
Subject: [RFC PATCH 1/2] unix sockets: add ability for search for peer from passed root

Posted by [Stanislav Kinsbursky](#) on Fri, 10 Aug 2012 12:57:30 GMT

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

This helper is used stream sockets yet.

All is simple: if non-NULL struct path was passed to unix_find_other(), then vfs_path_lookup() is called instead of kern_path().

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

```
include/net/af_unix.h | 2 ++
net/unix/af_unix.c    | 25 ++++++-----
2 files changed, 20 insertions(+), 7 deletions(-)
```

```
diff --git a/include/net/af_unix.h b/include/net/af_unix.h
```

```
index 2ee33da..559467e 100644
```

```
--- a/include/net/af_unix.h
```

```
+++ b/include/net/af_unix.h
```

```
@@ -67,6 +67,8 @@ struct unix_sock {
```

```
    long unix_inq_len(struct sock *sk);
```

```
    long unix_outq_len(struct sock *sk);
```

```
+int unix_stream_connect_root(struct path *root, struct socket *sock,
```

```
+    struct sockaddr *uaddr, int addr_len, int flags);
```

```
#ifdef CONFIG_SYSCTL
```

```
extern int unix_sysctl_register(struct net *net);
```

```
diff --git a/net/unix/af_unix.c b/net/unix/af_unix.c
```

```
index 641f2e4..a790ebc 100644
```

```
--- a/net/unix/af_unix.c
```

```
+++ b/net/unix/af_unix.c
```

```
@@ -759,7 +759,7 @@ out: mutex_unlock(&u->readlock);
```

```
    return err;
```

```
}
```

```
-static struct sock *unix_find_other(struct net *net,
```

```
+static struct sock *unix_find_other(struct net *net, struct path *root,
```

```
    struct sockaddr_un *sunname, int len,
```

```
    int type, unsigned int hash, int *error)
```

```
{
```

```
@@ -769,7 +769,11 @@ static struct sock *unix_find_other(struct net *net,
```

```
    if (sunname->sun_path[0]) {
```

```
        struct inode *inode;
```

```
-    err = kern_path(sunname->sun_path, LOOKUP_FOLLOW, &path);
```

```
+
```

```
+    if (root)
```

```

+ err = vfs_path_lookup(root->dentry, root->mnt, sunname->sun_path, LOOKUP_FOLLOW,
&path);
+ else
+ err = kern_path(sunname->sun_path, LOOKUP_FOLLOW, &path);
  if (err)
    goto fail;
  inode = path.dentry->d_inode;
@@ -979,7 +983,7 @@ static int unix_dgram_connect(struct socket *sock, struct sockaddr *addr,
  goto out;

restart:
- other = unix_find_other(net, sunaddr, alen, sock->type, hash, &err);
+ other = unix_find_other(net, NULL, sunaddr, alen, sock->type, hash, &err);
  if (!other)
    goto out;

@@ -1053,8 +1057,8 @@ static long unix_wait_for_peer(struct sock *other, long timeo)
  return timeo;
}

-static int unix_stream_connect(struct socket *sock, struct sockaddr *uaddr,
-    int addr_len, int flags)
+int unix_stream_connect_root(struct path *root, struct socket *sock,
+    struct sockaddr *uaddr, int addr_len, int flags)
{
  struct sockaddr_un *sunaddr = (struct sockaddr_un *)uaddr;
  struct sock *sk = sock->sk;
@@ -1098,7 +1102,7 @@ static int unix_stream_connect(struct socket *sock, struct sockaddr
*uaddr,

restart:
/* Find listening sock. */
- other = unix_find_other(net, sunaddr, addr_len, sk->sk_type, hash, &err);
+ other = unix_find_other(net, root, sunaddr, addr_len, sk->sk_type, hash, &err);
  if (!other)
    goto out;

@@ -1227,6 +1231,13 @@ out:
  sock_put(other);
  return err;
}
+EXPORT_SYMBOL_GPL(unix_stream_connect_root);
+
+static int unix_stream_connect(struct socket *sock, struct sockaddr *uaddr,
+    int addr_len, int flags)
+{
+ return unix_stream_connect_root(NULL, sock, uaddr, addr_len, flags);
+}

