# 任务

`socket` 的 IPV6 模式：

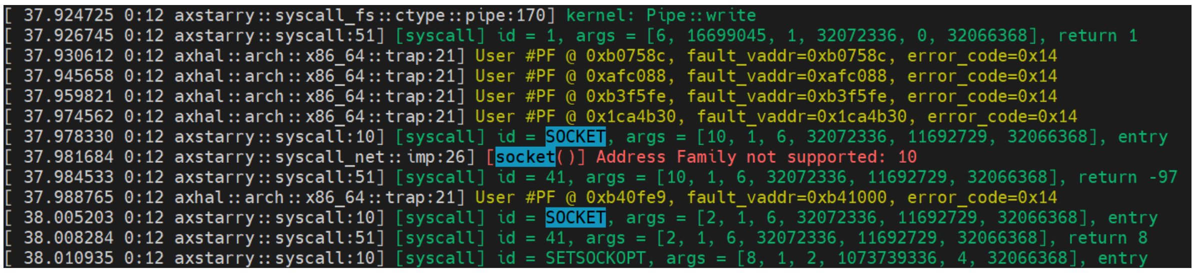
- ZLMediaKit 默认使用 IPv6：`socket(AF_INET6, SOCK_STREAM, IPPROTO_TCP) = 64`。在内核里参照目前的网络栈加个 ipv6 支持难度应该不大？实在不行再去改 ZLMediaKit 配置看能否改 ipv4。
- 还有与之相关的一系列 syscall。具体流程是：

  ```
  socket(AF_INET6, SOCK_STREAM, IPPROTO_TCP) = 64
  setsockopt(64, SOL_SOCKET, SO_REUSEADDR, [1], 4) = 0
  ioctl(64, FIONBIO, [1])           = 0
  fcntl(64, F_GETFD) = 0
  fcntl(64, F_SETFD, FD_CLOEXEC)    = 0
  setsockopt(64, SOL_IPV6, IPV6_V6ONLY, [0], 4) = 0
  bind(64, {sa_family=AF_INET6, sin6_port=htons(554), sin6_flowinfo=htonl(0),
  inet_pton(AF_INET6, "::", &sin6_addr), sin6_scope_id=0}, 28) = 0
  listen(64, 1024)                  = 0
  getsockname(64,{sa_family=AF_INET6, sin6_port=htons(554),
  sin6_flowinfo=htonl(0), inet_pton(AF_INET6, "::", &sin6_addr),
  sin6_scope_id=0}, [128 => 28]) = 0
  getpeername(64, 0x55fa52f82770, [128]) = -1 ENOTCONN (Transport endpoint is
  not connected)
  ```

  之后就是把它扔进 epoll 里不断等待连接了。这里涉及 ipv6 的有 `setsockopt` `bind` 和 `getsockname`。可以找找 libc-test 有没有相关测例可以用。

