Fedora Security Lab Test bench Documentation

Release 0.1

Fabian Affolter

October 29, 2017

Contents

The documentation of the Fedora Security Lab Test bench should provide the users with some basic information about the FSL Test bench and the steps taken for the creation.

Contents:

CHAPTER 1

Introduction

The Fedora Security Lab Test bench provides a save environment for security auditing and testing and can be used for teaching security testing methodologies. The purpose is to support students and teachers with Linux-based servers and services while they are working on information security, web security, reconnaissance, network analysis, network statistics, forensics, and rescue lessons.

As counterpart to the Fedora Security Lab, the Fedora Security Lab Test bench, is the bench where you can use your tools.

Beside vulnerable web applications, honeypots, and miscellaneous helpers a couple of well-known services are available to work with.

CHAPTER 2

Base

The initial idea behind the Test benches is that they can be built on-site by the customers. This way we don't need to ship pre-configured virtual machine images which are like blackboxes. The customers doesn't need to trust us about what's inside the VM or check everything by themselves. They should be in control of every setup step. Nothing is hidden and everything is transparent. No backdoors, no malware, no evil stuff.

The core components are installed out of the Fedora Package Collection, if they are available. This ensures that the operating system run the latest packages and behave with integrity.

After the setup of the FSL Test bench is possible to update the system with the package management tools.

```
$ sudo yum -y update
```

Vulnerable web application, PHP shells, and some helper tools are download directly from their upstream locations. It's not possible to update those application automatically.

One advantage of the on-site creation process, if creating a network host, is that the local network setup is detected and is used to configure the Test benches. The Test bench is ready to use.

A disadvantage is that a connection to the internet is needed during the setup process and a already working network infrastructure with DHCP/DNS has to be present. The customer needs minimal technical skill for the setup. It's not a one-click-thing.

It's also possible to build a Fedora Security Lab Test bench on a local machine which is very straight-forwards and easy to do. The local setup encapsulates the Test bench from the world and is only accessible from the host.

Architecture

The whole configuration of the Fedora Security Test bench is always made on top of a minimal Fedora installation or a default installation. It doesn't matter if the target system is a physical one or a virtual machine. After the installation of Fedora (or one of Fedora's downstream distribution like RHEL, Scientific Linux, etc. if wished[1]) is done and the setup the SSH connection is finished, Ansible is used to distribute the configurations of all included items.

[1] For EPEL aren't the same packages available as for Fedora. Please keep this in mind when trying to run the Test bench on a non Fedora machine.

It's possible to setup multiple Test benches at the same time with different features. Thanks to Ansible it's very easy to integrate new features or omit things. The so-called playbooks are easy to read and to write.

```
---
- hosts: all
  user: root
  tasks:
  - name: install default motd file
    template: src=fedora-motd.j2
              dest=/etc/motd
              owner=root
              mode=0755
```

Built-in modules facilitate the configuration tasks and templates supports the Jinja2 engine.

```
<Location />
  Order deny,allow
  Deny from all
  Allow from 127.0.0.1
  Allow from {{ ansible_eth0.ipv4.network }}/24
</Location>
```

Ansible is based on Python and doesn't need a client on the managed system.

For a permanent lab setup and for performance reasons separating and/or multiplying the Test benches would be a good choice.

All application and services included by the Fedora Security Lab Test bench are running on a current minimal Fedora installation. The Lighttpd server acts as primary web server and is serving the web interface of the Test bench. A MySQL server is available for database interactions and is hosting the databases for the vulnerable web applications.

System details

This section provides details about the different kind of Test bench setups which are possible and their specific default configuration.

Local virtual machine

This setup option (local-setup.yml) creates a local virtual machine with `virt-install` on the local host. The variables defined in the local.yml are fed to various files, a kickstart file, the `virt-install` script, and the network configuration shown later in this section.

The default settings in local.yml are very generic.

```
---
# User settings
language: en_US.UTF-8
keyboard: sg-latin1
timezone: Europe/Zurich

# Name of the virtual machine
```

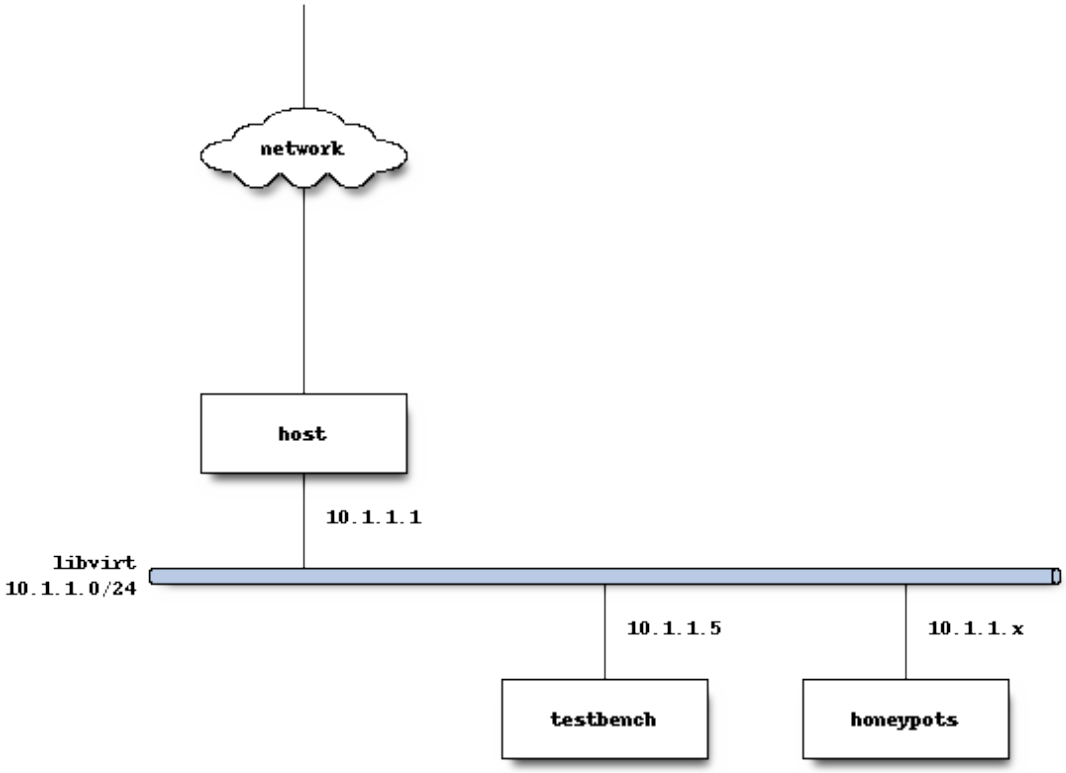
```
virtname: FSL-Test-bench

# Name of the disk image
img_name: fsl01

# Name of the bridge
bridge: virbr1

# Memory of the virtual machine
ram: 1024
```

The network for the Test bench is separated from the default libvirt network to avoid conflicts. It's using NAT and the interface **virbr1**.



The variables out of the local.yml file are used to fill the libvirt network template. The standard setup for the **testbench** network looks like the example below.

```
<network>
  <name>testbench</name>
  <uuid>391123e3-6666-154f-dd58-64b43435274755642</uuid>
  <forward mode='nat'/>
  <bridge name='virbr1' stp='on' delay='0' />
```

```
  <mac address='52:52:11:22:33:44'/>
  <ip address='10.1.1.1' netmask='255.255.255.0'>
    <dhcp>
      <range start='10.1.1.5' end='10.1.1.50' />
      <host mac='52:52:00:00:00:01' name='test-bench' ip='10.1.1.5' />
    </dhcp>
  </ip>
</network>
```

Additional details about the format and different option, can be found in the Network XML format documentation of the Libvirt Project.

Network host

If you choose this way to go, you need to take care of the network configuration and the installation of Fedora. In environments where a automatic provisioning solution for operating system is available this is the easiest way to start.

Virtualized host

The difference between be network host setup and the virtualized host setup is that multiple Test benches could run on a single host. The setup doesn't work with NAT with is the default of libvirt. You need to setup a bridge and your virtual machine must be able to connect to that bridge. You can use the fsl-virt-inst script. This script create a guest with a minimal Fedora installation which can be used for the Test bench.

This setup is perferred over a network host because it gives more flexibility.

Environment

> **Warning:** sorry, not implemented

The idea is to have an external USB/Firewire harddrive with all needed data on it. Meaning all packages (mirrored from the Fedora Package collection), all other components downloaded, and the configuration files ready. You plug that drive in and this machine act as server. Then DHCP/DNS services are setup automatically, a PXE configuration is put in place, and Test benches as virtual machines on the server are created. The clients use the server's PXE capability to setup themselfs. No access to the internet is needed and the whole environment can stay isolated. The only prerequirement then is hardware and a physical network which both are often present in class rooms.

CHAPTER 3

Installation & Setup

To setup a Test bench on a host in your network you need at least two systems either a physical machine with a virtual machine and a bridged network connection or two physical systems (one as system to perform the the actions and one which will serve as Test bench). In the latter case a working network is needed too, incl. DHCP/DNS. For the setup process a connection to the internet is mandatory because some files need to be downloaded. This guide will use the definitions from below for the two system to make it clear which one is involved:

Quick-start setup

This section describes the needed steps to setup a Test bench from a common point of view in quick and fast way. In the further sections covers special use cases which are described in detail then.

The setup of Ansible is explained on the Ansible Getting Started page. Here is only the setup of the managed nodes and special details for the management system covered. For every system you want to manage, you need to have the client's SSH key in the *authorized_keys* file of the managed system and Python.

Prerequisites

On the managing system you need the Ansible package.

```
# yum -y install ansible
```

Make sure that Python is installed on the managed node(s). If not, install the Python package. If you have performed a minimal Fedora installation Python is available. Otherwise:

```
# yum -y install python
```

Fedora Security Lab test bench git repository

You need to clone the Fedora Security Lab test bench git repository which contains all the playbooks. Playbooks are recipes to perform task on a remote system.

```
$ git clone git@github.com:fabaff/fsl-test-bench.git
```

If you want to contribute back to this Project, please fork it first.

SSH key

Then you must copy the SSH key of the managing system to the `authorized_keys` file of . Lauch the command from below on the managing system.

```
$ sudo ssh-copy-id -i /root/.ssh/id_rsa.pub root@[IP address of the node]
```

/etc/ansible/hosts

The file */etc/ansible/hosts* shall contain all managed hosts to be setup up. The available groups are:

- **fsl-tb**: Default group name for machine which uses the all-in-one playbook
- **fsl-tb-vpn**: Group name for machine which acts as VPN servers
- **fsl-tb-master**: Hosts for FSL Test bench guests when using virtualization
- **fsl_hosts**: Hosts to install the Fedora Security Lab package set

Those groups are mentioned in the playbooks to setup only the named hosts.

More information about this topic are available are in the Ansible documentation.

Variables

After cloning the git repository, please review and edit the variables files.

The file `variables/sensitive.yml` contains all passwords for root, the users, and the details for the certificate. Please edit this file according to your needs.

In the file `variables/local.yml` are several networking preferences stored. If you run into conflicts with your local network settings (e.g. IP range, etc.) please change the values.

Run it

Now let Ansible do the work. Below the command is shown to setup the Fedora Security Lab Test bench with the *all-in-one.yml* playbook.

```
$ sudo ansible-playbook fsl-test-bench/all-in-one.yml
```

All hosts which belongs to the **fsl-tb** group will be converted into Fedora Security Lab Test benches.

Setup on a local machine

The setup of the Fedora Security Lab Test bench as virtual machine on a local system is useful if you want to carry your Test bench around on your laptop and use it only for yourself. Another good reasons to use the Test bench as local machine are when you don't want to expose a system like this in your network or not want to use the Test bench on a dedicated/remote system.

At the moment there are two way available to deploy a Fedora Security Lab Test bench from scratch. One way is to use a simple bash script which is a collection of all needed steps. With the script the Test bench is setup without any user intervention and runs with the default values.

If you want to customize your Test bench the manual way is the right one to go.

Requirement

The requirements for running a Test bench on your local system are minimal.

- Fedora system which is capable to run a hypervisor (qemu/kvm)
- a working connection to the internet
- root access to the host
- disk space (at least 8 GB)

It may work on other distribution but this is not tested.

Automatic setup

For a fast setup of a local Test bench, just download the `fsl-tb-inst` script.

Get the script with curl.

```
$ curl https://git.fedorahosted.org/cgit/security-spin.git/plain/test-bench/fsl-tb-
↪inst -o fsl-tb-inst
```

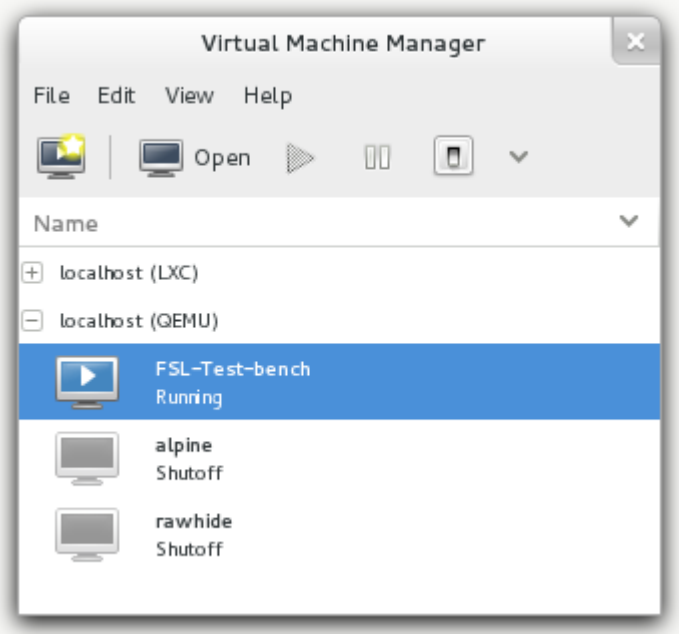
Set the execute permission as root.

```
# chmod +x fsl-tb-inst
```

Run the script as root.

```
# ./fsl-tb-inst
```

Depending on your hardware setup and the speed of your internet connection it takes some time to finish[1]. If you want to see what's going on, connect to the virtual machine **FSL-Test-bench** with Virtual Machine Manager per example.



After the setup of the virtual machine itself you can connect to the Test bench over ssh.

```
$ ssh -l root 10.1.1.5
```

[1] On a host system with an Intel(R) Core(TM)2 Quad CPU Q6600 @ 2.40 GHz CPU, 8 GB of memory, and Fedora 20 connected with a 35 Mbit/s line it take around 30 minutes to complete all tasks.

Check the logs to see what's happening.

```
# journalctl -f
```

Manual setup

If you want to have more control over the creation process, adding other settings, changing the configuration, or just not want to use the default values, the manual setup gives you that fexibility.

Create the file `/etc/ansible/hosts` with the following content.

```
[localhost]
127.0.0.1
```

Install the needed packages:

```
$ sudo yum -y install git ansible
```

Clone the FSL Test bench git repository.

```
$ git clone git://github.com/fabaff/fsl-test-bench.git
```

The file `variables/local.yml` contains variables for the virtual machine and the used virtal network. Modify the variables as you need. Especially when you detect a collision of the IP range with your local setup. If you have a host system with a lot of memory, increasing the amount of RAM assigned to the virtual machine is a good idea.

In the file `fsl-test-bench/all-in-one.yml` are all features listed. Comment- out unwanted playbook. Most playbook for services are independent but the ordering is relevant for tasks which modify the web interface. If the web server is not present, it doesn't make sense to add a page to the web interface.

All sensitive variables (password and certificate elements) are stored in the file `fsl-test-bench/variables/sensitive.yml`. Change the data in this file.

Run the playbooks. The first one setup your local machine with the needed elements (libvirt) and create the virtual maschine. The second is setting up the Test bench.

```
$ sudo ansible-playbook fsl-test-bench/local-setup.yml --connection=local
$ sudo ansible-playbook fsl-test-bench/all-in-one.yml
```

Now wait...If your are connected to your local machine then you can abort the task with `Ctrl+c c` when the virtual machine is ready. There are two wait cycles included, one for the vm setup and one for the launch of the vm.

Use `virsh` to check if the process is finished.

```
$ sudo virsh list --all
 Id    Name                           State
----------------------------------------------------
 1     FSL-Test-bench                 running
 -     alpine                         shut off
 -     rawhide                        shut off
```

If the vm is running, you can connect to the Test bench over ssh.

```
$ ssh -l root 10.1.1.5
```

Check the logs to see what's happening.

```
# journalctl -f
```

Setup on a host

To setup a Test bench as a separate machine you need at least two pysical systems. A management system and the system for the Test bench. We recommand to use the Test bench as virtual machine on a host. This has some benefits over the setup on a physical system. You can run multiple instances and creating new Test benches is fast and automated.The requirement are a host with a bridged network connection and a working network, incl. DHCP/DNS.

For the setup process a connection to the internet is mandatory because some files need to be downloaded. This guide will use the definitions from below for the two system to make it clear which one is involved:

- **System 1**: Is the managing system. Can be the same system from which you want to perform all testing task in the future.
- **System 2**: Will become the Test bench.

The first step is to create a running Fedora installation (a minimal installation is just fine) for the Test bench on System 2. The `fsl-virt-inst` script can help to get you in this matter. Make sure that SSH is working.

System 1 can run a Linux distribution of your choice. We assume that is a Fedora installation too. The setup process for System 2 will be done with Ansible. It enables us to manage systems over SSH in a simple, secure, and fast way. Install Ansible on System 1.

```
$ sudo yum -y install ansible
```

Now we need to clone the Fedora Security Lab test bench git repository which contains the playbooks on System 1. Playbooks are recipes to perform task on a remote system.

```
$ git clone git@github.com:fabaff/fsl-test-bench.git
```

System 2 needs Python. Make sure that it is available. If not install it.

Then we must copy the SSH key of System 1 to the `authorized_keys` file of System 2. Lauch the command from below on System 1.

```
$ sudo ssh-copy-id -i /root/.ssh/id_rsa.pub root@[IP address of System 2]
```

On System 1 edit the `/etc/ansible/hosts` file and add the IP address of System 2.

```
[fsl-tb]
IP address of System 1

[fsl-tb-vpn]
```

The file `variables/sensitive.yml` contains all passwords. If you don't want to run with default password, edit this file according your needs.

Now let Ansible do the work. Below the command is shown to setup the Fedora Security Lab Test bench on a single machine.

```
$ sudo ansible-playbook fsl-test-bench/all-in-one.yml
```

When all tasks are finished, the Test bench is ready. The overview page should be accessible: `http://[IP address of System 2]`.

Setup in an isolated environment

> **Warning:** sorry, not implemented. Development will hopefully happen in the future.

