The PIX show processes Command

Contents

Introduction

This document explains the output of the PIX **show processes** command. The **show processes** command displays information about the active processes on the PIX.

Hardware and Software Versions

The information in this document is based on this software version.

- PIX Firewall Software Release 6.1(1)

The show processes Command

The **show processes** command displays all the active processes running on the PIX at the time the command is executed. This command is useful in determining which processes are receiving too much CPU time and which processes are not receiving any CPU time. In order to examine the CPU usage, issue the **show processes** command twice, wai about one minute after you first issue the command before you issue it a second time. Then, subtract the second Runtime value from the first Runtime value. The result allows you to know how much CPU time (in milliseconds) that process has received in that interval of time. It is important to note that some processes are scheduled to run at particular intervals, while some processes only run when they have information to process.

The 577poll process will most likely have the largest Runtime of all your processes. This is normal because the 577poll process polls the Ethernet interfaces to see if they have any data that requires action. Examples of common polling processes include the following:

```
577poll            Polls the Ethernet interfaces
i82543_timer       Polls the 66-MHz Gigabit Ethernet interfaces
i82542_timer       Polls the 33-MHz Gigabit Ethernet interfaces
```

Since these are polling processes, they can be used as a reference when comparing their Runtimes to other running processes.

The output of the **show processes** command should be used to compare one process against another. For example, if the Logger process has a very large runtime compared to the ip/0:0 process, then the PIX is spending more time generating and sending syslogs than passing IP traffic out of the outside interface. While this may not necessarily be a bad thing, if your PIX is running low on CPU resources then you may want to try and cut down on your logging to save resources.

The above is just an example, but the logic can be extended to other processes. For more information about troubleshooting performance issues with your PIX, see Monitoring PIX Performance.

