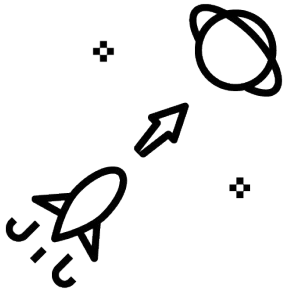
Why does THIS node have THAT config?

Martin Alfke
<ma@betadots.de>



Martin Alfke
CEO/Consultant/Trainer at betadots GmbH
Berlin, Germany

- Puppet Trainer and Puppet Solution Engineer
- Platform Engineering, Consulting and Training
- Agile methods, Scrum
- tuxmea (Twitter, GitHub, Slack)

Why does THIS node have THAT config?

or "How to do Hiera in Puppet and how to analyze data"

IT infrastructure consists of several nodes in different stages with different use cases.

Some configurations are identical, whereas others must be different.

Store these information differences in different files and directories as data in YAML structure.

Puppet has a built in data lookup tool (Hiera) to identify data location and read data values.

Why does THIS node have THAT config?

Why are systems different?

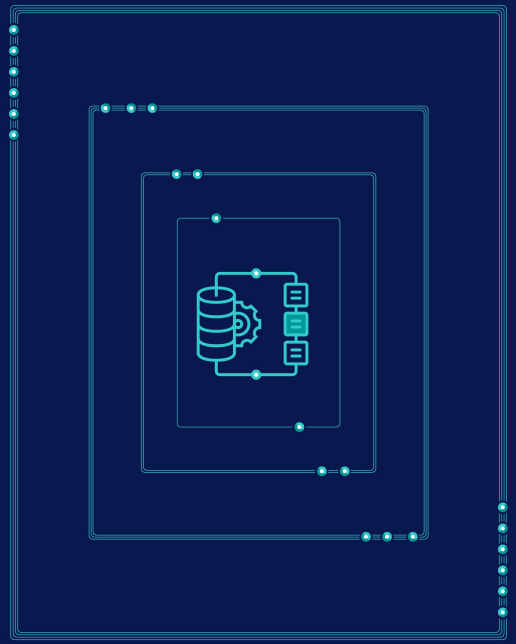
- prod servers must use prod db, dev servers must use dev db

- servers in network zone G use different DNS

- one of the load balanced web servers has an additional local script

There are many reasons why a systems configuration differs compared to another system.

With Puppet one can implement these differences as Hiera data.



Hiera Basics

Hiera basics - What is a hierarchy?

Hiera supports different data backends.

Just be sure:

- data lookups occur very often and long data lookup timings will result in higher load on your Puppet server.

It is recommended to have a very fast data backend.

The fastest backend is local files in YAML (https://yaml.org/) or JSON (https://json.org/) syntax.

Hiera basics - What is a hierarchy?

Hierarchies are a directory/file structure.

This structure is built from individual to generic layer.

Layers are indicators for configuration differences.

From our example:

- `one node = fqdn/certname`
- `prod/dev = stage`
- `network zone G = zone`
- `global`

Hiera basics - What is a hierarchy?

beladots

You can now view these layers as sheets with information.

1. On the ground we place the global sheet.
2. Then we add the zone sheet above the global sheet.

The zone sheet can show the information from the global sheet, extend global information or overwrite global information.

Same happens with all the remaining layers.

Now hiera looks from top and reads the data it sees.

Hiera basics - What is a hierarchy?

Global layer covers all nodes within a single file

The node layer is based on one file per host.

Just to be sure: You don't need to provide all files, only the ones where you have differences.

On node layer multiple files might be needed.

Createc a clear directory structure makes it easier to identify file usage.

Hiera basics - What is a hierarchy?

beladots

File structure of layers inside directories:

```
nodes/
  certname
apps/
  application-stage
app/
  application
zone/
  zone
global
```

Hiera basics - What is a hierarchy?

beladots

But what about differences?

Each node has its own set of Puppet Facts.
'facter' information are collected on the node and send to Puppet server.

