Foreman | Lifecyclemanagement for Server

Version: 1.0

Official Foreman Training created by NETWAYS GmbH

© NETWAYS

Table of Contents

© NETWAYS

1 Introduction

© NETWAYS

Training

This training course will introduce the architectual concepts of Foreman, will cover installation, usage and administration. The configuration management solution used will be Puppet.

- Foreman architecture
- Installation
- Provisioning
- Configuration management
- User management
- Advanced Use

This training course will introduce the architectual concepts of Foreman, will cover installation, usage and administration. The configuration management solution used will be Puppet.

Foreman architecture

This chapter will give you a basic understanding of the architecture of Foreman, its Smart proxies and Compute Resources.

continued...

Installation

In this chapter you will get an idea on the different setups you can create, what is required for running foreman, how the installer works and will create an All-in-one setup including DNS and DHCP in preparation for unattended installations.

Provisioning

This chapter provides basic knowledge on automated installation mechnism for Linux. You will learn how to configure them in the Foreman Web GUI. Afterwards several installation scenarios will be covered from bare metal installation using PXE over creating virtual machines using Compute Resources and metal as a service with Foreman's Discovery Plugin to creating a setup using installation media instead of PXE.

Configuration management

Some basic explainations will give you enough knowledge to understand how Puppet works and to use it to configure newly provisioned systems to your needs. Some existing Puppet modules will be imported, parameterized and assigned to the systems directly or via a host group. Configuration runs will be reported back to Foreman so the administrator can see success and failure.

continued...

© NETWAYS

User management

To give more users access to the Foreman Web GUI LDAP authentication will be added. Privileges will be added to the new users. Also the auditing capabilities of Foreman will be covered so you can see what your new users do.

Advanced Use

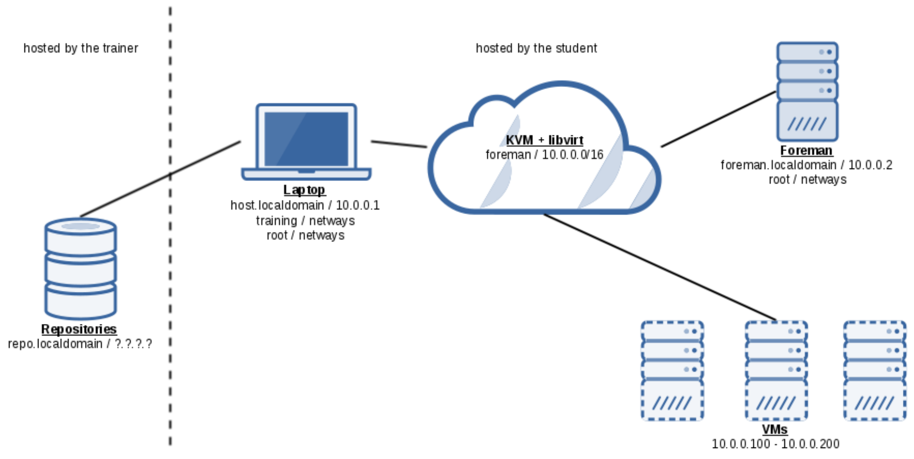
Some advanced topics like debugging, clean up, backup and restore will be covered in this chapter. Also the usage of the API and CLI will be shown. Furthermore additional plugins helpful in some environments will be introduced. And last but not least the trainer will show you Katello. This project adds content and subscription management to Foreman and is the upstream project for Red Hat Satellite 6.

Training Hints

- Ask questions! Dumb questions do not exist.
- Add your own notes! Having learnt something is reflected best with your own additions.
- Join the conversation! Many of these exercises require real world scenarios and your feedback and questions.
- Work together! Be part of the training team and find solutions together.
- Additional documentation in your browser at http://theforeman.org/manuals/latest

© NETWAYS

Training Environment



The laptop provided for the training is running CentOS 7 with Gnome 3 in Fallback mode. You can login with the unprivileged user "training" and password "netways". The password for user "root" required for some exercises is also "netways".

For virtualization the laptop runs KVM with libvirt. A virtual network named "foreman" is configured with the IP address "10.0.0.1" assigned to the laptop and a host entry "host.localdomain", the IP address "10.0.0.2" is assigned to an already existing VM used to install Foreman.

continued...

© NETWAYS

The virtual machine is named "foreman.localdomain" and allows login via SSH with user "root" and password "netways". Foreman will be installed on it including DNS and DHCP service. This system also runs a LDAP service which will be required for some exercises.

Additional VMs will be deployed in the address range of "10.0.0.100" to "10.0.0.200" using the repositories hosted by the trainer. He will give you information about the address to connect. Please do not use the upstream repositories, it will slow down the performance!

2 Foreman Architecture

© NETWAYS

Foreman Project

- Open Source Project
  - Homepage: http://theforeman.org
  - Initial release: 10 September 2009
  - Written in Ruby and Javascript
- Lifecyclemanagement for servers
  - provisioning
  - configuration
  - orchestration
  - monitoring
- Web based GUI
- Utilizes other tools via smart proxy

Foreman was originally writen by Paul Kelly and Ohad Levy and initially released on 10 September 2009.

Now it is sponsored by Red Hat who use it as base for many of their Enterprise solutions like Red Hat Network Satellite and Red Hat OpenStack distribution.

continued...

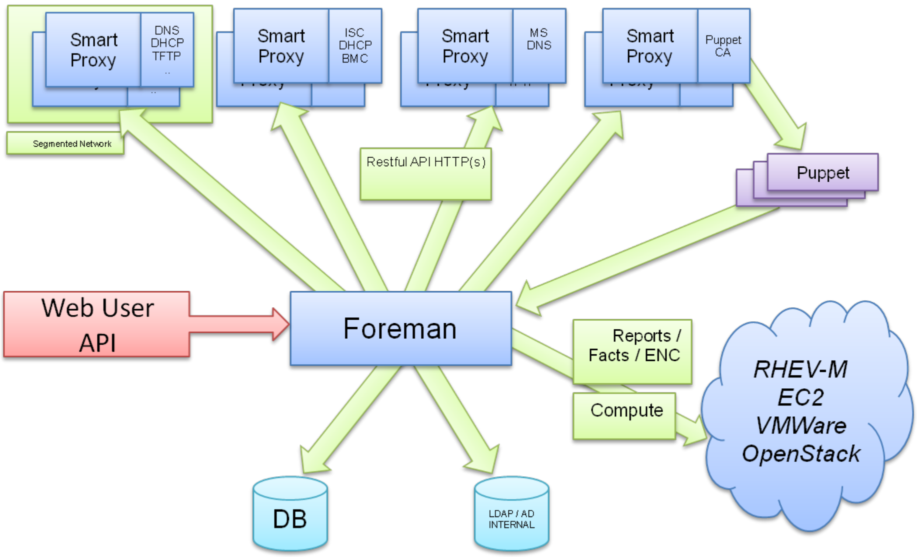
© NETWAYS

It is writen in Ruby and Javascript to provide a Web based GUI for server lifecyclemanagement, from provisioning and configuration to orchestration and monitoring. For integration in the IT infrastructure it utilizes other tools via its smart proxy architecture.

The project provides an open bug tracker for bug reporting and feature requests, its code on github, IRC support to help the community and consists of more than 200 contributors.

© NETWAYS

Architecture

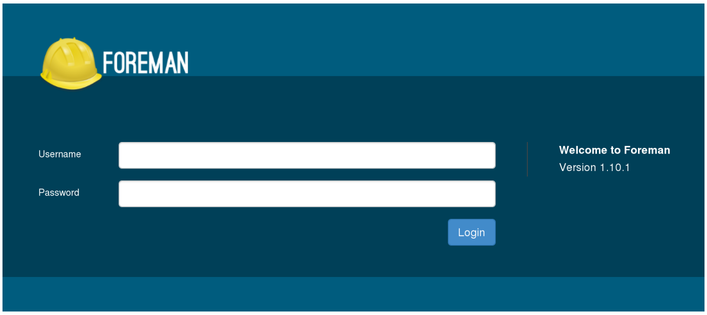


Foreman provides a really flexible architecture and many optional components. Only required component is the central Foreman installation, but depending on your needs it can manage every service required to fully automate the provisioning process. Basicly we have two kinds of additional components Smart proxies and Compute resources.

Image copyright by the Foreman project.

Foreman

- Web Interface + API + CLI
- Supported plattforms:
    - Server: RHEL/Fedora, Debian/Ubuntu, (Linux)
    - DB: PostgreSQL, MySQL, SQLite
    - Provisioning: RHEL/Fedora, Debian/Ubuntu, Solaris, SuSE, CoreOS, FreeBSD, Juniper Junos, Cisco NX-OS, (Windows, MacOS)
- Usermanagement:
    - Users and Groups
    - Internal / LDAP / Kerberos (via Apache Authentication)
    - Fine role based privileges



Welcome to Foreman
Version 1.10.1

Foreman provides in addition to the Web Interface an API and CLI.

It supports as plattform RHEL/Fedora and Debian/Ubuntu via packages and via installation from source also other Linux distribution.

As database backend PostgreSQL, MySQL and SQLite are supported, while the default is PostgreSQL, SQLite is not recommended for productive usage.

The following operating systems are known to successfully install from Foreman:
* RHEL/Fedora
* Debian/Ubuntu
* Solaris
* SuSE
* CoreOS
* FreeBSD
* Juniper Junos
* Cisco NX-OS

Also reported by the community are:
* Windows
* MacOS

© NETWAYS

Smart Proxy

- Autonomous web-based component
    - Restful API to connect to various systems from Foreman

- Supported Plattforms: RHEL/Fedora, Debian/Ubuntu, (Linux, Windows)

- Supported Subsystems:
    - DHCP - ISC DHCP, MS DHCP Servers, Libvirt
    - DNS - Bind, PowerDNS, Route53, MS DNS Server, Libvirt
    - BMC - IPMI
    - Puppet & Puppet CA / Salt / Chef
    - Realm - FreeIPA
    - TFTP

The smart proxy is an autonomous web-based component providing a restful API to connect to varios systems from higher ochestration tools such as Foreman.

The Project provides packages for installation on RHEL/Fedora and Debian/Ubuntu. Installing from source allows to support other Linux distributions and also Windows which is required for some implementations of subsystems.

continued...

© NETWAYS

There are different implementations of various subsystems included in the smart proxy by default and it easily allows to add additional subsystems and implementations as plugins. For configuration management the solutions differ to much to be covered by one subsystem. For joining a realm support for active directory is in development but for now only FreeIPA (an Open Source aquivalent to Active Directory focused on Linux) is supported.

© NETWAYS

Smart Proxy - DHCP

- Adds and removes host reservations
- Requires subnets to be configured
- Supports: ISC DHCP, MS DHCP, Libvirt
- ISC DHCP:
  - uses OMAPI
- MS DHCP:
  - uses netsh on a windows server
  - needs administrative privileges
- Libvirt:
  - not for productive use
  - uses virsh to manage dnsmasq for libvirt

The Smart Proxy DHCP is used to add and remove host reservations to preconfigured subnets and allows to import them to foreman.

continued...

© NETWAYS

On Linux typically a ISC compatible implementation is used to manage DHCP which allows to send commands via OMAPI. For Microsoft DHCP installation of the Smart Proxy on Windows Server system is required which needs netsh command installed and the user running the service needs administrative privileges, but the server does not need to be the DHCP server. For testing enviroments also an implementation for Libvirt using virsh to manage the dnsmasq underneath is available, a productive use is not recommended.

Smart Proxy - DNS

- Adds and removes dns records of type A and PTR
- Requires zone to be configured as dynamic zones
- Supports: Bind, PowerDNS, Route53, MS DNS, Libvirt
- Bind:
  - uses nsupdate with preshared key or Kerberos principal
- MS DNS:
  - uses nsupdate with Kerberos principal
  - alternativly: uses dnscmd on a windows server
- PowerDNS:
  - directly connects to the database backend
- Route53 (DNS in AWS cloud):
  - requires Amazon Web Service account
  - access and secret key of IAM account with access to Route53
- Libvirt:
  - not for productive use
  - uses virsh to manage dnsmasq for libvirt

© NETWAYS

The Smart Proxy DNS is used to add and remove dns records of type A and PTR, at the moment there is no support for IPv6 and additional records. For doing such updates it requires the zone to be a dynamic zone.

Commonly used on Linux is Bind which takes updates via nsupdate with preshared keys or if used in FreeIPA with Kerberos principal. The same mechanism could be used for sending updates to Microsofts DNS. Another possibilty is to install the Smart Proxy on a Windows server and give it the privileges to run dnscmd. Other implementations like PowerDNS and Route53 are also supported. Libvirt is again only supported as a testing environment.

© NETWAYS

Smart Proxy - TFTP

- Used to provide boot-images for PXE boot
- Downloads the required files with wget
- Creates and deletes PXE configuration based on MAC address
- Automatically configured during installation

The Smart Proxy TFTP provides boot-images for PXE boot, these files are simply downloaded using wget the first time needed. PXE configuration is created during provisioning based on MAC address. For this subsystem in the most cases no manual configuration is required because it is completly included in the basic setup.

© NETWAYS

Smart Proxy - Puppet / Puppet CA

- Puppet:
  - connects to Puppets API
  - allows to import puppet environments and classes
  - optionally triggers immediate Puppet runs
  - accepts facts and reports
  - uses Foreman as ENC

- Puppet CA:
  - requires access to ssl directory, autosign configuration and puppet cert command
  - allows certificate management using the Web GUI
  - creates autosign entry for hosts during provisioning

The Smart Proxy Puppet connects to the API of Puppet to query puppet environments and classes for import. It optionally triggers immediate Puppet runs using the deprecated puppet kick command, mcollective, executing puppet agent command via ssh on the remote system, salt or any custom script.

continued...

