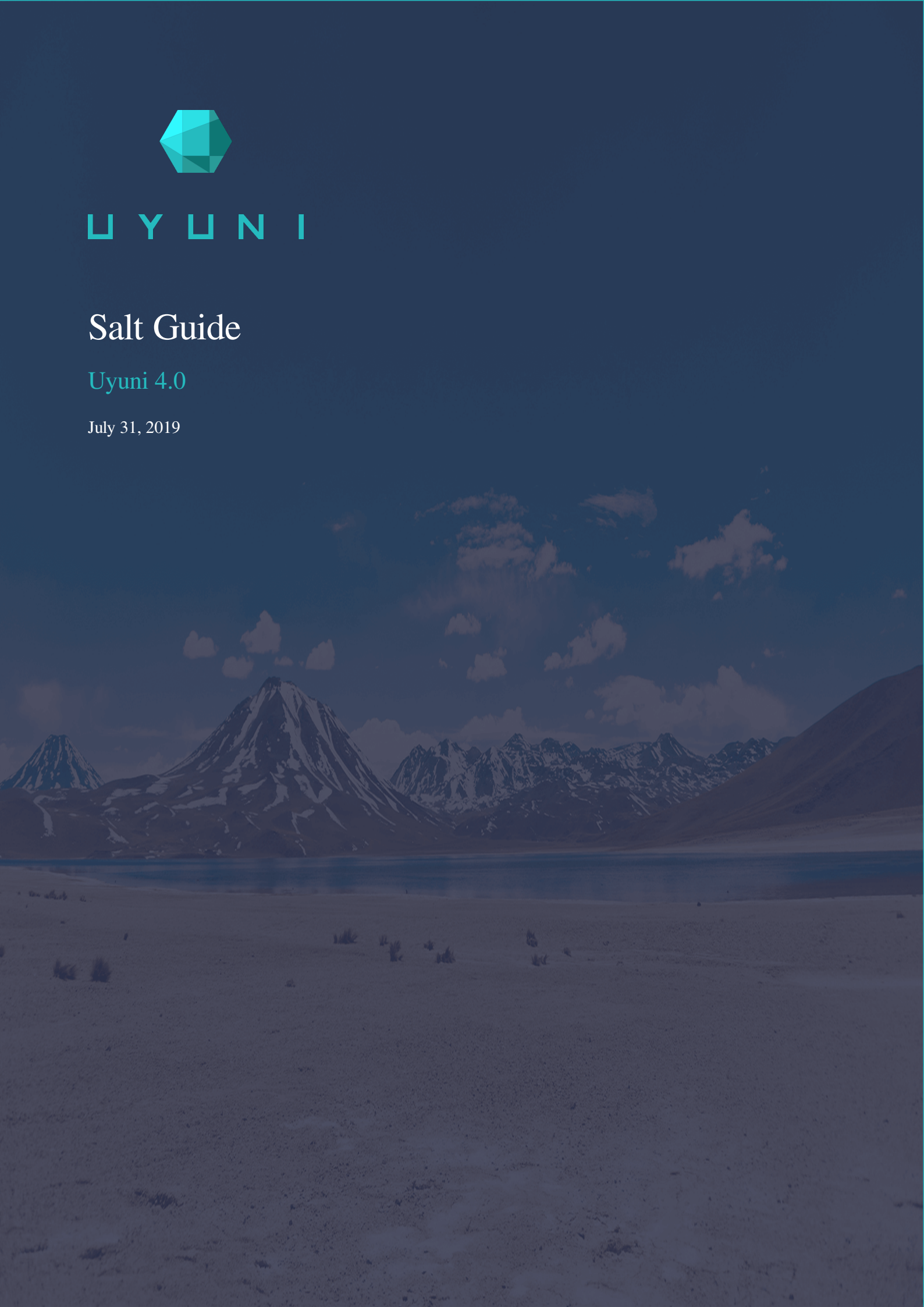


Salt Guide

Uyuni 4.0

July 31, 2019

Table of Contents

GNU Free Documentation License

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections

then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document’s license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version’s license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

```
Copyright (c) YEAR YOUR NAME.
   Permission is granted to copy, distribute and/or modify this document
   under the terms of the GNU Free Documentation License, Version 1.2
   or any later version published by the Free Software Foundation;
   with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
   A copy of the license is included in the section entitled{ldquo}GNU
   Free Documentation License{rdquo}.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “ with…Texts.” line with this:

```
with the Invariant Sections being LIST THEIR TITLES, with the
   Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Introduction

Salt is a configuration management system used by Uyuni to manage clients.

In Uyuni, the Salt master runs on the Uyuni Server, allowing you to register and manage Salt clients.

This book is designed to be a primer for using Salt with Uyuni.

For more information about Salt, see the Salt documentation at https://docs.saltstack.com/en/latest/contents.html.

The current version of Salt in Uyuni is 2019.2.0.

Terminology

Grains

Grains provide information about the hardware of a client. For example, the operating system, IP addresses, network interfaces, memory, etc. When running a Salt command from keep in mind any modules and functions called are run locally from the system being called. Salt modules are stored on clients and master within the following directory:

```
/usr/lib/python2.7/site-packages/salt/
```

List all available grains with the `grains.ls` function:

```
salt '*' grains.ls
```

List collected grain system data by using the `grains.ls` function:

```
salt '*' grains.items
```

For more information on grains, see https://docs.saltstack.com/en/latest/topics/grains/.

States

States are templates which place systems into a known configuration, for example which applications and services are installed and running on those systems. States are a way for you to describe what each of your systems should look like. Once written, states are applied to target systems automating the process of managing and maintaining a large numbers of systems into a known state. For more information on states, see https://docs.saltstack.com/en/latest/topics/tutorials/starting_states.html.

Updating Salt

Do not update salt itself using Salt states. First update all other system packages using Salt states then update salt as a separate stand-alone step from the Uyuni Web UI.

Pillar

Pillars unlike grains are created on the master. Pillar files contain information about a client or group of clients. Pillars allow you to send confidential information to a targeted client or group of clients. Pillars are useful for sensitive data, configuration of clients, variables, and any arbitrary data which should be defined. For more information on pillars, see https://docs.saltstack.com/en/latest/topics/tutorials/pillar.html.

Beacons

Beacons allow an administrator to use the event system in Salt to monitor non-Salt processes. Clients may use beacons to hook into many types of system proceses for constant monitoring. Once a targeted monitored activity occurs an event is sent on the Salt event bus that may be used to trigger a

reactor.

Enabling Beacons

To work with beacons on Salt clients the package python-pyinotify must be installed for SUSE systems. For RES systems install python-inotify. This package is not installed automatically during the salt minion package installation.

Peer Communication with salt-broker

The salt-broker acts like a switch and not like a hub, therefore Peer communication will only work for clients behind the same broker or proxy. For more information on Salt and peer communication, see https://docs.saltstack.com/en/latest/ref/peer.html.

Salt Environments

Uyuni implements Salt with a single environment. Multiple Salt environments are not supported.

Salt Calls

Salt Calls

Salt calls are defined by three main properties:

```
salt 'target' <function> [arguments]
```

Target

Use the second property in a Salt call to target a single machine or group of machines. Specify the client or group of clients you would like to run a function on.

General Targeting

List available grains on all clients:

```
salt '*' grains.ls
```

Ping a specific client:

```
salt 'web1.example.com' test.ping
```

Glob Targeting

Ping all clients using a domain:

```
salt '*example.com' test.ping
```

Display the OS name of all clients with the `webserver` label:

```
salt 'webserver*' grains.item oscodename
```

List Targeting

```
salt -L 'webserver.example.com,db.example.com' test.ping
```

Regular Expression Targeting

You may use PCRE-compliant regular expressions:

```
salt -E '(?!web)' test.ping
```

IP Address Targeting

List client IP addresses:

```
salt '*' network.ip_addrs
```

Ping a specific client IP address:

```
salt -S '172.31.60.74' test.ping
```

Ping all clients on a subnet:

```
salt -S 172.31.0.0/16 test.ping
```

Lookup a Subnet Using the `ip` Command

You can use the `ip` command to find the subnet mask in the format of `192.168.1.1/24`:

```
ip -o -f inet addr show | awk '/scope global/ {print $4}'
```

Function

Once you have specified a target, provide the function you would like to call. Functions also accept arguments. Arguments are space-delimited, for example:

```
salt '*' cmd.run 'echo "Hello: $FIRST_NAME"' env='{FIRST_NAME: "John"}'
```

Locating Additional Minion Functions

Find more functions which can be called on clients by running:

```
salt '*' sys.doc
```

For a full list of callable functions, see https://docs.saltstack.com/en/latest/ref/modules/all/index.html

Arguments

Provides the extra data needed by a function you are calling. The command `pkg.install` requires an argument specifying a package to install. YaST has been selected for installation, for example:

```
salt '*' pkg.install yast2
```

Salt Commands

This section shows useful Salt commands.

salt-run

Print a list of all clients that are up:

```
salt-run manage.up
```

Print a list of all clients that are down:

```
salt-run manage.down
```

Print a list with the current status of all Salt clients:

```
salt-run manage.status
```

Check the version of Salt running on the master and active clients:

```
salt-run manage.versions
```

salt-cp

Copy a file to a client or set of clients.

```
salt-cp '*' foo.conf /root
```

For more information, see https://docs.saltstack.com/en/latest/ref/cli/salt-cp.html.

salt-key -l

List public keys:

```
salt-key -l
```

salt-key -A

Accept all pending keys:

```
salt-key -A
```

Salt States

Salt is capable of applying states by matching clients with relevant state data. This data comes from Uyuni in the form of package and custom states.

State data comes from Uyuni in the form of package and custom states and targets clients at three specific levels of hierarchy. The state hierarchy is defined by the following order or priority: individual clients have priority on packages and custom states over groups; next a group has priority over the organization.

- Client Level

  Systems › Specific Minion › States

- Group Level

  Systems › System Groups

- Organization Level

  Systems › Manage System Types: › My Organization

For example:

- Org1 requires that vim version 1 is installed
- Group1 requires that vim version 2 is installed
- Group2 requires any version installed

This would lead to the following order of hierarchy:

- Client1 part of [Org1, Group1] wants vim removed, vim is removed (Client Level)
- Client2 part of [Org1, Group1] wants vim version 2 gets version 2 (Group Level)
- Client3 part of [Org1, Group1] wants any version, gets version 2 (Org Level)
- Client4 part of[Org1, Group2] wants any version, gets vim version 1 (Org Level)

Salt States Storage Locations

The Uyuni salt-master reads its state data from three file root locations.

The directory `/usr/share/susemanager/salt` It is shipped and updated together with Uyuni and includes certificate setup and common state logic to be applied to packages and channels.

The directory `/srv/susemanager/salt` is generated by Uyuni and based on assigned channels and packages for clients, groups and organizations. This file will be overwritten and regenerated. This could be thought of as the Uyuni database translated into salt directives.

The third directory `/srv/salt` is for custom state data, modules, etc. Uyuni does not operate within or utilize this directory. However, the state data placed here affects the Highstate of clients and is merged with the total state result generated by Uyuni.

Uyuni States

All user created SLS files will be saved to disk on the salt-master server. These files will be placed in `/srv/susemanager/salt/` and each organization will be placed within its own directory. Although these states are custom, these states are created using Uyuni . The following provides an overview of the directory structure:

```
├──── manager_org_DEVEL
│    ├──── files
│    │    ... files needed by states (uploaded by users)...
│    └──── state.sls
         ... other sls files (created by users)...
E.g.:
├──── manager_org_TESTING
│    ├──── files
│    │    └──── motd     # user created
│    │    ... other files needed by states ...
│    └──── motd.sls     # user created
           ... other sls files ...
```

Pillar Data

SUSE Manager exposes a small amount of internal data as Pillars which can be used with custom states. Data that is exposed includes group membership, organization membership, and file roots. These are managed either automatically by Uyuni, or manually by the user.

To avoid hard-coding organization IDs within SUSE Linux Enterprise Server files, a pillar entry is added for each organization:

```
org-files-dir: relative_path_to_files
```

The specified file is available for all clients which belong to the organization.

This is an example of a Pillar located at `/etc/motd`:

```
file.managed:
    - source: salt://{{ pillar['org-files-dir']}}/motd
    - user: root
    - group: root
    - mode: 644
```

Group States

Pillar data can be used to perform bulk actions, like applying all assigned states to clients within the

group. This section contains some example of bulk actions that you can take using group states.

In order to perform these actions, you will need to determine the ID of the group that you want to manipulate. You can determine the Group ID by using the `spacecmd` command:

```
spacecmd group_details
```

In these examples we will use an example Group ID of `GID`.

To apply all states assigned to the group:

```
salt -I 'group_ids:GID' state.apply custom.group_GID
```

To apply any state (whether or not it is assigned to the group):

```
salt -I 'group_ids:GID' state.apply ``state``
```

To apply a custom state:

```
salt -I 'group_ids:2130' state.apply manager_org_1.``customstate``
```

Apply the highstate to all clients in the group:

```
salt -I 'group_ids:GID' state.apply
```

Use Pillars to Set the Package Download Endpoint

By default, Uyuni assumes that the download endpoint to use is the FQDN of the Uyuni server, or the Uyuni Proxy. However, there are some cases where you might like to use a different FQDN as the download endpoint. The most common example is if you need to use load balancing, caching proxies, or in environments with complicated networking requirements.

To change the package download endpoint, you can manually adjust three Salt pillars: * `pkg_download_point_protocol`, defaults to `https`. * `pkg_download_point_host`, defaults to the FQDN of the Uyuni Server (or Proxy, if in use). * `pkg_download_point_port`, defaults to `443`.

If you do not adjust these pillars directly, Uyuni will fall back to the default values.

Procedure: Changing the package download endpoint pillar

1. Navigate to `/srv/pillar/` and create a file called `top.sls` with these contents:

```
base:
  '*':
    - rpm_download_points
```

This example directs Salt to look at the `rpm_download_points.sls` file to determine the base URL to use. You can adjust this file to target different clients or groups, depending on your environment.