Puppet code can access these facts.
Access to facts is also possible within hiera.

Hiera basics - What is a hierarchy?

The most important tasks are to identify

- why nodes differ and
- if nodes with same configuration can be grouped and
- if you can provide a name for the group.

Within the hiera configuration file (hiera.yaml) we can access Puppet variables using another syntax compared to Puppet code.

Hiera basics - What is a hierarchy?

For the node layer we can use the FQDN which is `%{facts.networking.fqdn}`. Within Puppet it is even better to use `%{trusted.certname}` instead.

For applications we can extend facter either by external facts or by custom facts or the usage of csr_attributes is possible.

Application: `%{betadots_facts.application}` or `%{trusted.extension.pp_application}`

Stage: `%{betadots_facts.stage}` or `%{trusted.extension.pp_stage}`

Zone: `%{betadots_facts.zone}` or `%{trusted.extension.pp_stage}`

Hiera basics - What is a hierarchy?

We can now finalize the layout of the hiera layers:

```
nodes/%{trusted.certname}.yaml
apps/%{betadots_facts.application}-%{betadots_facts.stage}.y
aml
app/%{betadots_facts.application}.yaml
zone/%{betadots_facts.zone}.yaml
global.yaml
```

Hiera basics - What is a hierarchy?

hiera.yaml file:

```
---
version: 5

defaults:
  datadir: data

hierarchy:
  - name: "yaml hierarchy"
    paths:
      - "nodes/%{trusted.certname}.yaml"
      - "apps/%{betadots_facts.application}-%{betadots_facts.stage}.yaml"
      - "app/%{betadots_facts.application}.yaml"
      - "zone/%{betadots_facts.zone}.yaml"
      - "global.yaml"
```

Hiera basics - What is a hierarchy?

betadots

common.yaml

```
---
motd: 'Welcome to
  example.com'

ntpserver:
  'us.pool.ntp.org'

yumrepo:
  'yum.example.com'

mysql_rootpw:
  'p@ssw0rd'
```

Image ©Puppet Inc.

Hiera basics - What is a hierarchy?

houston.yaml

```
---
motd: ‘Location:
  Houston Datacenter’


yumrepo:
  ‘houston.yum.example.com’
```

common.yaml

```
---
motd: ‘Welcome to
  example.com’

ntpserver:
  ‘us.pool.ntp.org’

yumrepo:
  ‘yum.example.com’

mysql_rootpw:
  ‘p@ssw0rd’
```

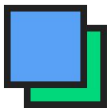


Image ©Puppet Inc.

Hiera basics - What is a hierarchy?

beladots

node1.example.com.yaml

```
---


mysql_rootpw:
  'hunter2'
```

houston.yaml

```
---
motd: 'Location:
  Houston Datacenter'


yumrepo:
  'houston.yum.example.com'
```

common.yaml

```
---
motd: 'Welcome to
  example.com'

ntpserver:
  'us.pool.ntp.org'

yumrepo:
  'yum.example.com'

mysql_rootpw:
  'p@ssw0rd'
```

Image ©Puppet Inc.

Hiera basics - What is a hierarchy?

Please note that a new layer is added very quickly.
Reducing your number of hierarchies, is work intense, as one must check if data needs to be overwritten or of data are shared data.

Please consider your hierarchies carefully, try to keep the number of layers as low as possible and as large as required.

Hiera basics - What is a hierarchy?

From now on Puppet can query for data.

In general there are two options:

- Automatic data binding
- Explicit lookup

Many people do explicit lookups, but automatic data binding is faster and more simple and less to type.

Hiera basics - Classes with parameters

Within Puppet one can specify parameters within the class header.

The parameters consist of the following entries:

```
class foo (
  [DataType] $parameter_name = 'default',
) {
  # Puppet DSL
}
```

Hiera basics - Classes with parameters

By adding the optional data type we can easily validate if an application admin has provided correct data.
As Puppet code developer we want to be sure that if we expect a bool value, we receive a bool value.

Puppet has several core data types, like `Boolean`, `String`, `Integer`, `Float`, `Array`, `Hash`, `Regexp` and many more.