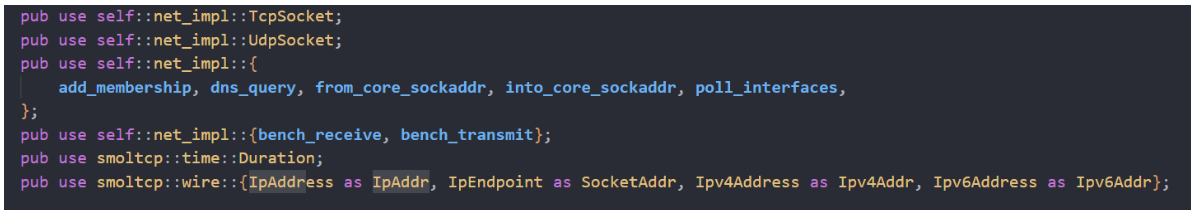
# 支持socket的ipv6模式

再次启动 `./MediaServer -d &`，出现以下问题，socket地址不支持10

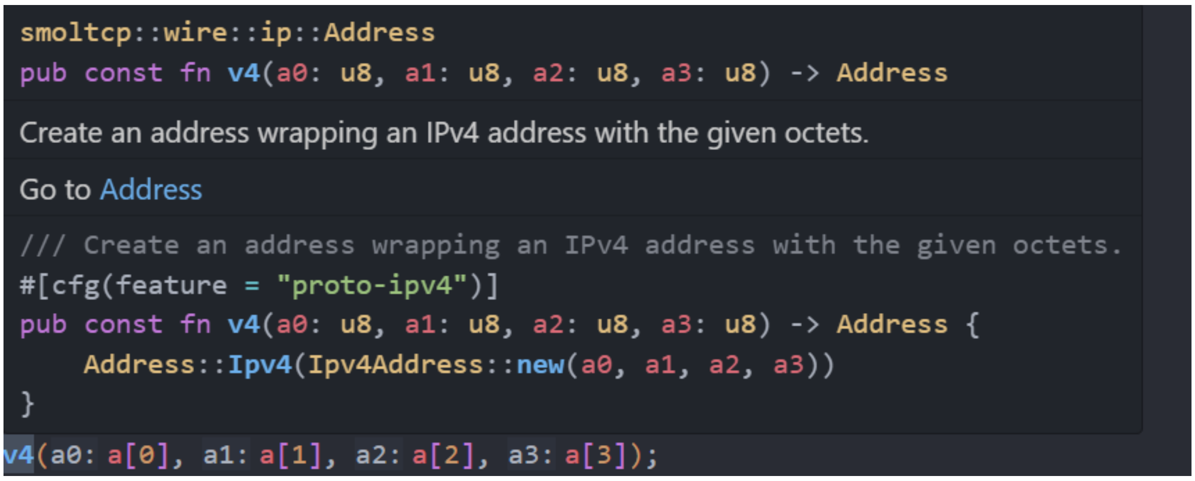


```
[ 37.924725 0:12 axstarry::syscall_fs::ctype::pipe:170] kernel: Pipe::write
[ 37.926745 0:12 axstarry::syscall:51] [syscall] id = 1, args = [6, 16699045, 1, 32072336, 0, 32066368], return 1
[ 37.930612 0:12 axhal::arch::x86_64::trap:21] User #PF @ 0xb0758c, fault_vaddr=0xb0758c, error_code=0x14
[ 37.945658 0:12 axhal::arch::x86_64::trap:21] User #PF @ 0xafc088, fault_vaddr=0xafc088, error_code=0x14
[ 37.959821 0:12 axhal::arch::x86_64::trap:21] User #PF @ 0xb3f5fe, fault_vaddr=0xb3f5fe, error_code=0x14
[ 37.974562 0:12 axhal::arch::x86_64::trap:21] User #PF @ 0x1ca4b30, fault_vaddr=0x1ca4b30, error_code=0x14
[ 37.978330 0:12 axstarry::syscall:10] [syscall] id = SOCKET, args = [10, 1, 6, 32072336, 11692729, 32066368], entry
[ 37.981684 0:12 axstarry::syscall_net::imp:26] [socket()] Address Family not supported: 10
[ 37.984533 0:12 axstarry::syscall:51] [syscall] id = 41, args = [10, 1, 6, 32072336, 11692729, 32066368], return -97
[ 37.988765 0:12 axhal::arch::x86_64::trap:21] User #PF @ 0xb40fe9, fault_vaddr=0xb41000, error_code=0x14
[ 38.005203 0:12 axstarry::syscall:10] [syscall] id = SOCKET, args = [2, 1, 6, 32072336, 11692729, 32066368], entry
[ 38.008284 0:12 axstarry::syscall:51] [syscall] id = 41, args = [2, 1, 6, 32072336, 11692729, 32066368], return 8
[ 38.010935 0:12 axstarry::syscall:10] [syscall] id = SETSOCKOPT, args = [8, 1, 2, 1073739336, 4, 32066368], entry
```

Starry中暂时不支持socket的ipv6，直接添加ipv6的支持难度较大，不易实现，石磊老师建议通过small tcp(不一定支持，或者其它的)等工具封装一个crate，调用它来实现ipv6的方式。



```rust
pub use self::net_impl::TcpSocket;
pub use self::net_impl::UdpSocket;
pub use self::net_impl::{
    add_membership, dns_query, from_core_sockaddr, into_core_sockaddr, poll_interfaces,
};
pub use self::net_impl::{bench_receive, bench_transmit};
pub use smoltcp::time::Duration;
pub use smoltcp::wire::{IpAddress as IpAddr, IpEndpoint as SocketAddr, Ipv4Address as Ipv4Addr, Ipv6Address as Ipv6Addr};
```



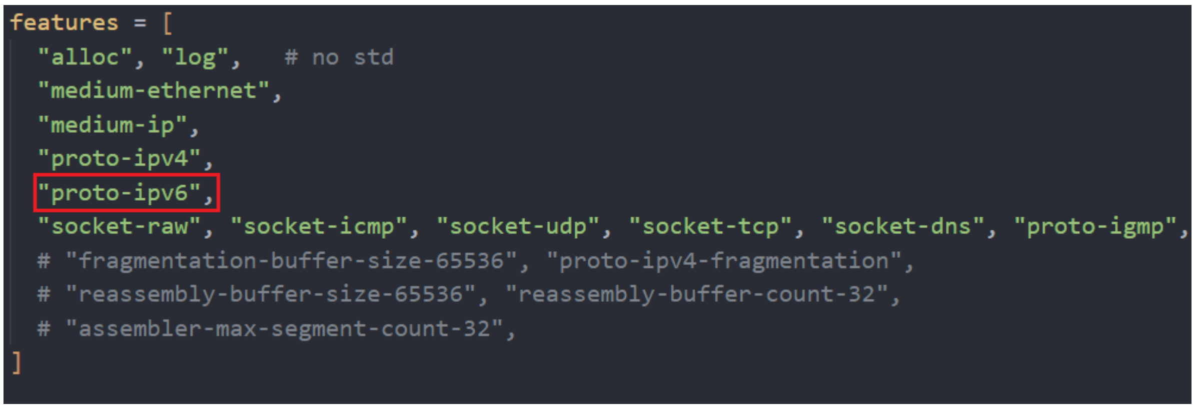
```
smoltcp::wire::ip::Address
pub const fn v4(a0: u8, a1: u8, a2: u8, a3: u8) -> Address
```

Create an address wrapping an IPv4 address with the given octets.

Go to Address

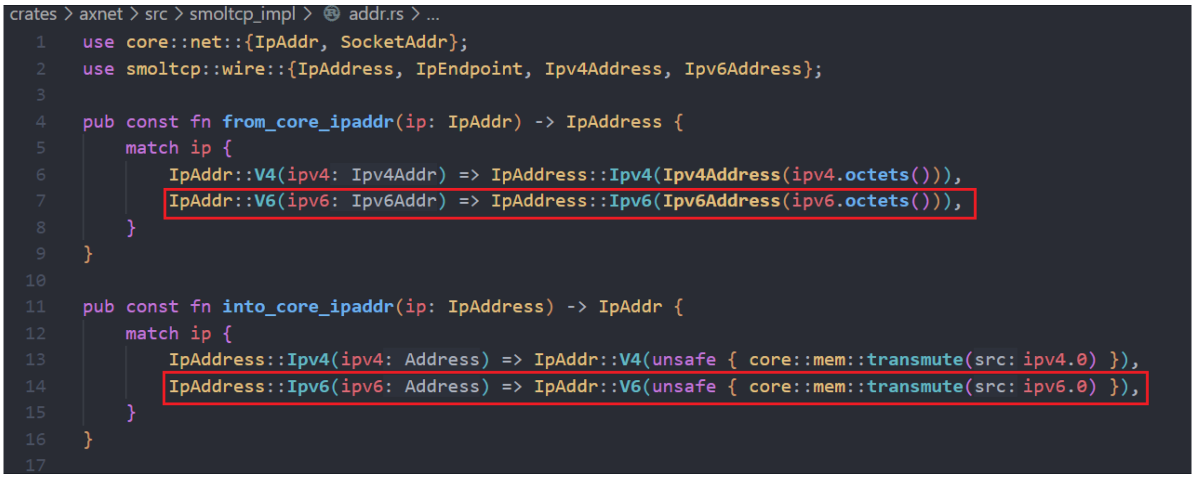
```
/// Create an address wrapping an IPv4 address with the given octets.
#[cfg(feature = "proto-ipv4")]
pub const fn v4(a0: u8, a1: u8, a2: u8, a3: u8) -> Address {
    Address::Ipv4(Ipv4Address::new(a0, a1, a2, a3))
}
v4(a0: a[0], a1: a[1], a2: a[2], a3: a[3]);
```

