Anyfi Networks | Carrier Wi-Fi System

GATEWAY

REFERENCE GUIDE

Overview
Installation
Basic Configuration
Wi-Fi Client Isolation
Wi-Fi Security Settings
Opportunistic Key Caching
WPA2 Preauthentication
SHA-256 Key Derivation
IEEE 802.11r Fast Transition
IEEE 802.1X and EAP
SDWN Settings
RADIUS for AAA
GRE for Payload
System Monitoring

Anyfi Networks

Västergatan 31 B
21121 Malmö
Sweden
info@anyfinetworks.com

COPYRIGHT

NOTICES

RELEASE DATE: 27th of April 2015

DOCUMENT REVISION: R1F v05

RELEASED WITH: CARRIER WI-FI SYSTEM R1F

Contents

Preface

This document details how to install, configure and integrate the Gateway component of our Carrier Wi-Fi System.

Intended Audience

This document is intended for system and network administrators. 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.

Readers lacking a basic understanding of Software-Defined Wireless Networking (SDWN) concepts are encouraged to study the materials available at www.anyfinetworks.com/resources.

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 running system will interrupt service.*

Notes provide important information about the structure or functioning of the system, as

in the following example:

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: Functionality Overview

The Software-Defined Wireless Networking (SDWN) architecture separates the radio access problem from service definition. The Gateway network element plays an essential role in the latter area, allowing an operator to design and implement a carrier Wi-Fi service in the trusted environment of a data center or mobile core, safe in the knowledge that the radio access problem can be separately addressed later.

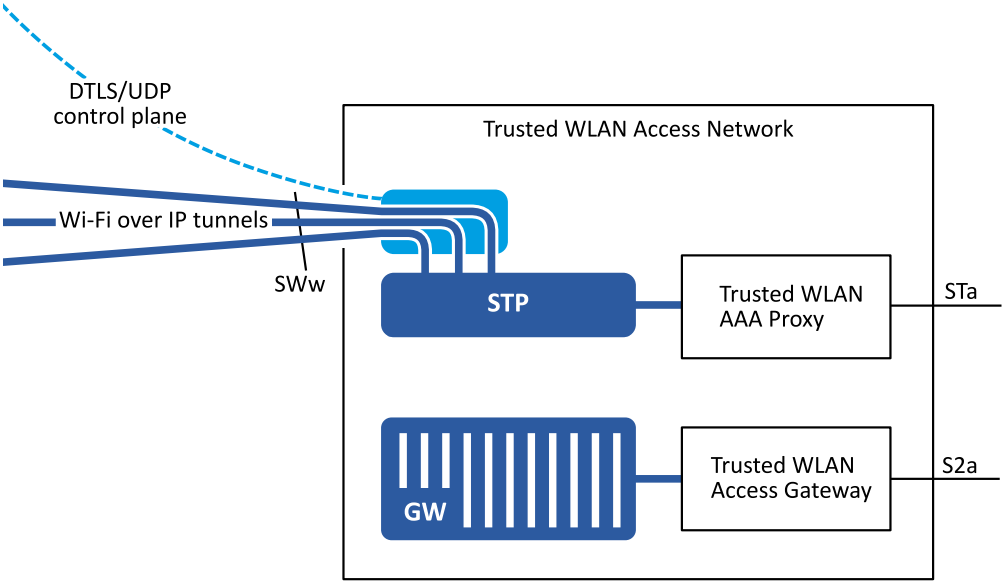

Figure 1: The Gateway integrated towards a WLAN gateway in a 3GPP mobile core.

The Gateway implements an IEEE 802.11 stack, complete with WPA and WPA2 security; CCMP (AES) and TKIP (RC4) ciphers; Pre-Shared Key (PSK) and IEEE 802.1X authentication; and RADIUS for Authentication, Authorization and Accounting (AAA).

NOTE *The Gateway serves as a gateway in the Software-Defined Wireless Networking (SDWN) sense; it processes IEEE 802.11 frames coming in and going out on SDWN data plane tunnels. While the Gateway can also be configured as an IP gateway we recommend integrating it as a pure Layer 2 element towards a third party WLAN gateway.*

Concepts and Principles

In this section we introduce some basic concepts and principles that we will use when configuring the Gateway.

Apart from the concepts introduced below you should also be familiar with the following IEEE 802.11 terms:

- Station (STA)
- Basic Service Set (BSS)
- Extended Service Set (ESS)
- Service Set Identifier (SSID)

Please refer to the IEEE 802.11 standard for their definitions.

Service Termination Point

In terms of interfaces the Gateway resembles a carrier-grade Wi-Fi access point, except that it sends and receives raw (encrypted) IEEE 802.11 frames not over radio but over UDP/IP based SDWN data plane tunnels. Reflecting this resemblance with an access point (AP) refer to it as a *service termination point*.

