

VirtLab | Secure Access Gateway

Projecto final de curso

Ricardo Malta

Universidade Lusófona de Humanidades de Tecnologias

Orientador: Professor José Rogado

Co-orientador: Professor José Faisca

Índice de conteúdos

| | |
|--|-----------|
| RESUMO | 4 |
| ABSTRACT | 5 |
| 1. INTRODUÇÃO | 6 |
| 1.1. ENQUADRAMENTO..... | 7 |
| 1.2. ABORDAGEM TECNOLÓGICA | 8 |
| 1.3. O VIRTLAB..... | 9 |
| 2. SECURE ACCESS GATEWAY | 10 |
| 2.1. VISÃO GERAL | 10 |
| 2.3. ACESSO AO REPOSITÓRIO DE MÁQUINAS VIRTUAIS..... | 14 |
| 3. ACESSO WEB À PLATAFORMA..... | 16 |
| 3.1. VISÃO GERAL DO ACESSO WEB | 16 |
| 3.2. ACESSO AO SAG E APRESENTAÇÃO DAS MÁQUINAS VIRTUAIS (VMs)..... | 16 |
| 4. CONSIDERAÇÕES FINAIS E PERSPECTIVAS | 18 |
| REFERENCIAS: | 19 |
| GLOSSÁRIO..... | 21 |
| ANEXOS | 23 |

Índice de ilustrações

| | |
|--|-----------|
| FIGURA 1 - CONFIGURAÇÃO DO LABORATÓRIO ACTUAL | 7 |
| FIGURA 2 - INTEGRAÇÃO DO VIRTLAB COM O CAMPUS UNIVERSITÁRIO | 9 |
| FIGURA 3 - ARQUITECTURA DO LIBVIRT NO VIRTLAB..... | 11 |
| FIGURA 4 - ARQUITECTURA DO SECURE ACCESS GATEWAY (SAG)..... | 11 |
| FIGURA 5 - FUNCIONAMENTO DA GESTÃO DE SESSÕES: INÍCIO DE SESSÃO..... | 12 |
| FIGURA 6 - PEDIDO DA LISTA DAS MÁQUINAS VIRTUAIS ACESSÍVEIS AO UTILIZADOR | 13 |
| FIGURA 7 - DIAGRAMA DE EXECUÇÃO DO PROCESSO INICIAR UMA MÁQUINA VIRTUAL COM O SAG | 14 |
| FIGURA 8 - DIAGRAMA DE EXECUÇÃO DO PROCESSO PARAR UMA MÁQUINA VIRTUAL..... | 15 |
| FIGURA 9 - DIAGRAMA DE EXECUÇÃO DO PROCESSO COLOCAR E TIRAR DA PAUSA UMA MÁQUINA VIRTUAL..... | 15 |
| FIGURA 10 - PÁGINA INICIAL DO ACESSO WEB AO VIRTLAB | 16 |
| FIGURA 11 - CONSOLA DE ACESSO ÀS MÁQUINAS VIRTUAIS..... | 17 |

Resumo

Este Projecto Final de Curso insere-se no projecto “Virtlab - Laboratórios Virtuais” [1] na Universidade Lusófona de Humanidades e Tecnologia (ULHT) que utiliza tecnologia de código fonte aberto e/ou licenciamento livre. Entre outros aspectos, este projecto contempla a criação de um *Broker* de ambientes virtuais, onde os alunos têm acesso a um ou vários ambientes virtuais (máquinas, redes e armazenamento virtuais) após autenticação e validação do respectivo perfil académico. Na terminologia do VirtLab, este *Broker* tem o nome de Secure Access Gateway [SAG].

O SAG está desenvolvido em Python [2], comunica com um repositório de autenticação OpenLDAP [3] e é configurado a partir de uma base de dados relacional MySQL [4], e comunica com um repositório de máquinas virtuais através do LibVirt [5], permitindo a criação de ambientes virtuais heterogéneos com abstracção do *Hypervisor* [m] de virtualização [n]. A comunicação entre o SAG e o ambiente de virtualização é feita através de um layer de serviços SOAP [WebServices].

Este projecto contempla ainda um interface visual Web com o SAG, desenvolvido recorrendo a PHP, HTML e AJAX.

Palavras chave

Laboratórios Virtuais; Autenticação; Controlo de Acessos; Libvirt;

Abstract

This project is part of the virtual laboratories Virtlab project [1] taking place at the University Lusófona de Humanidades e Tecnologia (ULHT), using open source code and /or free content technology. Among other concepts this project introduces the notion of a *Broker* for virtual environments, where the students, once authenticated, have access to one or various virtual environments (virtual machines, nets and storage) according to their academic enrolment profiles. In the Virtlab terminology, this *Broker* is denominated Secure Access Gateway [SAG].

The SAG is developed in Python [2], and uses OpenLDAP [3] as an authentication repository, retrieves its configuration from a MySQL [4] relational database, and gives access to a repository of virtual machines using the LibVirt [5] technology. This approach allows the creation of heterogeneous virtual environments which are independent of the virtualization Hypervisor [m]. The communication between SAG and the virtualization environment is performed through a SOAP service layer (WebServices).

This project also includes a Web visual interface to the SAG, developed using PHP, HTML and AJAX.

Keywords

Virtual laboratories; Authentication; Access Control; Libvirt;

1. Introdução

Os laboratórios de informática são essenciais para o desenvolvimento de disciplinas científicas no âmbito das tecnologias de informação. Têm que estar preparados para permitir várias configurações de rede, e a instalação de diversos sistemas operativos, aplicações, etc. A manutenção destas configurações nos laboratórios não é fácil e piora com o aumento do número de disciplinas que os utilizam. Assistimos neste momento a uma má utilização dos recursos disponíveis nos laboratórios, visto existir hoje em dia uma grande capacidade de processamento e armazenamento distribuído por todos os laboratórios da universidade, mas que por motivos ligados à falta de manutenção, dificuldades de configuração e diversidade de utilização, não consegue responder às necessidades de ensino.

A solução passa pela virtualização dos laboratórios de computação, com uma plataforma ampla, configurável e federada.

1.1. Enquadramento

Este projecto está a ser desenvolvido no âmbito da actividade iniciada com a publicação do documento “Virtlab: Virtual Laboratories for e-Learning Federated Environments” [1], pelo Professor José Rogado. O VirtLab introduz o conceito de uma plataforma e-Learning virtual de redes, sistemas operativos e aplicações com acesso federado, acessível dentro do Campus universitário da Universidade Lusófona de Humanidades e Tecnologia (ULHT), bem como através de uma possível federação com toda a comunidade académica.

