

## FQA 3 – Hardware



### 3.1 – Selecting Hardware

Selecting appropriate hardware to run your 9front system on is important, as it can mean the difference between success and failure of a project. Fortunately, most common PC hardware is at least minimally functional in Plan 9 (excluding certain exotic audio, VGA, and WiFi devices). Nowadays, thanks to `9boot(8)`, `realemu(8)`, and the VESA driver, it is at least very likely that your PC will boot. In addition, most popular virtualization platforms are reasonably well supported.

Check *FQA 3.2 – Known Working Hardware* as well as the various supported hardware pages on the Bell Labs Plan 9 wiki to help determine if your hardware or VM is supported.

### 3.2 – Known Working Hardware

This list adds to the various supported hardware pages on the Plan 9 from Bell Labs wiki. **Note:** NONE of these lists are all-inclusive. Some drivers listed on the Bell Labs wiki have not been tested by 9front developers. The following list consists of hardware, 1.) that we have actually used, or 2.) about which we have received reliable reports from users.

Some drivers and their options are also documented in `plan9.ini(8)`.

Read: *FQA 1.3.1.2 – New Hardware Support* for information about hardware drivers that are new in 9front.

## **3.2.0 – Input Devices**

### **3.2.0.1 – Mice**

Almost any PS/2 or USB mouse is going to work. The following are preferred for use with Plan 9.

#### **3.2.0.1.1 – IBM/Lenovo**

##### **N700 Wireless/Bluetooth, 3 button Mouse and Laser Pointer**

Part Number: 888015450

DPI: 1200

"Just works" with USB receiver. No additional driver required.



**ScrollPoint Optical Mouse, 3 button, USB/PS2**

Part Number: 31P7405

DPI: 800



**3.2.0.2 – Keyboards**

Almost any AT, PS/2, or USB keyboard is going to work. The following are preferred for use with Plan 9.