There is however one noteworthy difference between an access point and the Gateway: scale. An access point typically contributes one or two Basic Service Sets (BSSes) to an Extended Service Set (ESS). A single Gateway can contribute millions: all the virtual access points allocated on connected SDWN *radios*.

Service

The IEEE 802.11 standard refers to a logical network as an Extended Service Set (ESS), which in turn consists of Distribution System (DS) and a number of Basic Service Sets (BSS). An ESS is uniquely identified by its Service Set Identifier (SSID), a 32-byte string containing the Wi-Fi network name. All access points within the ESS must thus be configured with the same SSID.

We use the term *service* to refer to a logical network (essentially synonymous with ESS in an IEEE 802.11 context). The Software-Defined Wireless Networking (SDWN) architecture however operates on a scale (potentially millions of ESSes in a single system) at which it is difficult to ensure the uniqueness of SSIDs. We therefore use a Universally Unique Identifier (UUID), the *service* UUID, to uniquely identify a *service*. All *service termination points* providing access to a service must be configured with the same SSID and *service* UUID.

Chapter 2: Installation

We use the Vyatta Network OS as the base operating system for the Gateway. This ensures access to advanced IP networking functionality on a secure and trusted platform, and also facilitates installation in many different environments.

This software has been verified on the following versions of the Vyatta Network OS:

- Vyatta Core 6.5
- Vyatta Core 6.6
- Brocade Vyatta 5400 vRouter

The above operating systems can be run on all major hypervisors as well as bare metal x86 hardware. We provide software for both 32-bit and 64-bit architectures, but a 64-bit OS is recommended.

Installing as a Virtual Appliance

A Vyatta Core 6.6R1 64-bit virtual appliance is available for download at http://www.anyfinetworks.com/download/vyatta-anyfi-r1f.ova. Import the virtual appliance into your hypervisor of choice and follow the instructions in the next section to install the Gateway software.

We also make example SDWN cores with integrated Gateways available at www.anyfinetworks.com/download. Each example core comes with a Getting Started Guide containing step-by-step instructions on how to install the core on VMware ESXi or Oracle VM VirtualBox.

Installing as a Vyatta Package

It can sometimes be beneficial to install the Gateway software onto a running Vyatta system. One example is public clouds like Amazon AWS, where AMIs for the supported base operating systems are readily available. Another example is when the operator wishes to use physical hardware for the installation. Vyatta Network OS can then be installed on the hardware, followed by Gateway software.

First configure the Vyatta system to use Anyfi Networks' software repository.

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

Add Anyfi Networks' package repository	vyatta@vyatta# **edit system package repository anyfi** [edit system package repository anyfi] vyatta@vyatta# **set url http://packages.anyfinetworks.com/vyatta** [edit system package repository anyfi] vyatta@vyatta# **set components "main contrib non-free"** [edit system package repository anyfi] vyatta@vyatta# **set distribution r1f** [edit system package repository anyfi] vyatta@vyatta# **top** [edit]

Review changes	vyatta@vyatta# **show system package repository**
	+repository anyfi {
	+ components "main contrib non-free"
	+ distribution r1f
	+ url http://packages.anyfinetworks.com/vyatta
	+}
	repository community {
	components main
	distribution stable
	password ""
	url http://packages.vyatta.com/vyatta
	username ""
	}
	[edit]

Commit, save and exit configuration mode	vyatta@vyatta# **commit** vyatta@vyatta# **save** vyatta@vyatta# **exit**
Download Anyfi Networks' software signing key	vyatta@vyatta$ **wget ** **http://packages.anyfinetworks.com/vyatta/pubkey.gpg -O anyfi.gpg**
Verify key integrity	vyatta@vyatta$ **sha1sum anyfi.gpg** b5a3a3233e3348ef555ba4fae11941f2339bbb88 anyfi.gpg
Install key as trusted	vyatta@vyatta$ **sudo apt-key add anyfi.gpg**
Update the software package database	vyatta@vyatta$ **sudo apt-get update**

Once the repository has been added to the system installation of the Gateway software is trivial.

Install the Gateway software	vyatta@vyatta:~$ **sudo apt-get install -y ** **vyatta-anyfi-gateway anyfi-gateway**

This will install two packages: `vyatta-anyfi-gateway` containing the Vyatta CLI integration for the Gateway and `anyfi-gateway` containing the Gateway software itself.

***NOTE** The Gateway software is freely available as 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 regarding other licensing options.

Upgrading from a Previous Release

To upgrade from a previous release you only need to change distribution within the repository.

Enter configuration mode	`vyatta@vyatta:~$ configure` `[edit]`
Change distribution	`vyatta@vyatta# set system package repository anyfi distribution r1f` `[edit]`

Review changes	`vyatta@vyatta# show system package repository anyfi`
	` components "main contrib non-free"`
	`>distribution r1f`
	` password ""`
	` url http://packages.anyfinetworks.com/vyatta`
	` username ""`
	` [edit]`

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

Then follow the steps below to upgrade the Gateway software to the current release.

