Anyfi Networks | Carrier Wi-Fi System

ADVANCED SDWN CORE

GETTING STARTED GUIDE

User Experience
Architecture Overview
Installation
Basic Configuration
IEEE 802.1X with EAP-PEAP
IEEE 802.1X with EAP-SIM
Central Breakout with Optimizer
TR-069 Integration
Splunk Integration

Anyfi Networks

Västergatan 31 B
21121 Malmö
Sweden
info@anyfinetworks.com

RELEASE DATE: 27th of April 2015

DOCUMENT REVISION: R1F v03

RELEASED WITH: CARRIER WI-FI SYSTEM R1F

Contents

Preface

This document details how to install, configure and verify an advanced example SDWN core.

This example combines our Controller, Gateway and Optimizer products with a basic management server and WLAN gateway. The resulting core is capable of running our SIMPLE and HOTSPOT solutions, in the former case with central breakout of Internet-bound SDWN data plane traffic and in latter with IEEE 802.1X authentication (EAP-PEAP and EAP-SIM).

***NOTE** This example SDWN core is part of the Community Edition of our Carrier Wi-Fi System. Community Edition is unsupported and restricted to a maximum of 100 radios and services, but can be used for both commercial and non-commercial purposes. Contact sales@anyfinetworks.com for other licensing options.*

Intended Audience

This document is intended for system and network administrators with previous experience of Command Line Interfaces (CLIs) for administrative tasks. Readers should have specific knowledge in the following areas:

- Networking and data communications
- Internet protocol (IP)

Readers lacking experience with the Vyatta Network OS are encouraged to study its online documentation.

Document Conventions

This guide contains advisory paragraphs and uses the below typographic conventions.

Advisory Paragraphs

This guide uses the following advisory paragraphs:

Warnings alert you to situations that may pose a threat to your system or subscriber's security, as in the following example:

***WARNING** Bridging unauthenticated Wi-Fi traffic to a network interface may pose a security threat to the associated network.*

Cautions alert you to situations that might affect service, as in the following example:

***CAUTION** Restarting a production system will interrupt service.*

Notes provide important information about the structure or functioning of the system:

***NOTE** The Controller is a controller in the Software-Defined Networking (SDN) sense of the word, not in the typical corporate WLAN sense.*

Typographic Conventions

This document uses the following typographic conventions:

Monospace	Examples, command-line output, and representations of configuration nodes. Also commands, keywords, and file names, when mentioned inline.
bold Monospace	Your input: something you type at a command line.
Italics	An argument or variable where you supply a value. Also concepts and principles when mentioned inline.
<key>	A key on your keyboard, such as <Enter>. Combinations of keys are joined by plus signs ("+"), as in <Ctrl>+c.
[*arg1* | *arg2*]	Enumerated options for completing a syntax. An example is [enable | disable].
num1–numN	A inclusive range of numbers. An example is 1-65535, which means 1 through 65535, inclusive.
arg1..argN	A range of enumerated values. An example is eth0..eth3, which means eth0, eth1, eth2, or eth3.
arg[*arg...*] *arg*[,*arg...*]	A value that can optionally represent a list of elements (a space-separated list in the first case and a comma-separated list in the second case)

Chapter 1: User Experience

In this chapter we describe the end-user experience that can be achieved with this advanced example SDWN core.

SIMPLE – Home Wi-Fi Anywhere

Our SIMPLE solution provides every user with seamless and secure remote access to their home Wi-Fi network. We call this a “zero sign-on” user experience because there is absolutely nothing the end-user needs to do to connect. There is no separate registration process, no software that needs to be installed on the device, not even any settings to change – it just works.

Note that each user will see their own home Wi-Fi SSID – but not each other’s. This ensures that everybody can seamlessly connect to their network, without cluttering the list of networks in the connection manager.

Advanced radio policy control allows the operator to set a minimum quality of service level that must be reached before the network is even presented to the mobile device. This helps avoid the most common user experience problem in carrier Wi-Fi deployments: that mobile devices connect too early to the network – when the radio link quality is so poor that meaningful communication on Layer 3 is not possible.

The user data plane is protected end-to-end, all the way from the mobile device to the Optimizer or the mobile user's own home gateway, by the well-known and trusted WPA2 security mechanism. Not even an attacker in complete control of the access point can eavesdrop on or modify a guest’s communication.

Seamless hand-over from CPE to CPE with session continuity on Layer 3 is fully supported.

For more information about the SIMPLE solution and its user experience please see http://www.anyfinetworks.com/solutions/simple.

HOTSPOT – Traditional Hotspots and Homespots

Our HOTSPOT solution provides a traditional hotspot or homespot user experience. In this core we have also included a basic AAA server, integrated into the WLAN gateway. This allows for IEEE 802.1X authentication with EAP-PEAP and EAP-SIM. Mobile devices with appropriate Wi-Fi settings will automatically connect and authenticate, without requiring any client software. Note however that provisioning of Wi-Fi profiles and settings to subscriber devices is outside the scope of our solutions.

The HOTSPOT solution in combination with IEEE 802.1X authentication provides end-to-end security all the way from the mobile device to the SDWN core – even against an attacker in complete control of the access point.

For more information about the HOTSPOT solution and its user experience please see http://www.anyfinetworks.com/solutions/hotspot.

Chapter 2: Architecture Overview

This example SDWN core contains a management machine, a Controller, a Gateway, an Optimizer and a WLAN gateway. The management machine and the WLAN gateway are not part of our product portfolio, serving only as examples and placeholders for third party systems.

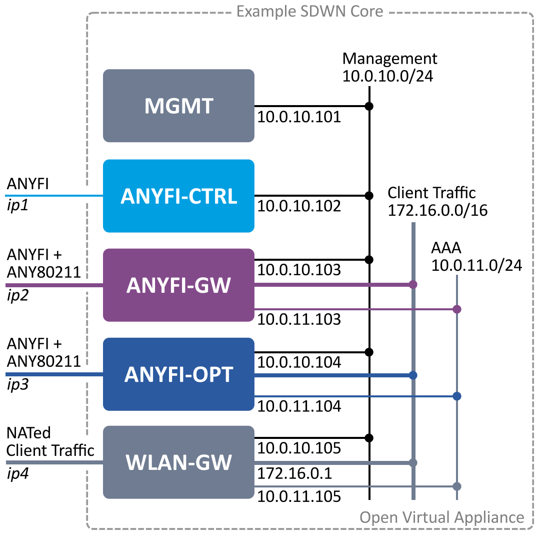


Figure 1: Network diagram of the advanced example SDWN core.

Software-Defined Wireless Networking (SDWN) is an SDN overlay architecture, with the control plane centralized in a Controller while the data plane remains distributed. Both user payload and control traffic are transported over UDP/IP, requiring no specific network integration. Access points can therefore be installed anywhere there is IP connectivity.

Note that the RADIUS traffic between the Gateway and the WLAN gateway is isolated on its own internal network, inaccessible from APs and even the Controller. This strong protection of the authentication interface preserves the strong mutual authentication of the underlying EAP mechanism and is one of the hallmarks of an SDWN architecture.

As you can see in Figure 1 you will need four IP addresses to install and run this basic SDWN core. In this guide we assume that these are four static public IP addresses and we refer to them as `ip1`, `ip2`, `ip3` and `ip4`.

SIMPLE – Home Wi-Fi Anywhere

Fixed-line subscribers are provided seamless and secure remote access to their home Wi-Fi networks by the SIMPLE SDWN App. In this advanced example we have included an Optimizer. The Optimizer is inserted into SDWN data plane tunnels by the Controller, and will break out any Internet-bound traffic it finds in these tunnels.

