
Subject: [PATCH 07/11] SUNRPC: xs tunables per network namespace introduced
Posted by [Stanislav Kinsbursky](#) on Wed, 14 Dec 2011 10:47:43 GMT
[View Forum Message](#) <> [Reply to Message](#)

This patch introduces independent sets of xs tunables for every network namespace context. Their value is accessible via per-net sysctls. Static tunables left as a storage for module params and now used for initialization of new network namespace tunables set.

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

net/sunrpc/netns.h | 6 ++++++
net/sunrpc/xprtsock.c | 47 ++++++++++++++++++++++++++++++++++++++-----
2 files changed, 44 insertions(+), 9 deletions(-)

diff --git a/net/sunrpc/netns.h b/net/sunrpc/netns.h
index da825e5..23d110d 100644

--- a/net/sunrpc/netns.h
+++ b/net/sunrpc/netns.h
@@ -17,6 +17,12 @@ struct sunrpc_net {
 struct ctl_table_header *debug_ctl_header;
 struct ctl_table_header *xs_tunables_header;
#endif
+ unsigned int xprt_udp_slot_table_entries;
+ unsigned int xprt_tcp_slot_table_entries;
+ unsigned int xprt_max_tcp_slot_table_entries;
+ unsigned int xprt_min_resvport;
+ unsigned int xprt_max_resvport;
+ unsigned int xs_tcp_fin_timeout;
};

extern int sunrpc_net_id;
diff --git a/net/sunrpc/xprtsock.c b/net/sunrpc/xprtsock.c
index 9e42db6..fe4b04b 100644

--- a/net/sunrpc/xprtsock.c
+++ b/net/sunrpc/xprtsock.c
@@ -89,7 +89,6 @@ static unsigned int xprt_max_resvport_limit = RPC_MAX_RESVPORT;
static ctl_table xs_tunables_table[] = {
 {
 .procname = "udp_slot_table_entries",
- .data = &xprt_udp_slot_table_entries,
 .maxlen = sizeof(unsigned int),
 .mode = 0644,
 .proc_handler = proc_dointvec_minmax,
@@ -98,7 +97,6 @@ static ctl_table xs_tunables_table[] = {
 },
 {

```

    .procname = "tcp_slot_table_entries",
-   .data = &xprt_tcp_slot_table_entries,
    .maxlen = sizeof(unsigned int),
    .mode = 0644,
    .proc_handler = proc_dointvec_minmax,
@@ -107,7 +105,6 @@ static ctl_table xs_tunables_table[] = {
    },
    {
        .procname = "tcp_max_slot_table_entries",
-   .data = &xprt_max_tcp_slot_table_entries,
        .maxlen = sizeof(unsigned int),
        .mode = 0644,
        .proc_handler = proc_dointvec_minmax,
@@ -116,7 +113,6 @@ static ctl_table xs_tunables_table[] = {
    },
    {
        .procname = "min_resvport",
-   .data = &xprt_min_resvport,
        .maxlen = sizeof(unsigned int),
        .mode = 0644,
        .proc_handler = proc_dointvec_minmax,
@@ -125,7 +121,6 @@ static ctl_table xs_tunables_table[] = {
    },
    {
        .procname = "max_resvport",
-   .data = &xprt_max_resvport,
        .maxlen = sizeof(unsigned int),
        .mode = 0644,
        .proc_handler = proc_dointvec_minmax,
@@ -134,7 +129,6 @@ static ctl_table xs_tunables_table[] = {
    },
    {
        .procname = "tcp_fin_timeout",
-   .data = &xs_tcp_fin_timeout,
        .maxlen = sizeof(xs_tcp_fin_timeout),
        .mode = 0644,
        .proc_handler = proc_dointvec_jiffies,
@@ -2874,24 +2868,59 @@ static struct xprt_class xs_bc_tcp_transport = {
int create_xs_tunables_table(struct net *net)
{
    struct sunrpc_net *sn = net_generic(net, sunrpc_net_id);
+ struct ctl_table *table;
+
+ table = xs_tunables_table;
+ if (!net_eq(net, &init_net)) {
+ table = kmemdup(xs_tunables_table, sizeof(xs_tunables_table),
+ GFP_KERNEL);
+ if (table == NULL)

```

```

+ goto err_alloc;
+ }
+
+ table[0].data = &sn->xprt_udp_slot_table_entries;
+ table[1].data = &sn->xprt_tcp_slot_table_entries;
+ table[2].data = &sn->xprt_max_tcp_slot_table_entries;
+ table[3].data = &sn->xprt_min_resvport;
+ table[4].data = &sn->xprt_max_resvport;
+ table[5].data = &sn->xs_tcp_fin_timeout;

- sn->xs_tunables_header = register_sunrpc_sysctl(net, xs_tunables_table);
+ sn->xs_tunables_header = register_sunrpc_sysctl(net, table);
  if (sn->xs_tunables_header == NULL)
- return -ENOMEM;
+ goto err_reg;
+
  return 0;

+err_reg:
+ if (!net_eq(net, &init_net))
+ kfree(table);
+err_alloc:
+ return -ENOMEM;
+ }
+
void destroy_xs_tunables_table(struct net *net)
{
  struct sunrpc_net *sn = net_generic(net, sunrpc_net_id);
+ struct ctl_table *table;
+
+ table = sn->xs_tunables_header->ctl_table_arg;
  unregister_sysctl_table(sn->xs_tunables_header);
  sn->xs_tunables_header = NULL;
+ if (!net_eq(net, &init_net))
+ kfree(table);
+ }
+ #endif

-
int socket_sysctl_init(struct net *net)
{
+ struct sunrpc_net *sn = net_generic(net, sunrpc_net_id);
+
+ sn->xprt_udp_slot_table_entries = xprt_udp_slot_table_entries;
+ sn->xprt_tcp_slot_table_entries = xprt_tcp_slot_table_entries;
+ sn->xprt_max_tcp_slot_table_entries = xprt_max_tcp_slot_table_entries;
+ sn->xprt_min_resvport = xprt_min_resvport;
+ sn->xprt_max_resvport = xprt_max_resvport;

```

```
+ sn->xs_tcp_fin_timeout = xs_tcp_fin_timeout;  
#ifdef RPC_DEBUG  
    return create_xs_tunables_table(net);  
#else
```
