EINE
Easy Internet vpN Extender
Large-scale plug&play x86 network appliance deployment over Internet

Olivier COCHARD-LABBÉ
olivier.cochard@orange.com
Agenda

- Orange in one slide
- Our needs
  - Automation tools
  - Less expensive solution for our small offices
- Our Solution
  - D.I.Y
  - Deployment steps
  - Hardware selection
  - Project current status
- Build an EINE infrastructure from scratch
Orange in one slide

- Worldwide Telco: 244M customers, 156,000 employees
  - Residential: ISP, phone, TV, etc…
  - Personal: Mobile
  - Professional (Orange Business Services): ISP, VPN, VoIP, Cloud, etc…
    - International & Backbone Networks Factory (1,700 employees)
    - 40 submarine cables (6 cable ships)
    - 2.5 TB/s worldwide IP network
    - 700,000 managed devices
    - 220 countries
      - Internal network
        - Experimental features allowed: Let’s put FreeBSD everywhere!

- Open Source Orange
  - Nov 2009: Orange started to provide source code of their triple play DSL box
  - [http://opensource.orange.com](http://opensource.orange.com)
Our needs: Network automation tools

• System admin: “I’ve just deployed 200 servers this morning, now I’m waiting for your 20 firewalls and 4 load-balancers, can you do it this afternoon?”

• Network admin: “OMG… can you wait a little month?”
Our needs: Network automation tools

- We are waiting for automation tools from Network Vendors since too many years
  - In 2015 appliance vendors solution (like NETCONF) are still not production ready for large-scale and heterogeneous environment
- Why not re-using well-known IT tools for network?
  - Because IT tools are for x86 servers
- Then, let’s use x86 appliance for network too!
  - Thanks to the SDN trend for introducing x86 world into network engineers mind
Our needs: Less expensive solution for small offices

- We have 278 sites worldwide with the following size distribution:
Our solution

DO IT YOURSELF
Our solution: Do It Yourself

- Build our own solution corresponding to OUR needs
  ⇒ and not to some network manufacturer roadmap
  ⇒ only OpenSource: No known backdoors

- Simplify management of the overall solution:
  - Plug & play appliance
  - Centralized management of all devices
  - WebGUI because next generation of engineers are too stupid for using command line

- Link cost reduction:
  - Replacing expensive dedicated link by cheaper local Internet Access
Our solution: Plug&Play Internet VPN

- Plug&Play VPN-Wifi-routers deployment for Office connectivity over low-cost local ISP
Our solution: Scalable and easy to manage

Manager
- WebGUI
- Centralized management of VPN gateways & VPN routers
- Certificate Authority
- Configuration versioning

VPN Gateway
- OpenVPN servers
- Radius proxy
- Dynamic routing (OSPF/RIP/ISIS)

VPN-Wifi-Router
- OpenVPN client
- Firewall
- Wifi Access Point
- Dynamic routing (OSPF/RIP/ISIS)

ONE FIRMWARE TO RULE THEM ALL

BSDCan 2015
Our solution: Deployment steps

1. Boxes are sent to offices from the manufacturer
   - Default factory configuration
Our solution: Deployment steps

2. Local user (no FOIS needed) plugs & power the box
   – Still no box configuration needed
Our solution: Deployment steps

3. VPN-Wifi-router:
   1. Get IP address & gateway using DHCP
   2. Get date/time using NTP (certificates usage!)
   3. Open Tunnel to a VPN “enrollment” Gateway

![Diagram of network setup]

- **VPN Router**
- **ISP Ethernet Modem/bridge**
- **Internet**
- **Encrypted tunnel**
- **Corporate Intranet**
- **VPN Enrollment Gateway** (block ALL traffic coming from the tunnel)
Our solution: Deployment steps

4. Administrator logon to the manager WebGUI
   - They select new client to enroll

<table>
<thead>
<tr>
<th>VPN-Routers</th>
<th>Gateways</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
<td><strong>geoIP</strong></td>
</tr>
<tr>
<td>PNEP1</td>
<td>Nepal</td>
</tr>
<tr>
<td>PUZB1</td>
<td>Uzbekistan</td>
</tr>
<tr>
<td>PNIG1</td>
<td>Niger</td>
</tr>
<tr>
<td>NONE</td>
<td>Sydney</td>
</tr>
<tr>
<td>NONE</td>
<td>Singapore</td>
</tr>
</tbody>
</table>
Our solution: Deployment steps