© NETWAYS

Also it automatically allows a Puppet master known to Foreman as Smart Proxy to upload facts and reports. In addition the Puppet master can access Foreman as an External Node Classifier to build its catalog.

The Smart Proxy Puppet CA is independent from the one for Puppet. It requires access to Puppet's ssl directory, the autosign configuration and puppet cert command via sudo. The Web GUI utilizes the Smart Proxy for certificate management and creates autosign entry for hosts during provisioning for accessing puppet without manual intervention.

© NETWAYS

Compute Resource

- Plugin to create and manage virtual hosts
  - based on fog (ruby cloud service library)

- Depending on virtualization provider
  - Unattended / Image-based installation
  - Console access
  - Power management
  - Changing virtual machine configuration

- Compute Profiles - predefined template for virtual hardware

Compute Resource refers to a virtualization or cloud service which is integrate into Foreman as a plugin. Those plugins are based on the ruby cloud service library named fog and allows to create and manage virtual hosts. Depending on the provider machines are installed from an image or unattended in the same fashion like a bare metal host. It allows to access the console and do power management operations like shutting down or resetting the system. Some providers also allow to change the virtual hardware afterwards.

continued...

© NETWAYS

The table shows the providers directly supported by the Foreman Project and its capabilities, additional providers like OpenNebula are community contributed.

Provider	Package	Unattended installation	Image-based	Console	Powe managei
EC2	foreman-ec2	no	yes	read-only	yes
Google Compute Engine	foreman-gce	no	yes	no	yes
Libvirt	foreman-libvirt	yes	yes	VNC or SPICE	yes
OpenStack Nova	foreman-compute	no	yes	no	yes
oVirt / RHEV	foreman-ovirt	yes	yes	VNC or SPICE	yes
Rackspace	foreman-compute	no	yes	no	yes
VMware	foreman-vmware	yes	yes	VNC	yes

Compute Profiles allow to predefine the virtual hardware as a template to select for provisioning.

© NETWAYS

3 Installation

© NETWAYS

Requirements - Operating System

- Red Hat Enterprise Linux 6 & 7
  - EPEL repository
  - Optional and RHSCL Channel
  - optionally: Puppetlabs Repository
- CentOS, Scientific Linux, Oracle Linux 6 & 7
  - EPEL repository
  - optionally: Puppetlabs Repository
- Fedora (not recommended)
- Debian 7
  - optionally: Puppetlabs Repository
- Debian 8
  - optionally: Puppetlabs Repository
- Ubuntu 12.04 & 14.04
  - optionally: Puppetlabs Repository

continued...

On the mentioned operating systems packages are provided by the project, a installation from source is not recommended. On all plattforms all updates should be applied before installation. Using the Puppetlabs Repository providing an up-to-date version of Puppet is preferred.

Requirements - Puppet & Facter

Puppet version	Foreman installer	Smart proxy	Report/fact processors	ENC
0.25.x	Not supported	Untested	Untested	No Parametrized Classes
2.6.0 - 2.6.5	Not supported	Untested	Untested	No Parametrized Classes
2.6.5+	Not supported	Supported	Supported	Supported
2.7.x	Supported	Supported	Supported	Supported
3.0.x	Limited support	1.1 or higher	Supported	Supported
3.1.x - 3.4.x	1.1 or higher	1.1 or higher	Supported	Supported
3.5.x	1.4.3 or higher	1.4.2 or higher	Supported	Supported
3.6.0+	1.4.3 or higher	1.5.1 or higher	Supported	Supported
4.x	Not supported	Partial support	Untested	Untested

continued...

© NETWAYS

- Puppet Enterprise is not supported
- Facter 1.x is supported, 2.x requires Foreman >= 1.4.2, Structured Facts are not supported for now

---

An up-to-date version of Puppet 3.x is recommended while other versions will work. Puppet 4.x support is pending. Puppet Enterprise is not supported, but can work with manual tweaking of the setup.

Facter 1.x is supported, Facter 2.x is supported by requires at least Foreman 1.4.2. Support for structured facts provided by Facter 2.x is not supported for now, feature request is pending.

© NETWAYS

Requirements - Communication

- Port matrix (depending on installation)

Port	Protocol	Required For
53	TCP & UDP	DNS Server
67, 68	UDP	DHCP Server
69	UDP	TFTP Server
80, 443	TCP	HTTP & HTTPS access to Foreman web UI - using Apache + Passenger
3000	TCP	HTTP access to Foreman web UI - using standalone WEBrick service
3306	TCP	Separate MySQL database
5910 - 5930	TCP	Server VNC Consoles
5432	TCP	Separate PostgreSQL database
8140	TCP	Puppet Master
8443	TCP	Smart Proxy, open only to Foreman

continued...

© NETWAYS

Depending on your installation the ports above are required to be accessable on the Foreman server, by Foreman and the managed systems.

© NETWAYS

Foreman Installer

- Separate project named kafo
- Recommended way of installation
- Utilizes Puppet modules to install and configure
  - Foreman web UI
  - Smart Proxy
  - Passenger (for Puppet master and Foreman)
  - TFTP
  - DNS
  - DHCP
- Runs parameterized or interactive

The Foreman installer is a separate project named kafo (Katello/Foreman Installer) which could also be used by other projects. It utilizes existing Puppet modules to install and configure all required components. Module parameter are provided as commandline arguments to the installer or by running the installer in interactive mode.

continued...

© NETWAYS

In a default installation this would be:

- Apache HTTP with SSL (using a certificate signed by Puppet CA)
- Foreman running under mod_passenger
- Puppet master running under mod_passenger
- TFTP server (on RHEL and derivates under xinetd)
- Smart Proxy using SSL configured for Puppet and TFTP

Depending on provided parameters other modules are enabled:

- ISC DHCP server
- Bind DNS server

Scenarios

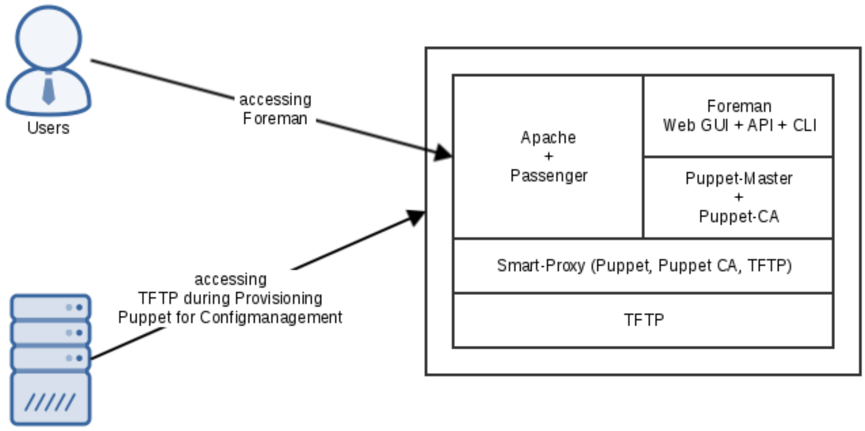
- All-in-one
- All-in-one with additional Puppet masters
- Separate Puppet masters
- Smart proxies
- Integration of PuppetDB

The Foreman installer allows to setup different scenarios depending on its parameters or answers in interactive mode. All the scenarios above are explained in the course material, commands required can be found in the Foreman manual on the project homepage. For the training we will stick to the default all-in-one setup and add additional Smart proxies.

© NETWAYS

All-in-one

- Default setup on unparameterized run



Users
accessing Foreman
accessing TFTP during Provisioning Puppet for Configmanagement
Apache + Passenger
Foreman Web GUI + API + CLI
Puppet-Master + Puppet-CA
Smart-Proxy (Puppet, Puppet CA, TFTP)
TFTP

- Expandable with additional plugins and Smart proxies

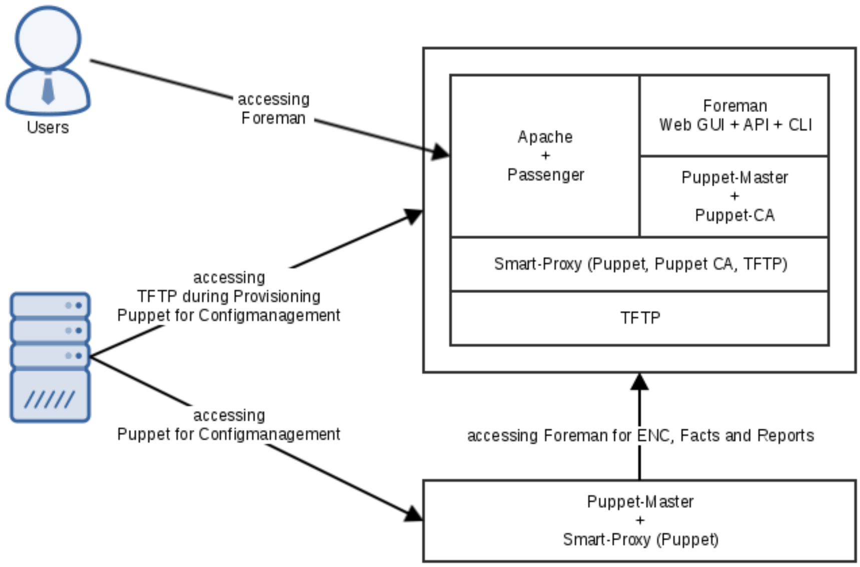
The Foreman installer by default installs an all-in-one scenario with Apache httpd and Passenger serving the Foreman Web GUI and API and the Puppet Master including the certificate authority. In addition it installs and configures TFTP and the Smart Proxy covering the installed components Puppet, Puppet CA and TFTP.

Additional plugins including Compute Resources could be enabled setting the corresponding parameter to true. Smart Proxies can also be added to the system hosting Foreman or on different machines which will be covered later.

© NETWAYS

All-in-one with additional Puppet masters

- Default setup on Foreman host
- Precreate certificates on Puppet CA
- Run installer with parameters to disable Foreman on additional Puppet Masters


Users
accessing Foreman
Apache + Passenger
Foreman Web GUI + API + CLI
Puppet-Master + Puppet-CA
Smart-Proxy (Puppet, Puppet CA, TFTP)
TFTP
accessing TFTP during Provisioning Puppet for Configmanagement
accessing Puppet for Configmanagement
accessing Foreman for ENC, Facts and Reports
Puppet-Master + Smart-Proxy (Puppet)

continued...

For the Foreman host run the same setup like before afterwards create certificates for the Puppet Masters to be added. If you do not do so, it will create an additional CA on this systems. Last run the Foreman installer on the systems with parameters to disable Foreman and the Puppet CA.

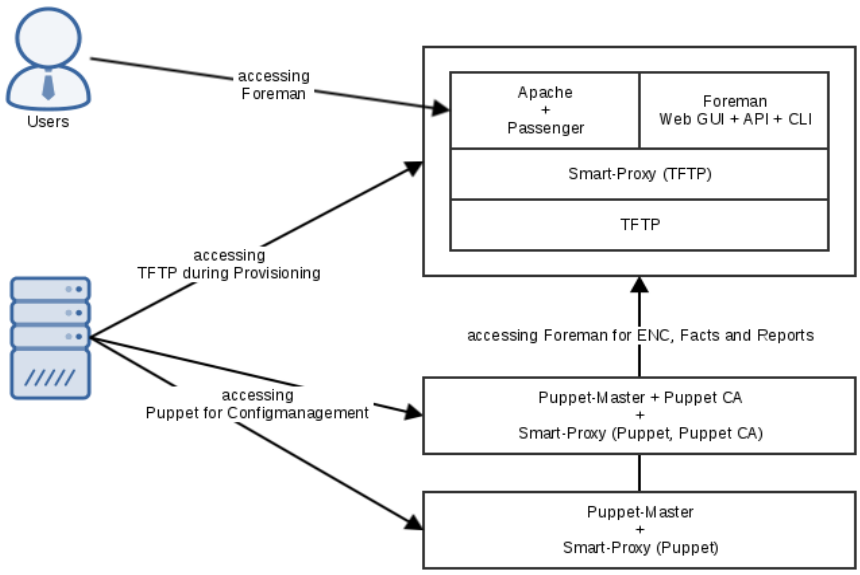
Depending on your infrastructure there are different ways to balance the load on the Puppet Masters:

- Manual point servers to a Puppet Master
- Use Round-robin DNS which requires your Puppet Master certificates to be created with an DNS alias
- Use a Loadbalancer solution which depending on SSL handling also requires DNS aliases
- Utilize DNS SRV records which requires Puppet 3 and appropriate agent configuration

© NETWAYS

Separate Puppet masters

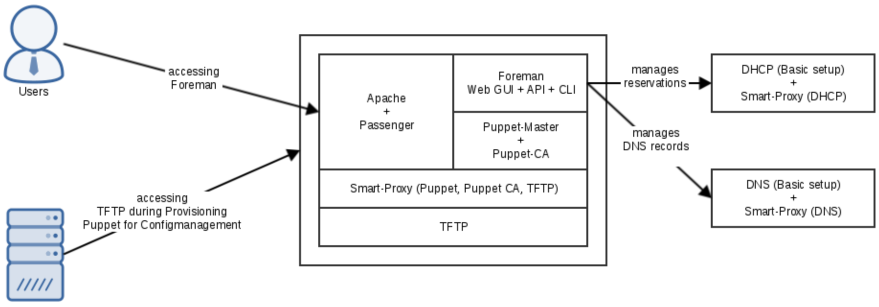
- Setup on Foreman host with Puppet Master and Smart Proxies disabled
- Run installer with parameters to disable Foreman on separate Puppet Masters


Users
accessing Foreman
Apache + Passenger
Foreman Web GUI + API + CLI
Smart-Proxy (TFTP)
TFTP
accessing TFTP during Provisioning
accessing Foreman for ENC, Facts and Reports
accessing Puppet for Configmanagement
Puppet-Master + Puppet CA + Smart-Proxy (Puppet, Puppet CA)
Puppet-Master + Smart-Proxy (Puppet)

This is nearly the same setup as before but removes the load created by being a Puppet Master from the Foreman server. You still have to remember to create only the first Puppet master as a certificate authority and setup a solution to load balance the agents.

