admin-marks

Content

enhance virtual-machines

```
apt-get install -y virtualbox-guest-x11
apt-get install -y open-vm-tools-desktop fuse
```

Merge VirtualBox's `Snapshots` into original image manually (in case you 'copied' your VM, not 'cloned' it):

```
VBoxManage clonehd ROSUbuntu1604.vdi ROSUbuntu1604-full.vdi
VBoxManage clonehd Snapshots/\{8a8b278b-db55-4b30-8e00-6460c858b0c2\}.vdi ROSUbuntu1604-full.vdi --existing # do it cc
```

Virtual box C&C:

mount shared folders:

- virtualbox mount: `mount -t vboxsf -o rw hostDir /home/phoenix/hostDir`
- vmware mount: `/usr/bin/vmhgfs-fuse .host:/hostDir /home/phoenix/hostDir -o subtype=vmhgfs-fuse,allow_other` (old): `mount -t vmhgfs .host:/hostDir /home/phoenix/hostDir`

Setting up virtual COM ports for virtual machine at VirtualBox:

- Setting up virtual COM ports under Windows host:
- Setting up virtual COM ports under Linux host:

tune double connection on windows:

```
$LAN_gateway = "10.1.2.3"
route add 10.0.0.0 MASK 255.0.0.0 $LAN_gateway
route add 172.0.0.0 MASK 255.0.0.0 $LAN_gateway

$WAN_gateway = "192.168.1.1"
route add 0.0.0.0 MASK 0.0.0.0 $WAN_gateway metric 25
```

tune connection on linux:

network problems in case of Windows host machine and Wifi adapter

Major drawbacks:

- vmware workstation bridge does not support promiscuous mode
- virtual box bridge may be buggy with Wifi interfaces (sometimes your virtual machine will remain fully disconnected)

Solution:

- Attach all your virtual machines (e.g. you can use vmware and vbox simultaneously) to “Host-only” adapter
- Create window’s bridge for your “Host-only” adapters and Wifi interface
  remark: window’s bridge is NOT a bridge, it is a ***Proxy ARP*** (пояснение)
  remark: window’s bridge will have two mac-addresses: mac-address of your first attached adapter and some randomly generated mac-address for others
- `netsh bridge show adapter` - show adapters in bridge
  `netsh bridge set adapter id=X forcecompatmode=enable` - enable for all adapters compatibility mode (= promiscous mode)

Remaining half-restriction:

- In general Wifi router must accept packets with mac-address separate from you wifi-adapter. However window’s bridge works like Proxy ARP, therefore you may still work with Wifi adapters and even connect adapters from different ip-subnets.
  (probably, nothing you can change here)

- google - `8.8.8.8` `8.8.4.4`
- OpenVPN - `208.67.222.222` `208.67.222.220`

Internel subnets:

- `10.0.0.0/8`
- `172.16.0.0/12`
- `192.168.0.0/16`
- `169.254.0.0/16` - microsoft windows idea

cisco-router short cheatsheet

- `show version` - show cisco IOS version
- `show running-config` - see all configuration
- `show ip dhcp binding` - check current ip-mac associations
  `clear ip dhcp binding 192.168.1.111`
- `show ip route` - show routes
- `configure terminal` - entry configuration mode (vs `exit`)
  - Create new user: `username <ИМЯ ПОЛЬЗОВАТЕЛЯ> privilege 15 secret <ПАРОЛЬ>`
  - Reserve ip-address ranges:
    `ip dhcp excluded-address 192.168.1.100 192.168.1.110`

```
    dns-server 8.8.8.8 8.8.4.4
    default-router 192.168.1.1
```

ideology: creation of pool with mac-addresses which will obtain specified ip-addreses, default route and dns servers

- Create and view routes

```
ip route 10.8.0.0 255.255.255.0 192.168.1.3 name JustAComment
show ip route
sh run | i ip route
```

- Remove any rule: `no <rule>`

- permanent save of cisco configuration: `copy running-config startup-config`

Linux cheatsheet

- tmux scripting
  example of tmux script

- debugging bash scripts:
  snippet

Several openvpn servers can be run:

```
systemctl start openvpn@server2.service
systemctl start openvpn@server.service
```

Setup VPN in NAT mode

Brilliant step-by-step manual (RU). Manual describes how to create certificate center, generate keys, sign certificate, … How to set up systemd service and start it, … how to recall certificates.
Key steps:

- Generation of certificate and openvpn configurations
- Depending on TCP/UDP usage, add exception for firewall `sudo ufw allow 1194/udp`
  - Is is better to use 443 port for OpenVPN, as it is rarely blocked (and https traffic is also - encrypted) (however there is a lot of nuances which port is better)
  - In a conditions of bad internet UDP is much more stable
- Enable ip_forwarding
- Check default gateway and other routes in server's openvpn config, which will be pushed on client-side

Certificates

```
cd ~/client-configs
./make_config.sh phonexicum
```

- Revoke user's certificate:

```
cd ~/openvpn-ca
source vars
./revoke-full phonexicum

sudo cp ~/openvpn-ca/keys/crl.pem /etc/openvpn

# Add "crl-verify crl.pem" to file /etc/openvpn/server.conf

systemctl restart openvpn@server
```