Upgrading to a Newer Version

The Gateway software can be upgraded with the commands below.

Update the index of configured repositories	`vyatta@vyatta:~$ sudo apt-get update`
Upgrade the Gateway software	`vyatta@vyatta:~$ sudo apt-get upgrade`

Restart the Gateway	`vyatta@vyatta:~$ restart anyfi gateway instance`
	`Stopping anyfi gateway: anyfi-gateway.`
	`Starting anyfi gateway: anyfi-gateway.`

This will install new versions of the packages containing the Gateway software, if such are available in Anyfi Networks' package repository. Note that the upgraded software will remain compatible with the interfaces described in this guide.

***CAUTION** Restarting the Gateway will disrupt service for associated Wi-Fi clients.*

Chapter 3: Configuration

In this chapter we show how to configure the Gateway for basic as well as more advanced use-cases.

Basic Networking

The Gateway will need at least two network interfaces; one for its SDWN data and control plane (UDP/IP) connections and a (logical) bridge for Wi-Fi client traffic (Ethernet). We here assume that `eth1` will be used for IP connectivity to the outside world and `eth2` will be used for Wi-Fi client traffic. In Chapter 4 we discuss how Wi-Fi client traffic can instead be tunneled out over GRE.

Enter configuration mode	vyatta@vyatta:~$ **configure** [edit]
Configure basic IP networking	vyatta@vyatta# **set interfaces ethernet *eth1* address *x.x.x.x/xx*** [edit] vyatta@vyatta# **set system name-server *x.x.x.x*** [edit] vyatta@vyatta# **set system gateway-address *x.x.x.x*** [edit]
Disable IP forwarding	vyatta@vyatta# **set system ip disable-forwarding** [edit]
Configure bridging	vyatta@vyatta# **set interfaces bridge br0** [edit] vyatta@vyatta# **set interfaces ethernet *eth2* bridge-group bridge br0** [edit]
Commit, save and exit configuration mode	vyatta@vyatta# **commit** vyatta@vyatta# **save** vyatta@vyatta# **exit**

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

Basic Examples

In this section we provide two example Wi-Fi network configurations.

Open Wi-Fi

We start by configuring an open Wi-Fi network with the SSID "ex-open".

Enter configuration mode	vyatta@vyatta:~$ **configure** [edit]
Create a Gateway instance	vyatta@vyatta# **edit service anyfi gateway "open-gw"** [edit service anyfi gateway open-gw] vyatta@vyatta# **set controller *x.x.x.x*** [edit service anyfi gateway open-gw] vyatta@vyatta# **set bridge br0** [edit service anyfi gateway open-gw] vyatta@vyatta# **set ssid "Open Wi-Fi"** [edit service anyfi gateway open-gw] vyatta@vyatta# **top** [edit]
Review changes	vyatta@vyatta# **show service anyfi gateway** +gateway open-gw { + bridge br0 + controller *x.x.x.x* + ssid "Open Wi-Fi" +} [edit]
Commit, save and exit configuration mode	vyatta@vyatta# **commit** vyatta@vyatta# **save** vyatta@vyatta# **exit**

The newly created *service* should now be registered with the Controller. For more information on how to distribute the *service* to *clients* refer to the Reference Guide for the Controller.

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

Secure Wi-Fi

We now configure a WPA2 protected Wi-Fi network with EAP authentication.

Enter configuration mode	vyatta@vyatta:~$ **configure** [edit]

Create a Gateway instance	vyatta@vyatta# **edit service anyfi gateway "1x-gw"** [edit service anyfi gateway 1x-gw] vyatta@vyatta# **set controller *x.x.x.x*** [edit service anyfi gateway 1x-gw] vyatta@vyatta# **set bridge br0** [edit service anyfi gateway 1x-gw] vyatta@vyatta# **set ssid "Secure Wi-Fi"** [edit service anyfi gateway 1x-gw] vyatta@vyatta# **set wpa2** [edit service anyfi gateway 1x-gw] vyatta@vyatta# **set authentication eap radius-server *x.x.x.x* secret *"secret"*** [edit service anyfi gateway 1x-gw] vyatta@vyatta# **top** [edit]

Review changes	vyatta@vyatta# **show service anyfi gateway**
	+gateway 1x-gw {
	+ authentication {
	+ eap {
	+ radius-secret <secret>
	+ radius-server x.x.x.x
	+ }
	+ }
	+ bridge br0
	+ controller x.x.x.x
	+ ssid "Secure Wi-Fi"
	+ wpa2 {
	+ }
	+}
	[edit]

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

The newly created *service* should now be registered with the Controller. For more information on how to distribute the *service* to *clients* please refer to the Reference Guide for the Controller.