2. Remain in `/srv/pillar/` and create a file called `rpm_download_points.sls` with the base URLs you want to use. For example:

```
rpm_download_point_protocol: http
rpm_download_point_host: example.com
rpm_download_point_port: 444
```

3. OPTIONAL: If you want to use external pillars, for example Group IDs, open the master configuration file and set the `ext_pillar_first` parameter to `true`. You can then Group IDs to set conditional values, for example:

```
{% if pillar['group_ids'] is defined and 8 in pillar['group_ids'] %}
  rpm_download_point_protocol: http
  rpm_download_point_host: example.com
  rpm_download_point_port: 444
{%else%}
  rpm_download_point_protocol: ftp
  rpm_download_point_host: example.com
  rpm_download_point_port: 445
{%- endif %}
```

4. OPTIONAL: You can also use grains to set conditional values, for example:

```
{% if grains['fqdn'] == 'client1.example.com' %}
    rpm_download_point: example1.com
{% elif grains['fqdn'] == 'client2.example.com'' %}
    rpm_download_point: example2.com
{%else%}
    rpm_download_point: example.com
{% endif %}
```

Salt File Locations and Structure

The following screen describes Salt file structures and their locations used by the Uyuni Server. These files are listed in `/etc/salt/master.d/susemanager.conf`:

```
# Configure different file roots

file_roots:
  base:
    - /usr/share/susemanager/salt    #Should not be touched by a user
    - /srv/susemanager/salt          #Should not be touched by a user
    - /srv/salt                      #Your custom states go here

# Configure different pillar roots

pillar_roots:
  base:
    - /usr/share/susemanager/pillar  #Should not be touched by a user
    - /srv/pillar                    #Custom pillars go here

# Extension modules path

extension_modules: /usr/share/susemanager/modules

# Master top configuration

master_tops:
  mgr_master_tops: True
```

The following tips should be kept in mind when working with `/etc/salt/master.d/susemanager.conf`.

- Files listed are searched in the order they appear.
- The first file found is called.

file_roots

Uyuni as the Salt master reads its state data from three specific file root directories.

/usr/share/susemanager/salt

This directory is created by Uyuni and its content generated by the `/usr/share/susemanager/modules/tops/mgr_master_tops.py` python module. It is shipped and updated together with Uyuni and includes certificate setup and common state logic that will be applied to packages and channels.

Do Not Edit

You should not edit or add custom Salt data to this directory.

/srv/susemanager/salt

This directory is created by Uyuni and contains assigned channels and packages for clients, groups, and organizations. These files will be overwritten and regenerated. A good analogy for this directory

would be the SUSE Manager database translated into Salt directives.

Do Not Edit

You should not edit or add custom Salt data to this directory.

/srv/salt

The directory `/srv/salt` is for your custom state data, salt modules etc. SUSE Manager does not perform any actions on this directory. However the state data placed here affects the Highstate of clients and is merged with the result generated by Uyuni.

Editable

Place custom Salt data here.

pillar_roots

Uyuni as the Salt master reads its pillar data from two specific pillar root directories.

/usr/share/susemanager/pillar

This directory is generated by Uyuni. It is shipped and updated together with Uyuni.

Do Not Edit

You should not edit or add custom Salt data to this directory.

/srv/pillar

Uyuni by default does not touch or do anything with this directory. However the custom pillar data placed here is merged with the pillar result created by Uyuni.

Editable Directory

Place your custom Salt pillar data here.

Configuration Management

Salt is capable of applying states by matching clients with relevant state data. This data comes from Uyuni in the form of package and custom states.

State Data: Levels of Hierarchy

State data comes from Uyuni in the form of package and custom states and targets clients at three specific levels of hierarchy. The state hierarchy is defined by the following order or priority: individual clients have priority on packages and custom states over groups; next a group has priority over the organization.

- Client Level

  Systems › Specific Minion › States

- Group Level

  Systems › System Groups

- Organization Level

  Systems › Manage System Types: › My Organization

For example:

- Org1 requires that vim version 1 is installed
- Group1 requires that vim version 2 is installed
- Group2 requires any version installed

This would lead to the following order of hierarchy:

- Client1 part of [Org1, Group1] wants vim removed, vim is removed (Client Level)
- Client2 part of [Org1, Group1] wants vim version 2 gets version 2 (Group Level)
- Client3 part of [Org1, Group1] wants any version, gets version 2 (Org Level)
- Client4 part of[Org1, Group2] wants any version, gets vim version 1 (Org Level)

Salt States Storage Locations

The Uyuni salt-master reads its state data from three file root locations.

The directory `/usr/share/susemanager/salt` is used by Uyuni and comes from the susemanager-sls. It is shipped and updated together with Uyuni and includes certificate setup and common state logic to be applied to packages and channels.

The directory `/srv/susemanager/salt` is generated by Uyuni and based on assigned channels and packages for clients, groups and organizations. This file will be overwritten and regenerated. This could be thought of as the Uyuni database translated into salt directives.

The third directory `/srv/salt` is for custom state data, modules etc. Uyuni does not operate within or utilize this directory. However the state data placed here affects the Highstate of clients and is merged with the total state result generated by Uyuni.

Uyuni States

All sls files created by users will be saved to disk on the salt-master server. These files will be placed in `/srv/susemanager/salt/` and each organization will be placed within its own directory. Although these states are custom, these states are created using Uyuni . The following provides an overview of directory structure:

```
├──── manager_org_DEVEL
│    ├──── files
│    │    ... files needed by states (uploaded by users)...
│    └──── state.sls
         ... other sls files (created by users)...
E.g.:
├──── manager_org_TESTING
│    ├──── files
│    │    └──── motd     # user created
│    │    ... other files needed by states ...
│    └──── motd.sls     # user created
           ... other sls files ...
```

Pillar Data

SUSE Manager exposes a small amount of internal data as Pillars which can be used with custom SUSE Linux Enterprise Server states. Data that is exposed includes group membership, organization membership, and file roots. These are managed either automatically by Uyuni, or manually by the user.

To avoid hard-coding organization IDs within SUSE Linux Enterprise Server files, a pillar entry is added for each organization:

```
org-files-dir: relative_path_to_files
```

The specified file is available for all clients which belong to the organization.

This is an example of a Pillar located at `/etc/motd`:

```
file.managed:
    - source: salt://{{ pillar['org-files-dir']}}/motd
    - user: root
    - group: root
    - mode: 644
```

Group States

Pillar data can be used to perform bulk actions, like applying all assigned states to clients within the group. This section contains some example of bulk actions that you can take using group states.

In order to perform these actions, you will need to determine the ID of the group that you want to manipulate. You can determine the Group ID by using the `spacecmd` command:

```
spacecmd group_details
```

In these examples we will use an example Group ID of `GID`.

To apply all states assigned to the group:

```
salt -I 'group_ids:GID' state.apply custom.group_GID
```

To apply any state (whether or not it is assigned to the group):

```
salt -I 'group_ids:GID' state.apply ``state``
```

To apply a custom state:

```
salt -I 'group_ids:2130' state.apply manager_org_1.``customstate``
```

Apply the highstate to all clients in the group:

```
salt -I 'group_ids:GID' state.apply
```

Salt Formulas

This chapter provides an introduction for using Salt Formulas with Uyuni. Creation of custom formulas will also be introduced.

What are Salt Formulas?

Formulas are collections of Salt States that have been pre-written by other Salt users and contain generic parameter fields. Formulas allow for reliable reproduction of a specific configuration again and again. Formulas can be installed from RPM packages or an external git repository.

This list will help you decide whether to use a state or a formula:

Formula Tips

- When writing states for trivial tasks, formulas are probably not worth the time investment.
- For large, non-trivial configurations use formulas.
- Formulas and States both act as a kind of configuration documentation. Once written and stored you will have a snapshot of what your infrastructure should look like.
- Pre-written formulas are available from the Saltstack formula repository on Github. Use these as a starting point for your own custom formulas.
- Formula data can be managed via the XMLRPC API.

Formula with Forms Improvements

Forms are a graphical representation of the formulas parameter data. You can customize these configuration data in the Uyuni Web UI, with entry fields, drop-down, check boxes, etc.

For more information, see https://www.suse.com/c/forms-formula-success/.

Installing Salt Formulas via RPM

SUSE releases formulas as RPM packages. Available formulas can be located within the `SUSE-Manager-Server-3.2-Pool` channel.

Salt State Name Clashes

If a Salt Formula uses the same name as an existing Salt State, the two names will collide, and could result in the formula being used instead of the state. Always check states and formulas to avoid name clashes.

Procedure: Installing Salt Formulas from an RPM

1. To search for available formulas, execute the following command on your Uyuni server:

```
zypper se --type package formula
```

You will see a list of available Salt formulas:

```
S | Name              | Summary                                                        |
Type
--+-------------------+----------------------------------------------------------------+
----------
  | locale-formula    | Locale Salt Formula for SUSE Manager                           |
package
```

2. For more information about a formula, run the following command:

```
zypper info locale-formula
```

```
Information for package locale-formula:
---------------------------------------
Repository: SUSE-Manager-Server-{productnumber}-Pool
Name:  locale-formula
Version:  0.2-1.1
Arch: noarch
Vendor:  SUSE LLC <https://www.suse.com/>
Support Level: Level 3
Status: not installed
Installed Size: 47.9 KiB
Installed: No
Source package : locale-formula-0.2-1.1.src
Summary        : Locale Salt Formula for SUSE Manager
Description    :
    Salt Formula for SUSE Manager. Sets up the locale.
```

3. To install a formula run as root:

```
zypper in locale-formula
```

File Structure Overview

RPM-based formulas must be placed in a specific directory structure to ensure proper functionality. A formula always consists of two separate directories: The `states` directory and the `metadata` directory. Folders in these directories need to have an exactly matching name, for example `locale`.

The Formula State Directory

The formula states directory contains anything necessary for a Salt state to work independently. This includes `.sls` files, a `map.jinja` file and any other required files. This directory should only be modified by RPMs and should not be edited manually. For example, the locale-formula states directory is located in:

```
/usr/share/salt-formulas/states/locale/
```

The Formula Metadata Directory

The metadata directory contains a `form.yml` file which defines the forms for Uyuni and an optional `metadata.yml` file that can contain additional information about a formula. For example, the locale-formula metadata directory is located in:

```
/usr/share/susemanager/formulas/metadata/locale/
```

Custom Formulas

Custom formula data or (non-RPM) formulas need to be placed into any state directory configured as a Salt file root:

State directory

Custom state formula data needs to be placed in:

```
/srv/salt/<custom-formula-name>/
```

Metadata Directory

Custom metadata (information) needs to be placed in:

```
/srv/formula_metadata/<custom-formula-name>/
```

All custom folders located in the following directories need to contain a `form.yml` file. These files are detected as form recipes and may be applied to groups and systems from the Web UI:

```
/srv/formula_metadata/<custom-formula-name>/form.yml
```

The Salt formula directory changed in Uyuni 4.0. The old directory location, `/usr/share/susemanager/formulas`, will continue to work for some time. You should ensure that you update to the new directory location, `/usr/share/salt-formulas/` as soon as possible.

Editing Pillar Data in Uyuni

Uyuni requires a file called `form.yml`, to describe how formula data should look within the Web UI. `form.yml` is used by Uyuni to generate the desired form, with values editable by a user.

For example, the `form.yml` that is included with the locale-formula is placed in:

```
/usr/share/susemanager/formulas/metadata/locale/form.yml
```

See part of the following locale-formula example:

```
# This file is part of locale-formula.
#
# Foobar is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# Foobar is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with Foobar.  If not, see <http://www.gnu.org/licenses/>.

timezone:
  $type: group

  name:
    $type: select
    $values: ["CET",
              "CST6CDT",
              "EET",
              "EST",
              "EST5EDT",
              "GMT",
              "GMT+0",
              "GMT-0",
              "GMT0",
              "Greenwich",
              "HST",
              "MET",
              "MST",
              "MST7MDT",
              "NZ",
              "NZ-CHAT",
              "Navajo",
              "PST8PDT",
              "UCT",
              "UTC",
              "Universal",
              "W-SU",
              "WET",
              "Zulu",
              "Etc/GMT+1",
              "Etc/GMT+2",
              "Etc/GMT+3",
              "Etc/GMT+4",
              "Etc/GMT+5",
              "Etc/GMT+6",
              "Etc/GMT+7",
              "Etc/GMT+8",
              "Etc/GMT+9",
              "Etc/GMT+10",
              "Etc/GMT+11",
              "Etc/GMT+12",
              "Etc/GMT-1",
              "Etc/GMT-2",
              "Etc/GMT-3",
              "Etc/GMT-4",
```

```
              "Etc/GMT-5",
              "Etc/GMT-6",
              "Etc/GMT-7",
              "Etc/GMT-8",
              "Etc/GMT-9",
              "Etc/GMT-10",
              "Etc/GMT-11",
              "Etc/GMT-12",
              "Etc/GMT-13",
              "Etc/GMT-14",
              "Etc/GMT",
              "Etc/GMT+0",
              "Etc/GMT-0",
              "Etc/GMT0",
              "Etc/Greenwich",
              "Etc/UCT",
              "Etc/UTC",
              "Etc/Universal",
              "Etc/Zulu"
              ]
    $default: CET

  hardware_clock_set_to_utc:
    $type: boolean
    $default: True
...
```

`form.yml` contains additional information that describes how the form for a pillar should look for Uyuni. This information is contained in attributes that always start with a `$` sign.

Ignored Values

All values that start with a `$` sign are annotations used to display the UI that users interact with. These annotations are not part of pillar data itself and are handled as metadata.

The following are valid attributes.

$type

The most important attribute is the `$type` attribute. It defines the type of the pillar value and the form-field that is generated. The following represent the supported types:

- `text`
- `password`
- `number`
- `url`
- `email`
- `date`
- `time`
- `datetime`

- `boolean`
- `color`
- `select`
- `group`
- `edit-group`
- `namespace`
- `hidden-group` (obsolete, renamed to `namespace`)

Text Attribute

The text attribute is the default and does not need to be specified explicitly.

Many of these values are self-explanatory:

- The `text` type generates a simple text field
- The `password` type generates a password field
- The `color` type generates a color picker

The `group`, `edit-group`, and `namespace` (formerly `hidden-group`) types do not generate an editable field and are used to structure form and pillar data. All these types support nesting. For providing default values with nesting, see edit-group Example with Nesting. The difference between `group` and `namespace` is `group` generates a visible border with a heading, and `namespace` shows nothing visually (and is only used to structure pillar data). The difference between `group` and `edit-group` is: `edit-group` allows to structure and restrict editable fields in a more flexible way. `edit-group` is a collection of items of the same kind; collections can have the following four "shapes":

- A list of primitive items
- A list of dictionaries
- A dictionary of primitive items
- A dictionary of dictionaries

The size of each collection is variable; users can add or remove elements.

For example, `edit-group` supports the `$minItems` and `$maxItems` attributes, and thus it simplifies complex and repeatable input structures. These, and also `itemName`, are optional. For an `edit-group` example, see Simple edit-group Example.

