w1r3 Network Documentation

Release 1.0

w1r3 Network

June 08, 2014

Contents

1 Infrastructure 3

1.1 Team orientation . . . 3

1.2 Accounts . . . 3

1.3 Scripts . . . 3

1.4 Kernel . . . 3

2 Services 5

2.1 nya.ch - linux-kvm vps hosting . . . 5

2.2 Arch Linux Repository . . . 10

The w1r3 network is an association from Switzerland. We're group of individuals, involved in information technology and related areas. We want to offer the oppertunity to use networks free and authentic. We are against any censorship and the elimination of net neutrality. We provide infrastructure and services for this purpose. We support education, scientific activity and the development of recommendations and specifications for methods in information technology.

We use SOPs (standard operating procedures) to list how we run things.

Contents:

CHAPTER 1

Infrastructure

Contents:

1.1 Team orientation

u know?

1.2 Accounts

Each user has an account in our ldap structure. It is possible to use the account credenials to authenticate on various services.

1.3 Scripts

We have a repository for scripts used in our infrastructure. The scripts are mainly helpers for users or for other software components.

This SOP is a guideline for writing new and maintaining existing scripts. It should help with the decision if it's a good idea to write a script for a specific function, how to document it and how to integrate it in our infrastructure.

1.3.1 When a script is a good idea

1.3.2 Why you shouldn't write one

1.3.3 Documentation

1.3.4 Integration

1.4 Kernel

Our Arch Linux powered servers are running a modified Linux kernel. The kernel is built with grsecurity patches and a working, semi-minimalist kernel configuration.

The linux-w1r3-xen kernel is tested with virtual private servers from Linode. The linux-w1r3-kvm kernel is tested with virtual private servers from nya.ch. We do not recommend to use this kernels in other infrastructures, because they are specifically designed and tested for the named environments.

CHAPTER 2

Services

Contents:

2.1 nya.ch - linux-kvm vps hosting

Contents:

2.1.1 virssh

Virssh offers you the possibility to control your vm. This includes a serial console.

Connect

For using virssh you need an SSH-key. In case your SSH-key isn't in our system please contact the support team.

```
ssh -t vmadmin@rachel.munich.w1r3.net
```

This shows you the possible commands

Example:

```
ALERT: NOT allowing command ""
Allowed commands for you are:

ssh -t vmadmin@rachel.munich.w1r3.net sudo virsh list
ssh -t vmadmin@rachel.munich.w1r3.net sudo virsh console example.nya.ch
ssh -t vmadmin@rachel.munich.w1r3.net sudo virsh destroy example.nya.ch
ssh -t vmadmin@rachel.munich.w1r3.net sudo virsh shutdown example.nya.ch
ssh -t vmadmin@rachel.munich.w1r3.net sudo virsh start example.nya.ch
ssh -t vmadmin@rachel.munich.w1r3.net sudo virsh start example.nya.ch --console
```