Wi-Fi Client Isolation

IEEE 802.11 provides full Layer 2 connectivity between all STAs connected to the same Extended Service Set (ESS). This is however rarely the desired user experience in a public Wi-Fi network, as it lets users access each other's shared folders etc. Carrier-grade Wi-Fi access points therefore often provide a configuration option for so-called client isolation. When this feature is enabled two devices connected to the same access point will be prevented from communicating directly with each other on Layer 2.

The Gateway provides a similar configuration option:

Enter configuration mode	`vyatta@vyatta:~$ configure` `[edit]`
Enable isolation of clients on Layer 2	`vyatta@vyatta# set service anyfi gateway "open-gw" isolation` `[edit]`

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

Enabling isolation will prevent all STAs connected to the Gateway instance from communicating directly with each other on Layer 2.

Wi-Fi Security Settings

The Gateway supports both WPA and WPA2 security protocols. Each security protocol can be configured to use CCMP and/or TKIP block ciphers. It is also possible to configure the group rekey interval and whether or not to renegotiate the group key after every disassociation (strict rekeying).

Enter configuration mode	`vyatta@vyatta:~$ configure` `[edit]`
Enable WPA2 with CCMP only	`vyatta@vyatta# set service anyfi gateway "1x-gw" wpa2 ciphers ccmp` `[edit]`
Enable WPA with TKIP only	`vyatta@vyatta# set service anyfi gateway "1x-gw" wpa ciphers tkip` `[edit]`
Set the group rekey interval is seconds	`vyatta@vyatta# set service anyfi gateway "1x-gw" rekey-interval 300` `[edit]`
Enable strict rekeying	`vyatta@vyatta# set service anyfi gateway "1x-gw" strict-rekey` `[edit]`

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

NOTE *The Gateway supports hardware AES acceleration on Intel CPUs with the AES-NI instruction set (Sandy Bridge and later). We therefore recommend this platform and the AES-based CCMP cipher for high throughput processing of encrypted Wi-Fi traffic.*

Advanced WPA2 Settings

The Gateway supports a number of optional and advanced features of the WPA2 security protocol.

PMKSA Cache Size

WPA2 defines a mechanism for caching of the Pairwise Master Key Security Association (PMKSA) derived in IEEE 802.1X autentication. The Gateway uses this mechanism in two contexts: when a *client* connects for the second time through a certain *radio* ("sticky key caching") and when the *client* roams from one *radio* to another ("opportunistic key caching"). This allows the Gateway to skip IEEE 802.11 authentication in these contexts, thereby reducing the number of frames exchanged as part of the authentication and shortening the associated delay, without negatie security impact. This feature also reduces the load on the authentication RADIUS server.

The Gateway will cache up to eight (8) PMKSAs per *client* by default, but this cache size is configurable.

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

Set the PMKSA cache size	`vyatta@vyatta# edit service anyfi gateway "1x-gw" wpa2`
	`[edit]`
	`[edit service anyfi gateway 1x-gw wpa2]`
	`vyatta@vyatta# set pmksa-cache-size 16`
	`[edit service anyfi gateway 1x-gw wpa2]`
	`vyatta@vyatta# top`
	`[edit]`

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

The Gateway will now cache up to 16 PMKSAs per *client*. The RADIUS server controls the maximum lifetime of each PMKSA by setting the `Session-Timeout` attribute in the `Access-Accept` or `CoA-Request` message.

WPA2 Preauthentication

The WPA2 security protocol also allows a client to preauthenticate with a new access point while staying associated to the previous access point. The Gateway can be configured to support such WPA2 Preauthentication.

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

Enable WPA2 Preauthentication	vyatta@vyatta# **edit service anyfi gateway "1x-gw" wpa2** [edit] [edit service anyfi gateway 1x-gw wpa2] vyatta@vyatta# **set preauthentication** [edit service anyfi gateway 1x-gw wpa2] vyatta@vyatta# **top** [edit]
Commit, save and exit configuration mode	vyatta@vyatta# **commit** vyatta@vyatta# **save** vyatta@vyatta# **exit**

The Gateway will now advertise support for WPA2 Preauthentication.

SHA-256 Key Derivation

A new key derivation mechanism leveraging the SHA-256 secure hash function is defined for WPA2. In its default configuration the Gateway will only support the most widely implemented SHA-1 Key Derivation mechanism, but it can be configured to support SHA-256 Key Derivation (as well or exclusively).

Enter configuration mode	vyatta@vyatta:~$ **configure** [edit]
Enable SHA-256 Key Derivation	vyatta@vyatta# **edit service anyfi gateway "1x-gw" wpa2** [edit] [edit service anyfi gateway 1x-gw wpa2] vyatta@vyatta# **set key-derivation sha1** [edit service anyfi gateway 1x-gw wpa2] vyatta@vyatta# **set key-derivation sha256** [edit service anyfi gateway 1x-gw wpa2] vyatta@vyatta# **top** [edit]
Commit, save and exit configuration mode	vyatta@vyatta# **commit** vyatta@vyatta# **save** vyatta@vyatta# **exit**

The Gateway will now advertise support for both SHA-1 and SHA-256 Key Derivation. The *client* ultimately determines which mechanism is used.

