
CERN

Using Configuration Management to deploy and manage network services

Quentin Barrand
CERN IT / Communication Services

Agenda

- Which network services ?
- How we used Puppet
- The new setup
- Questions & feedback

RADIUS services for Wi-Fi

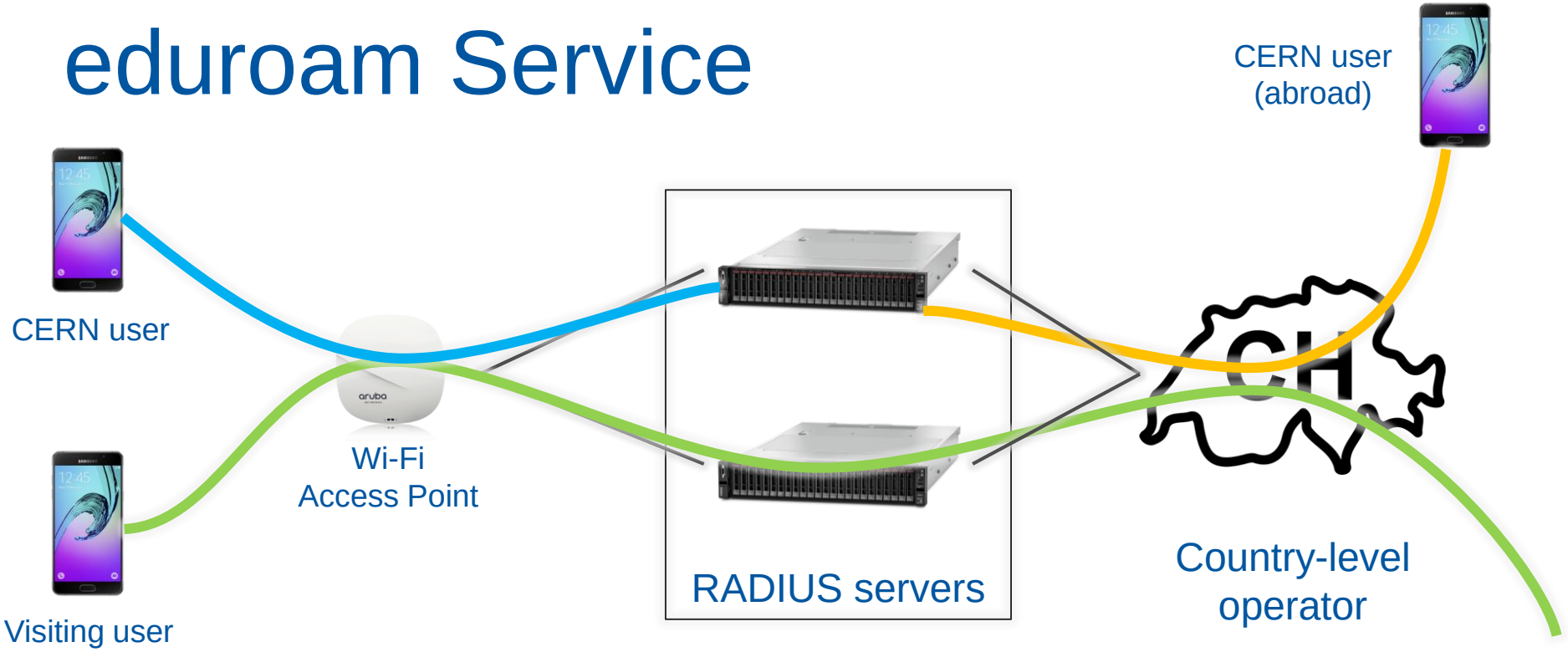
eduroam

- EAP-TLS authentication for CERN users
- Proxy to other institutes
- Integration into our device database

