Domains & Daemons

System and Network Administration

Revision 2 (2020/21)

Pierre-Philipp Braun <pbraun@nethence.com>

Table of contents

- System Preparation
- Essential Protocols
- Daemons’ Setup
- Name Resolution
- Tips & Tricks

Hosting requirements

How to run a public service?...

e.g. HTTP or FTP on **yourhost.com**

==> What you need

- a server w/ public IP
  - AWS EC2/EC3
  - GCP
  - Scaleway / Dedibox
  - white-ip on-premises…
- a daemon
- a registered domain and DNS hosting
- a bunch of SSL certificates

// Everything clear on hosting requirements?

System Preparation

- post-install
- tuning / optimization / hardening
- app-specific

GNU/Linux server post-install

assuming Slackware Linux or Ubuntu Server

- network & SSHD **+ SSHGUARD** (vs. fail2ban, …)
- MOST IMPORTANTLY `netstat`
- default runlevel/target
- boot-loader
- package repository mirror
- system updates & few more packages
- default editor & screenrc vs. tmux
- smtp relay & mail aliases
- ntpdate & hwclock

BSD-specific server post-install

assuming NetBSD or DragonFlyBSD

- timezone
- shell & skeletons
- clean-up services (why not on Ubuntu?)
- tuning syslogd
- indexing
- pkg vulns
- cron fixup
- tuning daily report

Network & boot-loader

Ubuntu 17+

- either you're happy with Netplan (`/etc/netplan/` and YAML)
- –or– you go the good old debian-style `/etc/network/interfaces` setup

Back to old Debian-style

```
apt install ifupdown net-tools
vi /etc/default/grub

netcfg/do_not_use_netplan=true
# mitigations=off

update-grub
reboot
```

SSHD

```
vi /etc/ssh/sshd_config

Port XXXX
AllowGroups wheel
PermitRootLogin without-password
PasswordAuthentication no
```

SVR4 runlevels

remember fundamentals lecture on what's a server?

no graphical interface needed

```
telinit 3
systemctl set-default multi-user.target
```

check

```
runlevel
systemctl get-default
```

Slackware package repository

```
mv -i /etc/slackpkg/mirrors /etc/slackpkg/mirrors.dist
vi /etc/slackpkg/mirrors

# FRANCE
http://nephtys.lip6.fr/pub/linux/distributions/slackware/
    slackware64-current/

slackpkg update
```

and upgrade

```
slackpkg upgrade pkgtools slackpkg
slackpkg upgrade-all
updatedb
locate \.new | grep new$ | grep -v sbopkg
```

Ubuntu package repository

```
vi /etc/apt/sources.list
```

```
# RUSSIA
deb http://ru.archive.ubuntu.com/ubuntu/ xenial \
    main restricted universe
deb http://ru.archive.ubuntu.com/ubuntu/ xenial-security \
    main restricted universe
deb http://ru.archive.ubuntu.com/ubuntu/ xenial-updates \
    main restricted universe
#deb http://ru.archive.ubuntu.com/ubuntu/ xenial-proposed \
    main restricted universe
#deb http://ru.archive.ubuntu.com/ubuntu/ xenial-backports \
    main restricted universe
#multiverse
```

```
apt update
```

and upgrade

```
apt full-upgrade
apt autoremove --purge

dpkg -l | grep linux-image
uname -r
dpkg --purge ...
reboot
```

Few more packages

```
apt install lynx curl wget lftp ksh htop dos2unix
apt install apt-transport-https software-properties-common
```

Default editor

```
export EDITOR=/usr/bin/vi
update-alternatives --list vi
update-alternatives --config vi
```

GNU Screen

```
vi /etc/screenrc

deflogin off
vbell on
term xterm
defutf8 on
utf8 on on
startup_message off
caption always ”%-Lw%{= BW}%50>%n%f* %t%{-}%+Lw%< | %l | %c:%s”
defscrollback 65000
shelltitle ””

bindkey ^[, prev
bindkey ^[. next
```

