Viri

Remote execution of Python scripts

Every time you use Viri, God kills a sysadmin

About me

Python experience

- XML
- Application integration
- Django
  - i18n
- Google App Engine

Working at NTT Europe

- We administer thousands of computers.
- Which means lots of repetitive work.
- As a developer, I try to automate processes.

garcia.marc@gmail.com
http://vaig.be

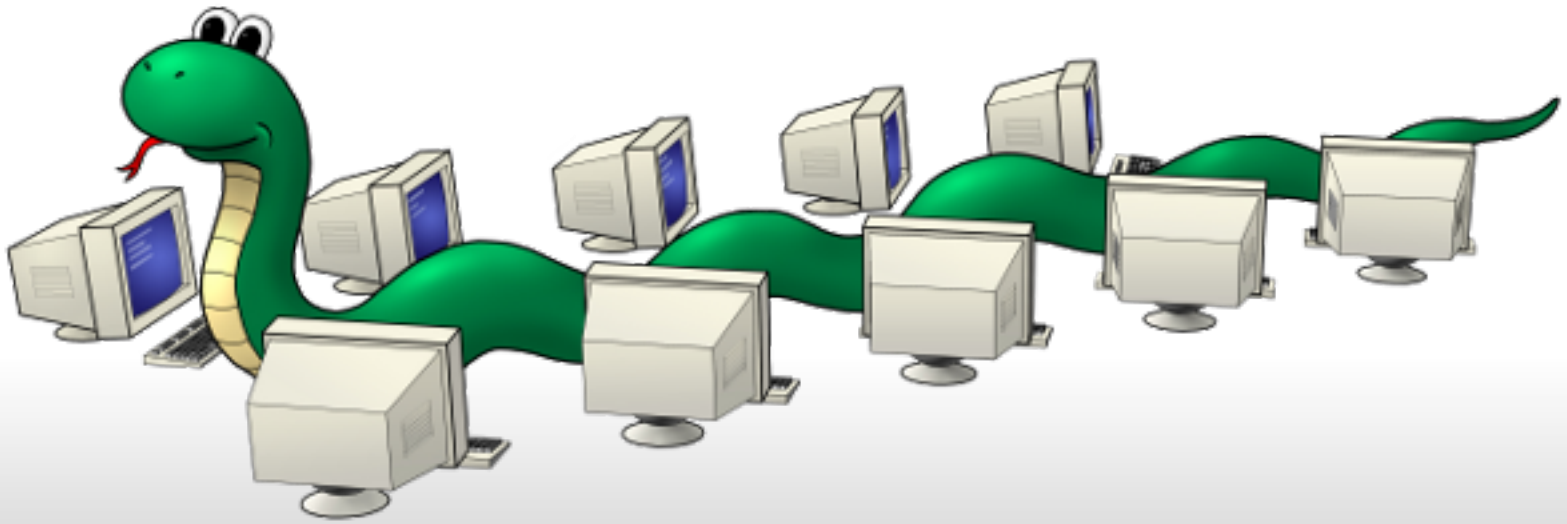
Talk summary

- **What is Viri?**
  - Motivation
  - Features
  - Real case examples
- **How is Viri?**
  - Components
  - Security
- **Using Viri**
  - Viri scripts
  - Commands and options
  - Extra features

What is Viri?