4. Administrator manually enroll device
   - Calling one-site personal for confirming router ID
   - This action pushes specific site configuration (certificates, hostname, addressing, etc…) to the device

<table>
<thead>
<tr>
<th>Enrolling VPN-Router</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Role:</td>
<td>VPN router and Wifi Access Point</td>
</tr>
<tr>
<td>Hostname:</td>
<td>Sydney</td>
</tr>
<tr>
<td>Loopback address:</td>
<td>10.1.1.1</td>
</tr>
<tr>
<td>Wireless subnet:</td>
<td>10.10.1.0/25</td>
</tr>
<tr>
<td>LAN subnet:</td>
<td>10.10.1.128/25</td>
</tr>
</tbody>
</table>

Current roles:
- VPN gateway
- VPN Wifi Router

Other roles planned:
- Serial Terminal Server
- Captive portal
x86 appliances

SOFTWARE SELECTION
Software selection

- Operating system: FreeBSD (nanoBSD)
  1. We target network administrator and not system administrator
  2. It’s the only OS I’m confident in
  3. Branch used: head
- Configuration management & deployment: Ansible
  - « just» python as dependency
  - I was able to use 2 days after discovering it (I’m not a sysadim)
- VPN: OpenVPN
  - IPSec is a filtering decision and not a routing decision
  - Need to use GRE/GIF tunnels for using routing protocol over it
- Routing software: Bird
  - Because… wow!
x86 appliances

HARDWARE SELECTION
VPN-Wifi-Router: PC Engines APU (1\textsuperscript{st} generation)

- x86 64bits dual cores at 1Ghz
- 2 or 4GB of RAM
- 3 Gigabit NIC (RTL8111E)
- 16GB SSD
- Wireless 802.11a/b/g/n
- Total price: 150€ (2G) / 170€ (4G)
PC Engines APU (1\textsuperscript{st} gen): Network performance

<table>
<thead>
<tr>
<th>mode</th>
<th>pps</th>
<th>Estimated Ethernet IMIX throughput</th>
</tr>
</thead>
<tbody>
<tr>
<td>routing</td>
<td>154,171</td>
<td>437 Mb/s</td>
</tr>
<tr>
<td>ipfw impact</td>
<td>114,152</td>
<td>324 Mb/s</td>
</tr>
<tr>
<td>pf impact</td>
<td>88,169</td>
<td>250 Mb/s</td>
</tr>
</tbody>
</table>
VPN gateway: Supermicro 5018A-FTN4

- 8 cores Intel Atom C2758 at 2.4GHz (support AES-NI)
- 4 Gigabits NIC
- 1U Network appliance size
- IPMI (serial over LAN) support
- Power consumption: Low
- + 8 GB RAM
- + Flash disk (4GB is enough)
- Total Price: 870€
8 cores or more: Tuning needed

Impact of queues number regarding forwarding/ipfw/pf rate with a 8 cores Intel Atom C2758 running FreeBSD 10-STABLE r262743

Note: fastforwarding enabled, 2 firewall rules only, traffic load at 1.48 Mpps

- Fastforwarding: hw.igb.num_queues=4
- Ipfw-statefull: hw.igb.num_queues=ncpu(default)=8
- Pf-statefull: hw.igb.num_queues=4

BSDCan 2015
Hardware Appliance for Manager

- Manager needs only to:
  - Will host small WebGUI
  - Store text file
  - Send SSH commands
- A simple VM or same hardware as VPN-Router is enough
Technical annex

PROJECT CURRENT STATUS
# Project status

<table>
<thead>
<tr>
<th>Tasks</th>
<th>advancement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Define Target &amp; minimum features</td>
<td>Done</td>
</tr>
<tr>
<td>Selecting &amp; buying PoC hardware</td>
<td>Done</td>
</tr>
<tr>
<td>Selecting &amp; generating OS firmware</td>
<td>Done</td>
</tr>
<tr>
<td>Cisco “Easy VPN” like with GRE over IPSec + Certificate Management</td>
<td>Replaced by OpenVPN</td>
</tr>
<tr>
<td>Wifi EAP-TLS authentication including RADIUS proxy</td>
<td>Done</td>
</tr>
<tr>
<td>Writing helper scripts for configuration management &amp; deployment tool</td>
<td>Done: Ansible, writing gateway and client management scripts</td>
</tr>
<tr>
<td>PoC Started</td>
<td>On going: 6 routers deployed</td>
</tr>
<tr>
<td>Manager WebGUI</td>
<td>On going</td>
</tr>
<tr>
<td>Writing “Best latency check” patch to OpenVPN</td>
<td>Not started</td>
</tr>
<tr>
<td>Approval for production deployment</td>
<td>On going</td>
</tr>
</tbody>
</table>