Several modules build their own data types (mostly based on regular expressions), like `Stdlib::Absolutepath`

Hiera basics - Classes with parameters

The parameter name must start with a dollar sign followed by a lower case letter afterwards one can use digits, lower case letters and underscores.

The default value is optional. If one does not specify a default value, the responsible application admin must provide a value using hiera data. If the admin does not set the required data, Puppet compiler will produce an error.

By this a Puppet developer can control that required information are not forgotten.

Hiera basics - Classes with parameters

An example:

```
class knowhow (
  String[1] $type,
  Boolean   $enable_brain   = true,
  String[1] $knowledge_base = 'dictionary',
) {
  # Puppet DSL
}
```

Whenever the class `knowhow` gets declared (e.g. using `include knowhow`), Puppet will automatically query Hiera for data.

Hiera basics - Classes with parameters

In our example:

```
knowhow::type
knowhow::enable_brain
knowhow::knowledge_base
```

Within Hiera we can now overwrite the default values and set required values:

```
# stage/dev.yaml
---
knowhow::type: 'human'
knowhow::enable_brain: false
knowhow::knowledge_base: 'wikipedia'
```

Hiera basics - Scope of data

So far we have not talked about the data location.

There are three locations (scopes) available and Puppet queries them in the provided order:

1. Global scope
2. Environment scope
3. Module scope

Hiera basics - Scope of data

Global scope:

- Hiera config version 3 (is kept in place for migration).
-

In a modern Puppet environment one should only use the environment or the module scope.

Global scope is configured in `/etc/puppetlabs/puppet/hiera.yaml`.
To disable global scope, one can ensure that the hiera.yaml file is empty. Puppet Enterprise customers must ensure, that the file exists and has Puppet Enterprise Console Data Lookup.

Hiera basics - Scope of data

Environment scope:

Configured by a hiera.yaml inside the control repo.

This allows a seamless integration as you write your profiles and integrate library modules.

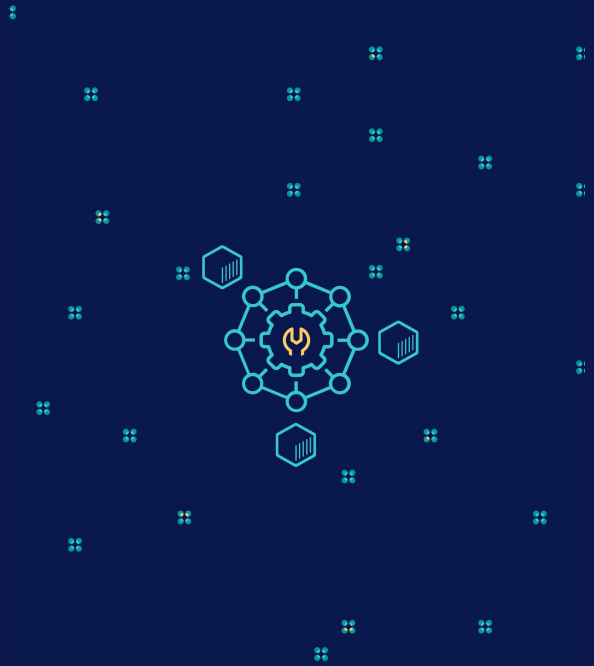
If you need different access rights between the control repository and the data structure, one can place the data into a separate git project and integrate the data in Puppetfile (similar to a module).
Please don't forget to also set the `datadir` configuration.

Hiera basics - Scope of data

Module scope:

If you are developing modules and you need to provide different OS package names, you can place the data in the module.
You only need to add a hiera.yaml config version 5 to your module.

Please note that within a module only data for that specific module can be placed.



Hiera lookup validation

Hiera lookup validation

Whenever you want to deploy a partial change using hiera data, it might happen, that another node receives the change, too.

Or:

You plan a change and the node does not get the data you were considering.

But why?