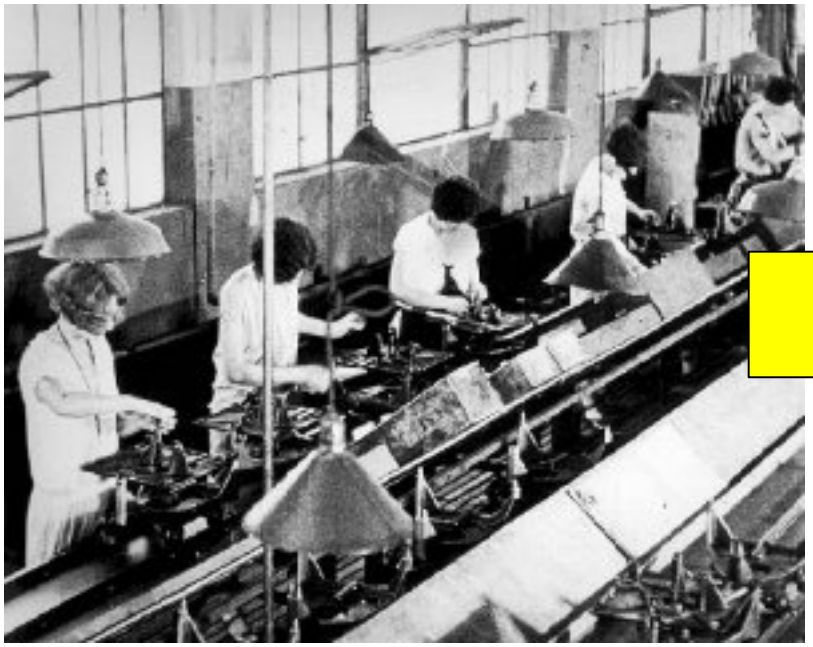
Motivation

Automate administration of large sets of hosts (e.g. datacenters) using **Python** scripts.



Which means, replacing...

...lots of repetitive tasks...



by

...a single creative task

Viri overview

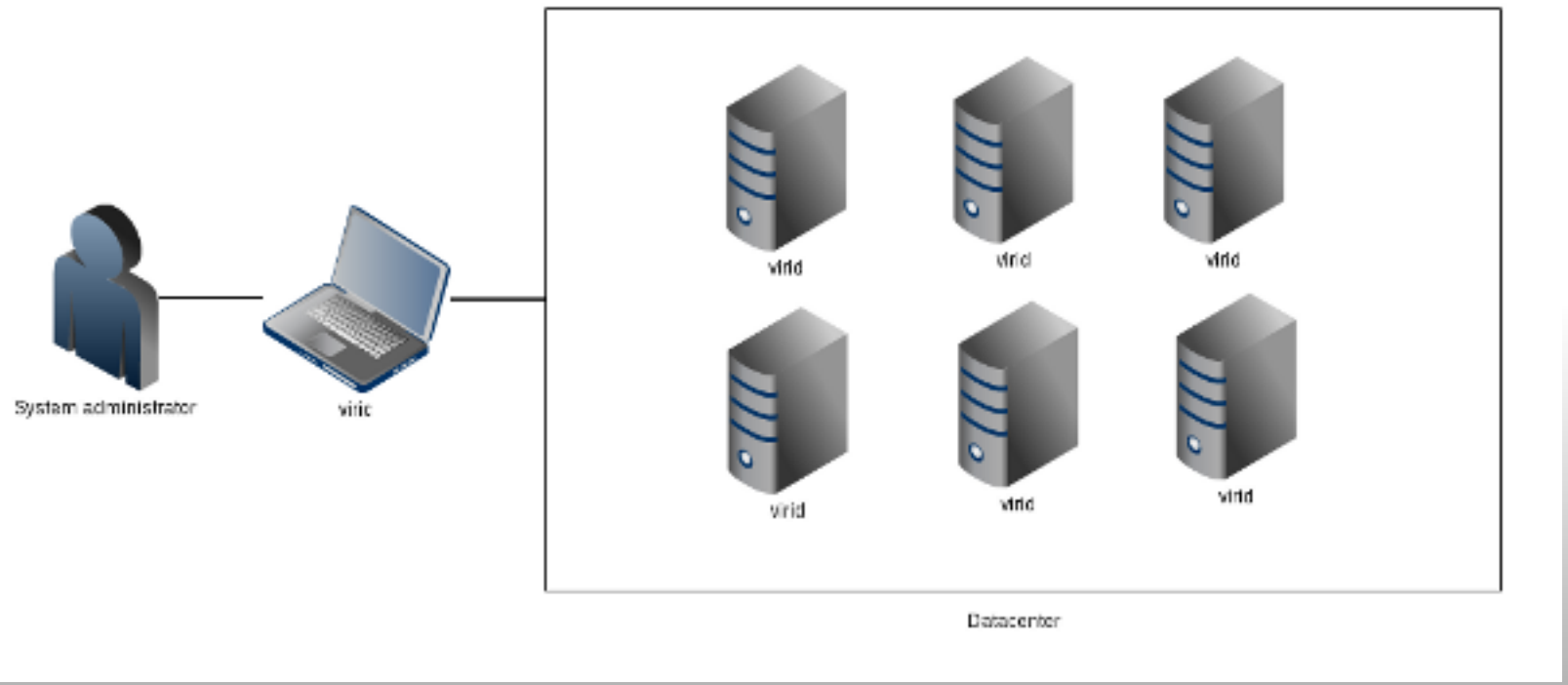
Viri is in BETA status, not ready for production yet, but very close.

