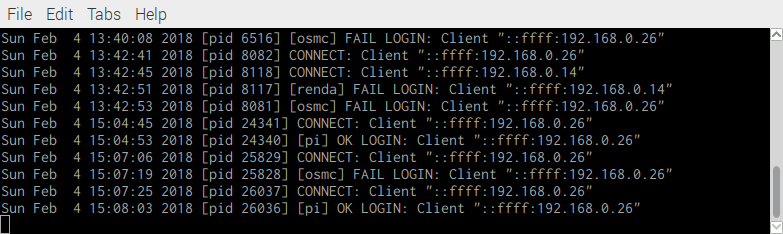


From the ftp log we can find that client 192.168.0.14 has successfully login at 12:46:46

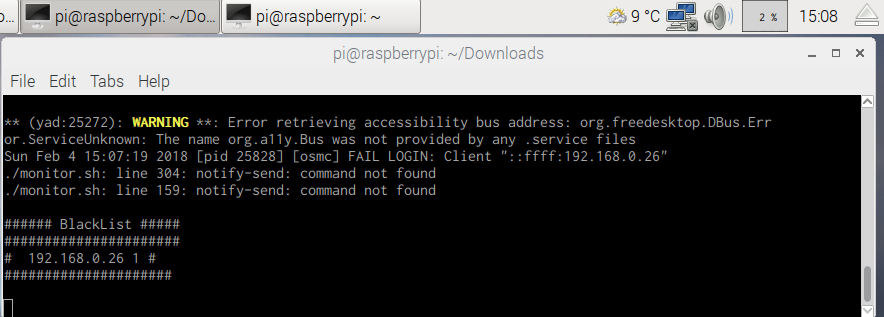


While from the program output message, there is nothing new after 12:46:08

Try again:



we can see that at 15:08 there is a time, client successfully logged in ftp server:

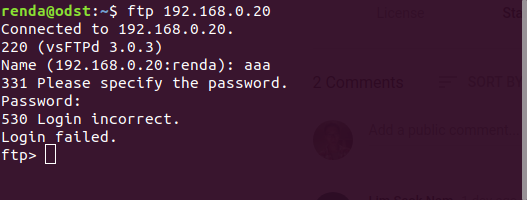


Test pass.

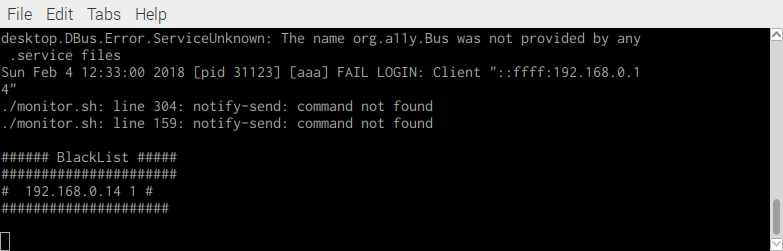
### 1.2.13 Test 13

This test case test if one visitor fail to input the correct password for the first time then how the monitor program react.

Run ftp client to log in the server and fail to input password:



The server side monitor would output the current blacklist:



The ftp log would be captured



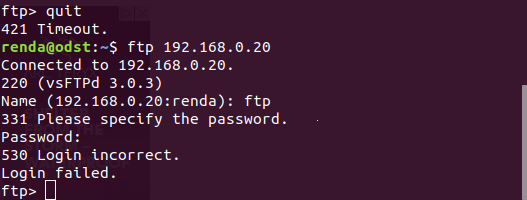
We can find the record by checking time 12:33:00

Test passed.

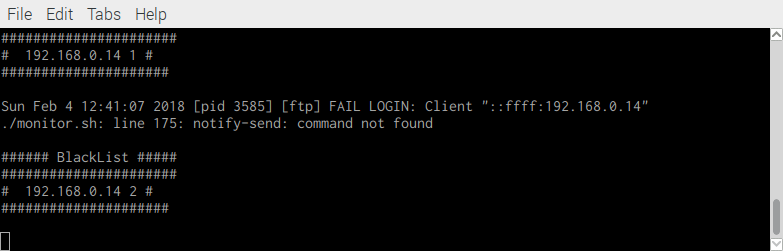
### 1.2.14 Test 14

This test case test if one visitor fail to input the correct password for the second time then how the monitor program react.