BSDCan 2015
Virtual lab example

HOW TO BUILD AN EINE INFRASTRUCTURE FROM SCRATCH?
Virtual Infrastructure to build

Manager

VPN Gateways

OSPF router
DHCP server

Corporate

Internet router
DHCP server

Internet

VPN-Routers

BSDCaN 2015
Running a full network virtual lab… in one command

- Download a Demo EINE firmware (RAW image disk) that already includes private keys archive
  
  https://sourceforge.net/projects/bsdrp/files/BSD_Router_Project/EINE/

- Or build an image from scratch:

  ```
  svn lite co https://github.com/ocochard/BSDRP/trunk BSDRP
  cd BSDRP
  ./make -p EINE
  ```

- From a FreeBSD 10.1 with this shell script:
  
  ```
  - BSDRP-lab-bhyve.sh -i EINE-0.9-full-amd64-vga.img.xz -n 9
  ```

- From a MS Windows with VirtualBox with this PowerShell script:
  
  ```
  - BSDRP-lab-vbox.ps1
  ```

- Linux users have their VirtualBox lab-script shell too!
Running a full network virtual lab

> BSDRP-lab-bhyve.sh -i EINE-0.1-full-amd64-serial.img -n 9

BSD Router Project (http://bsdrp.net) - bhyve full-meshed lab script
Setting-up a virtual environment with 9 VM(s):
  - Working directory: /tmp/BSDRP
  - Each VM have 1 core(s) and 256M RAM
  - 0 LAN(s) between all VM
  - Full mesh Ethernet links between each VM
VM 1 have the following NIC:
  - vtnet0 connected to VM 2.
  - vtnet1 connected to VM 3.
  - vtnet2 connected to VM 4.
  - vtnet3 connected to VM 5.
  - vtnet4 connected to VM 6.
  - vtnet5 connected to VM 7.
  - vtnet6 connected to VM 8.
  - vtnet7 connected to VM 9.
VM 2 have the following NIC:
  - vtnet0 connected to VM 1.
  - vtnet1 connected to VM 3.
  - vtnet2 connected to VM 4.
  - vtnet3 connected to VM 5.
  - vtnet4 connected to VM 6.
  - vtnet5 connected to VM 7.
  - vtnet6 connected to VM 8.
  - vtnet7 connected to VM 9.
(...)
Virtual lab: Full mesh connected VMs
Virtual lab: Interfaces name and hostname

- VM1
- VM2
- VM3
- VM4
- VM5
- VM6
- VM7
- VM8
- VM9

Manager
Internal OSPF router
VPN Gateways (OSPF router on internal side)
Internet router
VPN-Routers
Remote happy users

Networks:
- Corporate
- Internet
- vtnet0
- vtnet1
- vtnet2
- vtnet3
- vtnet4
- vtnet5
- vtnet6
- vtnet7
- vtnet8
Network Interfaces naming

- FreeBSD use drivers name for network interface
  - PC Engines APU has re0, re1 and re2 network interfaces
  - Supermicro has igb0, igb1, igb2 and igb3 network interfaces
- For simplifying the management, interface on VPN gateways/routers are renamed:
  - Internet facing interface are named “net0”
    re0, igb0, vmx0, vtnet0, em0 => net0
  - Internal interface are named “net1”
    re1, igb1, vmx1, vtnet1, em1 => net1
Manager role set-up

On the VM to be configured as “manager”:

Usage: role manager IP/SUBNET DEFAULT-GATEWAY INTERNAL-DNS-LIST INTERNAL-DOMAIN-LIST private-keys-archive

```
role manager 10.0.12.1/24 10.0.12.2 "10.0.12.2 10.0.23.2"
eine.bsdrp.net /root/DEMO.private.keys.tgz
```

This command will:

1. Disable openVPN client
2. Generate a full ansible hierarchy in /usr/local/etc/ansible
3. Extract private keys from the archive
Gateways role set-up

On VPN gateways:

Usage: `role gateway IP/SUBNET DEFAULT-GATEWAY`

```
sudo ifconfig net0 name vtinet0
sudo ifconfig vtinet3 name net0
sudo sysrc -x ifconfig_vtinet0_name
sudo sysrc ifconfig_vtinet3_name="net0"
```

VM3: `role gateway 10.0.23.3/24 10.0.23.2`
VM4: `role gateway 10.0.24.4/24 10.0.24.2`

