1 Quick Guide to Testing Machine

1.1 About the project

The Autonomous Province of Bolzano currently offers important digital services (for eg, online applications to start businesses, pay property taxes and initiate e-Payments etc), the number is expected to increase as eGoverment services continue to grow.

The FSCRS project intends to contribute to the creation and improvement of eGovernment services through an innovative process that verifies and tests functionality. The underlying idea of the project is that the accessibility of digital public services is necessary to increase the innovation potential of the region. .

The project takes its name from the Free Software Client Reference System, a specific reference system (OS + defined set of software applications) that during testing simulates a user accessing the services.

1.2 About the software

Testingmachine is used to test eGovernment services. The eGovernment services can vary from country to country but generally it grants citizens access to important documents and information. In most EU countries there are additional services like paying property tax that can be done online. Most governments in Europe are leaning in this direction in order to decrease administrative overhead.

Another challenge to overcome will be testing eGov services on mobile platforms, more specifially automating tests under Android. Good news is this is possible using Selenium. The one major obstacle that we face is getting around the smart card login. So far I am unaware of getting this to work under Android unless you patch the kernel and this is of course no an option. More documentation and research is needed concerning this.

1.3 The eGov Testing Machine

The expected result is the development of a systematic, auto validation process, currently not available on the market, which allows the testing of eGovernment services without the manual intervention of an operator

The eGov Testing Machine can be thought of as a virtual group of people, sitting at the computer and using the eGov services and checking if they work properly, allowing the local Public Administration to test eGov services on a daily basis that are being offered to all citizens.

1.4 Software Overview

The Testing Machine is currently made up by Virtual Machine Manager (tm-vmm) and documentation on how to wite, execute and automate tests of eGov sites in particular but also other softwares.

1.4.1 Virtual Machine Manager (VMM)

tm-vmm is made up by bash scripts that let the user manage various virtual machine software in a general way. See the tm-vmm manual for more information.

1.4.2 eGov Manuals

Writing tests of eGov sites is not hard but we believe that some help will still be useful for most people. The manuals and guides provided together with tm-vmm make it possible to test eGov sites automatically and unattended.

2 Features

2.1 Virtual Machine Management

With Testing Machine you can:

- start/pause/stop virtual machines
- take screenshots of running virtual machines
- execute commands in running virtual machines # Installation

2.1.1 Via a dist file (.tar.gz)

Coming later

2.2 Building and installing Virtual Machine Manager

- Go to the vvm directory

```
cd vvm
```

- Configure the software

```
./configure --prefix <installationdir>
```

- Build the software

```
make
```

- Install the software

```
sudo make install
```

- Verify the installation

```
<installationdir>/bin/tm-vmm --list-clients
```

2.3 Setup

- Create the directory $HOME/.testingmachine

```
mkdir $HOME/.testingmachine
```

- Create tm-vmm.conf in .testingmachine, typically with your favorite editor (emacs?)

```
emacs ~/.testingmachine/tm-vmm.conf
```

In this file you can configure settings you want to use as default in your clients. It is perfectly possible to override these settings in your individual client configurations.

For a list of variables, see section Configuration syntax below.

2.4 Creating a client

First of all you need to decide what machine you want to use with your client. In the example below we will assume it is called Debian6.0

- Use the command line option:
  `--create-client-conf`

Example usage of the option.

```
`tm-vmm --create-client-conf  Debian-6.0`
```

- Copy the public ssh key to the machine, e.g

```
ssh-copy-id 192.168.1.2
```

3 Communication

3.1 Reporting bugs

Try to be as precise as possible when reporting bugs. The more information we get the bigger chance we have of fixing the problem.

Use the mailing list below to report bugs.

3.2 Getting involved

How to join: clone the repo, try it out – join the mailing list :)

For more information read the developer guidelines.

3.3 Mailing list

We have one mailing list for the project: `community@testingmachine.eu`

Join this list here: `https://lists.testingmachine.eu/cgi-bin/mailman/listinfo/community`

If you send emails to this list as a non subscriber chances are it will get list.

If you want to report a bug: * use a github account and add an issue * subscribe to the mailing and send the report to the list

3.4 Home page

https://testingmachine.eu/

Source code is located here: `https://github.com/tis-innovation-park/vmm`

3.5 Social media

3.5.1 Twitter

4 Using Virtual Machine Manager

Regardless of the underlying virtual machine you manage the machine in one way. We provide the most basic uses of tm-vmm here and encourage you to read the complete list of command line options in the last section of the manual.

4.1 Starting

4.1.1 Starting with guest operating system visible

```
tm-vmm --start-client <CLIENTNAME>
```

Example:

```
tm-vmm --start-client Ubuntu-10
```

4.1.2 Starting without showing guest operating system (headless)

```
tm-vmm --start-client-headless <CLIENTNAME>
```

Example:

```
tm-vmm --start-client-headless Ubuntu-10
```

4.2 Checking status

```
tm-vmm --check-client-status <CLIENTNAME>
```

Example:

```
tm-vmm --check-client-status Ubuntu-10
```

4.3 Stopping

`tm-vmm --stop-client <CLIENTNAME>`

Example:

`tm-vmm --stop-client Ubuntu-10`