
Subject: [PATCH net-2.6.25 2/3] Uninline the __inet_lookup_established function
Posted by [Pavel Emelianov](#) on Wed, 19 Dec 2007 10:53:40 GMT
[View Forum Message](#) <> [Reply to Message](#)

This is -700 bytes from the net/ipv4/built-in.o

add/remove: 1/0 grow/shrink: 1/3 up/down: 340/-1040 (-700)

function	old	new	delta
__inet_lookup_established		- 339	+339
tcp_sacktag_write_queue		2254 2255	+1
tcp_v4_err	1304	973	-331
tcp_v4_rcv	2089	1744	-345
tcp_v4_do_rcv	826	462	-364

Exporting is for dccp module (used via e.g. inet_lookup).

Signed-off-by: Pavel Emelianov <xemul@openvz.org>

```
include/net/inet_hashtables.h | 40 +++-----  
net/ipv4/inet_hashtables.c   | 38 ++++++  
2 files changed, 41 insertions(+), 37 deletions(-)
```

diff --git a/include/net/inet_hashtables.h b/include/net/inet_hashtables.h

index 1a43125..a33eab7 100644

--- a/include/net/inet_hashtables.h

+++ b/include/net/inet_hashtables.h

@@ -374,43 +374,9 @@ typedef __u64 __bitwise __addrpair;

*

* Local BH must be disabled here.

*/

-static inline struct sock *

- __inet_lookup_established(struct inet_hashinfo *hashinfo,

- const __be32 saddr, const __be16 sport,

- const __be32 daddr, const u16 hnum,

- const int dif)

{

- INET_ADDR_COOKIE(acookie, saddr, daddr)

- const __portpair ports = INET_COMBINED_PORTS(sport, hnum);

- struct sock *sk;

- const struct hlist_node *node;

- /* Optimize here for direct hit, only listening connections can

- * have wildcards anyways.

- */

- unsigned int hash = inet_ehashfn(daddr, hnum, saddr, sport);

- struct inet_ehash_bucket *head = inet_ehash_bucket(hashinfo, hash);

- rwlock_t *lock = inet_ehash_lockp(hashinfo, hash);

-

```

- prefetch(head->chain.first);
- read_lock(lock);
- sk_for_each(sk, node, &head->chain) {
- if (INET_MATCH(sk, hash, acookie, saddr, daddr, ports, dif))
- goto hit; /* You sunk my battleship! */
- }
-
- /* Must check for a TIME_WAIT'er before going to listener hash. */
- sk_for_each(sk, node, &head->twchain) {
- if (INET_TW_MATCH(sk, hash, acookie, saddr, daddr, ports, dif))
- goto hit;
- }
- sk = NULL;
-out:
- read_unlock(lock);
- return sk;
-hit:
- sock_hold(sk);
- goto out;
-}
+extern struct sock * __inet_lookup_established(struct inet_hashinfo *hashinfo,
+ const __be32 saddr, const __be16 sport,
+ const __be32 daddr, const u16 hnum, const int dif);

static inline struct sock *
inet_lookup_established(struct inet_hashinfo *hashinfo,
diff --git a/net/ipv4/inet_hashtables.c b/net/ipv4/inet_hashtables.c
index 46f899b..db43ddd 100644
--- a/net/ipv4/inet_hashtables.c
+++ b/net/ipv4/inet_hashtables.c
@@ -190,6 +190,44 @@ sherry_cache:
}
EXPORT_SYMBOL_GPL(__inet_lookup_listener);

+struct sock * __inet_lookup_established(struct inet_hashinfo *hashinfo,
+ const __be32 saddr, const __be16 sport,
+ const __be32 daddr, const u16 hnum,
+ const int dif)
+{
+ INET_ADDR_COOKIE(acookie, saddr, daddr)
+ const __portpair ports = INET_COMBINED_PORTS(sport, hnum);
+ struct sock *sk;
+ const struct hlist_node *node;
+ /* Optimize here for direct hit, only listening connections can
+ * have wildcards anyways.
+ */
+ unsigned int hash = inet_ehashfn(daddr, hnum, saddr, sport);
+ struct inet_ehash_bucket *head = inet_ehash_bucket(hashinfo, hash);

```

```

+ rwlock_t *lock = inet_ehash_lockp(hashinfo, hash);
+
+ prefetch(head->chain.first);
+ read_lock(lock);
+ sk_for_each(sk, node, &head->chain) {
+ if (INET_MATCH(sk, hash, acookie, saddr, daddr, ports, dif))
+ goto hit; /* You sunk my battleship! */
+ }
+
+ /* Must check for a TIME_WAIT'er before going to listener hash. */
+ sk_for_each(sk, node, &head->twchain) {
+ if (INET_TW_MATCH(sk, hash, acookie, saddr, daddr, ports, dif))
+ goto hit;
+ }
+ sk = NULL;
+out:
+ read_unlock(lock);
+ return sk;
+hit:
+ sock_hold(sk);
+ goto out;
+}
+EXPORT_SYMBOL_GPL(__inet_lookup_established);
+
+ /* called with local bh disabled */
+ static int __inet_check_established(struct inet_timewait_death_row *death_row,
+ struct sock *sk, __u16 lport,

```

```
--
```

1.5.3.4
