functional Systems Administration

Fedora **U**nified **N**etwork **C**ontroller

https://fedorahosted.org/func/

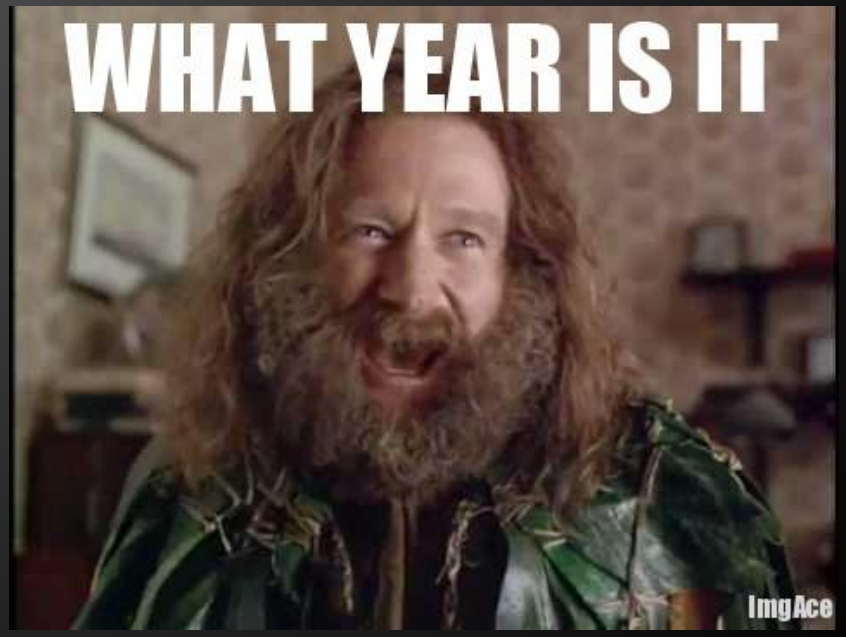
Thomas Howard Uphill (tuphill at narrabilis dot com)
Benjamin Rose (brose at allmybase dot com)

Func?

- Have you ever tried to command or query a large number of systems with SSH? Have you wanted a better way?
- Have you wanted a way to audit all of your remote commands on all of your systems?
- Tired of writing shell scripts and parsing command output?
- Well have we got a solution for you. It's Func.

What is it?

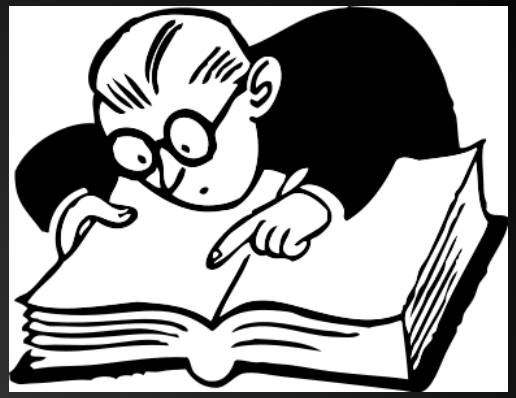
- Python API
- SimpleXMLRPCServer
  (port 51234)
- X509 Certificate Authority
  (certmaster port 51235)
- command line
  `func`



Why would I want to use it?

What it isn't	What it is
• No X11 (cssh)	• Python
• Not expect	• Secure
• Not pdsh	
• sed/awk/tr/xargs	

