Unigornel
Basic Ethernet

Henri Verroken

April 28, 2016

Mini-OS

- Mini-OS can be compiled with lwIP[1]
  - Leightweight TCP/IP stack
  - Written in C
- We want to use a full Go stack

[1] http://savannah.nongnu.org/projects/lwip/

Communication with Xen

- Mini-OS includes code to setup network interface
  - Handle ingoing and outgoing Ethernet frames
  - Implemented in `netfront.c`
  - Enabled if compiling with lwIP
- Not enabled by default

Communication with Xen

```
diff --git a/Makefile b/Makefile
index 0283dc6..3c40030 100644
--- a/Makefile
+++ b/Makefile
@@ -97,7 +97,8 @@ src-y += main.c
 src-y += mmap.c
 src-y += go_main.c
 src-y += mm.c
-src-$(CONFIG_NETFRONT) += netfront.c
+src-y += netfront.c
+src-y += network.c
 src-$(CONFIG_PCIFRONT) += pcifront.c
 src-y += go_pthread.c
 src-y += runtime.c
```

Communication with Xen

- Send or receive an event per frame
- Two functions from `netfront.c`
  - `netfront_xmit(dev, data, len)`
  - `netif_rx(data, len)`

Go Stack

- Keep TCP/IP stack out of the core
- Separate package `go-tcpip`
  - Communicate with Mini-OS using cgo
  - CSP-style approach
  - Currently sending and receiving ethernet packets

Go Stack

```
//export Main
func Main(unused int) {
    payload := []byte("Hello, World!")
    p := ethernet.Packet{
        Destination: ethernet.Broadcast,
        EtherType:   ethernet.EtherType(len(payload)),
        Payload:     payload,
    }

    nic := ethernet.NewNIC()
    go func() {
        for p := range nic.Receive() {
            fmt.Println("Packet", p)
        }
    }()

    for {
        nic.Send() <- p
        time.Sleep(1 * time.Second)
    }
}
```

Unigornel
Basic Ethernet

Henri Verroken

April 28, 2016