- **Python 3** (no other libraries/dependencies)
- XML-RPC (Python provides client/server)
- TLS (for security)
- GPLv3

- Multiplatform
  - Some packaging pending (specially for Windows)
- Custom packaging of Python 3 required
  - Pending for some systems (e.g. Debian 5)

Features

- Automation of tasks over a large set of computers using **Python scripts**.
  - Script deployment
  - Transfer of required data files
  - On-demand or scheduled execution
  - Recording execution history



Real case examples

Gather system data and send to a central location:

- System (Architecture, OS, etc)
- Network (IP addresses, networks, etc)
- User access
- Log information

Implement actions that require per host operations:

- Add users to all hosts /root/.ssh/authorized_keys
- Changes to network configuration
- Complex operations:
  - Parse apache config file
  - Check for errors in a specific directive
  - Fix it!

Any brilliant idea?

How is Viri?

Viri components

virid

Daemon running on remote hosts

- Receives scripts and data
- Records history
- Returns results
- Controls exceptions

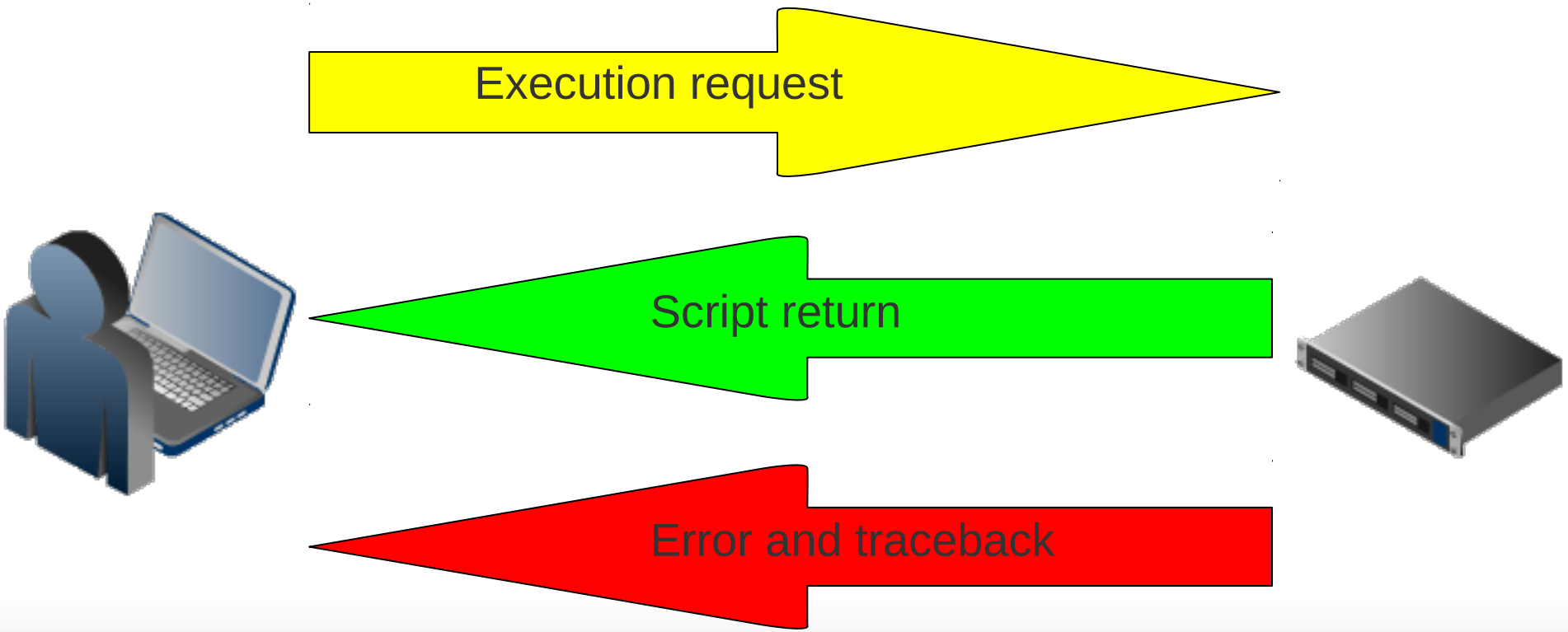
viric

User interface, command line utility.

```
viric execute test.py --host=10.0.0.9
```

Can be integrated with third-party apps.