添加axnet组件，并在/crates/axnet/cargo.toml中开启ipv6特性，在Starry中默认不开启ipv6，需要手动添加



```
features = [
  "alloc", "log",   # no std
  "medium-ethernet",
  "medium-ip",
  "proto-ipv4",
  "proto-ipv6",
  "socket-raw", "socket-icmp", "socket-udp", "socket-tcp", "socket-dns", "proto-igmp",
  # "fragmentation-buffer-size-65536", "proto-ipv4-fragmentation",
  # "reassembly-buffer-size-65536", "reassembly-buffer-count-32",
  # "assembler-max-segment-count-32",
]
```

然后，添加以下ipv6的实现代码，实现对ipv6的调用



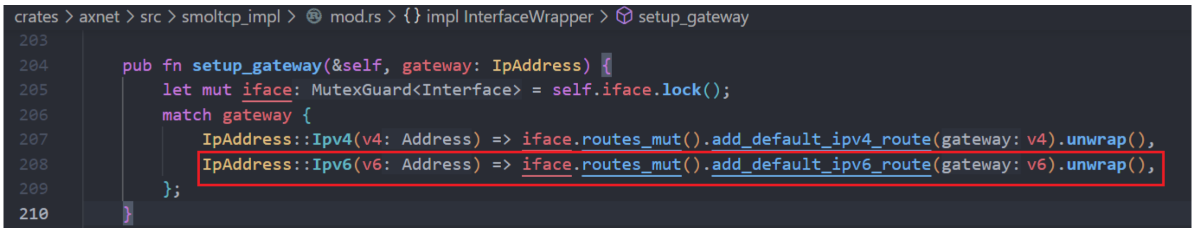
crates > axnet > src > smoltcp_impl > addr.rs > ...

```
use core::net::{IpAddr, SocketAddr};
use smoltcp::wire::{IpAddress, IpEndpoint, Ipv4Address, Ipv6Address};

pub const fn from_core_ipaddr(ip: IpAddr) -> IpAddress {
    match ip {
        IpAddr::V4(ipv4: Ipv4Addr) => IpAddress::Ipv4(Ipv4Address(ipv4.octets())),
        IpAddr::V6(ipv6: Ipv6Addr) => IpAddress::Ipv6(Ipv6Address(ipv6.octets())),
    }
}

pub const fn into_core_ipaddr(ip: IpAddress) -> IpAddr {
    match ip {
        IpAddress::Ipv4(ipv4: Address) => IpAddr::V4(unsafe { core::mem::transmute(src: ipv4.0) }),
        IpAddress::Ipv6(ipv6: Address) => IpAddr::V6(unsafe { core::mem::transmute(src: ipv6.0) }),
    }
}
```

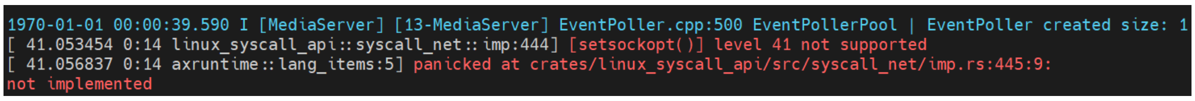
包括这一行



crates > axnet > src > smoltcp_impl > mod.rs > {} impl InterfaceWrapper > setup_gateway

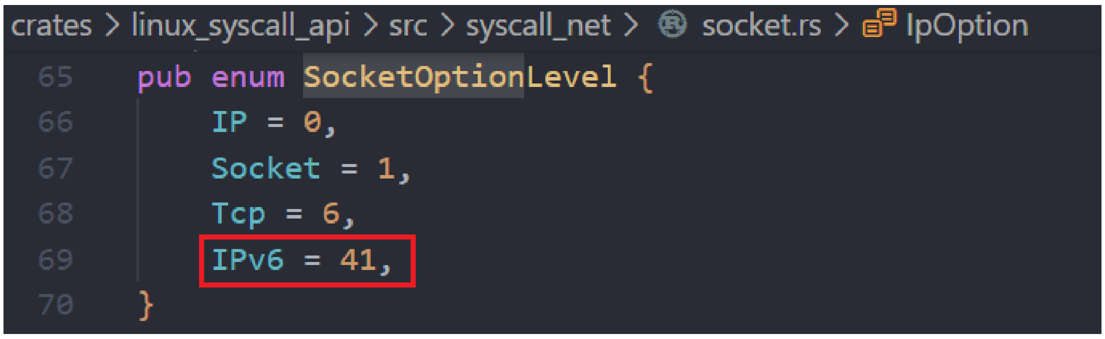
```
    pub fn setup_gateway(&self, gateway: IpAddress) {
        let mut iface: MutexGuard<Interface> = self.iface.lock();
        match gateway {
            IpAddress::Ipv4(v4: Address) => iface.routes_mut().add_default_ipv4_route(gateway: v4).unwrap(),
            IpAddress::Ipv6(v6: Address) => iface.routes_mut().add_default_ipv6_route(gateway: v6).unwrap(),
        };
    }
```