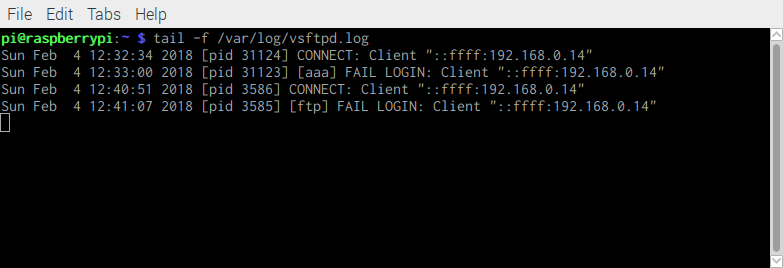
Run ftp client to log in the server and fail to input password again:



The server side monitor would renew the current blacklist:



The ftp log would be captured



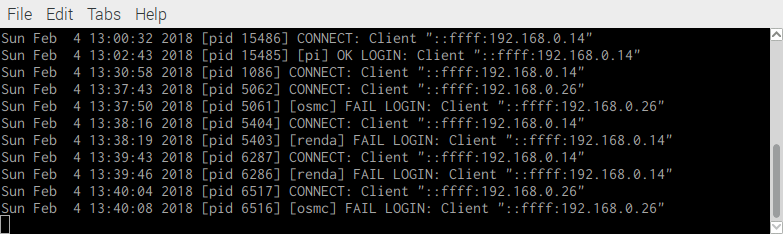
We can find the record by checking time 12:41:07

Test passed.

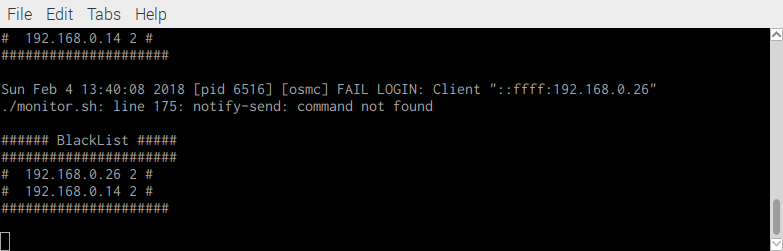
### 1.2.15 Test 15

This test case test if more than one visitor fail to input the correct password what the monitor would respond for it.

After experience multiple visitor input wrong password the ftp log shows:



And the monitor would output



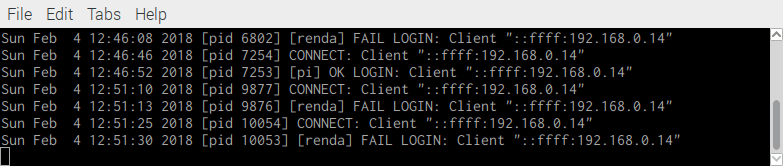
We can find that there are two ip in the current blacklist, and both of them try the password twice.

Test passed.

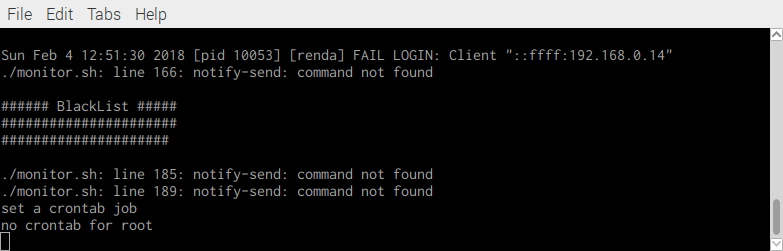
### 1.2.16 Test 16

This test case test if one visitor try many times that over the threshold then what the monitor would respond for it.

After trying for many times at least get the threshold attempt times bye one, the vsftp log:

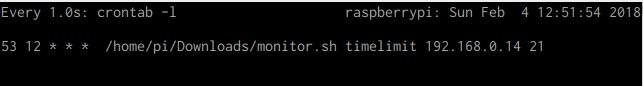


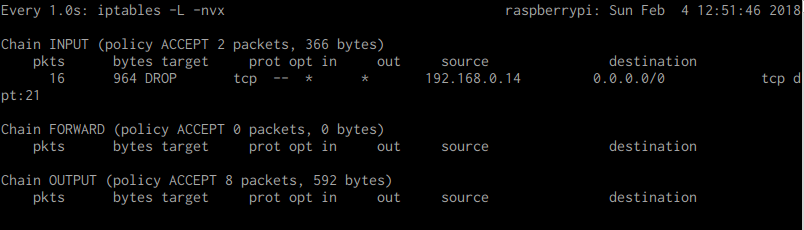
Monitor output



We can find that the blacklist is cleared;

at the same time check crontab and iptables rule:





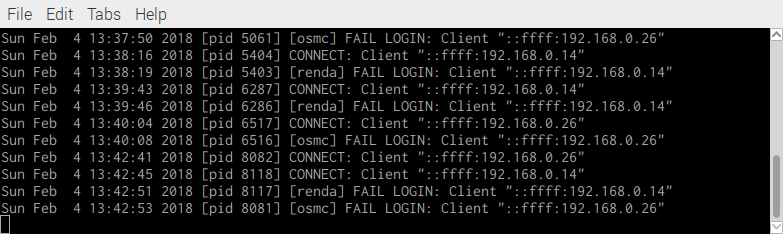
So we can see that the monitor add a firewall blocking rule and a crontab task in this process.

Test passed.