$default

`$default` allows you to specify a default value that is displayed and used, if no other value is entered. In an `edit-group` it allows to create initial members of the group and populate them with specified data.

$optional

`$optional` is a boolean attribute. If it is `true` and the field is empty in the form, then this field will not be generated in the formula data and the generated dictionary will not contain the field name key. If `$optional` is `false` and the field is empty, the formula data will contain a `<field name>: null` entry.

$ifEmpty

The value to be used if the field is empty (because the user did not input any value). `ifEmpty` can only be used when `$optional` is `false` or not defined at all! If `$optional` is `true`, then `$ifEmpty` is ignored. In the following example, the `DP2` string would be used if user leaves the field empty:

```
displayName:
  $type: string
  $ifEmpty: DP2
```

$name

`$name` allows you to specify the name of a value that is shown in the form. If this value is not set, the pillar name is used and capitalized without underscores and dashes. You reference it in the same section with `${name}`.

$help and $placeholder

The `$help` and `$placeholder` attributes are used to give a user a better understanding of what the value should be.

- `$help` defines the message a user sees when hovering over a field
- `$placeholder` displays a gray placeholder text in the field

`$placeholder` may only be used with text fields like text, password, email or date. It does not make sense to add a placeholder if you also use `$default` as this will hide the placeholder.

$key

`$key` is applicable if the `edit-group` has the "shape" of a dictionary; you use it when the pillar data is supposed to be a dictionary. The `$key` attribute then determines the key of an entry in the dictionary. Example:

```
user_passwords:
  $type: edit-group
  $minItems: 1
  $prototype:
    $key:
        $type: text
    $type: text
  $default:
    alice: secret-password
    bob: you-shall-not-pass
```

Pillar:

```
user_passwords:
  alice:
    secret-password
  bob:
    you-shall-not-pass
```

$minItems and $maxItems

In an `edit-group`, `$minItems` and `$maxItems` allow you to specify the lowest and highest number the group can occur.

$itemName

In an `edit-group`, `$itemName` allows you to define a template for the name to be used for the members of the group.

$prototype

In an `edit-group`, `$prototype` is mandatory and allows to define default (or pre-filled) values for newly added members in the group.

$scope

`$scope` allows you to specify a hierarchy level at which a value may be edited. Possible values are `system`, `group`, and `readonly`.

The default `$scope: system` allows values to be edited at group and system levels. A value can be entered for each system but if no value is entered the system will fall back to the group default.

If using `$scope: group`, a value may only be edited for a group. On the system level you will be able to see the value, but not edit it.

The `$scope: readonly` option makes a field read-only. It can be used to show a user data which should be known, but should not be editable. This option only makes sense in combination with the `$default` attribute.

$visibleIf

`$visibleIf` allows you to show a field or group if a simple condition is met. A condition always looks similar to the following example:

```
some_group#another_group#my_checkbox == true
```

The left part of the above statement is the path to another value, and groups are separated by `$` signs. The middle section of the command should be either `==` for a value to be equal or `!=` for values that should be not equal. The last field in the statement can be any value which a field should have or not have.

The field with this attribute associated with it will now be shown only when the condition is met. In this example the field will be shown only if `my_checkbox` is checked. The ability to use conditional statements is not limited to check boxes. It may also be used to check values of select-fields, text-fields, etc.

A check box should be structured like the following example:

```
some_group:
  $type: group

  another_group:
    $type: group

      my_checkbox:
        $type: boolean
```

Relative paths can be specified using prefix dots. One dot means sibling, 2 dots mean parent, etc. This is mostly useful for `edit-group`.

```
some_group:
  $type: group

  another_group:
    $type: group

    my_checkbox:
      $type: boolean

    my_text:
      $visibleIf: .my_checkbox

  yet_another_group:
    $type: group

    my_text2:
      $visibleIf: ..another_group#my_checkbox
```

By using multiple groups with the attribute, you can allow a user to select an option and show a completely different form, dependent upon the selected value.

Values from hidden fields may be merged into the pillar data and sent to the client. A formula must check the condition again and use the appropriate data. For example:

```
show_option:
  $type: checkbox
some_text:
  $visibleIf: show_option == true
```

```
{% if pillar.show_option %}
do_something:
  with: {{ pillar.some_text }}
{% endif %}
```

$values

`$values` can only be used together with `$type`: select to specify the different options in the select-field. `$values` must be a list of possible values to select. For example:

```
select_something:
  $type: select
  $values: ["option1", "option2"]
```

Or alternatively:

```
select_something:
  $type: select
  $values:
    - option1
    - option2
```

Simple edit-group Example

See the following **edit-group** example:

```
partitions:
  $name: "Hard Disk Partitions"
  $type: "edit-group"
  $minItems: 1
  $maxItems: 4
  $itemName: "Partition ${name}"
  $prototype:
    name:
      $default: "New partition"
    mountpoint:
      $default: "/var"
    size:
      $type: "number"
      $name: "Size in GB"
  $default:
    - name: "Boot"
      mountpoint: "/boot"
    - name: "Root"
      mountpoint: "/"
      size: 5000
```

After clicking **[Add]** for one time you will see `edit-group` Example in the Web UI filled with the default values. The formula itself is called `hd-partitions` and will appear as `Hd Partitions` in the Web UI.

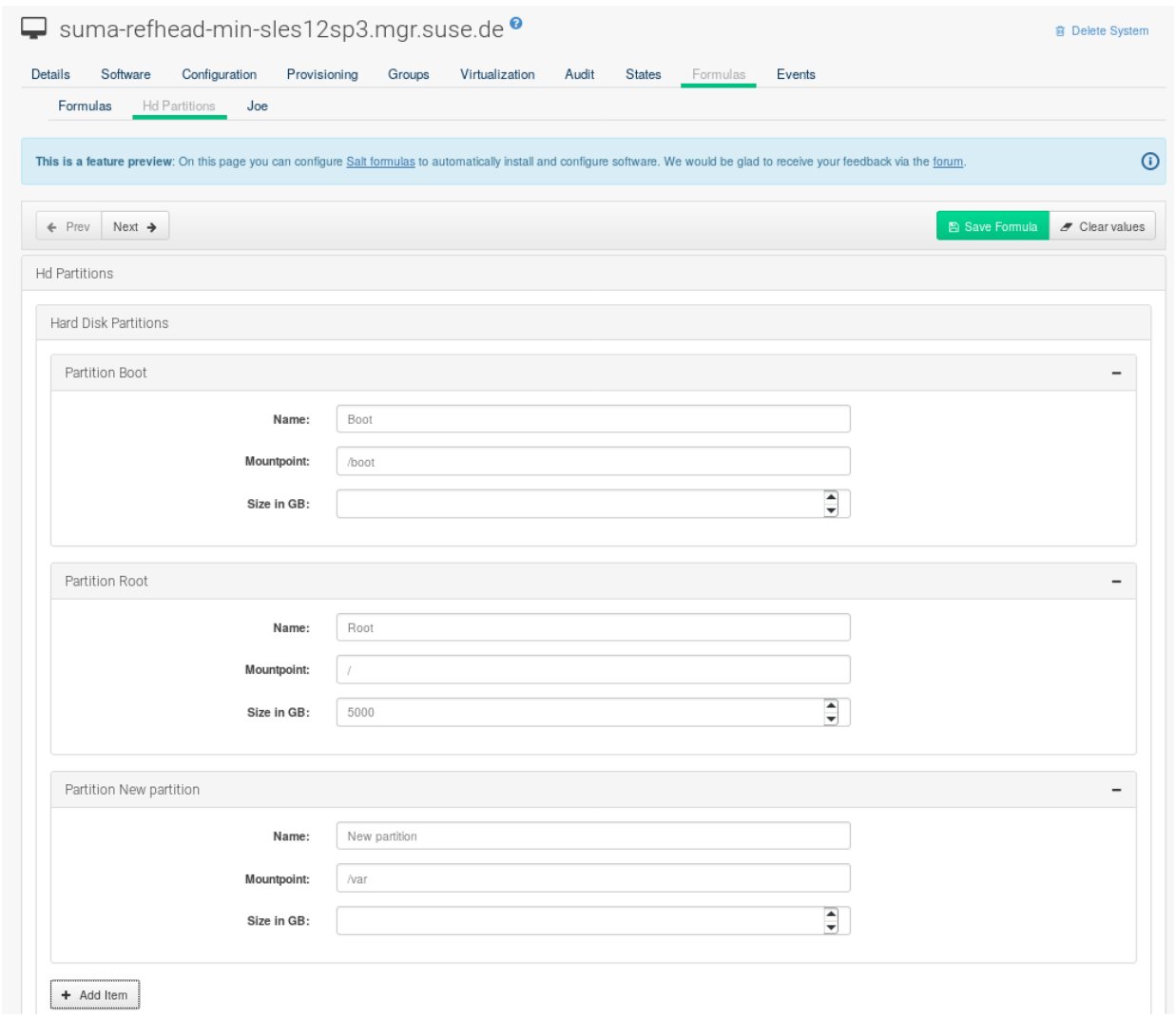


Figure 1. `edit-group` *Example in the Web UI*

To remove the definition of a partition click the minus symbol in the title line of an inner group. When form fields are properly filled confirm with clicking **[Save Formula]** in the upper right corner of the formula.

edit-group Example with Nesting

See the following **edit-group** example:

```
users:
  $name: "Users"
  $type: edit-group
  $minItems: 2
  $maxItems: 5
  $prototype:
    name:
      $default: "username"
    password:
      $type: password
    groups:
      $type: edit-group
      $minItems: 1
      $prototype:
        group_name:
          $type: text
  $default:
    - name: "root"
      groups:
        - group_name: "users"
        - group_name: "admins"
    - name: "admin"
      groups:
        - group_name: "users"
```

Writing Salt Formulas

Salt formulas are pre-written Salt states, which may be configured with pillar data. You can parametrize state files using Jinja. Jinja allows you to access pillar data by using the following syntax. This syntax works best when you are uncertain whether a pillar value exists as it will throw an error:

```
pillar.some.value
```

When you are sure a pillar exists you may also use the following syntax:

```
salt['pillar.get']('some:value', 'default value')
```

You may also replace the `pillar` value with `grains` (for example, `grains.some.value`) allowing access to grains.

Using data this way allows you to make a formula configurable. The following code snippet will install a package specified in the pillar `package_name`:

```
install_a_package:
  pkg.installed:
    - name: {{ pillar.package_name }}
```

You may also use more complex constructs such as `if/else` and `for-loops` to provide greater functionality:

```
{% if pillar.installSomething %}
something:
  pkg.installed
{% else %}
anotherPackage:
  pkg.installed
{% endif %}
```

Another example:

```
{% for service in pillar.services %}
start_{{ service }}:
  service.running:
    - name: {{ service }}
{% endfor %}
```

Jinja also provides other helpful functions. For example, you can iterate over a dictionary:

```
{% for key, value in some_dictionary.items() %}
do_something_with_{{ key }}: {{ value }}
{% endfor %}
```

You may want to have Salt manage your files (for example, configuration files for a program), and you can change these with pillar data. For example, the following snippet shows how you can manage a file using Salt:

```
/etc/my_program/my_program.conf:
  file.managed:
    - source: salt://my_state/files/my_program.conf
    - template: jinja
```

Salt will copy the file `salt-file_roots/my_state/files/my_program.conf` on the salt master to `/etc/my_program/my_program.conf` on the client and template it with Jinja. This allows you to use Jinja in the file, exactly like shown above for states:

```
some_config_option = {{ pillar.config_option_a }}
```

Separating Data

It is often a good idea to separate data from a state to increase its flexibility and add re-usability value. This is often done by writing values into a separate file named `map.jinja`. This file should be placed within the same directory as your state files.

The following example will set `data` to a dictionary with different values, depending on which system the state runs on. It will also merge data with the pillar using the `some.pillar.data` value so you can access `some.pillar.data.value` by just using `data.value`.

You can also choose to override defined values from pillars (for example, by overriding `some.pillar.data.package` in the example).

```
{% set data = salt['grains.filter_by']({
    'Suse': {
        'package': 'packageA',
        'service': 'serviceA'
    },
    'RedHat': {
        'package': 'package_a',
        'service': 'service_a'
    }
}, merge=salt['pillar.get']('some:pillar:data')) %}
```

After creating a map file like the above example, you can maintain compatibility with multiple system types while accessing "deep" pillar data in a simpler way. Now you can import and use `data` in any file. For example:

```
{% from "some_folder/map.jinja" import data with context %}

install_package_a:
  pkg.installed:
    - name: {{ data.package }}
```

You can also define multiple variables by copying the `{% set ···%}` statement with different values and then merge it with other pillars. For example:

```
{% set server = salt['grains.filter_by']({
    'Suse': {
        'package': 'my-server-pkg'
    }
}, merge=salt['pillar.get']('myFormula:server')) %}
{% set client = salt['grains.filter_by']({
    'Suse': {
        'package': 'my-client-pkg'
    }
}, merge=salt['pillar.get']('myFormula:client')) %}
```

To import multiple variables, separate them with a comma. For Example:

```
{% from "map.jinja" import server, client with context %}
```

Formulas utilized with Uyuni should follow formula conventions listed in the official documentation:

- https://docs.saltstack.com/en/latest/topics/development/conventions/formulas.html

Uyuni Generated Pillar Data

When pillar data is generated (for example, after applying the highstate) the following external pillar script generates pillar data for packages, group ids, etc. and includes all pillar data for a system:

```
/usr/share/susemanager/modules/pillar/suma_minion.py
```

The process is executed as follows:

1. The `suma_minion.py` script starts and finds all formulas for a system (by checking the `group_formulas.json` and `server_formulas.json` files).
2. `suma_minion.py` loads the values for each formula (groups and from the system) and merges them with the highstate (default: if no values are found, a group overrides a system if $scope: group etc.).
3. `suma_minion.py` also includes a list of formulas applied to the system in a pillar named formulas. This structure makes it possible to include states. The top file (in this case specifically generated by the `mgr_master_tops.py` script) includes a state called formulas for each system. This includes the `formulas.sls` file located in:

   ```
   /usr/share/susemanager/formulas/states/
   ```

   Or:

   ```
   /usr/share/salt-formulas/states/
   ```

   The content looks similar to the following:

   ```
   include: {{ pillar["formulas"] }}
   ```

   This pillar includes all formulas, that are specified in pillar data generated from the external pillar script.