O tema do presente projecto consiste no desenvolvimento do Secure Access Gateway [1] [Capítulo 4], que é responsável por interceptar as credenciais do utilizador e verificar se, de acordo com o seu perfil, tem autorização de acesso e a qual/quais recursos do VirtLab pode aceder (Broker).

Neste momento os laboratórios de Redes de Computação da ULHT, estão fisicamente configurados e a sua modificação e personalização é limitada. Qualquer experiência feita durante as aulas, terá que ser com a configuração existente (Figura 1). Com a existência de máquinas e redes virtuais tudo se poderá tornar mais fácil, sendo que a solução tem que cumprir os seguintes requisitos:

- Permitir a fácil configuração de ambientes virtuais (máquinas e redes);
- Permitir a virtualização de servidores e desktops;
- Permitir o acesso federado de todos os alunos da comunidade universitária;
- Recorrer a tecnologias de código aberto ou licenciamento livre.

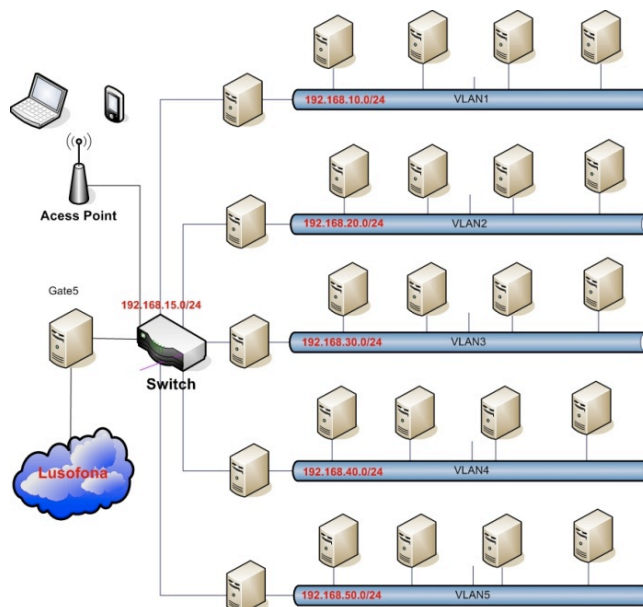


Figura. 1 NetLab. Configuração do laboratório actual.

1.2. Abordagem tecnológica

Actualmente nas tecnologias de informação assistimos a uma emergência da tecnologia de virtualização. Esta tornou-se um sinónimo de abstracção, onde se aplicam paradigmas de computação paralela [a], heterogénea e soluções de *Green Computing* [b], abrangendo várias áreas/tipos, entre elas a virtualização de desktops e servidores.

A virtualização de desktop tem como finalidade a simples emulação de um sistema de workstation, podendo assim executar uma aplicação que necessite de um sistema específico, ou diferente do nativo em que o utilizador normalmente trabalha. Existem várias ferramentas de virtualização para este fim, tais como *VMware Workstation* [6], *Microsoft Virtual PC* [7], *Sun VirtualBOX* [8], etc.

Nos servidores a virtualização torna-se um pouco mais complexa e com outros pontos-chave. Os principais objectivos passam por aproveitar a potência total que os servidores multi-core [c] de hoje em dia disponibilizam, reduzindo o número de máquinas servidor e assim reduzir os custos de manutenção de um datacenter e principalmente poder executar lado a lado sistemas operativos distintos, para propósitos distintos, nas mesmas máquinas físicas.

Existem várias técnicas de virtualização aplicadas nos servidores, das quais se destacam a *Full Virtualization* [d] e a *Paravirtualization* [e].

No processo de *Full virtualization* todo o hardware e firmware são emulados à semelhança do sistema pretendido. Esta técnica permite emular qualquer sistema ou arquitectura, sendo que quase tudo é possível, no entanto é lento, consequência de todo o sistema correr sobre um hardware totalmente emulado.

Pelo contrário, com a *Paravirtualization* o sistema virtualizado (guest) é criado à semelhança do sistema hospedeiro (host). Desta forma o sistema emulado é mais rápido na sua execução, devido ao hypervisor que recebe os hyper-calls[f] do sistema guest e traduz para os system-calls [g] do sistema hospedeiro. Até há pouco tempo esta técnica limitava o tipo de sistema virtualizável, já que o sistema guest teria de ser do mesmo tipo do host. Todavia com o aparecimento de tecnologias de Virtualização Intel VT [9] e AMD-V [10] é possível *Paravirtualization*, independentemente do sistema host instalado. Estas tecnologias permitem que o sistema guest possa executar os seus system-calls directamente no CPU físico, evitando recorrer a hyper-calls; tal veio trazer um acréscimo considerável em termos de performance. Os sistemas que recorrem à *Paravirtualization* são principalmente utilizados para virtualizar servidores, e entre eles existem alguns projectos abertos, como o XEN [11] e o KVM [12], os quais são actualmente bastante relevantes nesta área da computação. A evolução da virtualização vai

seguir naturalmente o caminho da *paravirtualization*, sendo que o futuro passa por aumentar a capacidade do acesso por parte das máquinas virtuais para acederem directamente não só a execuções no CPU mas também ao controlo do Input/Output, Gestão Memória e outros recursos de hardware.

1.3. O VirtLab

A virtualização é um paradigma da engenharia informática contemporânea, acelerado pela emergência do *Cloud Computing* [h], necessidades ecológicas e optimização da utilização dos recursos, que os processadores multi-core hoje nos permitem.

Os objectivos do presente trabalho surgem na sequência do já referido trabalho de Rogado [1], que dá a base para as várias fases de implementação de laboratórios virtuais no campus universitário. Este projecto explora e representa a integração de um ambiente de virtualização com o campus da Universidade (Figura. 2), bem como a apresentação dos ambientes virtuais aos alunos.



Figura. 2 Integração do VirtLab com o campus universitário.

O VirtLab é um projecto académico com o propósito de explorar a tecnologia no seu *State of the art* no que diz respeito à virtualização e computação distribuída. Com estes requisitos em mente, todas as ferramentas e tecnologias utilizadas são de licenciamento livre e/ou código fonte aberto.

As ferramentas utilizadas são essencialmente oriundas de projectos da RedHat [13], como o Libvirt [5], que é uma camada de software que abstrai o(s) hypervisor (s) [m], oferecendo um interface programático com suporte para várias linguagens de programação, entre elas o ANSI C e Python. São também utilizadas neste projecto ferramentas de apoio à configuração (opcionais) do Libvirt como o Virt-Manager [14] e o Virt-install [14]. Todas estas aplicações são provenientes de projectos de código fonte aberto da RedHat.