The system talks

- logs `/var/log/messages` vs. `/var/log/syslog`
- default inbox location `/var/mail/USER` or `/var/spool/mail/USER`
- `biff` says you got mail
- ideal for cron jobs' output (stdout & stderr)

SMTP relay & mail aliases

setup postfix

```
vi /etc/postfix/main.cf

relayhost = SMART-HOST

postfix reload
```

setup mail aliases

```
vi /etc/mail/aliases
vi /etc/aliases

root:       REAL@EMAIL

newaliases
```

Validate

```
slackpkg install s-nail


apt install bsd-mailx


date | mail -s `uname -n` root


tail /var/log/maillog
tail /var/log/mail.log
```

One shot / weekly job NTP is enough

no need for an additional NTP daemon unless this is a cluster

Russia

```
ntpdate -u ru.pool.ntp.org
```

France

```
ntpdate -u ntp.obspm.fr
```

Sync with CMOS

```
hwclock --utc --systohc
```

Indexing

How to search for files normally?…

==>

```
find . -name <string>
find . | grep <string>
```

How to get make it faster?...

==>

```
updatedb
locate <string>
```

// Questions on system preparation?

Essential Protocols

all about layer 7

HTTP

- NGINX (reverse-proxy & lbs capable)
- Apache v2.4 (reverse-proxy & lbs capable)
- Thttpd // LAB benchmark vs. nginx on static pages or images
- Bozohttpd, Mathopd, …

Reverse-proxy & lbs only

- HA-Proxy
- Traefik, …

SSL termination only

- Hitch
- Stunnel

Virtual hosts

Let’s assume

```
http domain.com --> 301 https domain.com
https domain.com --> 200
```

Now what happens for those?…

```
http 1.2.3.4 --> ?
https 1.2.3.4 --> ?
http vhost.domain.com --> ?
```

==> define default vhost as a CATCH ALL and redirect to some priviledged URL

And avoid duplicate vhosts for better search engine references

Acceptance testing with SNI

```
curl -i https://domain.com
# -k
```

PKI & SSL (brief intro)

- issuer / subject
- Confidentiality
- Integrity
- Authenticity
- Non-repudiation
- Extended Validation Certificates (EV SSL)

Where to get the certificates from?

==>

- self-signed
- bundled-trusted CAs
    - pay for it
    - get one from let’s encrypt
    - get one as a domain owner (e.g. Gandi)
    - get one as a customer (e.g. CloudFlare)
- your own CA
- wildcard vs single-host

SSL – Apache config

```
vi httpd.conf

SSLEngine on
SSLCertificateFile /etc/httpd/ssl/certificate.crt
SSLCertificateKeyFile /etc/httpd/ssl/certificate.key
SSLCertificateChainFile /etc/httpd/ssl/issuer-concat-cert.crt
```

SSL – NGINX config

```
vi nginx.conf

ssl_certificate /etc/ssl/fullchain.pem;
ssl_certificate_key /etc/ssl/privkey.pem;
```

That’s it really?…

==> not really

- HSTS
- protocol versions
- cipher suites
- (cipher preference/order)

Recommended SSL settings (updated regularly)

Strong Ciphers (see Other Software section)
<https://syslink.pl/cipherlist/>

Applied Crypto Hardening
<https://bettercrypto.org/>

Acceptance testing
<https://www.ssllabs.com/ssltest/analyze.html?d=nethence.com>

Yesterday’s HTTP

Usually we were going for

```
telnet HOST 80
```

But we should now go through SSL

```
openssl s_client -connect HOST:443
```

FTP

- vsftpd
- proftpd
- lukemftpd (tnftpd)
- pureftpd

Active vs Passive

- entry-point: port `21`
- active connection: port `20`
- passive connection: undefined port range

Firewalling & NAT

- NAT recap & port forwarding
- need to define the range for firewalling
- need to define public IP if going through NAT

lukemftpd (tnftpd) / port range example

```
vi /usr/local/etc/ftpd.conf

motd all none
portrange all 70000 70999
umask chroot 022
umask real 022

#in case you want the thing to work publicly while living
#behind a NAT, advertise the public IP,
#advertize all IP_ADDRESS

vi /usr/local/etc/ftpusers

storage     allow chroot
ftp     allow guest
anonymous   allow guest
*       deny
```