### 3.2.0.2.1 – IBM/Lenovo

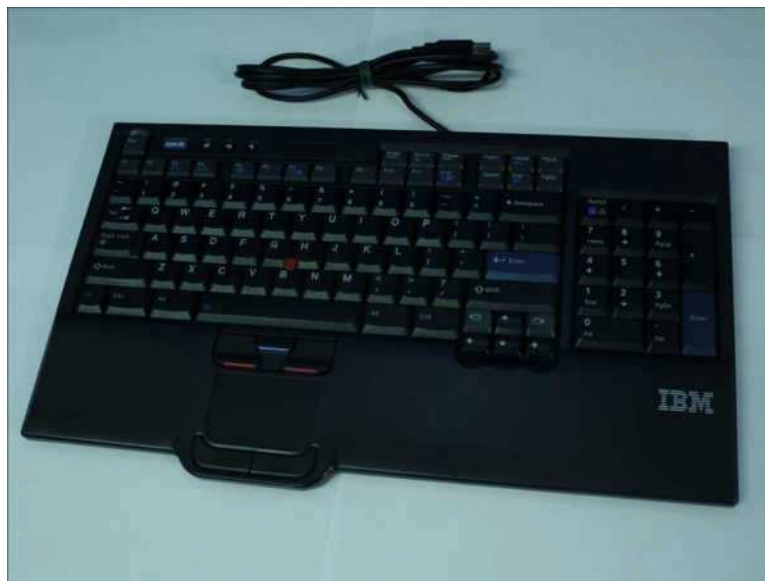
#### IBM Model M 1391401

Part Number: 1391401



#### IBM UltraNav SK-8835

Part Number: SK-8835



### 3.2.0.2.2 – TEX Electronics

#### TEX Shinobi



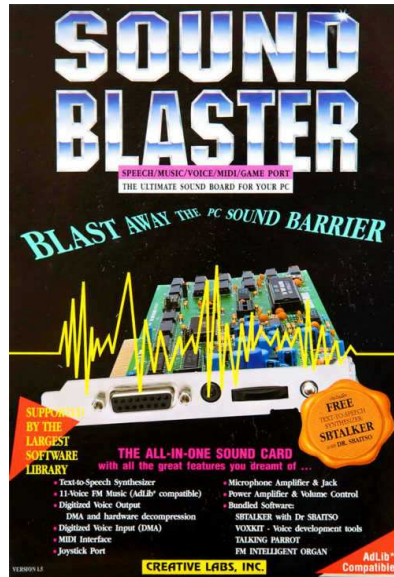
### 3.2.0.2.3 – MNT Research GmbH

#### MNT Reform USB Keyboard



Configuration:  
<http://plan9.stanleylieber.com/hardware/mnt/reform/>

**3.2.1 – Audio** Audio support is much improved in 9front, with added support for AC97, Intel HDA, and (ha!) Soundblaster 16.



**AMD FCH Azalia Controller**

vid/did: 1022/780d

**Intel 888 Microsoft UAA bus for HD audio**

vid/did: 8086/284b

**Intel 82801CA/CAM AC97**

vid/did: 8086/2485

**Intel 82801 DB DBM/DA AC 97**

vid/did: 8086/24c5

**Intel 486486 82801IB/IR/IH HD Audio**

vid/did: 8086/293e

**Intel Gemeni Lake**

vid/did 8086/3198

**Intel HD NM10/ICH7**

vid/did: 8086/27d8

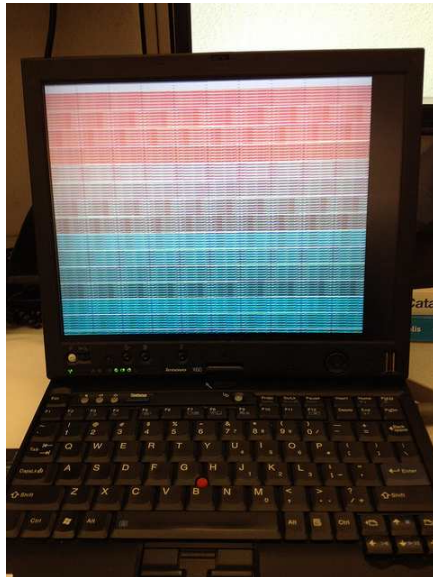
**Intel HD 6 Series/C200 Series**

vid/did: 8086/1c20

**Intel HD 7 Series/C210 Series**

vid/did: 8086/1e20

**3.2.2 – Graphics** Many video cards for which there exists no native VGA driver can be made to work with the generic VESA driver. Examples are provided below.



**3.2.2.1 – AGP**

**NVidia GeForce FX 5200 128MB VGA output**

vid/did: 10de/0322

monitor=vesa vgasize=1600x1200x32

monitor=dellst2210 vgasize=1920x1080x32

**NVidia GeForce FX 5700**

vid/did: 10de/0341

monitor=vesa vgasize=1600x1200x32

monitor=dellst2210 vgasize=1920x1080x32

### 3.2.2.2 – Integrated

#### **ATI Mobility Radeon 7500 128MB DVI/VGA output**

vid/did: 1002/4c57  
monitor=vesa vgasize=1024x768x32

#### **ATI Mobility Radeon FireGL V3200/X600**

vid/did: 1002/3154  
monitor=vesa vgasize=1600x1200x32

#### **ATI RS880**

monitor=vesa vgasize=1280x1024x32

#### **ATI X1300**

#### **Intel Mobile 945GM/GMS/GME, 943/940GML Express**

vid/did: 8086/27a6  
monitor=vesa vgasize=1400x1050x32  
monitor=x60t vgasize=1400x1050x32

#### **Intel X3100/GM965/PM965/GL960**

vid/did: 8086/2a03  
monitor=vesa vgasize=1680x1050x32

#### **Intel Mobile Intel 4 Series 4500MHD**

vid/did: 8086/2a42, 8086/2a43  
monitor=vesa vgasize=1440x900x32  
monitor=x301 vgasize=1440x900x32

#### **Intel HD 3rd Gen Core processor Graphics Controller**

vid/did: 8086/0166  
monitor=vesa vgasize=1366x768x32  
monitor=x230 vgasize=1366x768x32