2. Secure Access Gateway

2.1. Visão geral

O Secure Access Gateway (SAG) é a aplicação responsável pela gestão de identidade e acesso aos ambientes virtuais (Broker) do VirtLab, e implementa as seguintes funções:

- Interpretar e validar as credenciais do utilizador
- Verificar o perfil de utilizador e realizar o controlo de acesso aos recursos acedidos;
- Gerir a sessão do utilizador;
- Abstrair a complexa configuração dos ambientes virtuais, configurados com base nos vários perfis de utilizadores possíveis;
- Comunicar com o repositório de máquinas virtuais (VMs), através do Libvirt [5] (Figura 3).

Todo o SAG foi desenvolvido na linguagem de programação Python [2], e o interface com a aplicação é feito através de uma camada de serviços Web (WebServices) criado com recurso à *soaplib* [15] do Python, responsável pela geração do WSDL e criação do interface SOAP com a aplicação. O WSDL é disponibilizado via HTTP pelos componentes Web de *WSGI* [16] incluído na *Standard Lib* do Python desde a sua versão 2.5. A escolha da plataforma de desenvolvimento recaiu sobre o Python, por ser uma linguagem de Scripting [i] de alto nível, com uma forte componente orientada a objectos [j] e com o melhor suporte para a API da Libvirt.

O Libvirt possibilita não só executar funções de controlo das máquinas virtuais tais como Iniciar, Parar, Pausa, mas também gerir a configuração de redes virtuais e armazenamento, acesso remoto com encriptação TLS [k] e certificados x509 [l], autenticação Kerberos e SASL. Fornece APIs para várias linguagens, nomeadamente ANSI C, Python, Perl, Java, que acompanham o seu desenvolvimento. A Libvirt é um projecto da RedHat com um acentuado incremento do desenvolvimento no decorrer do último ano. Neste momento permite a comunicação com vários *Hypervisors* [m] de virtualização, tais como o XEN, KVM e o Sun VirtualBox, fornecendo também já algum suporte para o VMware ESX. A opção da utilização do Libvirt como meio de gestão do repositório de máquinas virtuais, em vez de programar directamente numa API específica de

um Hypervisor (ex. XEN, VMware ou Sun VirtualBOX) deve-se à capacidade de abstracção do Hypervisor utilizado. Com o Libvirt é assim possível gerir um ambiente virtual heterogéneo, onde as *Virtual Machines* podem estar em qualquer um dos Hypervisors suportados e serem geridas da utilizando funções e sintaxe idênticas, uma vez que a gestão de rede e armazenamento é feita no próprio Libvirt.

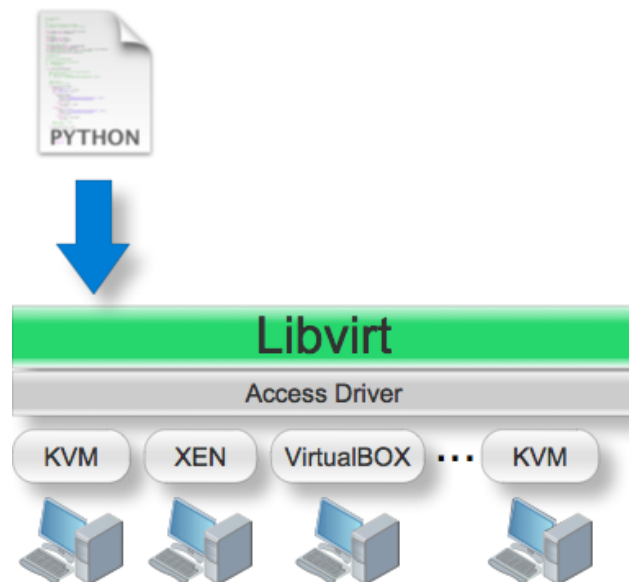


Figura. 3 Arquitectura do Libvirt.

O SAG comunica com um directório de utilizadores e grupos baseado em OpenLDAP [3], semelhante ao directório Microsoft Active Directory [17] que está implementado na ULHT, visto não ser possível comunicar com este último nesta fase do projecto. Todas as configurações de acesso aos recursos do VirtLab, estão armazenados numa base de dados MySQL [4]. Na Figura 4 é descrita a arquitectura da aplicação.

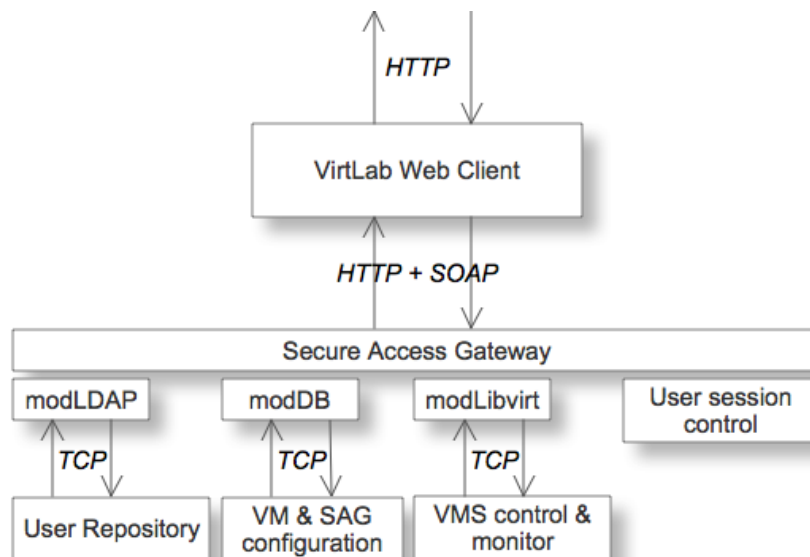


Figura. 4 Arquitectura do Secure Access Gateway (SAG).

A aplicação tem implementado um sistema de *logging* onde todas as ligações a servidores externos, criação de chaves de sessão e erros são enviados para a consola onde se está a executar. Isto permite um fácil diagnóstico de erro ou mau funcionamento.

2.2. Gestão de identidade

O SAG como anteriormente já foi referido, faz a gestão de sessões temporárias. Todos os utilizadores devem ter as suas credenciais no directório da Universidade, que são geralmente utilizadas para as várias ferramentas informáticas disponíveis no campus. No caso do SAG é utilizado um repositório baseado em OpenLDAP [3] que simula o da ULHT e nele estão configurados os utilizadores e grupos, bem como os seus dados pessoais, contactos, entre outros.

O funcionamento da gestão de sessões é representado no diagrama de sequência da Figura 5.