FTP is insecure, right?…

==>

- right, it’s all clear-text
- incl. login and password

Let’s find ways to do **FTPS**!

vsftpd / ssl example

```
rsa_cert_file=/etc/ssl/private/vsftpd.pem
rsa_private_key_file=/etc/ssl/private/vsftpd.pem
ssl_enable=YES
force_local_data_ssl=YES
force_local_logins_ssl=YES
```

proftpd / ssl example

```
TLSEngine on
TLSCACertificateFile /etc/ftpd/root.cert.pem

TLSRSACertificateFile /etc/ftpd/server-rsa.cert.pem
TLSRSACertificateKeyFile /etc/ftpd/server-rsa.key.pem
```

Elliptic Curves

```
TLSECCertificateFile /etc/ftpd/server-ec.cert.pem
TLSECCertificateKeyFile /etc/ftpd/server-ec.key.pem
```

make it mandatory?

```
TLSRequired on
```

auth clients?

```
TLSVerifyClient off
```

Protocols for email

- SMTP – different ways to think about it
- IMAP4 vs. POP3

–> will be covered in infrastructure services lecture

// Questions on essential L7 protocols?

Daemons' Setup

Types of documentation *aka « livrables »*

- system & network specifications
- install, post-install & (app specific) sysprep
- daemon/app install & configuration
- maintenance/operations

And finalize delivery with acceptance testing

Daemons' installation

- as distribution binary package
- as upstream binary package or executable
- from source

Install as binary package

Slackware

```
slackpkg search PKGNAME
slackpkg install PKGNAME
```

Ubuntu

```
apt search ^PKGNAME
apt install PKGNAME
```

Build from source

Preliminary notes

```
tar xaf software.tar.gz
tar xaf software.tar.bz2
tar xaf software.tar.xz

git clone https://...
git clone ssh://...

grep ^proc /proc/cpuinfo | tail -1
export MAKEFLAGS=-j8
```

GNU Autotools

```
cd software/
./configure --help | less
./configure
grep ^proc /proc/cpuinfo | tail -1
time nice make
su -c ”make install”
```

CMake

```
cd hackrf/
mkdir host/build/
cd host/build/
cmake ../
time nice make
su -c ”make install”
```

New libs in da place?

```
vi /etc/ld.so.conf
```

```
/usr/local/lib
```

```
ldconfig
```

What are the Pros & Cons when building from source?…

==>

Pros: full control

- latest version & features
- get/make patches faster
- custom build options
- possibly hardened and w/o SystemD

Cons: need to work more

- keep track / subscribe
- re-build

Daemon configuration

- backup as `.orig` or `.dist`
- keep it clean (wipe-out comments)
- eventually get rid of `folder.d/*` includes (e.g. Dovecot)

Btw how to RTFM?…

==>

Find manuals on a given topic

```
apropos postfix
apropos -r ^intro
apropos -r ^hier
```

Anything missing?

```
manpath
#echo $MANPATH
```

Short description

```
whatis intro
whatis hier
```

Manual section numbers

```
1   Executable programs or shell commands
2   System calls (functions provided by the kernel)
3   Library calls (functions within program libraries)
4   Special files (usually found in /dev)
5   File formats and conventions eg /etc/passwd
6   Games
7   Miscellaneous (including macro packages and conventions),
    e.g. man(7), groff(7)
8   System administration commands (usually only for root)
9   Kernel routines [Non standard]
```

Tricky example

```
postconf (1)          - Postfix configuration utility
postconf (5)          - Postfix configuration parameters

man 5 postconf
man –a postconf
```

Find an executable

```
which vi
whereis vi
```

Operations & troubleshooting

Where are the logs?…

==> ***as root***

```
vi ~/log

tail -n0 -F /var/log/* /var/log/*/*

chmod +x ~/log
~/log
```

Init scripts

SVR4

`/etc/init.d/`

`/etc/rc<RUNLEVEL>.d/<S|K>NNname`

RCNG & Slackware

`/etc/rc.d/`

SystemD

`systemctl`

Note sometimes distro keeps it retro-compatible for a while e.g. `postfix` & `pcsd`

Anything else in mind?...