Execution workflow



Integration with third-party apps

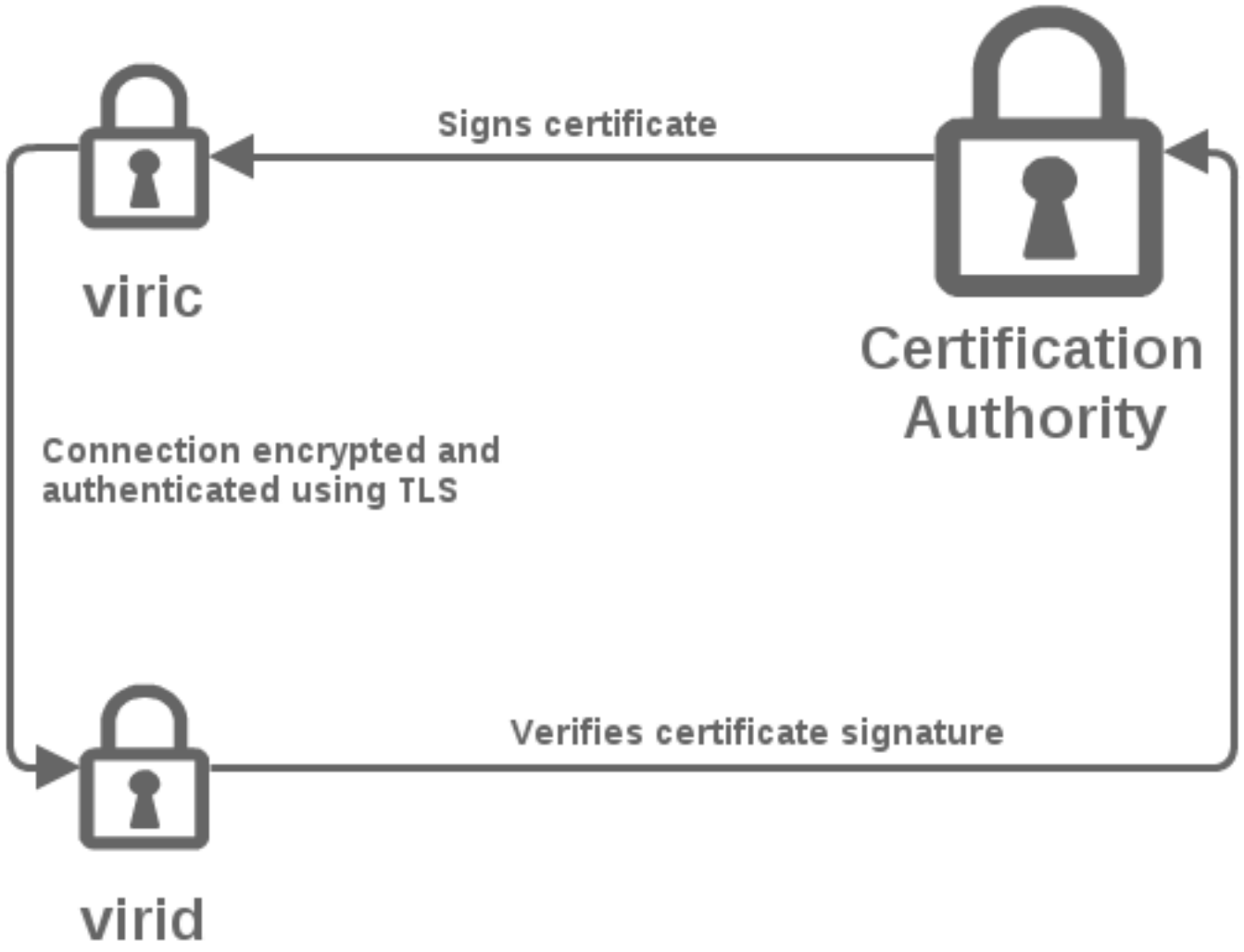
Viri daemon uses XML-RPC. Integrating an application (e.g. a Django app) to communicate with daemons is as easy as writting a XML-RPC client.

```
import xmlrpc.client

def execute_script_by_id(server_url, script_id):
    proxy = xmlrpc.client.ServerProxy(url)
    return proxy.execute({'script_id': script_id})

print(execute_script_by_id(
    url='https://10.0.0.9:6808/',
    script_id='99154c826fca745be859c6481a5f87631e4b2b78'))
```

Actually a little bit more difficult, because we need to authenticate the client, but Viri client is a great example.