Formula Requirements

Formulas should be designed/created directly after a Uyuni installation, but if you encounter any issues check the following:

- The external pillar script (`suma_minion.py`) must include formula data.
- Data is saved to `/srv/susemanager/formula_data` and the `pillar` and `group_pillar` sub-directories. These should be automatically generated by the server.
- Formulas must be included for every client listed in the top file. Currently this process is initiated by the `mgr_master_tops.py` script which includes the `formulas.sls` file located in:

  ```
  /usr/share/susemanager/formulas/states/
  ```

  Or:

```
/usr/share/salt-formulas/states/
```

This directory must be a salt file root. File roots are configured on the salt-master (Uyuni) located in:

```
/etc/salt/master.d/susemanager.conf
```

Using Salt Formulas with Uyuni

The following procedure provides an overview on using Salt Formulas with Uyuni.

1. Official formulas may be installed as RPMs. Place the custom states within `/srv/salt/your-formula-name/` and the metadata (`form.yml` and `metadata.yml`) in `/srv/formula_metadata/your-formula-name/`. After installing your formulas they will appear in **Salt › Formula Catalog**.
2. To begin using a formula, apply it to a group or system. Apply a formula to a group or system by selecting the **System Details › Formulas** tab of a `System Details` page or `System Group`. From the **System Details › Formulas** page you can select any formulas you wish to apply to a group or system. Click the **[Save]** button to save your changes to the database.
3. After applying one or more formulas to a group or system, additional tabs will become available from the top menu, one for each formula selected. From these tabs you may configure your formulas.
4. When you have finished customizing your formula values you will need to apply the highstate for them to take effect. Applying the highstate will execute the state associated with the formula and configure targeted systems. You can use the **[Apply Highstate]** button from any formulas page of a group.
5. When a change to any of your values is required or you need to re-apply the formula state because of a failure or bug, change values located on your formula pages and re-apply the highstate. Salt will ensure that only modified values are adjusted and restart or reinstall services only when necessary.

For more information about Salt formulas, see https://docs.saltstack.com/en/latest/topics/development/conventions/formulas.html

For more information about using Salt formulas in a SUSE Manager for Retail environment, see **[Retail › Retail-formulas-intro ›]**.

Locale

The locale formula allows setting `Timezone` and [guimenu]Keyboard and Language`.

Domain Name System (Bind)

With the bind formula you set up and configure a Domain Name System (DNS) server. For technical information about the bind formula and low-level pillar data, see the `README.rst` file on the Uyuni

server: `/usr/share/salt-formulas/metadata/bind/README.rst`.

DNS is needed to resolve the domain names and host names into IP addresses. For more information about DNS, see the SLES Administration Guide, Services, The Domain Name System.

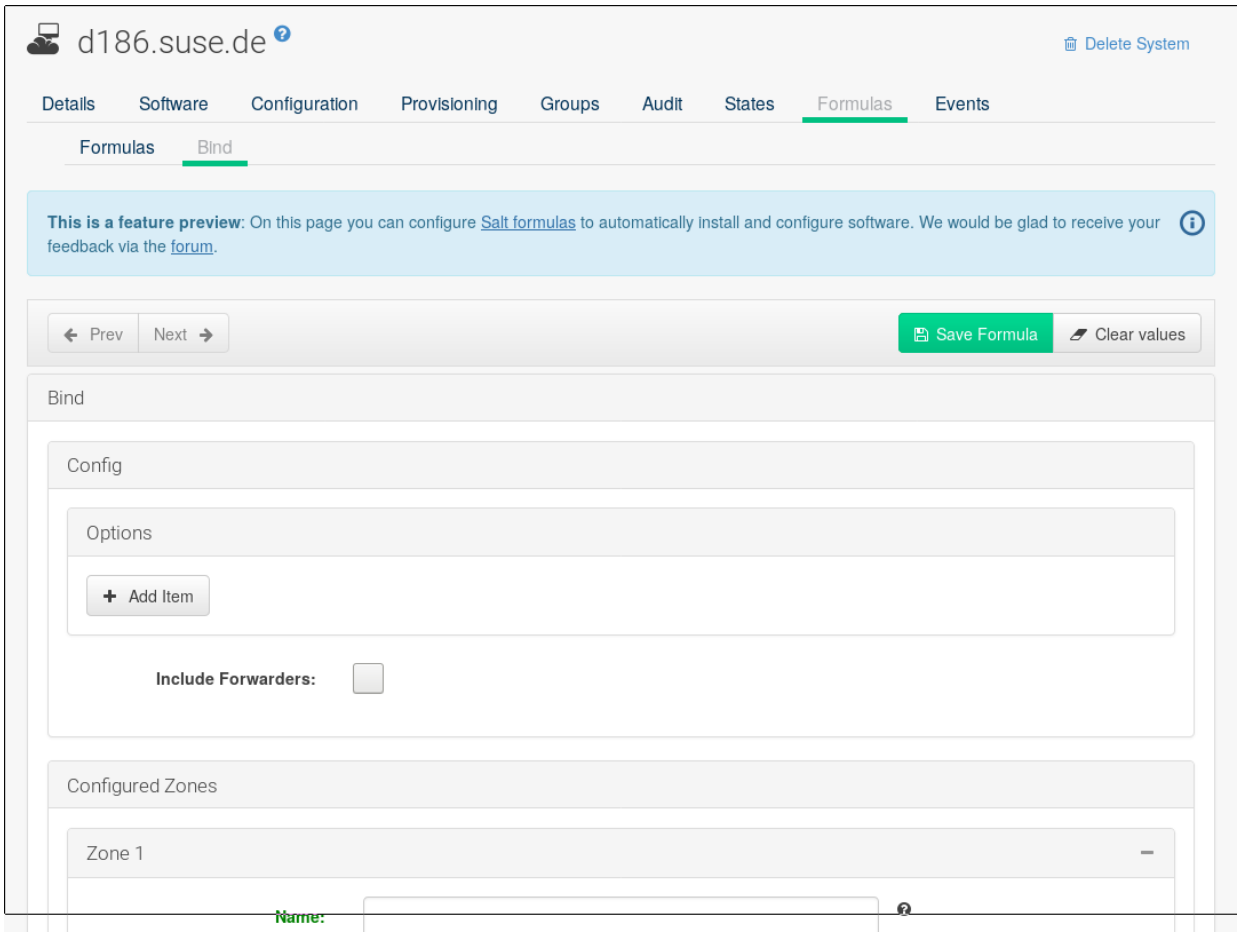


Figure 2. Bind Formula

In the `Config` group you can set arbitrary options such as `directory` where are the zone data files (usually `/var/lib/named/`) or `forwarders`. Click **[Add Item]** to provide more Key/Value fields for configuration.

Check `Include Forwarders` if you want to rely on an external DNS server if your DNS is down (or is otherwise not able to resolve an address).

At least, you will configure one zone. In `Configured Zones` define your zone; for example, `example.com`. Then in `Available Zones` configure this zone: as `Name` enter your zone (in this case `example.com`) and the `File` to which this configuration should be written (`example.com.txt`). Enter the mandatory `SOA` record (start of authority), and the A, NS, and CNAME `Records` you need.

On the other hand, if no `records` entry exists, the zone file is not generated by this state rather than taken from `salt://zones`. For how to overwrite this URL, see `pillar.example`.