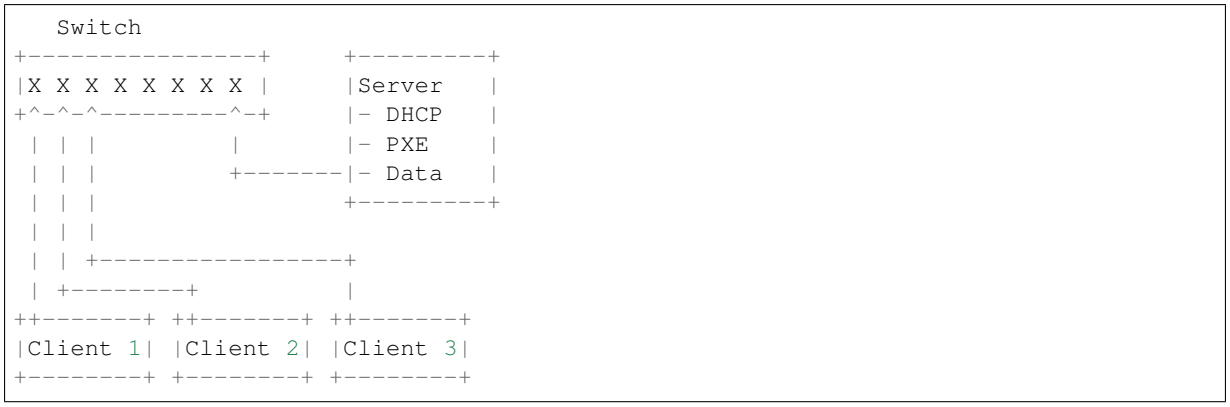
The Fedora Security Lab environment can be used to create a complete environment consisting of an attack target and attackers in an isolated area of an existing physical network or a class room.

Requirement

The requirements for running a Fedora Security Lab Environment are:

- A system which is capable of acting as server, is able to boot from external devices (USB), and have a network interface
- working physical network (all systems are connected to the same network segment)
- some systems capable for network booting (PXE boot)

Basically a class room for the computer science education is a good starting point.



Setup Course

- Disconnect the physical network from the outside
- Start the system which will become the server
- When the server is up and running, start your other system after you have changed their BIOS boot sequence to *Boot from network* or similar.

First steps

After the installation is done, the first step is to check if the Test bench is responding to ping requests. This should work because the whole setup process was relaying on a working network connection between the all involved systems. All examples are assuming that the Test bench was created with the default values as vm on a local machine (e.g. the vm has the IP address 10.1.1.5). If not, adjust the IP address of your Test bench according your setup.

```
$ ping -c 4 10.1.1.5
PING 10.1.1.5 (10.1.1.5) 56(84) bytes of data.
64 bytes from 10.1.1.5: icmp_seq=1 ttl=64 time=0.308 ms
64 bytes from 10.1.1.5: icmp_seq=2 ttl=64 time=0.407 ms
64 bytes from 10.1.1.5: icmp_seq=3 ttl=64 time=0.408 ms
64 bytes from 10.1.1.5: icmp_seq=4 ttl=64 time=0.248 ms

--- 10.1.1.5 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3000ms
rtt min/avg/max/mdev = 0.248/0.342/0.408/0.071 ms
```

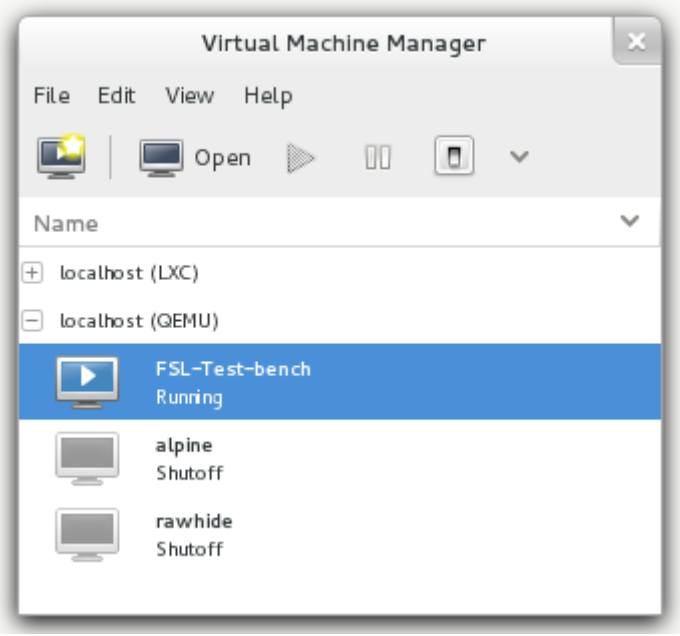


Fig. 3.1: Virtual Machine Manager

If you want to login directly in your virtual machine, launch Virtual Machine Manager (`virt-manager`) and connect to the virtual machine **FSL-Test-bench**.

Make a right-click on the `FSL-Test-bench` entry and choose **Open**. Login with username **root** and passwort **testbench**.

The fastest way is just to connect to the Test bench over ssh and login with username **root** and passwort **testbench**.:

```
$ ssh -l root 10.1.1.5
```

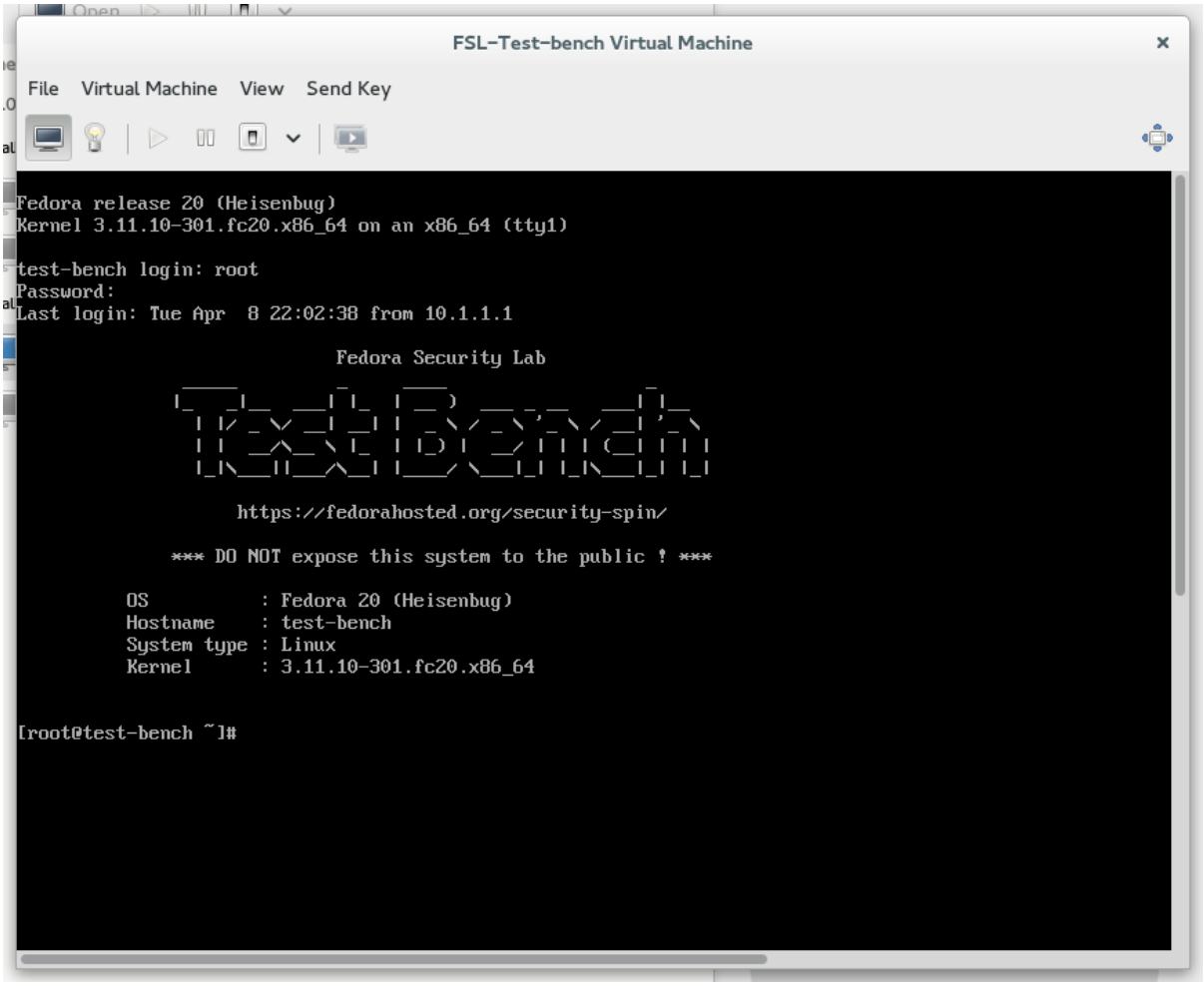


Fig. 3.2: Virtual Machine Manager console view

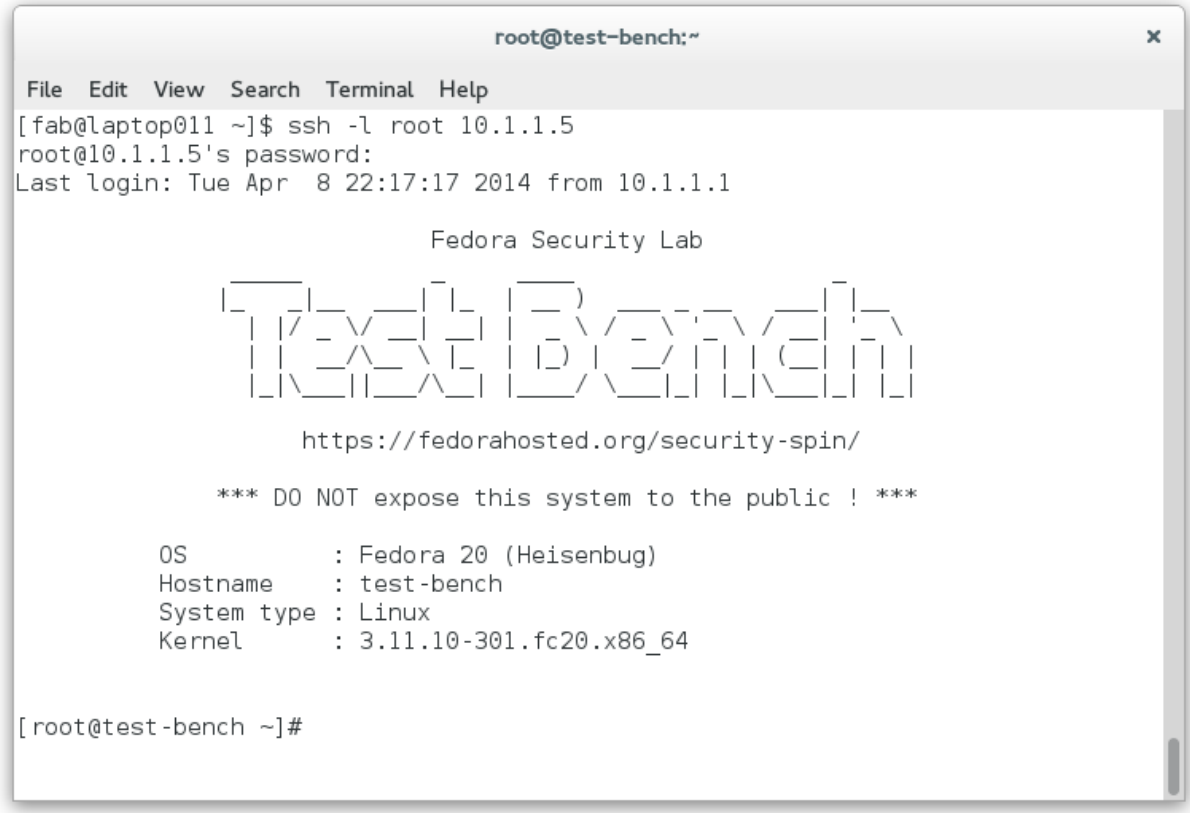


Fig. 3.3: motd of the Test bench

CHAPTER 4

Services

The Fedora Security Lab Test bench include a width variation of services. Most services are running with default configuration. If useful a web interface is provides.

Database server

The MariaDB database engine is used for the web applications but it is still possible to misuse it for your own requirements. All current available DBMS are accessible by remote systems with client tools. For management or administration tasks web interfaces are provided, please check the `Misc` section on the default start page of your FSL Test bench.

- MariaDB
- MySQL (replaced by MariaDB)
- mongoDB
- Sqlite

If you want to interact with the mongoDB instance, make your that you have the client tools installed on your system.

```
$ mongo testbench --host 10.0.0.64
```

File servers

Serving file is often an essential feature of a server. In the Linux world two popular systems are used, samba and nfs. It depends on the use case which system is more common. In a Linux-only environment nfs is a good choice. If you want to serve files for Microsoft Windows systems samba is an easy way to go.

- samba
- nfs

There is a samba share available.

```
$ nmbscan -h 10.0.0.64
nmbscan version 1.2.6 - laptop011 - Thu Apr 25 09:53:17 CEST 2013
domain MYGROUP
  server TEST-BENCH
    ip-address 10.0.0.64
      ip-name fedora-test.home.network
    server-software Samba 4.0.5
    operating-system Unix
    share samba
      share-type Disk
    share IPC$
      share-type IPC
      share-comment IPC Service (Samba Server Version 4.0.5)
```

FTP servers

File Transfer Protocol (FTP) is an importent protocol for transferring files from host to host. All FTP connections are unencrypted to make it possible to sniff the control and data connections between the client and the server. The listed ftp servers are ready to include:

- vsftpd
- proftpd
- pure-ftpd

To run all ftp servers on one machine it's needed that they use different ports. Table below shows the ports and the assigned ftp server.

Port	Server
21	vsftpd
2021	pure-ftpd
2221	proftpd

For vsftpd TLS support is coming soon and the configuration is not really working.

```
$ ftp 10.0.0.64
Connected to 10.0.0.64 (10.0.0.64).
220 (vsFTPd 3.0.2)
Name (10.0.0.64:fab): bob
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
500 OOPS: priv_sock_get_int
Passive mode refused.
```

Web servers

Every type of web server has its purpose and its unique fingerprint. To give the students the feeling of the real world, various web servers are running. They don't serve content, they are just lurking around for fingerprinting and bannergrabbing. The following web server are available.

apache

The Apache HTTP Server, commonly referred to as Apache, is a well-known web server application.

> **Warning:** sorry, not implemented/available at the moment.

cherokee

cherokee is a lightweight, high-performance web server/reverse proxy. This webserver offers support for FastCGI, SCGI, PHP, CGI, SSI, TLS and SSL encrypted connections, Virtual hosts, Authentication, on the fly encoding, Load Balancing, Apache compatible log files, Data Base Balancer, downtime-free updates and upgrades, and Reverse HTTP Proxy.

> **Warning:** sorry, not implemented/available at the moment.

darkhttpd

darkhttpd is a simple, fast HTTP 1.1 web server for static content. It does not support PHP or CGI etc but is designed to serve static content.

This example shows the details of the darkhttpd web server.

```
$ bannergrab 10.0.0.65 8887
HTTP/1.1 200 OK
Date: Wed, 29 May 2013 15:24:20 GMT
Last-Modified: Wed, 29 May 2013 14:44:55 GMT
Etag: "51a61467.3b0"
Content-Type: text/html
Content-Length: 944
Connection: close
Accept-Ranges: bytes
```

droopy

droopy is a mini web server with the purpose to let one upload files to a server. It's listening on port 8000.

flask

flask is a lightweight Python web application framework. It's and based on the Werkzeug WSGI toolkit and the Jinja2 template engine. This framework keeps the core simple but additional feature can be added through extensions.

http-server (node.js)

The http-server functionality is used on top of **'node.js'_**.

The next example shows a connection the http-server.

```
$ nc 10.0.0.64 8888
HEAD / HTTP/1.1
host: localhost

HTTP/1.1 200 OK
server: ecstatic-0.1.7
etag: "139483-944-Fri Apr 26 2013 19:09:31 GMT+0200 (CEST)"
last-modified: Fri, 26 Apr 2013 17:09:31 GMT
cache-control: max-age=3600
```

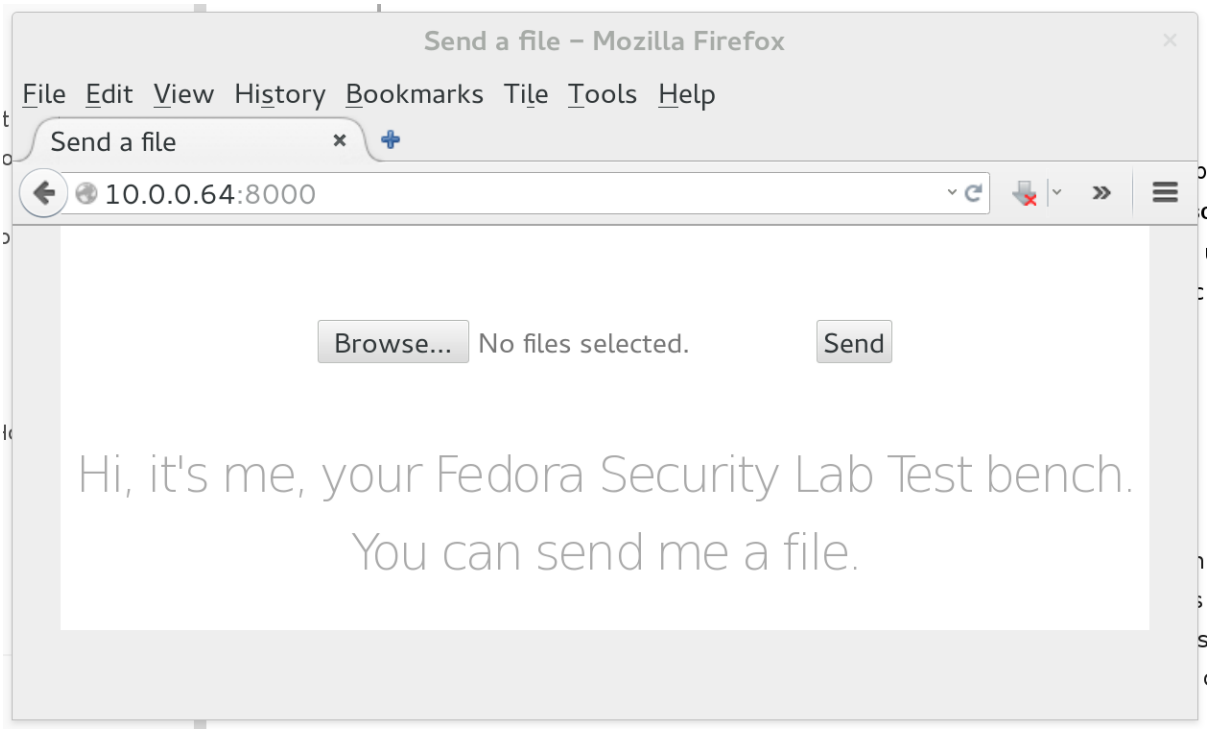


Fig. 4.1: droopy web interface

Fig. 4.2: http-server default page

```
content-type: text/html
Date: Fri, 26 Apr 2013 21:24:51 GMT
Connection: keep-alive
```

lighttpd

This is the server which is providing the web interface. lighttpd is optimized for speed while still standards-compliant, secure and flexible.