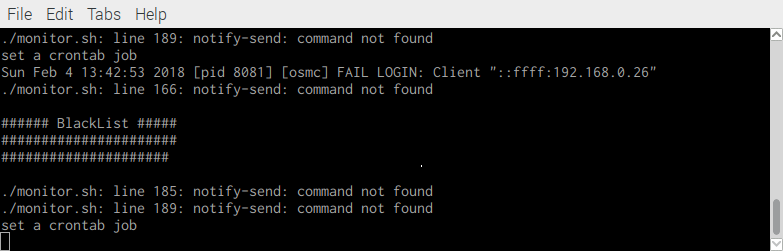
### 1.2.17 Test 17

This test case test if more than one visitors try many times that over the threshold then what the monitor would respond for it.

After trying for many times at least get the threshold attempt times by multiple visitors, the vsftp log:

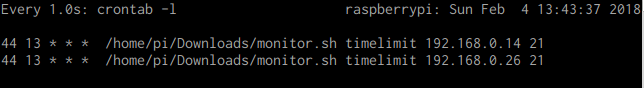


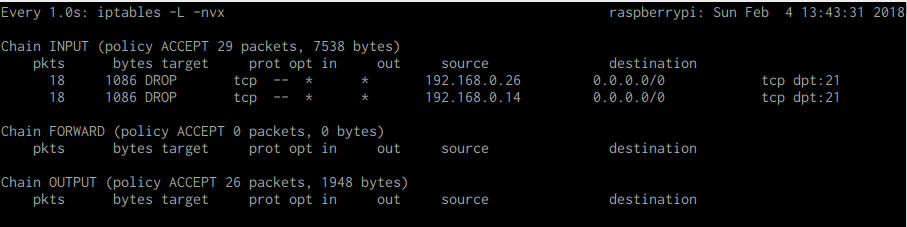
Monitor output



We can find that the blacklist is cleared;

at the same time check crontab and iptables rule:





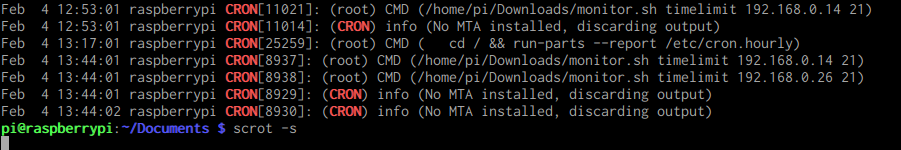
So we can see that the monitor add a firewall blocking rule and a crontab task in this process for both ip.

Test passed.

### 1.2.18 Test 18

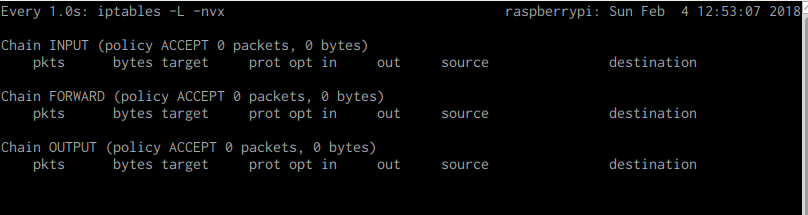
This test case test how crontab task activate the application after monitor insert a rule into crontab.

When the task start time pass, the crontab log record:

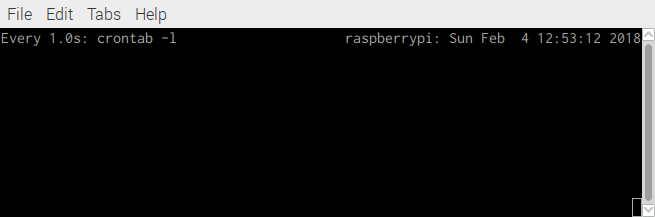


This cron task would clear the firewall rule:

After running the task, this task would be clear from the crontab:



And then clear itself from crontab:

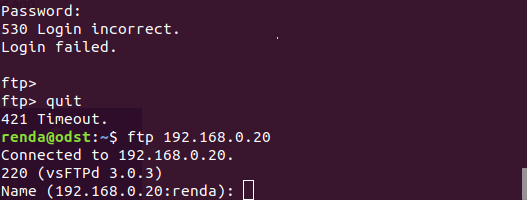


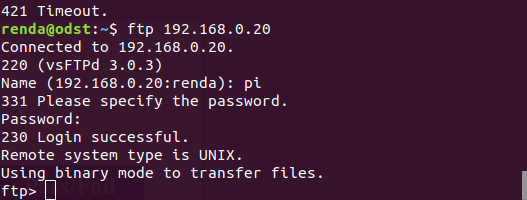
Test passed.

### 1.2.19 Test 19

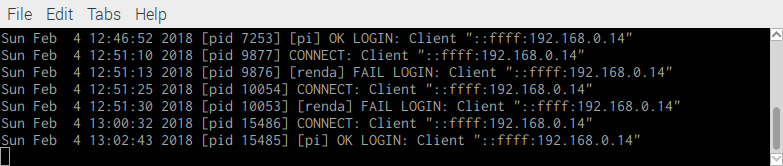
This test case test if after blocking period the visitor can access FTP service again

After the blocking time, visitor try to log in the ftp again:





And the vsftp log shows:



Test passed.