
Subject: [PATCH v6 04/10] ipc: add new SHM_SET command for sys_shmctl() call
Posted by [Stanislav Kinsbursky](#) on Mon, 15 Oct 2012 15:59:56 GMT
[View Forum Message](#) <> [Reply to Message](#)

New SHM_SET command will be interpreted exactly as IPC_SET, but also will update key, cuid and cgid values. IOW, it allows to change existent key value. The fact, that key is not used is checked before update. Otherwise -EEXIST is returned.

Signed-off-by: Stanislav Kinsbursky <skinsbursky@parallels.com>

```
include/uapi/linux/shm.h | 1 +
ipc/compat.c             | 1 +
ipc/shm.c                | 13 ++++++++--
security/selinux/hooks.c | 1 +
security/smack/smack_lsm.c | 1 +
5 files changed, 15 insertions(+), 2 deletions(-)
```

diff --git a/include/uapi/linux/shm.h b/include/uapi/linux/shm.h

index ec36fa1..d7413fd 100644

--- a/include/uapi/linux/shm.h

+++ b/include/uapi/linux/shm.h

@@ -56,6 +56,7 @@ struct shmid_ds {

/* ipcctl commands */

#define SHM_STAT 13

#define SHM_INFO 14

+#define SHM_SET 15

/* Obsolete, used only for backwards compatibility */

struct shminfo {

diff --git a/ipc/compat.c b/ipc/compat.c

index af30d13..35c750d 100644

--- a/ipc/compat.c

+++ b/ipc/compat.c

@@ -692,6 +692,7 @@ long compat_sys_shmctl(int first, int second, void __user *uptr)

case IPC_SET:

+ case SHM_SET:

if (version == IPC_64) {

err = get_compat_shmid64_ds(&s64, uptr);

} else {

diff --git a/ipc/shm.c b/ipc/shm.c

index 80b0046..aebc50d 100644

--- a/ipc/shm.c

+++ b/ipc/shm.c

@@ -636,6 +636,9 @@ copy_shmid_from_user(struct shmid64_ds *out, void __user *buf, int version)

```

    out->shm_perm.uid = tbuf_old.shm_perm.uid;
    out->shm_perm.gid = tbuf_old.shm_perm.gid;
    out->shm_perm.mode = tbuf_old.shm_perm.mode;
+ out->shm_perm.cuid = tbuf_old.shm_perm.cuid;
+ out->shm_perm.cgid = tbuf_old.shm_perm.cgid;
+ out->shm_perm.key = tbuf_old.shm_perm.key;

    return 0;
}
@@ -740,12 +743,13 @@ static int shmctl_down(struct ipc_namespace *ns, int shmid, int cmd,
    struct shmctl_kernel *shp;
    int err;

- if (cmd == IPC_SET) {
+ if (cmd == IPC_SET || cmd == SHM_SET) {
    if (copy_shmid_from_user(&shmctl64, buf, version))
        return -EFAULT;
    }

- ipcp = ipcctl_pre_down(ns, &shm_ids(ns), shmid, cmd,
+ ipcp = ipcctl_pre_down(ns, &shm_ids(ns), shmid,
+ (cmd != SHM_SET) ? cmd : IPC_SET,
    &shmctl64.shm_perm, 0);
    if (IS_ERR(ipcp))
        return PTR_ERR(ipcp);
@@ -759,6 +763,10 @@ static int shmctl_down(struct ipc_namespace *ns, int shmid, int cmd,
    case IPC_RMID:
        do_shm_rmid(ns, ipcp);
        goto out_up;
+ case SHM_SET:
+ err = ipc_update_key(&shm_ids(ns), &shmctl64.shm_perm, ipcp);
+ if (err)
+ break;
    case IPC_SET:
        err = ipc_update_perm(&shmctl64.shm_perm, ipcp);
        if (err)
@@ -938,6 +946,7 @@ SYSCALL_DEFINE3(shmctl, int, shmid, int, cmd, struct shmctl_ds __user
*, buf)
}
    case IPC_RMID:
    case IPC_SET:
+ case SHM_SET:
    err = shmctl_down(ns, shmid, cmd, buf, version);
    return err;
    default:
diff --git a/security/selinux/hooks.c b/security/selinux/hooks.c
index 24ab414..62b2447 100644
--- a/security/selinux/hooks.c

```

```

+++ b/security/selinux/hooks.c
@@ -5027,6 +5027,7 @@ static int selinux_shm_shmctl(struct shmid_kernel *shp, int cmd)
    perms = SHM__GETATTR | SHM__ASSOCIATE;
    break;
    case IPC_SET:
+ case SHM_SET:
    perms = SHM__SETATTR;
    break;
    case SHM_LOCK:
diff --git a/security/smack/smack_lsm.c b/security/smack/smack_lsm.c
index 38be92c..c7eabc9 100644
--- a/security/smack/smack_lsm.c
+++ b/security/smack/smack_lsm.c
@@ -2121,6 +2121,7 @@ static int smack_shm_shmctl(struct shmid_kernel *shp, int cmd)
    may = MAY_READ;
    break;
    case IPC_SET:
+ case SHM_SET:
    case SHM_LOCK:
    case SHM_UNLOCK:
    case IPC_RMID:

```

Subject: Re: [PATCH v6 04/10] ipc: add new SHM_SET command for sys_shmctl() call

Posted by [Serge E. Hallyn](#) on Tue, 23 Oct 2012 16:27:28 GMT

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

Quoting Stanislav Kinsbursky (skinsbursky@parallels.com):

> New SHM_SET command will be interpreted exactly as IPC_SET, but also will
> update key, cuid and cgid values. IOW, it allows to change existent key value.
> The fact, that key is not used is checked before update. Otherwise -EEXIST is
> returned.