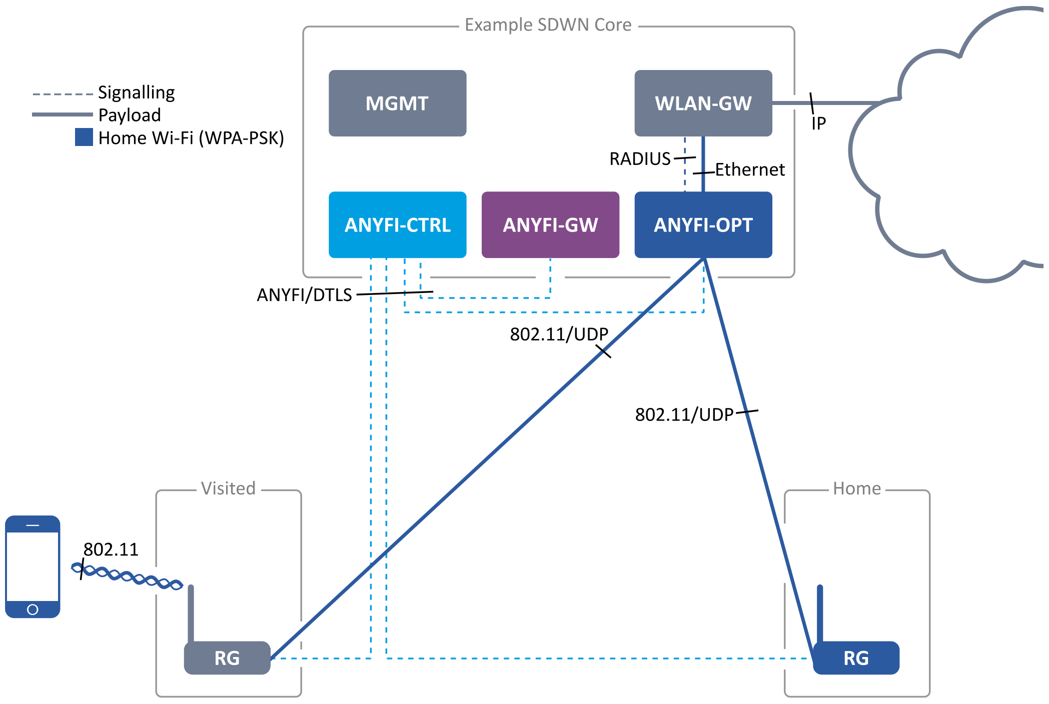


Figure 2: Architecture overview for the SIMPLE SDWN App with Optimizer.

Breakout traffic is bridged to the WLAN gateway, which provides DHCP, NAT and IP gateway functionality. In a production environment this example WLAN gateway will likely be replaced with a third party solution providing policy enforcement, traffic traceability and lawful intercept capabilities.

HOTSPOT – Traditional Hotspots and Homespots

The HOTSPOT SDWN App distributes an open or 802.1X authenticated Wi-Fi network across all (or a group of) access point radios. The user data plane is tunneled from the visited access point to the Gateway over UDP/IP, where the IEEE 802.11 protocol is terminated. All security critical packet processing, e.g. 802.1X authentication and 802.11 encryption, is thus performed within the trusted environment of the SDWN core.

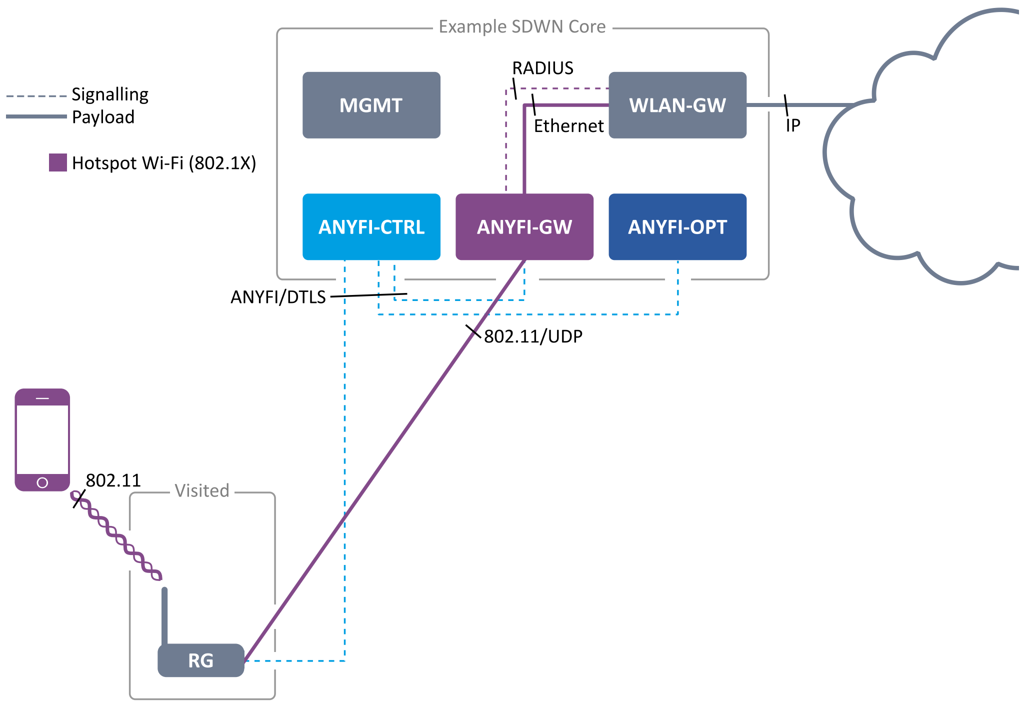


Figure 3: Architecture overview for the HOTSPOT SDWN App.

Client traffic is then bridged to the WLAN gateway, which provides DHCP, NAT and IP gateway functionality. In a production environment this example WLAN gateway will likely be replaced with a third party solution providing policy enforcement, traffic traceability and lawful intercept capabilities.

***NOTE** SDWN data plane tunnels carry the raw encrypted wireless protocol frames inside UDP/IP datagrams. In this system the air interface is IEEE 802.11 and the inner tunnel protocol is thus IEEE 802.11 Mac Protocol Data Units (MPDUs).*

Chapter 3: Installation

In this chapter we describe how to install the avanced example SDWN core on VMware ESXi or Oracle VM VirtualBox.

System Requirements

Your host system should meet the following requirements:

- 64-bit x86 CPU with Intel VT-x or AMD-V instruction set
- VMware ESXi 5.x hypervisor or host OS supported by Oracle VM VirtualBox:
    - Linux 2.6 or 3.x
    - Windows 64-bit Vista, Server 2008, 7, 8, Server 2012 or Server 2013
    - Mac OS X 10.6, 10.7, 10.8 or 10.9
- Minimum of 4 GB RAM available for VMs
- Minimum of 20 GB free disk space for VM disk images
- At least one Ethernet network interface

CPU support for the Intel AES-NI instruction set is recommended but not required.

Automated Provisioning on VMware ESXi

In this section we explain how to use the self-extracting provisioning script to provision an advanced example core on a host system running the VMware ESXi 5.x hypervisor.

Preparations

If you have not yet installed the VMware ESXi hypervisor on the host system you will of course need to do so before we begin. An evaluation version of ESXi 5.5, also known as vSphere Hypervisor 5.5, is freely available and can be downloaded from VMware's website.

Now start with downloading the self-extracting provisioning script containing the advanced example SDWN core.

You will need to run the self-extracting provisioning script from the ESXi Shell, so enable ESXi Shell access.

The self-extracting provisioning script uses SSH to configure VMs. The ESXi firewall must therefore be configured to allow outbound SSH connections.

Configure ESXi firewall to allow outbuound SSH

```
# esxcli network firewall ruleset set \
        --ruleset-id sshClient --enabled true
```

The self-extracting provisioning script also needs to temporarily store VMDKs and other large files during the installation process. The ESXi `/tmp` file system may have insufficient capacity to hold these files. We therefore recommend that you create a temporary directory under a mount point with at least 2 GB of spare capacity, and set the `TMPDIR` environment variable to point to this directory.

Create a directory for temporary storage

```
# mkdir /vmfs/volumes/datastore/tmp
```

Set the TMPDIR environment variable

```
# export TMPDIR=/vmfs/volumes/datastore/tmp
```

You will also need to specify vSwitch port groups for the external interfaces of the example core. It is possible to specify up to three different port groups; a management plane port group, a control plane port group and a data plane port group. The most common configuration is however to assign both control and data to one physical Ethernet interface (where *ip1-4* is available) and management to another (where the management VM will receive an IP address by DHCP). You should configure these port groups e.g. through the vSphere Client before running the provisioning script.

