

Debian Guides

- [Bootstrapping a Debian Machine](#)
- [Bootstrapping a Debian Workstation](#)
- [Installing NVIDIA Graphics Drivers](#)
- [Debian Machine Maintenance](#)
- [Networking Cheat Sheet](#)
- [Security Hardening](#)
 - [Protecting GRUB with a Password](#)
 - [Enforcing Password Quality](#)
 - [Verifying Installation Integrity](#)
- [Tweaking a Debian Machine](#)
 - [Tweaking GRUB](#)
 - [Tweaking Bash](#)
- [How to Setup a...](#)
 - [Using KeePassXC as the Secret Service](#)
 - [How to Setup a YubiKey Authentication](#)
- [Troubleshooting](#)
 - [Missing Disk Space?](#)

Bootstrapping a Debian Machine

[debian.org downloads debian](https://www.debian.org/downloads/debian)

Installation

Clean Up the Standard Installation

```
sudo apt purge debian-faq doc-debian installation-report tasksel
```

```
sudo apt --purge autoremove
```

Install Essentials

```
sudo apt install bash-completion bind9-dnsutils bzip2 curl ethtool htop iotop  
screen sudo unzip vim wget
```

Install Security Enhancements

[Enforcing Password Quality](#) [Verifying Installation Integrity](#)

```
sudo apt install debsums libpam-pwquality libpam-tmpdir
```

Install Virtual Machine Guest Agent

```
sudo apt install qemu-guest-agent
```

Configuration

Grant Password-less Sudo Access



Since `sudo` is likely not yet available, the following commands must be run as `root`.

```
( umask 0337; tee "/etc/sudoers.d/sudo" >/dev/null <<'EOF'
%sudo ALL=(ALL) NOPASSWD: ALL
EOF
)
chmod 0440 "/etc/sudoers.d/sudo"
```

```
gpasswd --add [USERNAME] sudo
```

Set the Machine Hostname

```
sudo hostnamectl set-hostname [HOSTNAME]
```

```
sudo sed -i '/^127.0.1.1/d' "/etc/hosts"
sudo sed -i '1i127.0.1.1\t[HOSTNAME].[DOMAIN] [HOSTNAME]' "/etc/hosts"
```

Set the Machine Timezone

```
sudo timedatectl set-timezone [TIMEZONE]
```

Set the Machine Timezone to UTC

```
sudo timedatectl set-timezone Etc/UTC
```

Generate Machine Locales

```
sudo locale-gen
```

Generate the Slovak Locale

```
sudo sed -i 's/^[#[:space:]]*sk_SK\.UTF-8 UTF-8$/sk_SK.UTF-8 UTF-8/'
"/etc/locale.gen"
sudo locale-gen
```

Generate the English Locale

```
sudo sed -i 's/^[#[:space:]]*en_US\.UTF-8 UTF-8$/en_US.UTF-8 UTF-8/'
"/etc/locale.gen"
sudo locale-gen
```

Tweak the Kernel Tunable Variables

TODO: add kernel variables to `/etc/sysctl.conf`

Time Synchronization Using NTP

[The Chrony Project](#) `chrony/chrony`

“ Debian uses `systemd-timesyncd` by default, which does not continuously synchronize the clock while the system is running. `chrony` provides continuous, periodical time synchronization via background daemon.

```
sudo apt install chrony
```

```
sudo systemctl restart chronyd
```

Bootstrapping a Debian Workstation

[Debian_Project](#) [Download Debian](#) [Debian](#)

Installation

[Bootstrapping a Debian Machine](#)

GNOME Desktop

[GNOME_Project](#) [GNOME](#)

Clean Up the Standard Installation

```
sudo apt purge gnome-clocks gnome-connections gnome-contacts gnome-maps gnome-  
music gnome-remote-desktop gnome-snapshot gnome-sound-recorder gnome-tour  
gnome-user-docs gnome-user-share gnome-weather \  
    libreoffice* lynx rygel shotwell simple-scan yelp
```

```
sudo apt --purge autoremove
```

Install Desktop Goodies

```
sudo apt install bmap-tools chromium firefox-esr rsync tcpdump tpm2-tools  
wakeonlan whois
```

Install Development Tools

```
sudo apt install git jq make pngquant podman python3-venv qrencode uuid xxd yq
```

```
sudo apt install cockpit cockpit-podman  
sudo mkdir -p "/etc/systemd/system/cockpit.socket.d"  
sudo tee "/etc/systemd/system/cockpit.socket.d/10-localhost.conf" >/dev/null  
<<'EOF'
```

```
[Socket]
ListenStream=
ListenStream=127.0.0.1:9090
EOF
```

```
curl --proto '=https' --tlsv1.2 -sSf "https://just.systems/install.sh" | sudo
bash -s -- --to "/usr/local/bin"
```

Configuration

Schedule Cron Jobs During Work Hours

```
17 * * * * root cd / && run-parts --report /etc/cron.hourly
25 14 * * * root test -x /usr/sbin/anacron || { cd / && run-parts --report
/etc/cron.daily; }
47 14 * * 2 root test -x /usr/sbin/anacron || { cd / && run-parts --report
/etc/cron.weekly; }
52 14 1 * * root test -x /usr/sbin/anacron || { cd / && run-parts --report
/etc/cron.monthly; }
```

Installing NVIDIA Graphics Drivers

[Official NVIDIA Drivers](#)

Installation

```
sudo sed -i 's/main/main non-free contrib non-free-firmware/g'
"/etc/apt/sources.list"
sudo apt update
```

```
sudo apt install linux-headers-$(uname --kernel-release) build-essential dkms
nvidia-detect
```

```
nvidia-detect
```

```
sudo apt install nvidia-driver nvidia-kernel-dkms
```

Debian Machine Maintenance

[Debian_Project](#)

Keeping the System Up to Date

```
sudo apt update
```

```
sudo apt upgrade
```

```
sudo apt autoremove  
sudo apt autoclean
```

Upgrade to a New Release

TODO: Write when Debian 14 releases.