```

```

static int unix_socketpair(struct socket *socka, struct socket *sockb)
{
@@ -1508,7 +1519,7 @@ restart:
    if (sunaddr == NULL)
        goto out_free;

- other = unix_find_other(net, sunaddr, namelen, sk->sk_type,
+ other = unix_find_other(net, NULL, sunaddr, namelen, sk->sk_type,
    hash, &err);
    if (other == NULL)
        goto out_free;

```

Subject: Re: [RFC PATCH 1/2] unix sockets: add ability for search for peer from passed root

Posted by [bfields](#) on Fri, 10 Aug 2012 18:10:02 GMT

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

On Fri, Aug 10, 2012 at 04:57:30PM +0400, Stanislav Kinsbursky wrote:

> This helper is used stream sockets yet.
> All is simple: if non-NULL struct path was passed to unix_find_other(), then
> vfs_path_lookup() is called instead of kern_path().

I'm having some trouble parsing the changelog. Maybe something like?:

unix sockets: add ability to look up using passed-in root

Export a unix_stream_connect_root() helper that allows a caller to optionally pass in a root path, in which case the lookup will be done relative to the given path instead of the current working directory.

I guess this is a question for the networking people, but: will it cause problems to have sunrpc calling directly into the unix socket code?

(And if so, what would be the alternative: define some variant of sockaddr_un that includes the root path? Something better?)

--b.

```

>
> Signed-off-by: Stanislav Kinsbursky <skinsbursky@parallels.com>
> ---
> include/net/af_unix.h | 2 ++
> net/unix/af_unix.c | 25 ++++++-----
> 2 files changed, 20 insertions(+), 7 deletions(-)
>

```

```

> diff --git a/include/net/af_unix.h b/include/net/af_unix.h
> index 2ee33da..559467e 100644
> --- a/include/net/af_unix.h
> +++ b/include/net/af_unix.h
> @@ -67,6 +67,8 @@ struct unix_sock {
>
> long unix_inq_len(struct sock *sk);
> long unix_outq_len(struct sock *sk);
> +int unix_stream_connect_root(struct path *root, struct socket *sock,
> + struct sockaddr *uaddr, int addr_len, int flags);
>
> #ifdef CONFIG_SYSCTL
> extern int unix_sysctl_register(struct net *net);
> diff --git a/net/unix/af_unix.c b/net/unix/af_unix.c
> index 641f2e4..a790ebc 100644
> --- a/net/unix/af_unix.c
> +++ b/net/unix/af_unix.c
> @@ -759,7 +759,7 @@ out: mutex_unlock(&u->readlock);
> return err;
> }
>
> -static struct sock *unix_find_other(struct net *net,
> +static struct sock *unix_find_other(struct net *net, struct path *root,
> struct sockaddr_un *sunname, int len,
> int type, unsigned int hash, int *error)
> {
> @@ -769,7 +769,11 @@ static struct sock *unix_find_other(struct net *net,
>
> if (sunname->sun_path[0]) {
> struct inode *inode;
> - err = kern_path(sunname->sun_path, LOOKUP_FOLLOW, &path);
> +
> + if (root)
> + err = vfs_path_lookup(root->dentry, root->mnt, sunname->sun_path, LOOKUP_FOLLOW,
> &path);
> + else
> + err = kern_path(sunname->sun_path, LOOKUP_FOLLOW, &path);
> if (err)
> goto fail;
> inode = path.dentry->d_inode;
> @@ -979,7 +983,7 @@ static int unix_dgram_connect(struct socket *sock, struct sockaddr
> *addr,
> goto out;
>
> restart:
> - other = unix_find_other(net, sunaddr, alen, sock->type, hash, &err);
> + other = unix_find_other(net, NULL, sunaddr, alen, sock->type, hash, &err);
> if (!other)

