AUGUMENTS FILE SETUP

DEF.JSON PER HOST

Bas van der Vlies

SURF SARA

TOPICS

- INTRODUCTION
- HISTORY
- MUSTACHE/JSON switch
- DEF.JSON
- CONCLUSION

INTRODUCTION (1/2)

- Bas van der Vlies
- SURFsara
- SGI computers (CF1)
- Beowulf cluster (CF2)
- National compute cluster (CF3)

INTRODUCTION (2/2)

- pbs_python
- email2trac
- SALI - Automatic Linux Installation
- CMT - Config Management Tool
- oss.trac.surfsara.nl/gitlab.com

HISTORY (1/2)

- no Master Policy Framework
- own template language
- no json support
- no standard location for variables

def.cf/def.json

- template data in policy files

HISTORY (2/2)

- OFFICE automation
- Move template data from the policy files
- EFL Framework by Neil Watson
  - https://github.com/neilhwatson/evolve_cfengine_freelib
- Henc by Marco Marongui
  - https://github.com/brontolinux/hENC
- MUSTACHE/JSON switch

MUSTACHE/JSON switch (1/6)

EASY in ntp.cf:

```
”ntp" usebundle => sara_mustache_autorun("ntp");
```

MUSTACHE/JSON switch (2/6)

```
bundle common ntp()
...
vars:

"template_2_destination" data => parsejson
{
  "ntp.mustache" : "$(config_file)",
  "ntp_default.mustache" : "$(ntp_default)",
  "ntpdate_default.mustache" : "$(ntpdate_default)",
}
...
```

MUSTACHE/JSON switch (3/6)

cf-serverd.conf: (policy hub)

```
"/data/cfengine3/templates”
  comment => ”Template dir for services”
  shortcut => "cf_templates_dir",
```

NTP template directory layout:

```
* /data/cfengine3/templates/ntp
      ntpdate_default.mustache, ntp_default.mustache,
      ntp.mustache
  * /data/cfengine3/templates/ntp/json
      default.json, debian.json, centos.json
```

MUSTACHE/JSON switch (4/6)

Default.json for ntp

```
"daemon_options": "",
"restrict": [
        { "rule" : "127.0.0.1 nomodify", "desc" : "Local users
may interrogate the ntp server more closely" },
        { "rule" : "-6 ::1 nomodify", "desc" : "Local users
may interrogate the ntp server more closely" }
],
"server": [ "chime2.surfnet.nl", "ntp0.nl.uu.net” ]
```

MUSTACHE/JSON switch (5/6)

ntpdate_default.mustache

```
{{#classes.debian}}
NTPSERVERS="{{#vars.sara_data.ntp.server}} {{{.}}}
{{/vars.sara_data.ntp.server}}"
{{/classes.debian}}

{{#classes.centos}}
OPTIONS="-U ntp -s -b {{#vars.sara_data.ntp.server}} {{{.}}}
{{/vars.sara_data.ntp.server}}"
{{/classes.centos}}
```

MUSTACHE/JSON switch (6/6)

- ntp.template_2_destinations
- default.json
- def.ntp_json_files (def.cf)

```
"ntp_json_files" slist => { "debian.json" }
```

- JSON file(s) merged

```
CFengine variables in json data are expanded: $(sys.policy_hub)
```

- Template(s) are generated

```
{{#vars.sara_data.ntp.server}} {{.}} {{/vars.sara_data.ntp.server}}
```

DEF.JSON (1/5)

Mustache shortcomings

- lot of json files
- json file(s) specification (def.cf)
- security
- No host overview

DEF.JSON (2/5)

```
{
    "classes": {
        "NTP_BUNDLE": "any",
        "SSH_BUNDLE": "any",
        "TSM_BUNDLE": "any"
    },
    "vars": {
        "ntp_json_files" : [ "debian.json" ],
        "tsm" : {
            "nodename": "X0059_PIEPERPODIUM"
        }
    }
}
```

DEF.JSON (3/5)

cf-serverd.conf:

```
"/cmdb/host/$(connection.hostname).json"
   comment => ”Hostname def.json",
   shortcut => "cf_host_def_json",
   admit_hostnames => { "$(connection.hostname)" };
```

- $(connection.key)
  - shortcut => "cf_host_key_def_json"

DEF.JSON (4/5)

update.conf:

```
"$(sys.inputdir)/def.json"
  copy_from => u_copy("cf_host_def_json"),
  perms => update_perms("700"),
  classes => u_repaired(update_host_def_json);
```

- cf_host_key_def_json

DEF.JSON (5/5)

sara_mustache_autorun:

1. *def.ntp_json_files* merged
2. *def.ntp json* data merged

```
"vars": {
  "ntp_json_files" : [ "debian.json" ],
  ”ntp”: {
     "daemon_options": ”this one wins”
  }
```

CONCLUSION

- Data/Policy separation
- Easy host overview
- Cloning a host
- Secure
- No Cfengine knowledge
- Graphical Interface

WISHLIST

- CFE-2365: allow missing sources
- Define namespace for vars
  - To setup a repository for templates
  - multiple json files (CFE-2084)
  - eg: def.template.<service_name>

```
bundle agent ssh_template
{
    vars:
        "ssh" data => parsejson('....'),
            namespace => "def.template";
}
```