Finally, don't forget to backup your data and ESXi configuration before proceeding.

Instructions

Simply run the self-extracting provisioning script in the ESXi Shell.

Provision an advanced example SDWN core on VMware ESXi

```
# sh provision-anyfi-sdwn-core-r1f.sh advanced vmware_esxi \
        --datastore=datastore --with-mgmt-pg=lan \
        --with-ctrl-pg=wan --with-data-pg=wan
```

Note that the script can take up to 30 minutes to complete. Do not terminate the script during this time. Once the example core is provisioned start up the virtual machines and complete the installation as detailed below.

WARNING *Once the example SDWN core is operational client devices connected to the resulting Wi-Fi services will have unrestricted routed IP access to the data plane port group.*

Automated Provisioning on VirtualBox

In this section we explain how to use the self-extracting provisioning script to provision an advanced example core on a Linux host system with Oracle VM VirtualBox hypervisor.

Preparations

First of all download the self-extracting provisioning script containing the advanced example SDWN core.

You will need a recent version of Oracle VM VirtualBox. This example core has been verified with version 4.3.10.

Instructions

Simply run the self-extracting provisioning script in the shell.

Provision an advanced example SDWN core on Oracle VM VirtualBox

```
# sh provision-anyfi-sdwn-core-r1f.sh advanced virtualbox \
        --with-ctrl-if=eth0 --with-data-if=eth0
```

Note that the script can take up to 30 minutes to complete. Do not terminate the script during this time. Once the example core is provisioned start up the virtual machines and complete the installation as detailed below.

WARNING *Once the example SDWN core is operational client devices connected to the resulting Wi-Fi services will have unrestricted routed IP access to the data plane network interface.*

Manual Provisioning on VirtualBox

For Mac OS or Windows hosts we recommend manually installing the advanced example SDWN core using the Open Virtual Appliance (OVA) file.

Preparations

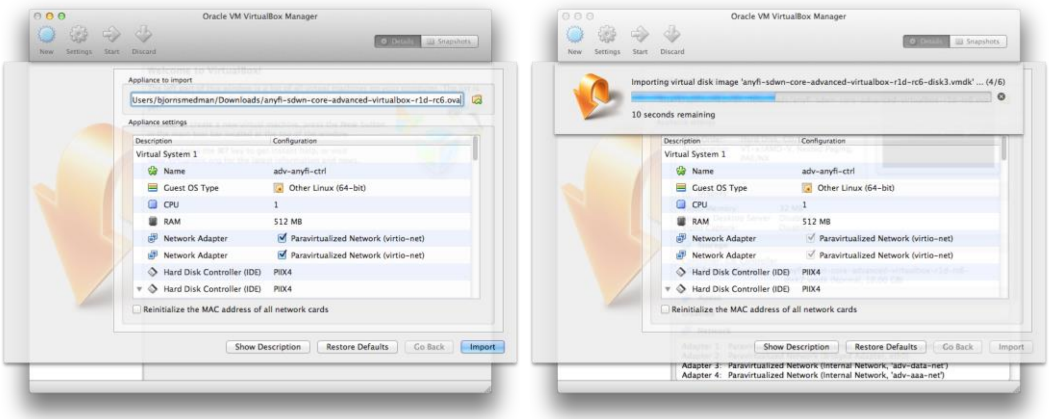
First of all download the Open Virtual Appliance (OVA) containing the advanced example SDWN core.

To manually install the Open Virtual Appliance (OVA) you will need a recent version of Oracle VM VirtualBox. This example core has been verified with version 4.3.10.

Instructions

We now go through the installation of the Open Virtual Appliance on Oracle VM VirtualBox step-by-step.

Import the Open Virtual Appliance into VirtualBox



Appliance to import

Users/bjornsmedman/Downloads/anyfi-sdwn-core-advanced-virtualbox-r1d-rc6.ova

Appliance settings

Description	Configuration
Virtual System 1	
Name	adv-anyfi-ctrl
Guest OS Type	Other Linux (64-bit)
CPU	1
RAM	512 MB
Network Adapter	✓ Paravirtualized Network (virtio-net)
Network Adapter	✓ Paravirtualized Network (virtio-net)
Hard Disk Controller (IDE)	PIIX4
Hard Disk Controller (IDE)	PIIX4

Reinitialize the MAC address of all network cards

Importing virtual disk image 'anyfi-sdwn-core-advanced-virtualbox-r1d-rc6-disk3.vmdk' ... (4/6)

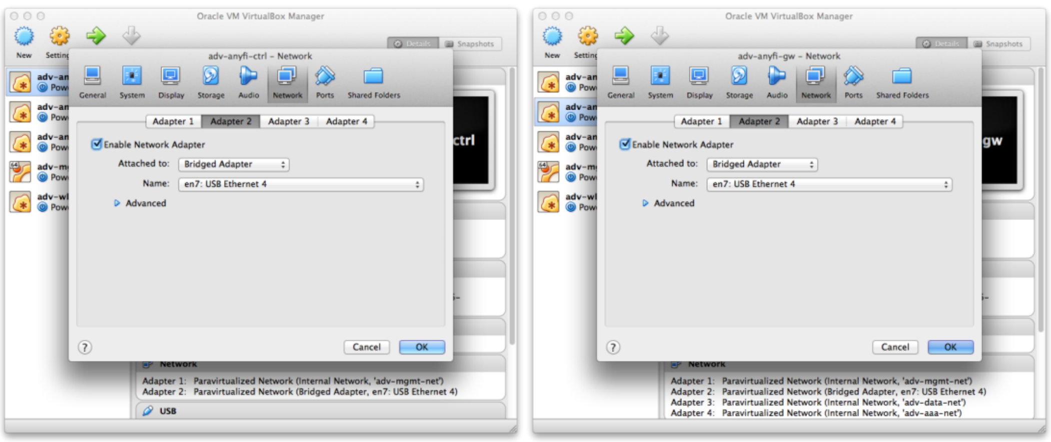
10 seconds remaining

Adapter 3: Paravirtualized Network (Internal Network, 'adv-data-net')
Adapter 4: Paravirtualized Network (Internal Network, 'adv-aaa-net')

Figure 4: Import the OVA file into VirtualBox.

Start VirtualBox and select the `File > Import Appliance...` menu option. Select the OVA file and press the `Continue` button. As you will note the OVA file contains multiple VMs, one for each network element in the Software-Defined Wireless Networking (SDWN) core. Import all the VMs into VirtualBox by pressing the `Import` button (without changing any settings).

Select External Network Interfaces



adv-anyfi-ctrl – Network

Enable Network Adapter ✓
Attached to: Bridged Adapter
Name: en7: USB Ethernet 4

Adapter 1: Paravirtualized Network (Internal Network, 'adv-mgmt-net')
Adapter 2: Paravirtualized Network (Bridged Adapter, en7: USB Ethernet 4)

adv-anyfi-gw – Network

Enable Network Adapter ✓
Attached to: Bridged Adapter
Name: en7: USB Ethernet 4

Adapter 1: Paravirtualized Network (Internal Network, 'adv-mgmt-net')
Adapter 2: Paravirtualized Network (Bridged Adapter, en7: USB Ethernet 4)
Adapter 3: Paravirtualized Network (Internal Network, 'adv-data-net')
Adapter 4: Paravirtualized Network (Internal Network, 'adv-aaa-net')

Figure 5: Select the correct external network interfaces.

Select the `adv-anyfi-ctrl` VM and press `Network`. Select `Adapter 2` and verify that it is configured to use a reasonable external network interface. We will later configure the VM to use *ip1* for this interface.

Repeat the process for the `adv-anyfi-gw`, `adv-anyfi-opt` and `adv-wlan-gw` VMs. These VMs will later be configured to use *ip2*, *ip3* and *ip4* respectively.