#### **NVidia GeForce FX Go5200 64M**

vid/did: 10de/0324  
monitor=cinema vgasize=1152x768x32



### **S3 SuperSavage IX/C 16MB**

vid/did: 5333/8c2e  
monitor=t23 vgasize=1024x768x32  
monitor=vesa vgasize=1024x768x32

### **3.2.2.3 – PCI Express**

#### **NVidia GeForce 6200 AGB**

vid/did: 10de/0220

#### **NVidia GeForce 6200 LE**

vid/did: 10de/0163  
monitor=e228wfp vgasize=1680x1050x32

#### **NVidia GeForce 8400 GS**

vid/did: 10de/0422  
monitor=vesa vgasize=1680x1050x32

#### **NVidia GeForce 8600 GT**

vid/did: 10de/0402  
monitor=vesa vgasize=1600x1200x32

#### **NVidia GeForce GTX 550**

vid/did: 10de/0bee  
monitor=vesa vgasize=1600x1200x32

### **3.2.3 – Networking**

#### **3.2.3.1 – Ethernet**

Ethernet is well supported across many vendors and chipsets. 9front introduces a "medium-to-low quality" driver for Broadcom BCM57xx cards, previously unsupported by Plan 9.

##### **3.2.3.1.1 – Integrated**

#### **Broadcom BCM5751M NetXtreme Gigabit**

vid/did: 14e4/167d  
tested 100/1000 mbps

**Broadcom BCM5755/5780 NetXtreme Gigabit**

vid/did: 14e4/167b  
tested 100/1000 mbps

**Broadcom BCM5782 NetXtreme Gigabit**

vid/did: 14e4/1696

**Intel X553/X550-AT 10GBASE-T**

vid/did: 8086/15c8

**Intel 82540EP Gigabit**

vid/did: 8086/101e  
tested 100/1000 mbps

**Intel 82562ET**

tested 10/100 mbps

**Intel 82566MM Gigabit**

vid/did: 8086/1049  
tested 100/1000 mbps

**Intel 82567LM 82567LM-2 Gigabit**

vid/did: 8086/10f5  
tested 100/1000 mbps

**Intel 82573L Gigabit**

vid/did: 8086/109a  
tested: 100/1000 mbps

**Intel 82579LM Gigabit**

vid/did: 8086/1502  
tested: 100/1000 mbps

**Intel 82801CAM PRO/100 VE**

vid/did: 8086/1031  
tested 10/100 mbps

### Realtek RTL8139

vid/did: 10ec/8139  
tested 10/100/1000 mbps

### Realtek RTL8156

### Realtek RTL8169/RTL8101E/RTL8102E

vid/did: 10ec/8136  
tested 10/100/1000 mbps

### 3.2.3.1.2 – USB

#### Beceem Communications CLEAR Stick

vid/did 198f:8160  
This is a WiMAX device that appears as a USB CDC Ethernet device  
Works with nusb/ether

### RNDIS

Android phones should work  
Works with nusb/ether

### 3.2.3.1.3 – PCMCIA

#### 3Com 3c589c

Set the following in `plan9.ini`: `irq=3 port=0x300`

### 3.2.3.2 – WiFi

9front adds support for several WiFi adapters from Ralink and Intel, as well as support for WPA and WPA2.

**Note:** Some WiFi hardware requires a corresponding firmware blob to exist under `/lib/firmware/`. Contents of this directory get included into the kernel paqfs when the kernel is rebuilt, so make sure you don't have so much firmware in there that your kernel gets too large for your machine to boot. This, of course, varies from machine to machine.

Read: `wpa(8)`, and `plan9.ini(8)`

### 3.2.3.2.1 – Bridge (external)

#### IOgear GWU627

802.11n

connect ethernet port to GWU627

HTTP management interface requires Javascript. Managed to program it using Inferno's charon browser, which supports ecma script 1.0.

#### IOgear GWU637

802.11n

connect ethernet port to GWU6

#### Vonets VAP11G

802.11g

connect ethernet port to VAP11G

Requires a proprietary Windows program (ships with the device) to program its settings before using it for the first time.