Usually this happens when we have many data and lots of hierarchy layers and somewhere data where placed into the wrong file.
Analyzing hiera lookup manually can be very time consuming.

Hiera lookup validation

One needs to

- review hiera.yaml file and check the facts which build the layers
- read the facts from a node and does in-memory replacement of facts
- parse each of the layers, check if a file is available

And please don't forget to first have a look if the key your are analyzing has a `lookup_option` set.
https://www.puppet.com/docs/puppet/latest/hiera_merging#merge_behaviors

Hiera lookup validation

On the other hand there are two utilities which help checking hiera data lookups:

- puppet lookup --explain
- Hiera Data Manager

Hiera lookup validation - Puppet lookup

Puppet lookup:

- built in command line utility to query data from hiera which must be run on the Puppet server.

The Puppet lookup command **must** be run as `root` user!

One can pass arguments like node (FQDN), environment to read data from, provide a json file with facts, set the merge behavior.

The output from puppet lookup will just show the result, but not the way, how the result was found.

One can also get this information by using the flag `--explain`.

Hiera lookup validation - Puppet lookup

Please note that you must also add the flag `--compile` if you use top scope variables in your `hiera.yaml` file (e.g. from `manifests/site.pp`).
If your code does not compile you can not analyze Hiera data.

We therefore recommend to only use facts in `hiera.yaml` file.

The output is difficult to parse.

Hiera lookup validation - Hiera Data Manager (HDM)

Hiera Data Manager (HDM) is an Open Source web application which is publicly available on GitHub.

https://github.com/betadots/hdm

The source code is licensed under the AGPL-3.0 license.

HDM allows you to view your node data or search for data in an environment.

Hiera lookup validation - Hiera Data Manager (HDM)

To make use of HDM some requirements **must** be met:

- hiera config v5 (environment scope only)
- properly configured and integrated puppetdb (facts upload)
- read access to puppetdb (http or https)
- read access to fully deployed code (r10k deploy)
- read access to eyaml private key (only if decryption is activated)

Hiera lookup validation - Hiera Data Manager (HDM)

Besides this HDM has some limitations:

- no top scope data (if they are there, they are not shown)
- no module data
- no execution of functions in hiera data
- needs up-to-date Ruby version (use HDM in container!)

Hiera lookup validation - Hiera Data Manager (HDM)

At the time of writing the following features are available:

- Usermanagement
  - LDAP
  - SAML
  - Local
  - without
- Security
  - RBAC
    - Node
    - Environment
    - Key
    - Encryption
- Security (continued)
  - EYAML usage
  - Read/Write mode
    - write goes to a git repository
- API
  - Foreman HDM Smart-Proxy
  - Foreman HDM View
- Data analysis
  - Search for a node value
  - Search for a key in an environment

Hiera lookup validation - Hiera Data Manager (HDM)

There are two ways how to run HDM:

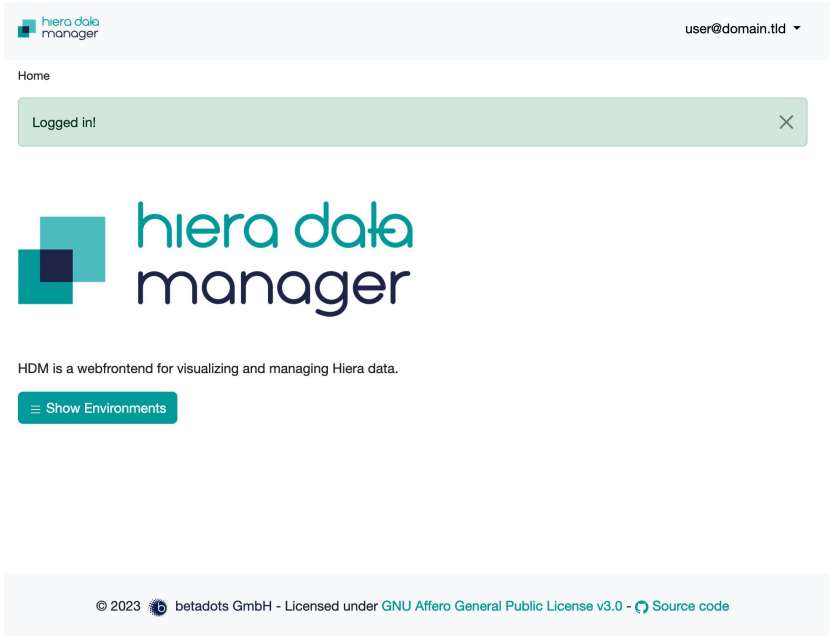
- RVM-based
- Container based

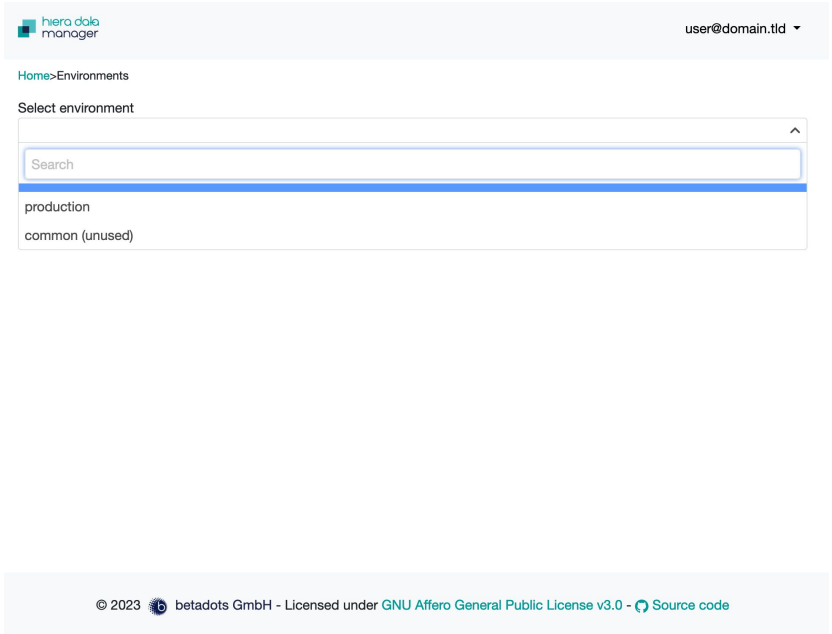
The RVM based solution is used by developers who want to collaborate.
The Container based solution is already successfully in use in several production environments.