***WARNING** Once the example SDWN core is operational client devices connected to the resulting Wi-Fi services will have unrestricted routed IP access to the WLAN gateway's bridged adaptor.*

Start the Virtual Machines

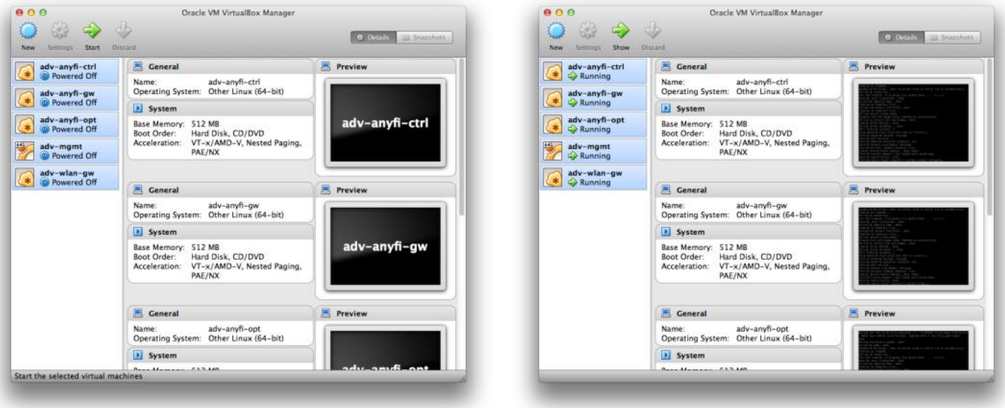


Figure 6: Start all the virtual machines by selecting them and pressing "Start".

Select all VMs and press `Start`. Make sure they all boot and then minimize their console windows. We will not be using the console to interact with the VMs.

Completing the Installation

Once all VMs have booted you should be able to log in to the management VM through Secure Shell (SSH). Under VirtualBox the management VM is reachable at localhost port 8022. With VMware ESXi the management VM is attached to the port group you specified using the `--with-mgmt-pg=` option and will request an IP address through DHCP. In both cases you should log in as the user `admin` with password `admin`.

Log in to the management VM with SSH

```
host:~$ ssh -l admin [localhost -p 8022 | x.x.x.x]
```

Once logged in complete the installation of the core by issuing the following command.

Complete the installation of the example core

```
admin@mgmt:~$ ./install-core
```

Chapter 4: Configuration

In this chapter we describe how to configure your newly installed virtualized Software-Defined Wireless Networking (SDWN) core.

Configuring the Controller

We start by connecting to the Controller with Secure SHell (SSH).

Connect to the Controller with SSH

```
admin@mgmt:~$ ssh anyfi-ctrl
vyatta@anyfi-ctrl:~$
```

The Controller is based on the Vyatta Network OS and provides a Command Line Interface (CLI) for administrative tasks.

Basic Networking

The Controller acts as a rendezvous point for all other elements in the SDWN architecture and therefore must have a fixed (and usually public) IP address. We refer to this address as `ip1`. You will later need to configure it (or the corresponding domain name) into the Gateway, Optimizer, access points and residential gateways, so make note of it.

Enter configuration mode

```
vyatta@anyfi-ctrl:~$ configure
[edit]
```

Configure basic networking

```
vyatta@anyfi-ctrl# set interfaces ethernet eth1 address ip1/xx
[edit]
vyatta@anyfi-ctrl# set system name-server x.x.x.x
[edit]
vyatta@anyfi-ctrl# set system gateway-address x.x.x.x
[edit]
```

Commit, save and exit configuration mode

```
vyatta@anyfi-ctrl# commit
vyatta@anyfi-ctrl# save
vyatta@anyfi-ctrl# exit
```

The Controller should now have basic IP connectivity and name resolution. You can verify this e.g. with the `ping` command.

SDWN Controller

Now we are ready to configure the SDWN controller.

Enter configuration mode

```
vyatta@anyfi-ctrl:~$ configure
[edit]
```

Define client, radio and service groups

```
vyatta@anyfi-ctrl# edit service anyfi controller
[edit service anyfi controller]
vyatta@anyfi-ctrl# set client-group "all"
[edit service anyfi controller]
vyatta@anyfi-ctrl# set radio-group "all"
[edit service anyfi controller]
vyatta@anyfi-ctrl# set service-group "gateways" ip-filter ip2
[edit service anyfi controller]
vyatta@anyfi-ctrl# set service-group "other" ip-filter !ip2
[edit service anyfi controller]
vyatta@anyfi-ctrl# top
[edit]
```

Create an instance of the SIMPLE app

```
vyatta@anyfi-ctrl# edit service anyfi controller app simple "simple1"
[edit service anyfi controller app simple simple1]
vyatta@anyfi-ctrl# set clients "all"
[edit service anyfi controller app simple simple1]
vyatta@anyfi-ctrl# set radios "all"
[edit service anyfi controller app simple simple1]
vyatta@anyfi-ctrl# set services "other"
[edit service anyfi controller app simple simple1]
vyatta@anyfi-ctrl# top
[edit]
```

Create an instance of the HOTSPOT app

```
vyatta@anyfi-ctrl# edit service anyfi controller app hotspot
    "hotspot1"
[edit service anyfi controller app hotspot hotspot1]
vyatta@anyfi-ctrl# set clients "all"
[edit service anyfi controller app hotspot hotspot1]
vyatta@anyfi-ctrl# set radios "all"
[edit service anyfi controller app hotspot hotspot1]
vyatta@anyfi-ctrl# set services "gateways"
[edit service anyfi controller app hotspot hotspot1]
vyatta@anyfi-ctrl# top
[edit]
```

Review changes

```
vyatta@anyfi-ctrl# show service anyfi controller
+app {
+    hotspot hotspot1 {
+        clients all
+        radios all
+        services gateways
+    }
+    simple simple1 {
+        clients all
```

	`+        radios all`
	`+        services other`
	`+    }`
	`+}`
	`+client-group all {`
	`+}`
	`+radio-group all {`
	`+}`
	`+service-group gateways {`
	`+    ip-filter ip2`
	`+}`
	`+service-group other {`
	`+    ip-filter !ip2`
	`+}`
	`[edit]`

Commit, save and exit configuration mode	`vyatta@anyfi-ctrl# commit`
	`vyatta@anyfi-ctrl# save`
	`vyatta@anyfi-ctrl# exit`

The Controller should now be configured and ready to use.

Disconnect from the Controller	`vyatta@anyfi-ctrl:~$ exit`
	`admin@mgmt:~$`

Configuring the Gateway

Now we connect to the Gateway with Secure SHell (SSH).

Connect to the Gateway with SSH	`admin@mgmt:~$ ssh anyfi-gw`
	`vyatta@anyfi-gw:~$`

The Gateway is based on the Vyatta Network OS and provides a Command Line Interface (CLI) for administrative tasks.

Basic Networking

The Gateway must be configured to use a static IP address. Otherwise it will not match the `ip-filter` previously configured into the Controller, and its virtual Wi-Fi network will not be distributed by the HOTSPOT app.

Enter configuration mode	`vyatta@anyfi-gw:~$ configure`
	`[edit]`

Configure basic networking	`vyatta@anyfi-gw# set interfaces ethernet eth1 address ip2/xx`
	`[edit]`
	`vyatta@anyfi-gw# set system name-server x.x.x.x`

```
[edit]
vyatta@anyfi-gw# set system gateway-address x.x.x.x
[edit]
```

Commit, save and exit configuration mode

```
vyatta@anyfi-gw# commit
vyatta@anyfi-gw# save
vyatta@anyfi-gw# exit
```

The Gateway should now have basic IP connectivity and name resolution. Verify that you can `ping` the controller.

SDWN Gateway

Now we are ready to configure the SDWN gateway.