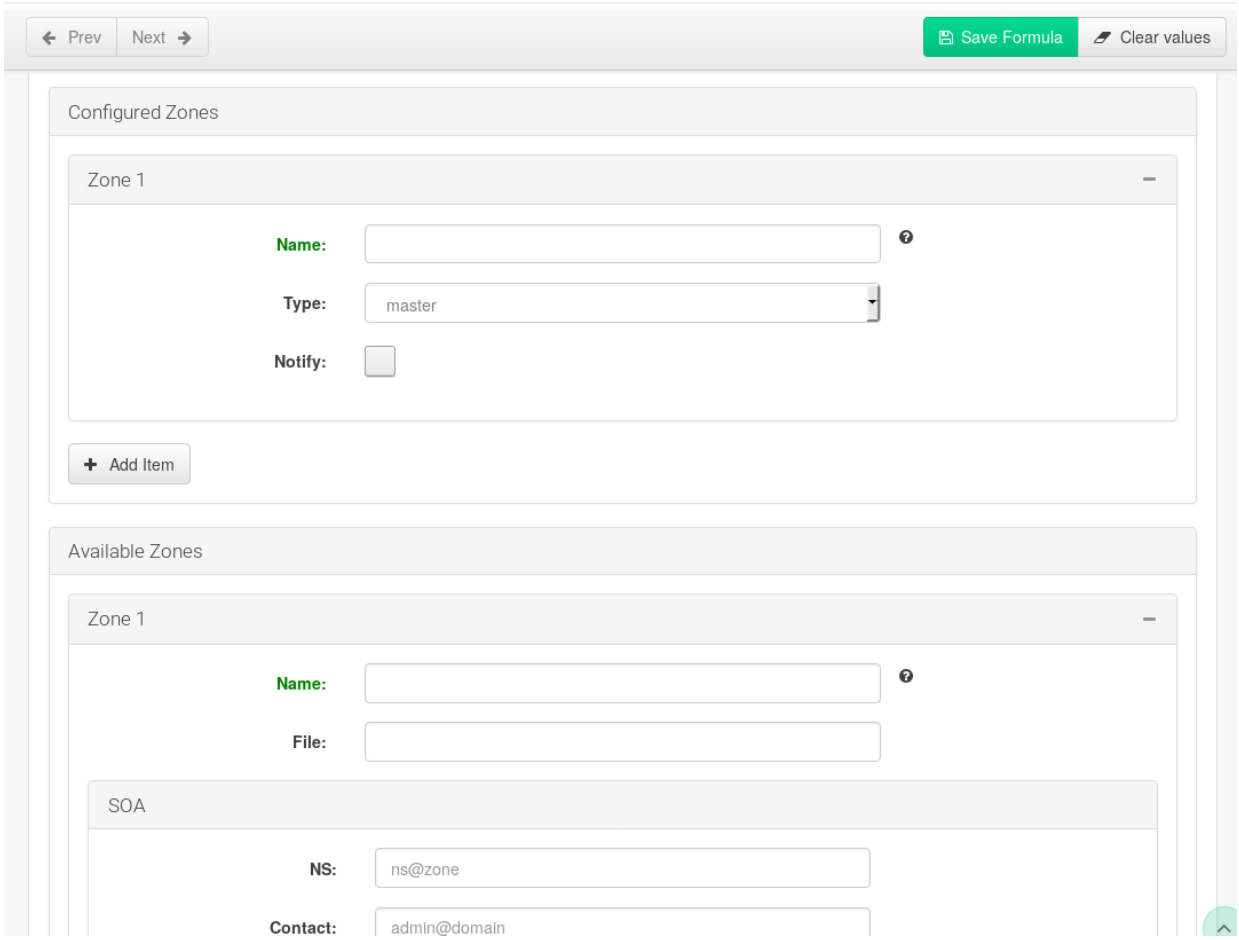


Figure 3. bind-02-zones

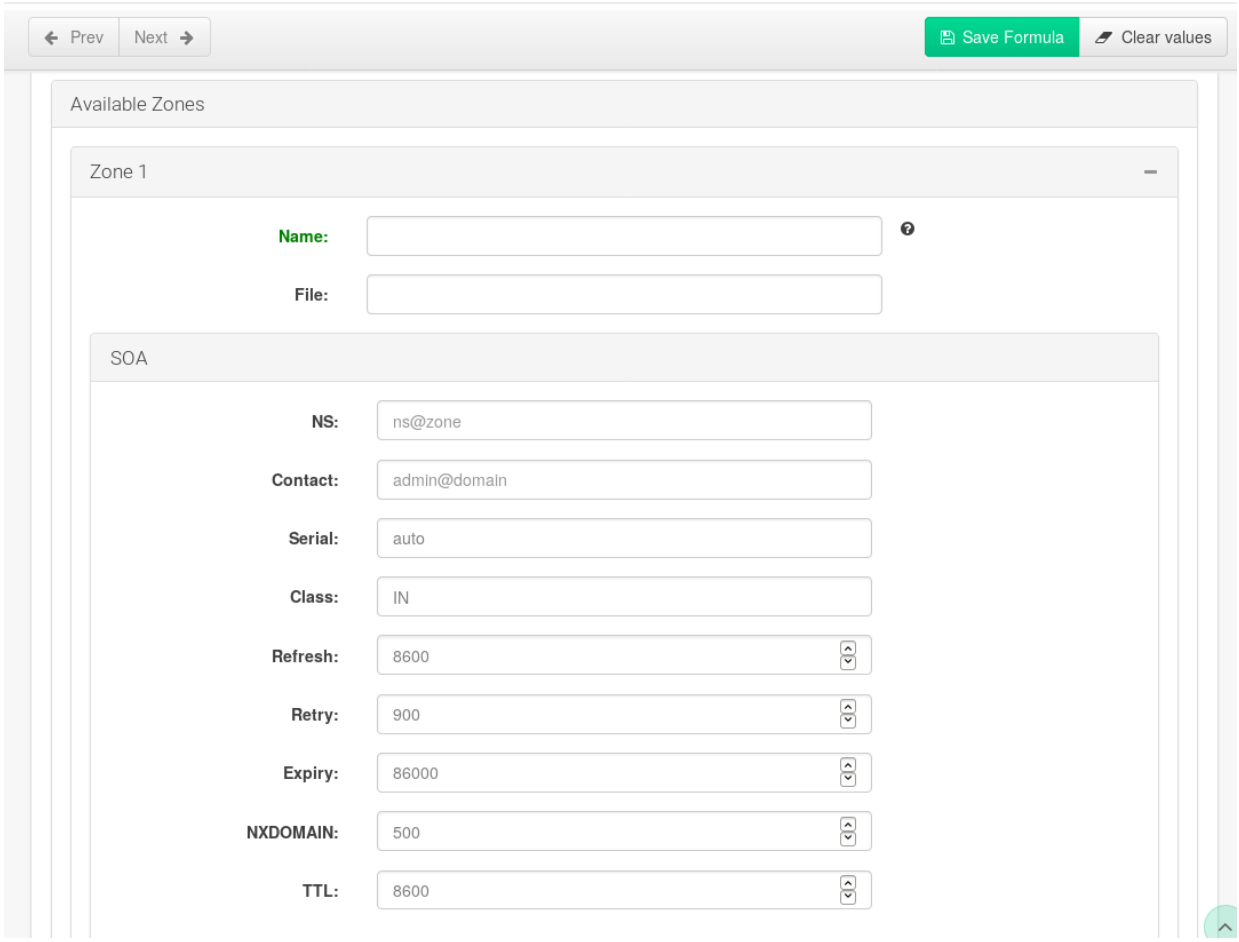


Figure 4. bind-03-records

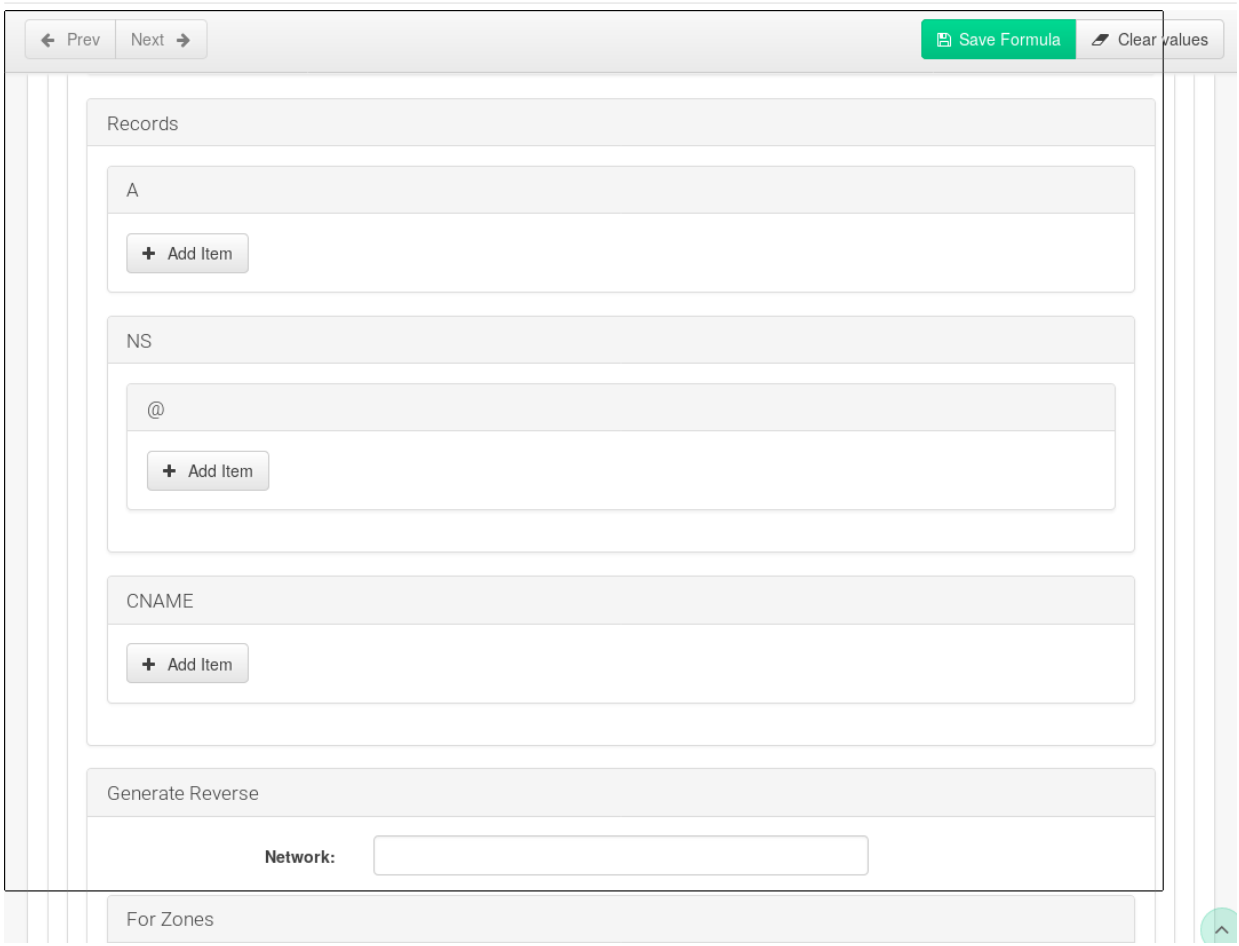


Figure 5. bind-03-records2

In `Generate Reverse`, and define reverse mapping and for which zones:

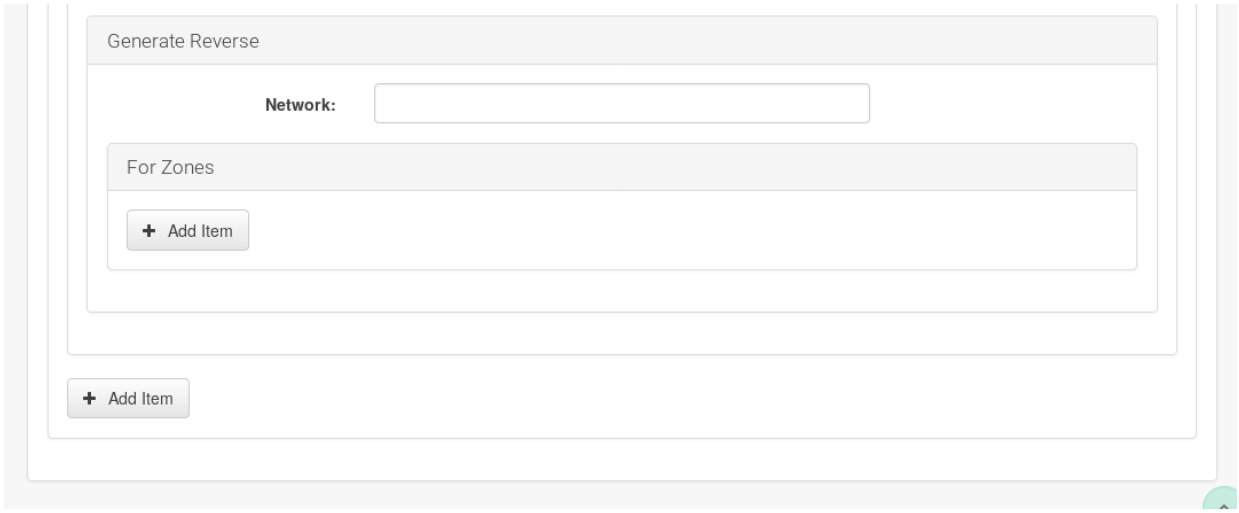


Figure 6. bind-04-reverse

When saved, data is written to `/srv/susemanager/formula_data/pillar/<salt-client.example.com>_bind.json`.

If you apply the highstate (**System Details › States › Highstate**), it first ensures that `bind` and all required packages will get installed. Then it will start the DNS service (`named`).

Dhcpd

With the dhcpd formula you set up and configure a DHCP server (Dynamic Host Configuration Protocol). For technical information about the dhcpd formula and low-level pillar data, see the Pillar example file `/usr/share/susemanager/formulas/metadata/dhcpd/pillar.example`.

DHCP is needed to define network settings centrally (on a server) and let clients retrieve and use this information for local host configuration. For more information about DHCP, see the SLES Administration Guide, Services, DHCP.

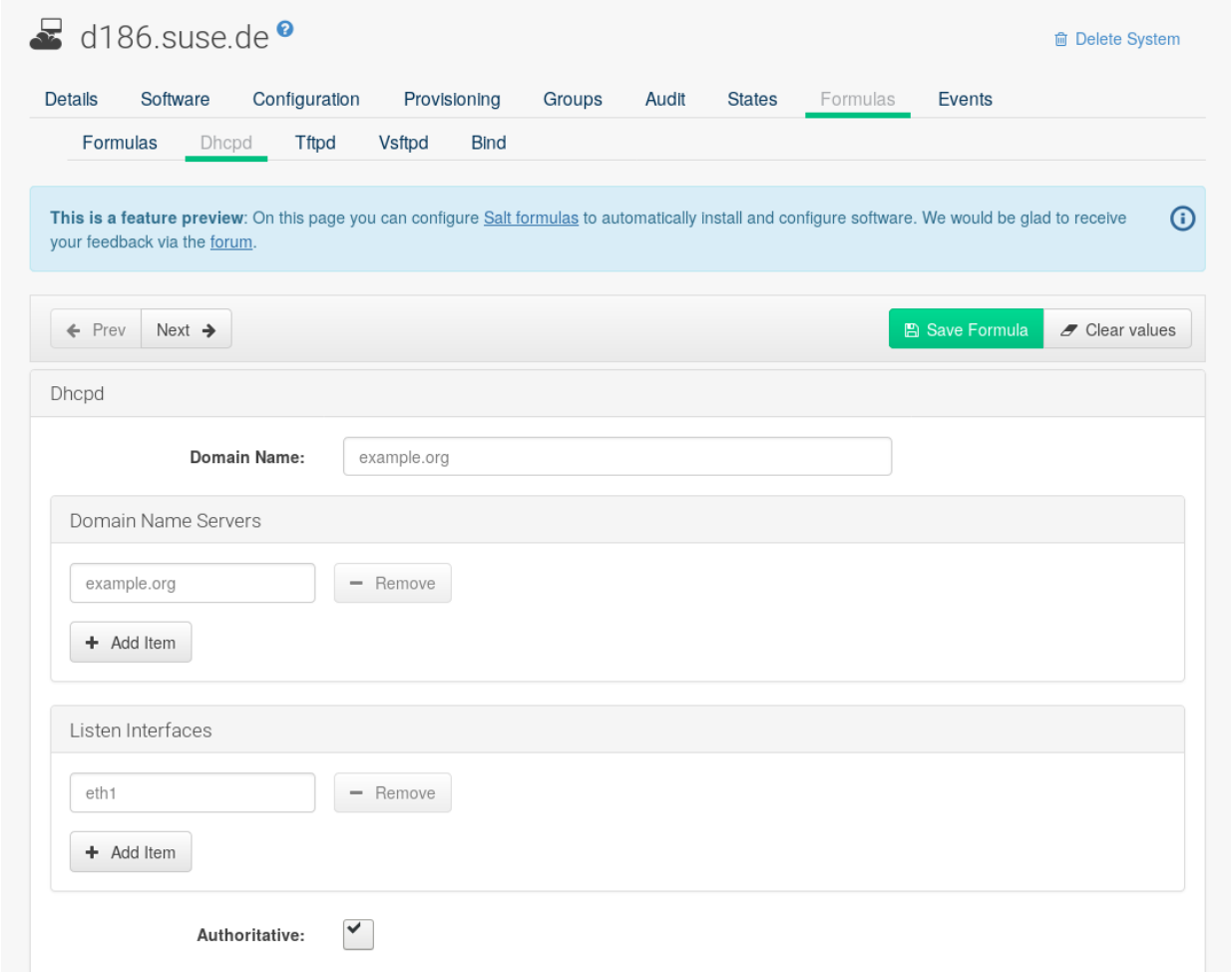


Figure 7. dhcpd formula

Domain Name.

`Domain Name Servers`. One or more Domain Name Service (DNS) servers.

On which interface(s) the DHCP server should listen (`Listen interfaces`). Set option for this interface: Authoritative: Max Lease Time: Default Lease Time:

Next is at least one network in the `Network configuration (subnet)` group (with IP address, netmask, etc.). You define every network with `Dynamic IP range`, `Routers`, and `Hosts with static IP addresses (with defaults from subnet)` (optionally).

And finally `Hosts with static IP addresses (with global defaults)`.

If you apply the highstate (**System Details › States › Highstate**), it first ensures that `dhcp-server` and all required packages will get installed. Then it will start the DHCP service (`dhcpd`).

Tftpd

With the tftpd formula you set up and configure a TFTP server (Trivial File Transfer Protocol). A TFTP server is a component that provides infrastructure for booting with PXE.

For more information about setting up TFTP, see the SLES Deployment Guide, Preparing Network Boot Environment, Setting Up a TFTP Server.

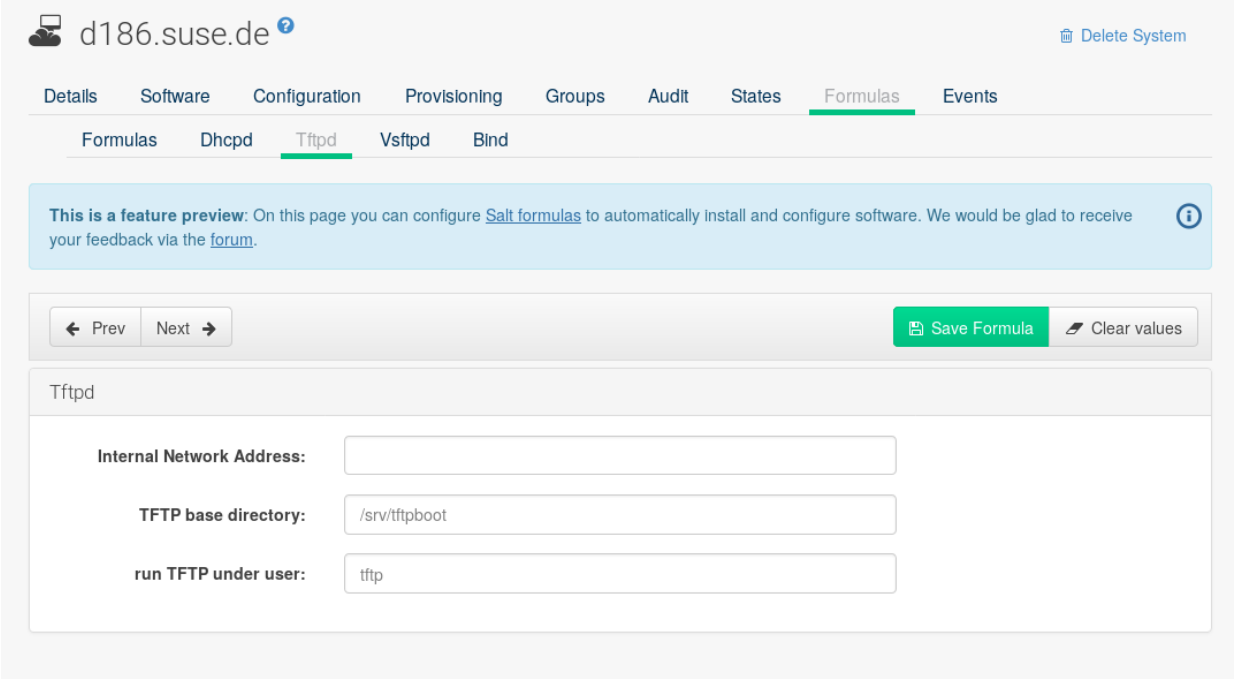


Figure 8. tftpd formula

For setting up a TFTP server, specify the `Internal Network Address`, `TFTP base directory` (default: `/srv/tftpboot`), and `run TFTP under user` (default: `sftp`).