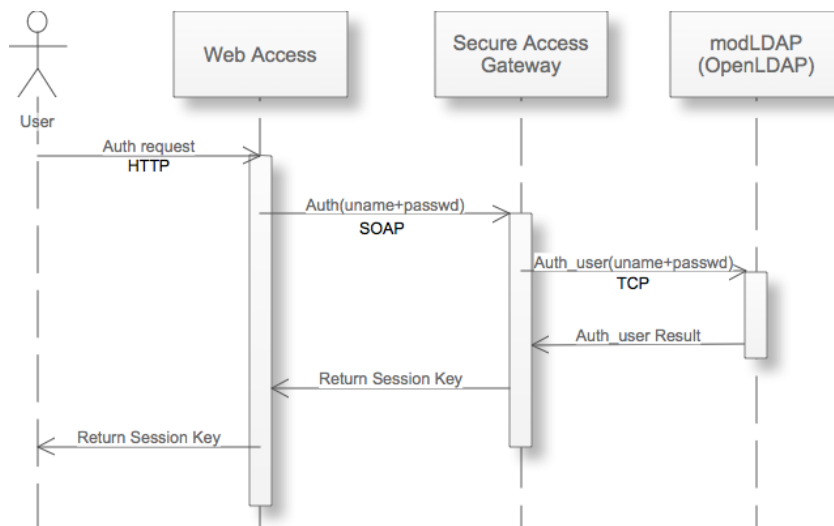


Figura. 5 Funcionamento da Gestão de Sessões: Início de sessão.

O utilizador, depois de autenticado com sucesso, obtém uma chave de sessão, devendo a partir desse momento fazer todos os pedidos ao SAG incluindo essa chave de sessão. No caso da plataforma desenvolvida, este requisito é realizado em nome do utilizador na aplicação Web de acesso ao repositório de máquinas virtuais [ver descrição abaixo].

Um típico pedido da lista das máquinas virtuais (VMs), partindo do princípio que o utilizador já está autenticado, processa-se segundo o diagrama da Figura 6.

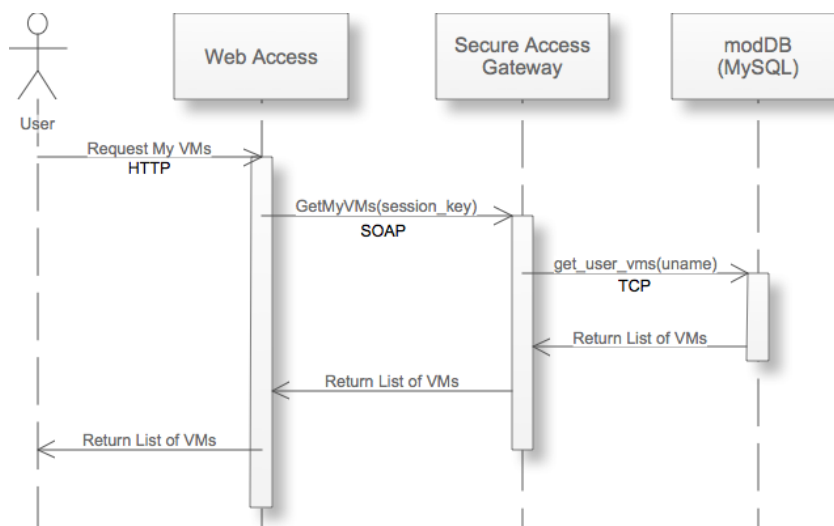


Figura. 6 Pedido da lista das máquinas virtuais acessíveis ao utilizador.

2.3. Acesso ao repositório de máquinas virtuais

O Libvirt utiliza o UUID para identificar as máquinas virtuais que estão configuradas. O UUID é um identificador único global das máquinas virtuais no Libvirt, com um formato compatível com o RFC 4122. Como se trata de um código não legível, as máquinas virtuais são apresentadas ao utilizador com o seu nome.

O SAG permite executar as seguintes funções sobre as máquinas virtuais: Iniciar, parar, pausa, devolver o estado. Nos diagramas de sequência seguintes (Figuras 7, 8 e 9) é demonstrada a forma como são executados os pedidos do utilizador às máquinas virtuais.

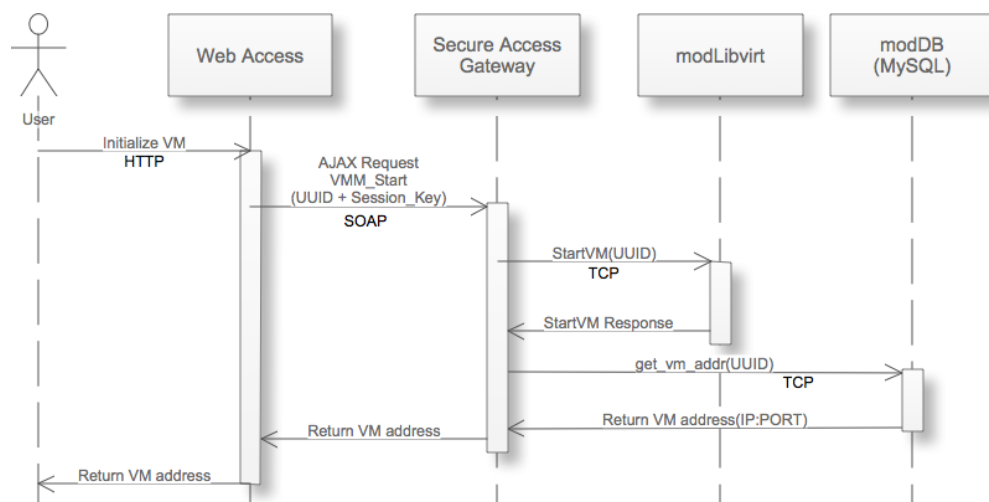


Figura. 7 Diagrama de execução do processo Iniciar uma máquina virtual com o SAG.

O acesso às máquinas virtuais é feito através de uma aplicação Web, que pode ser executado em qualquer explorador de Internet compatível com suporte para *Javascript* e *Java runtime* instalado no sistema operativo. O método para iniciar uma máquina virtual no SAG tem duas funções (Figura 7). Antes de qualquer acção verifica se a chave de sessão do utilizador é válida e de seguida tenta-se iniciar a máquina virtual. Caso esta seja iniciada com sucesso, é gerado o endereço de acesso para consola VNC [o], com recurso às configurações definidas na base de dados da aplicação, sendo este devolvido ao browser do utilizador. O endereço tem o formato IP:PORT, e pode ser utilizado com qualquer cliente VNC, mas que no caso do cliente Web está embebido no browser, através de uma *Java Applet*.