This example shows the details of the lighttpd web server.

```
$ bannergrab 10.0.0.64 80
HTTP/1.0 200 OK
Allow: OPTIONS, GET, HEAD, POST
Content-Length: 0
Connection: close
Date: Sat, 01 Nov 2014 13:18:35 GMT
Server: lighttpd/1.4.35
```

mongoose

mongoose is built on top of libmongoose embedded library. Libmongoose is used to serve Web GUI on embedded devices, implement RESTful services, RPC frameworks (e.g. JSON-RPC), handle telemetry data exchange, and perform many other tasks in various different industries.

This example shows the details of the mongoose web server.

```
$ bannergrab 10.0.0.65 8889
HTTP/1.1 200 OK
Date: Wed, 29 May 2013 15:24:20 GMT
Last-Modified: Wed, 29 May 2013 14:44:55 GMT
Etag: "51a61467.3b0"
Content-Type: text/html
Content-Length: 944
Connection: close
Accept-Ranges: bytes
```

nginx

nginx [engine x] is an HTTP and reverse proxy server, as well as a mail proxy server. Thanks to accelerated reverse proxying with caching, nginx is able to provide simple load balancing and fault tolerance.

nginx is the only web server with SSL support.

```
$ nc 10.0.0.64 8080
HEAD / HTTP/1.1
host: localhost

HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Accept-Ranges: bytes
ETag: W/"7777-1342949470000"
Last-Modified: Sun, 22 Jul 2012 09:31:10 GMT
Content-Type: text/html
Content-Length: 7777
Date: Fri, 26 Apr 2013 21:20:57 GMT
```

A connection to nginx over SSL.

```
$ openssl s_client -crlf -connect 10.0.0.64:443
CONNECTED(00000003)
depth=0 C = CH, ST = BE, L = Berne, O = Test bench, CN = test-bench.localdomain
verify error:num=18:self signed certificate
verify return:1
depth=0 C = CH, ST = BE, L = Berne, O = Test bench, CN = test-bench.localdomain
verify return:1
---
Certificate chain
 0 s:/C=CH/ST=BE/L=Berne/O=Test bench/CN=test-bench.localdomain
   i:/C=CH/ST=BE/L=Berne/O=Test bench/CN=test-bench.localdomain
---
Server certificate
-----BEGIN CERTIFICATE-----
[snip]
-----END CERTIFICATE-----
subject=/C=CH/ST=BE/L=Berne/O=Test bench/CN=test-bench.localdomain
issuer=/C=CH/ST=BE/L=Berne/O=Test bench/CN=test-bench.localdomain
---
No client certificate CA names sent
---
SSL handshake has read 1770 bytes and written 369 bytes
---
New, TLSv1/SSLv3, Cipher is DHE-RSA-AES256-SHA
Server public key is 2048 bit
Secure Renegotiation IS supported
Compression: NONE
Expansion: NONE
SSL-Session:
    Protocol  : TLSv1
    Cipher    : DHE-RSA-AES256-SHA
    Session-ID: 33515B817[snip]427BB415
    Session-ID-ctx:
    Master-Key: 0956B7B[snip]F729586
    Key-Arg   : None
    Krb5 Principal: None
    PSK identity: None
    PSK identity hint: None
    TLS session ticket lifetime hint: 300 (seconds)
    TLS session ticket:
    [snip]
    Start Time: 1367011380
    Timeout   : 300 (sec)
    Verify return code: 18 (self signed certificate)
---
HEAD / HTTP/1.1
host: localhost

HTTP/1.1 200 OK
Server: nginx/1.2.8
Date: Fri, 26 Apr 2013 21:23:16 GMT
Content-Type: text/html
Content-Length: 944
Last-Modified: Fri, 26 Apr 2013 17:01:48 GMT
Connection: keep-alive
Accept-Ranges: bytes
```

pywebserve

pywebserve aims to expose a local directory to the world. It is using only Python modules (BaseHTTPServer and SimpleHTTPServer) and can be controlled by systemd.

The server is listening on port 8880.

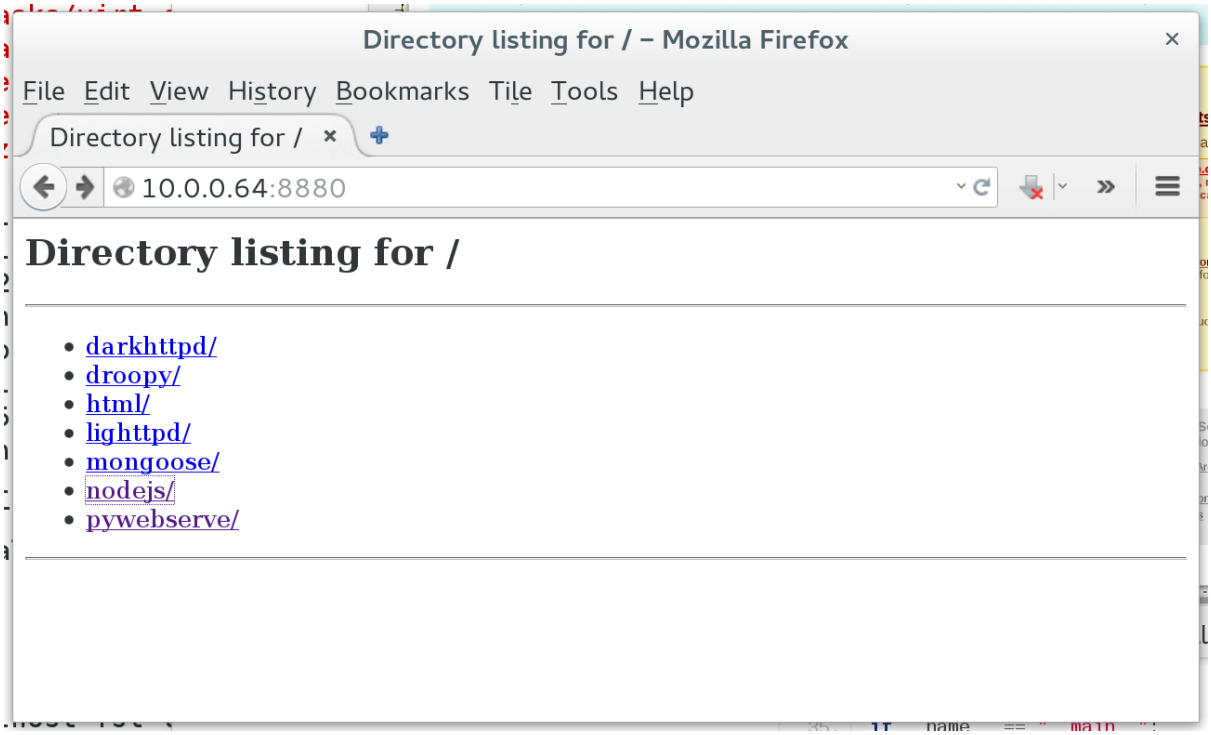


Fig. 4.3: pywebserve

This example shows the details of the pywebserve web server.

```
$ nc 10.0.0.64 8880
HEAD / HTTP/1.1
host: localhost

HTTP/1.0 200 OK
Server: SimpleHTTP/0.6 Python/2.7.5
Date: Sat, 01 Nov 2014 13:12:15 GMT
Content-type: text/html; charset=UTF-8
Content-Length: 434
```

tomcat

Apache Tomcat is an open source software implementation of the Java Servlet and JavaServer Pages technologies. This server is listening on port 8080. At the moment there are no pages served from this server.

This example shows the details of the Tomcat web server.

```
$ bannergrab 10.0.0.64 8080
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Type: text/html;charset=ISO-8859-1
Date: Sat, 01 Nov 2014 13:17:27 GMT
Connection: close
```

To run all web servers on one machine it's needed that they use different ports. Table below shows the ports and the assigned web server.

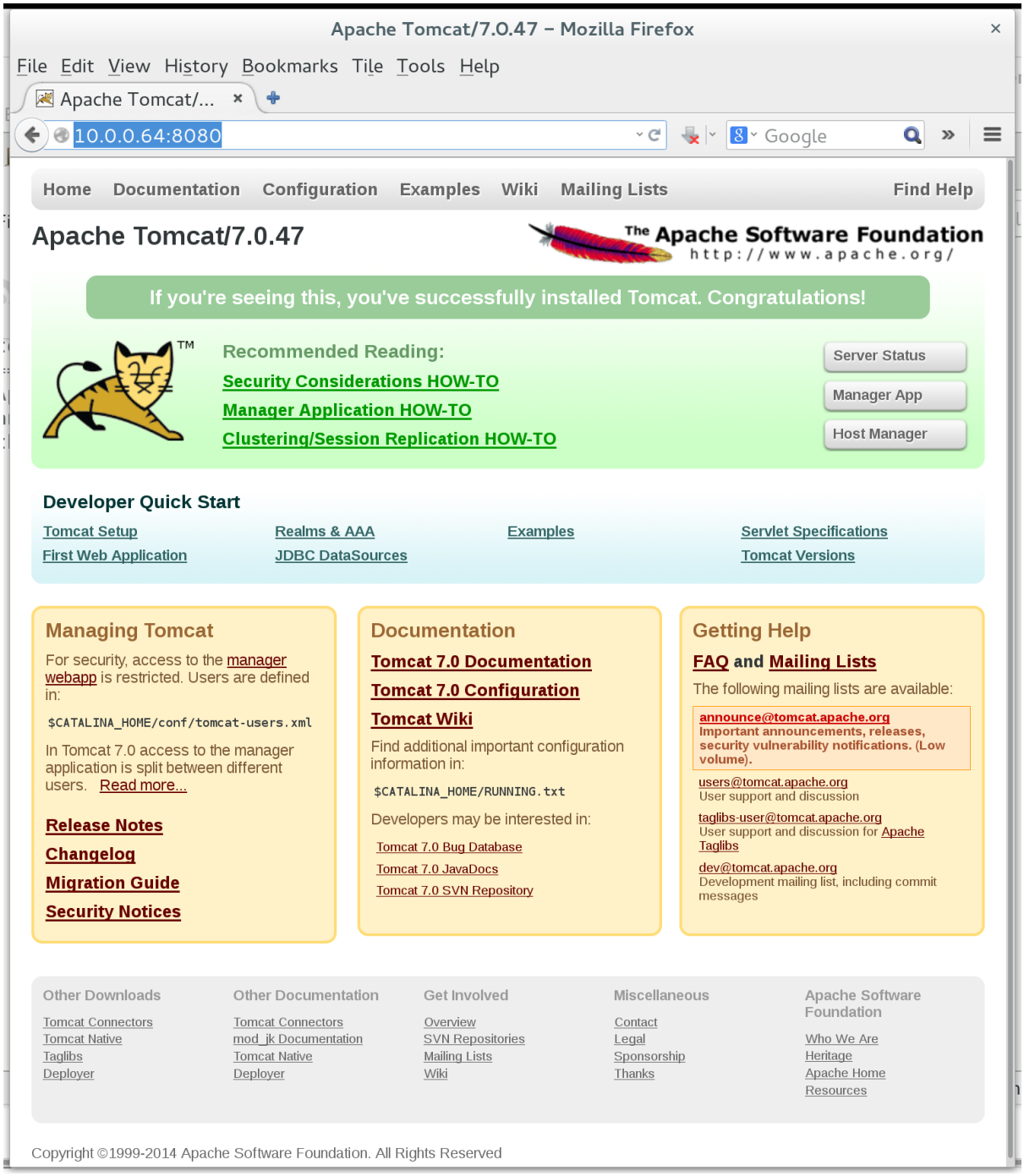


Fig. 4.4: Tomcat admin web interface

Port	Server
80	lighttpd
8000	droopy
8008	cherokee
8080	tomcat
8800	apache
8808	nginx
8880	pywebserve
8888	http-server
8889	mongoose
8887	darkhttpd
8886	flask

At the moment most web servers don't support https. This is a task for the future. The only web server with SSL support on the FSL Test bench is *nginx*.

Other servers/services

The Fedora Security Lab Test bench is hosting some services which are usualy not found on public accessible systems. `telnet` was replaced with more secure systems. Nowadays `tftp` is mainly used for provisioning VoIP installations. Print servers like `cups` are used in office environments.

To give the students the possibility to work with VPN, an OpenVPN setup with a static key is included.

tftp

tftp (xinetd) is a single port Trivial File Transfer Protocol server

The tftp server is serving a simple text file.

```
$ ls
$ tftp 10.0.0.64
tftp> get info.txt
tftp> quit
$ ls
info.txt
```

telnet

`telnet` supports bidirectional interactive text-oriented communication.

You should be able to connect to a telnet server.

```
[testbench@fsl-tb09 ~]$ telnet 10.1.1.5
Trying 10.1.1.5...
Connected to 10.1.1.5.
Escape character is '^]'.
Fedora release 20 (Heisenbug)
Kernel 3.13.8-200.fc20.x86_64 on an x86_64 (1)
test-bench login:
```

OpenVPN

OpenVPN is an software application which makes virtual private network (VPN) techniques available for creating secure point-to-point or site-to-site connections over unsecure networks like the internet. OpenVPN is capable of traversing firewall and common SOHO router with network address translators (NATs).

Peers are allowed to authenticate each other using certificates, a pre-shared secret key, or username/password. The FSL Test Bench only provides an OpenVPN server with a static key configuration.

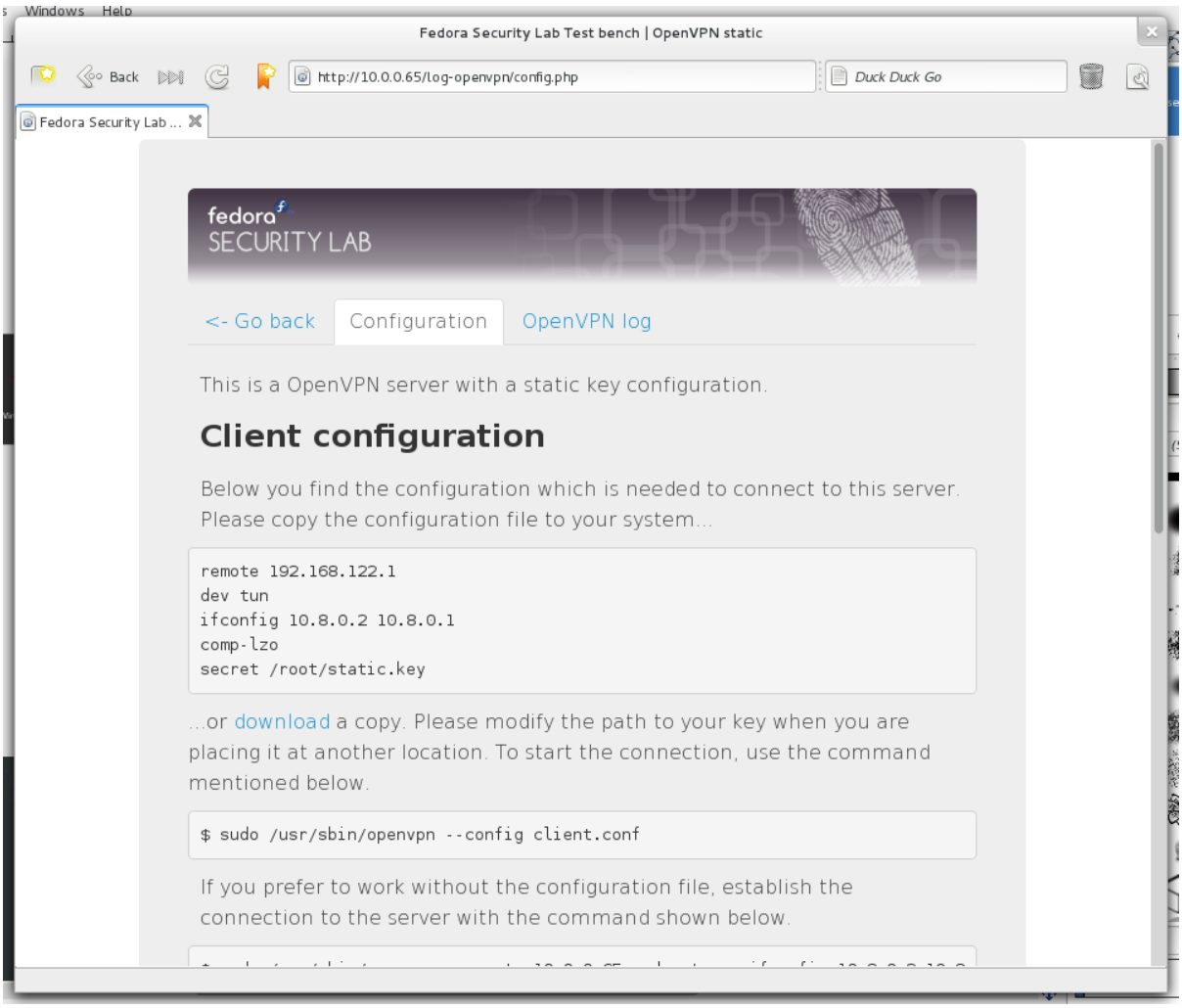


Fig. 4.5: Configuration page for OpenVPN

openssh

openssh (Port 22) encrypts communication sessions over a computer network using the SSH protocol.

Banner grabbing with `netcat` will give you the version back:

```
$ nc -v 10.0.0.65 22
Ncat: Version 6.25 ( http://nmap.org/ncat )
Ncat: Connected to 10.0.0.65:22.
SSH-2.0-OpenSSH_6.1
```

There are two other ways to retrieve the version with nmap. The first is

```
$ nmap -sV -p 22 10.0.0.65
[...]
Host is up (0.00053s latency).
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 6.1 (protocol 2.0)
```

The second is

```
$ nmap -sV -p 22 --script=banner 10.0.0.65
[...]
Host is up (0.00061s latency).
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 6.1 (protocol 2.0)
|_banner: SSH-2.0-OpenSSH_6.1
```

dropbear

dropbear which is running on port 222 is a lightweight SSH server.