If you apply the highstate (**System Details › States › Highstate**), it first ensures that `atftp` and all required packages will get installed. Then it will start TFTP (`atftpd`).

Vsftpd

With the vsftpd formula you set up and configure Vsftpd. Vsftpd is an FTP server or daemon, written with security in mind. "vs" in its name stands for "Very Secure".

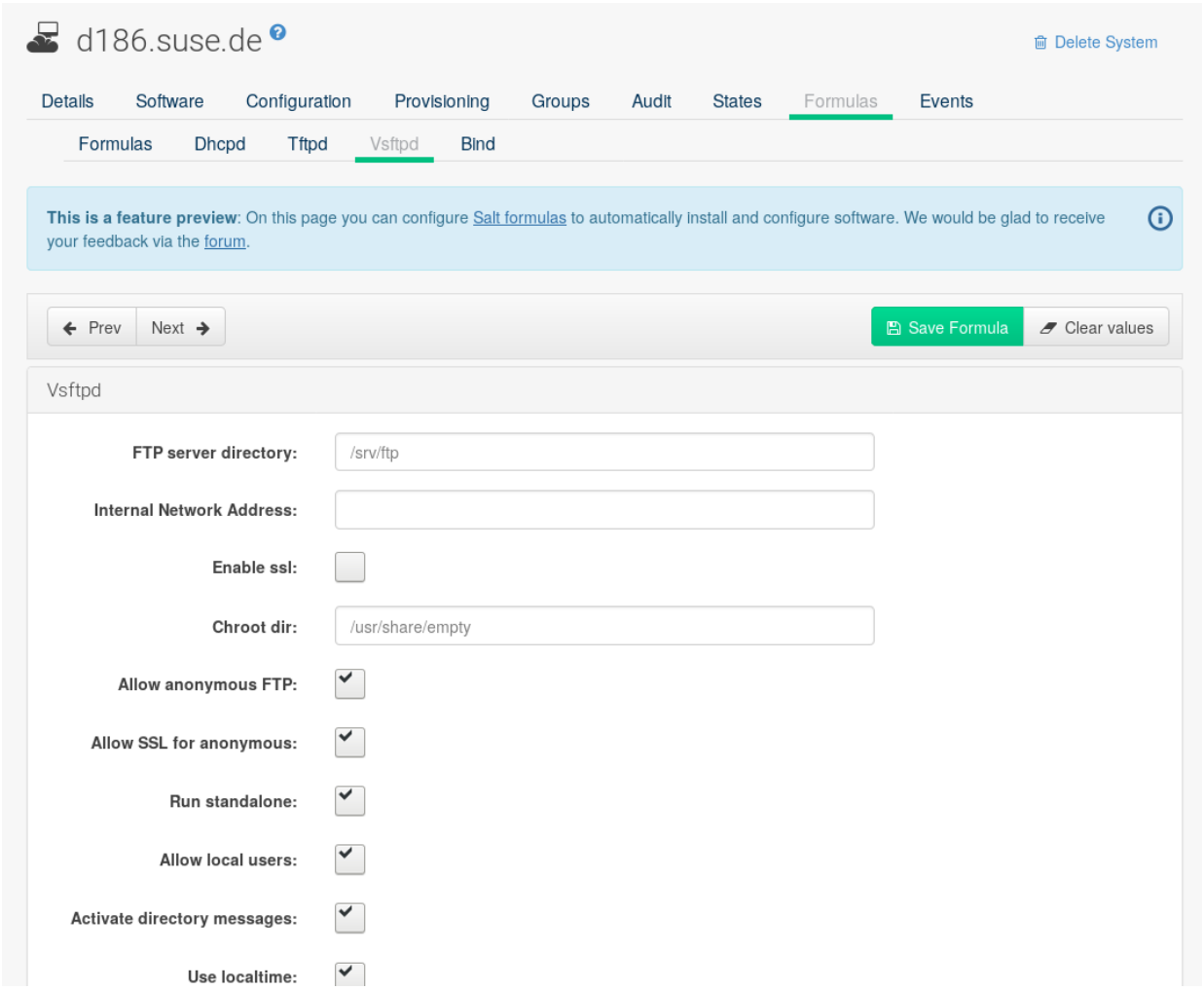


Figure 9. vsftpd formula

For configuring a VSFTP server, specify the settings and options in the Vsftpd formula. There are settings such as `FTP server directory`, `Internal Network Address` `Enable ssl`, etc.

If you apply the highstate (**System Details › States › Highstate**), it first ensures that `vsftpd` and all required packages will get installed. Then it will start the VSFTP service (`vsftpd`).

For more information about setting up and tuning Vsftpd, see the documentation coming with the `vsftpd` package (`/usr/share/doc/packages/vsftpd/` when the package is installed).

Install the Example Formula

This section provides guidance on installing and using SUSE-provided Salt formulas.

Procedure: Installing the Locale Formula

1. Install the locale formula with:

```
zypper install locale-formula
```

This installs the package contents to `/usr/share/susemanager/formulas/{metadata,states}`

2. After installing the RPM, log in to the Uyuni Web UI.
3. Browse to the **Main Menu** › **System Details** page of any client you would like to apply the formula to.
4. On the **Main Menu** › **System Details** page you will see a new **[Formulas]** tab. Select it to view a list of installed formulas.
5. From the **[Formulas]** list select **Formulas** › **Locale** and click **[Save]**.
6. A new tab will appear next to the **Formula** › **Locale** subtab. Select the new **Formulas** › **Locale** tab.
7. The **Formalas** › **Locale** tab contains options for setting the language, keyboard layout, timezone, and whether hardware clock is set to UTC. Select the desired options and click **[Save]**.
8. Run the following command to verify pillar settings. The output has been truncated.

```
salt '$your_client' pillar.items
```

```
...
   keyboard_and_language:
       ----------
       keyboard_layout:
           English (US)
       language:
           English (US)
   machine_password:
       foobar
   mgr_server:
       manager_server
   org_id:alt '$your_client_here'
       1
   timezone:
       ----------
       hardware_clock_set_to_utc:
           True
       name:
           CET
    ...
```

9. Apply this state to your client by applying the highstate from the command line with:

```
salt '$your_client' state.highstate
```

You can also apply the highstate from the previous formula tab from the Uyuni Web UI by selecting **System Details › States** and clicking **[Apply Highstate]**.

SSH Integration

This section provides an overview of the Salt SSH integration with SUSE Manager. This integration adds support for both ssh-push and ssh-push-tunnel connections for Salt clients.

SSH Push Overview

Like the traditional stack, Salt clients may use an ssh connection to manage clients in place of Zeromq. This additional functionality is based on Salt SSH. Salt SSH enables you to execute salt commands and states via ssh without ever needing to install a salt client.

When the server executes an action on a client an ssh connection is made on demand. This connection differs from the always-connected mode used by clients managed via Zeromq.

In SUSE Manager there are two ssh-push methods. In both use cases the server initiates an ssh connection to the client in order to execute a Salt call using `salt-ssh`. The difference in the two methods is how `zypper/yum` initially connects to the server repositories:

zypper Connection Methods:

ssh-push

zypper works as usual. The http(s) connection to the server is created directly.

ssh-push-tunnel

The server creates an http(s) connection through an ssh tunnel. The http(s) connection initiated by `zypper` is redirected through the tunnel by means of `/etc/hosts` aliasing (see below). This method should be used for in place firewall setups that block http(s) connections from a client to the server.

Salt SSH Integration

As with all Salt calls, SUSE Manager invokes `salt-ssh` via the `salt-api`.

Salt SSH relies on a Roster to obtain details such as hostname, ports, and ssh parameters of an ssh client. SUSE Manager keeps these details in the database and makes them available to Salt by generating a temporary Roster file for each salt-ssh call. The location of the temporary Roster file is supplied to salt-ssh using the `--roster-file= option`.

Authentication

salt-ssh supports both password and key authentication. SUSE Manager uses both methods:

Password and Key Authentication:

Bootstrapping Authentication

Password authentication is used only when bootstrapping. During the bootstrap step the key of the server is not authorized on the client and therefore a password must be utilized for a connection to be

made. The password is used transiently in a temporary roster file used for bootstrapping. This password is not stored.

Common Salt Call Authentication

All other common salt calls use key authentication. During the bootstrap step the ssh key of the server is authorized on the client (added to a client's `~/.ssh/authorized_keys` file). Therefore subsequent calls no longer require a password.

User Account for salt-ssh Calls

The user for `salt-ssh` calls made by SUSE Manager is taken from the `ssh_push_sudo_user` setting. The default value of this is `root`.

If the value of `ssh_push_sudo_user` is not `root` then the `--sudo` options of `salt-ssh` are used.

SSH Push Tunnel HTTP(s) Redirection

For the `ssh-push-tunnel` method the traffic originating from zypper/yum has to be redirected through an ssh tunnel in order to bypass any firewall blocking a direct connection from the client to the server.

This is achieved by using port `1233` in the repo url:

```
https://suma-server:1233/repourl...
```

Next alias the suma-server hostname to localhost in /etc/hosts:

```
127.0.0.1       localhost    suma-server
```

The server creates a reverse ssh tunnel that connects `localhost:1233` on the client to `suma-server:443` (`ssh ⋯ -R 1233:suma-server:443`)

The result is that zypper/yum will actually connect to `localhost:1233` which is then forwarded to `suma-server:443` via the ssh tunnel.

This implies that zypper can contact the server only if the tunnel is open. This happens only when the servers executes an action on the client. Manual zypper operations that require server connectivity are not possible in this case.

SUSE Manager Salt SSH Call Sequence

1. Prepare the Salt Roster for the call
    a. Create remote port forwarding option IF the contact method is ssh-push-tunnel
    b. Compute the ProxyCommand IF the client is connected through a proxy

c. create Roster content:

- `hostname`
- `user`
- `port`
- `remote_port_forwards`: The remote port forwarding ssh option
- `ssh_options`: other ssh options:
  - `ProxyCommand`: If the client connects through a SUMA proxy
- `timeout`: default 180s
- `minion_opts`:
  - `master`: set to the minion id if contact method is ssh-push-tunnel

2. create a temporary Roster file
3. execute a synchronous salt-ssh call via the API
4. remove the temporary Roster file

Additional Information:

SaltSSHService.callSyncSSH

Bootstrap Process Sequence

Bootstrapping clients uses salt-ssh under the hood. This happens for both regular and ssh client.

The bootstrap sequence is a bit different than the regular salt-ssh call:

1. For a regular client generate and pre-authorize the Salt key of the client
2. If this is an ssh client and a proxy was selected retrieve the ssh public key of the proxy using the mgrutil.chain_ssh_cmd runner. The runner copies the public key of the proxy to the server using ssh. If needed it can chain multiple ssh commands to reach the proxy across multiple hops.
3. Generate pillar data for bootstrap. Pillar data contains:

mgr_server

The hostname of the SUSE Manager server

minion_id

The hostname of the client to bootstrap

contact_method

The connection type

mgr_sudo_user

The user for salt-ssh

activation_key

If selected

minion_pub

The public client key that was pre-authorized

minion_pem

The private client key that was pre-authorized

proxy_pub_key

The public ssh key that was retrieved from the proxy if the target is an ssh client and a proxy was selected

4. If contact method is `ssh-push-tunnel` fill the remote port forwarding option
5. if the client connects through a SUMA proxy compute the `ProxyCommand` option. This depends on the path used to connect to the proxy, e.g. server → proxy1 → proxy2 → client
6. generate the roster for bootstrap into a temporary file. This contains:
   - `hostname`
   - `user`
   - `password`
   - `port`
   - `remote_port_forwards`: the remote port forwarding ssh option
   - `ssh_options`: other ssh options:
     - `ProxyCommand` if the client connects through a SUMA proxy
   - `timeout`: default 180s
7. Via the Salt API execute:

```
salt-ssh --roster-file=<temporary_bootstrap_roster> minion state.apply
certs,<bootstrap_state>`
```

<bootstrap_state> replaceable by **bootstrap** for regular clients or **ssh_bootstrap** for ssh clients.

The following image provides an overview of the Salt SSH bootstrap process.

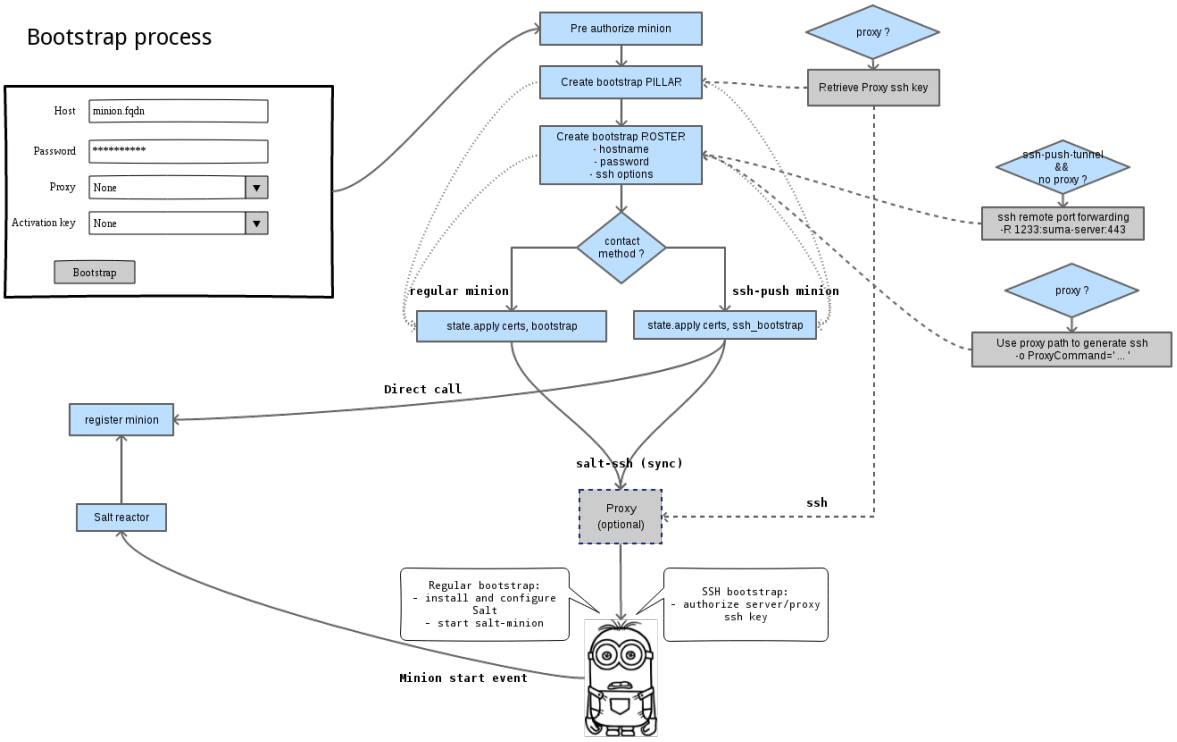


Figure 10. Salt SSH Bootstrap Process

Additional Information:

- SSHMinionBootstrapper.java
- RegularMinionBootstrapper.java
- bootstrap/init.sls
- ssh_bootstrap/init.sls

Proxy Support

In order to make salt-ssh work with SUSE Managers proxies the ssh connection is chained from one server/proxy to the next. This is also know as multi-hop or multi gateway ssh connection.