OpenRC

. . .

Runit

. . .

Upstart

. . .

And there’s even more exotic stuff…

Suckless sinit

. . .

S9

. . .

How to enable at boot-time and deal with daemons without an init system?...

CLI

`DAEMON AND ARGUMENTS`

`pgrep`

`pkill`

UNIX

`/etc/rc.local`

`/etc/rc.local_shutdown`

OpenBSD

`/etc/rc`

SVR4 on older RHEL/CentOS

```
chkconfig --list

chkconfig asterisk on
chkconfig asterisk off

service asterisk status
service asterisk start
service asterisk stop
```

SVR4 on Devuan

one-shot

```
service asterisk start
```

```
service asterisk status
```

```
service asterisk stop
```

at boot-time

```
update-rc.d asterisk defaults
```

```
service --status-all
```

```
update-rc.d asterisk -f remove
```

SystemD

one-shot

```
systemctl start daemon
systemctl stop daemon
```

at boot-time

```
systemctl list-unit-files
/search


systemctl enable daemon
systemctl status daemon
systemctl disable daemon
```

What is more important and efficient than setting up a ***system-firewall*** *locally?…*

==> no system-firewall required

Check what ports are listening and let true firewalls do the work on network segmentation and/or ACLs

- from the system – *how to do that locally?...*
- from another host – *how to do that remotely?...*

==> what is listening locally

no system-firewall required

```
netstat -an --inet --inet6
netstat -antu
netstat -antup
```

all (`-a`) –> listening (`-l`)

```
netstat -lntup
```

and `-ee` for UID

Example output

```
Proto Recv-Q Send-Q Local Address           Foreign Address
    State       PID/Program name

tcp        0      0 10.1.1.4:2222           0.0.0.0:*
    LISTEN      9827/sshd
tcp        0      0 127.0.0.53:53           0.0.0.0:*
    LISTEN      15559/systemd-resol
tcp        0      0 10.1.1.4:4567           0.0.0.0:*
    LISTEN      978/ruby2.5
udp        0      0 127.0.0.53:53           0.0.0.0:*
                15559/systemd-resol
udp        0      0 0.0.0.0:68              0.0.0.0:*
                29212/dhclient
```

==> what is listening remotely

```
nmap nethence.com -p 80,443
```

```
nc -vz nethence.com 80 443
```

and further validate with `telnet/s_client/curl/ftp`

What about UDP?…

==>

as root

```
nmap -sU xc.os3.su -p 53
```

```
netcat -uvz xc.os3.su 53
```

Service ports

```
grep http /etc/services
grep ftp /etc/services
grep smtp /etc/services
grep submission /etc/services
grep imap /etc/services
```

// Questions on daemons installation & configuration?

Name Resolution

```
 (\(~)/)
  )@_@(  #
 ((q_p))'
 /\|U|/\
/  `='  \
```

Static vs. dynamic

- static with `/etc/hosts`
- vs. dynamic (NIS+, DNS, NetBIOS-NS, …)
- define which one(s) to use in `/etc/nsswitch.conf`

DNS

yet another L7 protocol

Everybody clear on what it does?...

==> The principle should be clear already for 3rd year bachelors

It binds / maps IPs with names so you can call e.g.

```
http://domain.tld/
```

instead of

```
http://1.2.3.4/
```

DNS client setup

- directly with `/etc/resolv.conf`
- –or– by means of a stub-resolver with caching

Stub-resolver products

- resolvconf + dnsmasq
- systemd-resolved

Full-blown DNS server products

Popular ones

- ISC BIND - can do everything
- NLnet Labs NSD - authoritative only
- NLnet Labs Unbound - forwarding only & cache
- Knot DNS - authoritative only