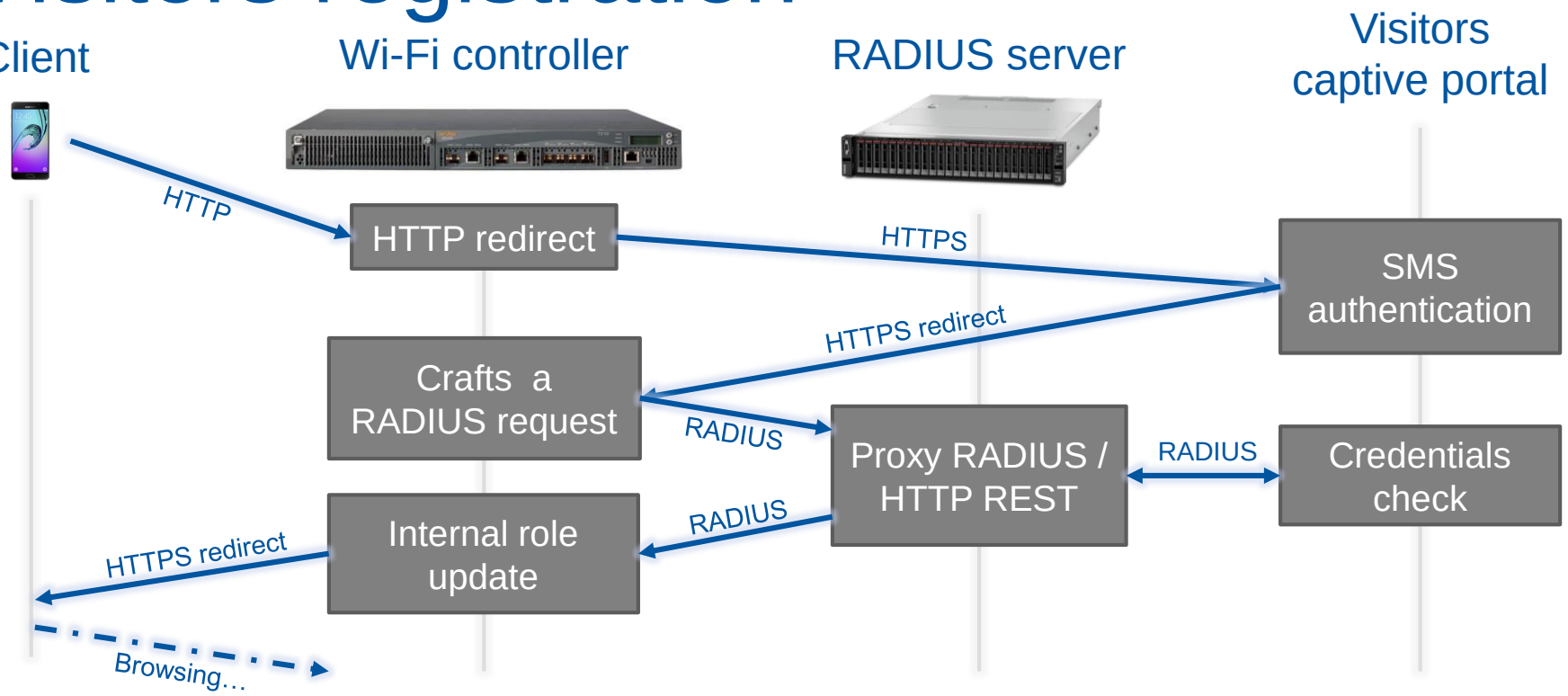
NEW **Visitors**

Key middleware for the new Visitors Wi-Fi service

eduroam Service



Visitors registration



Historical setup

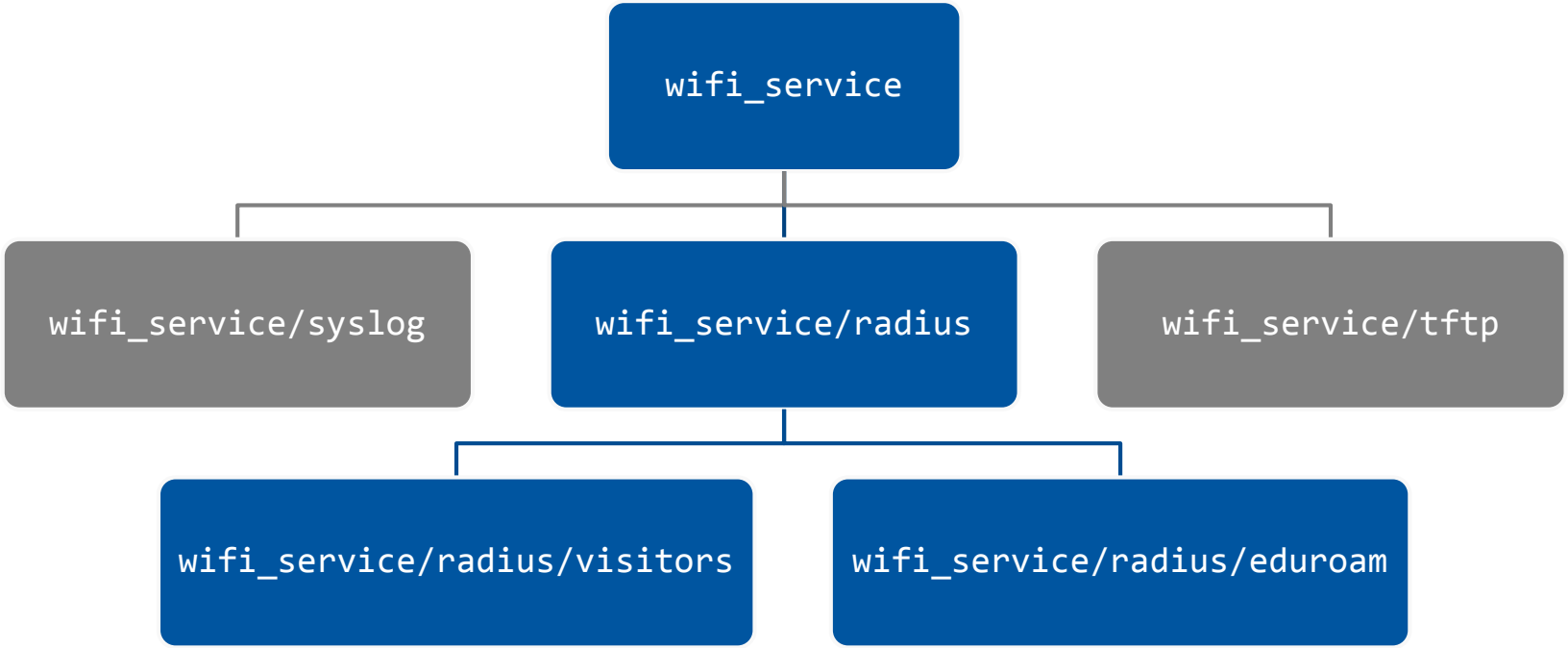
- Two physical SLC6 servers per service
  - Critical power room in Geneva
  - Also used for other services…
- Everything manual
  - FreeRADIUS compilation and installation
  - Replication of configuration changes
- **Upgrade or reinstallation is risky and painful**