Enter configuration mode

```
vyatta@anyfi-gw:~$ configure
[edit]
```

Create a Gateway instance

```
vyatta@anyfi-gw# edit service anyfi gateway "1x-gw"
[edit service anyfi gateway 1x-gw]
vyatta@anyfi-gw# set controller ip1
[edit service anyfi gateway 1x-gw]
vyatta@anyfi-gw# set bridge br0
[edit service anyfi gateway 1x-gw]
vyatta@anyfi-gw# set ssid "Hotspot 1X"
[edit service anyfi gateway 1x-gw]
vyatta@anyfi-gw# set isolation
[edit service anyfi gateway 1x-gw]
vyatta@anyfi-gw# set wpa2
[edit service anyfi gateway 1x-gw]
vyatta@anyfi-gw# set authentication eap radius-server 10.0.11.105
    secret "not-so-important"
[edit service anyfi gateway 1x-gw]
vyatta@anyfi-gw# top
[edit]
```

Review changes

```
vyatta@anyfi-gw# show service anyfi gateway
+gateway 1x-gw {
+    authentication {
+        eap {
+            radius-server 10.0.11.105 {
+                secret not-so-important
+            }
+        }
+    }
+    bridge br0
+    controller ip1
+    isolation
+    ssid "Hotspot 1X"
+    wpa2 {
```

```
+     }
+}
[edit]
```

Commit, save and exit configuration mode

```
vyatta@anyfi-gw# commit
vyatta@anyfi-gw# save
vyatta@anyfi-gw# exit
```

***NOTE** The RADIUS secret is not security critical in an SDWN architecture, since only the trusted core network has access to the AAA server.*

The newly created Gateway instance should now be registered with the Controller as an SDWN *service*. To register additional *services* (i.e. virtual Wi-Fi networks or in layman's terms "SSIDs") simply repeat the process above. Instances are entirely independent of each other and can each have their own SSID, bridge interface and security settings.

The Gateway should now be configured.

Disconnect from the Gateway

```
vyatta@anyfi-gw:~$ exit
admin@mgmt:~$
```

Configuring the Optimizer

Now we connect to the Optimizer with Secure SHell (SSH).

Connect to the Optimizer with SSH

```
admin@mgmt:~$ ssh anyfi-opt
vyatta@anyfi-gw:~$
```

The Optimizer is based on the Vyatta Network OS and provides a Command Line Interface (CLI) for administrative tasks.

Basic Networking

The Optimizer must be configured to use a static IP address.

Enter configuration mode

```
vyatta@anyfi-opt:~$ configure
[edit]
```

Configure basic networking

```
vyatta@anyfi-opt# set interfaces ethernet eth1 address ip3/xx
[edit]
vyatta@anyfi-opt# set system name-server x.x.x.x
[edit]
vyatta@anyfi-opt# set system gateway-address x.x.x.x
[edit]
```

Commit, save and exit configuration mode

```
vyatta@anyfi-opt# commit
vyatta@anyfi-opt# save
```

```
vyatta@anyfi-opt# exit
```

The Optimizer should now have basic IP connectivity and name resolution. Verify that you can `ping` the controller.

RSA Key Pair Generation

Before we can configure the Optimizer we must first generate the RSA key pair that will protect the IEEE 802.11 Temporal Key (TK) in transit between *service termination point* and Optimizer.

Generate an RSA key pair

```
vyatta@vyatta:~$ generate anyfi optimizer rsa-key-pair
    /config/auth/opt-key-pair.pem
Generating 2048 bit RSA key pair to file /config/auth/opt-key-
    pair.pem... Done.
[...]
=== OPTIMIZER KEY ===============================================

d2d90fb7480082761f80c8354e223ec3f9f728f6f3bffa08b26bcc974c847d24
```

The *optimizer key* is a cryptographic hash that uniquely identifies the key pair. It can be configured into your CPE through the Anyfi.net TR-069 vendor extension or in some cases the WEBUI of the CPE. For more information refer to your CPE manual.

SDWN Optimizer

We are now ready to configure the Optimizer.

Enter configuration mode

```
vyatta@vyatta:~$ configure
[edit]
```

Create an Optimizer instance

```
vyatta@vyatta# edit service anyfi optimizer "opt1"
[edit service anyfi optimizer opt1]
vyatta@vyatta# set controller x.x.x.x
[edit service anyfi optimizer opt1]
vyatta@vyatta# set port-range 10000-10100
[edit service anyfi optimizer opt1]
vyatta@vyatta# set rsa-key-pair file /config/auth/opt-key-pair.pem
[edit service anyfi optimizer opt1]
vyatta@vyatta# set bridge br0
[edit service anyfi optimizer opt1]
vyatta@vyatta# top
[edit]
```

Review changes

```
vyatta@vyatta# show service anyfi optimizer
+optimizer opt1 {
+    bridge br0
+    controller x.x.x.x
```

	`+     port-range 10000-10100`
	`+     rsa-key-pair {`
	`+         file /config/auth/opt-key-pair.pem`
	`+     }`
	`+}`
	`[edit]`

Commit, save and exit configuration mode	`vyatta@vyatta#` **`commit`**
	`vyatta@vyatta#` **`save`**
	`vyatta@vyatta#` **`exit`**

The Optimizer is now ready to receive IEEE 802.11 Temporal Keys (TKs) from *service termination points*. Using these encryption keys it will peek inside SDWN data plane tunnels and break out any Internet-bound traffic it finds. Breakout traffic will be processed in the Optimizer to appear as normal Wi-Fi access point traffic, and then bridged to the `br0` logical bridge that in turn is connected to the WLAN gateway.

Before the *service termination software* transmit an RSA-encrypted TK over the network it must however be configured with the appropriate *optimizer key*, essentially a cryptographic hash that uniquely identifies the RSA key pair of the Optimizer we wish to entrust with the TKs of client devices.

Use the command below to show the *optimizer key* assocated with a specific instance.

Show the optimizer key	`vyatta@anyfi-opt:~$` **`show anyfi optimizer opt1 key`**
	`d4577fe79131b980a5243a601ec6f311e2c483de660806fb7acf33dae92c2539`

The *optimizer key* can be configured into CPE using the Anyfi.net TR-069 vendor extension, or in some cases through a WEBUI or CLI. Refer to your CPE documentation for more information.

The Optimizer should now be configured.

Disconnect from the Optimizer	`vyatta@anyfi-opt:~$` **`exit`**
	`admin@mgmt:~$`

Configuring the WLAN Gateway

Now we connect to the WLAN gateway with Secure SHell (SSH).

Connect to the WLAN GW with SSH	`admin@mgmt:~$` **`ssh wlan-gw`**
	`vyatta@wlan-gw:~$`

The WLAN gateway in this example SDWN core is a virtual machine that runs the Vyatta Network OS, configured to act as an IP gateway with NAT and DHCP. The example WLAN gateway also has a simple built-in RADIUS server. This server is very basic but allows for demonstration of EAP-PEAP and EAP-SIM authentication. IEEE 802.1X user credentials can be defined through a Command Line Interface (CLI).

NOTE The WLAN gateway is expected to be replaced with a 3GPP WLAN Gateway or similar in a production deployment. This network element serves only as an example.

Basic Networking

The WLAN gateway must be configured with an IP address to use for NATed client traffic.

Enter configuration mode	`vyatta@wlan-gw:~$` **`configure`**
	`[edit]`

Configure basic networking	`vyatta@wlan-gw#` **`set interfaces ethernet eth1 address`** ***`ip4/xx`***
	`[edit]`
	`vyatta@wlan-gw#` **`set system name-server`** ***`x.x.x.x`***
	`[edit]`
	`vyatta@wlan-gw#` **`set system gateway-address`** ***`x.x.x.x`***
	`[edit]`

Commit, save and exit configuration mode	`vyatta@wlan-gw#` **`commit`**
	`vyatta@wlan-gw#` **`save`**
	`vyatta@wlan-gw#` **`exit`**

The WLAN GW should now have basic IP connectivity and name resolution. You can verify this with the `ping` command.

Gateway as RADIUS Client

The built-in AAA server is pre-configured to serve all RADIUS clients on the AAA network attached to `eth2`. For completeness we show its configuration below.

