
Subject: Re: [RFC][PATCH 2/7] UBC: core (structures, API)

Posted by [dev](#) on Thu, 17 Aug 2006 11:52:37 GMT

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

Rohit Seth wrote:

> On Wed, 2006-08-16 at 19:37 +0400, Kirill Korotaev wrote:

>

>>Core functionality and interfaces of UBC:

>>find/create beancounter, initialization,

>>charge/uncharge of resource, core objects' declarations.

>>

>>Basic structures:

>> ubparm - resource description

>> user_beancounter - set of resources, id, lock

>>

>>Signed-Off-By: Pavel Emelianov <xemul@sw.ru>

>>Signed-Off-By: Kirill Korotaev <dev@sw.ru>

>>

>>---

>> include/ub/beancounter.h | 157 ++++++

>> init/main.c | 4

>> kernel/Makefile | 1

>> kernel/ub/Makefile | 7

>> kernel/ub/beancounter.c | 398 ++++++

>> 5 files changed, 567 insertions(+)

>>

>>--- /dev/null 2006-07-18 14:52:43.075228448 +0400

>>+++ ./include/ub/beancounter.h 2006-08-10 14:58:27.000000000 +0400

>>@@ -0,0 +1,157 @@

>>+/*

>>+ * include/ub/beancounter.h

>>+ *

>>+ * Copyright (C) 2006 OpenVZ. SWsoft Inc

>>+ *

>>+ */

>>+

>>+#ifndef _LINUX_BEANCOUNTER_H

>>+#define _LINUX_BEANCOUNTER_H

>>+

>>+/*

>>+ * Resource list.

>>+ */

>>+

>>+#define UB_RESOURCES 0

>>+

>>+struct ubparm {

>>+ /*

>>+ * A barrier over which resource allocations are failed gracefully.

```

>>+ * e.g. if the amount of consumed memory is over the barrier further
>>+ * sbrk() or mmap() calls fail, the existing processes are not killed.
>>+ */
>>+ unsigned long barrier;
>>+ /* hard resource limit */
>>+ unsigned long limit;
>>+ /* consumed resources */
>>+ unsigned long held;
>>+ /* maximum amount of consumed resources through the last period */
>>+ unsigned long maxheld;
>>+ /* minimum amount of consumed resources through the last period */
>>+ unsigned long minheld;
>>+ /* count of failed charges */
>>+ unsigned long failcnt;
>>+};
>
>

```

> What is the difference between barrier and limit. They both sound like
> hard limits. No?
check `__charge_beancounter_locked` and `severity`.
It provides some kind of soft and hard limits.

```

>>+
>>+/*
>>+ * Kernel internal part.
>>+ */
>>+
>>+ #ifdef __KERNEL__
>>+
>>+ #include <linux/config.h>
>>+ #include <linux/spinlock.h>
>>+ #include <linux/list.h>
>>+ #include <asm/atomic.h>
>>+
>>+ /*
>>+ * UB_MAXVALUE is essentially LONG_MAX declared in a cross-compiling safe form.
>>+ */
>>+ #define UB_MAXVALUE ( (1UL << (sizeof(unsigned long)*8-1)) - 1)
>>+
>>+
>>+ /*
>>+ * Resource management structures
>>+ * Serialization issues:
>>+ * beancounter list management is protected via ub_hash_lock
>>+ * task pointers are set only for current task and only once
>>+ * refcount is managed atomically
>>+ * value and limit comparison and change are protected by per-ub spinlock
>>+ */

```

```
>>+
>>+struct user_beancounter
>>+{
>>+ atomic_t  ub_refcount;
>>+ spinlock_t  ub_lock;
>>+ uid_t  ub_uid;
>
>
> Why uid? Will it be possible to club processes belonging to different
> users to same bean counter.
oh, its a misname. Should be ub_id. it is ID of user_beancounter
and has nothing to do with user id.
```

```
>>+ struct hlist_node hash;
>>+
>>+ struct user_beancounter *parent;
>>+ void  *private_data;
>>+
>
>
> What are the above two fields used for?
the first one is for hierarchical UBs,
see beancounter_findcreate with UB_LOOKUP_SUB.
private_data is probably not used yet :)
```

```
>>+ /* resources statistics and settings */
>>+ struct ubparm  ub_parms[UB_RESOURCES];
>>+};
>>+
>
>
> I presume UB_RESOURCES value is going to change as different resources
> start getting tracked.
what's wrong with it?
```

> I think something like configs should be used for user interface. It
> automatically presents the right interfaces to user land (based on
> kernel implementation). And you wouldn't need any changes in glibc etc.

1. UBC doesn't require glibc modifications.
2. if you think a bit more about it, adding UB parameters doesn't
require user space changes as well.
3. it is possible to add any kind of interface for UBC. but do you like the idea
to grep 200(containers)x20(parameters) files for getting current usages?
Do you like the idea to convert numbers to strings and back w/o
thinking of data types?

Thanks,
Kirill