2.1.2 Rescue

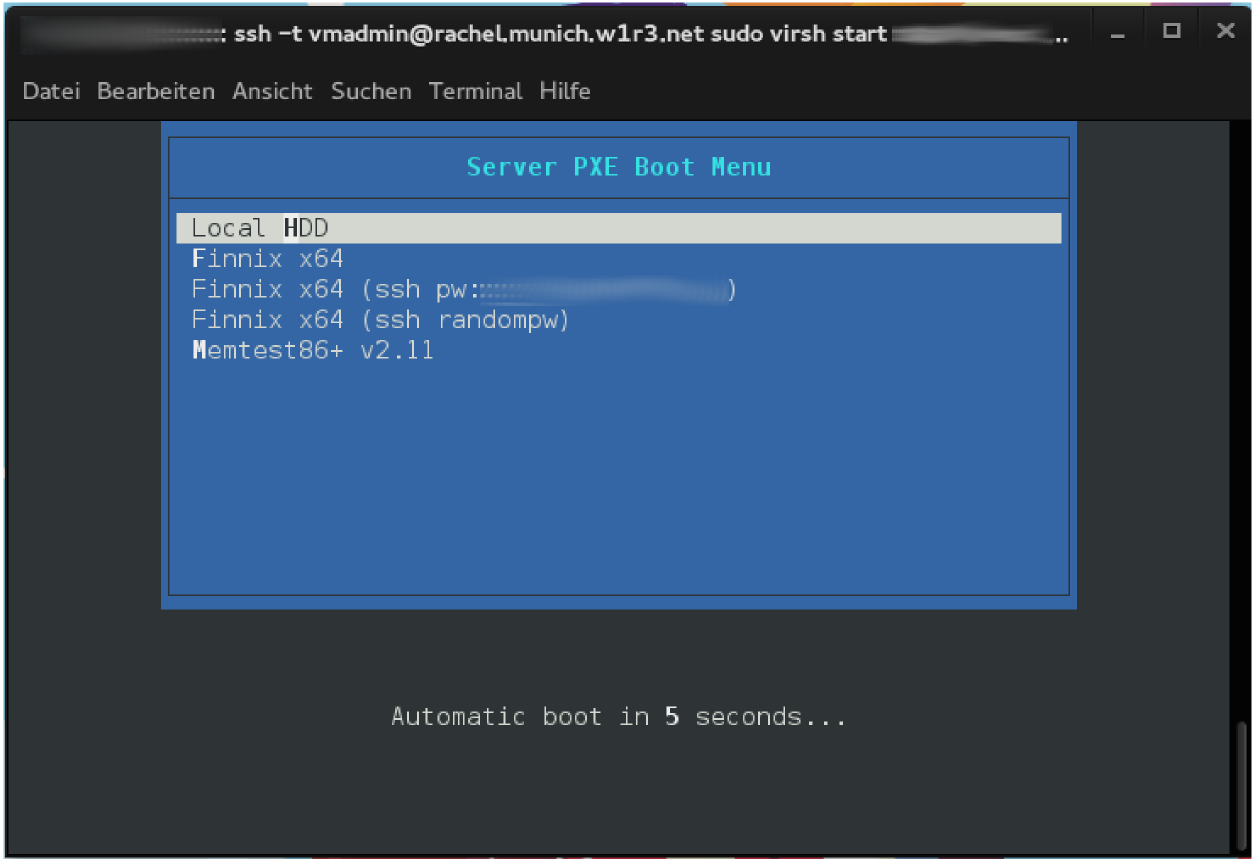
Everyone does mistakes. That's the reason why we provide you with Finnix. Finnix is a recovery system and can be used to fix errors and install certain distributions like Arch Linux.

Boot into Finnix

We have to shutdown the VM in order to boot Finnix. After doing so enter the command below to start it again with console.

```
ssh -t vmadmin@rachel.munich.w1r3.net sudo virsh start example.nya.ch --console
```

After doing so we can see a bootmenu like in the picture below:



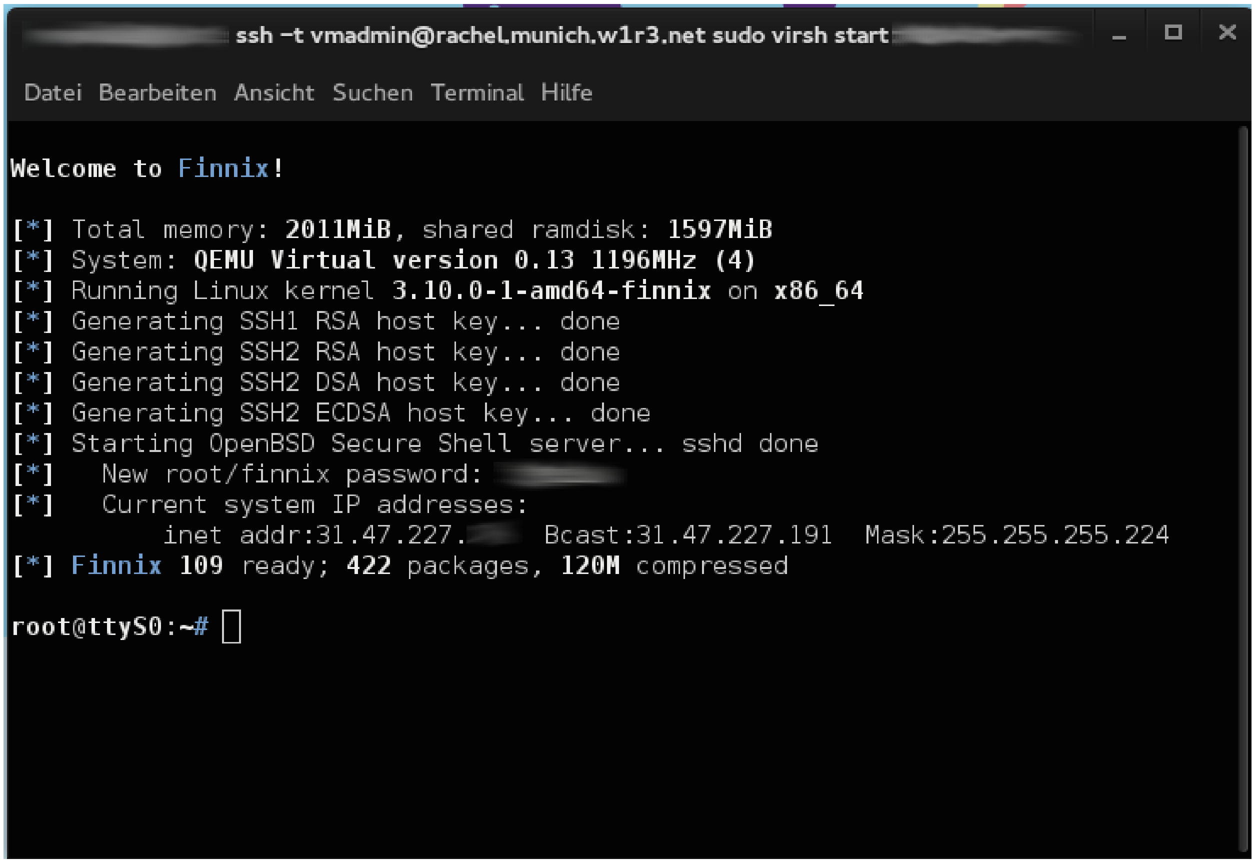
[redacted]: ssh -t vmadmin@rachel.munich.w1r3.net sudo virsh start [redacted] ..
Datei
Bearbeiten
Ansicht
Suchen
Terminal
Hilfe

```
Server PXE Boot Menu

Local HDD
Finnix x64
Finnix x64 (ssh pw:[redacted])
Finnix x64 (ssh randompw)
Memtest86+ v2.11

Automatic boot in 5 seconds...
```

Choose one of these Finnix entrys and you'll end up here like in the picture below (I chose the third one)



[redacted] ssh -t vmadmin@rachel.munich.w1r3.net sudo virsh start [redacted]

```
Welcome to Finnix!

[*] Total memory: 2011MiB, shared ramdisk: 1597MiB
[*] System: QEMU Virtual version 0.13 1196MHz (4)
[*] Running Linux kernel 3.10.0-1-amd64-finnix on x86_64
[*] Generating SSH1 RSA host key... done
[*] Generating SSH2 RSA host key... done
[*] Generating SSH2 DSA host key... done
[*] Generating SSH2 ECDSA host key... done
[*] Starting OpenBSD Secure Shell server... sshd done
[*]   New root/finnix password: [redacted]
[*]   Current system IP addresses:
          inet addr:31.47.227.[redacted]   Bcast:31.47.227.191  Mask:255.255.255.224
[*] Finnix 109 ready; 422 packages, 120M compressed

root@ttyS0:~# 
```

2.1.3 Installation of Arch Linux

We will need our Rescuesystem to install Arch Linux.