The complete setup of HDM can be automated using the voxpupuli-hdm module

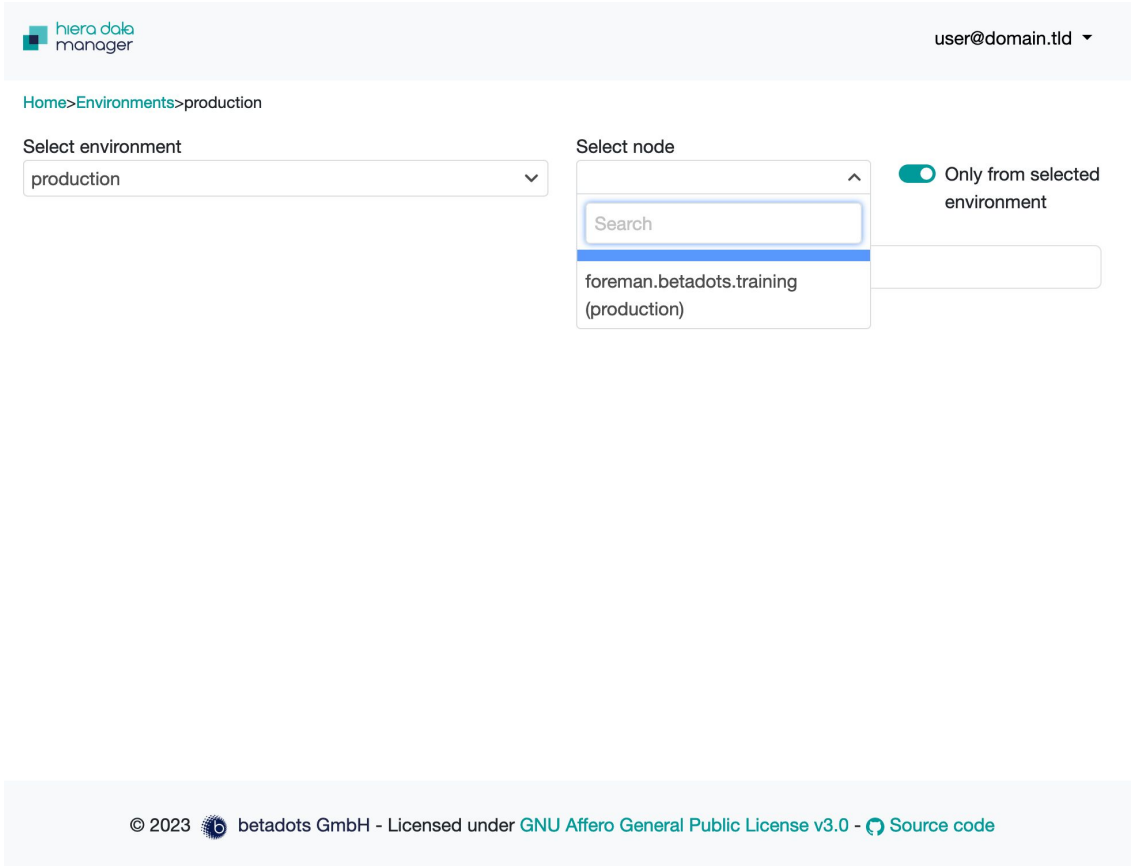
https://forge.puppet.com/modules/puppet/hdm

Hiera lookup validation - Hiera Data Manager (HDM)





Hiera lookup validation - Hiera Data Manager (HDM)



Hiera lookup validation - Hiera Data Manager (HDM)

beladots

hiera data manager

anonymous

Home>Environments>production>puppet.betadots.training

Select environment

production

Select node

puppet.betadots.training (production)

Only from selected environment

Search

- lookup_options
- classes
- hdm::disable_authentication
- hdm::version
- postgresql::globals::manage_dnf_module
- profile::base::additional_packages
- puppetdb::manage_firewall
- puppetdb::manage_package_repo
- puppetdb::postgres_version

Hiera lookup validation - Hiera Data Manager (HDM)

beladots

hiera data manager

anonymous

Home>Environments>production>puppet.betadots.training>lookup_options

Select environment

production

Select node

puppet.betadots.training (production)

Only from selected environment

Search

- lookup_options
- classes
- hdm::disable_authentication
- hdm::version
- postgresql::globals::manage_dnf_module
- profile::base::additional_packages
- puppetdb::manage_firewall
- puppetdb::manage_package_repo
- puppetdb::postgres_version

Lookup options: first | Show lookup result

Per-node data (yaml version) yaml

nodes/puppet.betadots.training.yaml

Other YAML hierarchy levels yaml

common.yaml

```
classes:
  merge: deep
```

Hiera lookup validation - Hiera Data Manager (HDM)

beladots

hiera data manager

anonymous

Home>Environments>production>classes

Select environment

production

Search for a key

classes

Search Results

Found key `classes` in 2 files.

