Fedora Unified Network Controller

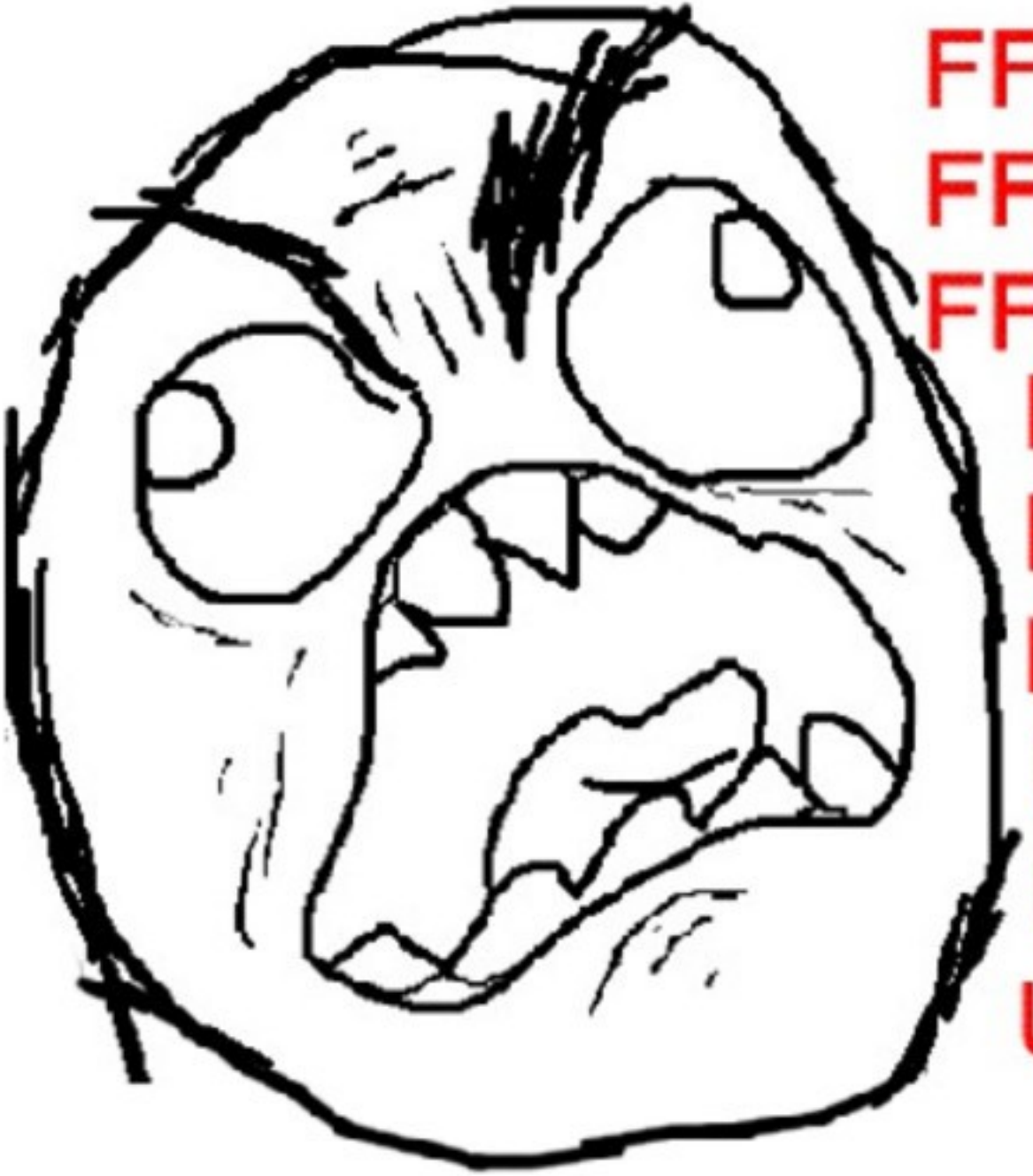
Marek Mahut
mmahut@redhat.com

The problem is simple

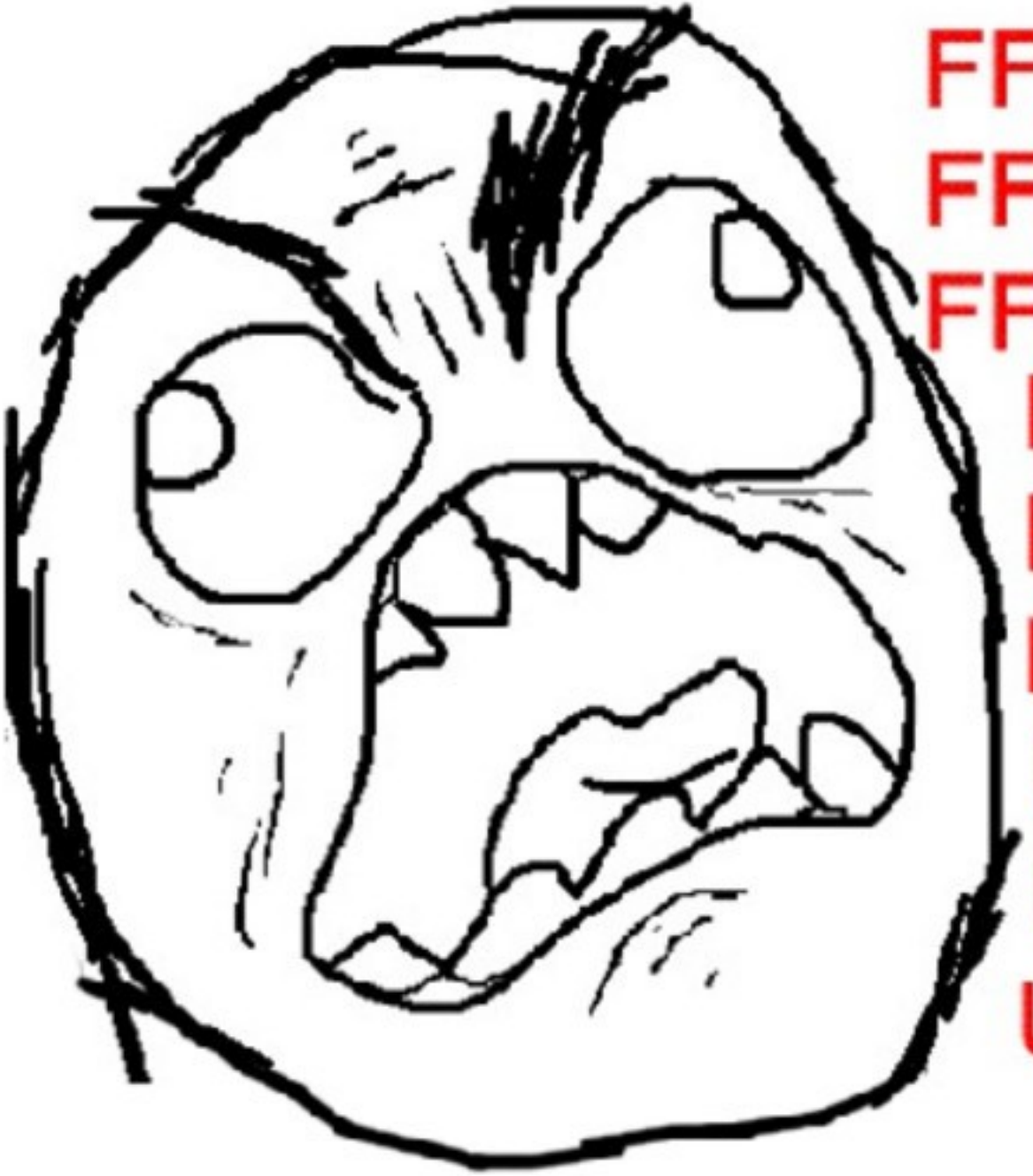
Run this command ...

```
# rpm -qV httpd
```


...
On
this!



FFFFFFF
FFFFFFF
FFFFFFF
FFFFF
FFFFF
FFFFF
UUUU
UUUU
UUUU-



FFFFFFF
FFFFFFF
FFFFFFF
FFFFF
FFFFF
FFFFF
UUUU
UUUU
UUUU-

NC!

The solution is simple

https://fedorahosted.org/func/

What is it?

- **F**edora **U**nified **N**etwork **C**ontroler
- Runs commands on a large number of remote machines
- Gather sane output
- XMLRPC over SSL in Python

What is it?