To retrieve the version with nmap, use the command mentioned below:

```
$ nmap -sV -p 222 --script=banner 10.0.0.65
[...]
Host is up (0.00045s latency).
PORT    STATE SERVICE VERSION
222/tcp open  ssh     Dropbear sshd 2012.55 (protocol 2.0)
|_banner: SSH-2.0-dropbear_2012.55
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Banner grabbing with `netcat` will give you the version back.:

```
$ nc -v 10.0.0.65 222
Ncat: Version 6.25 ( http://nmap.org/ncat )
Ncat: Connected to 10.0.0.65:222.
SSH-2.0-dropbear_2012.55
```

cups

cups is a standards-based printing system and uses the Internet Printing Protocol (IPP) to support printing to local and network printers. The web interface accessible at http://10.0.0.65:631/

ngircd

ngircd is a lightweight Internet Relay Chat server.

xrdp

xrdp is an remote desktop protocol (RDP) server. To connect to the FSL Test Bench use Vinagre which is named usually **Remote Desktop Viewer** in graphical user environment (`yum -y install vinagre`) and is a client that support various protocols (VNC, ssh, rdp, and spice) or a client of your choice.

ntp

The Network Time Protocol (ntp) is a networking protocol for the synchronization of clocks of computer systems over networks. NTP is providing the information in UTC (Coordinated Universal Time).

Login your FSL Test bench to check if you have connections to ntp servers.:

```
# ntpq -p
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
*ds1789963.dedic 192.53.103.103   2 u    1   64    1   16.390    2.484   0.674
 ns1.pmodwrc.ch  189.247.1.117    2 u    2   64    1   19.949    0.255   0.502
```

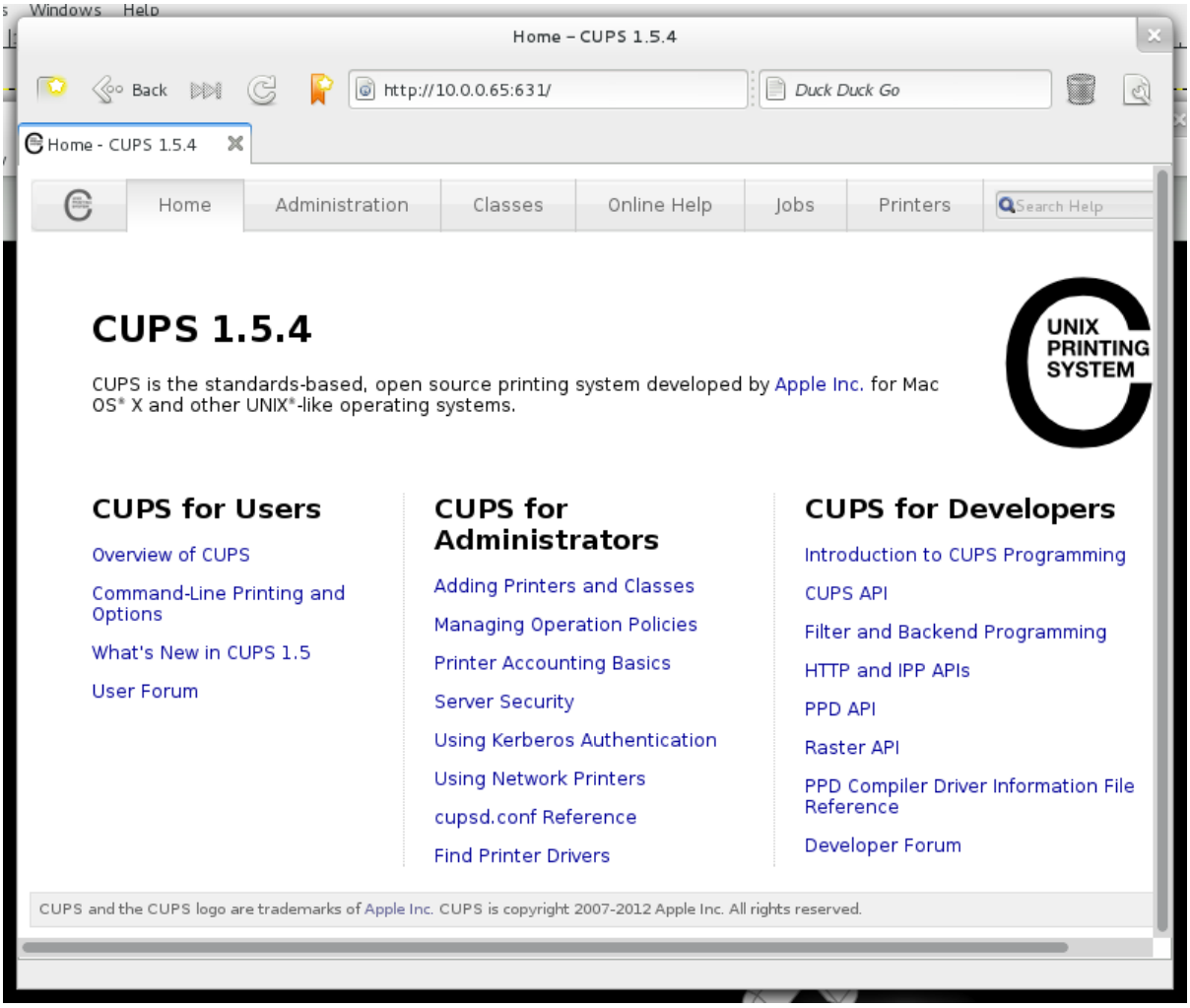


Fig. 4.6: CUPS web interface

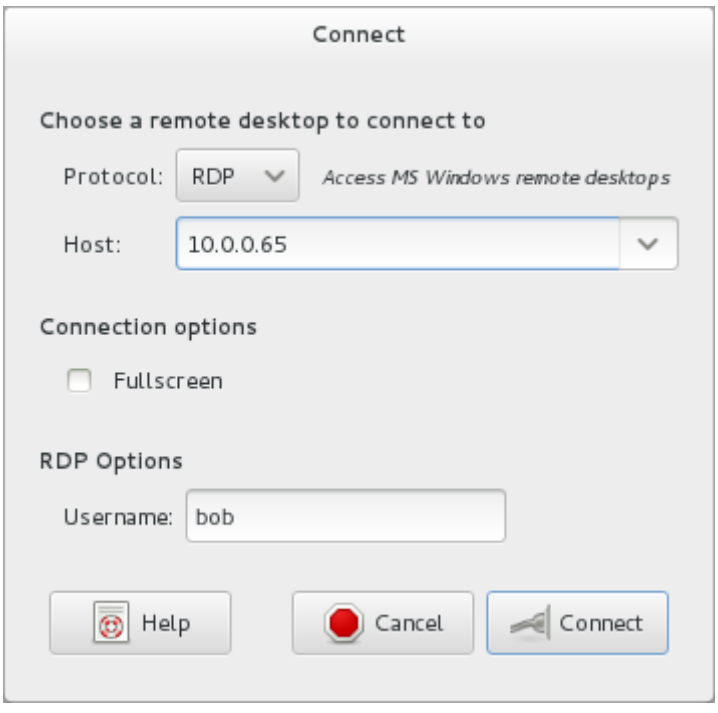


Fig. 4.7: Remote Desktop Viewer configuration

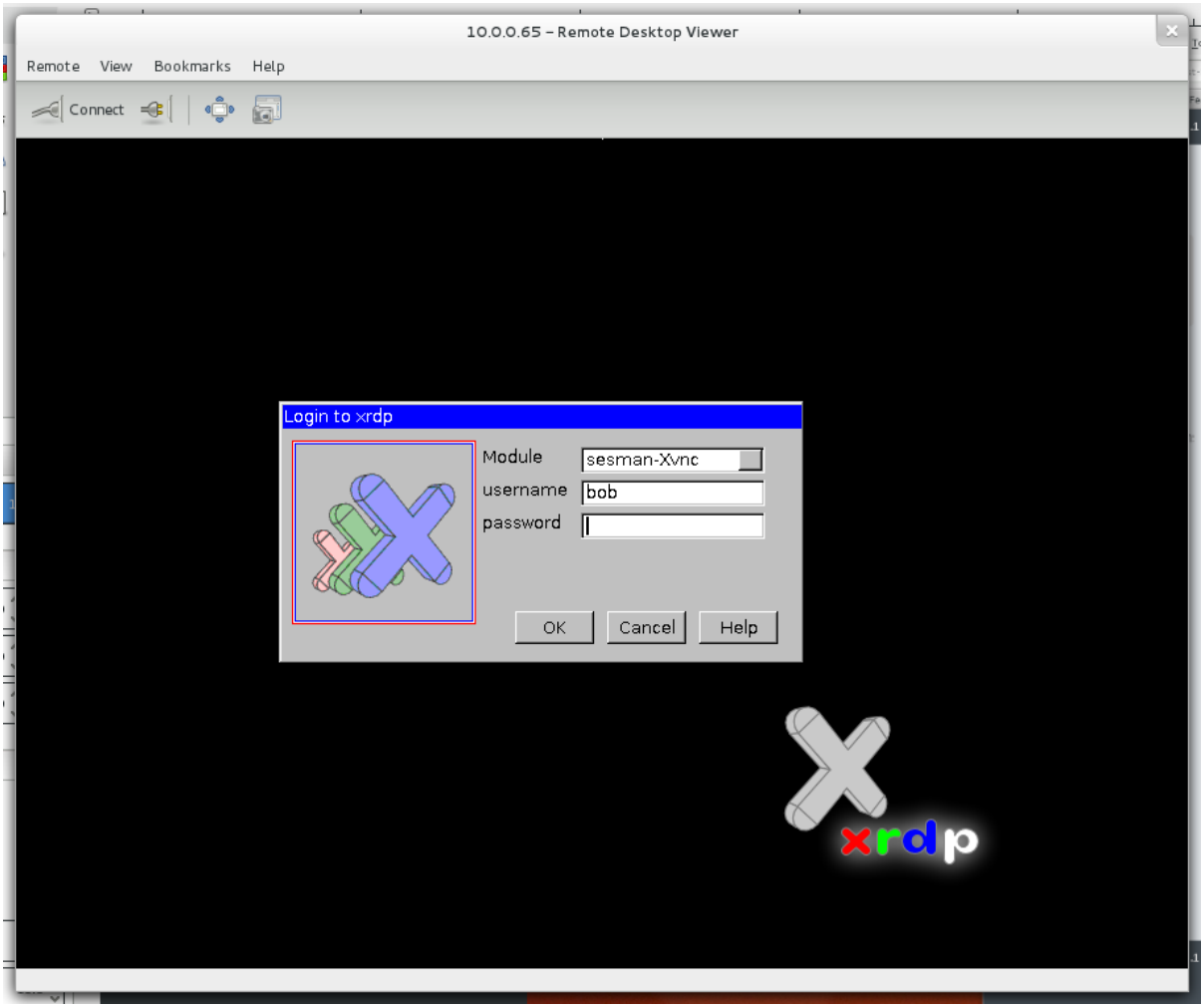


Fig. 4.8: Remote Desktop Viewer configuration

```
 ntppublic.uzh.c 130.60.205.7     3 u    1   64    1   16.315    2.361   0.341
 aerith.projectd 217.147.208.1    3 u    2   64    1   22.728   -0.846   0.022
```

Sync your clock with the Fedora Security Lab Test Bench:

```
$ sudo ntpdate 10.0.0.65
16 Aug 10:56:08 ntpdate[30588]: adjust time server 10.0.0.65 offset 0.002292 sec
```

Unless an error message is displayed, the system time of your local system should now be set.

Note: It was not tested if this works without a connection to the internet.

mosquitto

mosquitto is a MQ Telemetry Transport (MQTT) message broker. The MQTT protocol provides a lightweight method of carrying out messaging using a publish/subscribe model. It is useful and suitable for "machine to machine" messaging in various way, e. g. for connections with remote locations or just to collect your data from a microcontroller system.

Subscribing to the topic **fsl/testbench** of the MQTT broker from your local machine:

```
$ mosquitto_sub -h 10.0.0.65 -d -t fsl/testbench
Client mosqsub/24366-laptop011 sending CONNECT
Client mosqsub/24366-laptop011 received CONNACK
Client mosqsub/24366-laptop011 sending SUBSCRIBE (Mid: 1, Topic: fsl/testbench,␣
↪QoS: 0)
Client mosqsub/24366-laptop011 received SUBACK
Subscribed (mid: 1): 0
```

The FSL Test Bench is publishing permanently on a random value in the interval between 1 to 30 seconds messages. The default string contains **MQTT message from FSL Test Bench.** and a time stamp.

Manually publishing messages on your FSL Test bench can be done with the topic **fsl/testbench**. If you want to publish the message directly from your FSL Test Bench, use the command mentioned below:

```
$ mosquitto_pub -d -t fsl/testbench -m "This is a message from your FSL Test bench"
Client mosqpub/20531-test-bench sending CONNECT
Client mosqpub/20531-test-bench received CONNACK
Client mosqpub/20531-test-bench sending PUBLISH (d0, q0, r0, m1, 'fsl/testbench', .
↪.. (42 bytes))
Client mosqpub/20531-test-bench sending DISCONNECT
```

If you want to pusblish a message from your local machine, the broker's IP address is needed additionally.:

```
$ mosquitto_pub -h 10.0.0.65 -d -t fsl/testbench -m "This is a message from your␣
↪FSL Test bench"
```

You should now get the message from the FSL Test Bench.

```
Client mosqsub/24366-laptop011 received PUBLISH (d0, q0, r0, m0, 'fsl/testbench', .
↪.. (42 bytes))
This is a message from your FSL Test bench
```

prosody

prosody is a communications server for Jabber/XMPP.

Using the server

To connect the Jabber server with mcabber create the configuration file `.mcabberrc` in the home directory with the following content for the **admin**:

```
set jid = admin@10.0.0.65
set password = admin
set server   = 10.0.0.65
set resource = console
set priority = 1
set ssl_ignore_checks = 1
```

Or if you want to connect as user **bob**. Open an additional terminal and switch to user **bob**:

```
# su - bob
```

Create the `.mcabberrc` configuration file for **bob**:

```
set jid = bob@10.0.0.65
set password = bob
set server   = 10.0.0.65
set resource = console
set priority = 1
set ssl_ignore_checks = 1
```

Start `mcabber`

$ mcabber

Add all users you like to your roster and vice versa. Replace the usernames with the user you would like to add.:

```
/add bob@10.0.0.65
/authorization allow admin@10.0.0.65
```

When done, quit:

```
/quit
```

Telnet console

On the Fedora Security Lab Test bench the prosody server provides a telnet console to interact with.

```
$ telnet localhost 5582
Trying ::1...
telnet: connect to address ::1: Connection refused
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
|                     ____                  \   /     _
                     |  _ \ _ __ ___   ___  _-_   __| |_   _
                     | |_) | '__/ _ \/ __|/ _ \ / _` | | | | |
                     |  __/| | | (_) \__ \ |_| | (_| | |_| |
                     |_|   |_|  \___/|___/\___/ \__,_|\__, |
                     A study in simplicity             |___/


| Welcome to the Prosody administration console. For a list of commands, type: help
| You may find more help on using this console in our online documentation at
| http://prosody.im/doc/console
```

Let's get the uptime from the server as example.

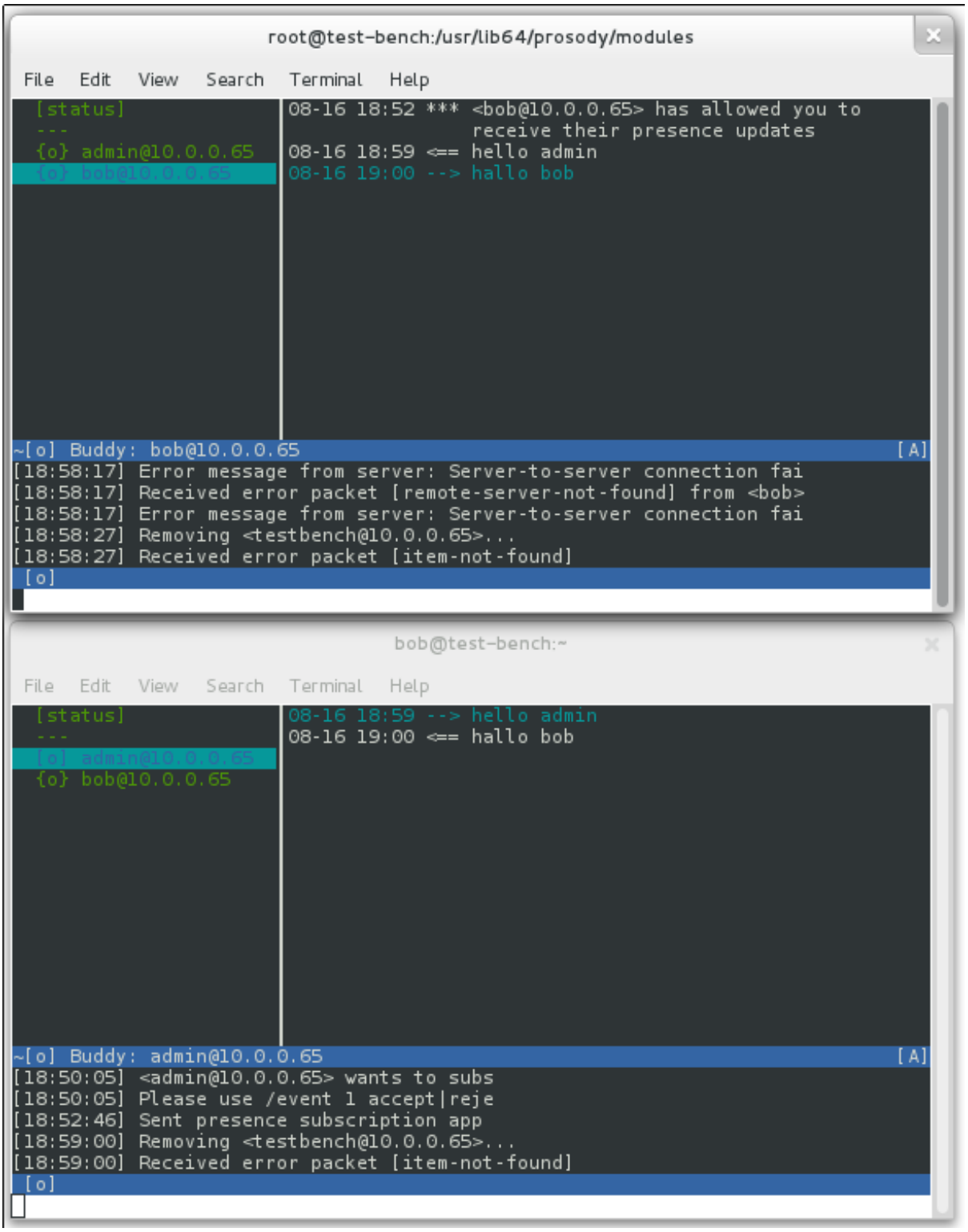


Fig. 4.9: Mcabber