### 3.2.3.2.2 – Mini-PCI

#### Actiontec 800MIP

802.11b

often branded Lucent WaveLAN

```
ether0=type=wavelanpci ssid=YOUR_AP station=T42 irq=11
```

#### Ralink RT2860 802.11b

### 3.2.3.2.3 – Mini-PCI Express

#### 3.2.3.2.3.1 – iwl

Intel Wireless WiFi Link mini PCI-Express adapters require firmware from <http://git.9front.org/plan9front/firmware/HEAD/info.html> to be present on attach in `/lib/firmware` or `/boot`. To limit the selected APs the options `essid=` and `bssid=` may be set at boot or in the ether interface clone file using a space as the separator between option and value, e.g. `echo essid left-arpit >/net/ether1/clone` Scan results appear in the `ifstats` file and can be read out like: `cat /net/ether1/ifstats` Ad-hoc mode or WEP encryption is currently not supported.

Example configuration for `plan9.ini` :

```
ether0=type=iwl essid=YOUR_AP  
wpapsk=PASSWORD
```

List of relevant Intel WiFi cards in their various hardware configurations:  
<https://ark.intel.com/content/www/us/en/ark/products/series/59485/wireless.html>

**Note:** Many of these cards come in different configurations (sometimes coupled with Bluetooth, sometimes with different physical dimensions, connectors, or antennas). The specific versions listed below are known to work based on user reports or the author's own testing. Every effort has been made to keep this list accurate and up to date. No refunds.

**Intel Centrino Advanced-N 6205**

vid/did: 8086/0085  
firmware: iwl-6005

**Intel Centrino Advanced-N 6235**

firmware: iwn-6030  
vid/did: 8086/088f

**Intel Centrino Ultimate-N**

firmware: iwl-6000

**Intel Centrino Wireless-N 100**

**Intel Centrino Wireless-N 2200/2230**

vid/did: 8086/0891

**Intel WiFi Link 1000/5350 AGN**

**Intel Wireless AC 3160**

**Intel Wireless 4965 AG or AGN**

vid/did: 1180/0476

**Intel Wireless 5100 AGN**

firmware: iwn-5000  
vid/did: 104c/ac56

### **Intel Ultimate N WiFi Link 5300**

firmware: iwn-5000  
vid/did: 1180/0476

### **Intel 5300 AGN**

firmware: iwn-5000  
vid/did: 8086/444e

### **Intel Wireless AC 7260**

firmware: iwm-7260  
vid/did: 8086/08b2

### **Intel Wireless AC 8260**

firmware: iwm-8000C-34

### **Intel Wireless 8265/8275**

firmware: iwm-8265-34  
vid/did: 8086/15c0

### **Intel Wireless AC 9260**

firmware: iwm-9260-34

#### **3.2.3.2.3.2 – wpi**

Intel PRO Wireless 3945abg PCI/PCI-Express wireless adapters require firmware from <http://git.9front.org/plan9front/firmware/HEAD/info.html> to be present on attach in `/lib/firmware` or `/boot`. See the `iwl` section above for configuration details.

Example configuration for `plan9.ini`:

```
ether0=type=wpi essid=YOUR_AP  
wpapsk=PASSWORD
```

#### **Intel PRO Wireless 3945ABG**

firmware: wpi-3945abg  
vid/did: 1180/0476

#### **3.2.3.2.3.3 – rt2860**

Ralink Technology PCI/PCI-Express wireless adapters require firmware from [http://firmware.openbsd.org/firmware/\\*/ral-firmware\\*.tgz](http://firmware.openbsd.org/firmware/*/ral-firmware*.tgz) to be present on attach in `/lib/firmware` or `/boot`. See the `aboveiwl` section

### **Ralink RT3090**

802.11g

### **3.2.3.2.4 – PCI**

### **Ralink RT3090**

802.11b

### **3.2.3.2.5 – PCMCIA**

### **Linksys WPC11**

802.11b  
Prism 2.5  
ISL37300P  
RevA

### **Lucent WaveLAN PC24E-H-FC**

802.11b

```
ether0=type=wavelan essid=YOUR_AP crypt=off station=x61 irq=11
```

## **3.2.4 – Tablet Digitizers**

Support for Wacom serial tablets was added in 2012. The touchscreen digitizers in some Lenovo ThinkPads (notably, the X230) also seem to function without need of any drivers (presumably, controlled by the BIOS).