再次运行MediaServer出现报错



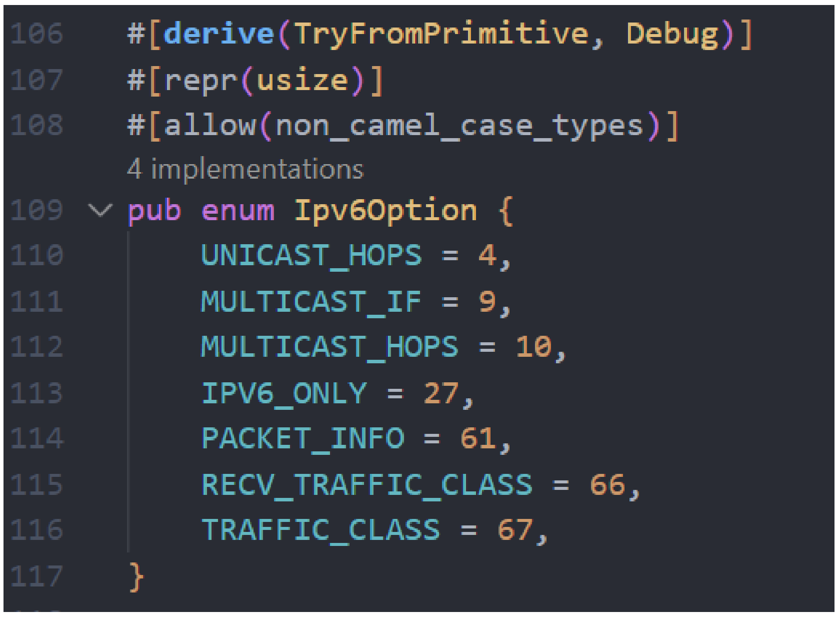
```
1970-01-01 00:00:39.590 I [MediaServer] [13-MediaServer] EventPoller.cpp:500 EventPollerPool | EventPoller created size: 1
[ 41.053454 0:14 linux_syscall_api::syscall_net::imp:444] [setsockopt()] level 41 not supported
[ 41.056837 0:14 axruntime::lang_items:5] panicked at crates/linux_syscall_api/src/syscall_net/imp.rs:445:9:
not implemented
```

添加IPv6Option，有些功能并没用到，暂时不用添加具体实现

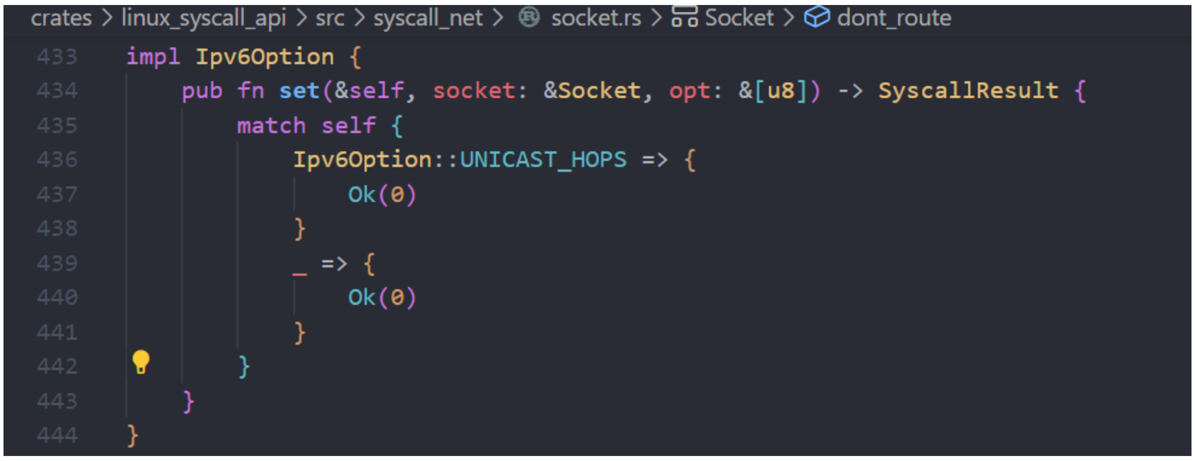


crates > linux_syscall_api > src > syscall_net > socket.rs > IpOption

```rust
pub enum SocketOptionLevel {
    IP = 0,
    Socket = 1,
    Tcp = 6,
    IPv6 = 41,
}
```



```rust
#[derive(TryFromPrimitive, Debug)]
#[repr(usize)]
#[allow(non_camel_case_types)]
4 implementations
pub enum Ipv6Option {
    UNICAST_HOPS = 4,
    MULTICAST_IF = 9,
    MULTICAST_HOPS = 10,
    IPV6_ONLY = 27,
    PACKET_INFO = 61,
    RECV_TRAFFIC_CLASS = 66,
    TRAFFIC_CLASS = 67,
}
```