IEEE 802.11r Fast Transition Settings

The Gateway supports IEEE 802.11r Fast Transition (FT). This allows *clients* that support this standard to roam from *radio* to *radio* with minimal reassociation delay.

Enter configuration mode	vyatta@vyatta:~$ **configure** [edit]

Enable IEEE 802.11r Fast Transition	vyatta@vyatta# **set service anyfi gateway "1x-gw" ft** [edit]
Commit, save and exit configuration mode	vyatta@vyatta# **commit** vyatta@vyatta# **save** vyatta@vyatta# **exit**

The Gateway will now advertise support for IEEE 802.11r Fast Transition (FT) in addition to any other configured security protocols. The cipher suites configured for WPA2 will be available to *clients* authenticating through the FT security protocol as well. If WPA2 is disabled then only AES CTR with CBC MAC Protocol (CCMP) is supported for *clients* using Fast Transition (FT).

Mobility Domain and Over-the-DS Mode

A number of parameters of the IEEE 802.11r Fast Transition implementation are configurable.

Enter configuration mode	vyatta@vyatta:~$ **configure** [edit]
Configure IEEE 802.11r Fast Transition	vyatta@vyatta# **edit service anyfi gateway "1x-gw" ft** [edit service anyfi gateway 1x-gw ft] vyatta@vyatta# **set mobility-domain 123** [edit service anyfi gateway 1x-gw ft] vyatta@vyatta# **set over-the-ds** [edit service anyfi gateway 1x-gw ft] vyatta@vyatta# **set reassociation-timeout 20** [edit service anyfi gateway 1x-gw ft] vyatta@vyatta# **top** [edit]
Commit, save and exit configuration mode	vyatta@vyatta# **commit** vyatta@vyatta# **save** vyatta@vyatta# **exit**

The Gateway will now advertise FT mobility domain 123 and support for FT Over-the-DS. The *client* will be given 20 seconds to complete reassociation from the time its IEEE 802.11 Authentication frame is received by the Gateway.

RADIUS Settings

In this section we illustrate how to configure the Gateway to use an external RADIUS server for Authentication, Authorization and Accounting (AAA). In Chapter 4 we will go into the details of RADIUS interface capabilities.

Authentication

The Gateway implements an IEEE 802.1X pass-through authenticator and can be configured to use an external RADIUS server for EAP authentication.

Enter configuration mode	vyatta@vyatta:~$ **configure** [edit]
Configure RADIUS authentication	vyatta@vyatta# **edit service anyfi gateway "1x-gw" authentication** [edit service anyfi gateway 1x-gw authentication] vyatta@vyatta# **set eap radius-server *x.x.x.x* secret *"secret"*** [edit service anyfi gateway 1x-gw authentication] vyatta@vyatta# **top** [edit]
Configure non-standard RADIUS authentication port	vyatta@vyatta# **edit service anyfi gateway "1x-gw" authentication** [edit service anyfi gateway 1x-gw authentication] vyatta@vyatta# **set eap radius-server *x.x.x.x* port *xxxx*** [edit service anyfi gateway 1x-gw authentication] vyatta@vyatta# **top** [edit]
Commit, save and exit configuration mode	vyatta@vyatta# **commit** vyatta@vyatta# **save** vyatta@vyatta# **exit**

All EAP types that can provide IEEE 802.11 keying material are supported, including EAP-SIM and EAP-AKA.

Authorization

The Gateway can also use an external RADIUS server to authorize access to the network, even when *clients* are not authenticated (open network). Some vendors refer to this as "MAC authentication". We use the term authorization instead to make the distinction between authentication and authorization clear.

Enter configuration mode	vyatta@vyatta:~$ **configure** [edit]
Configure RADIUS authorization	vyatta@vyatta# **edit service anyfi gateway "open-gw" authorization** [edit service anyfi gateway open-gw authorization] vyatta@vyatta# **set radius-server *x.x.x.x*** [edit service anyfi gateway open-gw authorization] vyatta@vyatta# **set radius-port *xxxx*** [edit service anyfi gateway open-gw authorization] vyatta@vyatta# **set radius-secret *secret*** [edit service anyfi gateway open-gw authorization] vyatta@vyatta# **top** [edit]
Commit, save and exit	vyatta@vyatta# **commit**

configuration mode	vyatta@vyatta# **save** vyatta@vyatta# **exit**

RADIUS authorization can be combined with pre-shared-key (PSK) authentication. When EAP authentication is configured the RADIUS authentication server is expected to also handle authorization and it is therefore not possible to configure a separate authorization server.

Accounting

Accounting information is provided on a RADIUS interface in standard RFC 2866 format. Use the below operational commands to configure RADIUS accounting.