This command will:

1. Disable openVPN client
2. Configure IP address on internal NIC
3. Start bird (routing protocol on internal NIC)
Others VMs set-up

Other lab specific VMs (routers, desktops, VPN routers) are configured with:

role vmX (with X the VM number)
Adding a VPN gateway

```bash
~> gateway create emea -i 10.0.23.3/24 -e 2.2.35.3/24 -l 10.254.254.3 -u 10.1.3.0/24 -r 10.0.3.0/24 -d 2.2.35.5
Creating gateway emea.
Checking user input... Done
Testing if online... OK
Checking if not already existing in /etc/hosts... No
Updating /etc/hosts with volatile IP... Done
Adding VPN-Gateway to Ansible inventory... Done
Generating site certificate... Done
Generating Ansible host variable file... Done
Downloading gateway SSH keys... Done
Uploading configuration... Done
Deleting internal IP entry from /etc/hosts... Done
Adding loopback IP entry into /etc/hosts... Done
Gateway correctly added
Don't forget to save configuration
```
Listing unenrolled VPN-Wifi-Routers

~> **vpn-wifi-router list -u**

<table>
<thead>
<tr>
<th>Common Name</th>
<th>Real IP</th>
<th>GeoIP</th>
<th>Virtual IP</th>
<th>Rcvd</th>
<th>Sent</th>
<th>Connected Since</th>
<th>Gateway</th>
</tr>
</thead>
<tbody>
<tr>
<td>unregistered</td>
<td>203.57.57.8</td>
<td>Australia</td>
<td><strong>10.1.3.5</strong></td>
<td>11018</td>
<td>10376</td>
<td>Sun Sep 14 06:36:49 2014</td>
<td>emea1</td>
</tr>
<tr>
<td>Unregistered</td>
<td>202.56.56.6</td>
<td>Singapore</td>
<td>10.1.3.4</td>
<td>11156</td>
<td>10514</td>
<td>Sun Sep 14 06:36:32 2014</td>
<td>emea1</td>
</tr>
</tbody>
</table>
Enrolling a VPN-Wifi-router

```
~> vpn-wifi-router create sydney -e 10.7.1.0/24 -w 10.7.2.0/24 -l 3.3.3.7 -u 10.1.3.5
-i vtnet7 -x vtnet4

Creating VPN-Wifi-router sydney.
Checking user input... Done
Testing if online... OK
Checking if not already existing in /etc/hosts... No
Updating /etc/hosts with volatile IP... Done
Adding VPN-wifi-router to Ansible inventory... Done
Generating site certificate... Done
Generating Ansible host variable file... Done
Generating OpenVPN CCD file... Done
Downloading VPN-Wifi-router' SSH keys... Done
Uploading OpenVPN CCD file to all gateways... Done
Uploading registered configuration to VPN-Wifi-router... Done
Asking VPN-Wifi-router to reboot in 5 seconds... Done
Deleting unregistered VPN-Wifi-router s IP entry from /etc/hosts... Done
Adding registered IP router entry into /etc/hosts... Done
Client correctly registered, don't forget to save configuration
```
Deleting a client

~> **vpn-wifi-router delete sydney**

Deleting VPN-Wifi-router sydney.
Checking old entry in /etc/hosts... Yes
  Checking if it's online... yes
    factory-reset the VPN-Wifi-router... Done
    Asking VPN-Wifi-router to reboot in 5 seconds... Done
    Deleting old entry... Done
Deleting entry in Ansible inventory... Done
Revoking and deleting certificate... Done
  Uploading new CRL to all gateway... Done
Checking if existing host variable file... Found
  Cleaning Ansible host variable file... Done
Checking if existing CCD file... Found
  Removing CCD file... Done
    Cleaning CCD file on all gateways... OK
Checking VPN-Wifi-router SSH key in know_hosts file... Found
  Delete VPN-Wifi-router SSH key... OK
Client deleted, don't forget to save manager configuration
Administrative task: It’s just Ansible!

- Displaying version of ALL ansible managed devices
  ```
  ansible all -a "cat /etc/version"
  ```
- List host impacted by the proposed change
  ```
  ansible-playbook vpn_wifi_routers.yml --list-hosts
  ```
- Synchronize to only one host:
  ```
  ansible-playbook vpn_wifi_routers.yml -l sydney
  ```
- Synchronize only “interface” task changes to all VPN-gateways
  ```
  ansible-playbook gateways.yml --tags interface
  ```
Questions ?