
Subject: [PATCH v8 7/9] Display current tcp memory allocation in kmem cgroup

Posted by [Glauber Costa](#) on Mon, 05 Dec 2011 21:35:01 GMT

[View Forum Message](#) <> [Reply to Message](#)

This patch introduces kmem.tcp.usage_in_bytes file, living in the kmem_cgroup filesystem. It is a simple read-only file that displays the amount of kernel memory currently consumed by the cgroup.

Signed-off-by: Glauber Costa <glommer@parallels.com>

Reviewed-by: Hiroyuki Kamezawa <kamezawa.hiroyu@jp.fujitsu.com>

CC: David S. Miller <davem@davemloft.net>

CC: Eric W. Biederman <ebiederm@xmission.com>

Documentation/cgroups/memory.txt | 1 +
net/ipv4/tcp_memcontrol.c | 21 ++++++-----
2 files changed, 22 insertions(+), 0 deletions(-)

diff --git a/Documentation/cgroups/memory.txt b/Documentation/cgroups/memory.txt

index 1c9779a..6922b6c 100644

--- a/Documentation/cgroups/memory.txt

+++ b/Documentation/cgroups/memory.txt

@@ -79,6 +79,7 @@ Brief summary of control files.

memory.independent_kmem_limit # select whether or not kernel memory limits are independent of user limits

memory.kmem.tcp.limit_in_bytes # set/show hard limit for tcp buf memory

+ memory.kmem.tcp.usage_in_bytes # show current tcp buf memory allocation

1. History

diff --git a/net/ipv4/tcp_memcontrol.c b/net/ipv4/tcp_memcontrol.c

index e353390..9481f23 100644

--- a/net/ipv4/tcp_memcontrol.c

+++ b/net/ipv4/tcp_memcontrol.c

@@ -17,6 +17,11 @@ static struct cftype tcp_files[] = {

.read_u64 = tcp_cgroup_read,

.private = RES_LIMIT,

},

+ {

+ .name = "kmem.tcp.usage_in_bytes",

+ .read_u64 = tcp_cgroup_read,

+ .private = RES_USAGE,

+ },

};

static inline struct tcp_memcontrol *tcp_from_cgproto(struct cg_proto *cg_proto)

@@ -167,6 +172,19 @@ static u64 tcp_read_stat(struct mem_cgroup *memcg, int type, u64 default_val)

return res_counter_read_u64(&tcp->tcp_memory_allocated, type);

```

}

+static u64 tcp_read_usage(struct mem_cgroup *memcg)
+{
+ struct tcp_memcontrol *tcp;
+ struct cg_proto *cg_proto;
+
+ cg_proto = tcp_prot.proto_cgroup(memcg);
+ if (!cg_proto)
+ return atomic_long_read(&tcp_memory_allocated) << PAGE_SHIFT;
+
+ tcp = tcp_from_cgproto(cg_proto);
+ return res_counter_read_u64(&tcp->tcp_memory_allocated, RES_USAGE);
+}
+
static u64 tcp_cgroup_read(struct cgroup *cont, struct cftype *cft)
{
    struct mem_cgroup *memcg = mem_cgroup_from_cont(cont);
@@ -176,6 +194,9 @@ static u64 tcp_cgroup_read(struct cgroup *cont, struct cftype *cft)
    case RES_LIMIT:
        val = tcp_read_stat(memcg, RES_LIMIT, RESOURCE_MAX);
        break;
+ case RES_USAGE:
+ val = tcp_read_usage(memcg);
+ break;
    default:
        BUG();
    }
--
```

1.7.6.4