```

```

> goto out;
>
> @@ -1053,8 +1057,8 @@ static long unix_wait_for_peer(struct sock *other, long timeo)
> return timeo;
> }
>
> -static int unix_stream_connect(struct socket *sock, struct sockaddr *uaddr,
> -    int addr_len, int flags)
> +int unix_stream_connect_root(struct path *root, struct socket *sock,
> +    struct sockaddr *uaddr, int addr_len, int flags)
> {
> struct sockaddr_un *sunaddr = (struct sockaddr_un *)uaddr;
> struct sock *sk = sock->sk;
> @@ -1098,7 +1102,7 @@ static int unix_stream_connect(struct socket *sock, struct sockaddr
> *uaddr,
>
> restart:
> /* Find listening sock. */
> - other = unix_find_other(net, sunaddr, addr_len, sk->sk_type, hash, &err);
> + other = unix_find_other(net, root, sunaddr, addr_len, sk->sk_type, hash, &err);
> if (!other)
> goto out;
>
> @@ -1227,6 +1231,13 @@ out:
> sock_put(other);
> return err;
> }
> +EXPORT_SYMBOL_GPL(unix_stream_connect_root);
> +
> +static int unix_stream_connect(struct socket *sock, struct sockaddr *uaddr,
> +    int addr_len, int flags)
> +{
> + return unix_stream_connect_root(NULL, sock, uaddr, addr_len, flags);
> +}
>
> static int unix_socketpair(struct socket *socka, struct socket *sockb)
> {
> @@ -1508,7 +1519,7 @@ restart:
> if (sunaddr == NULL)
> goto out_free;
>
> - other = unix_find_other(net, sunaddr, namelen, sk->sk_type,
> + other = unix_find_other(net, NULL, sunaddr, namelen, sk->sk_type,
> hash, &err);
> if (other == NULL)
> goto out_free;
>
> --

```

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

Subject: Re: [RFC PATCH 1/2] unix sockets: add ability for search for peer from
passed root

Posted by [Stanislav Kinsbursky](#) on Fri, 10 Aug 2012 18:43:59 GMT

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

> On Fri, Aug 10, 2012 at 04:57:30PM +0400, Stanislav Kinsbursky wrote:
>> This helper is used stream sockets yet.
>> All is simple: if non-NULL struct path was passed to unix_find_other(), then
>> vfs_path_lookup() is called instead of kern_path().
> I'm having some trouble parsing the changelog. Maybe something like?:
>
> unix sockets: add ability to look up using passed-in root
>
> Export a unix_stream_connect_root() helper that allows a caller
> to optionally pass in a root path, in which case the lookup will
> be done relative to the given path instead of the current
> working directory.

Yep, your variant is much better. Thanks.

>
> I guess this is a question for the networking people, but: will it cause
> problems to have sunrpc calling directly into the unix socket code?
>
> (And if so, what would be the alternative: define some variant of
> sockaddr_un that includes the root path? Something better?)

That was my first idea. But there are problems with this solution (add
root path to sockaddr_un) :

1) sockaddr_un size will change. I don't know, how this will affect
user-space. Of course, we can introduce something like:

```
struct sockaddr_un_kern {  
    struct sockaddr_un un;  
    struct path *path;  
}
```

But even in this case we need to color this structure somehow (for
example, set path to NULL for simple connect or bind call) . And to add
this color, we have to separate sys_connect () from our
sock->ops->connect() call. And I don't really know how to do it since we
don't have any info about socket type in sys_connect () in hands. I.e.