Enter configuration mode	vyatta@vyatta:~$ **configure** [edit]
Configure RADIUS accounting	vyatta@vyatta# **edit service anyfi gateway "1x-gw" accounting** [edit service anyfi gateway 1x-gw accounting] vyatta@vyatta# **set radius-server *x.x.x.x* secret *"secret"*** [edit service anyfi gateway 1x-gw accounting] vyatta@vyatta# **top** [edit]
Configure non-standard RADIUS accounting port	vyatta@vyatta# **edit service anyfi gateway "1x-gw" accounting** [edit service anyfi gateway 1x-gw accounting] vyatta@vyatta# **set radius-server *x.x.x.x* port *xxxx*** [edit service anyfi gateway 1x-gw accounting] vyatta@vyatta# **top** [edit]
Commit, save and exit configuration mode	vyatta@vyatta# **commit** vyatta@vyatta# **save** vyatta@vyatta# **exit**

Network Access Server

The NAS Identifier and local port used by the RADIUS client in the Gateway is configurable.

Enter configuration mode	vyatta@vyatta:~$ **configure** [edit]
Configure RADIUS client	vyatta@vyatta# **edit service anyfi gateway "1x-gw" nas** [edit service anyfi gateway 1x-gw nas] vyatta@vyatta# **set identifier *"identifier"*** [edit service anyfi gateway 1x-gw nas] vyatta@vyatta# **set port *xxxx*** [edit service anyfi gateway 1x-gw nas] vyatta@vyatta# **set ip-address *x.x.x.x*** [edit service anyfi gateway 1x-gw nas]

	vyatta@vyatta# **top** [edit]
Commit, save and exit configuration mode	vyatta@vyatta# **commit** vyatta@vyatta# **save** vyatta@vyatta# **exit**

NOTE *The RADIUS extensions for dynamic authorization defined in RFC 5176 require that Change-of-Authorization and Disconnect-Request messages be sent to port 3799. For full compliance the Gateway must thus be configured to use this port.*

SDWN Settings

In this section we show how to configure the SDWN data and control planes of the Gateway.

Controller

In the Software-Defined Wireless Networking (SDWN) architecture the control plane is centralized in a Controller, while the data plane remains distributed. Data plane elements like the Gateway are configured with the IP address or fully qualified domain name (FQDN) of the Controller.

Enter configuration mode	vyatta@vyatta:~$ **configure** [edit]
Configure the Controller IP or domain name	vyatta@vyatta# **set service anyfi gateway "1x-gw" controller *ip*** [edit]
Commit, save and exit configuration mode	vyatta@vyatta# **commit** vyatta@vyatta# **save** vyatta@vyatta# **exit**

The Gateway should now have registered with the specified Controller and is ready to accept incoming SDWN data plane tunnels from *radios* connected to the same Controller.

Enabling Controller Authentication

The SDWN control plane is protected with Datagram Transport Layer Security (DTLS). To enable Controller authentication configure the Gateway with a *controller key*.

Enter configuration mode	vyatta@vyatta:~$ **configure** [edit]
Configure the controller key	vyatta@vyatta# **set service anyfi gateway "1x-gw" controller *ip* key *9878d127f83b41dee54034e88c627b0ac886c44d2a209f3f4447a41a740cddcb*** [edit]

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

The Gateway will now authenticate the Controller during the initial DTLS handshake.

NOTE *The Controller is a controller in the Software-Defined Networking (SDN) sense of the word, not in the typical corporate WLAN sense. For example the Controller is not involved in IEEE 802.1X authentication and does not have access to end-user credentials or encryption keys. Configuring the Gateway with a Controller has no impact on system or user data plane security.*

UDP/IP Port Range

Gateways send and receive raw IEEE 802.11 frames over SDWN data plane UDP/IP tunnels and will also communicate with the Controller over UDP/IP. The UDP port range that a Gateway will use for such SDWN control and data plane communication is configurable.

Enter configuration mode	vyatta@vyatta:~$ **configure** [edit]
Configure the SDWN UDP port range	vyatta@vyatta# **set service anyfi gateway "1x-gw" port-range *xxxx-xxxx*** [edit]
Commit, save and exit configuration mode	vyatta@vyatta# **commit** vyatta@vyatta# **save** vyatta@vyatta# **exit**

Configuration of the UDP port range used for SDWN data and control can facilitate integration of the Gateway in environments with strict IP firewalling.

Load Balancing and Failover

Automatic load balancing and failover between multiple Gateways is built into the Software-Defined Wireless Networking (SDWN) architecture. All that is required from the operator is that they configure all Gateway instances with the same *service* UUID.

First generate a random UUID.

Generate a random UUID	vyatta@vyatta:~$ **cat /proc/sys/kernel/random/uuid** 635a7751-8b9e-4355-9e13-b86308d62b77

Configure the UUID into the first Gateway.