### **3.2.4.1 – Serial**

#### **3.2.4.1.1 – Integrated**

### **Wacom WACF004**

ThinkPad X4\* series tablets

To enable the tablet's serial port in `plan9.ini`:

```
uart2=type=isa port=0x200 irq=5
```

To turn on the tablet:

```
aux/wacom; aux/tablet &
```

### **Wacom WACF008**

ThinkPad X6\* series tablets

To enable the tablet's serial port in `plan9.ini`:

```
uart2=type=isa port=0x200 irq=5
```

To turn on the tablet:

```
aux/wacom; aux/tablet &
```

### **3.2.4.2 – USB**

#### **3.2.4.2.1 – Integrated**

**Wacom (from ThinkPad X230 Tablet, model unknown)**

Treated as a mouse.

**Wacom (from ThinkPad X1 Yoga 3rd Gen, model unknown)**

Treated as a mouse.

#### **3.2.4.2.2 – External**

**Wacom CTE-640**

Treated as a mouse.



### 3.2.5 – Desktop and Laptop Systems



The ever-expanding list of supported desktop and laptop systems has been redacted from this book and moved exclusively online. Access it here: <http://plan9.stanleylieber.com/hardware/>



### 3.3 – Virtual Machines

9front has been tested on several virtual machines. Details below.

**Note:** As a general rule it is a good idea to manually specify a unique MAC address for each virtual machine instance running on the network, to avoid collisions.

#### 3.3.1 – Qemu

The following generic setup is tested with qemu 1.5.0 and 2.0.50 running on Linux, using *FQA 3.3.3 – virtio* for disk and network. This same generic setup should work for most host operating systems.

##### 3.3.1.1 – Installation

Create a sparse disk image:

```
qemu-img create -f qcow2 9front.qcow2.img 30G
```

Boot the 9front.iso:

```
qemu-system-x86_64 -cpu host -m 1024 \  
-net nic,model=virtio,macaddr=00:20:91:37:33:77 -net user \  
-device virtio-scsi-pci,id=scsi \  
-drive if=none,id=vd0,file=9front.qcow2.img \  
-device scsi-hd,drive=vd0 \  
-drive if=none,id=vd1,file=9front.iso \  
-device scsi-cd,drive=vd1,bootindex=0
```

Finally, see: *FQA 4.3 – Performing a simple install*

### 3.3.1.1.1 – arm64 QCOW

The arm64 QCOW image is an arm64 9front image which can be run in QEMU.

It is intended to be used under a hypervisor (such as Linux KVM or macOS Hypervisor.framework), and thus uses the most general subset of arm64 features to ensure compatibility across hardware (4K page sizes, GICv3).

It currently supports XHCI USB and PCIe devices (used for VirtIO), but it is picky about which peripherals it requires since arm64 does not standardize them. It does not provide a graphical interface, and must be driven via serial (you can however use this to open a drawterm connection, and this is recommended!)

Requirements:

- virt-2.12 machine type because later versions place ECAM in higher memory
- GICv3
- VirtIO devices marked "non-transitional"

As well, U-Boot is required to boot the image. As there are no binaries for U-Boot arm64 QEMU, they must be built yourself:

```
; git clone https://source.denx.de/u-boot/u-boot.git
; make qemu_arm64_defconfig
; make
```

This will produce a u-boot .bin which we will use to run the system.

Running

To boot the system, use the correct machine type, GIC version and pass the QCOW:

```
qemu-system-aarch64 -M virt-2.12,gic-version=3 n
  -cpu cortex-a72 -m 4G -smp 4 n
  -bios u-boot.bin n
  -drive file=9front.arm64.qcow2,if=none,id=installer n
  -device virtio-blk-pci-non-transitional,drive=installer n
  -serial stdio
```

For hardware acceleration, pass accel=hvf in the -M line for e.g. macOS Hypervisor.framework.