```
server:uptime()
| OK: This server has been running for 0 days, 0 hours and 7 minutes (since Fri␣
↪Aug 16 17:03:24 2013)
```

snmp

The Simple Network Management Protocol (SNMP) protocol was designed for monitoring the health and welfare of computer and network equipment.

Get the data pn your Fedora Security Lab Test Bench:

```
$ snmpwalk -v2c -c public localhost system
```

Or check it from a system in the same network:

```
$ snmpwalk -v2c -c public 10.0.0.64 system
```

CHAPTER 5

Applications

All vulnerable web application and helper tools are accessible from the bootstrap-based web interface hosted on the Test bench.

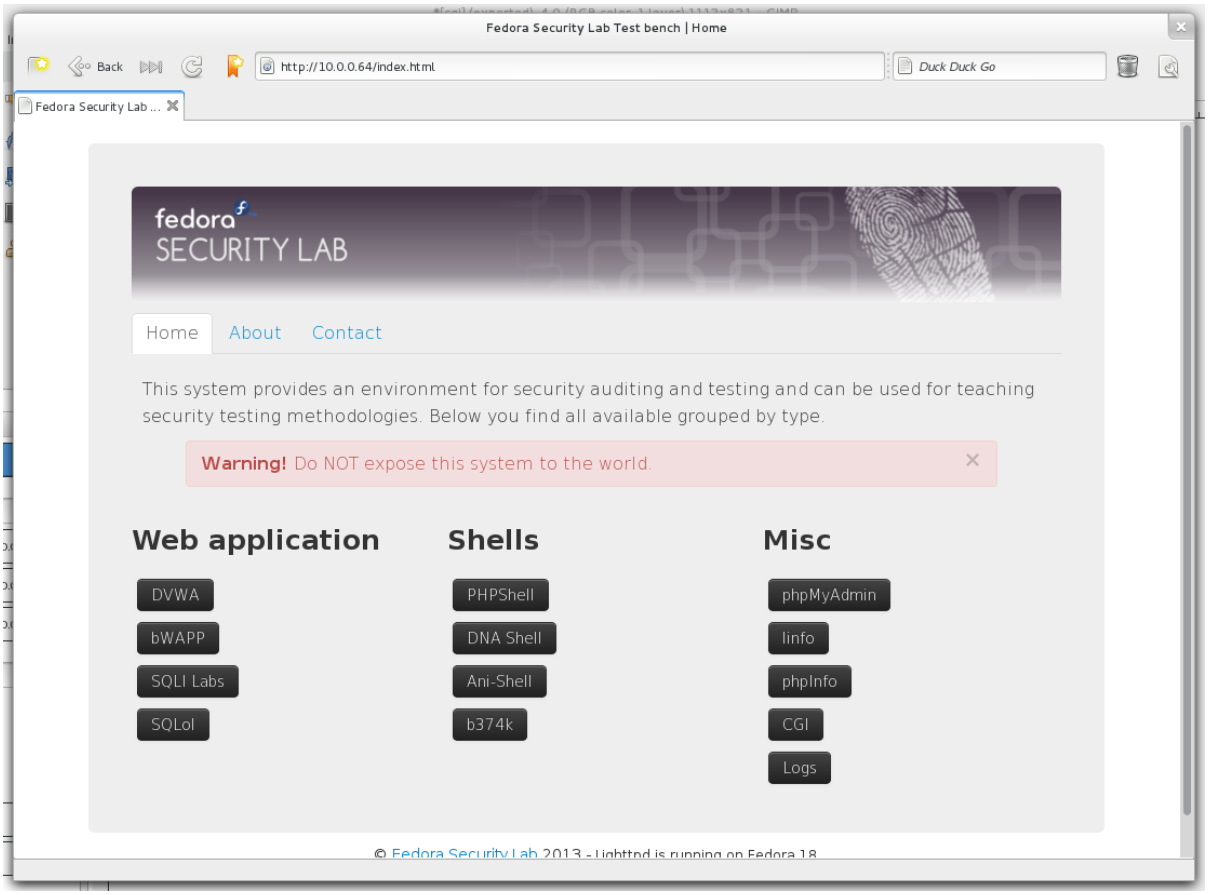


Fig. 5.1: Default start page of the web interface

The web interface is build during the setup process of the Test bench and only available features are shown.

Vulnerable web applications

The focus of vulnerable web application is to educate the people about security flaws in web application. SQL injection, file injection, cross-site scripting, code injection, and request forgery are threats which could have high impact.

- DVWA
- bWAPP
- SQLI Labs
- MCIR
- OWASP Hackademic Challenges Project
- XSSeducation
- Bricks

DVWA

Damn Vulnerable Web App (DVWA) is a PHP/MySQL web application that is damn vulnerable. The main goals of DVWA are to be an aid for security professionals to test their skills and tools. It should help web developers to better understand the processes of securing web applications and aid teachers/students to teach/learn web application security in a class.

bWAPP

Or a buggy web application is a web application build to allow security enthusiasts, students, and developers to better secure web applications. bWAPP prepares to conduct successful penetration testing and ethical hacking projects.

SQLI Labs

A platform to learn about SQL injection (SQLI). The labs are covering a wide range of injections (Union select, blind, update query, insert query, etc.).

MCIR

The Magical Code Injection Rainbow (MCIR) is a framework for building configurable vulnerability testbeds. It includes cryptomg, shellol, sqlol, xmlmao, and xssmh.

OWASP Hackademic Challenges Project

The OWASP Hackademic Challenges is an open source project that can be used to test and improve one's knowledge of web application security.

XSSeducation

XSSeducation is a set of Cross Site Scripting vulnerable PHP pages for learning about XSS Vulnerabilities.

Bricks

Bricks is built on PHP and MySQ and serves as a web application security learning platform. It has strong focuses on variations of commonly seen application security issues. All 'Bricks' has some sort of security issue and those can be leveraged.

PHP Shells

On a productive system shells are dangerous because they let an attacker to execute arbitrary shell-commands or browse the filesystem on your server. The shells mentioned below are available for easy integration to give a taste on how shells works and how they can be detected.

- AJAX shell
- Ani Shell
- b374k
- DNA Shell
- Escobar
- PHP Shell
- WSO Shell

Other web applications

Beside the vulnerable web applications, the services, and the PHP shells some additional web applications and tools are included. They facilitate the maintenance of the Test bench and are providing details about the various services and the system itself.

- linfo is a small PHP application that displays hardware details and real time health of your Test bench system.
- phpMyAdmin is a web appplication to handle the administration of MySQL servers.
- Log viewer for some services.
- CGIs which details about this system.
- PHP shell detector is a php script that helps you find and identify php/cgi(perl)/asp/aspx shells.

Note: PHP shell detector is not inclued by default. This tool is not able to work with the present amount of files.

CHAPTER 6

Machines

The Fedora Security Lab Test bench provides two types of additional systems for interaction. The first type are virtual systems which are using operating system-level virtualization (LXC) and the second are low-interaction honeypots based on honeyd.

Containers

The high-interaction "honeypots" are running as LXC (Linux containers). LXC provides system-level virtualization which has its own processes and own network space. This means that the containers are able to run linux systems in a isolated and virtual environment. The containers are separated from the Fedora Security Lab Test Bench and are using libvirt for the network.

Network

The containers are placed in a separated network which is running in **route** mode. This way the network traffic from the containers can pass back and forth without using NAT. The downside is that additional configuration on the clients is needed.

- Network mode: Routed
- Gateway: 10.10.1.1 (MAC address: 52:52:11:11:11:11)
- Network: 10.10.1.0/255
- DHCP: on
- DHCP range: 10.10.1.50 - 10.10.1.60

To access the container network you need to add a static route the this network. libvirt acts as virtual router on your Fedora Security Lab Test Bench and the hosts on the physical network do not know that there is a subnet.

```
$ sudo route add -net 10.10.1.0 netmask 255.255.255.0 gw [IP address of your FSL␣
↪Test Bench] dev [Interface]
```

After adding the route, check if the containers are responding.

Available systems

Container name	MAC address	IP address	Details
web01	52:52:22:22:22:22	10.10.1.60	•
web02	52:52:33:33:33:33	10.10.1.61	•

Management

The containers are launched automatically then the Fedora Security Lab Test Bench starts. It makes sense to shut them down if you are running a system which has only limited resources and you are working on different sections.

There are several ways for maintaining the containers. If you have a SSH connection to your Test Bench, you can use `virsh`.

Show all running containers:

```
# virsh --connect lxc:/// list --all
```

For stopping:

```
# virsh --connect lxc:/// shutdown [container name]
```

For starting:

```
# virsh --connect lxc:/// start [container name]
```

For more details about `virsh` please check the virsh man page.

```
$ man virsh
```

or the virsh command reference.

For managing the containers in a GUI way launch Virtual Machine Manager (`virt-manager`). The first step is to connect to your Fedora Security Lab Test Bench. Goto **File** and choose **Add connection...** after Virtual Machine Manager was started. Choose **LXC (Linux Containers)** as Hypervisor, **SSH** as Method, **root** as Username is ok, and enter the IP address of Fedora Security Lab Test Bench.

All containers can now be manipulated (shutdown, reboot, etc.) like virtual machine hosted on your local system if you have any.

Virtual Machine Manager will present you a login shell after you have open a container.

Honeypots

Currently the low-interaction honeypots make use of honeyd. Those honeypots are only intended to be targets for port scans. For details about the honeypot configuration, please check the configuration template.

- Microsoft Windows XP
- Microsoft Windows 2003 Server
- Linux 2.4.20

The honeypots are requesting IP addresses by DHCP.

```
Apr 24 10:09:35 test-bench honeyd[1077]: [eth0] got DHCP offer: 10.0.0.133
Apr 24 10:09:35 test-bench honeyd[1077]: [eth0] got DHCP offer: 10.0.0.134
Apr 24 10:09:35 test-bench honeyd[1077]: [eth0] got DHCP offer: 10.0.0.135
```

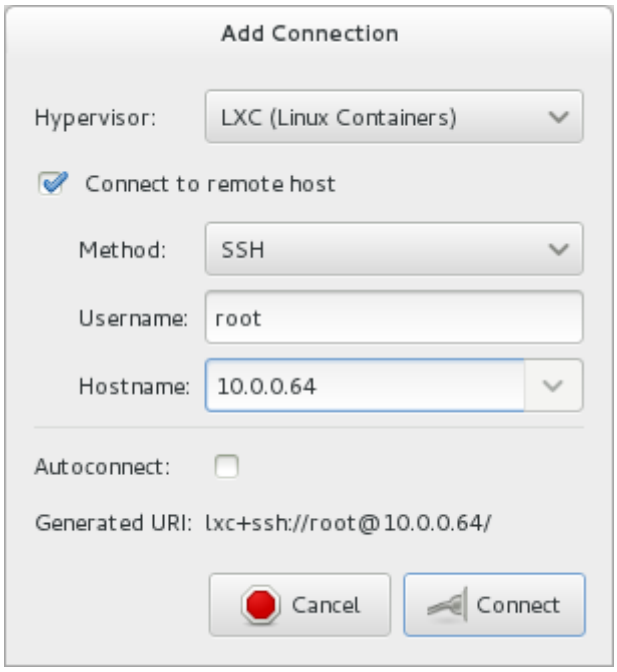


Fig. 6.1: Add connection in Virtual Machine Manager

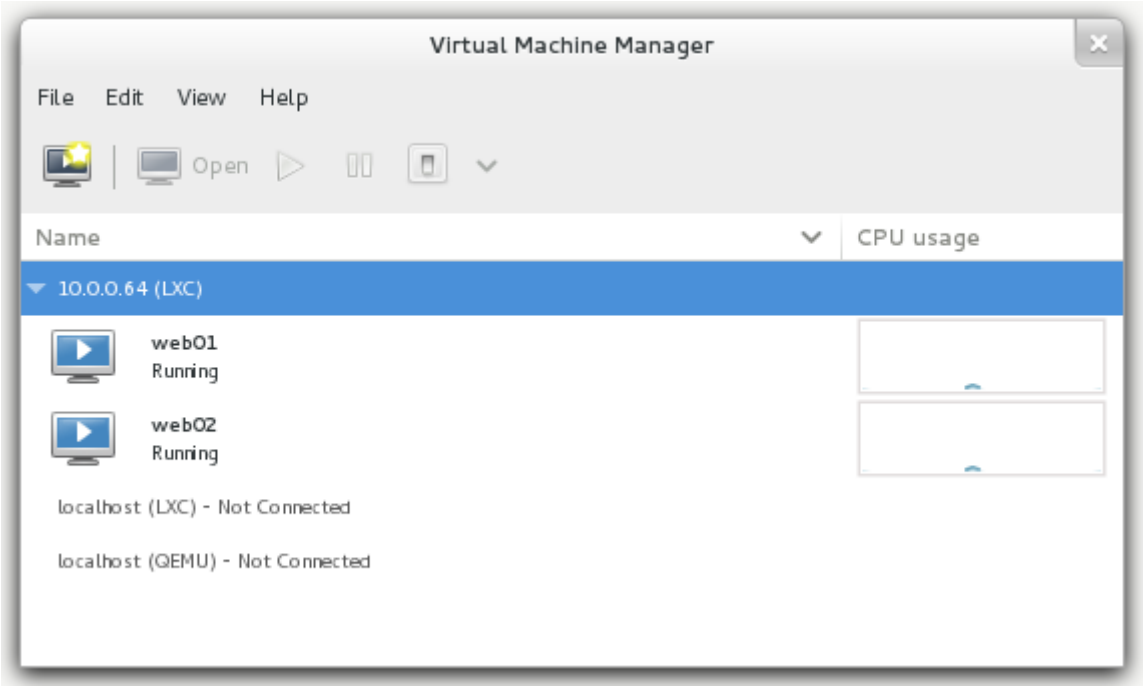


Fig. 6.2: LX containers in Virtual Machine Manager

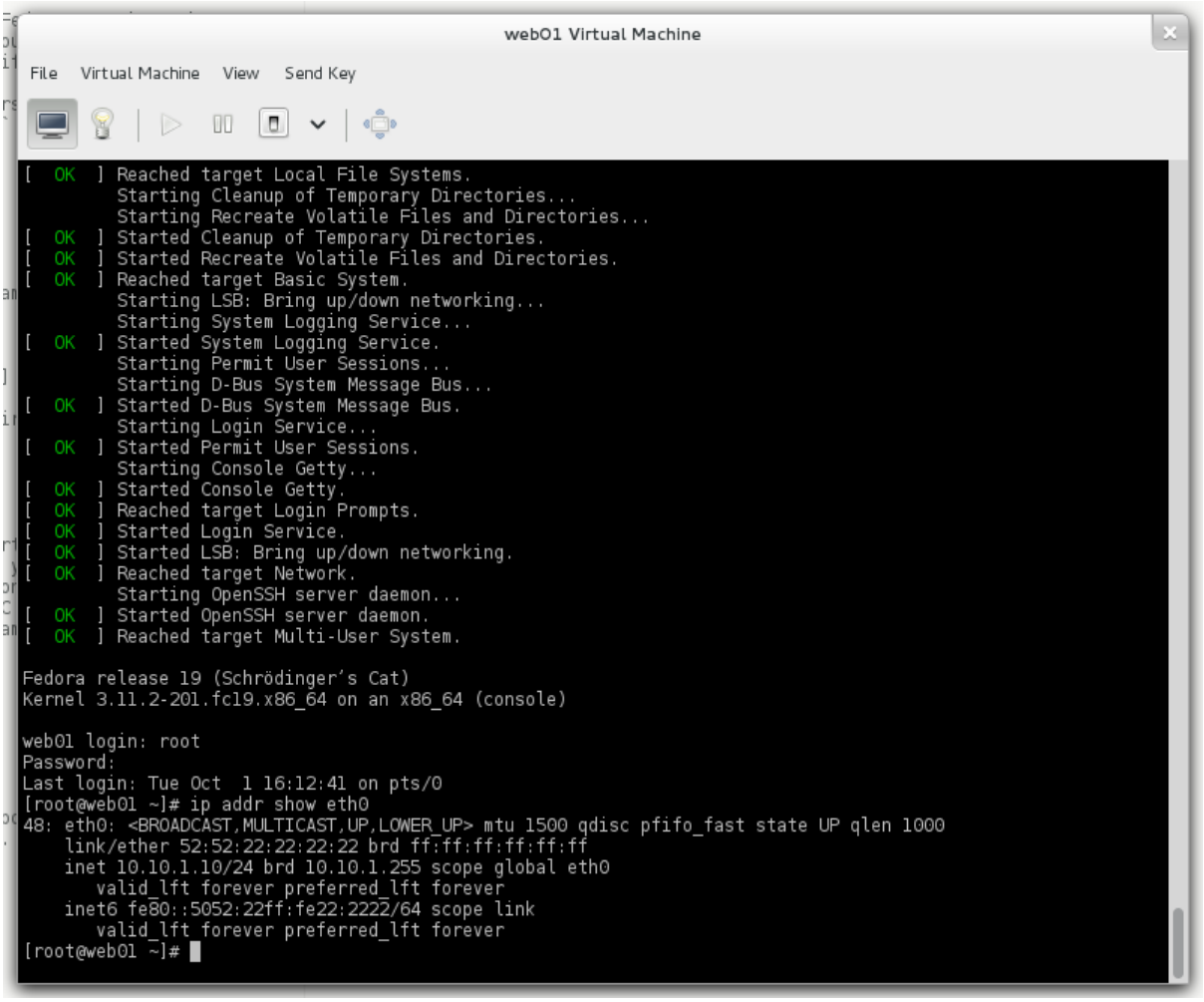


Fig. 6.3: Shell of the `web01` container in Virtual Machine Manager

A fast nmap scan shows the details about the honeypots:

```
$ sudo nmap -sVT 10.0.0.133 10.0.0.134 10.0.0.135

Starting Nmap 6.25 ( http://nmap.org ) at 2013-04-24 23:26 CEST
Nmap scan report for 10.0.0.133
Host is up (0.022s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE       VERSION
135/tcp open  msrpc?
139/tcp open  netbios-ssn?
445/tcp open  microsoft-ds?

Nmap scan report for 10.0.0.134
Host is up (0.016s latency).
Not shown: 996 closed ports
PORT    STATE SERVICE       VERSION
80/tcp  open  http?
135/tcp open  msrpc?
139/tcp open  netbios-ssn?
445/tcp open  microsoft-ds?