Smart Proxies

- Foreman installer helps to setup Smart Proxies


Users
accessing Foreman
accessing TFTP during Provisioning Puppet for Configmanagement
Apache + Passenger
Foreman Web GUI + API + CLI
Puppet-Master + Puppet-CA
Smart-Proxy (Puppet, Puppet CA, TFTP)
TFTP
manages reservations
manages DNS records
DHCP (Basic setup) + Smart-Proxy (DHCP)
DNS (Basic setup) + Smart-Proxy (DNS)

The Foreman installer can install a Smart Proxy and register it automatically to the Foreman instance. This could be done on the Foreman host itself or any other host communication is possible. This will require the certificate to be created in advance to allow the communication. For authentication it uses OAuth provided by Foreman available to the administrators.

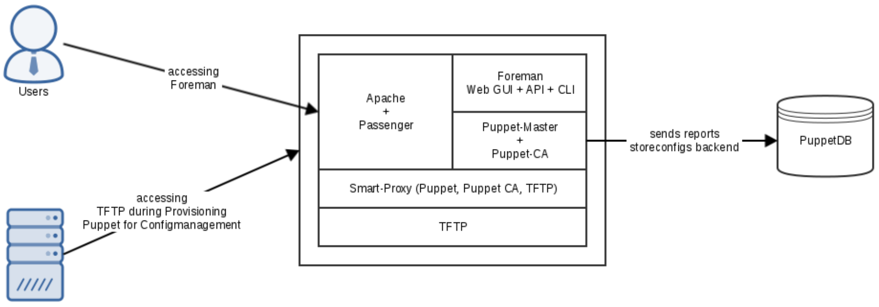
It also can install and configure the corresponding service like Bind for DNS, ISC DHCP or TFTP but this is a very basic setup and will typically require some additional configuration.

After the Smart proxy is registered to Foreman import of configured data like subnets or manual configuration is required.

© NETWAYS

Integration of PuppetDB

- Foreman installer helps to integrate PuppetDB



Users
accessing Foreman
accessing TFTP during Provisioning Puppet for Configmanagement
Apache + Passenger
Foreman Web GUI + API + CLI
Puppet-Master + Puppet-CA
Smart-Proxy (Puppet, Puppet CA, TFTP)
TFTP
sends reports storeconfigs backend
PuppetDB

The Foreman installer allows to add PuppetDB as reporting target and storeconfigs backend to the Puppet Masters. It does not setup PuppetDB server!

PuppetDB is not required in a typical setup with Foreman. Instead of using storeconfigs or PuppetDB queries Foreman's own database could also be used for queries. The required function is part of the Puppet module "foreman" provided by the Foreman Project.

Installation of PuppetDB if required for your setup could easily be done with the Puppet module "puppetdb" provided by Puppetlabs.

© NETWAYS

Lab 3.1: Prepare Installation

- Objective:
    - Prepare the installation of Foreman

- Steps:
    - Make Puppetlabs repository available
    - Make EPEL repository available
    - Make Foreman repository available
    - Install foreman-installer

© NETWAYS

Lab 3.2: Install an All-in-one setup

- Objective:
  - Install an All-in-one setup of Foreman with DNS and DHCP

- Steps:
  - Run foreman-installer with additional parameters

- Notes:
  - DNS (interface=eth0, zone=localdomain, reverse=0.0.10.in-addr.arpa, forwarders=8.8.8.8,8.8.4.4)
  - DHCP (interface=eth0, gateway=10.0.0.1, range=10.0.0.100-10.0.0.200, nameserver=10.0.0.1)

© NETWAYS

Lab 3.3: Add DNS configuration to Foreman

- Objective:
  - Create the domain 'localdomain' and associate Smart proxy

- Steps:
  - Login to Foreman
  - Navigate to 'Infrastructure > Domains'
  - Add the domain 'localdomain' and associate Smart Proxy 'foreman.localdomain'

© NETWAYS

Lab 3.4: Add DHCP configuration to Foreman

- Objective:
    - Create the subnet 'foreman' and associate Smart proxies

- Steps:
    - Login to Foreman
    - Navigate to 'Infrastructure > Smart proxies'
    - Add the subnet 'foreman' by importing from the Smart Proxy

© NETWAYS

4 Provisioning

© NETWAYS

Provisioning

- Definition:

Server provisioning is a set of actions to prepare a server with appropriate systems, data and software, and make it ready for network operation.

- Task:

Automate it!

Wikipedia defines server provisioning as *a set of actions to prepare a server with appropriate systems, data and software, and make it ready for network operation* and your task will be to automated it.

Basics on automated installation

- Installer asks questions during installation
- Answer file could be provided in several ways
    - Basic configuration like timezone and language settings
    - Partition layout
    - Software installation
    - Registration to management tools
- Additional scripts during and after installation

For an automated installation it is required that the installer of the operating system allows to answer the questions normally asked to the users by providing an answer file.

The different Linux installers can handle answer files provided via network protocols like http and ftp, network file systems or placed on the installation media. In this way basic configuration like timezone, language or network settings can be handled, furthermore partition layout can be created, software is installed and depending on the solution also registration to management tools is directly integrated.

continued...

If the installer could not solve requirements directly, scripts could be provided to be executed during and after installation.

The mechanism differs for the distributions.

Operating system family	Installer	Answer files
Red Hat	Anaconda	Kickstart
Debian	Debian-Installer	Preseed
SuSE	YaST2	AutoYaST2

Other operating systems have similar mechanisms but not all the capabilities. Microsoft Windows for example requires answer file to be placed on the installation media or a "physical" disk mounted during installation like floppy or usb.

Regardless of the capabilities with configuration management in place do a simple installation and let the configuration management solution do its work.

Kickstart

- Simple textfile with different sections
  - Main section with commands for basic configuration and partitioning
  - Package section for software installation
  - Pre and post scripts during installation
  - Additional sections can be added by addons

- Added as kernel parameter to boot media

Kickstart is the answer file to the installer used by Red Hat Enterprise Linux, CentOS, Scientific Linux, other derivates and Fedora.

It is basicly a simple text file providing commands for basic configuration and partitioning in its main section and a list of packages and package groups to install in addition to the core system in a package section. Skripts can be added in separate sections to run as pre-installation task or post-installation on the installer or using chroot on the installed system.

Lastest versions allow to extend the installer with addons which can also provide its own kickstart section for automation. One example is the OSCAP Anaconda Addon which allows to validate against a security profile already during installation.

continued...

The URL to the kickstart file can be provided during installation or for automation added as a kernel parameter to the boot media. Necessary answers missing will be queried.

Additional information:
https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/Installation_Guide/chap-kickstart-installations.html

© NETWAYS

Preseed

- Simple textfile with answers
  - Basic configuration and partitioning
  - Software installation
  - Scripts after installation

- Added as kernel parameter to boot media

Preseed is the answer file for the Debian Installer used by Debian and Ubuntu.

It is basicly a simple text file providing answers to the installation questions. It uses for all types of configuration the same syntax. Scripts can be downloaded to the system and be executed after the installation.

The URL to the preseed file can be added to the boot menu entry or as a kernel parameter to the boot media for automation, an additional parameter tells the installer to run unattended or use the answers only as default.

Additional information:
https://wiki.debian.org/DebianInstaller/Preseed

© NETWAYS

AutoYaST2

- Structured XML file
    - Basic configuration and partitioning
    - Software installation
    - Scripts during and after installation on different times
    - Configuration of selected software and devices

- Added as kernel parameter to boot media

AutoYaST2 is the answer file for SuSE Linux Enterprise Server, Desktop and openSuSE using YaST2 as installer.

It is a structured XML file meant to be created by YaST and not by hand which provides answers to the installer quests. Different data structures are used for all kinds of configuration. It can run scripts before installation starts, after partitioning, after package installation using chroot, post installation on first boot before and after starting services. In addition it can configure selected software and devices like printing service or soundcards. Other services can be configured with an file based configuration management.

The URL to the autoyast file has to be added as a kernel parameter to the boot media. Furthermore it allows to create a control server providing rules to automatically select autoyast files.

Additional information:
http://doc.opensuse.org/projects/autoyast/

© NETWAYS

Configuration in Foreman

- Operating System Version must be associated to:
  - Architecture
  - Installation media
  - Partition tables
  - Provisioning templates

In Foreman the provisioning is centred on the operating system which has to be associated to the hardware architectures like i386 or x86_64, the installation media being the URL where to find the boot media and software packages, partition tables and provisioning templates.

© NETWAYS

Provisioning templates

- ERB-Templates allow scripting and inclusion of snippets
- Types:
  - PXELinux
  - Provision
  - Finish
  - user_data
  - Script
  - iPXE
- Selected on best match:
  - Host group and Environment
  - Host group
  - Environment
  - Operating system

The templates are using ERB (embedded ruby) which allows to use parameters in the files, basic scripting like conditionals and inclusion of snippets. Snippets can be everything from scripts to configuration files you want to maintain independently because it is the same configuration for inclusion in different other files or it will bloat up one file to render it unmaintainable.

continued...

© NETWAYS

Depending on different provisioning mechanisms and methods other kinds of templates are required.

- PXELinux - Deployed to the TFTP server to ensure the Host boots the correct installer with the correct kernel options
- Provision - The main unattended installation file, e.g. Kickstart or Preseed
- Finish - A post-install script used to make custom actions after the main provisioning is complete
- user_data - Similar to a Finish script, this can be assigned to hosts built on user_data-capable images (e.g. Openstack, EC2, etc)
- Script - An arbitrary script, not used by default, useful for certain custom tasks
- iPXE - Used in {g,i}PXE environments in place of PXELinux

Templates can be associated to operating systems, host groups, environments or combinations of host group and environment. It will then select the templates to use on best match.

Partition tables are handled separately to allow the usage of the same host template with different disk layouts.

© NETWAYS

Lab 4.1: Prepare PXE installation of CentOS

- Objective:
  - Prepare the installation of CentOS using PXE

- Steps:
  - Change the Installation media "CentOS mirror" to the local repo
  - Associate the PXELinux template "Kickstart default PXELinux" with CentOS
  - Associate the Provision template "Kickstart RHEL default" with CentOS
  - Associate the operating system with the Partition table "Kickstart default", Installation media "CentOS mirror", select the Templates, set Parameter "enable-puppetlabs-repo" to "true" and change Minor version to "2.1511"

© NETWAYS

Lab 4.2: Prepare PXE installation of Debian

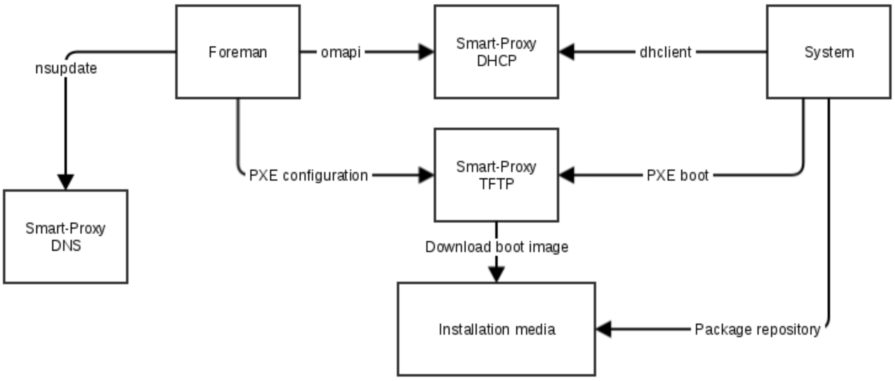
- Objective:
  - Prepare the installation of Debian using PXE

- Steps:
  - Change the Installation media "Debian mirror" to the local repo
  - Create the Operating system "Debian" with Major version "8", Minor version "3", Description "Debian jessie", Family "Debian", Release name "jessie"
  - Associate the PXELinux template "Preseed default PXELinux" with Debian
  - Associate the Provision template "Preseed default" with Debian
  - Associate the finish template "Preseed default finish" with Debian
  - Associate the operating system with the Templates and set Architecture "x86_64", Partition table "Preseed custom LVM" and "Preseed default", Installation media "Debian mirror"

© NETWAYS

Provisioning using PXE

- Identifier is the mac address



For the installation using PXE the identfier used is the mac address so it is required during configuration of the host in Foreman. For communication with Foreman a token is created as identifier.

After the host is created in Foreman, it reserves an IP address in DHCP, creates DNS records and places a PXE configuration on the TFTP server. If not already existing the Smart Proxy TFTP downloads the boot image to its directory.

continued...

© NETWAYS

When the host is started it gets its IP address using DHCP which also tells him to boot via PXE from the TFTP server. The PXE configuration points also to the answer file provided by Foreman which will point to the configured installation media for package installation.

Additional communication to Foreman, Puppet and other systems will be required depending on the PXE configuration or answer file provided. Typically a system should register to all management systems and finish its installation by contacting Foreman.

When Foreman is told an installation is finished, it will clean up by changing the PXE configuration to local boot and change other configurations only need during provisioning.

