# Releases

CFEngine

# Releases

- How do they happen?
- Sneak peek at 3.7.0

# How do releases happen?

- Why is this important?
- A significant change since 3.6.0
- What can be expected from:
  - New releases
  - Release schedule

# Versions

- The 3.6 series (maintenance releases)
  - Stable series
  - Only bug fixes and small optimizations
  - No new features
- The 3.7 series (next feature release)
  - The place for development
  - New features
  - Bigger rewrites and optimizations
  - Once released it becomes new stable series

# What is a feature?

- Anything that changes:
  - Syntax
  - Behavior (other than simple bugs)
  - Storage format
  - Network protocol
  - Query API (Enterprise)
- Not a feature:
  - Additions to masterfiles that are syntax compatible
  - Support for a new platform

# Version focus

- Focusing on 3.6 right now
  - Getting a stable base to work on
  - A lot of focus on tests
  - Platform support
- Focus will then shift to 3.7
  - Most likely 2015 Q1 or early Q2
  - Scope will still be smaller than previous feature releases

# Release timing

- Time based releases
  - Every six weeks a new release
  - Changes not ready will wait for the next
- Aiming for eight releases per year
  - Two feature releases
  - The rest maintenance releases
- It's a guideline, not a hard deadline

# The release

- Community release candidate one week before
  - Feature releases may get it earlier (not decided yet)
- Community and Enterprise released at the same time

# New features in 3.7.0

- Coming features
- Features being considered

# Coming features

- Improved logging format
- YAML support
- New functions

# Improved logging format

- 3.6 logs:

```
2015-01-29T13:05:45+0100  verbose: /default/cfe_internal_management/methods/
'any'/default/cfe_internal_log_rotation/methods: Evaluating promise
'rotate_outputs'
2015-01-29T13:05:45+0100  verbose: /default/cfe_internal_management/methods/
'any'/default/cfe_internal_log_rotation/methods/'rotate_outputs'[0]: Comment
'Delete outputs/* files older than 30 days'
2015-01-29T13:05:45+0100  verbose: /default/cfe_internal_management/methods/
'any'/default/cfe_internal_log_rotation/methods/'rotate_outputs'/default/prunedir
: Augment scope 'prunedir' with variable 'dir' (type: s)
2015-01-29T13:05:45+0100  verbose: /default/cfe_internal_management/methods/
'any'/default/cfe_internal_log_rotation/methods/'rotate_outputs'/default/prunedir
: Augment scope 'prunedir' with variable 'max_days' (type: s)
2015-01-29T13:05:45+0100  verbose: /default/cfe_internal_management/methods/
'any'/default/cfe_internal_log_rotation/methods/'rotate_outputs'/default/prunedir
: Evaluating bundle pass 1
2015-01-29T13:05:45+0100  verbose: /default/cfe_internal_management/methods/
'any'/default/cfe_internal_log_rotation/methods/'rotate_outputs'/default/prunedir
/files: Evaluating promise '$(dir)'
...
```

# Improved logging format

- 3.7 logs:

```
P: ..........................................................
P: BEGIN un-named promise of type "methods" (pass 1)
P:    Promiser/affected object: 'rotate_outputs'
P:    From parameterized bundle:
cfe_internal_log_rotation( {'@(def.cfe_log_dirs)','30'})
P:    Base context class: cfengine_internal_rotate_logs
P:    Container path :
'/default/cfe_internal_management/methods/'any'/default/cfe_internal_log_rotation/m
ethods/'rotate_outputs'[0]: '
P:
P:    Comment:  Delete outputs/* files older than 30 days
P: ..........................................................

B: ******************************************************************
B: BEGIN bundle prunedir( {'@(def.cfe_log_dirs)','30'})
B: ******************************************************************
```

**... Other promises and messages ...**

```
A: Promise was KEPT
P: END methods promise (rotate_outputs...)
```

# Improved logging format

- Headers
- Grouping
- Message type prefixes
  - Ex. “C:” for classes, “B:” for bundles
- No timestamps
  - Can be turned on though

# YAML support

- YAML extends JSON
  - Valid JSON is also valid YAML
- More human readable
- CFEngine implements YAML support through functions:
  - `parseyaml`
  - `readyaml`

# YAML support

- JSON:

```
{
  name: “John Smith”,
  possessions: [
    “House”,
    “Car”
  ]
}
```

- YAML:

```
name: John Smith
possessions:
  - House
  - Car
```

# New functions

- `expandrange`
  - Generate ranges of numbers
- `file_hash`
  - Compute hash of a file of a given type
- `readcsv`
  - Read CSV from a file
  - Similar to `data_readstringarrayidx`, but parsed according to RFC 4180

# Minor additions

- Simplified `methods` promises

```
methods:
   “mypromise” usebundle => “mybundle”; # <-- This is equivalent
   “mybundle”; TODO                          # <-- to this
```

- Shortened `ifvarclass`

```
vars:
 “var1” string => “1”
   ifvarclass => “linux”;   # <-- This is equivalent
 “var1” string => “1”
   if => “linux”;           # <-- to this
 “var1” string => “1”
   ifvarclass => “!linux”;  # <-- And this is equivalent
 “var1” string => “1”
   unless => “linux”;       # <-- to this
```

# Features being considered

- Network promises
- SQLite `databases` promise backend
- Docker support in `guest_environments` promises
- Redesigned `packages` promises

# interfaces

- Set interface properties, such as
  - Link state
  - IP addresses
  - Bridge interfaces
  - Duplex mode
  - Configure OSPF

# routes

- Define pathways in the network
  - Add subnets
  - Define gateways to reach them
  - Define interface leading to subnet

# addresses

- ARP manipulation for layer 2 tunnelling
  - Give explicit MAC addresses for IP addresses
  - Quite specialized
    - VXLAN is an example

# Redesigned package promise

- Too many problems with existing promise
- Redesigned promise
  - Focus on main use cases
    - Install latest or specific version
    - Remove package
    - Pin package (prevent upgrade)
    - Performance
  - Secondary
    - Wildcards and '<', '>' version operators

# Redesigned package promise

- Rewrite the C backend
- A lot of knowledge in masterfiles
  - Reuse as much as possible
  - But change attributes when it makes sense
- Keep the old C backend around for backwards compatitility

# New features in 3.7.0

- What would **you** like to see in 3.7?
- Discussion after