Nmap scan report for 10.0.0.135
Host is up (0.015s latency).
Not shown: 994 closed ports
PORT    STATE SERVICE    VERSION
21/tcp  open  tcpwrapped
22/tcp  open  tcpwrapped
23/tcp  open  tcpwrapped
25/tcp  open  smtp       Sendmail 8.12.2/8.12.2/SuSE
110/tcp open  tcpwrapped
143/tcp open  tcpwrapped
Service Info: Host: test-bench.; OS: Unix

Nmap done: 3 IP addresses (3 hosts up) scanned in 163.53 seconds
```

CHAPTER 7

Miscellaneous

This section contains various pieces of documention which doesn't fit in any other section.

Setup the Fedora Security Lab

The setup of the Fedora Security Lab can be done by several ways.

Live media

There are two different Live images available of the Fedora Security Lab. Those images can be used to create physical CDs or Live USB key.

- Download the 64-bit PC Edition: Fedora 20 x86_64 Live Security
- Download the 32-bit PC Edition: Fedora 20 i686 Live Security

For further information please check the Making Media section in the Fedora Installation Guide.

comps Package group

> **Warning:** This work only on Fedora 19 and beyond.

You have a default Fedora installation and want all Fedora Security Lab packages installed, you can use the *groupinstall* feature of `yum`.

```
$ sudo yum groupinstall security-lab
```

Ansible playbook

The *fsl.yml* playbook contains all packages which are included in the Fedora Security Lab.

Add all your hosts to `/etc/ansible/hosts` to the **[fsl_hosts]** group. Then run the playbook.

```
$ sudo ansible-playbook fsl.yml -f 10
```

Contribute

There are several ways users can contribute to the Fedora Security Lab Test bench project.

Development

Most parts of the Fedora Security Lab Test bench are Ansible playbooks. There are some helper script in Bash and Python. The web interface is made with simple HTML files.

Git setup

First install git on your system.

```
$ yum -y install git
```

After installing git, identify yourself to git with your name and your email address.

```
$ git config --global user.name "Your name"
$ git config --global user.email "your.name@example.com"
```

Github provides tools for collaboration including a way to easy fork existing repositories. Go to the Fedora Security Lab Test Bench git repository and fork it. For more detail please refer to the Fork A Repo page of Github. Clone your fork:

```
$ git clone git@github.com:your_github_username/fsl-test-bench.git
```

At the moment there is no connection to the upstream repository. You need to add another remote named "upstream" which points to the upstream repository.

```
$ cd fsl-test-bench
$ git remote add upstream git@github.com:fabaff/fsl-test-bench.git
```

Make changes, add new features, write documentation, or fix typos. Then commit all changes.

```
$ git add new_file
$ git commit new_file -m "This is a new file"
$ git push origin master
```

If you are done, send a **pull request**.

Don't forget to pull-in changes from the upstream repository from time to time as described in the Fork A Repo document.

```
$ git pull --rebase upstream master
```

Layout git repository

The all source files are located in the *'docs'*_ folder. All file are written with the reStructuredText syntax. The structure of the *'docs'*_ folder devide the content in various sub-folders those folder represent sections in the documentation

```
.
- docs ----------------- Documentation
- files ---------------- Template files
|   - kickstart -------- Kickstart files for the installation
- handlers ------------- Handlers for services
- tasks ---------------- A collection of tasks
|   - apps ------------- Vulnerable web applications
|   - cgi -------------- Common Gateway Interface (CGI)
|   - db-servers-------- Databases (DBMS)
|   - directory-servers- Directory servers
|   - file-servers ----- File servers
|   - ftp-servers ------ FTP servers
|   - helpers ---------- Helper tools
|   - honeypots -------- Low-interaction honeypots
|   - mail-servers ----- Mail server (SMTP, IMAP, POP)
|   - misc-servers------ Various servers (SSH, VPN, etc.)
|   - shells ----------- PHP shells
|   - web-servers------- Web servers
- variables ------------ Storage files for variables
```

Template files

All templates are located in *files* and are using the Jinja2 engine. This means that you can placed The example below shows a section of the *motd* file.

```
Hostname    : {{ ansible_hostname }}
System type : {{ ansible_system }}
Kernel      : {{ ansible_kernel }}
```

A nice way to check what variables are available is:

```
$ sudo ansible -m setup [IP of a managed host]
```

Handlers

The so-called handlers can be used as shortcut for managing services, like stop, start, and restart. If a new service is included, the handlers should be present. At the moment the playbook doesn't make heavy use of handlers.

Tasks

This folder contains all playbooks. If the playbook fits into an existing category it's placed in the corresponding sub-folder.

Variables

The files in *variables* contains values which can be accessed from other playbooks. This make it possible to reuse certain values over different plays.

Writing playbooks

Playbooks are written in YAML which makes the playbook very easy to read and to write. Ansible provides a bunch of modules for various operation. Those modules are used in the playbooks. Please read the playbooks section in the Ansible documentation to familar with the concept of the playbooks.

There is a template playbook online which acts as a starting point.

To have a basic level of modularization, one playbook should only include tasks for one application or tool and then included in another playbook:

```
tasks:
  - include: tasks/libvirt.yml
  - include: tasks/virt-install.yml
```

Please make sure that you use the full path when invoking commands `command: /usr/bin/mv` instead just `command: mv`.

Documentation

The documention source files are located in the *docs* folder. All files are written with the reStructuredText syntax. The structure of the *docs* folder devide the content in various sub-folders those folder represent sections in the documentation

```
.
|-- appendix --------- The documentation's appendix
|-- applications ----- Application section
|-- base ------------- Base information about the FSL Test bench
|-- _build ----------- Sphinx folder (the generated documentation)
|-- conf.py ---------- Configuration file for Sphinx
|-- images ----------- Screenshots
|-- index.rst -------- Default file for the documentation
|-- installation ----- Installation section
|-- intro ------------ Introduction section
|-- Makefile --------- Makefile for building the documentation locally
|-- misc ------------- This section contains various topics
|-- requirements.txt - This file is needed by Read the Docs
|-- services --------- Section with details about the available services
|-- _static ---------- Sphinx folder
`-- _templates ------- Sphinx folder
```

The documentation uses the sphinx-bootstrap-theme as theme. This theme is not available in the official Sphinx package. Because of this a separate installation is needed.

```
$ sudo yum -y install python-pip
$ sudo pip-python install sphinx_bootstrap_theme
```

For the generation of diagramms on-the-fly, the documentation uses blockdiag and nwdiag

```
$ sudo easy_install sphinxcontrib-blockdiag
$ sudo easy_install sphinxcontrib-nwdiag
```

For building the documentation locally, you need Sphinx.

```
$ sudo yum -y install python-sphinx python-docutils
```

If you want to build the documentation, switch to the *'docs'*_ folder and use *make* to build it.

```
$ cd docs
$ make html
```

The latest Documentation is always available at Read the Docs. After commit the changes to the git repository, Read the Docs rebuild the complete documentation.

Bugs and Improvements

Please report all wishes, bugs, improvements, or ideas. Depending on your preferences please use one of the systems mentionend below.

- Fedora Security Lab ticketing system
- Github issues

Web interface

The web interface is based on Twitter's bootstrap front-end framework. The *website.yml* playbook is delivering Jinja2 template pages.

The *file/website* folder contains the all template files which will be rendered as html during the setup process. The most importent files are:

- **about.j2** : This file contain further details about the Test bench.
- **contact.j2** : This file provides contact details and links to additional resources.
- **index.j2** : The index.html file shows all available application on the Test bench and gives the user easy access to those tools.

FAQ

Is the Fedora Security Lab Test bench a Live CD?

No, it's a default Fedora installation which is configured with the help of Ansible. For setup a system like the Fedora Security Lab Test bench some file modifications are needed. Live CDs don't allow to ship modified content or files.

Why not make RPMS of all parts?

Because the Package Review process in Fedora is slow (even for simple packages it takes months and I wanted to have the FSL Test bench now). Most software core parts are coming out of the Fedora Package Collection. The surrounding items and the configuration are installed by Ansible out of their upstream sources.

Will the Fedora Security Lab Test bench become a Live CD one day?

I don't think so, but never say never. We are customizing configuration files. Those configuration files are modified during the setup process to match the provided environment. The web interface is dynamically generated according your choises on the fly. This is not possible with a Spin.

Do you provide a VM or something similar?

No, because one big issue is trust. Providing a VM is like shipping a blackbox. You have to trust us about what's inside the VM. By using Ansible's playbooks you can see what steps are taken to setup the FSL Test bench. You are in control of every setup step, nothing is hidden and everything is transparent. The core components are installed out of the Fedora Package Collection on top of a minimal Fedora installation. This ensure that the operating system runs the latest packages and behave with integrity.

How should the network around the FSL Test bench looks like?

As mentioned on the setup page a DNS/DHCP server is a requirement. For security purposes we suggest that you use a dedicated network for setup your FSL Test bench.

Is internet access needed for the Fedora Security Lab Test bench?

For the setup access to the internet of the system which will host the FSL Test bench is needed. When the setup is finished, you can shutdown the internet access. DO NOT expose the FSL Test bench to the internet. Bad things could happen. You have been warned.

Can I use the FSL Test bench repository to setup a Fedora Security Lab?

Yes, you can. Periodically the fsl-packages.yml playbook get synced. This ways you don't need to clone the Fedora Security Lab git repository to install a Fedora Security Lab host.

Why are you still using yum and not dnf?

First `dnf` was used, then we switched back to `yum`. Soon we will switch to `dnf` again because DNF will become the next default Package manager for Fedora. And as always we wanted to be ahead of the rest of the world.

Do you have some reference installations?

No. This is project is a proof of concept only at the moment.

How long does it take from Zero to go?

Creating a libvirt-based virtual machine and using Ansible to configure it, takes something between 25 and 30 minutes. It heavily depends on your hardware and the speed of your internet connection.

Is The Fedora Security Lab Test Bench vulnerable for Heartbleed?

It depends on the point in time when you created your Test bench. Check if you are running at least with `openssl-1.0.1e-37.fc20`.

Is this something similar to Fedora Formulas?

Yes, it is. Basically we skipped the discussions and just made an implementation which we think is feasible for our needs.

Why is this project not hosted on Fedorahosted.org?

Because github offers easy access to the git repositories for everyone not only Fedora contributors. To get as many contributions as possible we need to be as open as possible. The Fedora Security Lab is hosted on Fedorahosted.org and we don't plan to change that.

Can I contribute?

Sure, contributions are appriciated. Please for the Fedora Security Lab Test bench repository and when you are done, open a **Pull request**.

Can I include item X?

Please follow the Fedora Project guidelines in this matter. The Forbidden items page in the Fedora Wiki is a good starting place. The item to include must be under an open source license, not proprietary, and not violate laws.

Testing

This section describe some basic steps to check if the Fedora Security Lab Test bench is properly setup and working.

Ansible

A simple test to check if Ansible is ready to work. Execute the command mentined below from the management system.

```
$ sudo ansible [IP address of the FSL Test bench] -m setup
```

If you get an authentication failure like the one shown below.

```
10.0.0.64 | FAILED => FAILED: Authentication failed.
```

Means this that the SSH key is not present in the `authorized_keys` file of your future Fedora Security Lab Test bench.

From the managed node:

```
ssh root@[IP address of your management system] 'cat ~/.ssh/id_rsa.pub' | cat - >>␣
↪~/.ssh/authorized_keys
```

From the management system:

```
$ sudo ssh-copy-id -i /root/.ssh/id_rsa.pub root@[IP address of your managed note]
```

Assuming that you already have an SSH key on your server.

```
$ sudo ssh-keygen -t rsa
```

Logging

The system log (aka `/var/log/messages`) is viewable on the web interface of the Fedora Security Lab Test bench. To check if the web interface is working proberly, send a message to the logging system.

Open two terminals and connect over SSH with the Fedora Security Lab Test bench. In one terminal execute the command from below to display the lastest log entries:

```
$ sudo journalctl -f
```

In the second terminal, send a message:

```
$ logger This is a test entry. To test the FSL Test bench log viewer.
```

The entry from above should now be visible in your browser's textarea of the System Log (http://{[}IP address of the FSL test bench]/log-system/) page.

Licenses

This section is to track the licenses of all parts which are not coming out of the Fedora Package Collection and didn't get a proper license check during a review process. To be as much as possible as Fedora the included tools should follow the Fedora Licensing policies.

Vulnerable web applications

Name	License
Bricks	unknown
bWAPP	CC BY-NC-ND 4.0
DVWA	GPL3+
hackademic	GPL2+
MCIR	GPL3+
SQLI Labs	unknown
XSSeducation	unknown

Shells

Name	License
ajaxshell	unknown
ani-shell	unknown
b374k	MIT
DNA Shell	GPL2+
escobar	GPL2+
PHP Shell	GPL2+
PHP Reverse Shell	GPL2
WSO	unknown

Helper tools

Name	License
linfo	GPL3+
phpLiteAdmin	GPL3+
phpMyAdmin	GPL2+
phpMOadmin	GPL3
phpLDAPadmin	GPL2+
PHP-Shell-Detector	MIT

CHAPTER 8

Appendix

The appendix contains additional details about the Test bench.

nmap

The `nmap` output below shows the view of the Test bench from the network side.

```
$ sudo nmap -sV --reason 10.0.0.64

Starting Nmap 6.45 ( http://nmap.org ) at 2014-10-28 09:44 CET
Nmap scan report for 10.0.0.64
Host is up, received arp-response (0.00060s latency).
Not shown: 983 filtered ports
Reason: 962 no-responses and 21 host-prohibiteds
PORT     STATE  SERVICE         REASON  VERSION
21/tcp   open   ftp             syn-ack vsftpd 3.0.2
22/tcp   open   ssh             syn-ack OpenSSH 6.4 (protocol 2.0)
23/tcp   open   telnet          syn-ack Linux telnetd
25/tcp   open   smtp            syn-ack Postfix smtpd
80/tcp   open   http            syn-ack lighttpd 1.4.35
110/tcp  open   pop3            syn-ack Dovecot pop3d
143/tcp  open   imap            syn-ack Dovecot imapd
222/tcp  open   ssh             syn-ack Dropbear sshd 2014.64 (protocol 2.0)
443/tcp  closed https           reset
631/tcp  closed ipp             reset
993/tcp  open   ssl/imap        syn-ack Dovecot imapd
995/tcp  open   ssl/pop3        syn-ack Dovecot pop3d
3389/tcp open   ms-wbt-server   syn-ack xrdp
8000/tcp open   http            syn-ack BaseHTTPServer 0.3 (Python 2.7.5)
8080/tcp open   http            syn-ack Apache Tomcat/Coyote JSP engine 1.1
8088/tcp closed radan-http      reset
8888/tcp open   sun-answerbook? syn-ack
1 service unrecognized despite returning data. If you know the service/version,␣
↪please submit the following fingerprint at http://www.insecure.org/cgi-bin/
↪servicefp-submit.cgi :
SF-Port8888-TCP:V=6.45%I=7%D=10/28%Time=544F577A%P=x86_64-redhat-linux-gnu
SF:%r(GetRequest,4EB,"HTTP/1\.1\x20200\x20OK\r\nserver:\x20ecstatic-0\.4\.
SF:13\r\netag:\x20\"11166-963-Sun\x20Jun\x2015\x202014\x2021:59:16\x20GMT\
```

```
SF:+0200\x20\(CEST\)\"\r\nlast-modified:\x20Sun,\x2015\x20Jun\x202014\x201
SF:9:59:16\x20GMT\r\ncache-control:\x20max-age=3600\r\ncontent-length:\x20
SF:963\r\ncontent-type:\x20text/html;\x20charset=UTF-8\r\nDate:\x20Tue,\x2
SF:028\x20Oct\x202014\x2008:44:42\x20GMT\r\nConnection:\x20close\r\n\r\n<!
SF:DOCTYPE\x20HTML\x20PUBLIC\x20\"-//W3C//DTD\x20HTML\x204\.01//EN\">\n<!-
SF:-\nThis\x20file\x20is\x20part\x20of\x20the\x20Fedora\x20Security\x20Lab
SF:\x20Test\x20bench\x20and\x20distributed\x20with\nAnsible\.\n\nCopyright
SF:\x20\(c\)\x202013-2014\x20Fabian\x20Affolter\x20<fab@fedoraproject\.org
SF:>\n\nThe\x20FSL\x20Test\x20bench\x20is\x20licensed\x20under\x20GPLv2\.\
SF:n-->\n<html\x20lang=\"en\">\n<head>\n\x20\x20\x20\x20<meta\x20charset=\
SF:"utf-8\">\n\x20\x20\x20\x20<title>Fedora\x20Security\x20Lab\x20Test\x20
SF:bench\x20\|\x20Webserver\x20is\x20up\x20and\x20running\.\.\.</title>\n\
SF:x20\x20\x20\x20<meta\x20name=\"viewport\"\x20content=\"width=device-wid
SF:th,\x20initial-scale=1\.0\">\n\x20\x20\x20\x20<meta\x20name=\"descripti
SF:on\"\x20content=\"Penetration\x20testing\x20system\">\n\x20\x20\x20\x20
SF:<meta\x20name=\"author\"\x20content=\"Joerg\x20Simon,\x20Fabian\x20Affo
SF:lter\">\n</head>\n")%r(HTTPOptions,8F,"HTTP/1\.1\x20404\x20Not\x20Found
SF:\r\nserver:\x20ecstatic-0\.4\.13\r\nContent-Type:\x20text/plain\r\nDate
SF::\x20Tue,\x2028\x20Oct\x202014\x2008:44:42\x20GMT\r\nConnection:\x20clo
SF:se\r\n\r\nNot\x20found\n")%r(FourOhFourRequest,8F,"HTTP/1\.1\x20404\x20
SF:Not\x20Found\r\nserver:\x20ecstatic-0\.4\.13\r\nContent-Type:\x20text/p
SF:lain\r\nDate:\x20Tue,\x2028\x20Oct\x202014\x2008:44:42\x20GMT\r\nConnec
SF:tion:\x20close\r\n\r\nNot\x20found\n");
MAC Address: 52:52:00:00:00:01 (Unknown)
Service Info: Host:  testbench01.localdomain; OSs: Unix, Linux; CPE: cpe:/
↪o:linux:linux_kernel

Service detection performed. Please report any incorrect results at http://nmap.
↪org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 53.28 seconds
```

masscan

The nmap output below shows the view of the Test bench from the network side.