Below is an example of the **show processes** command output. Note that many processes are created when needed. As such, the output below may differ considerably from the **show processes** output on your PIX. Click on the process name to find out more information about that process.

```
pixfirewall#show processes
Q Ty    PC       SP     STATE      Runtime  SBASE       Stack   Process
L si 8007752e 828a6720 803ece48          10 828a5760   3820/4096 arp_timer
L si 8007a92b 82949854 803ece48           0 82948898   3928/4096 FragDBGC
C we 80007480 808c083c 80298fb0          20 808bf888   3800/4096 CryptIC PDR poll
L we 8000baf0 82956034 803f0538       17410 82955170   3620/4096 dbgtrace
L we 8015885a 829581ac 803ca570        9143 82956200   7708/8192 Logger
H we 8015bb7f 82a3f258 803ca820           0 82a3d298   8092/8192 tcp_fast
H we 8015bb06 82a412e4 803ca820           0 82a3f328   8088/8192 tcp_slow
L si 800c52ea 82ab7498 803ece48           0 82ab64d8   4008/4096 xlate clean
L si 800c5210 82ab8524 803ece48           0 82ab7568   3672/4096 uxlate clean
M we 800c2b35 82c00810 803ece48           0 82bfe858   8004/8192
tcp_intercept_timer_process
L si 80195b56 82ca58b0 803ece48           0 82ca48f0   3996/4096 route_process
L si 800b59e1 82ca693c 803ece48           0 82ca5980   2876/4096 Hosts conn cleaner
H we 8008e35d 82cd66b8 803ece48           0 82cd2708 16168/16384 isakmp time keeper
L si 800b948d 82cec358 803ece48         560 82ceb398   3408/4096 perfmon
M we 80103d3f 82cee3e0 803ece48          40 82cec428   7780/8192 Crypto CA
H we 8008bb26 82d107c8 80335218           0 82d0f828   3984/4096 IPsec response
handler
M we 80087b71 82d12870 803ece48          20 82d108b8   7996/8192 IPsec timer handler
L we 800c3394 82d16374 803f99d0           0 82d154b0   3764/4096 pix/trace
L we 800c3594 82d17404 803f9b70           0 82d16540   3764/4096 pix/tconsole
H we 80078e6b 82d19544 82857aec         100 82d175d0   6688/8192 pix/intf1
H we 80078e6b 82d1b608 82857aa8      189660 82d19660   6620/8192 pix/intf0
H we 80078e6b 82d1d6a8 82857a64         330 82d1b700   7200/8192 pix/intf2
H we 80078e6b 82d1f748 82857a20         510 82d1d7a0   5644/8192 pix/intf3
H *  800100c9 7ffffe64 803ece38       23180 82d21840 13156/16384 ci/console
H we 801916bf 82d27050 8050bbe0           0 82d260e0   3840/4096 lu_ctl
C si 800be494 82d2810c 803ece48          10 82d27170   3564/4096 update_cpu_usage
H we 800b48a7 82db6f00 80336488           0 82db4ff8   7796/8192 uauth0
H we 800b48a7 82db8fa0 80336498           0 82db7098   7796/8192 uauth1
H we 800b48a7 82dbb040 803364a8           0 82db9138   7796/8192 uauth2
H we 800b48a7 82dbd0e0 803364b8           0 82dbb1d8   7796/8192 uauth3
H we 8015a90b 82dbf1c0 8066fabc           0 82dbd268   8008/8192 uauth
H we 801691fd 82dc02b4 803caae0           0 82dbf2f8   4012/4096 udp_timer
C rd 8006f33b 82dc1c10 803ed290  304079320 82dc0c40   3856/4096 i82543_timer
H si 800719cc 82dc2c8c 803ece48           0 82dc1cd0   4004/4096 557mcfix
C rd 8007198c 82dc3d34 803ed290  668648830 82dc2d60   3872/4096 557poll
L si 80071a22 82dc4dac 803ece48           0 82dc3df0   3876/4096 557timer
H we 80078e93 82dc5e18 82954a48           0 82dc4e80   3976/4096 fover_ip1
C we 800716a3 82dc6e60 806db754       48050 82dc5f10   2984/4096 ip/1:1
H we 80078e93 82dc7f44 82954a20           0 82dc6fa0   3672/4096 icmp1
M we 80168fc2 82dc8fc4 806ad174           0 82dc8030   3972/4096 riprx/1
M si 8012a882 82dca074 803ece48           0 82dc90c0   3980/4096 riptx/1
H we 80078e93 82dcb0e4 829549f8           0 82dca150   3972/4096 udp_thread
H we 80078e93 82dcc15c 829549d0           0 82dcb1e0   3948/4096 tcp_thread
```

```
  H we 80066078 82dcd214 803f3600          0 82dcc270   3440/4096 fover_thread
  H we 80078e93 82dce2a8 829549a8          0 82dcd310   3976/4096 fover_ip0
  C we 800716a3 82dcf310 8172630c      51630 82dce3a0   3012/4096 ip/0:0
  H we 80078e93 82dd03d4 82954980          0 82dcf430   3988/4096 icmp0
  M we 80168fc2 82dd1464 806ad134          0 82dd04d0   3972/4096 riprx/0
  M si 8012a882 82dd2524 803ece48          0 82dd1570   3980/4096 riptx/0
  H we 80078e93 82dd3594 82954958         60 82dd2600   3896/4096 udp_thread
  H we 80078e93 82dd460c 82954930          0 82dd3690   3948/4096 tcp_thread