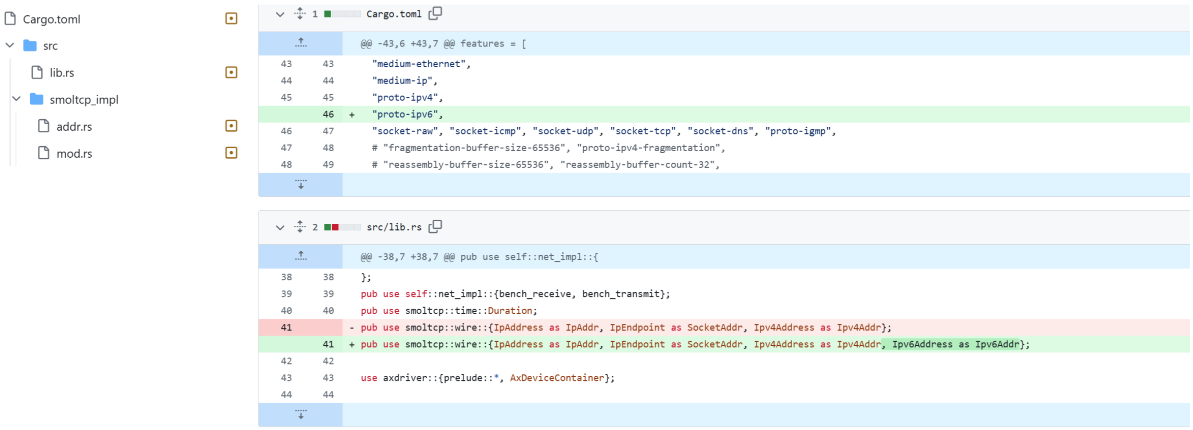
crates > linux_syscall_api > src > syscall_net > socket.rs > Socket > dont_route

```rust
impl Ipv6Option {
    pub fn set(&self, socket: &Socket, opt: &[u8]) -> SyscallResult {
        match self {
            Ipv6Option::UNICAST_HOPS => {
                Ok(0)
            }
            _ => {
                Ok(0)
            }
        }
    }
}
```

## 代码

当前主要完成ipv6的调用逻辑，缺少实际连接的实现，后续工作需补充完整

### axnet



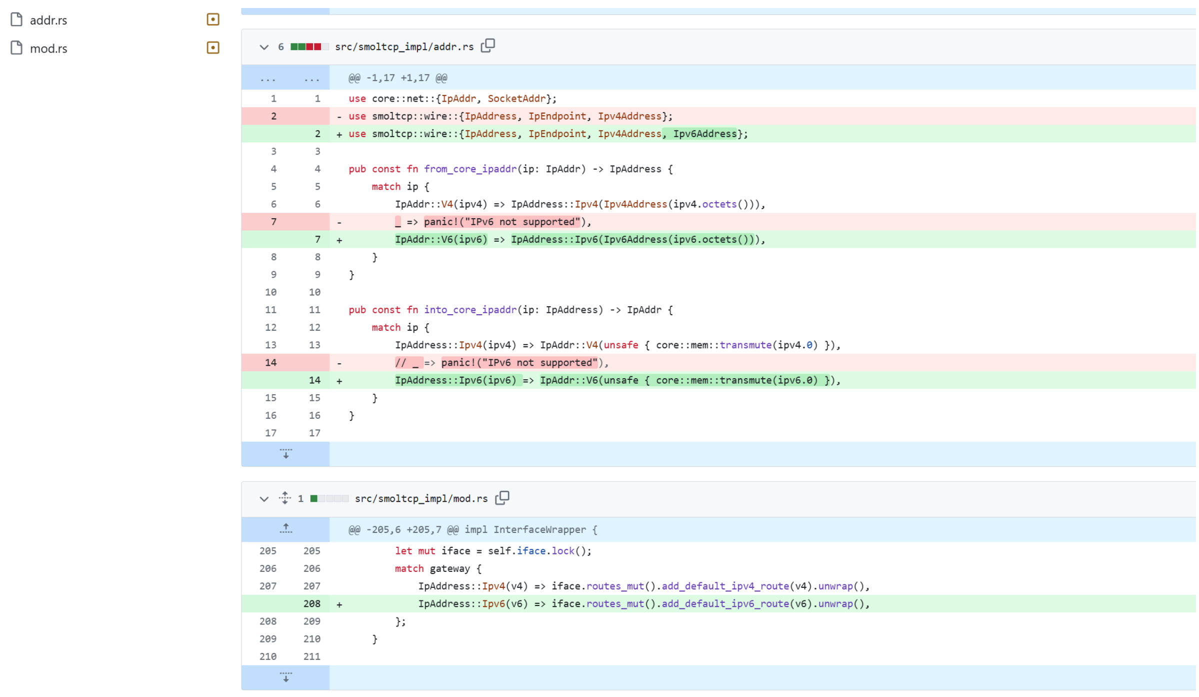
```
Cargo.toml
src
  lib.rs
  smoltcp_impl
    addr.rs
    mod.rs
```

```diff
1 Cargo.toml
@@ -43,6 +43,7 @@ features = [
   "medium-ethernet",
   "medium-ip",
   "proto-ipv4",
+  "proto-ipv6",
   "socket-raw", "socket-icmp", "socket-udp", "socket-tcp", "socket-dns", "proto-igmp",
   # "fragmentation-buffer-size-65536", "proto-ipv4-fragmentation",
   # "reassembly-buffer-size-65536", "reassembly-buffer-count-32",
```

```diff
2 src/lib.rs
@@ -38,7 +38,7 @@ pub use self::net_impl::{
 };
 pub use self::net_impl::{bench_receive, bench_transmit};
 pub use smoltcp::time::Duration;
- pub use smoltcp::wire::{IpAddress as IpAddr, IpEndpoint as SocketAddr, Ipv4Address as Ipv4Addr};
+ pub use smoltcp::wire::{IpAddress as IpAddr, IpEndpoint as SocketAddr, Ipv4Address as Ipv4Addr, Ipv6Address as Ipv6Addr};

 use axdriver::{prelude::*, AxDeviceContainer};