Handy commands

Start OpenVPN	`sudo systemctl start openvpn@server`
Check currently connected users	`cat /etc/openvpn/openvpn-status.log`
Check the log	`sudo journalctl -xe`
Show supported ciphers	`openvpn --show-ciphers --show-digests`
Check network settings	`sudo sysctl -p`
Reload firewall	`sudo ufw disable && sudo ufw enable`

- `persist-tun persist-key` (server side conf) - accelerate session restore (makes security worse, but not critically)
- `keepalive 3 30` (server side conf) - ping other side every 3 seconds, if there is no pings for 30 sec, decide the tunnel has failed and restart the tunnel.
- `route-nopull` - command on a client-side - prevents loading of routes
- `--reneg-sec n` (client and server side) - renegotiate data channel key after n seconds (default=3600)

Setup Openvpn in bridge mode

Here is a good manual.

- In case of virtual machines, your ***hypervisor must support promiscuous mode***, or packets intended to your clients (machines with a different mac-address unknown to your hypervisor) will be dropped.
- You ***will have to manually*** (or at least through systemd's `ExecStartPre/ExecStopPost`) set up `tap0` and `br0` interfaces (`bridge-start` and `bridge-stop` scripts can be easily found in the internet)
  Add to `/lib/systemd/system/openvpn@.service` strings for interfaces set up / tear down:
  - `ExecStartPre=/etc/openvpn/bridge/bridge-start.sh`
  - `ExecStopPost=/etc/openvpn/bridge/bridge-stop.sh`

  It is better to disable gateway change in your scripts and it is better to disable bridge's mac-address set.
- At your OpenVPN config comment out `server 10.8.0.0 255.255.255.0` and use instead smth like `server-bridge 192.168.1.1 255.255.255.0 192.168.1.200 192.168.1.250` (`192.168.1.1` is your network's gateway)

Linux may have problems with getting dns setting from OpenVPN, it can be patched using *resolvconf* package:

- `sudo apt-get install resolvconf`
- Uncomment at your client's VPN config lines:
  - `script-security 2`
  - `up /etc/openvpn/update-resolv-conf`
  - `down /etc/openvpn/update-resolv-conf`

- Now choose one of this options:
  - comment out `user nobody` and `group nogpoup` in your user's VPN config (this will make your security worse (if OpenVPN will be hacked by smbd)) (otherwise)
  - OpenVPN teardown (setup will be Okey) will fail (because of nobody privileges) and you will have to manually execute every time command: `sudo resolvconf -d tap0.openvpn`

Expose LAN's to clients

- To expose server's LAN to clients it is enough to add rule on server's config: `push "route x.y.z.0 255.255.255.0"`
- To expose client's LAN to other clients you must:
  In case VPN works in bridge mode (`tap` and `server-bridge`)
  - it is enough to add the default route on your gateway to the client's ip addr (which is may be the other device (e.g. cisco))

- if you have several openvpn servers on the same machine - add appropriate route on server-machine

Control openvpn client's access by IP with *duplicate-cn* enabled (tun mode)

- `/etc/openvpn/server.conf` - add some custom script execution
- `/etc/openvpn/ifconfig.set.sh` - allocate IP address
- `/etc/openvpn/ifconfig.unset.sh` - free IP address
- Add some access control with iptables

Linux traffic management (ip route, iptables, ...)

- manual white-IP setup (until reboot):

- change routing table (until reboot):

```
ip route add 10.8.2.0/24 via 10.0.0.1
ip route add 192.168.1.0/24 dev eth0 metric 50
ip route del 0/0 # route del default
ip route add default via 192.168.1.254
```

- permanent ip / routing setup:

  Using `vim` `/etc/network/interfaces`

  Using YAML configuration: `vim` `/etc/netplan/01-netcfg.yaml`

- setup dns servers for linux with NetworkManager (e.g. by default ubuntu-server has only networking service)

  `echo -e "\nnameserver 192.168.1.103 \nnameserver 8.8.8.8" >>/etc/resolv.conf`

- enable ip-forwarding
  - until reboot:
    - `echo 1 > /proc/sys/net/ipv4/ip_forward` OR
    - `sysctl -w net.ipv4.ip_forward=1`
  - permanent:
    - `grep forward /etc/sysctl.conf` for `net.ipv4.ip_forward = 1`

- `ip route get` (stackoverflow answer)
  - `ip route get 8.8.8.8`
  - `ip route get 8.8.8.8 mark 0x20` - check the route of marked packets to 8.8.8.8
  - `ip route get 8.8.8.8 from 192.168.0.200 iif eth1` - check the route of forwarded packets from 192.168.0.200 host received through eth1 interface
  - `ip route get 8.8.8.8 from 192.168.0.100 iif eth1 mark 0x30`

- *obtain several/multiple ip-addresses via dhcp* (not really good solution)

iptables

Brilliant article about iptables (RU):
25 iptable-examples
iptables-essentials - common firewall rules and commands