  H we 80078e93 82dd56c8 82954908          0 82dd4730   3976/4096 fover_ip2
  C we 800730d3 82dd672c 827705bc        170 82dd57c0   3076/4096 ip/2:2
  H we 80078e93 82dd77f4 829548e0          0 82dd6850   3988/4096 icmp2
  M we 80168fc2 82dd8884 806ad0f4          0 82dd78f0   3972/4096 riprx/2
  M si 8012a882 82dd9944 803ece48          0 82dd8990   3980/4096 riptx/2
  H we 80078e93 82dda9b4 829548b8          0 82dd9a20   3972/4096 udp_thread
  H we 80078e93 82ddba2c 82954890          0 82ddaab0   3948/4096 tcp_thread
  M we 80013069 80c50760 803ece48          0 80c4ffa0   1544/2048 DHCP Client
  M we 80168fc2 80c527d8 806ad0f4        260 80c50848   6492/8192 dhcpc_recv/0
  M we 8016d4ae 8094e518 80726044          0 8094c588   7296/8192 dhcpd_recv/1
  M we 80019ded 80c96ed0 803ece48          0 80c94f10   8012/8192 DHCPD Timer
  H we 80168fc2 80c932f0 806ad074       3100 80c928b0   1500/4096 snmp
  H we 80078e93 82ddcae8 82954868          0 82ddbb50   3976/4096 fover_ip3
  C we 800730d3 82dddb4c 827e3a64         20 82ddcbe0   3680/4096 ip/3:3
  H we 80078e93 82ddec14 82954840          0 82dddc70   3988/4096 icmp3
  M we 80168fc2 82ddfca4 806ad0b4          0 82dded10   3972/4096 riprx/3
  M si 8012a882 82de0d64 803ece48          0 82ddfdb0   3980/4096 riptx/3
  H we 80078e93 82de1dd4 82954818          0 82de0e40   3972/4096 udp_thread
  H we 80078e93 82de2e4c 829547f0          0 82de1ed0   3948/4096 tcp_thread
  H we 8019121e 82de454c 803ea320          0 82de3598   3984/4096 lu_tx
  H we 801912ba 82de55d4 803ea328          0 82de4628   3976/4096 lu_rx
  H we 800100c9 82de6608 8023e318          0 82de56b8   3896/4096 fover_rx
  H we 80068465 82de7710 803f389c          0 82de6748   4024/4096 fover_tx
  H we 8006626a 82de87a0 803f38a8          0 82de77d8   4024/4096 fover_rep
  C we 8006895b 82de9814 803f38b0          0 82de8868   3988/4096 fover_parse
  H we 80124ea7 82e276a0 803fc650          0 82e25718   7244/8192 qos_metric_daemon
  H we 80078e93 80de6a54 808bf110          0 80de4ad8   8044/8192 ahd
  H we 8006e76f 80e55f30 806b8d30     174420 80e54fb0   3184/4096 espd
  H we 8015c02a 80e57fbc 807ec84c      71970 80e56040   3788/8192 isakmp_receiver
  H we 801672af 80e59b98 8032f1c4         10 80e59410   1420/2048 ppp_timer_thread
  H we 801754e7 80e6ac5c 80330c60          0 80e69ca8   2996/4096 pptp_mgmt
  H we 80143fd6 80e6ca1c 806bac28        150 80e6ad50   4812/8192 pptp_control/0
  H we 8006e76f 80e6ed90 806b8d08       1230 80e6cdf0   6340/8192 pptp_gre/0
  H we 8015ab0a 82e27b10 8065bde4          0 82e277e8    616/1024 listen/http1
  H we 8015ab0a 80c93eb8 8065bfbc          0 80c93be0    516/1024 listen/pfm
  H we 8015ab0a 80c94348 8065c0a8          0 80c94070    516/1024 listen/telnet_1
  H we 8015ab0a 82e33578 8065c71c          0 82e332a0    516/1024 listen/ssh_0
  M we 8015a90b 80e8f36c 8066fadc          0 80e8d428   7988/8192 tacplus_get
  M we 80128254 80e91464 803fc6e0          0 80e8f4b8   8092/8192 tacplus_snd
  M we 80168fc2 80e934c8 806ad0b4          0 80e91548   7772/8192 radius_rcvauth
  M we 80168fc2 80e94558 806ad074          0 80e935d8   3676/4096 radius_rcvacct
  M we 801261ec 80e9561c 803c57b0          0 80e94668   4004/4096 radius_snd
  M we 80168fc2 80e97fa8 806ad034          0 80e96028   6580/8192 radius_rcvauth
```

```
  M we 80168fc2 80e99038 806acff4          0 80e980b8   3676/4096 radius_rcvacct
  M we 801261ec 80e9a0fc 803c5798       1180 80e99148   3368/4096 radius_snd
  H we 8015ec93 82e59788 806eec8c          0 82e58848   3888/4096 websns_rcv_tcp
  H we 8016d4ae 82e5a860 8072c304          0 82e598d8   3684/4096 websns_rcv_udp
  M rd 80193644 82e5b924 80469ff0       1340 82e5a968   3028/4096 websns_snd
  L si 80194abe 82e5c9b8 80469fc8          0 82e5b9f8   4008/4096 websns_clean_cache
  M si 801943e2 82e5cf58 80469fc8          0 82e5ca88    432/4096 websns_keepalive
  M we 80156349 80cc4068 803ece48          0 80cc20b0   7976/8192 ssh/timer
  M *  8015c404 7ffffe60 803ece60        310 80cca2d8   4520/8192 ssh
  M we 8015b32f 80cc815c 8065c630        790 80cc6288   5788/8192 ssh_init
  M we 800af731 80cb1e14 80336230        270 80cb0020   5996/8192 http1
  H we 80168f87 80ccb234 806acff4          0 80cca2d8   3616/4096 tftp
```