For installation, create an additional disk:

```
qemu-img create -f qcow2 9front.qcow2 64G
```

And add it to the QEMU machine as follows:

```
-drive file=9front.qcow2,if=none,id=disk n
-device virtio-blk-pci-non-transitional,drive=disk
```

Then, the machine can be installed as per usual over serial.

For networking devices, attach a virtio-net-pci-non-transitional pointed to your network interface (such as user networking).

USB Devices

USB devices can be attached after adding an XHCI PCIe device:

```
-device qemu-xhci,id=xhci -device usb-tablet,bus=xhci.0
```

### 3.3.1.2 – Post-Installation Booting

```
qemu-system-x86_64 -cpu host -m 1024 \  
-net nic,model=virtio,macaddr=00:20:91:37:33:77 -net user \  
-device virtio-scsi-pci,id=scsi \  
-drive if=none,id=vd0,file=9front.qcow2.img \  
-device scsi-hd,drive=vd0
```

#### 3.3.1.2.1 – Multiboot

Multiboot can be used to start the 9front kernel directly, skipping the bootloader step:

```
qemu -kernel 9pc -initrd plan9.ini
```

#### 3.3.1.4 – Networking

User networking is the default and works the same on every platform. More advanced options are particular to specific host operating systems; several are described below.

**Note:** On many operating systems ICMP is limited to the superuser. One consequence is that a VM running with guest networking cannot ping remote hosts.

##### 3.3.1.4.1 – Linux VDE

Install vde2.

Setup a tap interface:

```
sudo tuncctl -u $USER -t tap0
```

Start a virtual switch connected to the tap interface:

```
vde_switch --tap tap0 -daemon
```

Connect the switch to the network of the host. Use DHCP:

```
slirpvde --dhcp --daemon
```

When booting 9front, add the following to the `qemu` command line arguments:

```
-net vde
```

### 3.3.1.4.2 – OpenBSD TAP

Tested: OpenBSD/amd64 6.0-STABLE, qemu-2.6.0

**Note:** Read over this first. Be careful not to clobber any system settings you may already have configured. If you don't understand something, read the relevant man pages until you do. Feel free to substitute arbitrary network values below.

```
# as root
pkg_add bzip2 plan9port qemu ssvnc wget
cp -f /usr/local/plan9/bin/rc /bin/      # for scripts
sysctl net.inet.ip.forwarding=1
echo 'net.inet.ip.forwarding=1' >>/etc/sysctl.conf
echo inet 192.168.54.1 255.255.255.0 NONE >/etc/hostname.vether0
ed /etc/pf.conf
/ext_if
a
int_if="vether0"

match out from $int_if:network to any nat-to ($ext_if:0)
.
w
q
pfctl -f /etc/pf.conf
echo link0 up >/etc/hostname.tap0
echo add vether0 add tap0 up >/etc/hostname.bridge0
sh /etc/netstart
>/etc/dhcpd.conf
ed /etc/dhcpd.conf
i
option domain-name "example.com";
option domain-name-servers 192.168.54.1;

subnet 192.168.54.0 netmask 255.255.255.0 {
    option routers 192.168.54.1;

    range 192.168.54.100 192.168.54.199;
}
.
w
q
rcctl enable dhcpd
rcctl start dhcpd
ed /var/unbound/etc/unbound.conf
/interface
a
    interface: 192.168.54.1
.
/access-control
a
    access-control: 192.168.54.0/24 allow
w
q

rcctl enable unbound
rcctl start unbound
echo 'permit setenv { -ENV PS1=$DOAS_PS1 SSH_AUTH_SOCK } :wheel' \
    >/etc/doas.conf

# as user who is in wheel group
mkdir -p $HOME/9 $HOME/bin
cd $HOME/9
qemu-img -f qcow2 9front.qcow2.img 30G
```

```
# adjust url for current iso
wget http://9front.org/iso/9front-5561.df1dc1ff2475.iso.bz2
bunzip2 9front-5561.df1dc1ff2475.iso.bz2
mv 9front-5561.df1dc1ff2475.iso 9front.iso
cd $HOME/bin
wget http://openbsd.stanleylieber.com/rc/q9
chmod 775 q9
cd
# boot from iso (install)
doas -u root q9 -i
# boot from qcow image (after completing the install)
doas -u root q9
# connect to qemu via vnc
q9 -v
```