Enter configuration mode	`vyatta@wlan-gw:~$` **`configure`**
	`[edit]`

Show the RADIUS server configuration	`vyatta@wlan-gw#` **`show service radius`**
	` client gateway {`
	`     ip-filter 10.0.11.0/24`
	`     secret not-so-important`
	` }`
	` interface eth3`
	`[edit]`

Exit configuration mode	`vyatta@wlan-gw#` **`exit`**

All we need to do is therefore to define some IEEE 802.1X identities.

NOTE *The RADIUS secret is not security critical in an SDWN architecture, since only the trusted core network elements have access to the RADIUS AAA server.*

EAP-PEAP User Credentials

The built-in AAA server allows for configuration of EAP-PEAP with MS-CHAP-v2 user credentials through the CLI.

Enter configuration mode	`vyatta@wlan-gw:~$` **`configure`**
	`[edit]`

Add an EAP-PEAP authenticated user	`vyatta@wlan-gw#` **`edit service radius user local peap-mschapv2 "user1"`**
	`[edit service radius user local peap-mschapv2 user1]`
	`vyatta@wlan-gw#` **`set password "password1"`**
	`[edit service radius user local peap-mschapv2 user1]`
	`vyatta@wlan-gw#` **`top`**

Commit, save and exit configuration mode	`vyatta@wlan-gw#` **`commit`**
	`vyatta@wlan-gw#` **`save`**
	`vyatta@wlan-gw#` **`exit`**

EAP-SIM User Credentials

The built-in AAA server allows for configuration of an EAP-SIM user by entering the IMSI and three so-called SIM triplets through the command line interface.

NOTE *In order to complete the steps below you will need a SIM card reader capable of reading Kc, SRES and RAND triplets from your SIM card.*

Enter configuration mode	`vyatta@wlan-gw:~$` **`configure`**
	`[edit]`

Add an EAP-SIM authenticated user	`vyatta@wlan-gw#` **`edit service radius user local sim "240016093137478"`**
	`[edit service radius user local sim 240016093137478]`
	`vyatta@wlan-gw#` **`set triplet`**

	`"823f162bF73a801a:B1ca1de0:73FCDA5EF8494b99A80CD45E4B394FF6"`
	`[edit service radius user local sim 240016093137478]`
	`vyatta@wlan-gw#` **`set triplet "54c7be6583f111a8:4D95a274:E02C635163E5416e8DE21BA5981F0251"`**
	`[edit service radius user local sim 240016093137478]`
	`vyatta@wlan-gw#` **`set triplet "23758aaf4Ade9a9f:B844b6d3:57307411621A88caBC365CA637B8C8F7"`**
	`[edit service radius user local sim 240016093137478]`
	`vyatta@wlan-gw#` **`top`**

Commit, save and exit configuration mode	`vyatta@wlan-gw#` **`commit`**
	`vyatta@wlan-gw#` **`save`**
	`vyatta@wlan-gw#` **`exit`**

The WLAN gateway is now be configured and ready to use.

Disconnect from the WLAN GW	`vyatta@wlan-gw:~$` **`exit`**
	`admin@mgmt:~$`

This concludes the configuration of the example SDWN core.

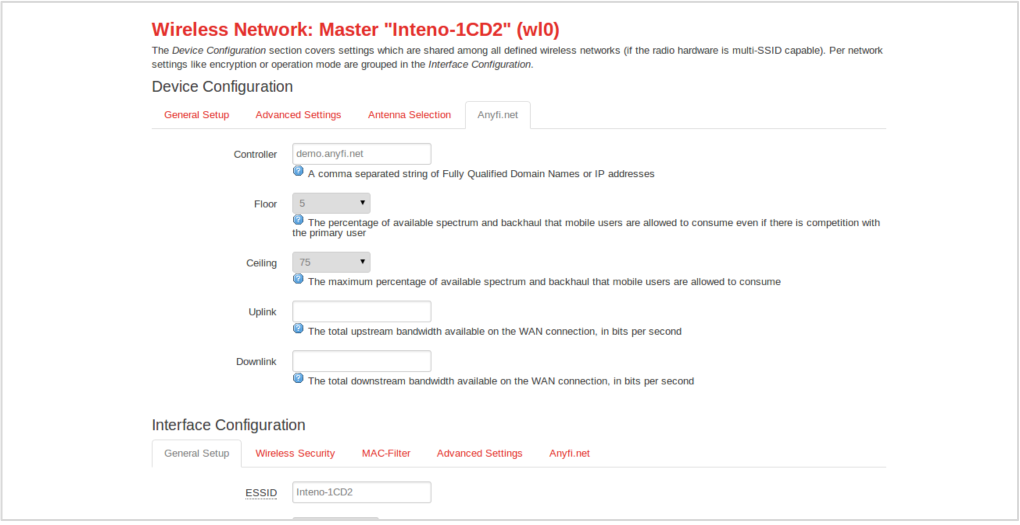
Chapter 5: Verification

In this chapter we describe how to put your newly installed and configured Software-Defined Wireless Networking (SDWN) core to good use.

Preparations

To follow the instructions below you will need two *Anyfi.net* *enabled* residential gateways. We will use Inteno VG50, a dual-WAN Ethernet and ADSL2+ residential gateway that is available from the manufacturer with *Anyfi.net* *software* pre-installed.

***NOTE** Anyfi.net software can easily be installed on any Wi-Fi router that runs OpenWrt. Please see http://anyfi.net/openwrt/INSTALL for step-by-step instructions.*



Wireless Network: Master "Inteno-1CD2" (wl0)

The *Device Configuration* section covers settings which are shared among all defined wireless networks (if the radio hardware is multi-SSID capable). Per network settings like encryption or operation mode are grouped in the *Interface Configuration*.

Device Configuration

Controller: demo.anyfi.net
A comma separated string of Fully Qualified Domain Names or IP addresses

Floor: 5
The percentage of available spectrum and backhaul that mobile users are allowed to consume even if there is competition with the primary user

Ceiling: 75
The maximum percentage of available spectrum and backhaul that mobile users are allowed to consume

Uplink:
The total upstream bandwidth available on the WAN connection, in bits per second

Downlink:
The total downstream bandwidth available on the WAN connection, in bits per second

Interface Configuration

ESSID: Inteno-1CD2

Figure 8: The Inteno VG50 comes with Anyfi.net software pre-installed and the Controller IP address or fully qualified domain name can be configured through the WEBUI admin interface.

Log into the VG50’s WEBUI as the "admin" user and navigate to the Wi-Fi configuration page. Once there select the `Anyfi.net` tab under `Device Configuration`. Enter *`ip1`* into the `Controller` field. Press the `Save` button. Since the Controller acts as a rendezvous point for all other network elements you do not need to configure any other IP addresses.

The VG50's WEBUI doesn't support configuring an optimizer key, but you can accomplish the same task using the commands below (or the Anyfi.net TR-069 vendor extension).

Connect to the VG50 with SSH

```
user@host$ ssh root@192.168.1.1
root@Inteno:~#
```

Configure optimizer key

```
root@Inteno:~# uci set anyfi.optimizer.key="d4577fe79131b980a5243a601ec6"
root@Inteno:~# uci commit anyfi
```

Restart Wi-Fi subsystem to ensure settings take effect

```
root@Inteno:~# wifi
...
wl0.1: starting myfid
```

Disconnect from the VG50

```
root@Inteno:~# exit
user@host$
```

Repeat the process for the other VG50, and ensure that they both have IP connectivity on the WAN port. Then verify that your Gateway, Optimizer and CPE have all registered successfully with the Controller using the operational commands below.