Connect to the first Gateway with SSH	vyatta@vyatta:~$ **ssh gw-1** vyatta@gw-1:~$

Enter configuration mode	`vyatta@gw-1:~$ configure` `[edit]`
Configure the service UUID	`vyatta@gw-1# set service anyfi gateway "1x-gw" uuid uuid` `[edit]`

Commit, save and exit configuration mode	`anyfi@gw-1# commit`
	`anyfi@gw-1# save`
	`anyfi@gw-1# exit`

Disconnect from the first Gateway	`vyatta@gw-1:~$ exit` `vyatta@vyatta:~$`

Configure the UUID into the second Gateway.

Connect to the second Gateway	`vyatta@vyatta:~$ ssh gw-2` `vyatta@gw-2:~$`
Enter configuration mode	`vyatta@gw-2:~$ configure` `[edit]`
Configure the service UUID	`vyatta@gw-2# set service anyfi gateway "1x-gw" uuid uuid` `[edit]`

Commit, save and exit configuration mode	`anyfi@gw-2# commit`
	`anyfi@gw-2# save`
	`anyfi@gw-2# exit`

Disconnect from the first Gateway	`vyatta@gw-2:~$ exit` `vyatta@vyatta:~$`

The Controller will now load balance clients across both Gateways. Should one of the Gateways fail (or be shutdown) then the Controller will send all Wi-Fi over IP tunnels to the other Gateway.

Chapter 4: Integration

In this section we detail how to integrate the Gateway towards external systems.

RADIUS for AAA

RADIUS interfaces allow for integration towards external Authentication, Authorization and Accounting (AAA) servers. In Chapter 3 we illustrated how to configure these interfaces. In this section we provide technical specifications necessary to verify compatibility with external systems.

Authentication

The Gateway implements an IEEE 802.1X pass-through authenticator with RADIUS support for Extensible Authentication Protocol (EAP) as defined in RFC 3579. In addition the following RADIUS message types are supported:

- `Access-Request`
- `Access-Accept`
- `Access-Reject`
- `Disconnect-Request`
- `Disconnect-ACK`
- `Disconnect-NAK`
- `CoA-Request`
- `CoA-ACK`
- `CoA-NAK`

The following table lists the supported RADIUS message attributes as well as the message types that may contain them.

Table 1: Supported RADIUS authentication message attributes.

Attribute	Message types
`State`	`all`
`Message-Authenticator`	`all`
`Event-Timestamp`	`CoA-Request, Disconnect-Request`

NAS-Identifier	Access-Request
NAS-IP-Address	Access-Request
NAS-Port-Type	Access-Request
User-Name	Access-Request, CoA-Request, Disconnect-Request
Chargeable-User-Identity	Access-Request, Access-Accept, CoA-Request, Disconnect-Request
Calling-Station-Id	Access-Request, CoA-Request, Disconnect-Request
Called-Station-Id	Access-Request
Acct-Session-Id	Access-Accept, CoA-Request, Disconnect-Request
Service-Type	Access-Request
Error-Cause	CoA-NAK, Disconnect-NAK
Session-Timeout	Access-Accept, CoA-Request
Termination-Action	Access-Accept, CoA-Request
Acct-Interim-Interval	Access-Accept, CoA-Request
Filter-Id	Access-Accept, CoA-Request
NAS-Filter-Rule	Access-Accept, CoA-Request
Tunnel-Type	Access-Accept, CoA-Request
Tunnel-Medium-Type	Access-Accept, CoA-Request
Tunnel-Private-Group-ID	Access-Accept, CoA-Request
WISPr-Redirection-URL	Access-Accept, CoA-Request
WISPr-Bandwidth-Max-Up	Access-Accept, CoA-Request
WISPr-Bandwidth-Max-Down	Access-Accept, CoA-Request
Cisco-AV-Pair: url-redirect	Access-Accept, CoA-Request

The `NAS-Filter-Rule` attribute can contain any IP filter conforming to the `IPFilterRule` format as defined in RFC 6733. In addition the following list of `Filter-Id` shorthand values are recognized by the Gateway.

Table 2: Supported `Filter-Id` values and their equivalent `NAS-Filter-Rule`.

Value	Interpretation
`block`	`permit in 17 from any to any 53,67` `deny in ip from any to any`

Dynamic IEEE 802.1Q VLAN assignment as defined in RFC 3580 is supported.

Table 3: Supported `Tunnel-Type` values.

Value	Interpretation
`13`	`Tunnel type is Virtual LAN (VLAN)`

Table 4: Supported `Tunnel-Medium-Type` values.

Value	Interpretation
`6`	`Tunnel medium type is IEEE Std 802`

Table 5: Supported `Tunnel-Private-Group-ID` values.