Is Viri secure?

Communication is **encrypted** using TLS.
Viri daemon requires **authentication** using a PKI.



Using Viri

Installation

RHEL / CentOS (version 5, so far)

/etc/yum.repos.d/
wget http://www.viriproject.com/redhat/Viri.repo
yum install viri

Debian (version 6 so far)

echo "deb http://www.viriproject.com/debian squeeze main"
>> /etc/apt/sources.list
apt-get update apt-get install viri

Creating a PKI

OpenSSL can do everything we need.

Certification Authority

- Create a private key
- Self sign it and create the certificate
- Sign certificates of users

Users

- Create the private key
- Create the certificate signing request

Daemon instances

- Create a self signed private key

Viri scripts

```
import os

class ViriScript:
    hello_file = '/tmp/viri.hello'

    def say_hello(self):
        with open(self.hello_file) as f:
            f.write('Viri was here!\n')

    def run(self):
        if not os.path.isfile(self.hello_file):
            self.say_hello()
            return 'Viri said hello'
        else:
            return 'Viri has already been here'
```

Basic viric commands

viric COMMAND [OPTIONS]

- help
  - Show usage information
- ls
  - Show installed scripts
  - Show copied data files
- put
  - Send scripts / data files
- get
  - Downloads scripts / data files
- **execute**
  - Executes a script

Basic viric options

viric COMMAND [OPTIONS]

- --host
  - Remote host IP or domain
- --port
  - Remote port (Default is 6808)
- --data
  - On some commands like ls, put or get, specifies that the operation is for data files instead of scripts.

Base script

Special __base__.py *script*:

```
class ViriScript:
    def custom_log(self, msg):
        with open('/tmp/viri.custom_log', 'a') as f:
            f.write('%s\n' % msg)
```

```
./viric put __base__.py --host=10.0.0.9
```

All scripts inherit from it:

```
class ViriScript:
    def run(self):
        # do something
        self.custom_log('I did something')
```

Scheduling

Special __crontab__ *data* file:

./viric put --data __crontab__ --host=10.0.0.9

Cron syntax (using script id):

```
# daily at midnight
0 0 * * * 99154c826fca745be859c6481a5f87631e4b2b78

# Just once, on January 1st, 2015 at 9:00
0 9 1 1 * 2015 99154c826fca745be859c6481a5f87631e4b2b78
```

To conclude

Coming soon

- Use **sqlite** to manage daemon datat
- **Windows**
  - Make fixes to make the daemon run as a Windows service
  - Package for Windows (installer)
- Packages for **more UNIX systems**
  - Debian 5, BSD, Mac OS, etc.

... and make the **first release.**

Other short term plans

- Support full cron syntax
- Save user who requests executions
- Create a public repository for scripts
- Performance optimizations
- Create a Viri community
- Contribute fixes back to Python

More ideas?

Contributing

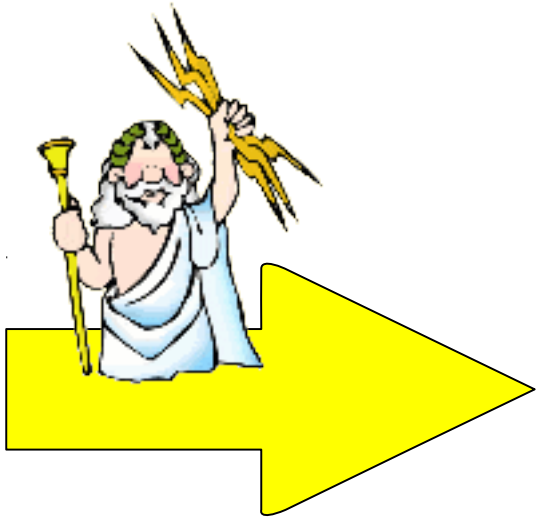
Start hacking:
git://github.com/garcia-marc/viri.git

Discuss ideas:
http://groups.google.com/group/viri-users

Submit bug reports:
https://github.com/garcia-marc/viri/issues/

Share your scripts:
Publish them anywhere, a public repository will be available when having a community

So, will God really kill sysadmins?



No! He will convert them in Python ninjas,
and they will write excellent Viri scripts.