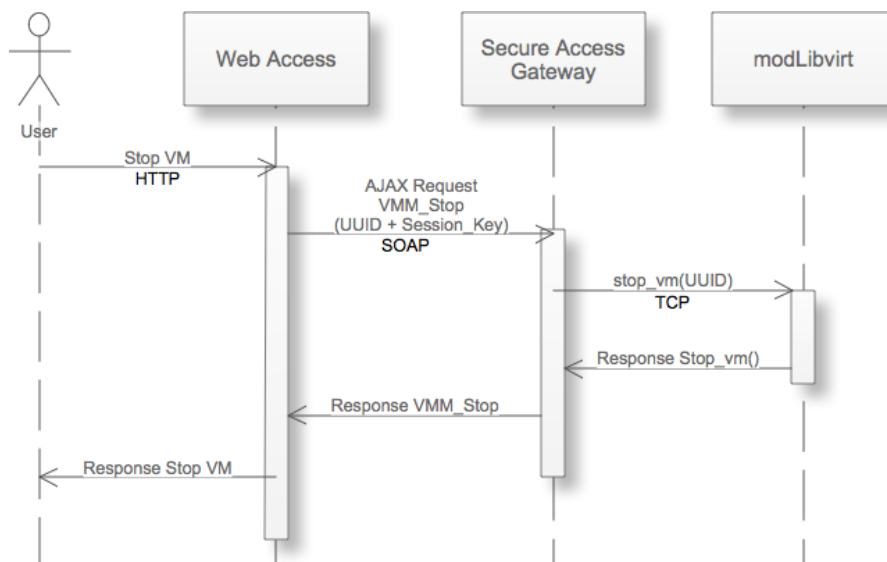


Figura. 8 Diagrama de execução do processo Parar uma máquina virtual.

Para parar uma máquina virtual é apenas necessário que o seu estado seja diferente de parado, e que a sessão do utilizador seja válida (Figura 8). O cliente Web envia dois parâmetros, a chave de sessão e o UUID da máquina virtual, caso a máquina virtual esteja em modo activo é parada através do comando Libvirt *force shutdown*.

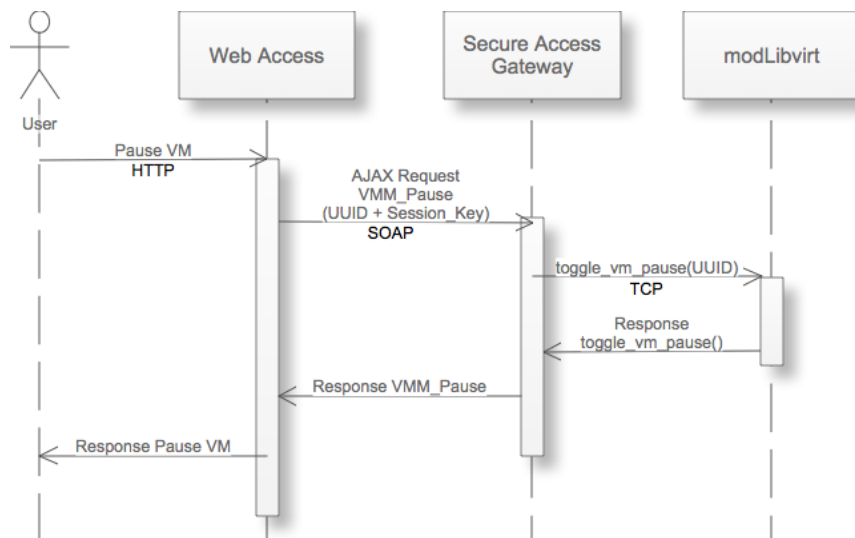


Figura. 9 Diagrama de execução do processo Colocar e tirar da pausa uma máquina virtual.

O método que efectua a pausa de uma máquina virtual é idêntico ao utilizado para a parar. Quando é feito o pedido de pausa, o método verifica se está iniciada e coloca-a em pausa. Se for invocado e a máquina já estiver em pausa, volta a colocar em execução. A sequência de execução do método está ilustrada na Figura 9.

3. Acesso WEB à plataforma

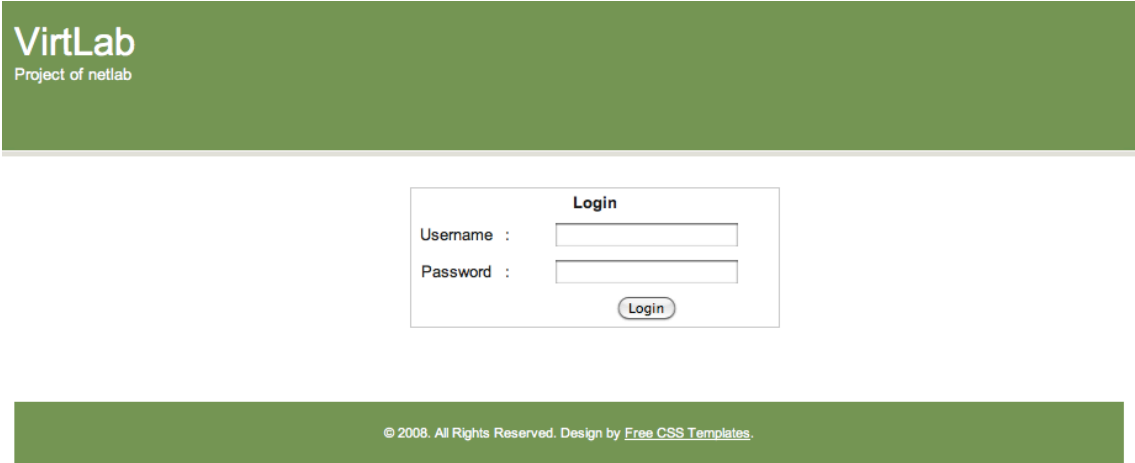
3.1. Visão geral do acesso WEB

A comunicação com o SAG é feita através de uma camada de serviços SOAP [p] com todos os métodos necessários para gerir a sessão dos utilizadores e acesso às máquinas virtuais. O interface Web foi construído com o intuito de interagir com o WebService [p] do SAG de uma forma prática e intuitiva.

Toda a aplicação Web está desenvolvida em PHP [18] e Javascript [19]. A origem do layout provém da comunidade freecsstemplates [20]. O acesso ao WebService do SAG é feito em PHP e todos os pedidos são feitos com apoio da Framework de Javascript Prototype [21], permitindo implementar AJAX [q] de forma rápida e eficiente.

3.2. Acesso ao SAG e apresentação das máquinas virtuais (VMs)

A primeira página do cliente Web é a de autenticação do utilizador. A descrição do processo de autenticação está realizada na Figura 5 e tem o aspecto visual representado na Figura 10.



VirtLab
Project of netlab

Login

Username :

Password :

© 2008. All Rights Reserved. Design by [Free CSS Templates](#).

Figura. 10 Página inicial do acesso Web ao VirtLab.

