Subject: [PATCH 1/2] Convert uid hash to hlist
Posted by Pavel Emelianov on Fri, 14 Sep 2007 13:05:34 GMT
View Forum Message <> Reply to Message

Surprisingly, but (spotted by Alexey Dobriyan) the uid hash
still uses list_heads, thus occupying twice as much place as
it could. Convert it to hlist_heads.

Signed-off-by: Pavel Emelyanov <xemul@openvz.org>
Signed-off-by: Alexey Dobriyan <adobriyan@openvz.org>
Cc: Serge Hallyn <serue@us.ibm.com>

---

 include/linux/sched.h | 2 +-
 include/linux/user_namespace.h | 2 +-
 kernel/user.c | 15 ++++++++-------
 kernel/user_namespace.c | 2 +-
 4 files changed, 11 insertions(+), 10 deletions(-)

```
diff --git a/include/linux/sched.h b/include/linux/sched.h
index f6cf87e..a2afa88 100644
--- a/include/linux/sched.h
+++ b/include/linux/sched.h
@@ -535,7 +535,7 @@ struct user_struct {
 #endif

   /* Hash table maintenance information */
- struct list_head uidhash_list;
+ struct hlist_node uidhash_node;
   uid_t uid;
 };

diff --git a/include/linux/user_namespace.h b/include/linux/user_namespace.h
index 1101b0c..b5f41d4 100644
--- a/include/linux/user_namespace.h
+++ b/include/linux/user_namespace.h
@@ -11,7 +11,7 @@

 struct user_namespace {
   struct kref  kref;
- struct list_head uidhash_table[UIDHASH_SZ];
+ struct hlist_head uidhash_table[UIDHASH_SZ];
   struct user_struct *root_user;
 };

diff --git a/kernel/user.c b/kernel/user.c
index e080ba8..add57c7 100644
```

```
--- a/kernel/user.c
+++ b/kernel/user.c
@@ -55,21 +55,22 @@ struct user_struct root_user = {
 /*
  * These routines must be called with the uidhash spinlock held!
  */
-static inline void uid_hash_insert(struct user_struct *up, struct list_head *hashent)
+static inline void uid_hash_insert(struct user_struct *up, struct hlist_head *hashent)
 {
- list_add(&up->uidhash_list, hashent);
+ hlist_add_head(&up->uidhash_node, hashent);
 }

 static inline void uid_hash_remove(struct user_struct *up)
 {
- list_del(&up->uidhash_list);
+ hlist_del(&up->uidhash_node);
 }

-static inline struct user_struct *uid_hash_find(uid_t uid, struct list_head *hashent)
+static inline struct user_struct *uid_hash_find(uid_t uid, struct hlist_head *hashent)
 {
  struct user_struct *user;
+ struct hlist_node *h;

- list_for_each_entry(user, hashent, uidhash_list) {
+ hlist_for_each_entry(user, h, hashent, uidhash_node) {
   if(user->uid == uid) {
    atomic_inc(&user->__count);
    return user;
@@ -118,7 +119,7 @@ void free_uid(struct user_struct *up)

 struct user_struct * alloc_uid(struct user_namespace *ns, uid_t uid)
 {
- struct list_head *hashent = uidhashentry(ns, uid);
+ struct hlist_head *hashent = uidhashentry(ns, uid);
  struct user_struct *up;

  spin_lock_irq(&uidhash_lock);
@@ -207,7 +208,7 @@ static int __init uid_cache_init(void)
    0, SLAB_HWCACHE_ALIGN|SLAB_PANIC, NULL);

  for(n = 0; n < UIDHASH_SZ; ++n)
-  INIT_LIST_HEAD(init_user_ns.uidhash_table + n);
+  INIT_HLIST_HEAD(init_user_ns.uidhash_table + n);

  /* Insert the root user immediately (init already runs as root) */
  spin_lock_irq(&uidhash_lock);
```

```
diff --git a/kernel/user_namespace.c b/kernel/user_namespace.c
index df1d2cf..7af90fc 100644
--- a/kernel/user_namespace.c
+++ b/kernel/user_namespace.c
@@ -39,7 +39,7 @@ static struct user_namespace *clone_user
  kref_init(&ns->kref);

  for (n = 0; n < UIDHASH_SZ; ++n)
-  INIT_LIST_HEAD(ns->uidhash_table + n);
+  INIT_HLIST_HEAD(ns->uidhash_table + n);

  /* Insert new root user.  */
  ns->root_user = alloc_uid(ns, 0);
```

Subject: Re: [PATCH 1/2] Convert uid hash to hlist
Posted by serue on Fri, 14 Sep 2007 18:25:12 GMT
View Forum Message <> Reply to Message

Quoting Pavel Emelyanov (xemul@openvz.org):
> Surprisingly, but (spotted by Alexey Dobriyan) the uid hash
> still uses list_heads, thus occupying twice as much place as
> it could. Convert it to hlist_heads.
>
> Signed-off-by: Pavel Emelyanov <xemul@openvz.org>
> Signed-off-by: Alexey Dobriyan <adobriyan@openvz.org>
> Cc: Serge Hallyn <serue@us.ibm.com>

Looks good.