```


addr.rs
mod.rs

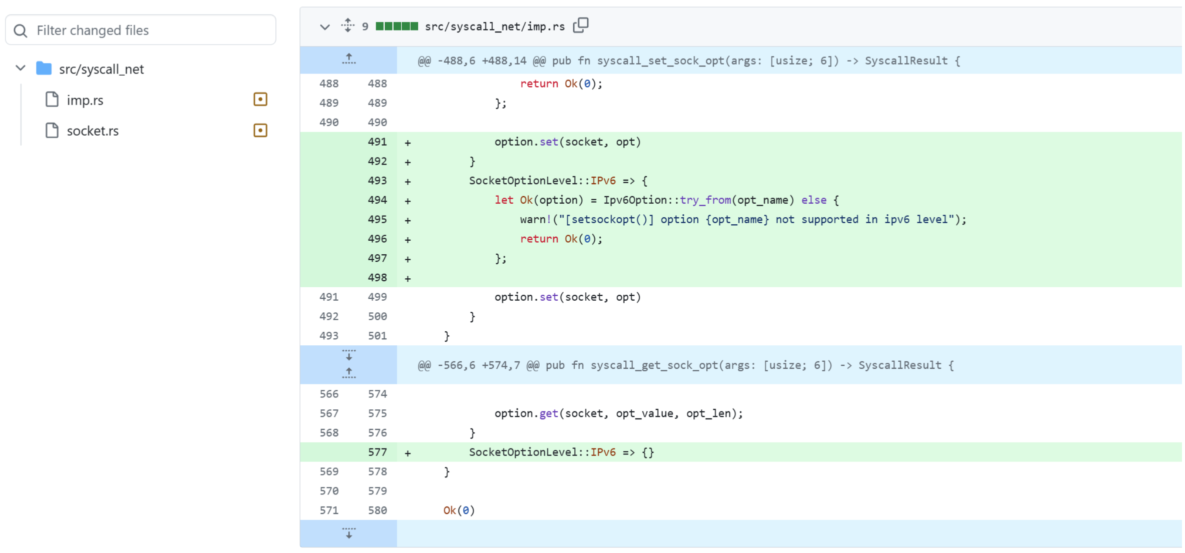
6 src/smoltcp_impl/addr.rs

```diff
@@ -1,17 +1,17 @@
 use core::net::{IpAddr, SocketAddr};
- use smoltcp::wire::{IpAddress, IpEndpoint, Ipv4Address};
+ use smoltcp::wire::{IpAddress, IpEndpoint, Ipv4Address, Ipv6Address};
 
 pub const fn from_core_ipaddr(ip: IpAddr) -> IpAddress {
     match ip {
         IpAddr::V4(ipv4) => IpAddress::Ipv4(Ipv4Address(ipv4.octets())),
-        _ => panic!("IPv6 not supported"),
+        IpAddr::V6(ipv6) => IpAddress::Ipv6(Ipv6Address(ipv6.octets())),
     }
 }
 
 pub const fn into_core_ipaddr(ip: IpAddress) -> IpAddr {
     match ip {
         IpAddress::Ipv4(ipv4) => IpAddr::V4(unsafe { core::mem::transmute(ipv4.0) }),
-        // _ => panic!("IPv6 not supported"),
+        IpAddress::Ipv6(ipv6) => IpAddr::V6(unsafe { core::mem::transmute(ipv6.0) }),
     }
 }
 
```

1 src/smoltcp_impl/mod.rs

```diff
@@ -205,6 +205,7 @@ impl InterfaceWrapper {
         let mut iface = self.iface.lock();
         match gateway {
             IpAddress::Ipv4(v4) => iface.routes_mut().add_default_ipv4_route(v4).unwrap(),
+            IpAddress::Ipv6(v6) => iface.routes_mut().add_default_ipv6_route(v6).unwrap(),
         };
     }
 