Preparing the Installationenvironment

Run the following commands:

```
wget http://mirror.de.leaseweb.net/archlinux/iso/latest/arch/x86_64/root-image.fs.sfs
unsquashfs root-image.fs.sfs
cd squashfs-root
mount -o loop root-image.fs /mnt
/mnt/bin/arch-chroot /mnt
mkdir /run/shm
```

Select a mirror

We provide a mirror which makes everything faster

```
echo "Server = http://mirror.w1r3.org/\$repo/os/\$arch" > /etc/pacman.d/mirrorlist
```

Partitioning

We are going to allocate the whole block device.

```
fdisk /dev/vda
```

Press n to create a new partition followed with four times enter and w.

Formating

```
mkfs.ext4 /dev/vda1
```

Mount the new Filesystem

```
mount /dev/vda1 /mnt
```

Install basic packages

We are going to install the package groups base and base-devel but first we have to generate our keyring. We need haveged for this task because of the missing entropy. The second one is for building packages.

```
sed -i "s/SigLevel    = Required DatabaseOptional/SigLevel = Never/" /etc/pacman.conf
pacman -Sy haveged procps-ng --noconfirm
haveged -w 1024
pkill haveged
pacman -Rs haveged procps-ng --noconfirm
pacman-key --populate archlinux
pacman-key --refresh-keys
sed -i "s/SigLevel = Never/SigLevel = Required DatabaseOptional/" /etc/pacman.conf
pacstrap /mnt base base-devel
```

Generate /etc/fstab

```
genfstab -U -p /mnt > /mnt/etc/fstab
```

Chroot into your new system

```
arch-chroot /mnt
```

Set root password

```
passwd
```

Install some extra packages which are always good to have

These packages are mostly optional except for grub which will be used as bootloader.

```
pacman -S vim zsh git grub openssh rsync ntp ferm atop htop mtr traceroute wget net-tools dnsutils
```

Install and configure GRUB

First we need to edit `/etc/default/grub` to enable serial console support. Replace the existiing `GRUB_CMDLINE_LINUX_DEFAULT` with the following:

```
GRUB_CMDLINE_LINUX_DEFAULT="console=tty0 console=ttyS0,38400n8"
```

Further add these lines:

```
GRUB_TERMINAL=serial
GRUB_SERIAL_COMMAND="serial --speed=38400 --unit=0 --word=8 --parity=no --stop=1"
```

Now execute the commands below to install grub to blockdevice and generate the config file.

```
grub-install --recheck /dev/vda
grub-mkconfig -o /boot/grub/grub.cfg
```

Configure networking

Now create a netctl profile and configure the network configurations. Change the IP settings to the IPs you were given.

```
nano /etc/netctl/ethernet-static

Description='nya.ch connection'
Interface=ens3v1
Connection=ethernet

IP=static
Address='31.47.227.CHANGEME/CHANGE'
Gateway='31.47.227.CHANGEME'
DNS=('8.8.4.4')
DNSDomain='nya.ch'

IP6=static
Address6=('2a02:2260:2:1f60::CHANGEME/64')
Gateway6='2a02:2260:2:1f60::1'
```

Enable the netctl profile

```
netctl enable ethernet-static
```

Reboot

Now it's time to reboot into your own system.

```
exit
umount /mnt
exit
reboot
```

Timezone and Hostname

Set timezone and hostname. Please use the hostname which were given to you.

```
timedatectl set-timezone Europe/Berlin
hostnamectl set-hostname HOSTNAME
```

Generate and set locales

Uncomment the locales for DE, AT, CH, US (all UTF-8) in the file `/etc/locale.gen` and run the command `locale-gen`.

Now set the standard system locale and keymap

```
localectl set-locale LANG="en_US.UTF-8"
localectl set-keymap de
```

Reboot and have Fun

Nothing more to say ;)

2.2 Arch Linux Repository

This page is empty and will be updated as soon as there is a productive repository.