Signed-off-by: Serge Hallyn <serue@us.ibm.com>

thanks,
-serge

>
> ---
>
> include/linux/sched.h | 2 +-
> include/linux/user_namespace.h | 2 +-
> kernel/user.c | 15 ++++++++-------
> kernel/user_namespace.c | 2 +-
> 4 files changed, 11 insertions(+), 10 deletions(-)
>
> diff --git a/include/linux/sched.h b/include/linux/sched.h
> index f6cf87e..a2afa88 100644
> --- a/include/linux/sched.h
> +++ b/include/linux/sched.h

```
> @@ -535,7 +535,7 @@ struct user_struct {
>  #endif
>
>   /* Hash table maintenance information */
> - struct list_head uidhash_list;
> + struct hlist_node uidhash_node;
>   uid_t uid;
>  };
>
> diff --git a/include/linux/user_namespace.h b/include/linux/user_namespace.h
> index 1101b0c..b5f41d4 100644
> --- a/include/linux/user_namespace.h
> +++ b/include/linux/user_namespace.h
> @@ -11,7 +11,7 @@
>
>  struct user_namespace {
>   struct kref  kref;
> - struct list_head uidhash_table[UIDHASH_SZ];
> + struct hlist_head uidhash_table[UIDHASH_SZ];
>   struct user_struct *root_user;
>  };
>
> diff --git a/kernel/user.c b/kernel/user.c
> index e080ba8..add57c7 100644
> --- a/kernel/user.c
> +++ b/kernel/user.c
> @@ -55,21 +55,22 @@ struct user_struct root_user = {
>  /*
>   * These routines must be called with the uidhash spinlock held!
>   */
> -static inline void uid_hash_insert(struct user_struct *up, struct list_head *hashent)
> +static inline void uid_hash_insert(struct user_struct *up, struct hlist_head *hashent)
>  {
> - list_add(&up->uidhash_list, hashent);
> + hlist_add_head(&up->uidhash_node, hashent);
>  }
>
>  static inline void uid_hash_remove(struct user_struct *up)
>  {
> - list_del(&up->uidhash_list);
> + hlist_del(&up->uidhash_node);
>  }
>
> -static inline struct user_struct *uid_hash_find(uid_t uid, struct list_head *hashent)
> +static inline struct user_struct *uid_hash_find(uid_t uid, struct hlist_head *hashent)
>  {
>   struct user_struct *user;
> + struct hlist_node *h;
```

```
>
> - list_for_each_entry(user, hashent, uidhash_list) {
> + hlist_for_each_entry(user, h, hashent, uidhash_node) {
>    if(user->uid == uid) {
>     atomic_inc(&user->__count);
>     return user;
> @@ -118,7 +119,7 @@ void free_uid(struct user_struct *up)
>
>  struct user_struct * alloc_uid(struct user_namespace *ns, uid_t uid)
>  {
> - struct list_head *hashent = uidhashentry(ns, uid);
> + struct hlist_head *hashent = uidhashentry(ns, uid);
>   struct user_struct *up;
>
>   spin_lock_irq(&uidhash_lock);
> @@ -207,7 +208,7 @@ static int __init uid_cache_init(void)
>     0, SLAB_HWCACHE_ALIGN|SLAB_PANIC, NULL);
>
>   for(n = 0; n < UIDHASH_SZ; ++n)
> -  INIT_LIST_HEAD(init_user_ns.uidhash_table + n);
> +  INIT_HLIST_HEAD(init_user_ns.uidhash_table + n);
>
>   /* Insert the root user immediately (init already runs as root) */
>   spin_lock_irq(&uidhash_lock);
> diff --git a/kernel/user_namespace.c b/kernel/user_namespace.c
> index df1d2cf..7af90fc 100644
> --- a/kernel/user_namespace.c
> +++ b/kernel/user_namespace.c
> @@ -39,7 +39,7 @@ static struct user_namespace *clone_user
>   kref_init(&ns->kref);
>
>   for (n = 0; n < UIDHASH_SZ; ++n)
> -  INIT_LIST_HEAD(ns->uidhash_table + n);
> +  INIT_HLIST_HEAD(ns->uidhash_table + n);
>
>   /* Insert new root user.  */
>   ns->root_user = alloc_uid(ns, 0);
```