Per-node data (yaml version) yaml - *1 file*

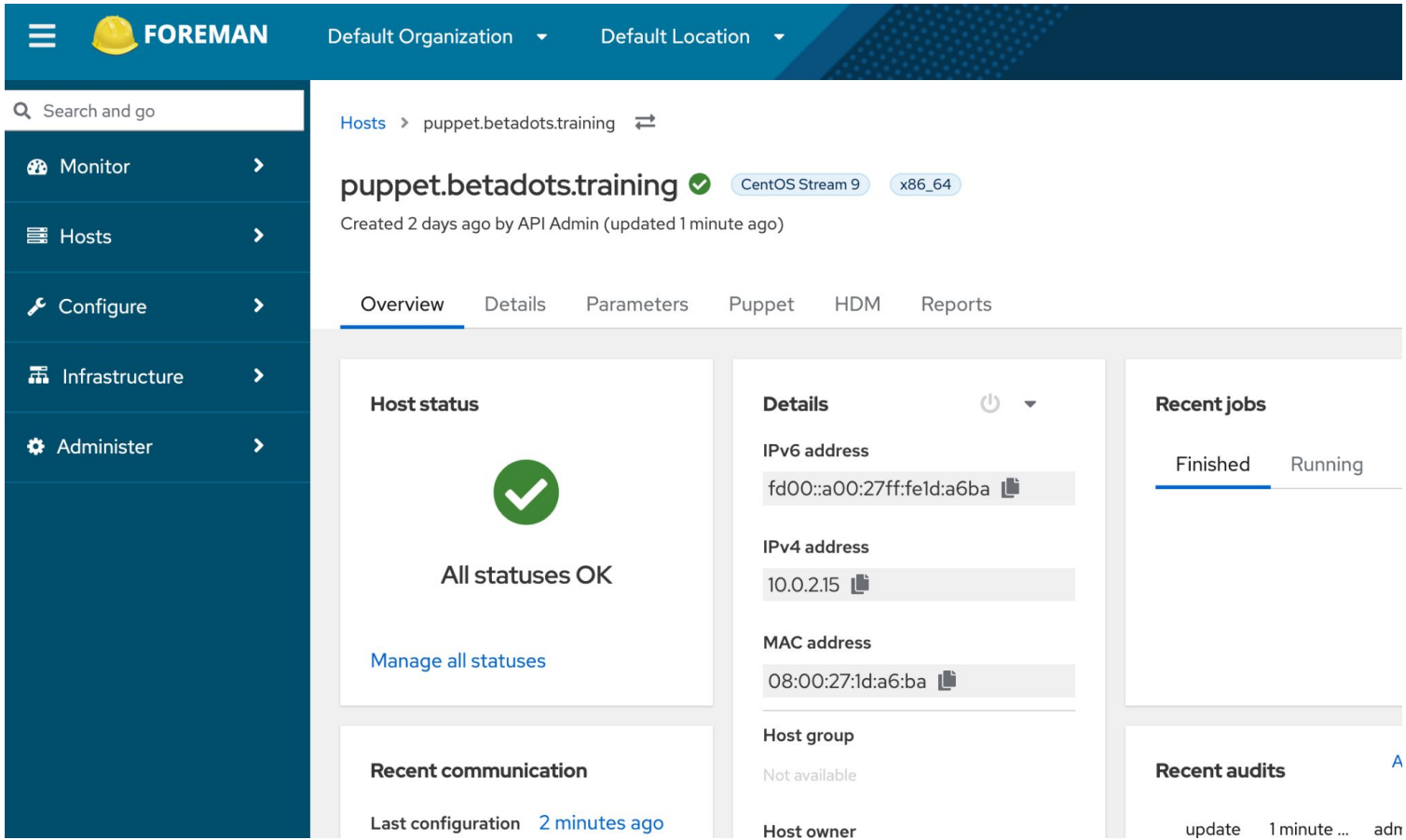
nodes/puppet.betadots.training.yaml

```
90_hdm_class: hdm
91_puppetdb_class: puppetdb
92_puppetdb_master_class: puppetdb::master::config
```

Other YAML hierarchy levels yaml - *1 file*

common.yaml

Hiera lookup validation - HDM-Foreman integration



Hiera lookup validation - HDM-Foreman integration

Hosts > puppet.betadots.training

puppet.betadots.training CentOS Stream 9 x86_64

Created 2 days ago by API Admin (updated 1 minute ago)

Schedule a job | Edit

Overview | Details | Parameters | Puppet | HDM | Reports

- lookup_options
- classes
- hdm::disable_authentication
- hdm::version
- postgresql::globals::manage_dnf_module
- profile::base::additional_packages
- puppetdb::manage_firewall
- puppetdb::manage_package_repo
- puppetdb::postgres_version

No key selected

Please select a key from the list on the left.