The Foreman manual provides some more detailed workflow diagrams:
http://theforeman.org/manuals/1.10/index.html#4.4.6Workflow

Lab 4.3: Create a virtual machine "pxe"

- Objective:
  - Create a virtual machine "pxe" for PXE installation

- Steps:
  - Open "Virtual Machine Manager" application
  - Select "New virtual machine" from the menu or by pressing the button
  - Select PXE boot
  - Select "Linux" and "Red Hat Enterprise Linux 7.2" for CentOS or "Debian jessie" for Debian according to your preferences
  - Keep the minimum requirements for RAM, CPU and Disk
  - Name your VM "pxe" and select the network "foreman"
  - Create the VM and immediately pause it so in the next lab the required configuration in Foreman can be created

© NETWAYS

Lab 4.4: Configure the system "pxe" in Foreman

- Objective:
  - Configure the system "pxe" in Foreman and start the installation

- Steps:
  - Open Foreman's host dialog using "Host > New Host"
  - On the Host tab name it "pxe" and select the Environment, Puppet CA and Master
  - On the Interface tab click edit to configure the interface with the MAC address, identifier "eth0", select Domain and Subnet and keep the suggested IP address.
  - On the Operating system tab select the Architecture, Operating System, Media, Partition table and set a Root password.
  - Unpause the VM

continued...

© NETWAYS

In the Interface tab it is possible to add multiple interfaces and also virtual interfaces like VLAN tagged interfaces or aliases. Only one can be assigned as the primary interface mapped to the DNS record and only one as provisioning interface for PXE configuration or connecting for executing scripts during installation. This can be the same interface but does not have to be in cases you have a dedicated installation network.

Instead of setting the root password for every host created, a default can be set by providing a password hash on "Administer > Settings" in the Provisioning tab as Root password. This is an MD5 hash for being supported by every Linux distribution.

© NETWAYS

Hostgroups

- Grouping Hosts
  - Nested groups in a hierarchical way
- Adds defaults to hosts
  - Required options for provisioning
  - Options for configuration management
  - Options for compute resources
  - Parameters
- Allows to associate templates

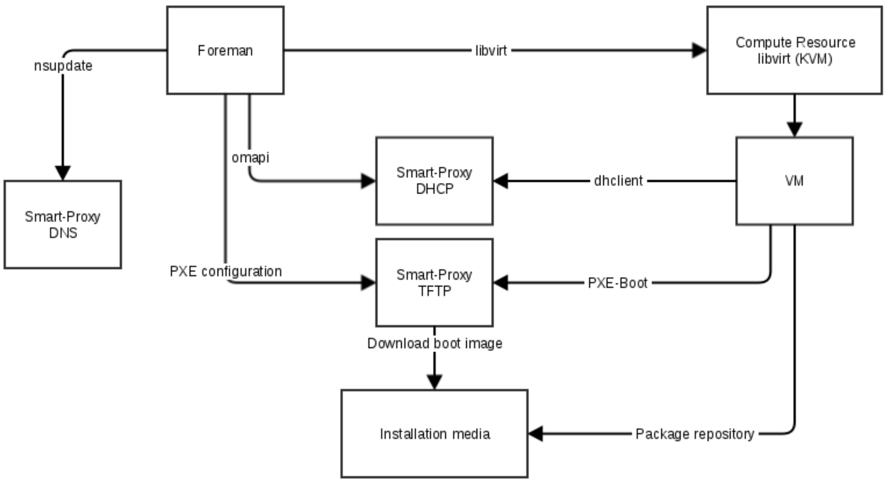
Hostgroups are used to group hosts, the groups can be organized in a hierarchical way by nesting them.

The Hostgroups allows to add defaults for all options required by Foreman for provisioning and depending on your environment also for configuration management and compute resources. Additional parameters can be set.

Furthermore to providing defaults configured at the hostgroup "Provisioning templates" can be associated to change from the operating system default.

Provisioning using Compute resource libvirt

- Identifier is the name of the system



For the installation using a Compute resource the identfier used is the name of the system. A virtual machine will be created with the name and the mac address will be returned to use for PXE. For communication with Foreman a token is created as identifier.

continued...

© NETWAYS

After the host is created in Foreman, it creates the virtual machine, reserves an IP address in DHCP for the mac address returned, creates DNS records and places a PXE configuration on the TFTP server. If not already existing the Smart Proxy TFTP downloads the boot image to its directory.

Then the virtual machine is powered on and it works the same way like simple PXE installation.

Lab 4.5: Prepare Compute resource libvirt

- Objective:
  - Prepare Compute resource libvirt to install a virtual machine

- Steps:
  - Install the Compute resource using the Foreman installer
  - Create a passphraseless ssh-key for user foreman
  - Copy the public key to the root account of the host
  - Configure the Compute resource in Foreman Web GUI

The Compute resource does not only provide the possibility to provision virtual machines, it also allows access to power management of unmanaged systems and to deleted them from the virtualization plattform. Also assiocating an unmanaged system to an already existing one is possible.

© NETWAYS

Lab 4.6: Create the virtual machine "compute" from Foreman

- Objective:
    - Create the virtual machine "compute" from Foreman Web GUI and start an unattended installation

- Steps:
    - Open Foreman's host dialog using "Host > New Host"
    - In the Host tab name it "compute" and select to deploy on the Compute resource, the Environment, Puppet CA and Master
    - In the Interface tab click edit to configure the interface with identifier "eth0", select Domain and Subnet and keep the suggested IP address, for the Libvirt options choose the virtual network "foreman"
    - In the Operating system tab select the Architecture, Operating System, Media, Partition table and set a Root password.
    - In the Virtual Machine tab change the Storage type to "QCOW2"

continued...

© NETWAYS

Compute resource options depend on the used Compute resource so other options will be available for VMware than for libvirt.

Defaults for the Compute resource options can be set using a Compute profile.

Storage type "QCOW2" enables you to use snapshots unlike "RAW".

Compute profiles

- Used to provide defaults for virtual machine creation
- Default provides three for different sized VMs
- Same profile names could be used for different Compute resources
  - Different defaults for same type of systems based on virtualization plattform
  - Only available to Compute resource if configured
- Available in "New Host" dialog

Compute profiles are used to provide defaults for the virtual machine creation. Per default three profiles are available and only have to be configured, but you can create as many as needed. The same profile can differ based on the Compute resource used to provide defaults matching the virtualization plattform.

If one is configured for a Compute resource option to choose from it is available on new host dialog after choosing the Compute resource to deploy on.

Images

- Available to install from instead of unattended installation
- Image has to be configured
  - On Compute resource using "New image"
  - Has to provide access via ssh to run finish scripts
  - API can allow to run user_data scripts for additional changes
- Available in "New host" dialog
  - Speeds up installation
  - Should be very basic image, but of course can be very featureful if required

Images are available as install source to all Compute resources, for some as the only source. To be available to Foreman it has also to be configured using the "New image" dialog after selecting a Compute resource. The image has to be created with an user which has shell access to run finish scripts and depending on the Compute resource also user_data scripts can run during virtual machine creation to change settings.

continued...

© NETWAYS

If a image is configured it is possible to choose it from the "New host" dialog on the "Operating system" tab. In some cases it can speed up the installation process, but of course an unattended installation will give you a cleaner and more up to date system. If using a configuration management a very basic image should be prefered, but if required a image can be very featureful like having some proprietary software installed which is quite complicated to install with configuration management.

© NETWAYS

Console

- Direct console access on virtual machines
- Javascript library noVNC
- Protocols: VNC or SPICE
- Depending on Compute Resource provider
- Encryption and Authentication based on environment

Foreman uses the javascript library noVNC to give the user direct access to the console of a virtual machine depending on the Compute Resource provider. The Foreman manual explains additional steps required on the Compute Resource if not available by default.

The protocols available are VNC and SPICE, encryption and authentication depends on your setup.

If you want to try it in the training environment you have to enable the connection by executing the following command on the laptop.

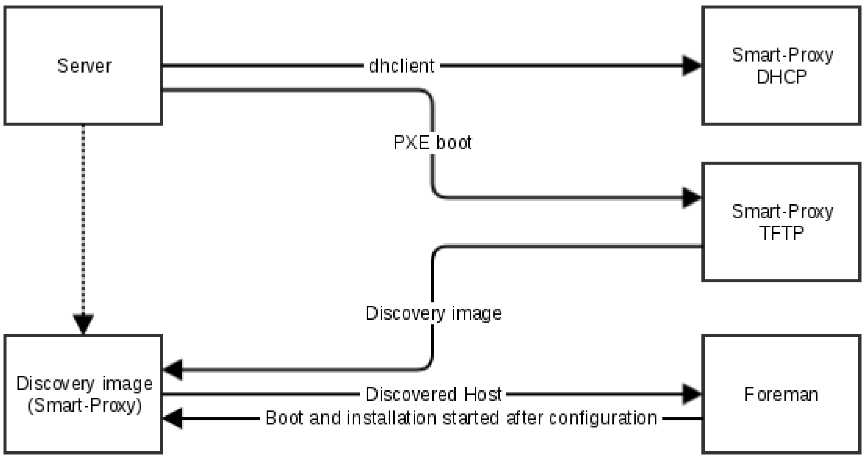
```
# iptables -I INPUT -p tcp -m multiport --dports 5901:6000 -i virbr1 -j AC
CEPT
```

Furthermore Firefox will block the websocket connection unless you change setting network.websocket.allowInsecureFromHTTPS to true in about:config.

© NETWAYS

Provisioning using Discovery plugin

- Identifies itself as discovered host with its mac address



For the installation the system boots a discovery image and identifies itself as "Discovered Host" in the Foreman Web GUI. As an identifier the host uses its mac address. Then configuration is done in Foreman and after submitting it the image is told to reboot and starts an installation via PXE afterwards.

continued...

© NETWAYS

For this communication the discovery image uses the Smart proxy included in it.

Lab 4.7: Install and configure the Discovery plugin

- Objective:
  - Install and configure the Discovery plugin

- Steps:
  - Run the Foreman installer to install the Discovery plugin and download the image
  - Adjust and deploy the PXE default configuration
  - Enable the discovery widget in the dashboard

© NETWAYS

Lab 4.8: Create a virtual machine "discovery"

- Objective:
  - Create a virtual machine "pxe" for PXE installation

- Steps:
  - Open "Virtual Machine Manager" application
  - Select "New virtual machine" from the menu or by pressing the button
  - Select PXE boot
  - Select "Linux" and "Red Hat Enterprise Linux 7.2" for CentOS or "Debian jessie" for Debian according to your preferences
  - Keep the minimum requirements for RAM, CPU and Disk
  - Name your virtual machine "discovery" and select the network "foreman"
  - Create the virtual machine and when the PXE menu appears select "(discovery)"

© NETWAYS

Lab 4.9: Configure the system "discovery" in Foreman

- Objective:
  - Configure the system "discovery" in Foreman and start installation

- Steps:
  - Select the newly discoverd host from the widget
  - On the Host tab name it "discovery" and select the Environment, Puppet CA and Master
  - On the Interface tab click edit to configure the interface add the Domain "localdomain"
  - On the Operating system tab select the Architecture, Operating System, Media, Partition table and set a Root password
  - Submit to start the installation

Discovery rules

- Automatic rule based installation
  - Located in "Configure > Discovery rules"
  - Match on facts provided by the discovery image
  - Naming based on template
  - Installation and configuration based on hostgroup
  - Provides a limit and a priority for ordering

- Not enabled by default
  - Requires setting "discovery_auto" changed to "true"

Discovery rules allow an automatic rule based installation. Depending on facts provided by the discovery image a rule is selected and the system is installed based on the configuration of the selected hostgroup. Naming is done based on a template using ERB which allows to use facts or random numbers, by default the macaddress is used. Setting a limit and a priority allows some ordering like deploy first two backend systems of this size and then two frontend systems.

Usage of the rule based installation is not enabled by default but simple switched on by changing the setting "discovery_auto" to "true".

Discovery image

- Provided by the project
  - Allows to add custom facts
  - Allows to add custom extensions like drivers
  - Could be remastered to provide defaults
  - Can use kexec to directly boot a new kernel

The discovery image is provided by the project also with corresponding tools and documentation on extending and remastering it.

It also allows to add custom facts in its interface, as boot parameter or as an extension which enables a workflow like booting the image and adding desired system type as fact.

Furthermore it could be used in enviroments without PXE and DHCP by providing all settings including an ip address and boot directly into a new kernel with kexec. This is only available for Red Hat derivates at the moment.

© NETWAYS

Provisioning using Bootdisk plugin

- Foreman Plugin Bootdisk provides 4 kinds of boot images
    - Host images
    - Full host image
    - Generic image
    - Subnet image
- Based on iPXE for environments without control over network

For the installation in an environment you do not have total control over the network infrastructure the Foreman Plugin Bootdisk provides 4 kinds of boot images based on iPXE.

The host image contains a static network configuration, loads the installer from the media configured in Foreman and the Provisioning configuration from Foreman itself. So it requires no DHCP and TFTP in the network.

continued...

© NETWAYS

The full host image contains the operating system specific installer so it requires no downloading of it, but is configured to boot from DHCP instead of having a static network configuration. To get its Provisioning configuration from Foreman it identifies itself with a token only valid for one deployment.

The generic image boots from a dynamic IP address of the DHCP pool and is identified by the MAC address for Foreman providing the correct installer via TFTP and Provisioning configuration to load.

The subnet image is basicly the same as the generic image but uses another TFTP server specified for the subnet.

© NETWAYS

Lab 4.10: Install and configure the Bootdisk plugin

- Objective:
    - Install and configure the Bootdisk plugin

- Steps:
    - Run the Foreman installer to install the Bootdisk plugin
    - Associate iPXE template for operating systems

© NETWAYS

Lab 4.11: Reinstall the virtual machine "pxe"

- Objective:
  - Reinstall the virtual machine "pxe" from a host image

- Steps:
  - Set the Host in "Build" mode and download image
  - Configure virtual machine to boot from image
  - Boot and reinstall the virtual machine

© NETWAYS

5 Configuration management

© NETWAYS

Configuration management

- Definition:

Configuration management is a systems engineering process for establishing and maintaining consistency of a product performance, functional, and physical attributes with its requirements, design and operational information throughout its life.