### 3.3.1.4.3 – Windows TAP

This is tested with the qemu for windows distribution. Download and run the installer from openvpn to install the windows TAP driver. Create a new TAP interface with the "Add a new TAP virtual ethernet adapter" from the openvpn start menu. Go to the network manager and rename that new TAP interface to something more sane like: "qemu-tap". Configure ip addresses or bridge that interface with the network manager.

Now you should be able to run qemu on that interface:

```
qemu.exe -net nic -net tap,ifname="tap-qemu" ...
```

### 3.3.1.4.4 – Linux TAP

*Contributed by joe9:*

on the host:

```
sudo ip tuntap add dev tap0 mode tap user joe
sudo ip address add 10.0.0.1/24 dev tap0
```

start qemu using (do not need sudo for qemu):

```
SDL_VIDEO_X11_DGAMOUSE=0 qemu-system-x86_64 \
-cpu host -enable-kvm -m 1024 \
-netdev tap,id=eth,ifname=tap0,script=no,downscript=no \
-device e1000,netdev=eth,mac=00:20:91:37:33:77 \
-device virtio-scsi-pci,id=scsi -drive \
if=none,id=vd0,file=9front.qcow2.img \
-device scsi-hd,drive=vd0 \
-usb -usbdevice tablet -sdl \
-ctrl-grab
```

on 9front: add the below line to `/lib/ndb/local`

```
sys=cirno ether=52540000ee03 ip=10.0.0.2 ipmask=255.255.255.0
ipgw=10.0.0.1
dns=10.0.0.1
dom=cirno.9front
```

run: `ip/ipconfig -N`

Now, "ping 10.0.0.2" from linux host and "ip/ping 10.0.0.1" from qemu 9front should work.

check the communication between the vm and the linux host using (on the linux host):

```
sudo tcpdump -nS -vv -i tap0
```

*Contributed by hiro:*

If you want to enable internet access enable NAT forwarding on the linux host (as root).

To do this, first globally enable forwarding:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Enable Masquerading for everything coming from the VM's tap device (eth0 being your host's way to the internet):

```
iptables -t nat -A POSTROUTING -s 10.0.0.0/24 -o eth0 -j MASQUERADE
```

block everything else from being forwarded:

```
iptables -A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -A FORWARD -s 10.0.0.0/24 -i tap0 -j ACCEPT
iptables -P FORWARD DROP
```

### 3.3.1.5 – Audio

Run qemu with the flag `-soundhw sb16` and put the following line in `plan9.ini`:

```
audio0=type=sb16 port=0x220 irq=5 dma=5
```

**Note:** `irq` and `dma` values may vary.

### 3.3.1.6 – Graphics

Use `monitor=vesa`

**Note:** Some versions of QEMU running on OSX have exhibited graphical glitches when using a 16-bit color mode (for example: 1024x768x16. Try a 32-bit mode instead (for example: 1024x768x32).



### 3.3.2 – Virtualbox

Don't use Virtualbox. It tends to break between versions.



Read: <http://www.landley.net/notes-2015.html#25-06-2015>

If you can't be dissuaded, the following sections detail empirical observations re: Virtualbox.