The following table lists and describes the columns in the **show processes** command output:

Column	Description
Q	Process queue priority. Possible values: C (critical), H (high), M (medium), and L (low).
Ty	Scheduler test. Possible values: * (currently running), E (waiting for an event), S (ready to run, voluntarily relinquished processor), rd (ready to run, wake up conditions have occurred), we (waiting for an event), sa (sleeping until an absolute time), si (sleeping for a time interval), sp (sleeping for a time interval (alternate call), st (sleeping until a timer expires), hg (hung; the process will never execute again), and xx (dead: the process has terminated, but has not yet been deleted.).
PC	Current program counter.
SP	Current stack pointer.
State	Address of a thread queue.
Runtime (ms)	CPU time the thread has used, in milliseconds.
SBASE	Stack Base Address
Stack	Currently used and total stack space available, shown in bytes.
Process	Name of the thread's function. See the Processes section below for more information.

The Processes

The table below explains the individual processes in the **show processes** command output.

Note: This is not a complete list.

Process	Explanation
`arp_timer`	Address Resolution Protocol (ARP) timer to clear out the ARP cache.
`FragDBGC`	Thread for cleaning up the fragment database.
`CryptIC PDR poll`	Thread that handles sending and receiving requests to VPN accelerator card.
`dbgtrace`	Thread that prints out debug information to the console/Telnet session.
`Logger`	Syslog thread for syslog, console, buffer, PIX Device Manager (PDM), Simple Network Management Protocol (SNMP), and monitor logs.
`tcp_fast`	TCP stack fast timer process for protocol related functions (data path).
`tcp_slow`	TCP stack slow timer process for protocol related functions (session management).
`xlate clean`	Thread to clean up PIX translations.
`uxlate clean`	Used for keeping track of used and maximum connections.
`tcp_intercept_timer_process`	Thread to handle retransmission used in TCP intercept.
`route_process`	Thread that keeps track of routing table additions, deletions, and updates.
`Hosts conn cleaner`	Thread that removes connections marked for deletion. Also known as the garbage collector.
`isakmp_time_keeper`	Thread to keep track of all the ISAKMP timers. It acts on any one that goes off.

isakmp_receiver	Thread to listen for ISAKMP connections.
Crypto CA	Certificate authority (CA) and Public Key Infrastructure (PKI) client thread.
Crypto PKI RECV	Thread to handle incoming PKI messages from CA.
IPsec response handler	Thread to handle incoming IP Security (IPSec) packets.
IPsec timer handler	IPSec key timer.
perfmon	Keeps track of performance statistics.
qos_metric_daemon	Keeps historical track of different PIX metrics. Data used to display graphs in PDM.
pix/trace	Thread for Telnet traces/debugs.
pix/tconsole	Thread for console traces/debugs.
pix/intf(x)	Thread to process traffic from the interface (per interface).
ci/console	Console session thread for user input/output.
lu_ctl	Logical update control thread. It controls the stateful failover update thread.
lu_tx	Thread to send stateful failover messages.
lu_rx	Thread to receive stateful failover messages.
update_cpu_usage	Thread to keep track of the PIX's CPU statistics.
uauth(x)	User authentication thread daemon (per interface).
uauth	Thread that prompts users for authentication and communicates with authentication process.
udp_timer	Keeps track of UDP connections and marks UDP connections that exceed the timeout for deletion.

i82543_timer	66MHz gigabit interface timer used to check interface statistics for SNMP traps. Since this is a polling process, it is normal for the runtime value of this process to be very large. The above output indicates normal operation.
i82542_timer	33MHz gigabit interface timer used to check interface statistics for SNMP traps. Since this is a polling process, it is normal for the runtime value of this process to be very large. The above output indicates normal operation.
557mcfix	Thread to watch interface statistics for errors.
557poll	Thread which polls the Ethernet interfaces to see if they have received traffic that can be removed. Since this is a polling process, it is normal for the runtime value of this process to be very large. The above output normal operation.
557timer	Ethernet interface timer used to check interface statistics for SNMP traps.
poll_process	Thread created if there are no Ethernet interfaces in the PIX. Used to poll for CPU usage.
fover_ip(x)	Failover thread for receiving IP messages (per interface).
fover_thread	Main thread to keep track of PIX failover.
fover_rx	Receive failover messages across serial failover cable.
fover_tx	Transmit failover messages across serial failover cable.
fover_rep	Failover configuration replication thread. Used to verify the configuration replication from one PIX to another.
fover_parse	Thread to parse the failover messages.