- Tools integrated in Foreman:
    - Puppet
    - Chef
    - Salt
    - Ansible

Wikipedia defines configuration management as a *systems engineering process for establishing and maintaining consistency of a product's performance, functional, and physical attributes with its requirements, design and operational information throughout its life.*

So configuration management software is used to describe a desired state, configure a system to be in this state and recognize drifts from this state to revert them.

continued...

© NETWAYS

Foreman integrates different tools to solve this task. Puppet is deeply integrated because Foreman started as a WebGUI for it, but it is planned to move on to a state being a plugin equal to all the other solutions. The features of the plugins differ depending on the capabilities based ont the tools they depend on. For example some can only report system information and state while others can assign a configuration using the web interface. Some tools also have a focus more on rapid deployment than on managing the system state.

We will use puppet in this training because it provides the complete feature set and its descriptive language is also quite accessible/readable without knowing puppet in-depth.

© NETWAYS

Puppet

- Written in ruby
- Choice between Open Source or Enterprise version
- Runs on Linux, Unix, Windows
- Describes state in its own declarative language or a Ruby DSL

```
package { 'openssh':
  ensure => 'installed'
}
```

- Workflow
  - Manifests stored on a central server "Puppet Master"
  - Agent collects system information using facter
  - Agent contacts central server with this information
  - Master compiles catalog for agent to realize using an abstraction layer
  - Agent reports back to master
  - Master transfers reports to other tools

Puppet is written in ruby and provided as a true Open Source version and as an Enterprise version with additional features and packages with a defined software stack for easier support.

continued...

© NETWAYS

Independently of the version it runs on Linux, Unix and Windows it can also configure some network devices. For configuration it uses its own declarative language called Puppet DSL (Domain Specific Language) you can see above (example) or it can also use a Ruby DSL. The desired state is described in so called manifests which are stored on one or multiple central servers. To connect the different configuration items with the node to be configured these central servers can use an ENC (External Node Classifier). The agent runs on the nodes and collects system information using a tool named facter before contacting the central server. The master compiles then a catalog based on the facts provided by the agent and the manifests. This catalog is then realized by the agent using an abstraction layer and also sends a report to the master. The master uses different handlers to send the report to other tools.

A diagram showing this workflow is provided on the next page.

© NETWAYS

Puppet Workflow

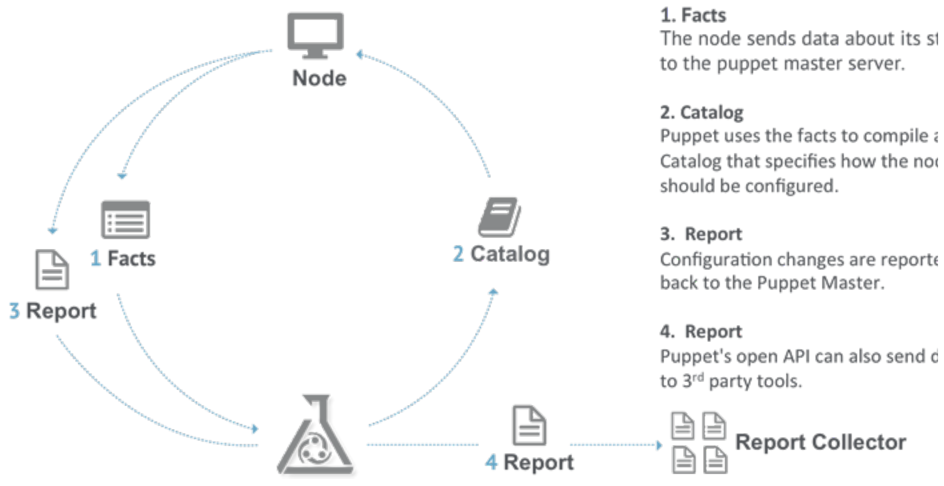

Node
1 Facts
2 Catalog
3 Report
4 Report
Report Collector
1. Facts
The node sends data about its st
to the puppet master server.
2. Catalog
Puppet uses the facts to compile a
Catalog that specifies how the noc
should be configured.
3. Report
Configuration changes are reporte
back to the Puppet Master.
4. Report
Puppet's open API can also send d
to 3rd party tools.

Image copyright by Puppetlabs.

Foreman Puppet Integration

- Foreman -> Puppet
  - Smart proxy Puppet allows to import Puppet modules
  - Smart proxy Puppet allows to trigger agent runs
  - Smart proxy Puppet CA integrates certificate handling
- Puppet -> Foreman
  - Puppet uploads facts to Foreman
  - Puppet uses Foreman as ENC
  - Puppet transfers reports to Foreman

Foreman integrates Puppet in several ways and also integrates itself into Puppet. Communication from the WebGUI to Puppet is handled using the Smart proxy from Puppet. It allows to import Puppet modules known to Puppet and to trigger Puppet agent runs using several protocols. The Smart Proxy Puppet CA integrates certificate handling into provisioning so auto signing of the agents certificate requests during build is allowed and also allows to manage the complete CA in the WebGUI.

continued...

© NETWAYS

On the Puppet master a script is deployed which integrates Foreman as an ENC so classes selected in the WebGUI are deployed on the system. This mechanism is also used to upload the facts provided by the agent during Puppet agent run and creating a host entry if facts are provided for an not already existing system. Also Puppet is integrated as a reporting target to enable the web interface to show reports.

© NETWAYS

Lab 5.1: Import of Puppet classes

- Objective:
  - Make Puppet code available to Puppet and Foreman

- Steps:
  - Place Puppet modules in Puppet environment "production"
  - Import classes in Foreman

- Optional:
  - Configure Foreman to ignore specific classes

© NETWAYS

Parameters vs. Smart class parameters vs. Smart Variables

- Parameters
  - Simple string
  - Usable in Foreman's Provisioning Templates
  - Usable in Puppet as global parameters
  - Override by creating one of the same name in a more specific scope

- Smart class parameters
  - Available from Puppet classes
  - Different types
  - Validators
  - Override options to handle override order and behaviour

- Smart variables
  - Global parameters assigned to a Puppet class
  - Same options like Smart class parameters

- All are hideable from unprivileged users

continued...

© NETWAYS

Foreman does differentiate between three kinds of parameters.

Parameters are global parameters in a very simple fashion. Their values can only be strings and override is simply done by creating a parameter with the same name in a more specific scope. To Puppet they are presented as a global parameter via the ENC, in Foreman they can also be used in the Provisioning Templates.

Smart class parameters become available from imported Puppet classes and can have different types like boolean, hash or yaml. For this types an input validator can be created to verify user input. An override behavior and order can be defined to enable merging values depending on facts.

Smart variable provides the same options as Smart class parameters but are manually assigned to a Puppet Class and provided to Puppet as global parameters and not class parameters.

All types have to be created on the global scope to be available in more specific scopes and all allow to hide them from unprivileged users.

© NETWAYS

Lab 5.2: Parameterize and assign Puppet classes

- Objective:
  - Parameterize and assign Puppet classes to at least one host

- Steps:
  - Set defaults to the Smart class parameters provided by the imported class
  - Assign the Puppet class in the host menu to one host

- Optional:
  - Assign the Puppet class to another host and override the defaults

© NETWAYS

Lab 5.3: Trigger Puppet agent run and inspect the report

- Objective:
  - Trigger an Puppet agent run and inspect the report

- Steps:
  - Run the Puppet agent on the host you assigned the class
  - Inspect the report of the Puppet agent run

© NETWAYS

Triggering an Puppet agent run

- Multiple protocols available
  - Puppet kick - requires Puppet agent to listen for incoming events
  - Mcollective - requires a message queue and the Mcollective framework
  - Puppetssh - runs Puppet agent command via ssh
  - Salt - uses Salts remote execution to trigger a Puppet agent run
  - Customrun - runs whatever script you provide
- Has to be enabled via a setting
- Remote Execution plugin provides more flexibility

The Smart proxy Puppet allows to trigger the Puppet run we manually triggered in the last exercise from the WebGUI. For this we enable it by setting the option "Puppetrun" to true and configure the Smart Proxy to use one of the mentioned providers, but all of them will also require some configuration on the agent side.

The Foreman plugin Remote Execution is more flexible and also handles the agent configuration but only provides SSH for now. We will have a deeper look into this instead of configuring the "Puppetrun" feature.

Managing Foreman with Puppet

- Puppet modules provided by Foreman Project
  - Already utilized by Foreman Installer
  - Compatible with all supported plattforms
  - Version compatibility to observe

- Configuration of:
  - Foreman
  - Smart Proxy
  - Puppet - in the Foreman way
  - Depending and managed services

It is also possible to manage Foreman and/or its Smart Proxies using Puppet. The modules to do so are provided by the Foreman Project itself and are already used in the Foreman Installer. The modules are written to be compatible with all supported plattforms. For compatibility of the modules with the Foreman or Smart Proxy version observe the notes in the README. File.

The modules provided can configure Foreman, the Smart Proxy and Puppet in the way it is required by Foreman and the services required to run Foreman or managed by the Smart Proxy.

© NETWAYS

Utilizing Foreman within Puppet

- Function to query the API
    - Alternative to exported resources and PuppetDB query
    - Login data and query as hash
    - Returns hash

```
$foreman = foreman({ item => 'hosts',
                     search => 'status.failed = 0',
                     per_page => 1000,
                     foreman_url => 'https://foreman.localdomain',
                     foreman_user => 'admin',
                     foreman_pass => 'PASSWORD' })
```

The Puppet module "foreman" provided by the Foreman project includes a function to query the Foreman API in a puppet class. This is an alternative for exported resources or a PuppetDB query. It takes a hash with the login data and the query and returns a result hash including an array of hashes describing the hosts.

The hash is best used with a defined resource and create_resource function or within a template.

Next release of the function will also allow to provide a filter for reducing the data for easier handling.

© NETWAYS

6 User management

© NETWAYS

LDAP

- Natively supported including webbased configuration
- Supports:
  - Protocol: LDAP, LDAPS, no StartTLS
  - Schema: POSIX, Active Directory, FreeIPA
- Allows multiple backends
- Autocreation of user possible
- Optional synchronisation of group membership
- Getting stored avatars

Foreman supports LDAP authentication natively and provides a webbased configuration for it.

Supported are LDAP and LDAPS as protocol, (at the moment) for now no StartTLS. If using certificate based encyrption trust to the certificate is mandatory. It supports the POSIX, Active Directory and FreeIPA schema for storing users and groups, but attribute mapping can be configured manually. For the autocreation of users the attributes "Login Name", "First Name", "Surname" and "Email" are required, optionally a avatar can be stored as base64 encoded string in a "Photo" attribute.

© NETWAYS

Lab 6.1: LDAP Authentication

- Objective:
  - Allow the administrative accounts from the LDAP to work as Foreman admins

- Steps:
  - Configure the LDAP authentication including group synchronisation
  - Add a administrative group to grant the administrative accounts from the LDAP privileges

© NETWAYS

External Authentication

- External authentication utilizing the webserver
- Kerberos
  - Foreman Installer can handle FreeIPA automatically
  - Manual configuration and tell Foreman to trust
- PAM
  - Apache module to validate login form input against PAM
- Autocreation of user possible
- Differentiates between WebGUI and API

Foreman can also be configure to use an external authentication provided by the webserver.

In most cases this will be Kerberos to achieve a true Single Sign-On. If you use FreeIPA as your authentication provider the Foreman Installer can create the required configuration by passing the corresponding parameters to it.

continued...

If you use another provider like Active Directory or a manually managed Kerberos it is possible to create the required Kerberos and PAM authentication on your own and tell Foreman to accept external users.

Another way to configure Foreman to use external authentication providers is by using the Apache module "intercept_form_submit" to pass the login form input to PAM for validation.

It is also possible to enable autocreation of users in Foreman for external authentication.

Also note that Foreman differentiates between WebGUI and API login for external authentication so it is possible to allow external users to only connect to one or both.

Permissions

- Fine granular permission system
- Abstracted to roles
- Complemented by filters
- Plugins can add to the permission system

Lab 6.2: Add unprivileged users

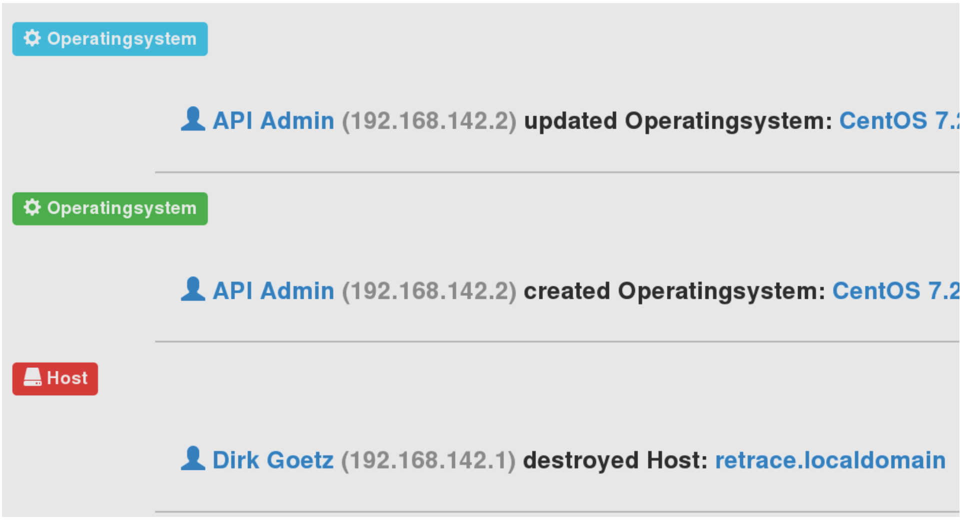
- Objective:
  - Grant access and privileges for some unprivileged users

- Steps:
  - Assign the role "Viewer" to the user "viewer"
  - Create a role "Selfservice" to allow creation of new hosts and management of own hosts
  - Assign the new role "Selfservice" to the user "selfservice"

© NETWAYS

Auditing

- Almost all changes from WebGUI and API
- User, timestamp, change and parent object
- For templates additional diff of the changes
- Browse and searchable in the WebGUI



Operatingsystem

API Admin (192.168.142.2) **updated Operatingsystem:** CentOS 7.