Terminology

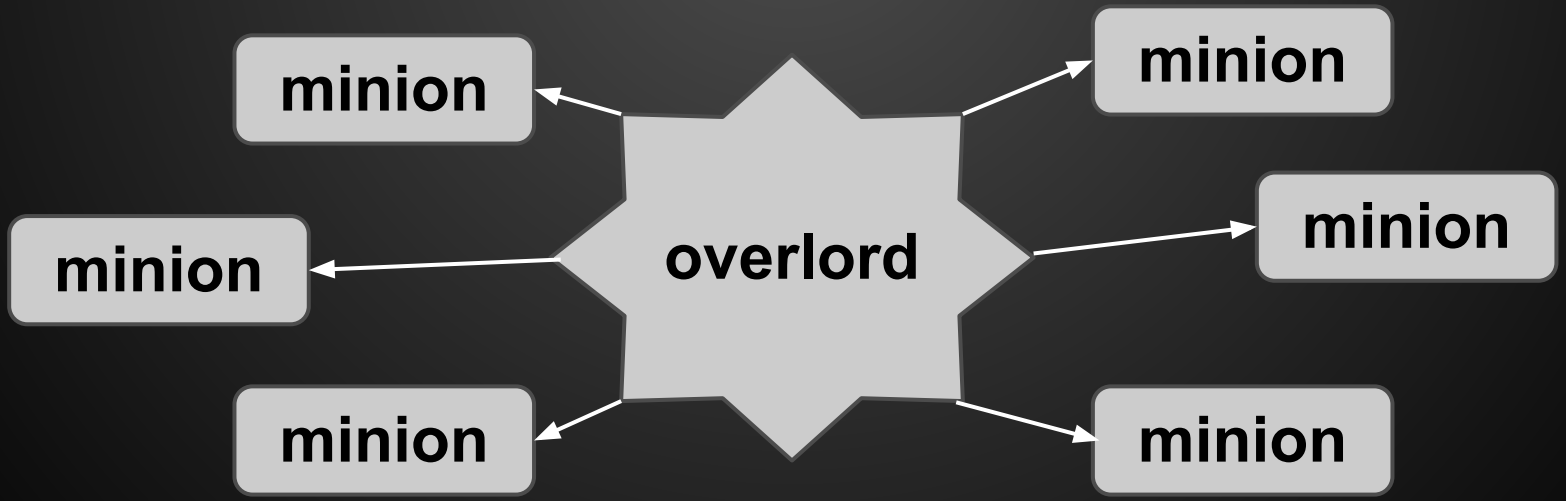


minions

a follower devoted to serving his/her master relentlessly.

overlord

a particularly powerful lord, who has power over many people



PKI

certmaster
https://fedorahosted.org/certmaster/

- client
  - /etc/pki/certmaster
- server
  - /var/lib/certmaster

certmaster-ca, certmaster-request

Allow users to run func

```
setfacl -R -m 'u:username:rX' /var/lib/certmaster
setfacl -d -R -m 'u:username:rX' /var/lib/certmaster
```

Installation/Configuration

- Install the server
- Install the client
- start certmaster
- sign certificates / autosign certificates
- run something

Server

- certmaster
  - install
  - start
- firewall (51235)
- sign certificates

Client

- func
  - install
  - /etc/certmaster/minion.conf
    certmaster=myfunc.example.com
  - start
  - open firewall 51234 (overlord)

Demo

FUNC command line

```
[root@func ~]# func 'minion1*' call command run 'sync'
```

`minion1*` = Host regex

to which hosts shall we apply the following?

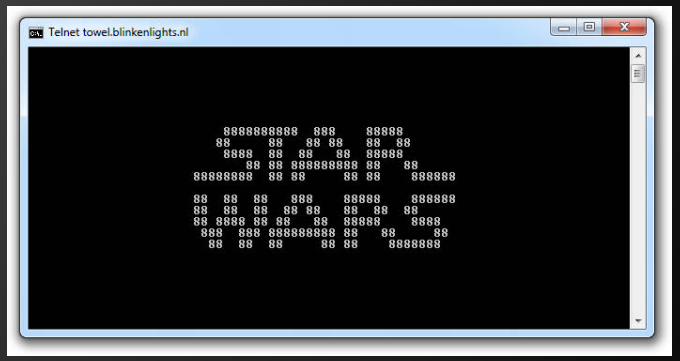
`call` = Command.

Usually "call" but there's also check, ping, etc

`command` = Module

Many are available, which do you choose?