ip/(x:x)	IP packet process (per interface).
icmp(x)	Internet Control Message Protocol (ICMP) packet process (per interface).
riprx/(x)	Routing Information Protocol (RIP) receive process (per interface).
riptx/(x)	RIP transmit process (per interface).
udp_thread/(x)	UDP thread for handling UDP packets (per interface).
tcp_thread/(x)	TCP thread for removing packets from the interface (per interface).
DHCP Client	Dynamic Host Configuration Protocol (DHCP) client thread.
dhcpc_config	DHCP client thread to configure address on PIX interface.
dhcpc_recv/(x)	DHCP client thread to accept a DHCP address on an interface (per interface).
dhcpd_recv/(x)	DHCP server thread to issue DHCP addresses (per interface).
DHCPD Timer	DHCP timer to keep track of when client addresses expire.
DHCP background	DHCP client background process. Handles queue events such as clients renewing their lease.
snmp	SNMP thread to send traps and receive polls from Network Management Station (per interface).
ahd	Authentication header daemon (used for IPSec authentication header) for connections to the PIX.
espd	Encapsulating security payload daemon (used for IPSec encapsulating security payload) for connections to the PIX.
ssh_init	Thread to initialize a Secure Shell (SSH) session

ssh/timer	Thread to process SSH timeouts.
ssh	SSH command line entry. One process per SSH session.
tacplus_get	TACACS+ thread to receive response from TACACS+ server.
tacplus_snd	TACACS+ thread to send authentication, authorization, and accounting (AAA) requests to TACACS+ server.
radius_rcvauth	RADIUS thread to receive authentication responses from RADIUS server.
radius_rcvacct	RADIUS Accounting thread for receiving accounting response from RADIUS server.
radius_snd	RADIUS thread to send AAA requests to RADIUS server.
websns_rcv_tcp	Reads response from Websense server.
websns_rcv_udp	Reads response from Websense server.
websns_snd	Sends requests to Websense server.
websns_clean_cache	Keeps track of cached Websense information.
websns_keepalive	Verify that the Websense server is still operational.
tftp	Thread to handle Trivial File Transfers (TFTP).
HTTP PROXY Server	Thread for the Hypertext Transfer Protocol (HTTP) auth-proxy server daemon process.
HTTP Server	HTTP server daemon process.
HTTP Timer	Thread to wait for an HTTP event.
http1	Thread for PDM connections.

L2TP data daemon	Layer 2 Tunneling Protocol (L2TP) daemon thread for data and flow control updates.
L2TP mgmt daemon	L2TP master daemon thread for all management related timers and queues.
l2tp_recv/(x)	L2TP receiver thread (per interface).
ppp_timer_thread	Thread to wait for PPP requests.
pptp_control/(x)	Point-to-Point Tunneling Protocol (PPTP) control daemon thread (per interface) to receive PPTP connections.
pptp_gre/(x)	PPTP generic routing encapsulation (GRE) receiver thread (per interface).
pptp_mgmt	PPTP master daemon for all management related timers and queues.
PPTP create idb	Thread to create the PPTP session.
listen/http(x)	Thread to listen for PDM connections (per interface). This thread is created per interface once a HTTP command is applied to a given interface.
listen/pfm	Thread to listen to connections to the PIX using PIX Secure Telnet (such as PIX Firewall Manager or Cisco Secure Policy Manager) (per interface). This thread is created per interface once a Telnet command is applied to a given interface.
listen/telent_(x)	Thread to listen for Telnet connections to the PIX (per interface). This thread is created per interface once a Telnet command is applied to a given interface.
listen/ssh_(x)	Thread to listen for SSH connections to the PIX (per interface). This thread is created per interface once an SSH command is applied to a given interface.

telnet/ci	Telnet command interface. One per Telnet session (max 5).

Related Information

- **PIX Support Page**
- **Monitoring PIX Performance**
- **Documentation for PIX Firewall**
- **PIX Command Reference**
- **Field Notices for PIX Firewall**
- **Requests for Comments (RFCs)**