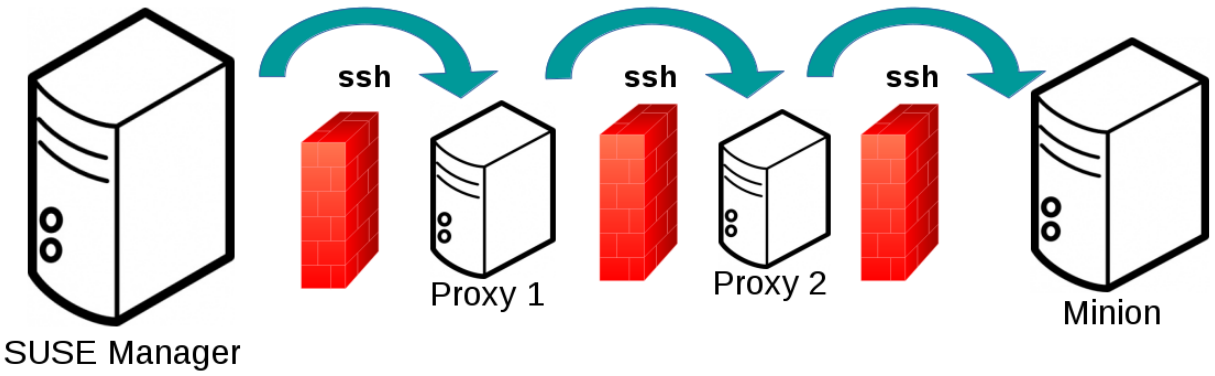


Figure 11. Salt SSH Proxy Multiple Hops

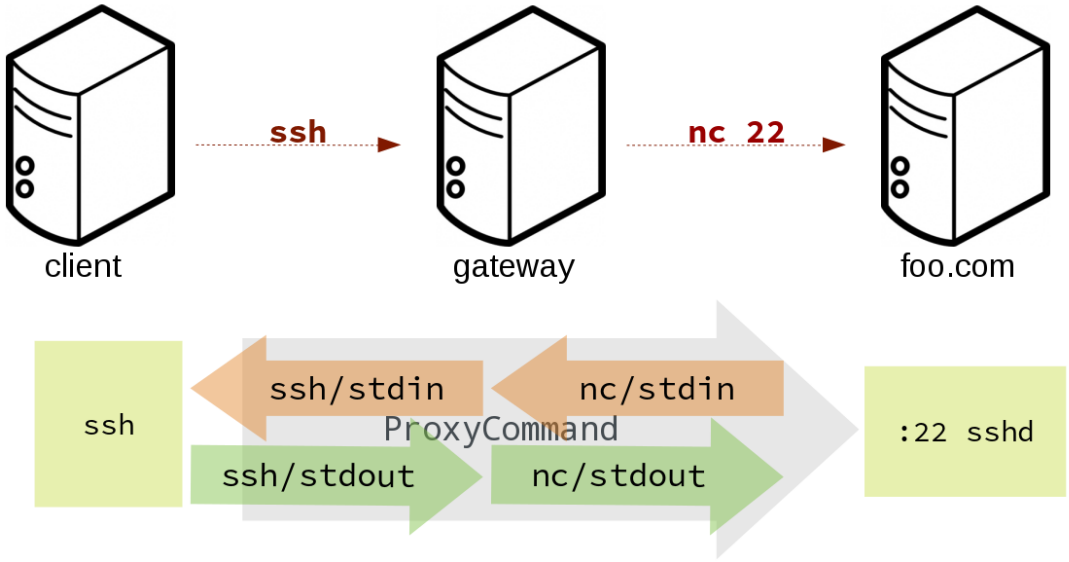
The ProxyCommand

In order to redirect the ssh connection through the proxies the ssh `ProxyCommand` option is used. This options invokes an arbitrary command that is expected to connect to the ssh port on the target host. The standard input and output of the command is used by the invoking ssh process to talk to the remote ssh daemon.

The ProxyCommand basically replaces the TCP/IP connection. It doesn't do any authorization, encryption, etc. Its role is simply to create a byte stream to the remote ssh daemon's port.

E.g. connecting to a server behind a gateway:

```
ssh -o ProxyCommand=<stdio/stdout to remote port> ...
ssh -o ProxyCommand='ssh gateway nc foo.com 22' root@foo.com
```



In this example netcat (nc) is used to pipe port 22 of the target host into the ssh std i/o.

Salt SSH Call Sequence via Proxy

Salt SSH Call sequence via a proxy.

1. SUSE Manager initates the ssh connections as described above.
2. Additionally the ProxyCommand uses ssh to create a connection from the server to the client through the proxies.

Twin Proxies and SSH Push

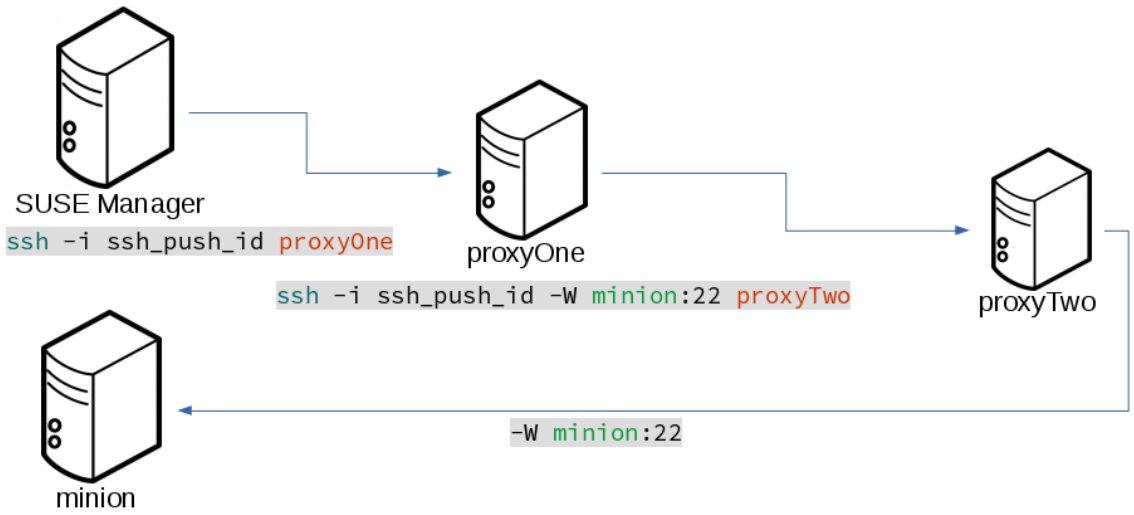
The following example uses the ProxyCommand option with two proxies and the usual ssh-push method

This is a test.

```
# 1
/usr/bin/ssh -i /srv/susemanager/salt/salt_ssh/mgr_ssh_id -o StrictHostKeyChecking=no -o
User=mgrsshtunnel  proxy1
# 2
/usr/bin/ssh -i /var/lib/spacewalk/mgrsshtunnel/.ssh/id_susemanager_ssh_push -o
StrictHostKeyChecking=no -o User=mgrsshtunnel -W client:22  proxy2
```

1. Connect from the server to the first proxy
2. Connect from the first proxy to the second and forward standard input/output on the client to client:22 using the -W option.

```
ssh -i salt_ssh_id -o ProxyCommand='ssh -i ssh_push_id proxyOne ssh -i
ssh_push_id proxyTwo -W minion:22' root@minion <cmd>
```



Twin Proxies and SSH Push Tunnel

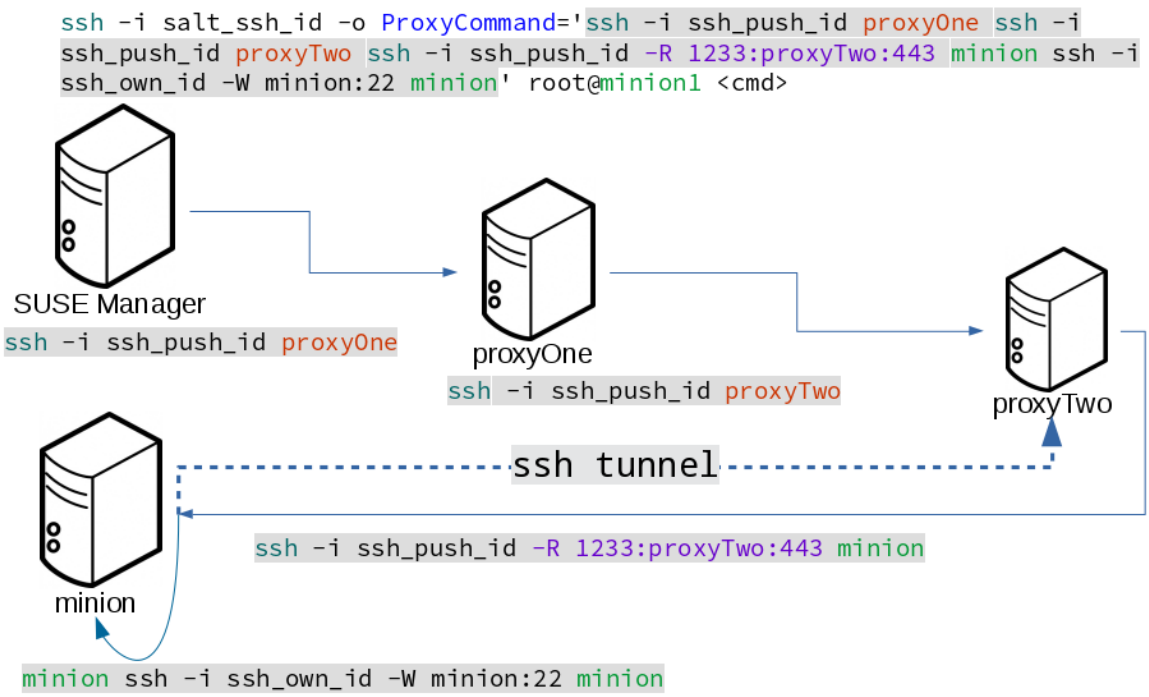
The following example uses the ProxyCommand option with two proxies over an ssh-push-tunnel connection:

```
# 1
/usr/bin/ssh -i /srv/susemanager/salt/salt_ssh/mgr_ssh_id -o User=mgrsshtunnel  proxy1
# 2
/usr/bin/ssh -i /home/mgrsshtunnel/.ssh/id_susemanager_ssh_push -o User=mgrsshtunnel  proxy2
# 3
/usr/bin/ssh -i /home/mgrsshtunnel/.ssh/id_susemanager_ssh_push -o User=root -R
1233:proxy2:443 client
# 4
/usr/bin/ssh -i /root/.ssh/mgr_own_id -W client:22 -o User=root client
```

1. Connect from the server to the first proxy.
2. Connect from the first proxy to the second.
3. connect from the second proxy to the client and open an reverse tunnel (-R 1233:proxy2:443) from

the client to the https port on the second proxy.

4. Connect from the client to itself and forward the std i/o of the server to the ssh port of the client (-W client:22). This is equivalent to ssh … proxy2 netcat client 22 and is needed because ssh doesn't allow to have both the reverse tunnel (-R 1233:proxy2:443) and the standard i/o forwarding (-W client:22) in the same command.



Additional Information:

- SaltSSHService.sshProxyCommandOption

Users and SSH Key Management

In order to connect to a proxy the parent server/proxy uses a specific user called `mgrsshtunnel`.

The ssh config `/etc/ssh/sshd_config` of the proxy will force the execution of `` `/usr/sbin/mgr-proxy-ssh-force-cmd `` when `mgrsshtunnel` connects.

`` `/usr/sbin/mgr-proxy-ssh-force-cmd `` is a simple shell script that allows only the execution of `scp`, `ssh` or `cat` commands.

The connection to the proxy or client is authorized using ssh keys in the following way:

1. The server connects to the client and to the first proxy using the key in `` `/srv/susemanager/salt/salt_ssh/mgr_ssh_id ``.
2. Each proxy has its own key pair in

`/home/mgrsshtunnel/.ssh/id_susemanager_ssh_push`.