Operatingsystem

API Admin (192.168.142.2) **created Operatingsystem:** CentOS 7.2

Host

Dirk Goetz (192.168.142.1) **destroyed Host:** retrace.localdomain

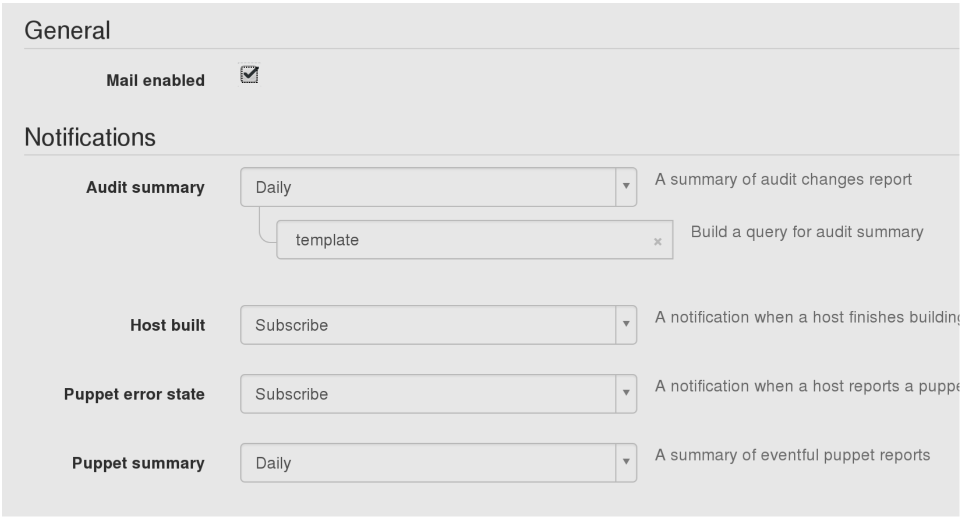
continued...

Foreman logs almost all changes from the WebGUI and API including the user, a timestamp, the change and the parent object like the host or a parameter got changed. For templates an additional diff of the changes is provided.

All this auditing data are browse and searchable in the WebGUI via "Monitor > Audits". To get detailed auditing data or specific change history simply select an entry.

Email Notifications

- General configuration via Foreman Installer or configuration file
- Optional setting by user
- Frequent summary of specific audit events
- Finished host provisioning
- Puppet run with error on every run or as frequent summary



General

Mail enabled	✓	

Notifications

Audit summary	Daily	A summary of audit changes report
	template	Build a query for audit summary
Host built	Subscribe	A notification when a host finishes buildin
Puppet error state	Subscribe	A notification when a host reports a puppe
Puppet summary	Daily	A summary of eventful puppet reports

© NETWAYS

Foreman can send email notifications to the user. This requires email settings configured in Foreman via the Foreman Installer or manual editing of the configuration file "/etc/foreman/email.yaml". With this configuration in place a user can decide to opt-in for several kinds of notification. It can send a daily, weekly or monthly summary of audit events which allows to add a search to specify a subset of events. It also can notify on every successful host build and for Puppet runs on every failed run or as a frequent summary.

© NETWAYS

7 Plugins

© NETWAYS

Plugins

- Extend Foreman
  - Add permissions and roles
  - Alter the menu
  - Provide dashboard widgets and pagelets
  - Add URLs and actions
  - Deface existing views and add new ones
  - Add parameters to a host
  - Authenticate a Smart proxy
  - Add to the database
  - Create their own logging

- Extend Smart Proxy
  - Additional providers for existing services
  - Additional services

Plugins can extend Foreman and the Smart proxy in many different ways and depending on the needs it can be very simple or highly complex.

Some of the options will be shown in the next slides where some of the plugins are introduced.

For all options have a look at the "how to create a plugin" at http://projects.theforeman.org/projects/foreman/wiki/How_to_Create_a_Plugin

Default Hostgroup

- Sets a default hostgroup on host creation
- Configured as a yaml hash
- Use case: Introduce Foreman in an already puppetized environment

---

If a host does not already exist, it will be created when the first puppet run triggers the upload of the facts. Normally the host is added without any additional configuration in Foreman, the plugin "Default Hostgroup" simply adds the possibility to set a default hostgroup to assign to these newly created hosts. This allows to add a classification via the hostgroup already on the first run and without manual intervention.

To use it simply install the package and create a file "/etc/foreman/plugins/default_hostgroup.yaml" containing a hash which maps facts to hostgroups to assign.

```
---
:default_hostgroup:
  :facts_map:
    "Default":
      "hostname": ".*"
```

continued...

© NETWAYS

The plugin is very useful if you plan to embed Foreman in an already puppetized environment as Puppet ENC.

More details on:
https://github.com/theforeman/foreman_default_hostgroup

Templates

- Creates a synchronisation job for templates
  - Community Templates
  - Your own git repository

- Use case:
  - Get additional templates maintained by the community
  - Manage your own templates with external versioning

By installing the plugin you get a synchronisation job for templates which per default import the community templates from https://github.com/theforeman/community-templates matching your Foreman version and adding support for more operating systems. This job can also be used to import your own git repository if you want to manage the templates on an external version control system. In addition the imported templates will also be associated to the existing operating systems.

More details on:
https://github.com/theforeman/foreman_templates

© NETWAYS

Lab 7.1: Templates

- Objective:
  - Import the Community templates

- Steps:
  - Install the Foreman Plugin Templates
  - Run the synchronisation job

© NETWAYS

DHCP Browser

- Adds the DHCP management interface
  - Browse DHCP reservations
  - Create DHCP reservations for hosts not managed by Foreman

- Use case: More transparency for the DHCP management

The Foreman Plugin DHCP Browser adds a DHCP management interface to the Foreman WebGUI next to the subnets which allows to browse DHCP reservations and it is also quite useful for managing reservations for hosts not managed by the Foreman.

More details on:
https://github.com/theforeman/foreman_dhcp_browser

© NETWAYS

Lab 7.2: DHCP Browser

- Objective:
  - Inspect DHCP reservations

- Steps:
  - Install the Foreman Plugin DHCP Browser
  - Inspect DHCP reservations via the Plugin

© NETWAYS

Hooks

- Adds hooks to events in Foreman
  - Run additional scripts on events
  - Show status of the scripts during orchestration
  - Handles failure and rollback

- Use case:
  - Extend the Foreman without writing a plugin
  - Work around the limitations and problems of existing tools

The Foreman Plugin Hooks allows to add scripts on events handled by the Foreman. Typically it is used to integrate other tools during the orchestration process of deploying new hosts instead of writing a new plugin. For this the status of a script executed is also shown in the WebGUI and handling for failures and rollback is also provided.

Another use case is to work around the limitations and problems of tools like changing parameters of VMs which is only available after the creation but are required before starting them.

More details on:
https://github.com/theforeman/foreman_hooks

© NETWAYS

PuppetDB

- Disable hosts in the PuppetDB during deletion
- Display the PuppetDB in the dashboard
- Use case: Should always be installed when using PuppetDB with Foreman

When PuppetDB is used with Foreman there is normally no interaction between both, they are completely two different independent report targets. With this plugin a host is also disabled in PuppetDB when it is deleted in Foreman.

In addition the PuppetDB dashboard is also integrated in the Foreman WebGUI to display the performance of the backend.

More details on:
https://github.com/theforeman/puppetdb_foreman

© NETWAYS

Column View

- Adds additional columns to the "All Hosts" view

- Use case:
  - Present more information in the WebGUI
  - Add links to the other webinterfaces

The Foreman Plugin Column View is a very simple example of enhancement an existing view in Foreman but could be verify useful in combination with some custom facts like showing the responsible team or project. It is also capable of adding additional links to other webinterfaces providing additional information.

More details on:
https://github.com/theforeman/foreman_column_view

Lab 7.3: Column View

- Objective:
  - Add Architecture and Uptime to the "All Hosts" view

- Steps:
  - Install the Foreman Plugin Column View
  - Configure it to show the facts for architecture and uptime

© NETWAYS

ABRT

- Automatic Bug Reporting Tool
- Available on RHEL, CentOS, Fedora, OpenSuSE, Mandriva
- Collects crash dumps
- Forwards them to a retracing server
- Use case:
  - Get detailed crash reports from your systems
  - Forward crash reports for easier bug analyses

The Automatic Bug Reporting Tool is per default installed on Fedora and RHEL derivates and also available for OpenSuSE and Mandriva to automatically open bug reports on their bug trackers to provide a crash dump for easier bug analyses. The retracing server allows for grouping and prioritizing.

With the Foreman Plugin ABRT you can collect those crash dumps on the Foreman host instead of directly forwarding them to the internet. It also allows to forward them automatically to the upstream retracing server, your own or manual selection which crash report is forwarded.

More details on:
https://github.com/theforeman/foreman_abrt

ABRT - Workflow

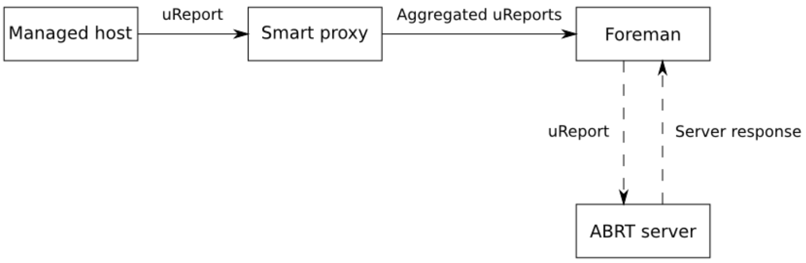


Image copyright by the Foreman Project.

© NETWAYS

Lab 7.4: ABRT

- Objective:
  - Collect crash dumps on your Foreman server

- Steps:
  - Install the Foreman Plugin ABRT
  - Set up the Smart proxy
  - Configure the host
  - Create a crash dump to test the setup

© NETWAYS

OpenSCAP

- Open Source implementation of Security Content Automation Protocol
- Combines different security standards
- Validate systems for security compliance
  - Provides guides
  - Provides reports and remediation
- Profiles provided by different resources
  - Files to start with included
  - Other resources like the "National Institute of Standards and Technology"
- <u>Use case:</u> Collect security compliance reports in the WebGUI

continued...

© NETWAYS

OpenSCAP is the Open Source implementation of Security Content Automation Protocol which combines different pre-existing security standards like CVE, CCE, CPE, CVSS, XCCDF, OVAL, OCIL, AI, ARF, CCSS and TMSAD. All these informations are combined in datastream files which can contain different profiles a system can be validated against. To get a compliant system a guide can be created or a compliance report including some remediation scripts. The required files can be created by hand but are in XML so best practice is to use a tool like the Workbench to tailor the existing files to the like of the one provided by the OpenSCAP project or the "National Institute of Standards and Technology" (NIST).

Not used by Foreman for now is the Anaconda Plugin OpenSCAP which can also add security compliance as part of the installation process.

With Foreman 1.11 the Plugin will get a rewrite to remove the dependency on "scaptimony"!

More details on:
https://github.com/theforeman/foreman_openscap

Lab 7.5: OpenSCAP

- Objective:
    - Inspect the Security compliance of your system

- Steps:
    - Install the Foreman Plugin OpenSCAP
    - Install the Smart Proxy Plugin OpenSCAP
    - Make the Puppet Module "foreman_scap_client" available
    - Create a Policy for CentOS 7 and assign it to a host
    - Initiate a Puppet agent run on the host
    - Create a report on the host and upload it to the Smart proxy
    - Upload the report from the Smart proxy to the Foreman

© NETWAYS

Cockpit

- WebGUI to manage your server
- Available on RHEL, CentOS, Fedora, Arch Linux and Ubuntu
- Available as native package or docker container for atomic
- Use case:
  - Manage atomic installations
  - Manage servers via WebGUI instead of direct console access

---

The Cockpit project provides a WebGUI to manage Linux servers. It is available as native package or priviledged docker container for atomic on RHEL, CentOS, Fedora, Arch Linux and Ubuntu.

The Foreman plugin searches for an existing installation and integrates it into the Foreman WebGUI.

More details on:
https://github.com/theforeman/foreman_cockpit

© NETWAYS

Lab 7.6: Cockpit

- Objective:
  - Inspect your system using Cockpit integrated in the Foreman

- Steps:
  - Install and enable Cockpit
  - Install the Foreman Plugin Cockpit

© NETWAYS

Remote Execution

- Execute jobs on remote systems from the Foreman WebGUI

- It can handle different protocols
  - SSH - completed
  - Ansible - work in progress
  - Salt - planned
  - Mcollective - planned

- Use case:
  - Trigger configuration management runs immediately
  - Execute one-time or irregular commands
  - Orchestrate operations on servers

The Foreman Plugin Remote Execution adds WebGUI and workflow for executing jobs on remote systems. It utilizes different providers, but for now only SSH is implemented. In the design concept are plans for more providers, Ansible which is already work in progress, Salt and Mcollective.