- Command line or API
- Supports Asynchronous mode
- Supports bindings in YAML, JSON or Java
- Modular

Func Overview

Func Overview

Overlord distributes command to **Minions**

yum -y install mencal

service jboss restart

Func Overview

Minions return output to the **Overlord**

Package installed!

Command returned 0

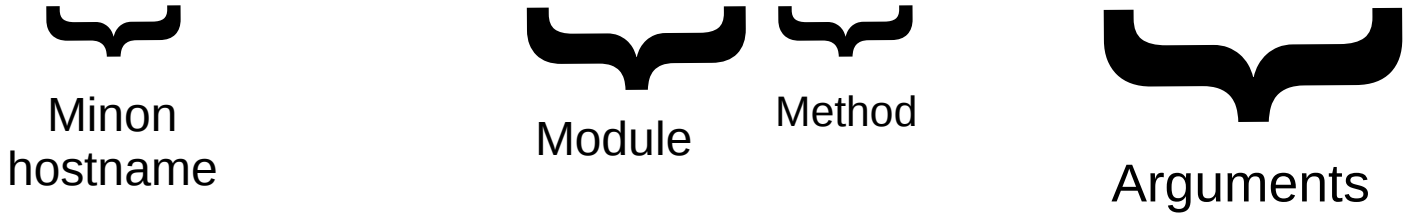
Func examples

Func Command Line

```
func "*" call service restart httpd

func "*.example.org" call yumcmd update

func "db01*" call command run “du –sh /var/db”
```



Func Python API

```
import func.overlord.client as fc

client = fc.Client("web*.example.org")

results = client.yumcmd.install(“httpd”)
results = client.service.start(“httpd”)
results = client.iptables.port.open(80)
```

Func Bindings

- YAML or JSON to func-transmit

```
client: server1.example.com
sync: False
nforks: 1
module: command
method: run
parameters: “uptime”
```

Func Bindings

```
# func-transmit --json < run.yaml
["example1", [0, " 06:19:46 up 12 days...\n", ""]]
```

Or **--yaml** to get result back in YAML


MOAR!
Because MOAR!

Multiplexer

Execute batch of 1000 systems on 10 threads

```
# func --forks=10 "*" call service start puppetd
```

Asynchronous mode

Run (long-running) jobs in parallel without persistent network connection

Groups

Group your systems

```
# func "*" group -la
Group: zimbra
   Host: z01.example.com
   Host: z02.example.com
   Host: z03.example.com
Group: nagios
   Host: n01.example.com
```

Modules

Easy to write!

Command
CopyFile
Disk
GetFile
Jboss
Hardware
Iptables
Nagios
Netapp
Rpm
Smart
Sysctl
...

Modules (example)

