Subject: Kernel OOPS with kernel 2.6.18 and openvz 0.28.18.1
Posted by MrDigi on Fri, 27 Apr 2007 22:28:23 GMT
View Forum Message <> Reply to Message

Hello,

I am using openvz on debian etch on an AMD64.

Kernel: debian 2.6.18 kernel (2.6.18.dfsg.1-12) with openvz patch (debian package kernel-patch-openvz 028.18.1)

Network interfaces: eth0 with ipv4 and ipv6 adresses, eth1 with ipv4 address
I've deleted the arp entry for the openvz machines on eth1 (ip neigh del ...)

Firewall: I am using iptables and ip6tables to control ipv4 and ipv6 traffic.

After some days of operation I wanted to stop a running virtual machine:

vzctl stop 101

During the execution of that command, the following OOPS happened:

Apr 27 23:33:50 janus kernel: Unable to handle kernel NULL pointer dereference a
t 0000000000000040 RIP:
Apr 27 23:33:50 janus kernel: [<ffffffff880a1bdc>] :ip6_tables:ip6t_unregister_
table+0xe/0x16f
Apr 27 23:33:50 janus kernel: PGD 113419067 PUD 113422067 PMD 0
Apr 27 23:33:50 janus kernel: Oops: 0000 [1] SMP
Apr 27 23:33:50 janus kernel: CPU: 0
Apr 27 23:33:50 janus kernel: Modules linked in: ip6table_mangle ip6table_filter
ip6_tables ip_nat_irc ip_nat_ftp ip_conntrack_irc ip_conntrack_ftp ipt_MASQUERA
DE ip_nat_pptp ip_conntrack_pptp simfs sha1 arc4 ppp_mppe ppp_deflate zlib_defla
te bsd_comp ppp_async crc_ccitt xt_mac ppp_generic slhc vmnet vmmon nfs vznetdev
vzethdev vzrst vzcpt vzdquota vzmon vzdev xt_length ipt_ttl xt_tcpmss ipt_TCPMS
S iptable_mangle xt_limit ipt_tos ipt_REJECT nfsd exportfs lockd nfs_acl sunrpc
iptable_nat ip_nat xt_tcpudp xt_multiport ipt_LOG xt_state ip_conntrack nfnetlin
k iptable_filter ip_tables x_tables tun capi capifs nls_iso8859_1 nls_cp437 vfat
fat loop ipv6 xfs ext2 raid1 cpufreq_userspace powernow_k8 freq_table fcpci ker
nelcapi serio_raw floppy i2c_nforce2 evdev pcspkr parport_pc parport i2c_core ex
t3 jbd mbcache dm_mirror dm_snapshot dm_mod raid456 md_mod xor ide_generic sd_mo
d ide_cd cdrom sata_nv libata scsi_mod via_rhine mii ehci_hcd forcedeth generic
amd74xx ide_core ohci_hcd thermal processor fan
Apr 27 23:33:50 janus kernel: Pid: 27433, comm: vzmond/101 Tainted: P 2.6.
18-1-openvz #2

Apr 27 23:33:50 janus kernel: RIP: 0060:[<ffffffff880a1bdc>] [<ffffffff880a1bdc
>] :ip6_tables:ip6t_unregister_table+0xe/0x16f
Apr 27 23:33:50 janus kernel: RSP: 0000:ffff8100884b9e80 EFLAGS: 00010282
Apr 27 23:33:50 janus kernel: RAX: ffff810010f00000 RBX: 0000000000000000 RCX: f
fff810010f00000
Apr 27 23:33:50 janus kernel: RDX: ffff81001932a540 RSI: 0000000000000005 RDI: 0
000000000000000
Apr 27 23:33:50 janus kernel: RBP: ffff810010f00000 R08: 0000000000000246 R09: f
fff810126a0ee00
Apr 27 23:33:50 janus kernel: R10: ffff810126a0ee00 R11: ffffffff80399c38 R12: 0
000000000000007
Apr 27 23:33:50 janus kernel: R13: ffffffff885b4f80 R14: ffff81011b95adc0 R15: f
fffc200107374e0
Apr 27 23:33:50 janus kernel: FS: 00002ac34c20d6d0(0000) GS:ffffffff8052d000(00
00) knlGS:00000000b2a94bb0
Apr 27 23:33:50 janus kernel: CS: 0060 DS: 0000 ES: 0000 CR0: 000000008005003b
Apr 27 23:33:50 janus kernel: CR2: 0000000000000040 CR3: 0000000113430000 CR4: 0
0000000000006e0
Apr 27 23:33:50 janus kernel: Process vzmond/101 (pid: 27433, veid=0, threadinfo
 ffff8100884b8000, task ffff810120774620)
Apr 27 23:33:50 janus kernel: Stack: ffff8100bf7337c0 0000000000000000 ffff8100
10f00000 0000000000000007
Apr 27 23:33:50 janus kernel: ffffffff885b4f80 ffff81011b95adc0 ffffc200107374e
0 ffffffff880ac031
Apr 27 23:33:50 janus kernel: ffff810010f00000 ffffffff885aaa71 ffff810010f0000
0 ffff810010f00040
Apr 27 23:33:50 janus kernel: Call Trace:
Apr 27 23:33:50 janus kernel: [<ffffffff880ac031>] :ip6table_mangle:fini_ip6tab
le_mangle+0x31/0x4e
Apr 27 23:33:50 janus kernel: [<ffffffff885aaa71>] :vzmon:do_ve_iptables+0x93f/
0xf2e
Apr 27 23:33:50 janus kernel: [<ffffffff885ac17a>] :vzmon:env_cleanup+0xb0/0x16
7
Apr 27 23:33:50 janus kernel: [<ffffffff885ac266>] :vzmon:vzmond_helper+0x35/0x
43
Apr 27 23:33:50 janus kernel: [<ffffffff8025bb28>] child_rip+0xa/0x12
Apr 27 23:33:50 janus kernel: [<ffffffff885ac231>] :vzmon:vzmond_helper+0x0/0x4
3
Apr 27 23:33:50 janus kernel: [<ffffffff8025bb1e>] child_rip+0x0/0x12
Apr 27 23:33:50 janus kernel:
Apr 27 23:33:50 janus kernel:
Apr 27 23:33:50 janus kernel: Code: 4c 8b 77 40 e8 ae 00 41 00 49 89 c5 65 8b 04
 25 2c 00 00 00
Apr 27 23:33:50 janus kernel: RIP [<ffffffff880a1bdc>] :ip6_tables:ip6t_unregis
ter_table+0xe/0x16f
Apr 27 23:33:50 janus kernel: RSP <ffff8100884b9e80>
Apr 27 23:33:50 janus kernel: CR2: 0000000000000040