The SSH provider runs per default command as root, but can also be configure to run as unpriviledged user and run sudo to accquire elevated privileges.

continued...

© NETWAYS

It is usefully to trigger configuration management runs immediately to get an adhoc deployment, execute one-time or irregular commands and also to orchestrate operations like updates on your servers. It also allows to schedule jobs or reoccurring execution.

More details on:
http://theforeman.org/plugins/foreman_remote_execution/

Lab 7.7: Remote Execution

- Objective:
  - Trigger a Puppet run on a remote system

- Steps:
  - Install the Foreman Plugin Remote Execution
  - Bring out the SSH key
  - Initiate the Puppet run

© NETWAYS

Remote Execution - Job Templates

- Similar to Provisioning Templates
- Input
  - User Input - free-form or list of values
  - Fact value
  - Variable
  - Puppet parameter
  - Reference on other templates
  - Special ERB function "input"
- Default Templates:
  - Puppet Run Once
  - Package Action
  - Service Action
  - Run Command

It is also possible to write templates for jobs in a similar manner like Provisioning Templates. In this templates you can use a special function "input" to get the value of input fields associated to it. These fields can be a free-form or list of values for a user to add, facts, variables or Puppet parameters.

continued...

© NETWAYS

Furthermore you can reference another template, which you can also render in your newly created one with the "render_template" function.

The default templates provided are "Puppet Run Once" to trigger a puppet run, "Package Action" for handling package management and "Service Action" for managing service which both include operating system specific handling and "Run Command" to run a simple commandline.

© NETWAYS

Lab 7.8: Remote Execution - Job Template

- Objective:
    - Create a Job Template "ping" to run the ping command on remote hosts

- Steps:
    - Create a Job Template to run ping with default values and input field
    - Run it without input
    - Run it with input

© NETWAYS

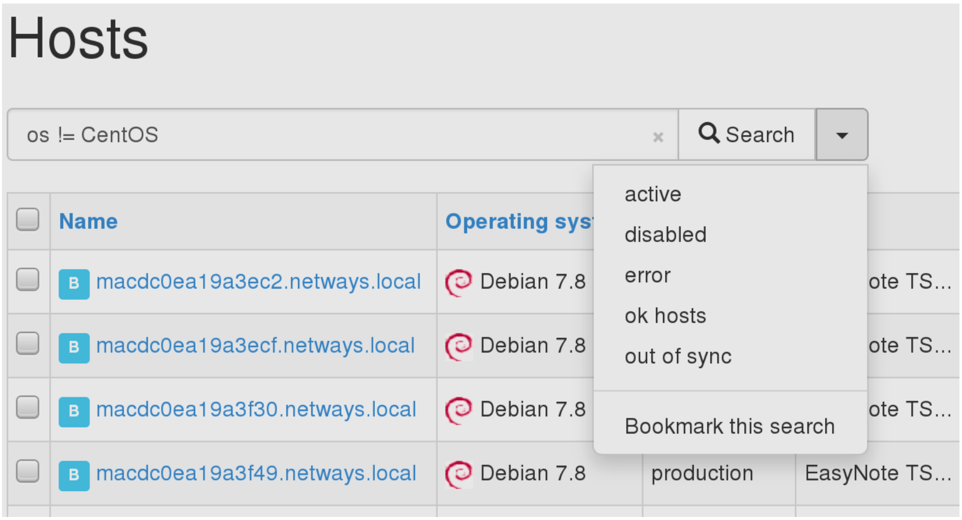
8 Advanced Topics

© NETWAYS

WebGUI

Searches

- Almost all views are simple lists
- Some views like "Reports" have a default filter
- Most views provide additional searches
- Bookmark and share your own search



	Name	Operating sys		
	macdc0ea19a3ec2.netways.local	Debian 7.8		ote TS...
	macdc0ea19a3ecf.netways.local	Debian 7.8		ote TS...
	macdc0ea19a3f30.netways.local	Debian 7.8		ote TS...
	macdc0ea19a3f49.netways.local	Debian 7.8	production	EasyNote TS...

continued...

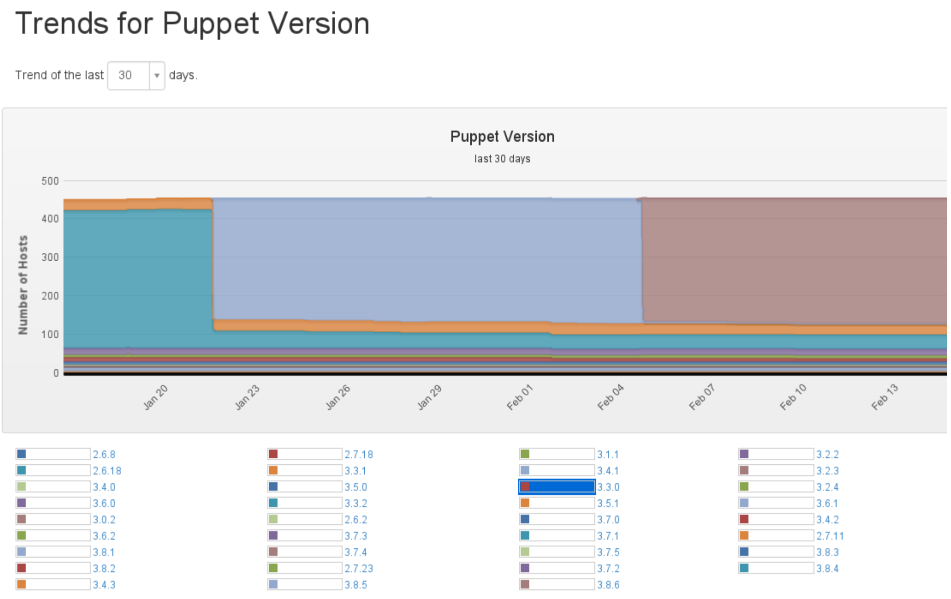
In the Foreman WebGUI almost all views are simple unfiltered lists, some views like the "Reports" view are filtered by default using a search and most provide additional searches to quickly filter for typically views like 'all hosts out of sync'. But the WebGUI is not limited to these searches you can always create your own and bookmark it for later. Setting a bookmark to public allows to share it with other users.

The search field can be used for a free text search but gets more powerful if using the autosuggestions. It provides different comparison operators depending on the type of the field compared including SQL like wildcard matching.

For more details see:
http://theforeman.org/manuals/latest/index.html#4.1.5Searching

Trends

- Graphs changes in your enviroment
- Configured in the WebGUI and collected by a cronjob
- Internal Host parameters or Facts provided by Configuration Management



Trends for Puppet Version

Trend of the last 30 days.

continued...

© NETWAYS

Foreman can provide graphs about changes in your environment. Those are configured and displayed in the WebGUI in "Monitor > Trends" and a cronjob is performs the data collection. By default this cronjob runs every 30 minutes matching Puppet's default run interval because most trends will be based on the facts collected by the configuration management solution, another source are Foreman's internal parameters like the operatingsystem of the host.

Cleanup / Backup / Restore / Upgrade

© NETWAYS

Cleanup

- Reports are send to the Foreman and stored in the database
  - Puppet reports every 30 minutes
  - Chef reports every 30 minutes
  - Salt reports every 10 minutes
  - Ansible reports on every run
  - Requires a lot of space

- Cleanup job is provided by the Foreman
  - Execute via cron
  - Can run with different parameters

The configuration management solution sends reports to Foreman which are then stored in the database afterwards. Depending on the solution, configuration and interval the required storage space can differ.

Foreman provides a cleanup job for this. Best Practice would be an execution as a cronjob. The command takes parameters for the maximum age to keep the status of the reports.

continued...

© NETWAYS

Depending on your needs configure a daily cronjob like this to delete all reports 'without event', 'after one day' or those with events 'after 7 days'.

```
#!/bin/sh
foreman-rake reports:expire days=1 status=0
foreman-rake reports:expire days=7
```

~

© NETWAYS

Backup

- In a regular interval, at least before an upgrade!

- Foreman
  - Configuration - Archive the configuration directory
  - Database - Backupjob provided by Foreman

- Puppet
  - CA - Archive the certificates
  - Puppet Code - Archive the modules

- Smart proxies
  - Smart proxies itself - Archive the Smart proxy configuration
  - Managed Service - Follow instructions for the service

Backup should be done in a regular interval, but at least performed before any upgrade, and it should cover all components.

continued...

Foreman backup can be done by archiving the configuration directory "/etc/foreman" and for the database a dump can be generated with the following command "foreman-rake db:dump" provided by the Foreman.

The Puppet backup should include the certificates which are located in "/var/lib/puppet/ssl" on the Puppet CA server and the Puppet Code underneath "/etc/puppet/environments". Other configuration management solutions will be handled in a similar way.

To backup the Smart proxy, archive the folder "/etc/foreman-proxy" and do not forget about the managed service. For this follow the instructions for the service. With the "Orchestration rebuilder" feature, the Foreman can also rebuild all configuration issued via the Smart proxy from the "All Hosts" menu as an action.

~

Restore

- <u>Always stop the service before doing any restore!</u>
- Foreman
  - Configuration - Restore the configuration directory
  - Database - Restorejob provided by Foreman
- Puppet
  - CA - Restore the certificates
  - Puppet Code - Restore the modules
- Smart proxies
  - Smart proxies itself - Restore the Smart proxy configuration
  - Managed Service - Follow instructions for the service

Before starting a restore always stop the service!

continued...

© NETWAYS

Restore the configuration directory of the Foreman carefully and inspect it before overwritting the current one. The database dump can be restored with the command "foreman-rake db:import_dump file=/usr/share/foreman/db/foreman.TIMESTAMP.sql". Drop an existing database in advance to have a clean restore.

For Puppet restore simply copy back the files, the same goes for the Smart proxy. The managed services should be restored according to their instructions.

~

© NETWAYS

Upgrade

- Always follow the instructions in the Foreman manual!

- In General:
  - Backup
  - Change package repository to the new release
  - Update the packages
  - Run Database migration and seed script
  - Clear the cache and sessions
  - Optionally run the foreman-installer to verify
  - Restart the service

Always follow the instructions in the Foreman manual providing release and operating system specific steps to do.

continued...

In general you should start by creating an up to date backup of the old configuration. Afterwards you have to change the package repository to the newest release because Foreman is always providing a separate repository for any major release. Then cleanup the package metadata and update the packages. It should run the database migration and seed script during the package update but if running them manually will show you any error then clear the cache and existing sessions. As a last step before restarting the service you can optionally run the foreman-installer in simulation mode to verify the installation and see pending config changes, if some are shown run the foreman-installer again to apply them finally.

~

API

© NETWAYS

API

- Version 1 of the API is deprecated
- Version 2
    - JSON API
    - Shared with Katello
    - Requires username and password
    - Collections are paged
    - Search strings like provided in the WebGUI

Foreman provides a web based JSON API which is shared with Katello. This is Version 2 of the API which is favored instead of the already deprecated version 1. It requires an authentication by username and password. Perhaps unusal is the handling of collections which are paged in the same way like in the webinterface. The same way it handles search strings like they are provided in the WebGUI but they have to be URL encoded.

Usage explainations can be found at
http://theforeman.org/manuals/latest/index.html#5.1API

API documentation is located at
http://theforeman.org/api/1.10/index.html

© NETWAYS

Lab 8.1: Working with the API

- Objective:
  - Use the API to query, create and update objects

- Steps:
  - Query the API for all subnets
  - Query the API for all Debian hosts
  - Create a hostgroup "training" using the API
  - Rename the hostgroup "training" to "renamed" using the API

© NETWAYS

CLI

© NETWAYS

Hammer CLI

- Written in Ruby based on clamp
- Modular CLI
  - Basicly a framework
  - Plugins for Foreman
  - Some Foreman plugins provide a Hammer plugin
  - Other tools (especially around Katello)
  - Create your own plugin

The commandline interface for Foreman is based on the Hammer CLI which is basicly a framework written in Ruby based on clamp. Foreman provides a plugin to manage most aspects of Foreman like the WebGUI and the API. Some of the Foreman plugins also provide a Hammer plugin, like some other tools especially around Katello do. But the framework is not limited to the Foreman environment, so feel free to create your own plugins to solve your own administrative tasks.

A list of plugins is provided by the Github page of the framework: https://github.com/theforeman/hammer-cli

Lab 8.2: Working with the CLI

- Objective:
  - Use the CLI to prepare a new Operatingsystem entry

- Steps:
  - Create the new Operatingsystem entry for "CentOS 6.8"
  - Associate the template "Kickstart default PXELinux" and set it as default template
  - Associate the template "Kickstart RHEL default" and set it as default template

© NETWAYS

Multitenancy

© NETWAYS

Multitenancy

- Adds Organizations and/or Locations
    - Can be nested

- All Objects require one context

- User
    - can have multiple contexts
    - can have a default context
    - keep care of in which context he works

Foreman has build in multitenancy which is not enabled by default. To enable it run the Foreman Installer with the corresponding parameters or edit "/etc/foreman/settings.yaml".

Depending on your needs you can add Organizations and/or Locations. After that every object will require exactly one context with the exception of users who can have multiple contexts and one of these assigned as their default context. But he has to keep care of in which context he works because objects he creates will be in this context.

Organization and Location can both be nested to represent a hierarchy with a top-down approach.

continued...

© NETWAYS

The best way to think about multitenancy is in advance and if it could be required directly from the start of your enviroment. Later enabling is possible but migration is a quite challenging and time consuming task.

In Katello it is enabled by default, so we will see it in the last chapter.

Troubleshooting

© NETWAYS

Logging

- Foreman
  - Increase log level in general
  - Enable additional specific logs
- Smart proxy
  - Increase log level
- Plugins can add their own logging

Both Foreman and the Smart proxy write an excellent log file.