>

> Signed-off-by: Stanislav Kinsbursky <skinsbursky@parallels.com>

Looks sane (modulo 'fallsthrough' comment already requested)

Acked-by: Serge E. Hallyn <serge.hallyn@ubuntu.com>

> ---

> include/uapi/linux/shm.h | 1 +
> ipc/compat.c | 1 +
> ipc/shm.c | 13 ++++++++
> security/selinux/hooks.c | 1 +
> security/smack/smack_lsm.c | 1 +
> 5 files changed, 15 insertions(+), 2 deletions(-)
>

```

> diff --git a/include/uapi/linux/shm.h b/include/uapi/linux/shm.h
> index ec36fa1..d7413fd 100644
> --- a/include/uapi/linux/shm.h
> +++ b/include/uapi/linux/shm.h
> @@ -56,6 +56,7 @@ struct shmid_ds {
> /* ipcctl commands */
> #define SHM_STAT 13
> #define SHM_INFO 14
> +#define SHM_SET 15
>
> /* Obsolete, used only for backwards compatibility */
> struct shminfo {
> diff --git a/ipc/compat.c b/ipc/compat.c
> index af30d13..35c750d 100644
> --- a/ipc/compat.c
> +++ b/ipc/compat.c
> @@ -692,6 +692,7 @@ long compat_sys_shmctl(int first, int second, void __user *uptr)
>
>
> case IPC_SET:
> + case SHM_SET:
> if (version == IPC_64) {
> err = get_compat_shmid64_ds(&s64, uptr);
> } else {
> diff --git a/ipc/shm.c b/ipc/shm.c
> index 80b0046..aebc50d 100644
> --- a/ipc/shm.c
> +++ b/ipc/shm.c
> @@ -636,6 +636,9 @@ copy_shmid_from_user(struct shmid64_ds *out, void __user *buf, int
version)
> out->shm_perm.uid = tbuf_old.shm_perm.uid;
> out->shm_perm.gid = tbuf_old.shm_perm.gid;
> out->shm_perm.mode = tbuf_old.shm_perm.mode;
> + out->shm_perm.cuid = tbuf_old.shm_perm.cuid;
> + out->shm_perm.cgid = tbuf_old.shm_perm.cgid;
> + out->shm_perm.key = tbuf_old.shm_perm.key;
>
> return 0;
> }
> @@ -740,12 +743,13 @@ static int shmctl_down(struct ipc_namespace *ns, int shmid, int cmd,
> struct shmid_kernel *shp;
> int err;
>
> - if (cmd == IPC_SET) {
> + if (cmd == IPC_SET || cmd == SHM_SET) {
> if (copy_shmid_from_user(&shmid64, buf, version))
> return -EFAULT;
> }

```

```

>
> - ipcctl_pre_down(ns, &shm_ids(ns), shmctl, cmd,
> + ipcctl_pre_down(ns, &shm_ids(ns), shmctl,
> + (cmd != SHM_SET) ? cmd : IPC_SET,
> + &shmctl.shm_perm, 0);
> if (IS_ERR(ipcctl))
> return PTR_ERR(ipcctl);
> @@ -759,6 +763,10 @@ static int shmctl_down(struct ipc_namespace *ns, int shmctl, int cmd,
> case IPC_RMID:
> do_shm_rmid(ns, ipcctl);
> goto out_up;
> + case SHM_SET:
> + err = ipc_update_key(&shm_ids(ns), &shmctl.shm_perm, ipcctl);
> + if (err)
> + break;
> case IPC_SET:
> err = ipc_update_perm(&shmctl.shm_perm, ipcctl);
> if (err)
> @@ -938,6 +946,7 @@ SYSCALL_DEFINE3(shmctl, int, shmctl, int, cmd, struct shmctl_ds
__user *, buf)
> }
> case IPC_RMID:
> case IPC_SET:
> + case SHM_SET:
> err = shmctl_down(ns, shmctl, cmd, buf, version);
> return err;
> default:
> diff --git a/security/selinux/hooks.c b/security/selinux/hooks.c
> index 24ab414..62b2447 100644
> --- a/security/selinux/hooks.c
> +++ b/security/selinux/hooks.c
> @@ -5027,6 +5027,7 @@ static int selinux_shm_shmctl(struct shmctl_kernel *shp, int cmd)
> perms = SHM__GETATTR | SHM__ASSOCIATE;
> break;
> case IPC_SET:
> + case SHM_SET:
> perms = SHM__SETATTR;
> break;
> case SHM_LOCK:
> diff --git a/security/smack/smack_lsm.c b/security/smack/smack_lsm.c
> index 38be92c..c7eabc9 100644
> --- a/security/smack/smack_lsm.c
> +++ b/security/smack/smack_lsm.c
> @@ -2121,6 +2121,7 @@ static int smack_shm_shmctl(struct shmctl_kernel *shp, int cmd)
> may = MAY_READ;
> break;
> case IPC_SET:
> + case SHM_SET:

```

> case SHM_LOCK:
> case SHM_UNLOCK:
> case IPC_RMID:
>
> --
> To unsubscribe from this list: send the line "unsubscribe linux-security-module" in
> the body of a message to majordomo@vger.kernel.org
> More majordomo info at <http://vger.kernel.org/majordomo-info.html>
