Understanding the difference between pid_max, ulimit -u and thread_max

I am trying to understand the Linux processes. I have confusion on the terms `pid_max` , `ulimit -u` and `thread_max` . What exactly is the difference between these terms? Can someone clarify me the differences?

/ process / linux-kernel / ulimit / pid / thread

edited Jun 13 '14 at 1:52

asked Jun 12 '14 at 22:41
Ramesh
16.6k 17 78 128

1 Answer

Let us understand the difference between a process and a thread. As per this link,

> The typical difference is that threads (of the same process) run in a shared memory space, while processes run in separate memory spaces.

Now, we have the `pid_max` parameter which can be determined as below.

```
cat /proc/sys/kernel/pid_max
```

So the above command returns **32,768** which means I can execute **32,768** processes simultaneously in my system that can run in separate memory spaces.

Now, we have the `threads-max` parameter which can be determined as below.

```
cat /proc/sys/kernel/threads-max
```

The above command returns me the output as **126406** which means I can have **126406** threads in a shared memory space.

Now, let us take the 3rd parameter `ulimit -u` which says the total processes a user can have at a particular time. The above command returns me the output as **63203**. This means for all the processes that a user has created at a point of time the user can have **63203** processes running.

Hypothetical case

So assuming there are 2 processes simultaneously being run by 2 users and each process is consuming memory heavily, both the processes will effectively use the **63203** user limit on the processes. So, if that is the case, the 2 users will have effectively used up the entire **126406** `threads-max` size.

Now, I need to determine how many processes an user can run at any point of time. This can be determined from the file, `/etc/security/limits.conf` . So, there are basically 2 settings in this file as explained over here.

A *soft limit* is like a **warning** and *hard limit* is a **real max limit**. For example, following will prevent anyone in the student group from having more than 50 processes, and a warning will be given at 30 processes.

```
@student        hard    nproc           50
@student        soft    nproc           30
```

Hard limits are maintained by the kernel while the soft limits are enforced by the shell.

edited Jun 29 at 15:31
Aaron
187 1 1 8

answered Jun 12 '14 at 22:41
Ramesh
16.6k 17 78 128

I guess you have mentioned wrongly that `ulimit -u` denotes maximum `threads` a user can have, but actually it means the maximum `processes` a user can have. – Kannan Mohan Nov 5 '14 at 7:40

@KannanMohan, I think you are right. When I checked the man page, it showed up as process only. I will try and change the answer in sometime. Thanks for letting me know. I appreciate it. – Ramesh Nov 6 '14 at 0:39

I've edited the answer to fix the references of `ulimit -u` being associated with threads to being associated with processes. – Aaron Jun 29 at 15:32

There is a small connection between "max user processes" and threads. It limits the amount of "native" threads that user is allowed to have. For example use 3 java processes to open 500 threads each assuming that "max user processes" is 1000. Then try to run java -version and see the JVM crush miserably – Roman M Sep 25 at 15:32