Foreman log files are located in "/var/log/foreman" and its main log is called "production.log". You can find all accessed URLs and errors generated by accessing it. If the standard log level does not provide any information it is possible to increase the log level by changing the configuration in the "/etc/foreman/settings.yaml".

```
:logging:
  :level: debug
```

continued...

© NETWAYS

Furthermore it is possible to enable additional specific loggers.

- app - web requests and all general application logs (default: true)
- ldap - high level LDAP queries (e.g. find users in group) and LDAP operations performed (default: false)
- permissions - evaluation of user roles, filters and permissions when loading pages (default: false)
- sql - SQL queries made through Rails ActiveRecord, only debug (default: false)

For those add following to the configuration:

```
:loggers:
  :ldap:
    :enabled: true
```

The log files of the Smart proxy are located in "/var/log/foreman-proxy" containing all errors issued by the Smart proxy like downloading of boot images failed and failed updates of services. If additional details are needed increase the log level in "/etc/foreman-proxy/settings.yml".

```
:log_level: DEBUG
```

Plugins to both components can add directly to the log or write their own, so have a look into their configuration if attempting any troubleshooting with a specific plugin.

Provisioning process

- Watch the output in the WebGUI
- Spoof the provisioning configuration
- Find the communication in Foreman log
- In the Smart proxy log
    - Download of boot images
    - Configuration changes failing
- On the TFTP server
    - Checksum of boot image
    - PXE configuration
- Reports of configuration management

Troubleshooting of the provisioning process can be required if deployment of a system fails or differs from expectation. In most cases the output of the WebGUI is quite useful, but sometimes a deeper look is required to find a solution.

You can get the provisioning configuration files by spoofing the URL:

```
https://foreman.localdomain/unattended/provision?spoof=10.0.0.100
```

continued...

© NETWAYS

In this URL you can change the template type provision to every other type to get those files.

If everything here seems to be correct, check the communication in the Foreman log. Getting the exact URLs from the log and browsing them can help to find issues like token on a boot disk expired.

Next step would be to look for errors in the Smart proxy log if the download of the boot images or some configuration changes issued to the managed services have failed.

On the TFTP server have a look at the boot image, it is downloaded only once during the first deployment of a host started. If somehow the image was already downloaded you will not find any new log entry about the failed download. Also in some rare cases distribution releases a newer version of the boot image to fix some installation issues. So check the size and checksum of the image if the boot fails.

In the case there is no PXE configuration you can cancel and restart the build process or execute a rebuild from the "All Hosts" menu.

If the deployment of the system works but something is not configured in the expected way have a look in the reports of the configuration management. In many cases Puppet code is not well designed or has some cross system dependencies so the agent has to run more than once to complete the full configuration.

Upstream support

- Common issues in the Wiki
- IRC
  - User channel for support
  - Developer channel for development support
- Mailing lists
  - Users list for support, questions, etc
  - Developer list for development support
- Server fault
- Youtube
- Issue tracker
- Debug tool

The Foreman team tries to help with issues as good as they can. To achieve this there are several ways to get support available.

First have a look into the Wiki page collecting common issues:
http://projects.theforeman.org/projects/1/wiki/Troubleshooting

continued...

© NETWAYS

If your problem is not covered there the fastest way to get in touch with the team is the IRC Channel. The channel #theforeman is hosted on irc.freenode.net, the developer channel #theforeman-dev is only for support on development of Foreman and plugins.

Same goes for the mailing lists hosted via Google Groups. To subscribe simply send a email to foreman-users+subscribe@googlegroups.com for the user list, to foreman-dev+subscribe@googlegroups.com for development support and for release and security information to foreman-announce+subscribe@googlegroups.com.

On 'Server Fault' you can ask questions and find answers tagged with foreman: http://serverfault.com/questions/tagged/foreman

On Youtube in the Foreman channel you can find introductions, deep dives, community demos and other ressources which can also be very useful: https://www.youtube.com/channel/UCCo7AZ1oG6TbG0-dwjRqCmw

Looking at all those resources will solve most issues, if not it is probably a bug, so search for it in the issue tracker and if you can not find a similar issue file create a new one with as much information as possible to help fixing the bug and others with the same issue. http://projects.theforeman.org/projects/foreman/issues

Sometimes you could be asked to send a debug report which could be generated with "foreman-debug" including the complete configuration stripped of security data. With "foreman-debug -u" it could be uploaded to a location only accessable by the Foreman core developers.

© NETWAYS

9 Katello

© NETWAYS

Katello

- Defined set of Foreman plugins to provide Content management:
    - YUM repositories (RPM + Errata)
    - Puppet modules
    - Docker container
- Subscription Management
    - Red Hat Subscriptions
    - Track internal use
- Provides its own installer
- Multitenancy enabled by default
- Upstream project for Red Hat Network Satellite 6

Katello is a defined set of Foreman plugins which add Content management and Subcription management.

The Content management feature can manage YUM repositories to provide software in RPM package format and errata explaining the importance of the package updates, puppet modules for configuration management and docker container for deploying application container.

continued...

The Subscription management feature allows to subscribe to a software distributor which will be in most cases Red Hat. It also provides the possiblity to track the internal use of some software which can be very useful for support contracts or with multitenancy to handle customer environments.

It uses the same technology for installation as the Foreman but provides its own installer. The usage of this installer is the supported way of installation for Katello, so there is way of adding the plugin later to an existing Foreman installation for now.

It alters Foreman in several ways including enabling multitenancy by default to enable internal Subscription managment.

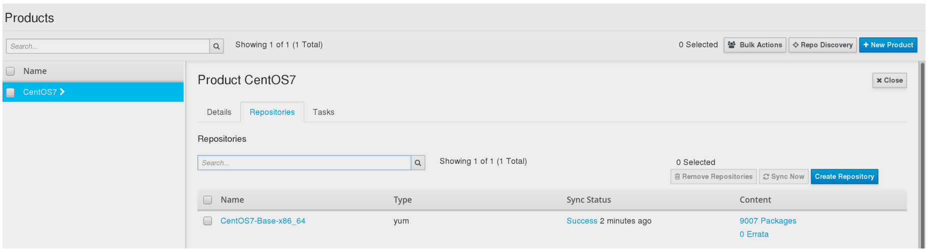
Katello is also the upstream project for Red Hat Network Satellite 6 like Spacewalk was for Satellite 5.

Detailed information on the project homepage:
http://www.katello.org/

© NETWAYS

Content management - Products

- Content management is based on products
- Products consists of one or more repositories of any content type



Name	Type	Sync Status	Content
CentOS7-Base-x86_64	yum	Success 2 minutes ago	9007 Packages 0 Errata

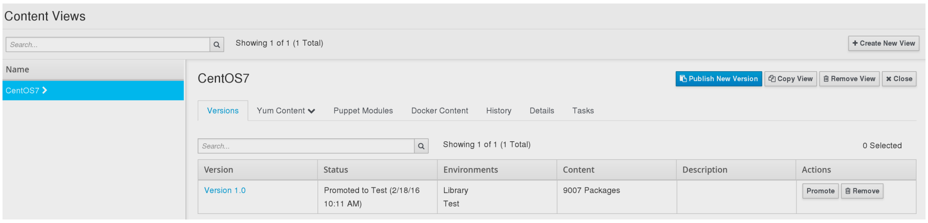
In Katello the Content management is based on products which consist of one or more repositories of any content type.

This allows different configurations depending on your focus. For example you can create a product named after the operatingsystem you run and add all repositories you require and have a second product named Puppet containing all Puppet modules in use. Or you can create a product named after your application containing all repositories for every operatingsystem you use and one for the puppet module to manage the application itself.

© NETWAYS

Content management - Content Views

- Content Views create versioned snapshots of one or more repositories
- Allow to filter several software packages
- Composite Content Views are possible



Content Views

Search... | Showing 1 of 1 (1 Total) | + Create New View

Name

CentOS7

CentOS7

Publish New Version | Copy View | Remove View | Close

Versions | Yum Content | Puppet Modules | Docker Content | History | Details | Tasks

Search... | Showing 1 of 1 (1 Total) | 0 Selected

Version	Status	Environments	Content	Description	Actions
Version 1.0	Promoted to Test (2/18/16 10:11 AM)	Library Test	9007 Packages		Promote | Remove

A Content View allows to create a versioned snapshot of one or more repositories and if required to filter specific packages or problematic versions of a package. New versions of a content view can be published afterwards and be promoted to a Lifecyle Environment.

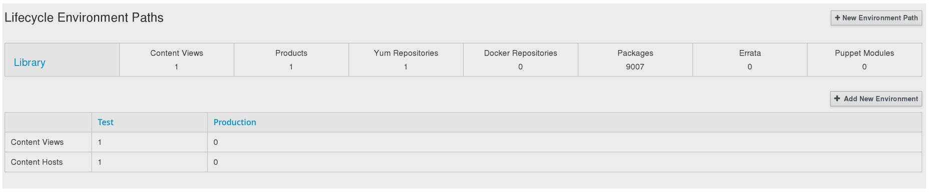
Incremental updates allow to push a security hotfix or something similar directly to all snapshots.

Composite Content Views are possible to combine existing Content Views in one.

© NETWAYS

Content management - Lifecycle Environments

- Lifecycle Environments represent different stages
- Multiple Lifecycle Environments Paths allow for different staging models


Lifecycle Environment Paths

Library	Content Views	Products	Yum Repositories	Docker Repositories	Packages	Errata	Puppet Modules
	1	1	1	0	9007	0	0

	Test	Production
Content Views	1	0
Content Hosts	1	0

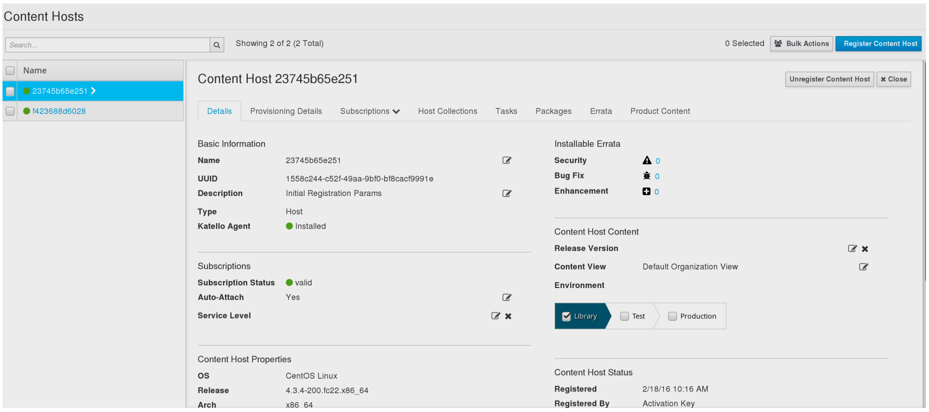
Lifecycle Environments represent different stages and are connected to a path which only allows to promote a Content View to a stage after it has hit all stages before on the path and of course was tested there successfully.

You can create multiple Lifecycle Environments Paths to allow different staging models representing different workflows of projects or departments.

© NETWAYS

Content management - Content Hosts

- Consumer of content and subscriptions - different view on a host than Hosts in Foreman
- Optional Agent allows to execute commands



Content Host 23745b65e251

Basic Information

Name	23745b65e251
UUID	1558c244-c52f-49aa-9bf0-bf8cacf9991e
Description	Initial Registration Params
Type	Host
Katello Agent	● Installed

Subscriptions

Subscription Status	● valid
Auto-Attach	Yes
Service Level	

Content Host Properties

OS	CentOS Linux
Release	4.3.4-200.fc22.x86_64
Arch	x86_64

Installable Errata

Security	0
Bug Fix	0
Enhancement	0

Content Host Content

Release Version	
Content View	Default Organization View
Environment	[✓ Library] Test Production

Content Host Status

Registered	2/18/16 10:16 AM
Registered By	Activation Key

The Content Host is a different view on the host representing it as consumer of content and subscriptions for Katello. The Host in Foreman is used for provisioning and configuration management. A host can be represented as both or only one in Katello.

continued...

© NETWAYS

An optional agent allows to execute commands issued in the WebGUI on the host like installing a package or updating all packages. A chat is used for this feature allowing for just in time execution.

By creating an Activation Key and using it to register a Content Host setting defaults like consumed products is possible.

Subscription management

- Red Hat
  - Subscription Manifest - certificates for communication and subscription data
  - Red Hat Repositories - available products and repositories
  - Products and Repositories on click
  - Content Hosts consume subscriptions
- Other software
  - Tracks subscribed Content Hosts for products

Subscription management is primarily implemented for cosuming Red Hat Content you subcribed.

Red Hat allows to create a Subscription Manifest to move Subscription management from Red Hat Network to your environment. This manifest includes the certificates required for communication and the subscription data. Red Hat Repositories View shows available products and repositories covered by the subscription data and allows to select them which will automatically create and synchronize them to your system.

continued...

The Content Hosts cosume these subscriptions in the same way like they were directly connected to the Red Hat Network. If you purchased Virtual Subscriptions it requires you to run virt-what and report your virtualization hosts before you can consume subscriptions with the virtual machines.

For other software Subscription Management can also be useful by simply tracking all subscribed Content Hosts for products.

© NETWAYS

Demo

Katello Demo

Setting up Katello is only a bit more challenging then Foreman, but the team provides an easy way to create a demo setup.

Requirements:

- git
- vagrant (libvirt or virtualbox provider)
- docker and docker-compose

The repository https://github.com/Katello/katello-deploy/ includes everything else you need. Simply clone it with git, use vagrant to provide a Katello server and docker-compose to create Content Hosts.

Contributors

David Okon

Dirk Götz

Markus Waldmüller

© NETWAYS