What do we need ?

Visitors

Recent FreeRADIUS	Recent FreeRADIUS
FreeRADIUS REST back-end	FreeRADIUS TLS back-end
	FreeRADIUS Oracle back-end
`clients.conf` updater	`clients.conf` updater
Monitoring script	Monitoring script
That's all !	

Leveraging Foreman's hostgroups



Installing FreeRADIUS

- Yay, there's a CentOS package ! ☺
  - ... but the release is two years old ☹
  - The certificate verification feature is broken in that release
  - We depend on Red Hat for security fixes
- There are Puppet modules for FreeRADIUS ! ☺
  - … they all rely on the package ☹

- **So, let's compile it ourselves**
  - Create custom facts
  - Write a dedicated class that rebuilds the server if needed
  - Specify the requirements via Hiera

Installing FreeRADIUS

Want to upgrade FreeRADIUS ?

```
hg_wifi_service::radius::fr_version: '3.0.14' ➔ '3.0.15'
```

Need the FreeRADIUS REST module ?

```
hg_wifi_service::radius::fr_required_modules:
  - rlm_rest
hg_wifi_service::radius::fr_dependencies:
  - libcurl-devel
  - json-c-devel
```

`./configure` cannot find the Oracle headers !

```
hg_wifi_service::radius::fr_configure_args:
  - --with-oracle-include-dir=/usr/include/oracle/12.2/client64/
```

A few management applets

`radius-clients-updater`	Updates FreeRADIUS `clients.conf`
`eduroam-monitoring`	Test many authentication scenarios for their respective services
`visitors-monitoring`	

- Less than 2000 SLOC each !
- Written in Python
  - First 2.7, now 3.5 (via SCL)
  - Very large ecosystem of public libraries
  - Reuse existing tools developed for the Wi-Fi project

How to install Python scripts ?

The "trivial" way

- Copy the scripts as static resources
- Install `python-*` with `yum`
- Manipulate `$PYTHONPATH` to use shared modules