```
$ sudo masscan -p0-65535 10.0.0.64

Starting masscan 1.0.3 (http://bit.ly/14GZzcT) at 2014-10-28 08:45:22 GMT
 -- forced options: -sS -Pn -n --randomize-hosts -v --send-eth
Initiating SYN Stealth Scan
Scanning 1 hosts [65536 ports/host]
Discovered open port 23/tcp on 10.0.0.64
Discovered open port 25/tcp on 10.0.0.64
Discovered open port 8880/tcp on 10.0.0.64
Discovered open port 21/tcp on 10.0.0.64
Discovered open port 8000/tcp on 10.0.0.64
Discovered open port 8889/tcp on 10.0.0.64
Discovered open port 8887/tcp on 10.0.0.64
Discovered open port 3389/tcp on 10.0.0.64
Discovered open port 993/tcp on 10.0.0.64
Discovered open port 22/tcp on 10.0.0.64
Discovered open port 27017/tcp on 10.0.0.64
Discovered open port 222/tcp on 10.0.0.64
Discovered open port 8888/tcp on 10.0.0.64
Discovered open port 80/tcp on 10.0.0.64
Discovered open port 110/tcp on 10.0.0.64
Discovered open port 995/tcp on 10.0.0.64
Discovered open port 8080/tcp on 10.0.0.64
```

arachni

The `arachni` output below shows the view of the Test bench from the network side.:

```
Coming soon...
```

lynis

The lynis output is from a virtual instance of a FSL Test bench.

```
[root@test-bench ~]# lynis --auditor "FSL Test bench" --check-all

[ Lynis 1.5.0 ]

################################################################################
 Lynis comes with ABSOLUTELY NO WARRANTY. This is free software, and you are
 welcome to redistribute it under the terms of the GNU General Public License.
 See the LICENSE file for details about using this software.

 Copyright 2007-2014 - Michael Boelen, http://cisofy.com
 Enterprise support and plugins available via CISOfy - http://cisofy.com
################################################################################

[+] Initializing program
------------------------------------
  - Detecting OS...                                           [ DONE ]
  - Clearing log file (/var/log/lynis.log)...                 [ DONE ]

  ---------------------------------------------------
  Program version:           1.5.0
  Operating system:          Linux
  Operating system name:     Fedora
  Operating system version:  Fedora release 20 (Heisenbug)
  Kernel version:            3.13.9-200.fc20.x86_64
  Hardware platform:         x86_64
  Hostname:                  test-bench
  Auditor:                   FSL Test bench
  Profile:                   /etc/lynis/default.prf
  Log file:                  /var/log/lynis.log
  Report file:               /var/log/lynis-report.dat
  Report version:            1.0
  Plugin directory:          /usr/share/lynis/plugins
  ---------------------------------------------------

[ Press [ENTER] to continue, or [CTRL]+C to stop ]

  - Checking profile file (/etc/lynis/default.prf)...
  - Program update status...                                  [ SKIPPED ]

[+] System Tools
------------------------------------
  - Scanning available tools...
  - Checking system binaries...
    - Checking /bin...                                        [ FOUND ]
    - Checking /sbin...                                       [ FOUND ]
    - Checking /usr/bin...                                    [ FOUND ]
    - Checking /usr/sbin...                                   [ FOUND ]
    - Checking /usr/local/bin...                              [ FOUND ]
    - Checking /usr/local/sbin...                             [ FOUND ]
    - Checking /usr/local/libexec...                          [ FOUND ]
```

```
    - Checking /usr/libexec...                                  [ FOUND ]
    - Checking /usr/sfw/bin...                                  [ NOT FOUND ]
    - Checking /usr/sfw/sbin...                                 [ NOT FOUND ]
    - Checking /usr/sfw/libexec...                              [ NOT FOUND ]
    - Checking /opt/sfw/bin...                                  [ NOT FOUND ]
    - Checking /opt/sfw/sbin...                                 [ NOT FOUND ]
    - Checking /opt/sfw/libexec...                              [ NOT FOUND ]
    - Checking /usr/xpg4/bin...                                 [ NOT FOUND ]
    - Checking /usr/css/bin...                                  [ NOT FOUND ]
    - Checking /usr/ucb...                                      [ NOT FOUND ]
    - Checking /usr/X11R6/bin...                                [ NOT FOUND ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Plugins (phase 1)
------------------------------------
  - Plugins enabled                                             [ NONE ]

[+] Boot and services
------------------------------------
  - Checking boot loaders
    - Checking presence GRUB...                                 [ NOT FOUND ]
    - Checking presence LILO...                                 [ NOT FOUND ]
    - Checking boot loader SILO                                 [ NOT FOUND ]
    - Checking boot loader YABOOT                               [ NOT FOUND ]
  - Check running services (systemctl)...                       [ DONE ]
        Result: found 31 running services
  - Check enabled services at boot (systemctl)...               [ DONE ]
        Result: found 35 enabled services
  - Check startup files (permissions)...                        [ OK ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Kernel
------------------------------------
  - Checking default runlevel...                                [ runlevel 3 ]
  - Checking CPU support (NX/PAE)
    CPU support: PAE and/or NoeXecute supported                 [ FOUND ]
  - Checking kernel version and release                         [ DONE ]
  - Checking kernel type                                        [ DONE ]
  - Checking loaded kernel modules                              [ DONE ]
      Found 61 active modules
  - Checking Linux kernel configuration file                    [ FOUND ]
  - Checking default I/O kernel scheduler                       [ FOUND ]
  - Checking core dumps configuration...                        [ DISABLED ]
    - Checking setuid core dumps configuration...               [ DEFAULT ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Memory and processes
------------------------------------
  - Checking /proc/meminfo...                                   [ FOUND ]
  - Searching for dead/zombie processes...                      [ OK ]
  - Searching for IO waiting processes...                       [ OK ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Users, Groups and Authentication
------------------------------------
```

```
  - Search administrator accounts...                          [ OK ]
  - Checking for non-unique UIDs...                           [ OK ]
  - Checking consistency of group files (grpck)...            [ OK ]
  - Checking non unique group ID's...                         [ OK ]
  - Checking non unique group names...                        [ OK ]
  - Checking password file consistency...                     [ OK ]
  - Query system users (non daemons)...                       [ DONE ]
  - Checking NIS+ authentication support                      [ NOT ENABLED ]
  - Checking NIS authentication support                       [ NOT ENABLED ]
  - Checking sudoers file                                     [ FOUND ]
    - Check sudoers file permissions                          [ OK ]
  - Checking PAM password strength tools                      [ OK ]
  - Checking PAM configuration file (pam.conf)                [ NOT FOUND ]
  - Checking PAM configuration files (pam.d)                  [ FOUND ]
  - Checking PAM modules                                      [ FOUND ]
  - Checking user password aging                              [ DISABLED ]
  - Checking Linux single user mode authentication            [ WARNING ]
  - Determining default umask
    - Checking umask (/etc/profile)                           [ UNKNOWN ]
    - Checking umask (/etc/login.defs)                        [ OK ]
    - Checking umask (/etc/init.d/functions)                  [ SUGGESTION ]
  - Checking LDAP authentication support                      [ NOT ENABLED ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Shells
------------------------------------
  - Checking shells from /etc/shells...
    Result: found 6 shells (valid shells: 6).

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] File systems
------------------------------------
  - Checking mount points
    - Checking /home mount point...                           [ SUGGESTION ]
    - Checking /tmp mount point...                            [ OK ]
  - Checking LVM volume groups...                             [ FOUND ]
    - Checking LVM volumes...                                 [ FOUND ]
  - Checking for old files in /tmp...                         [ OK ]
  - Checking /tmp sticky bit...                               [ OK ]
  - ACL support root file system...                           [ ENABLED ]
  - Checking Locate database...                               [ NOT FOUND ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Storage
------------------------------------
  - Checking usb-storage driver (modprobe config)...          [ NOT DISABLED ]
egrep: /etc/modprobe.d/*: No such file or directory
egrep: /etc/modprobe.d/*: No such file or directory
  - Checking firewire ohci driver (modprobe config)...        [ NOT DISABLED ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] NFS
------------------------------------
  - Query rpc registered programs...                          [ DONE ]
  - Query NFS versions...                                     [ DONE ]
```

```
  - Query NFS protocols...                                  [ DONE ]
  - Check running NFS daemon...                             [ NOT FOUND ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Software: name services
------------------------------------
  - Checking default DNS search domain...                   [ NONE ]
  - Checking /etc/resolv.conf options...                    [ NONE ]
  - Searching DNS domain name...                            [ UNKNOWN ]
  - Checking nscd status...                                 [ NOT FOUND ]
  - Checking BIND status...                                 [ NOT FOUND ]
  - Checking PowerDNS status...                             [ NOT FOUND ]
  - Checking ypbind status...                               [ NOT FOUND ]
  - Checking /etc/hosts
    - Checking /etc/hosts (duplicates)                      [ OK ]
    - Checking /etc/hosts (hostname)                        [ OK ]
    - Checking /etc/hosts (localhost)                       [ SUGGESTION ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Ports and packages
------------------------------------
  - Searching package managers...
    - Searching RPM package manager...                      [ FOUND ]
      - Querying RPM package manager...
  - Checking YUM package management consistency             [ OK ]
  - Checking package database duplicates...                 [ OK ]
  - Checking package database for problems...               [ OK ]
  - Checking missing security packages                      [ SKIPPED ]
  - Checking GPG checks (yum.conf)                          [ DISABLED ]
  - Checking package audit tool...                          [ NONE ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Networking
------------------------------------
  - Checking configured nameservers...
    - Testing nameservers...
      Nameserver: 10.1.1.1...                               [ SKIPPED ]
    - Minimal of 2 responsive nameservers...                [ SKIPPED ]
  - Checking default gateway...                             [ DONE ]
  - Getting listening ports (TCP/TCP)...                    [ DONE ]
      * Found 16 ports
  - Checking promiscuous interfaces...                      [ OK ]
  - Checking waiting connections...                         [ OK ]
  - Checking status DHCP client...                          [ RUNNING ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Printers and Spools
------------------------------------
  - Checking cups daemon...                                 [ NOT FOUND ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Software: e-mail and messaging
------------------------------------
```

```
  - Checking Exim status...                                   [ NOT FOUND ]
  - Checking Postfix status...                                [ RUNNING ]
  - Checking Postfix configuration...                         [ FOUND ]
    - Checking Postfix banner...                              [ WARNING ]
  - Checking Dovecot status...                                [ RUNNING ]
  - Checking Qmail smtpd status...                            [ NOT FOUND ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Software: firewalls
------------------------------------
  - Checking iptables kernel module                           [ NOT FOUND ]
  - Checking iptables in config file                          [ FOUND ]
    - Checking for empty ruleset                              [ OK ]
    - Checking for unused rules                               [ WARNING ]
    Status pf                                                 [ NOT FOUND ]
  - Checking host based firewall                              [ ACTIVE ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Software: webserver
------------------------------------
  - Checking Apache...                                        [ NOT FOUND ]
  - Checking nginx...                                         [ NOT FOUND ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] SSH Support
------------------------------------
  - Checking running SSH daemon...                            [ FOUND ]
    - Searching SSH configuration...                          [ FOUND ]
    - Checking defined SSH options...                         [ DONE ]
    - SSH option: PermitRootLogin...                          [ DEFAULT ]
    - SSH option: Protocol...                                 [ DEFAULT ]
    - SSH option: StrictModes...                              [ DEFAULT ]
    - SSH option: AllowUsers...                               [ NOT FOUND ]
    - SSH option: AllowGroups...                              [ NOT FOUND ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] SNMP Support
------------------------------------
  - Checking running SNMP daemon...                           [ FOUND ]
    - Checking SNMP configuration...                          [ FOUND ]
  - Checking SNMP community strings...                        [ WARNING ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Databases
------------------------------------
  - MySQL process status...                                   [ NOT FOUND ]
  - PostgreSQL processes status...                            [ NOT FOUND ]
  - Oracle processes status...                                [ NOT FOUND ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] LDAP Services
```

```
------------------------------------
  - Checking OpenLDAP instance...                           [ NOT FOUND ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Software: PHP
------------------------------------
  - Checking PHP...                                         [ FOUND ]
    - Checking PHP disabled functions...                    [ NONE ]
    - Checking register_globals option...                   [ OK ]
    - Checking expose_php option...                         [ ON ]
    - Checking enable_dl option...                          [ OFF ]
    - Checking allow_url_fopen option...                    [ ON ]
    - Checking allow_url_include option...                  [ OFF ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Squid Support
------------------------------------
  - Checking running Squid daemon...                        [ NOT FOUND ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Logging and files
------------------------------------
  - Checking for a running log daemon...                    [ OK ]
    - Checking Syslog-NG status                             [ NOT FOUND ]
    - Checking Metalog status                               [ NOT FOUND ]
    - Checking RSyslog status                               [ NOT FOUND ]
    - Checking RFC 3195 daemon status                       [ NOT FOUND ]
    - Checking klogd                                        [ NOT FOUND ]
    - Checking minilogd instances                           [ NOT FOUND ]
  - Checking logrotate presence                             [ OK ]
  - Checking log directories (static list)                  [ DONE ]
  - Checking open log files                                 [ SKIPPED ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Insecure services
------------------------------------
  - Checking inetd status...                                [ ACTIVE ]
    - Checking inetd.conf...                                [ NOT FOUND ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Banners and identification
------------------------------------
  - /etc/motd...                                            [ FOUND ]
    - /etc/motd permissions...                              [ OK ]
    - /etc/motd contents...                                 [ WEAK ]
  - /etc/issue...                                           [ FOUND ]
    - /etc/issue contents...                                [ WEAK ]
  - /etc/issue.net...                                       [ FOUND ]
    - /etc/issue.net contents...                            [ WEAK ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]
```

```
[+] Scheduled tasks
------------------------------------
  - Checking crontab/cronjob                                  [ DONE ]
  - Checking atd status                                       [ NOT RUNNING ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Accounting
------------------------------------
  - Checking accounting information...                        [ NOT FOUND ]
  - Checking sysstat accounting data                          [ NOT FOUND ]
  - Checking auditd                                           [ ENABLED ]
    - Checking audit rules                                    [ SUGGESTION ]
    - Checking audit configuration file                       [ OK ]
    - Checking auditd log file                                [ FOUND ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Time and Synchronization
------------------------------------
  - Checking running NTP daemon (ntpd)...                     [ NOT FOUND ]
  - Checking running NTP daemon (timed)...                    [ NOT FOUND ]
  - Checking running NTP daemon (dntpd)...                    [ NOT FOUND ]
  - Checking NTP client in crontab file (/etc/anacrontab)...  [ NOT FOUND ]
  - Checking NTP client in crontab file (/etc/crontab)...     [ NOT FOUND ]
  - Checking NTP client in cron.d files...                    [ NOT FOUND ]
  - Checking for a running NTP daemon or client...            [ WARNING ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Cryptography
------------------------------------
  - Checking SSL certificate expiration...                    [ OK ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Virtualization
------------------------------------

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Security frameworks
------------------------------------
  - Checking presence AppArmor                                [ NOT FOUND ]
  - Checking presence SELinux                                 [ FOUND ]
    - Checking SELinux status                                 [ ENABLED ]
      - Checking current mode and config file                 [ OK ]
        Current SELinux mode: enforcing
  - Checking presence grsecurity                              [ NOT FOUND ]
  - Checking for implemented MAC framework                    [ OK ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Software: file integrity
------------------------------------
  - Checking file integrity tools...
    - AFICK...                                                [ NOT FOUND ]
```

```
    - AIDE...                                                [ NOT FOUND ]
    - Osiris...                                              [ NOT FOUND ]
    - Samhain...                                             [ NOT FOUND ]
    - Tripwire...                                            [ NOT FOUND ]
    - OSSEC (syscheck)...                                    [ NOT FOUND ]
  - Checking presence integrity tool...                      [ NOT FOUND ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Software: Malware scanners
------------------------------------
  - Checking chkrootkit...                                   [ NOT FOUND ]
  - Checking Rootkit Hunter...                               [ NOT FOUND ]
  - Checking ClamAV scanner...                               [ NOT FOUND ]
  - Checking ClamAV daemon...                                [ NOT FOUND ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] System Tools
------------------------------------
  - Starting file permissions check...
    /etc/lilo.conf                                           [ NOT FOUND ]
    /root/.ssh                                               [ OK ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Home directories
------------------------------------
  - Checking shell history files...                          [ OK ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Kernel Hardening
------------------------------------
  - Comparing sysctl key pairs with scan profile...
    - kernel.core_uses_pid (exp: 1)                          [ OK ]
    - kernel.ctrl-alt-del (exp: 0)                           [ OK ]
    - kernel.sysrq (exp: 0)                                  [ DIFFERENT ]
    - net.ipv4.conf.all.accept_redirects (exp: 0)            [ DIFFERENT ]
    - net.ipv4.conf.all.accept_source_route (exp: 0)         [ OK ]
    - net.ipv4.conf.all.bootp_relay (exp: 0)                 [ OK ]
    - net.ipv4.conf.all.forwarding (exp: 0)                  [ OK ]
    - net.ipv4.conf.all.log_martians (exp: 1)                [ DIFFERENT ]
    - net.ipv4.conf.all.mc_forwarding (exp: 0)               [ OK ]
    - net.ipv4.conf.all.proxy_arp (exp: 0)                   [ OK ]
    - net.ipv4.conf.all.rp_filter (exp: 1)                   [ DIFFERENT ]
    - net.ipv4.conf.all.send_redirects (exp: 0)              [ DIFFERENT ]
    - net.ipv4.conf.default.accept_redirects (exp: 0)        [ DIFFERENT ]
    - net.ipv4.conf.default.accept_source_route (exp: 0)     [ OK ]
    - net.ipv4.conf.default.log_martians (exp: 1)            [ DIFFERENT ]
    - net.ipv4.icmp_echo_ignore_broadcasts (exp: 1)          [ OK ]
    - net.ipv4.icmp_ignore_bogus_error_responses (exp: 1)    [ OK ]
    - net.ipv4.tcp_syncookies (exp: 1)                       [ OK ]
    - net.ipv4.tcp_timestamps (exp: 0)                       [ DIFFERENT ]
    - net.ipv6.conf.all.accept_redirects (exp: 0)            [ DIFFERENT ]
    - net.ipv6.conf.all.accept_source_route (exp: 0)         [ OK ]
    - net.ipv6.conf.default.accept_redirects (exp: 0)        [ DIFFERENT ]
    - net.ipv6.conf.default.accept_source_route (exp: 0)     [ OK ]
```

```
[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Hardening
------------------------------------
    - Installed compiler(s)...                                [ FOUND ]
    - Installed malware scanner...                            [ NOT FOUND ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]


[+] Custom Tests
------------------------------------
  - Running custom tests...                                   [ NONE ]

