# IoT: Client Devices

## Networking Configurations

# Networking in QEMU

## A NUMBER OF NETWORKING MODES

- We'll use SLIRP most (all?) of the time
- You can fall back to TAP mode, though we won't discuss much

## SLIRP IS SLOW

- Lots of overhead, not efficient
- Host has connectivity

## TAP IS MUCH FASTER

- Host network adapter is bridged to Guest VM

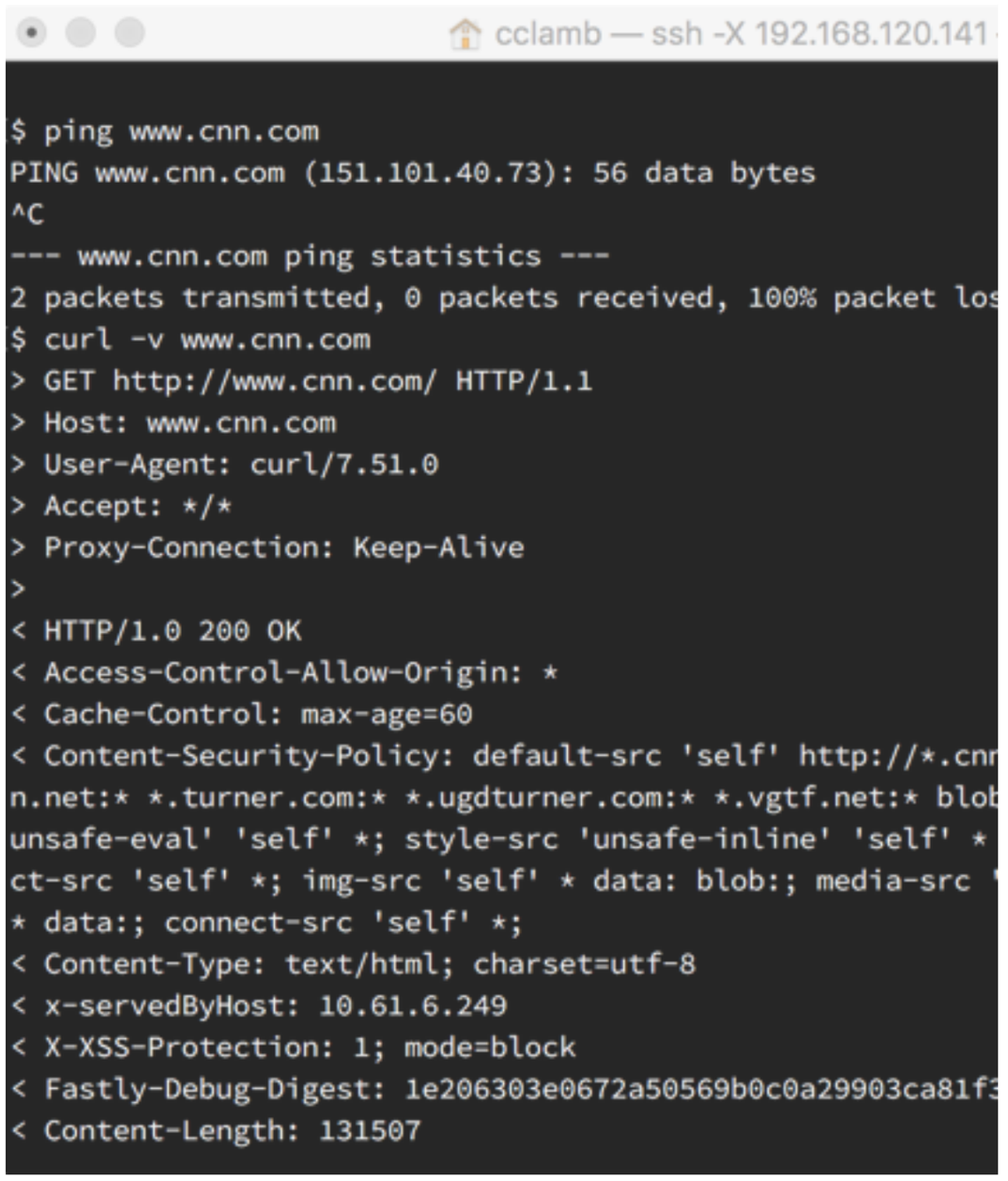
# Bring up your QEMU

## This config uses SLIRP

- Note, in the command that brings it up: **-net user**

## But it seems to work

- Ping doesn't work (you can configure ping to work though)
- Web access does work (we can test this thanks to **curl**)



```
[cclamb@ubuntu:~ $ ifconfig ens33
ens33     Link encap:Ethernet  HWaddr 00:0c:29:7f:9d:45
          inet addr:192.168.120.141  Bcast:192.168.120.255  Mask:255.255.255.0
          inet6 addr: fe80::e029:c10c:4763:4168/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1768 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1122 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:588198 (588.1 KB)  TX bytes:340479 (340.4 KB)

[cclamb@ubuntu:~ $ sudo python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
```

# HTTP to Host OS

Get IP on host and start Python SimpleHTTPServer

```
$ curl -v 192.168.120.141
> GET / HTTP/1.1
> Host: 192.168.120.141
> User-Agent: curl/7.51.0
> Accept: */*
>
< HTTP/1.0 200 OK
< Server: SimpleHTTP/0.6 Python/2.7.12
< Date: Wed, 11 Jan 2017 23:29:41 GMT
< Content-type: text/html; charset=UTF-8
< Content-Length: 2040
<
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 3.2 Final//EN"><html>
<title>Directory listing for /</title>
<body>
<h2>Directory listing for /</h2>
<hr>
<ul>
<li><a href=".bash_history">.bash_history</a>
<li><a href=".bash_logout">.bash_logout</a>
<li><a href=".bashrc">.bashrc</a>
<li><a href=".binwalk/">.binwalk/</a>
<li><a href=".cache/">.cache/</a>
```

# Use Curl on Guest

Send an HTTP GET request to the Host

```
[cclamb@ubuntu:~ $ ifconfig ens33
ens33     Link encap:Ethernet  HWaddr 00:0c:29:7f:9d:45
          inet addr:192.168.120.141  Bcast:192.168.120.255  Mask:255.255.255.0
          inet6 addr: fe80::e029:c10c:4763:4168/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1768 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1122 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:588198 (588.1 KB)  TX bytes:340479 (340.4 KB)

[cclamb@ubuntu:~ $ sudo python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
192.168.120.141 - - [11/Jan/2017 16:29:41] "GET / HTTP/1.1" 200 -
```

# Check the Server

We received a request!

# Why not TAP?

## TAP WORKS FINE

- I've used it, but it requires more work on your part
  - Configure local network topology
  - Hijacks host adapter as a bridge endpoint
  - Guest has IP from Host virtualization solution (in my case, VMWARE)
  - Makes some of what we're doing more difficult

```
#!/bin/sh

ip link add br0 type bridge
ip addr flush dev ens33

ip link set ens33 master br0

tunctl -u $(whoami)

ip link set tap0 master br0

ip link set dev br0 up
ip link set dev tap0 up

qemu-ifup (END)
```

```
#!/bin/sh

ip link set dev ens33 down
ip link set dev br0 down
ip link set dev tap0 down
ip link del dev br0
ip link del dev tap0
ip link set dev ens33 up
qemu-ifdown (END)
```

# TAP Config Files

Run **qemu-ifup** to activate TAP on host, **qemu-ifdown** to tear down, change -**net user** to **-net tap,ifname=tap0**