The system was still running, but I couldn't stop the openvz machine "101" any more after the oops. I think this is a openvz-specific problem. If you need further information, please tell me.

Hope somebody can help me.

Best regards
MrDigi

P.S.: I've already opened a bugzilla bug report: http://bugzilla.openvz.org/show_bug.cgi?id=561

Subject: Re: Kernel OOPS with kernel 2.6.18 and openvz 0.28.18.1
Posted by Vasily Tarasov on Sat, 28 Apr 2007 06:51:51 GMT
View Forum Message <> Reply to Message

Thanks for the report, we'll take a look.

Subject: Re: Kernel OOPS with kernel 2.6.18 and openvz 0.28.18.1
Posted by MrDigi on Mon, 14 May 2007 19:52:44 GMT
View Forum Message <> Reply to Message

no response yet, but problem still occurs.

so pushing this topic upwards and waiting for any suggestions

---

Subject: Re: Kernel OOPS with kernel 2.6.18 and openvz 0.28.18.1
Posted by MrDigi on Mon, 14 May 2007 22:21:47 GMT
View Forum Message <> Reply to Message

That oops occurs in:

net/ipv6/netfilter/ip6_tables.c:

```
void ip6t_unregister_table(struct xt_table *table)
{
        struct xt_table_info *private;
        void *loc_cpu_entry;
        struct module *me;

        me = table->me; <--- HERE is OOPSed, so table was null
        ...
```

It was called from:

net/ipv6/netfilter/ip6table_mangle.c

```
void fini_ip6table_mangle(void)
{
	nf_unregister_hooks(ip6t_ops, ARRAY_SIZE(ip6t_ops));
	ip6t_unregister_table(ve_packet_mangler);
#ifdef CONFIG_VE_IPTABLES
	ve_packet_mangler = NULL;
#endif
}
```

I think the problem is somewhere here in cleanup:

net/ipv6/netfilter/ip6table_mangle.c

```
int init_ip6table_mangle(void)
{
	int ret;
	struct ip6t_table *tmp_mangler;

	/* Register table */
	tmp_mangler = ip6t_register_table(&packet_mangler,
			&initial_table.repl);
	if (IS_ERR(tmp_mangler))
		return PTR_ERR(tmp_mangler);
#ifdef CONFIG_VE_IPTABLES
	ve_packet_mangler = tmp_mangler;
#endif

	/* Register hooks */
	ret = nf_register_hooks(ip6t_ops, ARRAY_SIZE(ip6t_ops));
	if (ret < 0)
		goto cleanup_table;

	return ret;

 cleanup_table:
	ip6t_unregister_table(ve_packet_mangler);
#ifdef CONFIG_VE_IPTABLES
	ve_packet_mangler = NULL;
#endif
	return ret;
}
}
```

So perhaps something like the following two added lines are enough ?!

```
void ip6t_unregister_table(struct xt_table *table)
{
        struct xt_table_info *private;
        void *loc_cpu_entry;
        struct module *me;

+       if (table == null)
+         return;

        me = table->me; <--- HERE is OOPSed, so table was null
        ...
---
```

Best regards
Thimo