```

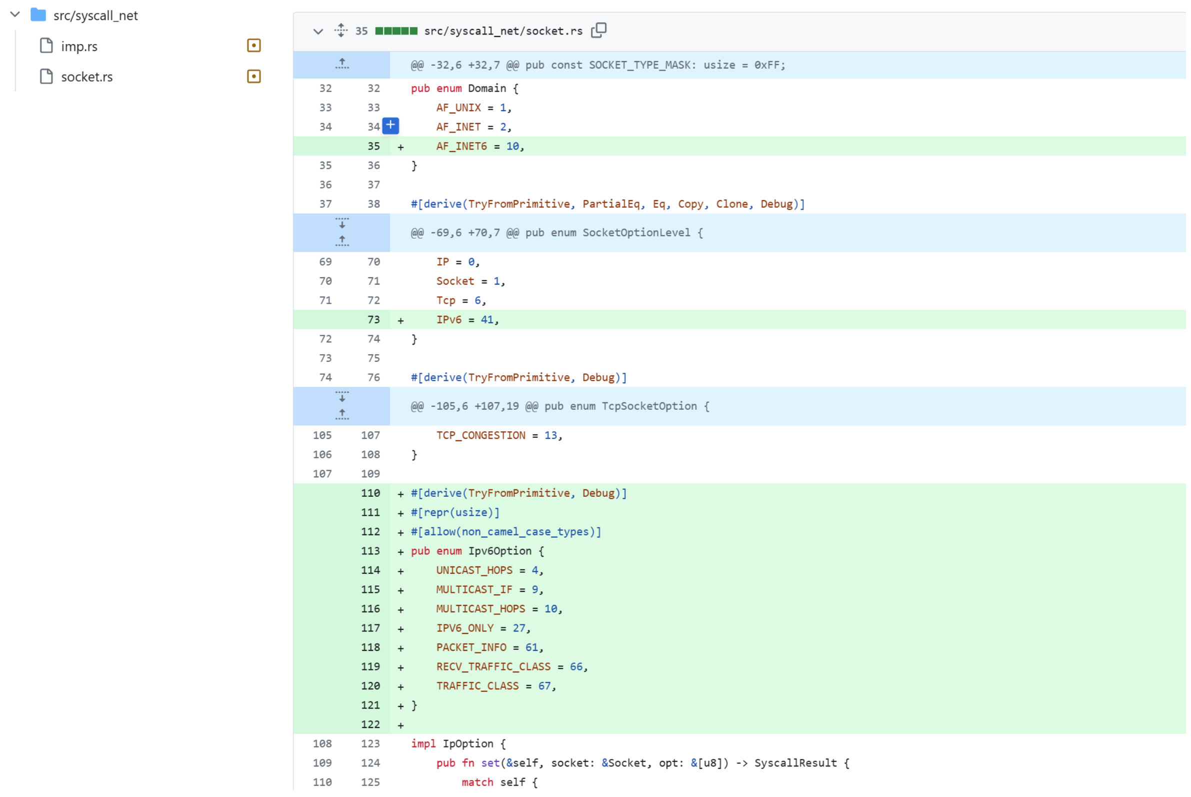
## linux_syscall_api


Filter changed files

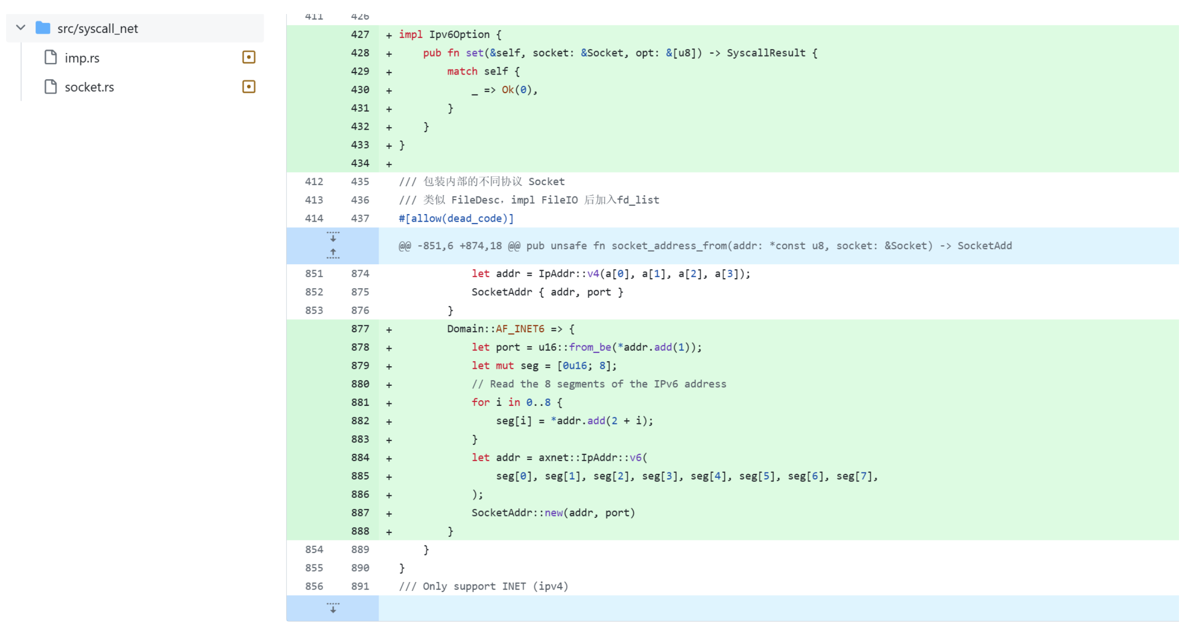
src/syscall_net
imp.rs
socket.rs

9 src/syscall_net/imp.rs

```diff
@@ -488,6 +488,14 @@ pub fn syscall_set_sock_opt(args: [usize; 6]) -> SyscallResult {
                 return Ok(0);
             };
 
+            option.set(socket, opt)
+        }
+        SocketOptionLevel::IPv6 => {
+            let Ok(option) = Ipv6Option::try_from(opt_name) else {
+                warn!("[setsockopt()] option {opt_name} not supported in ipv6 level");
+                return Ok(0);
+            };
+
             option.set(socket, opt)
         }
     }
@@ -566,6 +574,7 @@ pub fn syscall_get_sock_opt(args: [usize; 6]) -> SyscallResult {
 
             option.get(socket, opt_value, opt_len);
         }
+        SocketOptionLevel::IPv6 => {}
     }
 
     Ok(0)
```



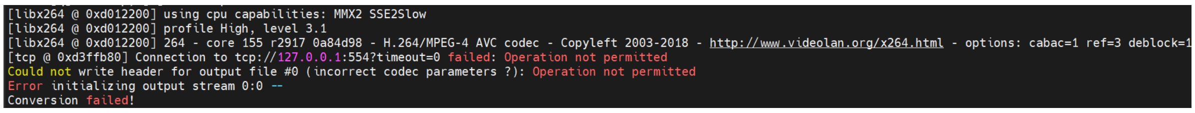
```diff
@@ -32,6 +32,7 @@ pub const SOCKET_TYPE_MASK: usize = 0xFF;
 pub enum Domain {
     AF_UNIX = 1,
     AF_INET = 2,
+    AF_INET6 = 10,
 }
 
 #[derive(TryFromPrimitive, PartialEq, Eq, Copy, Clone, Debug)]
@@ -69,6 +70,7 @@ pub enum SocketOptionLevel {
     IP = 0,
     Socket = 1,
     Tcp = 6,
+    IPv6 = 41,
 }
 
 #[derive(TryFromPrimitive, Debug)]
@@ -105,6 +107,19 @@ pub enum TcpSocketOption {
     TCP_CONGESTION = 13,
 }
 
+#[derive(TryFromPrimitive, Debug)]
+#[repr(usize)]
+#[allow(non_camel_case_types)]
+pub enum Ipv6Option {
+    UNICAST_HOPS = 4,
+    MULTICAST_IF = 9,
+    MULTICAST_HOPS = 10,
+    IPV6_ONLY = 27,
+    PACKET_INFO = 61,
+    RECV_TRAFFIC_CLASS = 66,
+    TRAFFIC_CLASS = 67,
+}
+
 impl IpOption {
     pub fn set(&self, socket: &Socket, opt: &[u8]) -> SyscallResult {
         match self {
```