Connect to the Controller

```
admin@mgmt:~$ ssh anyfi-ctrl
vyatta@anyfi-ctrl:~$
```

List connected radios

```
vyatta@anyfi-ctrl:~$ show anyfi controller radios
======== RADIOS ==========================================================
hwaddr                                ip  groups
==========================================================================
00:22:07:d9:67:b2      118.13.81.8:52814  all
00:22:07:d9:69:14    220.88.12.230:43769  all
```

List connected optimizers

```
vyatta@anyfi-ctrl:~$ show anyfi controller optimizers
======== OPTIMIZERS ======================================================
ip                   optimizer key     load
==========================================================================
83.145.41.85:10000   d4577fe79131b980    0%
```

List connected services

```
vyatta@anyfi-ctrl:~$ show anyfi controller services
======== SERVICES ========================================================
uuid                                  ssid                auth  groups
==========================================================================
31fc0dc6-625f-d8fe-1d8a-f86ab33b47c2  Inteno-D967B2       psk   other
5cbdc1ee-625f-d8fe-1f8a-bc177e36f1b3  Inteno-D96914       psk   other
bc67d7d0-0c8d-b505-e17f-6372bda8993e  Hotspot 1X          eap   gateways
```

Disconnect from the Controller

```
vyatta@anyfi-ctrl:~$ exit
admin@mgmt:~$
```

Residential gateways should show up both as *radios* and as *services* in the lists above. You should also see the Gateway, but registered only as a *service* since it does not have a local Wi-Fi access point *radio*.

Instructions

We are now ready to try out the user experiences previously described, using our newly installed and configured Software-Defined Wireless Networking core.

SIMPLE Verification

Verify that a device previously connected to one of the VG50s will automatically connect to the same logical Wi-Fi network through the *radio* of the other, now as a guest user.

Note that the solution provides complete mobility with session continuity on Layer 3, even to and from the home Wi-Fi network. The user can thus start e.g. a Skype call in their own home, connected locally to the first VG50, and then walk out the door, being handed over to the other VG50 without disruption to the call.

Also note that while devices previously connected to one of the VG50s will continue to display the SSID of its home network even when connected through the other VG50, other devices will not detect this SSID. Each device sees only the networks that are relevant to them.

Optimizer Verification

If you have configured an *optimizer key* into one or both of the VG50s you will also be able to verify that the central breakout of Internet-bound traffic works as expected. Note that your private IP address does not change and that you can still reach network resources on the LAN when connected remotely, but that your public IP address will now be `ip4`, the IP address configured into of the WLAN gateway.

You can connect to the Optimizer and use its operational commands to examine its state.

Connect to the Optimizer

```
admin@mgmt:~$ ssh anyfi-opt
vyatta@anyfi-opt:~$
```

List connected clients

```
vyatta@anyfi-opt:~$ show anyfi optimizer opt1 clients
======== CLIENTS ========================================================
                                                total        breakout
mac                 breakout ip     vlan     up   down     up   down     access
==========================================================================
30:f7:c5:26:df:80   172.16.1.20/16     -   85.8K   193K  85.5K   153K     allow
```

Disconnect from the Optimizer

```
vyatta@anyfi-opt:~$ exit
admin@mgmt:~$
```

HOTSPOT Verification

Verify that you can connect a device to the IEEE 802.1X authenticated Wi-Fi network with SSID "Traditional Hotspot". This network should be distributed by both VG50s. Connect a device to the network and authenticate with either the EAP-PEAP or EAP-SIM credentials you configured into the AAA server.

You can now use the below operational commands to examine the state in the Gateway.

Connect to the Gateway

```
admin@mgmt:~$ ssh anyfi-gw
vyatta@anyfi-gw:~$
```

List connected clients

```
vyatta@anyfi-gw:~$ show anyfi gateway 1x-gw clients
======== CLIENTS ==========================================================
                                              total
mac                ip                vlan     up  down    access  duration
===========================================================================
28:cf:e9:49:d7:09  192.168.2.10         2  29.8K  245K     allow       25s
```

Disconnect from the Gateway

```
vyatta@anyfi-gw:~$ exit
admin@mgmt:~$
```

Note that the mobile device will retain the same IP address as it roams across the two VG50s, ensuring full mobility with session continuity on Layer 3. This is because the Gateway serves as the termination point for the IEEE 802.11 protocol and also as a mobility anchor. This architecture also effectively isolates the traffic on the hotspot network from the host’s own traffic.

Chapter 6: Integration

In this chapter we describe how your verified SDWN core can be further integrated towards third party systems for CPE management and network monitoring. We exemplify by installing GenieACS and Splunk, both available for download free of charge, on the management VM. In a production environment these systems would however typically run independently of the SDWN core.

GenieACS for TR-069 CPE Management

GenieACS is an open source TR-069 Automatic Configuration Server (ACS) implementation flexible enough to support configuration of the Anyfi.net TR-069 vendor extension attributes without customization.

Before the management VM can serve as a TR-069 ACS it will need an external network interface on which it can be reached by CPEs under management. Add a network interface bridged to your external network and make note of a public IP address (*ip5*) to be used for the ACS.

Configure the management VM to use *ip5* on its newly added external network interface.

Configure networking on management VM

```
admin@mgmt:~$ sudo vi /etc/network/interfaces
```

Review changes

```
admin@mgmt:~$ sudo cat /etc/network/interfaces
# The loopack network interface
auto lo
iface lo inet loopback

# Management
auto eth0
iface eth0 inet dhcp

# Management
auto eth1
iface eth1 inet static
  address 10.0.10.101
  network 10.0.10.0
  netmask 255.255.255.0

# WAN
auto eth2
```

```
iface eth2 inet static
  address ip6
  network x.x.x.x
  netmask x.x.x.x
  gateway x.x.x.x
  dns-nameservers x.x.x.x
```

Apply changes

```
admin@mgmt:~$ sudo ifup eth2
```

Now we are ready to install and start GenieACS. The management VM contains an installation script for the purpose.

Install GenieACS

```
admin@mgmt:~$ ./install-genieacs
```

Start GenieACS

```
admin@mgmt:~$ sudo service genieacs start
```

GenieACS comes with an administration WEBUI reachable at localhost port 3000. Use SSH port forwarding to access this interface from your host machine.

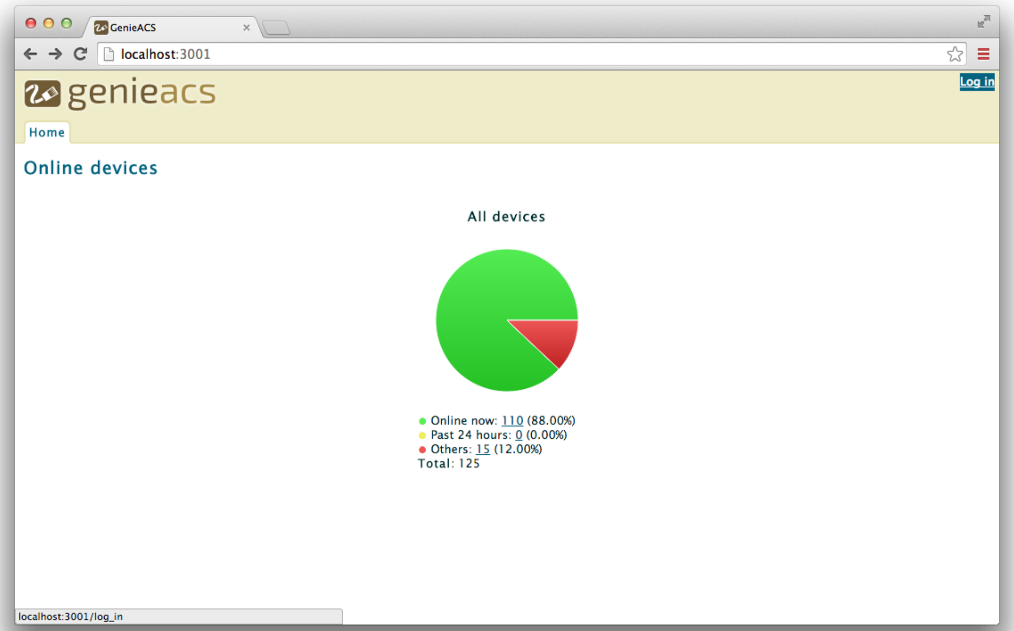
Log out of the management VM

```
admin@mgmt:~$ exit
host:~$
```

Log into the management VM with port forwarding

```
host:~$ ssh -l admin [localhost -p 8022 | x.x.x.x]
    -L 3001:localhost:3000
admin@mgmt:~$
```