3. Each proxy authorizes the key of the parent proxy or server.
4. The client authorized its own key.

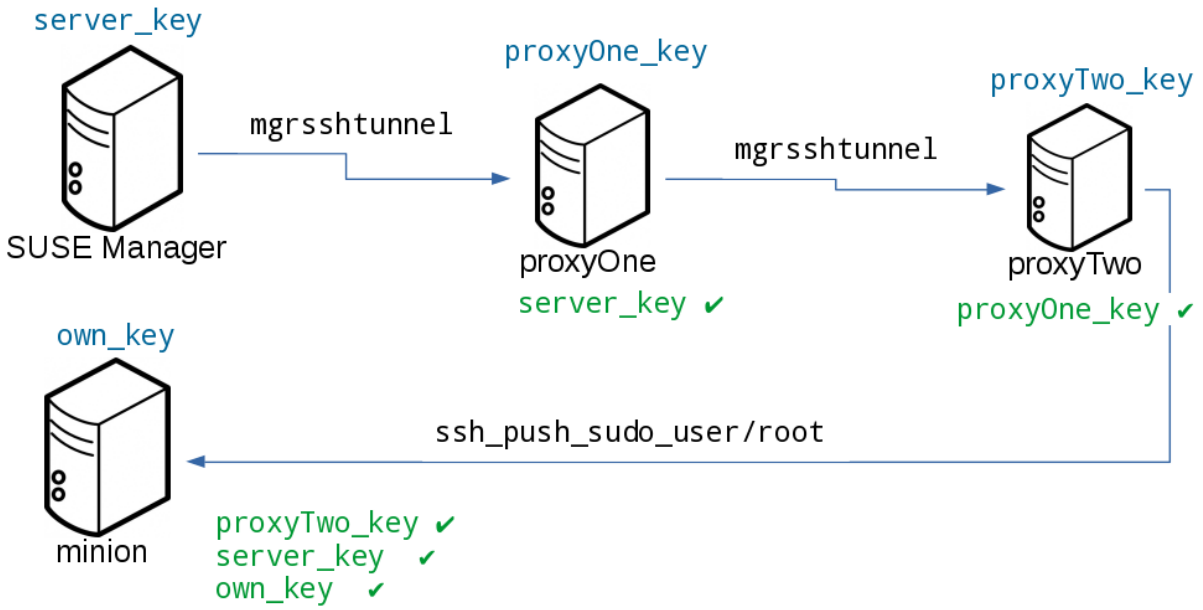


Figure 12. Salt SSH Key Authorization Process

Additional Information:

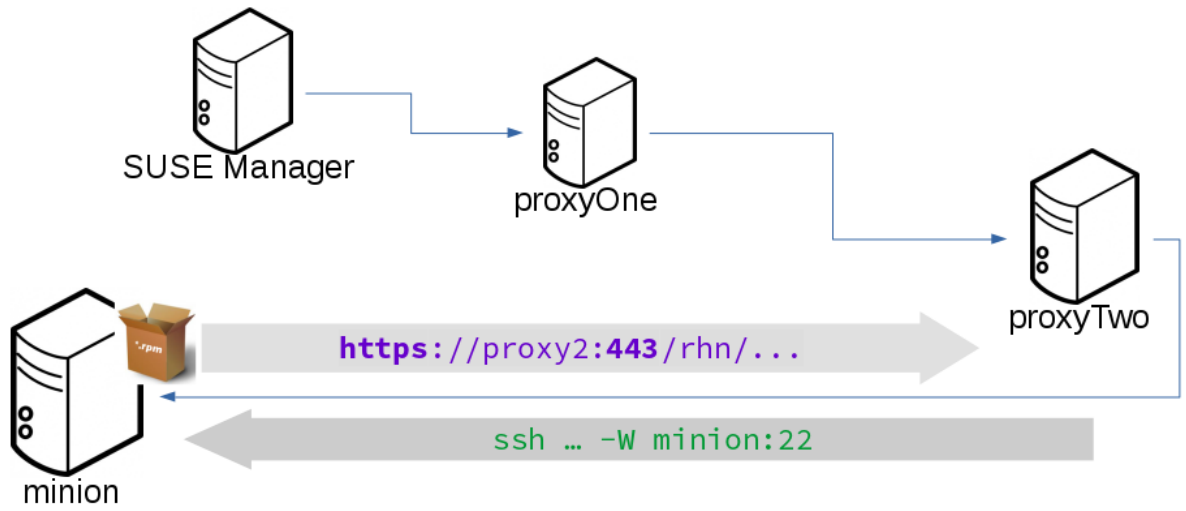
- mgr-proxy-ssh-force-cmd

Repository access via proxy

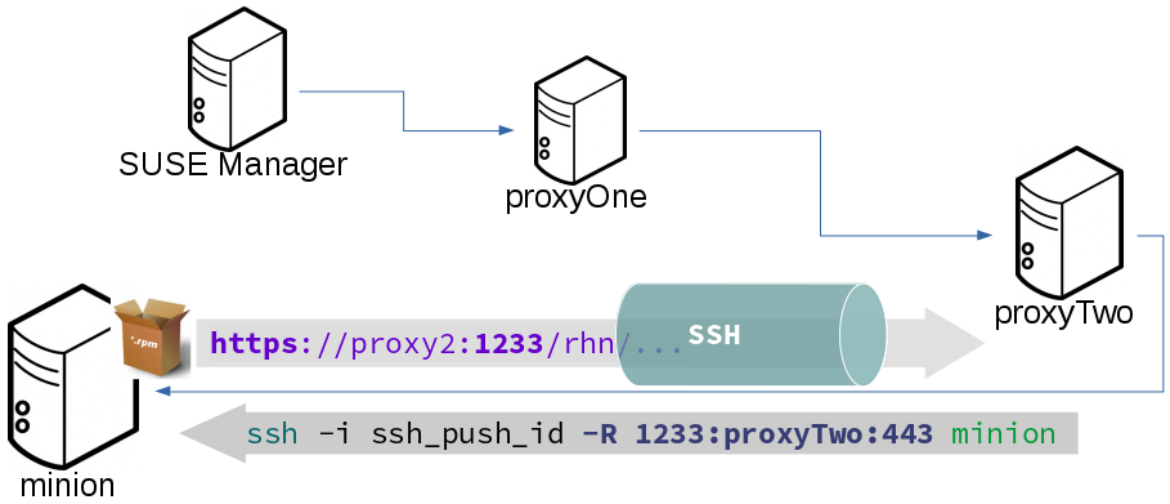
For both ssh-push and ssh-push-tunnel the client connects to the proxy to retrieve packages and repo data.

The difference is how the connection works:

- In case of ssh-push, zypper or yum connect directly to the proxy using http(s). This assumes there's not firewall between the client and the proxy that would block http connections initiated by the client.



- In case of ssh-push-tunnel, the http connection to the proxy is redirected through a reverse ssh tunnel.



Proxy setup

When the `spacewalk-proxy` package is installed on the proxy the user `mgrsshtunnel` is created if it doesn't already exist.

During the initial configuration with `configure-proxy.sh` the following happens:

1. Generate a ssh key pair or import an existing one
2. Retrieve the ssh key of the parent server/proxy in order to authorize it on the proxy
3. Configure the `sshd` of the proxy to restrict the user `mgrsshtunnel`

This configuration is done by the `mgr-proxy-ssh-push-init` script. This is called from `configure-proxy.sh` and the user doesn't have to invoke it manually.

Retrieving the parent key is done by calling an HTTP endpoint on the parent server or proxy.

1. First `https://$PARENT/pub/id_susemanager_ssh_push.pub` is tried. If the parent is proxy this will return the public ssh key of that proxy.
2. If a `404` is received then it's assumed the parent is a server not a proxy and `https://$PARENT/rhn/manager/download/saltssh/pubkey` is tried.
    a. If `/srv/susemanager/salt/salt_ssh/mgr_ssh_id.pub` already exists on the server it's returned.
    b. If the public key doesn't exist (because `salt-ssh` has not been invoked yet) generate the key by calling the `mgrutil.ssh_keygen` runner.

salt-ssh generates a key pair the first time it is invoked in `/srv/susemanager/salt/salt_ssh/mgr_ssh_id`. The previous sequence is needed in case a proxy is configured before salt-ssh was invoked for the first time.

Additional Information:

- com.suse.manager.webui.controllers.SaltSSHController
- mgrutil.ssh_keygen
- mgr-proxy-ssh-push-init
- spacewalk-proxy.spec

Rate Limiting

Salt is able to run commands in parallel on a large number of clients. This can potentially create large amounts of load on your infrastructure. You can use these rate-limiting parameters to control the load in your environment.

These parameters are all configured in the `/etc/rhn/rhn.conf` configuration file.

Salt commands that are executed from the command line are not subject to these parameters.

Batching

There are two parameters that control how actions are sent to clients, one for the batch size, and one for the delay.

When the Salt master sends a batch of actions to the target clients, it will send it to the number of clients determined in the batch size parameter. After the specified delay period, commands will be sent to the next batch of clients. The number of clients in each subsequent batch is equal to the number of clients that have completed in the previous batch.

Choosing a lower batch size will reduce system load and parallelism, but might reduce overall performance for processing actions.

The batch size parameter sets the maximum number of clients that can execute a single action at the same time. Adjust the `java.salt_batch_size` parameter. Defaults to 100.

Increasing the delay increases the chance that multiple clients will have completed before the next action is issued, resulting in fewer overall commands, and reducing load.

The batch delay parameter sets the amount of time, in seconds, to wait after a command is processed before beginning to process the command on the next client. Adjust the `java.salt_batch_delay` parameter. Defaults to 1.0 seconds.

Presence Ping Timeout

There are two parameters that control how presence pings from the Salt master are handled, one for the ping timeout, and one for the ping gather job.

Salt batch calls begin with the Salt master performing a presence ping on the target clients. A ping gather job runs on the Salt master to handle the incoming pings from the clients. Batched commands will begin only after all clients have either responded to the ping, or timed out.

The presence ping is an ordinary Salt command, but is not subject to the same timeout parameters as all other Salt commands (`timeout`/`gather_job_timeout`), rather, it has its own parameters (`presence_ping_timeout`/`presence_ping_gather_job_timeout`). You can configure the

global timeout values in the `/etc/salt/master.d/custom.conf` configuration file. However, to allow for quicker detection of unresponsive clients, the timeout values for presence pings are by default significantly shorter than those used elsewhere. You can configure the presence ping parameters in `/etc/rhn/rhn.conf`, however the default values should be sufficient in most cases.

A lower total presence ping timeout value will increase the chance of false negatives. In some cases, a client might be marked as non-responding, when it is responding but did not respond quickly enough. Additionally, setting this total presence ping timeout value too low could result in a client hanging at the boot screen. A higher total presence ping timeout will increase the accuracy of the test, as even slow clients will respond to the presence ping before timing out. Additionally, a higher presence ping timeout could limit throughput if you are targeting a large number of clients, when some of them are slow.

If a client does not reply to a ping within the allocated time, it will be marked as `not available`, and will be excluded from the command. The Web UI will show a `minion is down` message in this case.

For more information on client timeouts, see scale-minions.pdf.

The presence ping timeout parameter changes the timeout setting for the presence ping, in seconds. Adjust the `java.salt_presence_ping_timeout` parameter. Defaults to 4 seconds.

The presence ping gather job parameter changes the timeout setting for gathering the presence ping, in seconds. Adjust the `java.salt_presence_ping_gather_job_timeout` parameter. Defaults to 1 second.

Disabling the Salt Mine

In older versions, Uyuni used a tool called Salt mine to check client availability. The Salt mine would cause clients to contact the server every hour, which created significant load. With the introduction of a more efficient mechanism in Uyuni 3.2, the Salt mine is no longer required. Instead, the Uyuni server uses Taskomatic to ping only the clients that appear to have been offline for twelve hours or more, with all clients being contacted at least once in every twenty four hour period by default. You can adjust this by changing the `web.system_checkin_threshold` parameter in `rhn.conf`. The value is expressed in days, and the default value is `1`.

Newly registered Salt clients will have the Salt mine disabled by default. If the Salt mine is running on your system, you can reduce load by disabling it. This is especially effective if you have a large number of clients.

Disable the Salt mine by running this command on the server:

```
salt '*' state.sls util.mgr_mine_config_clean_up
```

This will restart the clients and generate some Salt events to be processed by the server. If you have a large number of clients, handling these events could create excessive load. To avoid this, you can execute the command in batch mode with this command:

```
salt --batch-size 50 '*' state.sls util.mgr_mine_config_clean_up
```

You will need to wait for this command to finish executing. Do not end the process with Ctrl+C.

Scaling Salt Clients

Salt Client Onboarding Rate

The rate at which SUSE Manager can on-board clients (accept Salt keys) is limited and depends on hardware resources. On-boarding clients at a faster rate than SUSE Manager is configured for will build up a backlog of unprocessed keys slowing the process and potentially exhausting resources. It is recommended to limit the acceptance key rate pro-grammatically. A safe starting point would be to on-board a client every 15 seconds, which can be implemented via the following command:

```
for k in $(salt-key -l un|grep -v Unaccepted); do salt-key -y -a $k; sleep 15; done
```

Clients Running with Unaccepted Salt Keys

Clients which have not been on-boarded, (clients running with unaccepted Salt keys) consume resources, in particular inbound network bandwidth for ~2.5 Kb/s per client. 1000 idle clients will consume around ~2.5 Mb/s, and this number will drop to almost 0 once on-boarding has been completed. Limit non-onboarded systems for optimal performance.

Salt's official documentation suggests the maximum number of opened files should be set to at least 2 × the client count. Current default is 16384, which is sufficient for ~8000 clients. For larger installations, this number may be increased by editing the following line in `/usr/lib/systemd/system/salt-master.service` :

```
LimitNOFILE=16384
```

Salt Timeouts

Background Information

Salt features two timeout parameters called `timeout` and `gather_job_timeout` that are relevant during the execution of Salt commands and jobs—it does not matter whether they are triggered using the command line interface or API. These two parameters are explained in the following article.

This is a normal workflow when all clients are well reachable:

- A salt command or job is executed:

  ```
  salt '*' test.ping
  ```

- Salt master publishes the job with the targeted clients into the Salt PUB channel.
- Clients take that job and start working on it.

- Salt master is looking at the Salt RET channel to gather responses from the clients.
- If Salt master gets all responses from targeted clients, then everything is completed and Salt master will return a response containing all the client responses.

If some of the clients are down during this process, the workflow continues as follows:

1. If `timeout` is reached before getting all expected responses from the clients, then Salt master would trigger an aditional job (a Salt `find_job` job) targeting only pending clients to check whether the job is already running on the client.
2. Now `gather_job_timeout` is evaluated. A new counter is now triggered.
3. If this new `find_job` job responses that the original job is actually running on the client, then Salt master will wait for that client's response.
4. In case of reaching `gather_job_timeout` without having any response from the client (neither for the initial `test.ping` nor for the `find_job` job), Salt master will return with only the gathered responses from the responding clients.

By default, Uyuni globally sets `timeout` and `gather_job_timeout` to 120 seconds. So, in the worst case, a Salt call targeting unreachable clients will end up *with 240 seconds of waiting* until getting a response.

Salt SSH Clients (SSH Push)

Salt SSH clients are slightly different that regular clients (zeromq). Salt SSH clients do not use Salt PUB/RET channels but a wrapper Salt command inside of an SSH call. Salt `timeout` and `gather_job_timeout` are not playing a role here.

Uyuni defines a timeout for SSH connections in `/etc/rhn/rhn.conf`:

```
# salt_ssh_connect_timeout = 180
```

The presence ping mechanism is also working with SSH clients. In this case, Uyuni will use `salt_presence_ping_timeout` to override the default timeout value for SSH connections.

GNU Free Documentation License

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections

then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document’s license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version’s license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

```
Copyright (c) YEAR YOUR NAME.
   Permission is granted to copy, distribute and/or modify this document
   under the terms of the GNU Free Documentation License, Version 1.2
   or any later version published by the Free Software Foundation;
   with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
   A copy of the license is included in the section entitled{ldquo}GNU
   Free Documentation License{rdquo}.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “ with…Texts.” line with this:

```
with the Invariant Sections being LIST THEIR TITLES, with the
   Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.