Após uma autenticação realizada com sucesso, o utilizador tem acesso a uma chave de sessão, que lhe permite fazer todos os pedidos ao SAG. As operações a que o utilizador tem acesso sobre as máquinas virtuais são: iniciar, parar e colocar em pausa. Estes processos estão descritos no ponto 2.3. O utilizador pode seguidamente aceder directamente ao ambiente de execução gráfico da máquina virtual através de uma Applet Java VNC [TightVNC] [22] embutida na página HTML. A interacção com as máquinas virtuais apresenta-se da seguinte forma [Figura 11]:

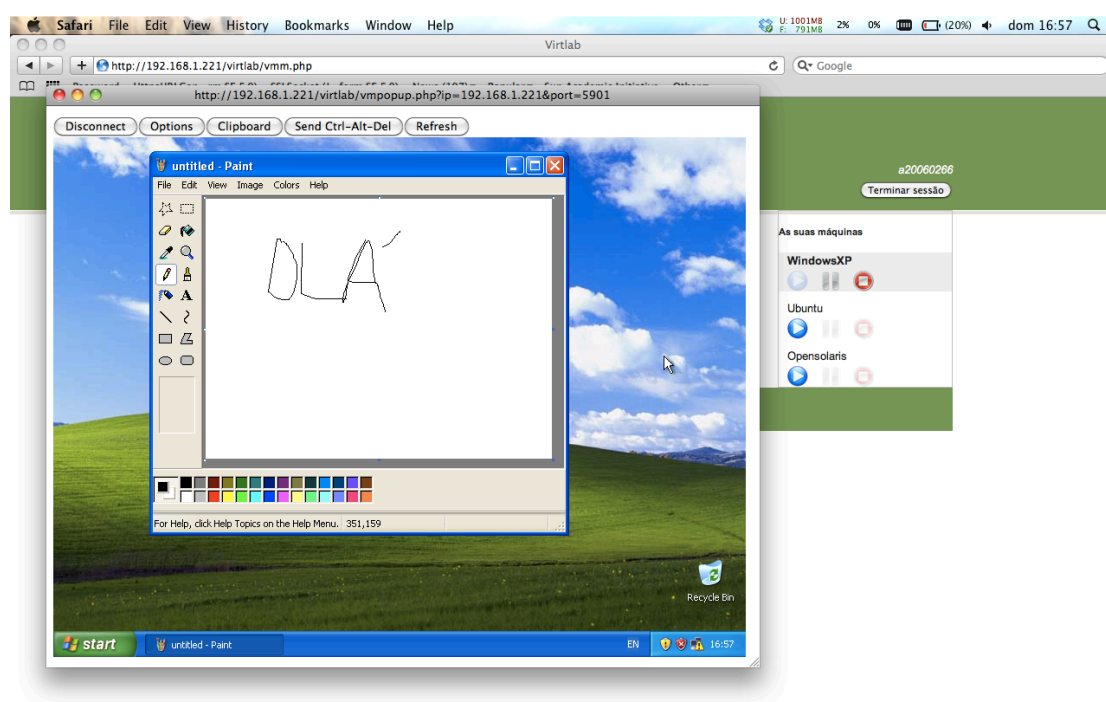


Figura. 11 Consola de acesso às máquinas virtuais.

O interface e a interacção com a consola das máquinas virtuais, está totalmente integrado com recurso a AJAX. Sendo um dos pontos-chave da WEB2.0 [r], o AJAX permite executar pedidos ao servidor Web de forma assíncrona, sem ter que fazer *reload* de todo o conteúdo HTML no browser, recorrendo a Javascript e à Framework prototype [20].

4. Considerações finais e perspectivas

A implementação do SAG do projecto VirtLab é resultado de um semestre de investigação na área de virtualização de servidores, recorrendo a software livre. Este trabalho permitiu-me adquirir competências numa nova linguagem de programação, o Python, que para além de ser de fácil aprendizagem, mostrou ser também muito eficiente e de rápido desenvolvimento. Foi relativamente fácil colocar todos os módulos da aplicação a funcionar, recorrendo apenas a Python e a algumas *libs* extra.

No âmbito e de acordo com as especificações/requisitos do projecto VirtLab, ficou demonstrado o funcionamento do acesso às máquinas virtuais com base no perfil de cada utilizador e o acesso remoto interactivo com as máquinas pelo cliente Web.

Para que o VirtLab possa funcionar de acordo com o previsto no projecto Virtlab citado, falta dimensionar os ambientes virtuais a serem criados, de acordo com as características do servidor (neste momento um HP DL380 com um CPU Xeon 5400, 16 GB RAM e 1 TB de armazenamento). Falta igualmente implementar a autenticação e o controlo de acesso com base em tecnologia federativa Shibboleth [32] em vez do repositório OpenLDAP e do controlo de sessões implementado neste momento.

Contudo, este projecto representa um importante primeiro passo na implementação do interface de acesso ao ambiente virtualizado, e da lógica do SAG no VirtLab.

Referências:

- [1] Rogado, J.Q. 2009. VirtLab: Virtual Laboratories in Federated Environments. In Méndez-Vilas, A., Solano Martín, A., Mesa González, J.A. and Mesa González, J.. Research, Reflections and Innovations in Integrating ICT in Education, Vol. 1, 593-597. Formatex, Spain
- [2] Python Programming Language. <http://www.python.org>
- [3] OpenLDAP - community developed LDAP software. <http://www.openldap.org>
- [4] Sun Microsystems, Inc. MySQL Server Community Edition.
<http://dev.mysql.com>
- [5] Libvirt The virtualization API. <http://www.libvirt.org>
- [6] VMware Workstation. Copyright © 2009 VMware, Inc. All rights reserved.
<http://www.vmware.com/products/workstation>
- [7] Windows Virtual PC. © 2009 Microsoft.
<http://www.microsoft.com/windows/virtual-pc>
- [8] Sun VirtualBox. © 2009 Sun Microsystems, Inc. <http://www.virtualbox.org>
- [9] Intel® VT. ©Intel Corporation.
<http://www.intel.com/technology/virtualization/technology.htm?iid=SEARCH>
- [10] AMD-V™ ©2009 Advanced Micro Devices, Inc.
<http://www.amd.com/uk/products/technologies/virtualization/Pages/virtualization.aspx>
- [11] The Xen® hypervisor. Citrix Systems, Inc. <http://www.xen.org>
- [12] KVM - Kernel Based Virtual Machine. <http://www.linux-kvm.org>
- [13] Redhat® Copyright © 2009 Red Hat, Inc. <http://www.redhat.com>
- [14] Virtual Machine Manager. RedHatET. <http://virt-manager.et.redhat.com>
- [15] soaplib 0.8.1 <http://wiki.github.com/jkp/soaplib>
- [16] WSGI <http://wsgi.org/wsgi>
- [17] Microsoft Active Directory. © 2009 Microsoft.
<http://www.microsoft.com/windowsserver2008/en/us/active-directory.aspx>

[18] PHP Scripting Language. Copyright © 2001-2009 The PHP Group.
<http://www.php.net/>

[19] Javascript Programming Language. <http://www.ecma-international.org/publications/standards/Ecma-262.htm>

[20] Prototype JavaScript Framework. © 2006-2007 Prototype Core Team.
<http://www.prototypejs.org/>

[21] Free CSS Templates. Copyright © 2008-2009 Free CSS Templates.
<http://www.freecsstemplates.com/>

[22] TightVNC. <http://www.tightvnc.com/>

[32] Shibboleth <http://shibboleth.internet2.edu/>

Glossário

[a] Computação paralela - Técnica de computação onde vários cálculos ou execuções são feitos simultaneamente, para o mesmo fim ou não. Associado à computação paralela está quase sempre também o processamento e computação distribuída, multi-thread, multi-processador e multi-core.