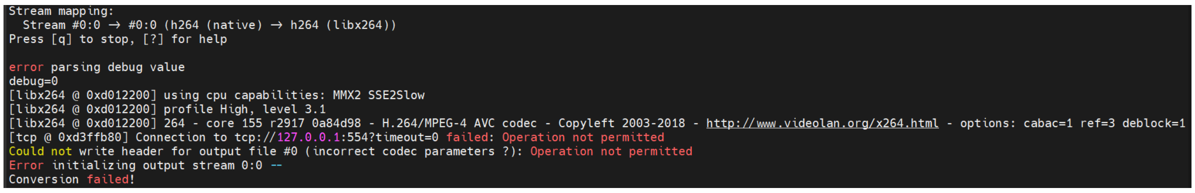
```diff
+impl Ipv6Option {
+    pub fn set(&self, socket: &Socket, opt: &[u8]) -> SyscallResult {
+        match self {
+            _ => Ok(0),
+        }
+    }
+}
+
 /// 包装内部的不同协议 Socket
 /// 类似 FileDesc，impl FileIO 后加入fd_list
 #[allow(dead_code)]
@@ -851,6 +874,18 @@ pub unsafe fn socket_address_from(addr: *const u8, socket: &Socket) -> SocketAdd
             let addr = IpAddr::v4(a[0], a[1], a[2], a[3]);
             SocketAddr { addr, port }
         }
+        Domain::AF_INET6 => {
+            let port = u16::from_be(*addr.add(1));
+            let mut seg = [0u16; 8];
+            // Read the 8 segments of the IPv6 address
+            for i in 0..8 {
+                seg[i] = *addr.add(2 + i);
+            }
+            let addr = axnet::IpAddr::v6(
+                seg[0], seg[1], seg[2], seg[3], seg[4], seg[5], seg[6], seg[7],
+            );
+            SocketAddr::new(addr, port)
+        }
     }
 }
 /// Only support INET (ipv4)
```

## 测试

可以尝试进行连接，虽然连接失败，但至少能够执行连接动作

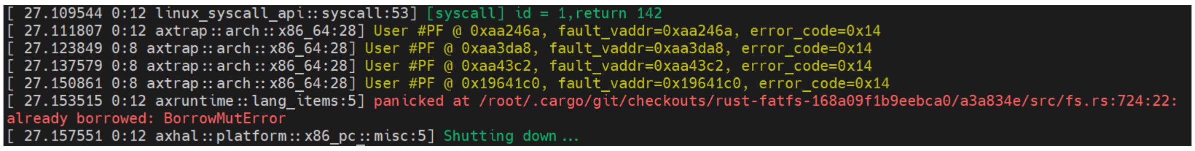


```
[libx264 @ 0xd012200] using cpu capabilities: MMX2 SSE2Slow
[libx264 @ 0xd012200] profile High, level 3.1
[libx264 @ 0xd012200] 264 - core 155 r2917 0a84d98 - H.264/MPEG-4 AVC codec - Copyleft 2003-2018 - http://www.videolan.org/x264.html - options: cabac=1 ref=3 deblock=1
[tcp @ 0xd3ffb80] Connection to tcp://127.0.0.1:554?timeout=0 failed: Operation not permitted
Could not write header for output file #0 (incorrect codec parameters ?): Operation not permitted
Error initializing output stream 0:0 --
Conversion failed!
```



```
Stream mapping:
  Stream #0:0 -> #0:0 (h264 (native) -> h264 (libx264))
Press [q] to stop, [?] for help

error parsing debug value
debug=0
[libx264 @ 0xd012200] using cpu capabilities: MMX2 SSE2Slow
[libx264 @ 0xd012200] profile High, level 3.1
[libx264 @ 0xd012200] 264 - core 155 r2917 0a84d98 - H.264/MPEG-4 AVC codec - Copyleft 2003-2018 - http://www.videolan.org/x264.html - options: cabac=1 ref=3 deblock=1
[tcp @ 0xd3ffb80] Connection to tcp://127.0.0.1:554?timeout=0 failed: Operation not permitted
Could not write header for output file #0 (incorrect codec parameters ?): Operation not permitted
Error initializing output stream 0:0 --
Conversion failed!
```



```
[ 27.109544 0:12 linux_syscall_api::syscall:53] [syscall] id = 1,return 142
[ 27.111807 0:12 axtrap::arch::x86_64:28] User #PF @ 0xaa246a, fault_vaddr=0xaa246a, error_code=0x14
[ 27.123849 0:8 axtrap::arch::x86_64:28] User #PF @ 0xaa3da8, fault_vaddr=0xaa3da8, error_code=0x14
[ 27.137579 0:8 axtrap::arch::x86_64:28] User #PF @ 0xaa43c2, fault_vaddr=0xaa43c2, error_code=0x14
[ 27.150861 0:8 axtrap::arch::x86_64:28] User #PF @ 0x19641c0, fault_vaddr=0x19641c0, error_code=0x14
[ 27.153515 0:12 axruntime::lang_items:5] panicked at /root/.cargo/git/checkouts/rust-fatfs-168a09f1b9eebca0/a3a834e/src/fs.rs:724:22:
already borrowed: BorrowMutError
[ 27.157551 0:12 axhal::platform::x86_pc::misc:5] Shutting down ...
```