✗ Forget about testing and CI

A more agile way

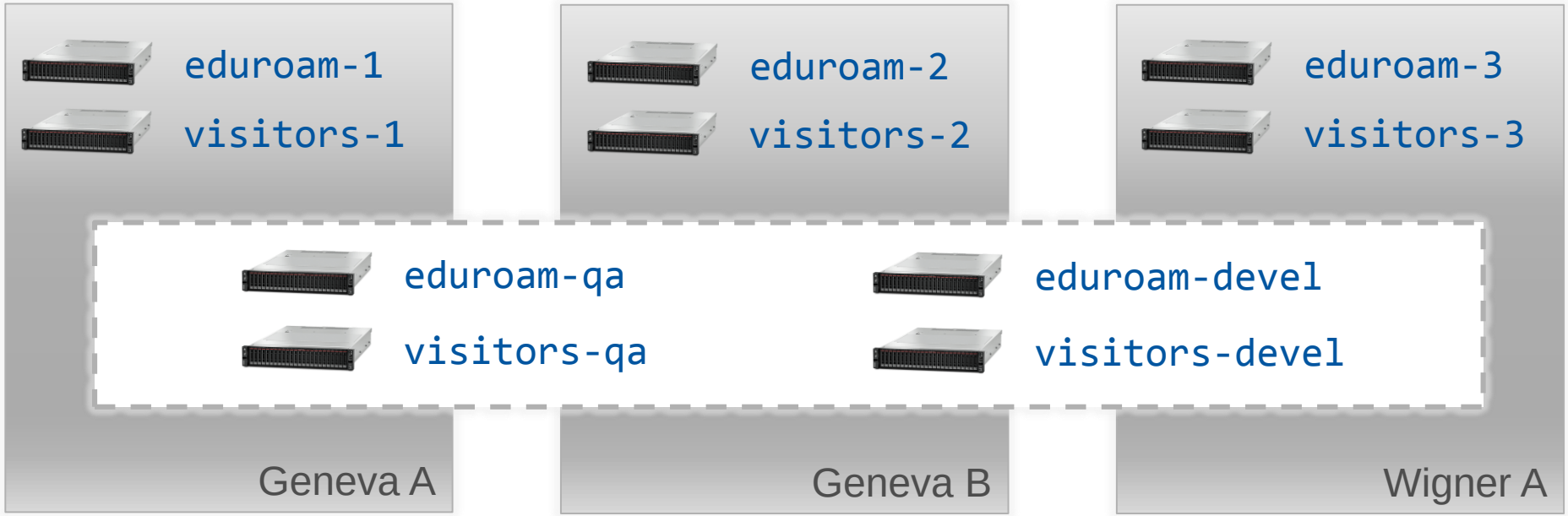
- Write **distributable** packages
  - `setuptools`
  - `install_requires`
- Install them in **virtual environments** using `puppet-python`

✓ Test after each commit

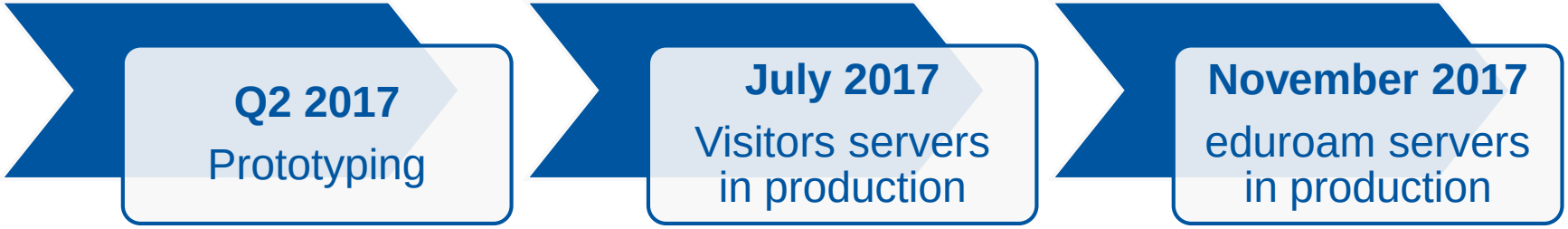
The new setup

- IT department's policy:
  - All services on VMs
  - Ensure reliability by balancing across OpenStack Availability Zones
- Five nodes for each service:
  - 3 production
  - 1 quality assurance
  - 1 development
- CentOS 7.4
- FreeRADIUS 3.0.14

The new setup: VMs only



Status and plans



- Other services already Puppet-managed
  - Syslog collector
  - TFTP server that collects AP crash dumps
  - … more coming !

Wrap-up

- Puppet makes services more manageable and resilient
  - New nodes can be deployed in less than an hour
  - Improve reliability by adding more nodes vs. using critical power servers
- We are gaining experience with two (relatively) small services
  - Compile software ourselves to avoid surprises
  - **Products / packages** are easier to maintain than **bare scripts**
  - **Modernizing** and **Puppetizing** a service are different tasks
- eduroam will be easier to troubleshoot than ever before !
  - Redirect failing authentication requests to a debug server
  - Stage the rollout of new FreeRADIUS releases

Thank you !
ありがとう

Questions, remarks & thoughts


CERN

Configuration as software

Our simplest manifest

```
class hg_wifi_service::syslog {

  $package_name = 'syslog-ng'
  $service_name = 'syslog-ng'

  $conf_d = '/etc/syslog-ng/conf.d'

  # Install the TFTP server
  package {$package_name: ensure => installed}

  # Install the configuration files
  -> file {$conf_d:
    ensure  => directory,
    source  => 'puppet:///modules/hg_wifi_service/syslog/conf.d',
    recurse => true,
    purge   => true,
    notify  => Service[$service_name],
  }

  # Add the firewall rule
  -> firewall {'100 allow Syslog':
    ensure => present,
    action => 'accept',
    dport  => '514',
    proto  => 'udp',
    state  => 'NEW',
  }

  # Start the service
  -> service {$service_name:
    ensure => running,
    enable => true,
  }
}
```