`run` = Method



`'sync'` = Input to the method (if required).

Grouping

```
[root@example1 ~]# cat /etc/func/groups
[sqlservers]
host = sql1.example.com, sql2.example.com

[subdomain]
host = *.subdomain.example.com

>>> from func.overlord import client
>>> client.Client(@sqlservers).service.restart("mysqld");
>>> client.Client(@subdomain).command.run("sync; reboot")
```

facter

Modules

python-site-packages/func/modules

func-create-module

```
import func_module
import os
import subprocess


class Facter(func_module.FuncModule):

    # Update these if need be.
    version = "0.0.1"
    api_version = "0.0.1"
    description = "Puppet Facter Interface, returns facts about the minion"

    def facts(self):
        """
        Return an array of facts
        """
     facter = subprocess.Popen(['facter'], stdout=subprocess.PIPE).communicate()[0]
     facts = {}
     for fact in facter.splitlines():
       (name,value) = fact.split('=>')
       facts[name] = value
        return facts
```

Calling from Python

```
import func.overlord.client as fc
fc.Client('minion_regex').module.method('input')
```

where **minion_regex** can take many forms:

```
   minion1;minion2;@group1;*.example.com
```

Examples

python API

Turning off swap for virtual servers

```
#!/usr/bin/python
import func.overlord.client as fc
info = fc.Client("*").hardware.hal_info()

for host, details in info.iteritems():

if type(details) != dict:
    print "Host {0} had an error.".format(
        host
    )
    break

for (device, full_output) in details.iteritems():
    if full_output.upper().find("VMWARE") != -1:
        fc.Client(host).command.run("swapoff -a")
        print "Disabled swap on host {0}".format(
            host
        )
        break
```

Managing Updates - Report Generation

```
#!/usr/bin/python

import func.overlord.client as fc
client = fc.Client("*")
client.timeout = 100

client.command.run("yum clean all")
output = client.yumcmd.check_update()

for i in sorted(output):
  print "{0}[ {1} ]{0}{2}\n".format(
  "="*25,
  i.upper(),
  (output[i][1])[:-1]
  )
```

Managing Updates - Report Content

```
========================[ EXAMPLE1.CS.PRINCETON.EDU ]========================
ipmitool.x86_64           1.8.11-12.el6_2.1            PUIAS6_core_Updates_local

========================[ EXAMPLE2.CS.PRINCETON.EDU ]========================

========================[ EXAMPLE3.CS.PRINCETON.EDU ]========================

========================[ EXAMPLE4.CS.PRINCETON.EDU ]========================
ipmitool.x86_64           1.8.11-12.el6_2.1            PUIAS6_core_Updates_local

========================[ EXAMPLE5.CS.PRINCETON.EDU ]========================

========================[ EXAMPLE6.CS.PRINCETON.EDU ]========================
ipmitool.x86_64           1.8.11-12.el6_2.1            PUIAS6_core_Updates_local
```

Kernel Version Tracking

```
#!/usr/bin/python
# Thomas Uphill <thomas at narrabilis dot com>
# suggestion to use setdefault from Matthew Cahn

import func.overlord.client as fc
client = fc.Client('*')

hardware = client.hardware.info()
kernels = {}
for host, fields in hardware.items():
        if type(fields) == dict:
             kernels.setdefault(fields['kernelVersion'],[])
             kernels[fields['kernelVersion']].append(host)
k = kernels.keys()
k.sort()
for version in k:
        print '%s:' % version
        for host in kernels[version]:
                print '\t%s' % host
```

Demo

Examples

Command Line

command line usage

Running kernel on all minions:

```
func '*' call command run 'uname -r'
```

Available modules:

```
func minion_name call system list_modules
```

Methods of a module:

```
func minion_name call modulename list_methods
```

Demo

Questions/Comments/Concerns

#func on freenode

brose at allmybase dot com

thomas at narrabilis dot com