Authoritative (server conf points to zone-file)

```
vi /var/chroot/nsd/etc/nsd/nsd.conf

zone:
    name: ”os3.su”
    zonefile: ”%s.db”
```

DNS records (zone-file format)

```
vi /var/chroot/nsd/etc/os3.su.db

$ORIGIN os3.su.
$TTL 1800

@                IN NS              ns
@                IN NS              ns2
ns               IN A               62.210.110.7
ns2              IN A               62.210.16.8

@                IN A               62.210.110.7
*                IN A               62.210.110.7

@                IN MX 5            mx
mx               IN A               188.130.155.139

some-host        IN A               x.x.x.x
npf              IN CNAME           some-host
```

DNAME example

redhat got bought by IBM, right? now imagine they want to get rid of the name

```
$ORIGIN redhat.com.

@       IN DNAME redhat.ibm.com.
```

now `anything.redhat.com` goes and resolves `anything.redhat.ibm.com`.

Authoritative features

Delegations

Master-slave with XFR & notify

DNSSEC island vs full chain of trust

```
* Unbound possibly validating
* still optional...
```

Alternatives to DNSSEC

```
* DNS over HTTPS (DoH)
* DNS over TLS (DoT)
```

How a forwarder works

non-authoritative

It does iterative queries (so you can do recursive queries on him)

```
cat /usr/share/dns/root.hints
```

```
.
net.
online.net.
```

DNS queries

iterative vs. recursive

```
host nethence.com
host nethence.com 8.8.8.8
host -r # non-recursive query

dig nethence.com +short
dig nethence.com +short @8.8.8.8
# +[no]recurse
# +[no]trace
```

Root servers

The 13 root name servers are operated by 12 independent organisations

Are some in Russia (not counting Belarus & friends)?...

==> Yes, those are spread everywhere now. As of Feb 2021 in Russia we’ve got

E - NASA Ames Research Center - 1 in Moscow

F - Internet Systems Consortium, Inc. ==> 2 in Moscow + 1 St-Peter

J - Verisign, Inc. ==> 1 in Moscow + 1 St-Peter

K - RIPE NCC ==> 1 in Moscow + 1 St-Peter + 1 Novosibirsk

L - ICANN ==> 3 in Moscow + 1 St-Peter

I - Netnod ==> 1 St-Peter

Super-duper server for Siberia

```
Novosibirsk, RU
Operator    RIPE NCC
IPv4    193.0.14.129
IPv6    2001:7fd::1
ASN 25152
```

Recursive queries

Old-school client setup

```
vi /etc/resolv.conf

    nameserver ...
```

–or– new-school stub-resolvers

```
...
```

–or– validating-resolver on localhost!

```
vi /etc/unbound/unbound.conf

    forward-zone:
    name: ”.”
    forward-addr: x.x.x.x@53
```

Why a caching forwarder is a good thing to have

- saves some traffic (if not bandwidth)
- safer / internal
- possibly also a DNSSEC **validating resolver**

```
 (\(~)/)
  )@_@(   #
 ((q_p))'
 /\|U|/\
/   `='   \
```

// Questions on name services?…

Daemon Tips & Tricks

- Easy package management
- Init scripts and pids
- Process name is tricky
- Clean configs
- Troubleshoot a daemon

Easy package management

debian/ubuntu

```
apt update
apt search ...
apt install ...
```

find packages

```
dpkg -S filename # belongs to installed package ...
apt file-search
```

slackware (no package mgmt)

http://docs.slackware.com/slackware:package_and_dependency_management_shouldn_t_put_you_off_slackware

Init scripts and pids

- Ubuntu starts and enables by default
- Slackware eventually deploys the init script

Otherwise use `/etc/rc.local` or `/etc/rc.d/rc.local` as follows

starting daemons at boot-time

```
vi /etc/rc.local
```

```
echo -n starting DAEMON...
DAEMON && echo done
```

```
chmod +x /etc/rc.local
```

ubuntu

```
systemctl status rc-local
```

slackware & RHEL/CentOS/Fedora

```
ln -s rc.d/rc.local /etc/rc.local
```

and for shutdown

```
vi /etc/rc.local_shutdown

echo -n killing DAEMON...
pkill DAEMON && echo done

chmod +x /etc/rc.local_shutdown
```

