Subject: TCP checkpoint/restart (Re: MCR)
Posted by Cedric Le Goater on Thu, 14 Dec 2006 12:57:29 GMT
View Forum Message <> Reply to Message

Thanks Daniel for moving that thread on the containers@ list.

When you have some time, could you just recap the main topics
of this discussion on tcp stack checkpoint/restart. I'm pretty
sure the openvz team as plenty to say.

Thanks,

C.

Daniel Lezcano wrote:
> Masahiko Takahashi wrote:
>> Thank you Daniel. I'm now getting much understanding about
>> what MCR does. Then, the next step you are going to do is
>> to implement sk_filter and check its performance ?
>
> Yes after network isolation is finished.
>
>>
>> On Wed, 2006-12-13 at 11:34 +0100, Daniel Lezcano wrote:
>>> There are 2 aspects:
>>> - TCP buffer
>>> - TCP re-transmission
>>>
>>> When you checkpoint, the outgoing packets are dropped when they are sent
>>> but the buffer is still there until the packets are acked by the peer
>>> (and that never happens because the traffic is dropped). These buffers
>>> are checkpointed. When you restart, you restore these buffers, you
>>> release the traffic and the TCP layer continue to send the packets. It
>>> is like you unplugged the network cable, wait a little and plugged it,
>>> the TCP traffic is resumed. We rely on the TCP mechanisms.
>> :
>>> In the TCP communication process, the receiver send the remaining opened
>>> windows he has. The sender rely on that to send the nb bytes
>>> corresponding. If you send a zero window while the sender is expecting
>>> to have at least "last windows - nb bytes sent", depending on the TCP
>>> implementation, nothing can happen (traffic will block), sender can
>>> decide to drop the connection because it thinks it is inconsistent or
>>> you can have other behaviors like retransmissions ...
>>
>> I understand MCR relies on TCP's robustness and zero-window
>> advertisement has a delicate issue, but I, as a network
>> amateur, still couldn't believe zero-window's inconsistency
>> is a serious problem...

>
> Both client and server have a 4096 window size. The server application
> is blocked (control+Z, data processing, stucked, ...) so it never reads
> the incoming traffic.
> 1. Client send 1024 bytes to server
> 2. Server ack these data and the window is 3072 (4096-1024)
> 3. Client send 1024 bytes again to server
> 4. Client ack these data but with a window advertisement of zero,
> normally this one should be between 2048 and 4096, no less than 2048.
>
>
> Client   Server
> | |
> | 1:1025(1024), win 4096 |
> |------------------------------>|
> | |
> | ack 1025, win 3072 |
> |<------------------------------|
> | |
> | 1025:2049(1024), win 4096 |
> |------------------------------>|
> | |
> | ack 2049, win 0 |
> |<------------------------------|
>
>
> Perhaps, it could work, perhaps not. To use the zero window we should
> check that all TCP stacks are compatible with that. If we have a server
> running on linux and we checkpoint it, it will not be cool if all
> windows client lose their connecions while all linux client or mac
> clients stay blocked.
>
>>
>>> When an application creates a socket, write to it and close it, the
>>> connection stay alive until all data are sent and wait for a moment in a
>>> specific tcp state (LAST_ACK, CLOSE_WAIT, TIME_WAIT, ...). After a time
>>> the socket is destroyed, freezing the sockets timer will avoid the
>>> socket to be destroyed. Do netstat -tano command...
>>
>> It seems we should check there is no packet in receive_queue
>> before closing the socket. If there is, the kernel tries to
>> send a RESET packet.
>
> The receive and the send queue are checkpointed. If the receive queue is
> not flushed before destroying the socket, there is no issue because the
> RST packet will be dropped because of the blocked traffic.
>
> Regards.

>
> -- Daniel
>
>
>
>

_____

Containers mailing list
Containers@lists.osdl.org
https://lists.osdl.org/mailman/listinfo/containers