[b] Green Computing - Conceito de sustentabilidade energética aplicada à computação. Criação de tecnologia de computação com menor impacto ambiental, principalmente através de hardware energeticamente mais eficiente.

[c] Multi-Core - Técnica de processamento composta por dois ou mais cores num único chip de silício.

[d] Full Virtualization - Técnica de virtualização onde todo o hardware do sistema é virtualizado por software ou com suporte de hardware.

[e] Para Virtualization - Técnica de virtualização que altera o sistema guest de forma a integrar com o sistema host. Apenas algumas funções de sistema são realmente virtualizadas.

[f] Hyper calls - Chamadas ao sistema hypervisor que por sua vez interagem com os system calls do sistema operativo base.

[g] System calls - Chamadas ao Kernel do sistema operativo, que por sua vez interage com o hardware do computador.

[h] Cloud Computing - Paradigma da computação que abstrai uma grid de computadores, de forma escalável e muitas vezes virtualizada, sendo disponibilizada com uma arquitectura orientada ao serviço (SOA).

[i] Scripting - Linguagens script são executadas dentro de um interpretador em tempo de execução ao contrário das compiladas.

[j] Programação orientada a objectos - Paradigma de programação onde os dados são trabalhados como se de entidades se tratassem. As entidades têm o nome de classes e podem ser instanciadas tantas vezes quantas as necessárias. As classes têm várias características como propriedades, métodos, suporte para polimorfismo, herança de características entre classes, entre outros.

[k] TLS - Protocolo de criptografia.

[l] Certificados X.509 - Padrão para validação de identidade nas infra-estruturas de chave pública (PKI).

[m] Hypervisor – Plataforma de software e hardware que permite virtualizar um ou mais sistemas operativos em paralelo.

[n] Virtualização – Simulação de Hardware de um computador de forma a executar um sistema operativo virtualmente.

[o] VNC (Virtual Network Computing) – Protocolo de acesso remoto ao ambiente gráfico de um determinado sistema operativo.

[p] WebService – Interface de comunicação entre aplicações via Web, tipicamente com recurso à definição do interface em XML e troca de informação via SOAP.

[q] AJAX - Técnica de desenvolvimento Web 2.0, que permite executar pedidos ao servidor Web de forma assíncrona, sem ter que fazer *reload* de todo o conteúdo HTML no browser, recorrendo a Javascript e ao protocolo XMLHttpRequest

[r] WEB 2.0 – Novo conceito de World Wide Web como uma plataforma de serviços.

Anexos

Implementação

As instruções estão descritas com base num sistema semelhante ao de desenvolvimento. Este ambiente consistiu numa distribuição de GNU/Linux Debian 5 e todos os softwares adicionais foram instalados a partir dos repositórios oficiais da respectiva distribuição.

No final da instalação, estão instruções de como criar uma ambiente virtual; uma simulação do que será realmente o VirtLab a funcionar.

Requisitos e instalação do SAG e acesso WEB

Antes de avançar com a instalação da aplicação propriamente dita, será necessário instalar alguns pré-requisitos, bem como o servidor OpenLDAP e MySQL.

Todos os comandos indicados a seguir, são antecipados do uma cardinal (#) e terão de ser executados com **privilégios de administrador** da máquina:

Actualizar o repositório de pacotes do Debian5:

```
# apt-get update
```

Instalar os pré-requisitos:

```
# apt-get install apache2 php5 python-setuptools python-dev  
python-lxml gcc g++ python-gtk-vnc libglade2-0 libglade2-dev  
libgtk-vnc-1.0-0 libgtk-vnc-1.0-dev gnutls-bin libgnutls-dev  
libxen-dev libxenstore3.0 libxml2-dev libgnutls-dev libsasl2-  
dev intltool automake autoconf libselinux1 libselinux1-dev  
libdevmapper1.02.1 libdevmapper-dev
```

```
# easy_install pytz  
# easy_install soaplib
```

Instalar o servidor de repositório de utilizadores OpenLDAP e a configuração inicial para o VirtLab; será pedida a password para o administrador [admin]:

```
# apt-get install slapd ldap-utils python-ldap
```

No próximo comando terá que configurar o domínio do repositório de utilizadores [OpenLDAP]. O nome que deve introduzir, para o ambiente de testes descrito mais à frente é virtlab.com e a organização virtlab, base de dados HDB; não deve activar LDAPv2 e deverá aceitar todas as outras opções como opção predefinida:

```
# sudo dpkg-reconfigure slapd
```

Instalar o sistema de gestão de base de dados MySQL. Irá ser pedida a password para o administrador [root].

```
# apt-get install mysql-server-5.0 python-mysqldb-dbg
```

Instalar e configurar o Libvirt. Este é o componente crítico do VirtLab, já que é com o Libvirt que serão geridas todas as máquinas virtuais.

Para instalar o Libvirt com o comando abaixo, a máquina host que está a ser utilizada deverá suportar tecnologia de virtualização no CPU (Intel VT-x ou AMD-v):

```
# apt-get install libvirt0 libvirt-bin libvirt-dev python-libvirt virtinst kvm virt-manager
```

Para que as máquinas virtuais criadas tenham acesso à rede local, terá que ser criado uma bridge network [br0]. Partindo do princípio que a rede onde nos encontramos é a 192.168.1.0 e o gateway 192.168.1.254, os comando a executar serão os seguintes:

```
# brctl addbr br0
# ifconfig eth0 0.0.0.0
# brctl addif br0 eth0
# ifconfig br0 192.168.1.222 netmask 255.255.255.0 up
# route add -net 192.168.1.0 netmask 255.255.255.0 br0
# route add default gw 192.168.1.1 br0
```

Para configurar o virtlab terão que ser configurados os servidores (OpenLDAP, MySql e Libvirt) no ficheiro *sag.conf*.