(Ubuntu)

However there's no `rc-local-shutdown` *service, what do?...*

==> manually create the systemd service

```
vi /lib/systemd/system/rc-local-shutdown.service

[Unit]
Description=/etc/rc.local_shutdown Compatibility
ConditionFileIsExecutable=/etc/rc.local_shutdown
DefaultDependencies=no
After=rc-local.service basic.target
Before=shutdown.target

[Service]
ExecStop=/etc/rc.local_shutdown
StandardInput=tty
RemainAfterExit=yes

[Install]
WantedBy=shutdown.target
EOF
```

check that the process is running with childs

```
ps auxfww | less
```

example output

```
nsd        965  0.0  1.0 109716 88888 ?        Ss   Jan03
    0:00 /usr/local/sbin/nsd
nsd        966  0.0  0.0  11472  3940 ?        S    Jan03
    0:00  \_ /usr/local/sbin/nsd
nsd      21486  0.0  0.0  27432  5120 ?        S    12:04
    0:00      \_ /usr/local/sbin/nsd
nsd      21487  0.0  0.0  27636  5460 ?        S    12:04
    0:00      \_ /usr/local/sbin/nsd
nsd      21488  0.0  0.0  27636  5132 ?        S    12:04
    0:00      \_ /usr/local/sbin/nsd
nsd      21489  0.0  0.0  27636  5460 ?        S    12:04
    0:00      \_ /usr/local/sbin/nsd
nsd      21490  0.0  0.0  27636  5464 ?        S    12:04
    0:00      \_ /usr/local/sbin/nsd
nsd      21491  0.0  0.0  27432  5460 ?        S    12:04
    0:00      \_ /usr/local/sbin/nsd
nsd      21492  0.0  0.0  27636  5460 ?        S    12:04
    0:00      \_ /usr/local/sbin/nsd
```

by PID

```
ps ufww --pid PID
ps uww --quick-pid PID
```

by process name

```
pgrep -a PROCESS-NAME
pidof PROCESS-NAME
```

Process name is tricky

```
grep ^Name /proc/PID/status
```

NSD

```
Name:   xfrd
Umask:  0022
```

Postfix?

```
Name:   main
Umask:  0022
```

Clean configs

Make a backup copy before tuning! This is a **regular expression** aka `regex` or `regexp`

```
mv daemon.conf daemon.conf.dist
sed -r '/^[[:space:]]*(#|$)/d' daemon.conf.dist > daemon.conf
vi daemon.conf
```

Troubleshoot a daemon

How to troubleshoot a daemon?…

==> Read the logs IN REAL TIME

The only way to troubleshoot anything on a Unix system. General log

```
tail -F /var/log/syslog   # Debian/Ubuntu
tail -F /var/log/messages # RHEL & Slackware
```

E.g. for solving authentication issues, check

```
tail -F /var/log/auth.log # Debian/Ubuntu
tail -F /var/log/secure   # RHEL & Slackware
```

Super-duper log reader

gnu/linux

```
tail -n0 -F /var/log/* /var/log/*/*
```

slackware got too much folders over there

```
tail -n0 -F /var/log/* /var/log/nginx/*
```

netbsd

```
tail -F /var/log/messages
```

Other CLI-based log readers worth mentioning

`lnav`

`logwatch`

`swatch`

MariaDB / MySQL

Typical usage after installation

```
mysql_secure_installation
mysql -u root

    show databases;

    CREATE DATABASE netxms;
    CREATE USER 'netxms'@'localhost' IDENTIFIED BY 'PASSWORD-HERE';
    GRANT ALL on netxms.* to 'netxms'@'localhost';
```

// Questions on daemon tips & tricks?

Advanced LAB

- rebuild SSHD without OpenSSL and tune it like hell
- rebuild SSHD against LibreSSL and validate which ciphers you can use (Camellia in da place?)