Value	Interpretation
`1-4096`	`IEEE Std 802.1Q VLAN tag`

IEEE 802.1Q VLAN assignment as well as access restrictions expressed through `Filter-Id`, `NAS-Filter-Rule`, `WISPr-Redirection-URL`, `WISPr-Bandwidth-Max-Up` and `WISPr-Bandwidth-Max-Down` attributes can be changed at any time by sending a `Change-of-Authorization (CoA) Request` to the Gateway as specified in RFC 5176.

Authorization

The Gateway can also use an external RADIUS server to limit access to an open or pre-shared key (PSK) authenticated network. In order to distinguish authorization from EAP authentication the Gateway will set the `Service-Type` attribute in the `Access-Request` message to `Call Check`.

All RADIUS message types and attributes listed above are supported for authorization. However, since the EAP identity of the *client* is not known the Gateway will instead set the `User-Name` attribute of the `Access-Request` message to the same value as `Calling-Station-Id`, i.e. to the *client*'s MAC address. Since the RADIUS server typically uses this MAC address to determine the authorization of the *client* other vendors sometimes refer to this functionality as "MAC authentication". A RADIUS server

configured for such "MAC authentication" should not require any changes to interoperate with the Gateway.

Once the RADIUS server has determined that a *client* is authorized to access the network it should reply with an `Access-Accept` message, optionally with access restrictions expressed e.g. through attribute values. Alternatively the RADIUS server can reply with an `Access-Reject` message, thereby refusing the *client* access to the network. In this latter case the device will be forcefully disassociated from the network on the IEEE 802.11 level. The RADIUS server can also change the authorization status of a connected client at any time by sending a `Disconnect-Request` or `CoA-Request` message to the Gateway as defined in RFC 5176.

Accounting

The following RADIUS accounting message types are supported:

- `Accounting-Request`
- `Accounting-Response`

The following table lists the supported RADIUS accounting message attributes, as well as the message types that may contain them.

Table 6: Supported RADIUS accounting attributes.

Attribute	Message types
`Message-Authenticator`	`all`
`NAS-IP-Address`	`Accounting-Request`
`NAS-Port-Type`	`Accounting-Request`
`User-Name`	`Accounting-Request`
`Chargeable-User-Identity`	`Accounting-Request`
`Calling-Station-Id`	`Accounting-Request`
`Called-Station-Id`	`Accounting-Request`
`Acct-Session-Id`	`Accounting-Request`
`Acct-Status-Type`	`Accounting-Request`
`Acct-Session-Time`	`Accounting-Request`
`Acct-Input-Packets`	`Accounting-Request`
`Acct-Output-Packets`	`Accounting-Request`
`Acct-Input-Octets`	`Accounting-Request`

Acct-Output-Octets	Accounting-Request
Acct-Input-Gigawords	Accounting-Request
Acct-Output-Gigawords	Accounting-Request

GRE for User Payload

The preferred interface for integration towards a WLAN gateway is native Layer 2 bridged Ethernet. But the Vyatta Network OS also supports GRE tunneling of bridged Ethernet frames.

Enter configuration mode	vyatta@vyatta:~$ **configure** [edit]
Configure GRE tunnel in bridge mode	vyatta@vyatta# **edit interfaces tunnel tun0** [edit interfaces tunnel tun0] vyatta@vyatta# **set encapsulation gre-bridge** [edit interfaces tunnel tun0] vyatta@vyatta# **set local-ip *x.x.x.x*** [edit interfaces tunnel tun0] vyatta@vyatta# **set parameters ip bridge-group bridge br0** [edit interfaces tunnel tun0] vyatta@vyatta# **set remote-ip *x.x.x.x*** [edit interfaces tunnel tun0] vyatta@vyatta# **top** [edit]
Commit, save and exit configuration mode	vyatta@vyatta# **commit** vyatta@vyatta# **save** vyatta@vyatta# **exit**

SNMP for System Monitoring

The Vyatta Network OS supports system monitoring through the Simple Networking Monitoring Protocol (SNMP) versions v2c and v3. Configure the system for remote monitoring with the following commands.

Enter configuration mode	vyatta@vyatta:~$ **configure** [edit]
Configure SNMP v2c read-only with traps	vyatta@vyatta# **edit service snmp** [edit service snmp] vyatta@vyatta# **set listen-address *x.x.x.x*** [edit service snmp] vyatta@vyatta# **set trap-source *x.x.x.x*** [edit service snmp]

	vyatta@vyatta# **set community *name* authorization ro** [edit service snmp] vyatta@vyatta# **set community *name* client *x.x.x.x*** [edit service snmp] vyatta@vyatta# **set trap-target *x.x.x.x* community *name*** [edit service snmp] vyatta@vyatta# **set location *location*** [edit service snmp] vyatta@vyatta# **set contact *contact*** [edit service snmp] vyatta@vyatta# **top** [edit]
Commit, save and exit configuration mode	vyatta@vyatta# **commit** vyatta@vyatta# **save** vyatta@vyatta# **exit**