- save and restore iptables rules (not automatic)
  `iptables-save >/etc/iptables.rules` (by default iptables-save stores rules at /etc/iptables.rules)
  `iptables-restore </etc/iptables.rules`
  For *automatic* iptables rules setup add into `/etc/rc.local` line `iptables-restore </etc/iptables.rules` and make it executable: `chmod u+x /etc/rc.local`
- iptables masquerade

- port redirect:

```
iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT --to-port 8069
```

- port-forwarding:

  until reboot:

```
iptables -t nat -A PREROUTING -m tcp --dst 1.2.3.4 -p tcp --dport 9885:9889 -j DNAT --to-destination 10.0.0.3
iptables -t nat -A PREROUTING -m udp --dst 1.2.3.4 -p udp --dport 9885:9889 -j DNAT --to-destination 10.0.0.3
iptables -t nat -A OUTPUT -m tcp --dst 1.2.3.4 -p tcp --dport 9885:9889 -j DNAT --to-destination 10.0.0.3
iptables -t nat -A OUTPUT -m udp --dst 1.2.3.4 -p udp --dport 9885:9889 -j DNAT --to-destination 10.0.0.3
iptables -t nat -A POSTROUTING -p tcp --dst 10.0.0.3 --dport 9885:9889 -j MASQUERADE
iptables -t nat -A POSTROUTING -p udp --dst 10.0.0.3 --dport 9885:9889 -j MASQUERADE
```

  permanent (if ufw firewall is enabled):

```
# START PORT FORWARDING RULES
# NAT table rules
*nat
:PREROUTING ACCEPT [0:0]
:POSTROUTING ACCEPT [0:0]
# Forward packets to another location
-A PREROUTING -m tcp --dst 1.2.3.4 -p tcp --dport 9885:9889 -j DNAT --to-destination 10.0.0.3
-A PREROUTING -m udp --dst 1.2.3.4 -p udp --dport 9885:9889 -j DNAT --to-destination 10.0.0.3
# Insert correct source ip for forwarded packets
-A POSTROUTING -p tcp --dst 10.0.0.3 --dport 9885:9889 -j MASQUERADE
-A POSTROUTING -p udp --dst 10.0.0.3 --dport 9885:9889 -j MASQUERADE
COMMIT
# END PORT FORWARDING RULES
```

- windows masquerade
  - You already have 1st interface with subnet 172.16.0.0/16 you want to share.
  - You have 2nd interface (e.g openvpn tap) you wish to grant access to 172.16.0.0/16

  1. *ip-forwarding NOT needed*
  2. Open 1st interface properties and inable ICS (Доступ -> Разрешить другим пользователям сети ...).
  3. Check ipv4 settings for 2nd interface (set it to static ip or dynamic according to your needs (it happens to be *static* after enabling ICS, because windows thinks of itself as a router))

  For ms-servers exists more flexible settings: netsh routing IP NAT context commands))

- route change

```
route print
route add <destination_network> MASK <subnet_mask> <gateway_ip>
route delete <destination_network>
```

- port-forwarding

```
netsh interface portproxy add v4tov4 listenport=3340 listenaddress=10.10.1.110 connectport=3389 connectaddress=10.
netsh interface portproxy show all
netsh interface portproxy delete v4tov4 listenport=3340 listenaddress=10.10.1.110
netsh interface portproxy reset # полна очистка
```

- enable IP forwarding:

Set up transparent DNS

This DNS will resolv known names from `/etc/hosts` and question unknown's to customized DNS server (e.g. 8.8.8.8)

- `sudo apt-get install dnsmasq` - everything works from the box, BUT
- at `/etc/dnsmasq.conf` close internet interface: `except-interface=enp0s4`
- specify our internal DNS names at `/etc/hosts`: `10.0.0.1 phonexicum phonexicum.ct`

Transparent socks proxification

- Proper iptables transparent redirection:

```
##### TCP #####
iptables -t nat -A PREROUTING -p tcp -d 10.0.0.0/8 -j REDIRECT --to-ports 8081

##### UDP #####
iptables -t mangle -A PREROUTING -p udp -d 10.0.0.0/8 -j TPROXY --tproxy-mark 0x1/0x1 --on-port 8082 --on-ip 127
ip rule add fwmark 0x01/0x01 table 100
ip route add local 10.0.0.0/8 dev lo table 100
```

- TCP proxification:
  3proxy - supports transparent TCP proxying, proxy chaining and access control (by IPs, users, ...)
  3proxy proper configuration (configuration may contain more proxying options and instances):

- UDP proxification (socks5):
  socks5 udp works like this (therefore it requires DISABLED firewall):
  - client: hey server I need to send some udp traffice
  - server: send it to this random udp port: 49637
  - client sends udp data to 49637

  redsocks (`apt-get install redsocks`) - works perfectly well
  redsocks proper configuration:

Windows RDP managing

- Enable RDP: `reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections /t REG_DWORD /d 0 /f`

- How to logout smbd:
  `session query` -> `logoff <id>`

Windows administration

- All Windows Update configurations in GPO

Information Security

Information Security
phonexicum @ yandex.ru

phonexicum
phonexicum

I created this site in a burst of information security studying to organize my mind and create some kind of cheatsheet.