**3.3.2.1 – Ethernet** The emulated "Intel PRO/1000 MT Server" ethernet controller is known to work.

#### 3.3.2.2 – Audio

Put the following in `plan9.ini`:

```
audio0=type=sb16
```

**3.3.2.3 – Graphics** Use `monitor=vesa`

#### 3.3.2.4 – Known Working Versions

- 4.3.14 r95030 on Windows 7
- 4.3.16 on Mac OS X
- 4.3.18 r96516 on Linux x86\_64 kernel 3.14.22
- 4.3.18 on Windows 7:

just tried with vbox 4.3.18 on windows7. 9front boots fine in BIOS mode, but the PCnet nic dosnt work. reason is that vbox pllX pci irq routing is fucked so the ethernet doesnt get interrupts. if i boot with \*nopcirouting=1, it works fine. theres a option to select the chipset so i tried ICH9 with IO-APIC enabled. normal mp mode fails because of broken mp tables, but works with \*acpi=. also, it works with UEFI mode (which always uses ACPI). the usual intel mt server nic also works (thats what is usually recommended for working around the broken ethernet).

pci routing issue has been fixed in latest kernel, should be available in iso release after 3960.

- 4.3.20 r96996 on Mac OS X 10.6.8/10.9 and Ubuntu 14.04/14.10:

General -> Basic

Type: Other

Version: Other/Uknown

System -> Motherboard

Chipset: PIIX3

Pointing Device: PS/2 Mouse

Extended Features:  Enable I/O APIC

System -> Processor

Extended Features:  PAE/NX (not sure this matters)

System -> Acceleration

Enable VT-x/AMD-V

Enable Nested Paging

Display -> Video

Extended Features:  Enable 3D Acceleration (not sure this matters)

Storage -> Attributes

Name: IDE

Type: PIIX4

Use Host I/O Cache

Audio ->

Enable Audio

Host Audio Driver: CoreAudio (Can be PulseAudio or otherwise for Linux, etc. Shouldn't be hard to set this)

Audio Controller: Soundblaster 16

Network -> Adapter 1

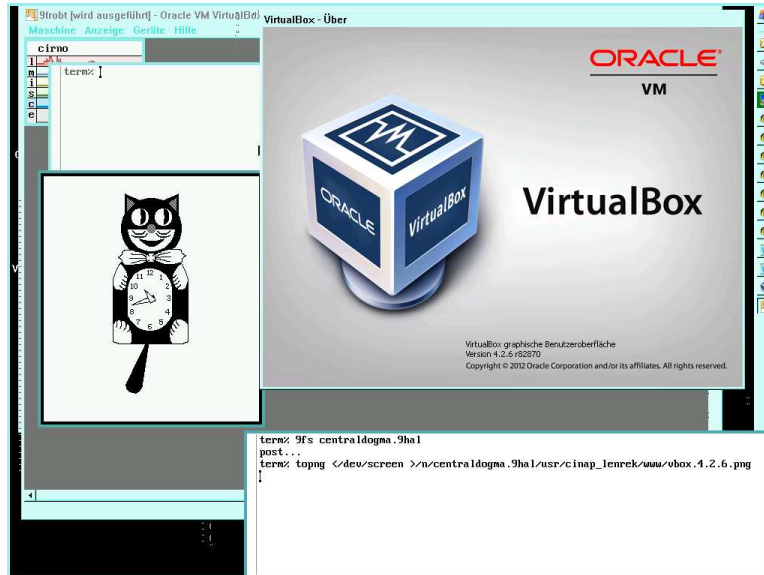
Attached to: NAT

-> Advanced

Adapter Type: Intel PRO/1000 MT Server

Promiscuous Mode: Deny (Not sure this matters)

**Note:** Enabling USB 2.0 Controll in 'Ports -> USB' works just fine in 9front, mounting under /shr flawlessly as long as the host has the Virtualbox Extension Pack running.



### 3.3.3 – Virtio

Current versions of qemu/kvm and virtualbox as of 3.1 support faster paravirtualized devices. Presently, 9front provides drivers for virtio hard disk and network.

The virtio-blk disk device should show up as: `/dev/sdF0`

The virtio-scsi disk device should show up as: `/dev/sd00`

### 3.3.4 – bhyve

Two different guides (and a bonus video) have appeared elsewhere on the Internet:

FreeBSD Wiki: <https://wiki.freebsd.org/bhyve/9front>

9front Wiki: <http://wiki.9front.org/freebsd-bhyve>

Watch: <https://youtu.be/m7igZ1fR7ZA>