A partir deste momento, o interface de rede com o computador é o br0, bem como para todas as máquinas virtuais que serão configuradas mais à frente. É também já possível executar o SAG com sucesso com este sistema, no entanto não há conteúdo para que este funcione no seu propósito. Para tal, a seguir estão

as instruções para criar um ambiente de teste ou simulação. A ordem pela qual são configurados os servidores é fundamental que seja a que se segue.

Implementar o SAG:

Para começar terá que ser instalado um conjunto de máquinas virtuais no servidor, nesta fase é útil ter instalado o virt-manager. No entanto pode ser tudo feito sem recorrer a este utilitário:

Instalar uma máquina virtual Linux KVM, utilizando como exemplo o DSL Linux, com as seguintes características:

1 CPU, 512 Ram, 6GB disco, rede Bridge e instalar a partir do CD-ROM:

```
# virt-install --connect qemu:///system -n DSLinux -r 64 --  
vcpus=2 -f ~/DSL.qcow2 -s 6 -c "/home/dsl-4.4.10.iso" -vnc --  
vncport=5901 --noautoconsole --os-type linux --accelerate --  
network=bridge:br0 --hvm
```

Instalar uma máquina virtual WindowsXP KVM com as seguintes características:

1 CPU, 512 Ram, 8GB disco, rede Bridge e instalar a partir do CD-ROM:

```
# virt-install -connect qemu:///system -n WindowsXP --ram 512  
--disk path=/tmp/winxp.img,size=8 --vcpus=1 --accelerate --  
network=bridge:br0 --vnc --vncport=5902 --noautoconsole --os-  
variant winxp --cdrom /dev/sr0 -hvm
```

Instalar uma máquina virtual Opensolaris KVM, utilizando como exemplo o DSL Linux, com as seguintes características:

1 CPU, 512 Ram, 10GB disco, rede Bridge e instalar a partir do CD-ROM:

```
# virt-install --connect qemu:///system -n Opensolaris -r 512  
--vcpus=2 -f ~/DSL.qcow2 -s 10 -c "/home/osol-0906-x86.iso" -  
vnc --vncport=5903 --noautoconsole --os-type solaris --  
accelerate --network=bridge:br0 --hvm
```

Para popular o repositório OpenLDAP com 3 utilizadores, basta copiar a configuração abaixo para o ficheiro "virtlab.ldif":

```
dn: ou=lei,dc=virtlab,dc=com
objectClass: organizationalUnit
ou: lei
```

```
dn: ou=lig,dc=virtlab,dc=com
objectClass: organizationalUnit
ou: lig
```

```
dn: uid=a20060266,ou=lei,dc=virtlab,dc=com
objectClass: inetOrgPerson
uid: a20060266
sn: Malta
givenName: Ricardo
cn: Ricardo Malta
displayName: Ricardo Malta
userPassword: password
mail: a20060266@virtlab.com
```

```
dn: uid=a20070051,ou=lei,dc=virtlab,dc=com
objectClass: inetOrgPerson
uid: a20070051
sn: Fontes
givenName: Marcelo
cn: Marcelo Fontes
displayName: Marcelo Fontes
userPassword: password
mail: a20070051@virtlab.com
```

```
dn: uid=a20083890,ou=lei,dc=virtlab,dc=com
objectClass: inetOrgPerson
uid: a20083890
sn: Malta
givenName: Rui
cn: Rui Malta
displayName: Rui Malta
userPassword: password
mail: a20083890@virtlab.com
```

Depois de guardado no ficheiro virtlab.ldif, executar o seguinte comando, introduzindo a password de administrador do OpenLDAP:

```
ldapadd -x -D cn=admin,dc=virtlab,dc=com -W -f
virtlab.ldif
```

O próximo passo é configurar a base de dados do SAG onde estão configuradas as máquinas virtuais e permissões dos utilizadores. Os UUID das máquinas virtuais são aleatórios na sua criação, caso não se force esse valor. Logo, no UUID apresentados podem não corresponder aos das máquinas que acabamos de criar. Nesse caso será necessário consultar o UUID atribuído no XML de configuração da máquina virtual [/etc/libvirt/qemu/nomedamáquina.xml]

Para criar a base de dados e as tabelas, utilizar os scripts que vão anexados ao projecto. Depois de criada a base de dados, executar os seguintes comandos DDL:

```
INSERT INTO machine (ip_port, uuid, machine_name)
VALUES
(5901, '965f0c28-ed3-4786-1fd5-bf37ceeda91f', 'WindowsXP'),
(5902, '8c195391-6ef4-790b-7cf7-7a960ba440ef', 'Ubuntu'),
(5903, 'c5b7168e-a95b-b1a5-6be3-86b7d3f86d50', 'Opensolaris');

INSERT INTO v_perm (t_uuid, u_uuid, perms) VALUES
('admin', '965f0c28-ed3-4786-1fd5-bf37ceeda91f', 'cr'),
('admin', '8c195391-6ef4-790b-7cf7-7a960ba440ef', 'cr'),
('admin', 'c5b7168e-a95b-b1a5-6be3-86b7d3f86d50', 'cr'),
('a20060266', '965f0c28-ed3-4786-1fd5-bf37ceeda91f', 'cr'),
('a20060266', '8c195391-6ef4-790b-7cf7-7a960ba440ef', 'cr'),
('a20060266', 'c5b7168e-a95b-b1a5-6be3-86b7d3f86d50', 'cr'),
('a20070051', '965f0c28-ed3-4786-1fd5-bf37ceeda91f', 'cr'),
('a20070051', '8c195391-6ef4-790b-7cf7-7a960ba440ef', 'cr'),
('a20083890', '965f0c28-ed3-4786-1fd5-bf37ceeda91f', 'cr');
```

A infra-estrutura está montada para funcionar em ambiente de teste, com 3 utilizadores e 3 máquinas virtuais. Cada utilizador tem permissões de acesso diferentes dos restantes e o administrador tem acesso a todas as máquinas virtuais.

Com tudo configurado basta apenas configurar a porta HTTP onde irá correr a aplicação e servidores auxiliares no ficheiro de configuração da aplicação (sag.conf), que se encontra na raiz do SAG. Depois de configurado basta apenas executar a aplicação de uma das seguintes formas:

```
# python sag.py  
  
ou  
  
# chmod +x sag.py  
# ./sag.py
```

Implementar o Acesso Web:

Com todos os pré-requisitos instalados anteriormente, instalar o cliente Web resume-se a:

- Copiar o portal para a root do servidor Apache [/var/www/virtlab]
- Alterar o endereço do WSDL da aplicação, que se encontra no ficheiro sagws_client_conf.php