Hiera lookup validation - HDM-Foreman integration

Hosts › puppet.betadots.training

puppet.betadots.training CentOS Stream 9 x86_64

Created 2 days ago by API Admin (updated 1 minute ago)

Schedule a job | Edit

Overview | Details | Parameters | Puppet | HDM | Reports

- lookup_options
- classes
- hdm::disable_authentication
- hdm::version
- postgresql::globals::manage_dnf_module
- profile::base::additional_packages
- puppetdb::manage_firewall
- puppetdb::manage_package_repo
- puppetdb::postgres_version

Key: lookup_options

Per-node data (yaml version)

nodes/puppet.betadots.training.yaml

Other YAML hierarchy levels

common.yaml

```
classes:
  merge: deep
```

Hiera lookup validation - HDM-Foreman integration

betadots

Hosts > puppet.betadots.training

puppet.betadots.training CentOS Stream 9 x86_64

Created 2 days ago by API Admin (updated 1 minute ago)

Schedule a job | Edit

Overview | Details | Parameters | Puppet | HDM | Reports

- lookup_options
- classes
- hdm::disable_authentication
- hdm::version
- postgresql::globals::manage_dnf_module
- profile::base::additional_packages
- puppetdb::manage_firewall
- puppetdb::manage_package_repo
- puppetdb::postgres_version

Key: classes

Per-node data (yaml version)

nodes/puppet.betadots.training.yaml

```
90_hdm_class: hdm
91_puppetdb_class: puppetdb
92_puppetdb_master_class: puppetdb::master::config
```

Other YAML hierarchy levels

common.yaml

```
{}
```

Hiera lookup validation - Hiera Data Manager (HDM)

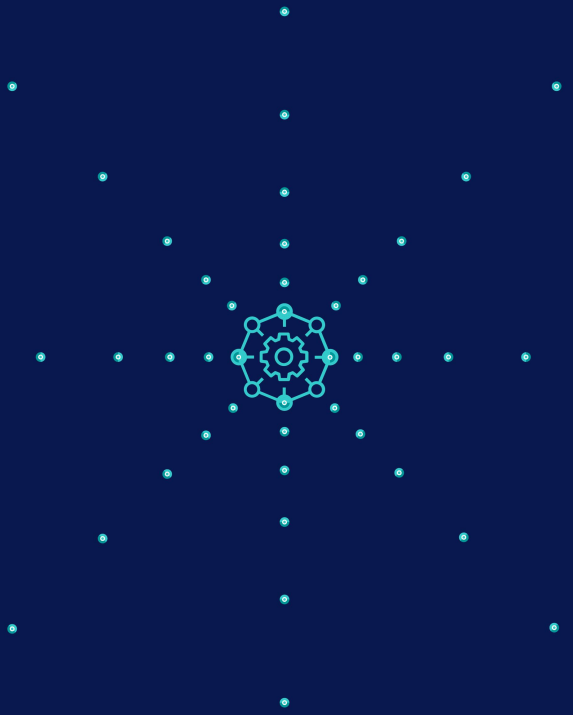
HDM planned features

Besides this we are talking about additional features:

- data from modules
- data result
- compare data between environments

Feel free to communicate your needs in our discussions

https://github.com/betadots/hdm/discussions



Summary

Summary

Hiera is super powerful.

It allows one to write clean Puppet code without large data processing parts.

People responsible for data don't need to understand Puppet on expert level. Basic knowledge is helpful and can be learned over time.

Also see our posting on "Puppet is YAML" https://dev.to/betadots/puppet-is-yaml-2e32

Hiera has an internal option to configure if the most specific result should be returned or if data must be merged. This gives additional flexibility for differences in platforms.

All YAML 1.1 internal specifications like anchors or aliases can be used.

Analysis and comparison can be done using Hiera Data Manager
Hiera and Hiera Data Manager will provide insight and details into flexible IT automation.

Why does THIS node have THAT config?

HDM provides insight.
Thank you