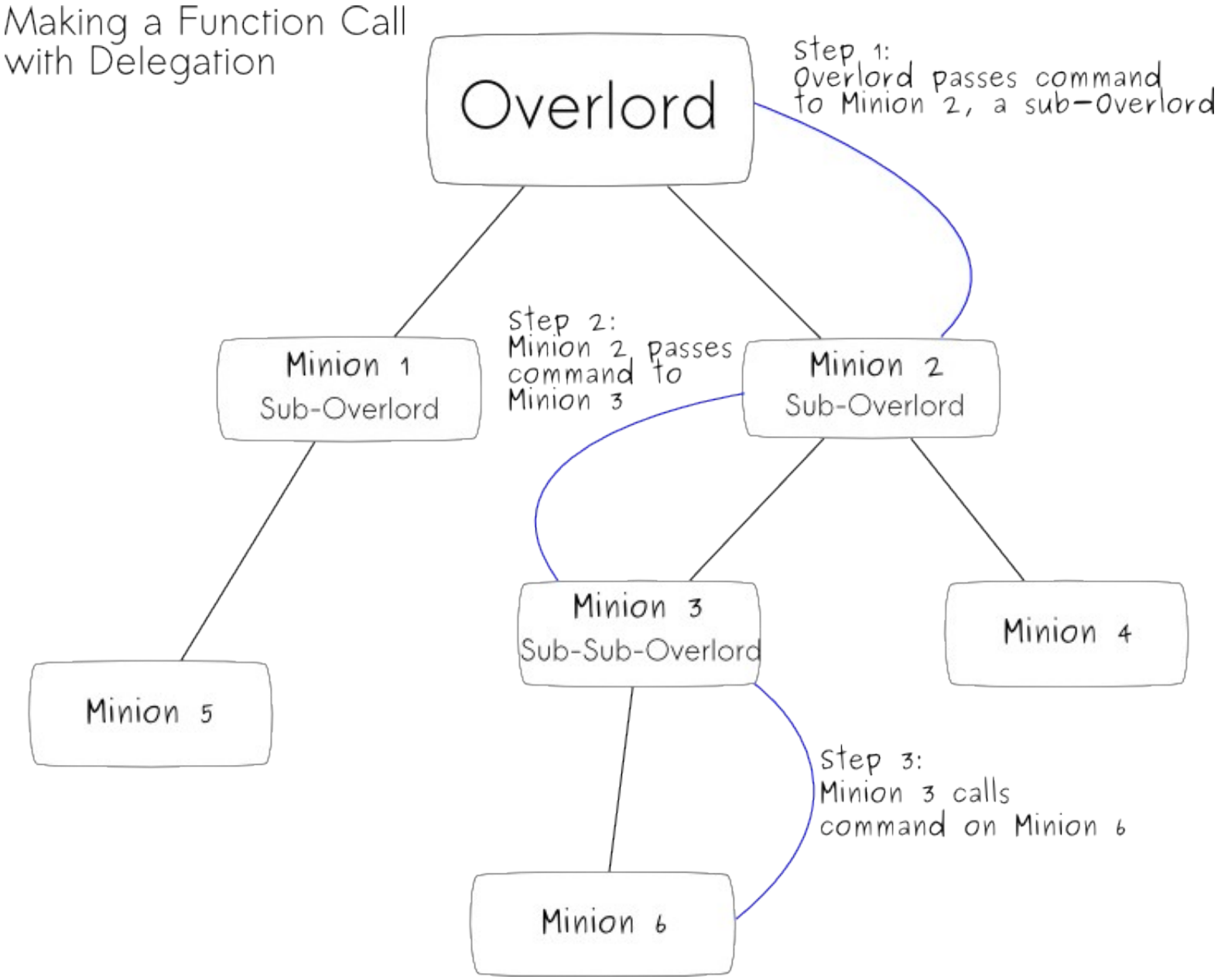
```
import func_module
from func.minion import sub_process

class Reboot(func_module.FuncModule):

    version = "0.0.1"
    api_version = "0.0.1"
    description = "Reboots a machine."

    def reboot(self, when='now', message=''):
        return sub_process.call(["/sbin/shutdown", '-r', ↵
when, message], close_fds=True)
```

Delegation


Making a Function Call with Delegation
Overlord
Step 1: Overlord passes command to Minion 2, a sub-Overlord
Minion 1
Sub-Overlord
Step 2: Minion 2 passes command to Minion 3
Minion 2
Sub-Overlord
Minion 3
Sub-Sub-Overlord
Minion 4
Minion 5
Step 3: Minion 3 calls command on Minion 6
Minion 6

ACLs

X.509 PKI for ACLs
In certificate hashes

```
overlord.example.com = reboot, yum.update
```


OMG! Func rocks!

Genome

Contribute

https://fedorahosted.org/func/

#func on irc.freenode.org

Similar projects

- Digmia Secure Shell (DSSH)
  - htttp://digmia.com/
- Marionetter Collective IP
  - Acquired by Puppet Labs
- Parallel SSH (PSSH)
  - http://code.google.com/p/parallel-ssh/

Questions?

Func you!

Func as in thank,
don't be dirty-minded.

References and credits

http://www.flickr.com/photos/theplanetdotcom/4879421740/

http://www.flickr.com/photos/13106517@N02/4630118749/

http://www.flickr.com/photos/richardlowkes/10302456/

http://www.flickr.com/photos/amoates/5206175248/

http://www.flickr.com/photos/dokas/5289794748/

http://www.flickr.com/photos/a_aepli/4946005857/

http://www.flickr.com/photos/donaldmacleod/4572341460/

http://www.flickr.com/photos/shadowgate/4517058755/

http://www.flickr.com/photos/suswar/4244118452/

http://www.flickr.com/photos/ideacreamanuelapps/3541383441/

http://www.flickr.com/photos/quinnanya/5351196252/