we have it, but then we have to add some specific UNIX socket logic to completely generic `sys_connect()` and `sys_bind()`.

```
> --b.
>
>> Signed-off-by: Stanislav Kinsbursky <skinsbursky@parallels.com>
>> ---
>> include/net/af_unix.h | 2 ++
>> net/unix/af_unix.c | 25 ++++++++-----
>> 2 files changed, 20 insertions(+), 7 deletions(-)
>>
>> diff --git a/include/net/af_unix.h b/include/net/af_unix.h
>> index 2ee33da..559467e 100644
>> --- a/include/net/af_unix.h
>> +++ b/include/net/af_unix.h
>> @@ -67,6 +67,8 @@ struct unix_sock {
>>
>> long unix_inq_len(struct sock *sk);
>> long unix_outq_len(struct sock *sk);
>> +int unix_stream_connect_root(struct path *root, struct socket *sock,
>> + struct sockaddr *uaddr, int addr_len, int flags);
>>
>> #ifdef CONFIG_SYSCTL
>> extern int unix_sysctl_register(struct net *net);
>> diff --git a/net/unix/af_unix.c b/net/unix/af_unix.c
>> index 641f2e4..a790ebc 100644
>> --- a/net/unix/af_unix.c
>> +++ b/net/unix/af_unix.c
>> @@ -759,7 +759,7 @@ out: mutex_unlock(&u->readlock);
>> return err;
>> }
>>
>> -static struct sock *unix_find_other(struct net *net,
>> +static struct sock *unix_find_other(struct net *net, struct path *root,
>> struct sockaddr_un *sunname, int len,
>> int type, unsigned int hash, int *error)
>> {
>> @@ -769,7 +769,11 @@ static struct sock *unix_find_other(struct net *net,
>>
>> if (sunname->sun_path[0]) {
>> struct inode *inode;
>> - err = kern_path(sunname->sun_path, LOOKUP_FOLLOW, &path);
>> +
>> + if (root)
>> + err = vfs_path_lookup(root->dentry, root->mnt, sunname->sun_path, LOOKUP_FOLLOW,
>> &path);
>> + else
>> + err = kern_path(sunname->sun_path, LOOKUP_FOLLOW, &path);
```

```

>>  if (err)
>>      goto fail;
>>  inode = path.dentry->d_inode;
>> @@ -979,7 +983,7 @@ static int unix_dgram_connect(struct socket *sock, struct sockaddr
*addr,
>>      goto out;
>>
>>  restart:
>> - other = unix_find_other(net, sunaddr, alen, sock->type, hash, &err);
>> + other = unix_find_other(net, NULL, sunaddr, alen, sock->type, hash, &err);
>>  if (!other)
>>      goto out;
>>
>> @@ -1053,8 +1057,8 @@ static long unix_wait_for_peer(struct sock *other, long timeo)
>>  return timeo;
>>  }
>>
>> -static int unix_stream_connect(struct socket *sock, struct sockaddr *uaddr,
>> -    int addr_len, int flags)
>> +int unix_stream_connect_root(struct path *root, struct socket *sock,
>> +    struct sockaddr *uaddr, int addr_len, int flags)
>>  {
>>      struct sockaddr_un *sunaddr = (struct sockaddr_un *)uaddr;
>>      struct sock *sk = sock->sk;
>> @@ -1098,7 +1102,7 @@ static int unix_stream_connect(struct socket *sock, struct sockaddr
*uaddr,
>>
>>  restart:
>>  /* Find listening sock. */
>> - other = unix_find_other(net, sunaddr, addr_len, sk->sk_type, hash, &err);
>> + other = unix_find_other(net, root, sunaddr, addr_len, sk->sk_type, hash, &err);
>>  if (!other)
>>      goto out;
>>
>> @@ -1227,6 +1231,13 @@ out:
>>  sock_put(other);
>>  return err;
>>  }
>> +EXPORT_SYMBOL_GPL(unix_stream_connect_root);
>> +
>> +static int unix_stream_connect(struct socket *sock, struct sockaddr *uaddr,
>> +    int addr_len, int flags)
>> +{
>> + return unix_stream_connect_root(NULL, sock, uaddr, addr_len, flags);
>> +}
>>
>>  static int unix_socketpair(struct socket *socka, struct socket *sockb)
>>  {

```

```
>> @@ -1508,7 +1519,7 @@ restart:
>>   if (sunaddr == NULL)
>>     goto out_free;
>>
>> - other = unix_find_other(net, sunaddr, namelen, sk->sk_type,
>> + other = unix_find_other(net, NULL, sunaddr, namelen, sk->sk_type,
>>     hash, &err);
>>   if (other == NULL)
>>     goto out_free;
>>
>> --
>> To unsubscribe from this list: send the line "unsubscribe linux-nfs" in
>> the body of a message to majordomo@vger.kernel.org
>> More majordomo info at http://vger.kernel.org/majordomo-info.html
```