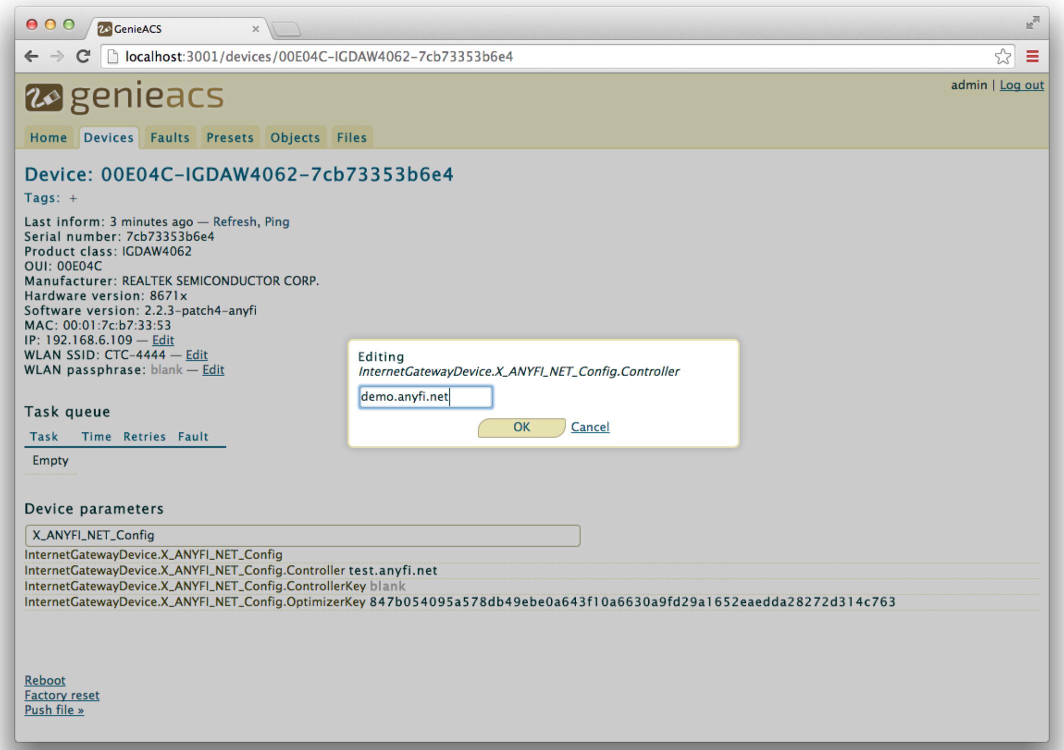
The management web interface should now be accessible at http://localhost:3001.



● Online now: 110 (88.00%)
● Past 24 hours: 0 (0.00%)
● Others: 15 (12.00%)
Total: 125

Figure 9: The GenieACS welcome screen. Note the log in link in the top right corner.

Log in with the username "admin" and password "admin" and familiarize yourself with the user interface. If you have a CPE that implements the Anyfi.net TR-069 vendor extension connected to the ACS you can change its configuration by entering `X_ANYFI_NET` in the parameter search box and clicking `Edit` on the relevant parameter.



Device: 00E04C-IGDAW4062-7cb73353b6e4
Tags: +

Last inform: 3 minutes ago — Refresh, Ping
Serial number: 7cb73353b6e4
Product class: IGDAW4062
OUI: 00E04C
Manufacturer: REALTEK SEMICONDUCTOR CORP.
Hardware version: 8671x
Software version: 2.2.3-patch4-anyfi
MAC: 00:01:7c:b7:33:53
IP: 192.168.6.109 — Edit
WLAN SSID: CTC-4444 — Edit
WLAN passphrase: blank — Edit

Editing
InternetGatewayDevice.X_ANYFI_NET_Config.Controller
demo.anyfi.net
OK Cancel

Task queue

Task	Time	Retries	Fault
Empty			

Device parameters

X_ANYFI_NET_Config

```
InternetGatewayDevice.X_ANYFI_NET_Config
InternetGatewayDevice.X_ANYFI_NET_Config.Controller test.anyfi.net
InternetGatewayDevice.X_ANYFI_NET_Config.ControllerKey blank
InternetGatewayDevice.X_ANYFI_NET_Config.OptimizerKey 847b054095a578db49ebe0a643f10a6630a9fd29a1652eaedda28272d314c763
```

Reboot
Factory reset
Push file »

Figure 10: Screenshot of the GenieACS GUI when editing the Controller parameter.

The process of configuring *Anyfi.net enabled* devices can be automated using the preset functionality of GenieACS. Simply create a preset that unconditionally sets e.g. `InternetGatewayDevice.X_ANYFI_NET_Config.Controller` to *`ip1`*.

Splunk for Network Monitoring and Dashboards

Splunk is an operational intelligence and structured data analysis platform well suited for visualization and monitoring of radio link-level accounting information from the Controller.

The management VM contains an installation script for installing Splunk.

Install Splunk

```
admin@mgmt:~$ ./install-splunk
```

Start Splunk

```
admin@mgmt:~$ sudo service splunk start
```

Splunk's WEBUI is reachable at localhost port 8000. Use SSH port forwarding to access this interface from your host machine.

Log out of the management VM

```
admin@mgmt:~$ exit
host:~$
```

Log into the management VM with port forwarding

```
host:~$ ssh -l admin [localhost -p 8022 | x.x.x.x]
    -L 8001:localhost:8000
admin@mgmt:~$
```

Splunk web interface should now be accessible at http://localhost:8001.

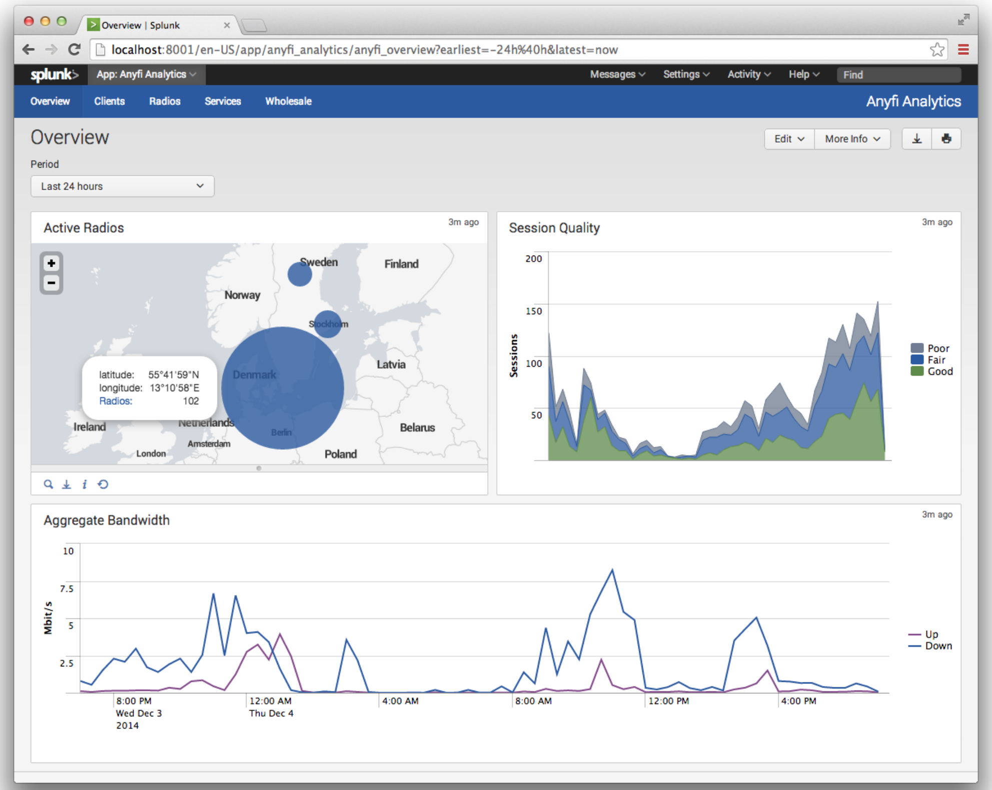


Figure 11: The Anyfi Analytics app for Splunk provides basic dashboards for monitoring your SDWN network and can easily be extended with more specialized data analysis.