================================================================================

  -[ Lynis 1.5.0 Results ]-

  Tests performed: 173   Plugins enabled: 0

  Warnings:
  ----------------------------
  - No password set for single mode [AUTH-9308]
      http://cisofy.com/controls/AUTH-9308/

  - No GPG signing option found in yum.conf [PKGS-7387]
      http://cisofy.com/controls/PKGS-7387/

  - Found mail_name in SMTP banner, and/or mail_name contains 'Postfix' [MAIL-8818]
      http://cisofy.com/controls/MAIL-8818/

  - Found easy guessable SNMP community string [SNMP-3306]
      http://cisofy.com/controls/SNMP-3306/

  - PHP option expose_php is possibly turned on, which can reveal useful␣
↪information for attackers. [PHP-2372]
      http://cisofy.com/controls/PHP-2372/

  - klogd is not running, which could lead to missing kernel messages in log files␣
↪[LOGG-2138]
      http://cisofy.com/controls/LOGG-2138/


  Suggestions:
  ----------------------------
  - Run systemctl --full --type=service to see all services
      http://cisofy.com/controls/[22:30:27 Suggestion: Run systemctl --full --
↪type=service to see all services/
  - Run systemctl list-unit-files --type=service to see all services
      http://cisofy.com/controls/[22:30:29 Suggestion: Run systemctl list-unit-
↪files --type=service to see all services/
  - Configure password aging limits to enforce password changing on a regular base␣
↪[AUTH-9286]
      http://cisofy.com/controls/AUTH-9286/
  - Set password for single user mode to minimize physical access attack surface␣
↪[AUTH-9308]
      http://cisofy.com/controls/AUTH-9308/
  - To decrease the impact of a full /home file system, place /home on a separated␣
↪partition [FILE-6310]
      http://cisofy.com/controls/FILE-6310/
  - The database required for 'locate' could not be found. Run 'updatedb' or
↪'locate.updatedb' to create this file. [FILE-6410]
```

```
      http://cisofy.com/controls/FILE-6410/
  - Disable drivers like USB storage when not used, to prevent unauthorized␣
↪storage or data theft [STRG-1840]
      http://cisofy.com/controls/STRG-1840/
  - Disable drivers like firewire storage when not used, to prevent unauthorized␣
↪storage or data theft [STRG-1846]
      http://cisofy.com/controls/STRG-1846/
  - Check DNS configuration [NAME-4028]
      http://cisofy.com/controls/NAME-4028/
  - Split resolving between localhost and the hostname of the system [NAME-4406]
      http://cisofy.com/controls/NAME-4406/
  - Install package yum-plugin-security if possible, to maintain security updates␣
↪easier (yum install yum-plugin-security) [PKGS-7386]
      http://cisofy.com/controls/PKGS-7386/
  - Install a package audit tool to determine vulnerable packages [PKGS-7398]
      http://cisofy.com/controls/PKGS-7398/
  - You are adviced to hide the mail_name (option: smtpd_banner) from your postfix␣
↪configuration. Use postconf -e or change your main.cf file (/etc/postfix/main.
↪cf) [MAIL-8818]
      http://cisofy.com/controls/MAIL-8818/
  - Check iptables rules to see which rules are currently not used [FIRE-4513]
      http://cisofy.com/controls/FIRE-4513/
  - Harden PHP by disabling risky functions [PHP-2320]
      http://cisofy.com/controls/PHP-2320/
  - Change the expose_php line to: expose_php = Off [PHP-2372]
      http://cisofy.com/controls/PHP-2372/
  - Change the allow_url_fopen line to: allow_url_fopen = Off, to disable␣
↪downloads via PHP [PHP-2376]
      http://cisofy.com/controls/PHP-2376/
  - Check why klogd is not running [LOGG-2138]
      http://cisofy.com/controls/LOGG-2138/
  - Add legal banner to /etc/motd, to warn unauthorized users [BANN-7122]
      http://cisofy.com/controls/BANN-7122/
  - Add a legal banner to /etc/issue, to warn unauthorized users [BANN-7126]
      http://cisofy.com/controls/BANN-7126/
  - Add legal banner to /etc/issue.net, to warn unauthorized users [BANN-7130]
      http://cisofy.com/controls/BANN-7130/
  - Enable sysstat to collect accounting (no results) [ACCT-9626]
      http://cisofy.com/controls/ACCT-9626/
  - Audit daemon is enabled with an empty ruleset. Disable the daemon or define␣
↪rules [ACCT-9630]
      http://cisofy.com/controls/ACCT-9630/
  - Use NTP daemon or NTP client to prevent time issues. [TIME-3104]
      http://cisofy.com/controls/TIME-3104/
  - Install a file integrity tool [FINT-4350]
      http://cisofy.com/controls/FINT-4350/
  - One or more sysctl values differ from the scan profile and could be tweaked␣
↪[KRNL-6000]
      http://cisofy.com/controls/KRNL-6000/
  - Harden the system by removing unneeded compilers. This can decrease the chance␣
↪of customized trojans, backdoors and rootkits to be compiled and installed [HRDN-
↪7220]
      http://cisofy.com/controls/HRDN-7220/
  - Harden compilers and restrict access to world [HRDN-7222]
      http://cisofy.com/controls/HRDN-7222/
  - Harden the system by installing one or malware scanners to perform periodic␣
↪file system scans [HRDN-7230]
      http://cisofy.com/controls/HRDN-7230/

  Follow-up:
  ------------------------------
  - Fix findings, see security controls overview and documentation
  - Upload data to Lynis Enterprise for further analysis
```

```
  - Create a report and implementation plan

  Enterprise support and plugins available via CISOfy - http://cisofy.com
================================================================================
  Hardening index : [60]   [############        ]
================================================================================
  Files:
  - Test and debug information      : /var/log/lynis.log
  - Report data                     : /var/log/lynis-report.dat
================================================================================
  Tip: Disable all tests which are not relevant or are too strict for the
       purpose of this particular machine. This will remove unwanted suggestions
       and also boost the hardening index. Each test should be properly analyzed
       to see if the related risks can be accepted, before disabling the test.
================================================================================
  Lynis 1.5.0
  Copyright 2007-2014 - Michael Boelen, http://cisofy.com
================================================================================
```

fsl-tb-detect

The `fsl-tb-detect` script makes it possible to check your network for Fedora Security Lab Test Bench Web interfaces which leads to the conclusion that the Fedora Security Lab Test bench is available on those systems. The script is pretty simple: It is looking for the string "Fedora Security Lab Test Bench" in the meta data of any html files provided by a web server.:

```
local http = require "http"
local shortport = require "shortport"
local string = require "string"

description = [[
Checks for the Fedora Security Lab Test Bench web interface.
]]

author = "Fabian Affolter"
license = "Same as Nmap--See http://nmap.org/book/man-legal.html"
categories = {"discovery", "safe"}

---
-- @usage
-- nmap --script fsl-detect <host>
--
--@output
-- Nmap scan report for testbench01.lab-ex.security (10.0.0.64)
-- PORT   STATE SERVICE
-- 80/tcp open  http
-- |_fsl-tb-detect: Fedora Security Lab Test bench Web interface FOUND.

-- Changelog:
-- 2013-05-09 Fabian Affolter  <fabian@affolter-engineering.ch>:
--   + initial release

portrule = shortport.http

action = function(host, port)
    local resp, title
    resp = http.get( host, port, '/' )
    title = string.match(resp.body, "<[Tt][Ii][Tt][Ll][Ee][^>]*>([^<]*)</
↪[Tt][Ii][Tt][Ll][Ee]>")
    if string.find(title, "Fedora Security Lab Test bench") then
```

```
        title = "Fedora Security Lab Test bench Web interface FOUND."
        else
        title = "Fedora Security Lab Test bench Web interface NOT found."
    end
    return title
end
```

Run the script against your network.:

```
$ sudo nmap --script=./fsl-tb-detect.nse 10.0.0.0/24

Starting Nmap 6.40 ( http://nmap.org ) at 2013-10-22 09:12 CEST

Nmap scan report for config01.lax-ex.network (10.0.0.30)
Host is up (0.0056s latency).
PORT   STATE SERVICE
80/tcp open  http
|_fsl-tb-detect: Fedora Security Lab Test bench Web interface NOT found.
MAC Address: 54:54:44:47:C6:78 (QEMU Virtual NIC)

Nmap scan report for testbench01.lab-ex.network (10.0.0.64)
Host is up (0.0048s latency).
PORT   STATE SERVICE
80/tcp open  http
|_fsl-tb-detect: Fedora Security Lab Test bench Web interface FOUND.
MAC Address: 54:54:44:21:14:01 (Unknown)

Nmap scan report for testbench02.lab-ex.network (10.0.0.65)
Host is up (0.0053s latency).
PORT   STATE    SERVICE
80/tcp filtered http
MAC Address: 54:54:00:D3:D2:02 (Unknown)

Nmap done: 256 IP addresses (13 hosts up) scanned in 2.06 seconds
```

The script can be found in the FSL Test bench git repository.

Kickstart file

The `fsl-testbench.ks` kickstart file is used to setup a minimal installation of Fedora as libvirt-based virtual machine.

```
# Minimal Kickstart file for the Fedora Security Lab test bench
# Installation, not an upgrade
install

# No graphical things needed
skipx
text

# Language
lang en_US.UTF-8

# Kexboard setup
keyboard sg-latin1
#keyboard us

# Networking
network --onboot yes --device eth0 --bootproto dhcp --ipv6 auto --hostname test-
↪bench
```

```
# Authentication
auth --enableshadow --passalgo=sha512
#rootpw {{ server_root_password }}
rootpw testbench

# Services, SELinux and firewall
firewall --enabled --ssh
services --enabled network,sshd
selinux --enforcing
#firstboot --disable
logging --level=info

# Time zone
timezone Europe/Zurich

# Disk setup
zerombr
bootloader --location=mbr --append="rd_NO_PLYMOUTH"
ignoredisk --only-use=vda
clearpart --none --initlabel --drives=vda
autopart

poweroff

%packages
@core
chrony
#dnf
bash-completion
%end
```

The template can be found in the FSL Test bench git repository.

virt-install

`virt-install` creates a virtual machine with the a minimal kickstart file shown in appendix-kickstart.

```
virt-install \
    --name FSL-Test-bench \
    --os-variant fedora18 \
    --ram 1024 \
    --disk /var/lib/libvirt/images/fsl-tb-f18.img,size=6 \
    --location http://mirror.switch.ch/ftp/mirror/fedora/linux/releases/18/Fedora/
↪x86_64/os/ \
    --initrd-inject fsl-testbench.ks \
    --extra-args "ks=file:fsl-testbench.ks" \
    --noautoconsole \
    --vnc \
    --network=network:testbench \
    --mac=52:52:00:00:00:01
```

The template can be found in the FSL Test bench git repository.

Playbook

The **all-in-one.yml** playbook contains all items which are installed on the FSL Test bench. This should give the reader a bit of an inside view of the creation workflow of the Fedora Security Lab Test bench. This playbook calls

every include playbook which contains the tasks for the service it represents.:

```
# This playbook contains tasks to perform on a fresh Fedora installation to
# create a Fedora Security Lab Test bench.
#
# Copyright (c) 2013 Fabian Affolter <fabian@affolter-engineering.ch>
#
# Licensed under CC BY 3.0. All rights reserved.
#
# Usage: sudo ansible-playbook all-in-one.yml -f 10
#
---
- hosts: fsl-tb
  user: root
  vars_files:
    - variables/application-versions.yml
    - variables/sensitive.yml

  tasks:
# Common tasks
################################################################################
    - include: tasks/preparation.yml
    - include: tasks/motd.yml
    - include: tasks/hosts.yml
    - include: tasks/users.yml

# FSL Test bench specific stuff
################################################################################
    - include: tasks/web-servers/lighttpd.yml
    - include: tasks/web-interface.yml

# Services
################################################################################
## Web servers
    - include: tasks/web-servers/nginx.yml
    - include: tasks/web-servers/tomcat.yml
    - include: tasks/web-servers/pywebserve.yml
    - include: tasks/web-servers/nodejs.yml
    - include: tasks/web-servers/mongoose.yml
    - include: tasks/web-servers/darkhttpd.yml
## Database server
    - include: tasks/db-servers/mysql.yml
## FTP server
    - include: tasks/ftp-servers/vsftpd.yml
## Misc servers
    - include: tasks/misc-servers/openssh.yml
    - include: tasks/misc-servers/dropbear.yml
    - include: tasks/misc-servers/tftp.yml
    - include: tasks/misc-servers/telnet.yml
    - include: tasks/misc-servers/cups.yml
    - include: tasks/misc-servers/openvpn-static.yml
    - include: tasks/misc-servers/xrdp.yml
    - include: tasks/misc-servers/mosquitto.yml
## File servers
    - include: tasks/file-servers/samba.yml
    - include: tasks/file-servers/nfs.yml  # Needs manual start
## Mail server
    - include: tasks/mail-servers/postfix.yml
    - include: tasks/mail-servers/dovecot.yml

# Helpers
################################################################################
    - include: tasks/helpers/log-system.yml
```

```
    - include: tasks/helpers/phpinfo.yml
# Shell Detector is not able to handle the present amount of files. Memory?
#    - include: tasks/helpers/php-shell-detector.yml
    - include: tasks/helpers/linfo.yml
    - include: tasks/helpers/phpmyadmin.yml

# Common Gateway Interface (CGI)
################################################################################
    - include: tasks/cgi/cgi.yml
## C (default)
    - include: tasks/cgi/time.yml
## Python
    - include: tasks/cgi/time-py.yml
## Bash
    - include: tasks/cgi/env-sh.yml
    - include: tasks/cgi/system-sh.yml
## Perl
    - include: tasks/cgi/time-pl.yml

# Vulnerable Web Application
################################################################################
    - include: tasks/apps/sqlol.yml
    - include: tasks/apps/sqli.yml
    - include: tasks/apps/bwapp.yml
    - include: tasks/apps/dvwa.yml
    - include: tasks/apps/hackademic.yml
    - include: tasks/apps/xssed.yml

# Honeypots
################################################################################
    - include: tasks/honeypots/honeyd.yml

# Shells
################################################################################
    - include: tasks/shells/b374k.yml
    - include: tasks/shells/dnashell.yml
    - include: tasks/shells/phpshell.yml
    - include: tasks/shells/ajaxshell.yml

# Common tasks
################################################################################
    - include: tasks/cleanup.yml

  handlers:
   - include: handlers/system.yml
   - include: handlers/services.yml
```

The all-in-one.yml playbook can be found in the FSL Test bench git repository.

Host system

The host system was a machine running Fedora 20. CPU is an Intel(R) Core(TM)2 Quad CPU Q6600 @ 2.40 GHz with 8 GB of memory. The hypervisor was KVM and libvirt 1.1.3.4.

```
$ cat /etc/fedora-release
Fedora release 20 (Heisenbug)
```

With one of the latest kernel.

```
$ uname -a
Linux laptop011 3.13.5-202.fc20.x86_64
#1 SMP Mon Mar 3 19:08:00 UTC 2014 x86_64 x86_64 x86_64 GNU/Linux
```

The network configuration looked like this:

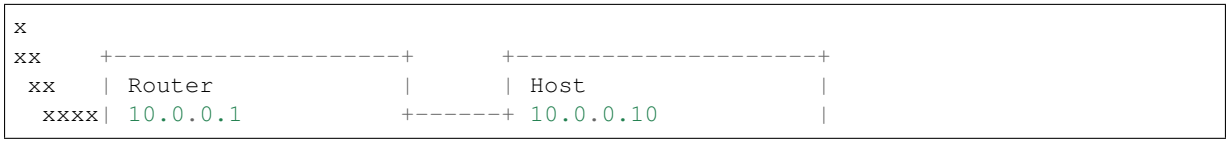
```
$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: em1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
state UP qlen 1000
    link/ether 90:e6:ba:69:f2:76 brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.10/24 brd 10.0.0.255 scope global em1
    inet6 fe80::92e6:baff:fe69:f276/64 scope link
       valid_lft forever preferred_lft forever
3: p37p1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
master vnet0 state UP qlen 1000
    link/ether 90:e6:ba:69:d6:ae brd ff:ff:ff:ff:ff:ff
    inet6 fe80::92e6:baff:fe69:d6ae/64 scope link
       valid_lft forever preferred_lft forever
4: p3p1: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast
state DOWN qlen 1000
    link/ether 00:30:4f:53:fa:b7 brd ff:ff:ff:ff:ff:ff
5: vnet0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue
state UP
    link/ether 90:e6:ba:69:d6:ae brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.104/24 brd 10.0.0.255 scope global vnet0
    inet6 fe80::92e6:baff:fe69:d6ae/64 scope link
       valid_lft forever preferred_lft forever
6: virbr0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue
state DOWN
    link/ether 52:54:00:31:fe:4d brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.1/24 brd 192.168.122.255 scope global virbr0
[snip]
14: vnet5: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast master
vnet0 state UNKNOWN qlen 500
    link/ether fe:52:00:00:00:01 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::fc52:ff:fe00:1/64 scope link
       valid_lft forever preferred_lft forever
```

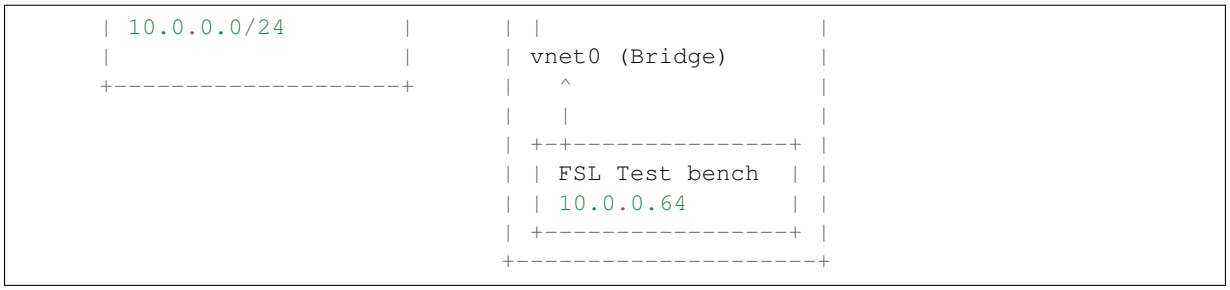
Network

> **Warning:** This information could be obsolete.

The diagram below shows the layout of the network during the creation and the setup of the Fedora Test bench. The FSL Test bench needs an IP address out of 10.0.0.0/24 because some services have this IP address range in their configuration files. The hardcoded IP address is 10.0.0.64. This is a drawback of the distribution as virtual machine.

The IP range needs to be changed in the livirtd configuration when putting this virtual machine on a live media.




| 10.0.0.0/24 |
| vnet0 (Bridge)
| FSL Test bench
| 10.0.0.64