Cleaning Up the APT Cache

```
sudo apt clean
```

Networking Cheat Sheet

Limitting NIC Auto-Negotiation

Cap Auto-Negotiation at 10 Mbps

```
sudo ethtool -s [NIC] autoneg on advertise 0x002
```

Cap Auto-Negotiation at 100 Mbps

```
sudo ethtool -s [NIC] autoneg on advertise 0x008
```

Cap Auto-Negotiation at 1000 Mbps

```
sudo ethtool -s [NIC] autoneg on advertise 0x3F
```

Security Hardening

Protecting GRUB with a Password

GRUB Project

Installation

```
sudo vi "/usr/local/sbin/grub-set-password"  
sudo chmod 0755 "/usr/local/sbin/grub-set-password"
```

```
#!/bin/sh  
set -eu  
  
AUTH_CFG="/boot/grub/auth.cfg"  
  
if [ "$(id -u)" -ne 0 ]; then  
    echo "This script must be run as root." >&2  
    exit 1  
fi  
  
printf 'Enter password: '  
stty -echo  
read -r password1  
stty echo  
printf '\n'  
  
printf 'Confirm password: '  
stty -echo  
read -r password2  
stty echo  
printf '\n'  
  
if [ "$password1" != "$password2" ]; then  
    echo "Passwords do not match, aborting." >&2  
    exit 1  
fi  
  
# Empty password (confirmed twice) disables protection entirely.
```

```

if [ -z "$password1" ]; then
    if [ -f "$AUTH_CFG" ]; then
        rm -f "$AUTH_CFG"
        echo "Empty password entered – GRUB password protection disabled."
    else
        echo "Empty password entered – nothing to do."
    fi
    exit 0
fi

hash=$(printf '%s\n%s\n' "$password1" "$password1" \
    | grub-mkpasswd-pbkdf2 \
    | sed -n 's/^\.*is \((grub\.pbkdf2\..*\)$/\1/p')

if [ -z "$hash" ]; then
    echo "Failed to generate password hash." >&2
    exit 1
fi

umask 077
cat > "$AUTH_CFG" <<EOF
GRUB_PASSWORD=$hash
EOF
chmod 0600 "$AUTH_CFG"

echo "GRUB password set. Run 'update-grub' to apply it."

```

```

sudo vi "/etc/grub.d/09_password"
sudo chmod 0755 "/etc/grub.d/09_password"

```

```

#!/bin/sh
exec tail -n +3 $0

if [ -f $prefix/auth.cfg ]; then
    source $prefix/auth.cfg
    if [ -n "${GRUB_PASSWORD}" ]; then
        set superusers="admin"
        password_pbkdf2 admin ${GRUB_PASSWORD}
        menuentry_id_option="--unrestricted $menuentry_id_option"
    fi
fi

```

Configuration

Having submenu poses a security risk as it allows bypassing password protection on sub-items.

```
GRUB_DISABLE_SUBMENU=true
```

Usage

```
sudo grub-set-password
```

```
sudo update-grub
```

Enforcing Password Quality

Installation

```
sudo apt install libpam-pwquality
```

Configuration

“ Passwords must be at least 8 characters long, include at least one uppercase letter, one lowercase letter, and one digit, this policy is enforced only for local users.

```
sudo sed -i 's/^[#[:space:]]*minlen[[:space:]]*=.*minlen = 8/'  
"/etc/security/pwquality.conf"  
sudo sed -i 's/^[#[:space:]]*dcredit[[:space:]]*=.*dcredit = -1/'  
"/etc/security/pwquality.conf"  
sudo sed -i 's/^[#[:space:]]*ucredit[[:space:]]*=.*ucredit = -1/'  
"/etc/security/pwquality.conf"  
sudo sed -i 's/^[#[:space:]]*lcredit[[:space:]]*=.*lcredit = -1/'  
"/etc/security/pwquality.conf"  
sudo sed -i 's/^[#[:space:]]*minclass[[:space:]]*=.*minclass = 3/'  
"/etc/security/pwquality.conf"  
sudo sed -i 's/^[#[:space:]]*enforcing[[:space:]]*=.*enforcing = 1/'  
"/etc/security/pwquality.conf"  
sudo sed -i 's/^[#[:space:]]*local_users_only[[:space:]]*$/local_users_only/'  
"/etc/security/pwquality.conf"
```

Verifying Installation Integrity

Installation

```
sudo apt install debsums
```

Configuration

```
sudo sed -i 's/^[#[:space:]]*CRON_CHECK=.*/CRON_CHECK=weekly/'  
"/etc/default/debsums"
```

Usage

Run Scan Manually

```
sudo debsums --changed
```

Tweaking a Debian Machine

Tweaking a Debian Machine

Tweaking GRUB

[GRUB Project](#)

Automatically Boot to Windows

```
GRUB_DEFAULT="Windows Boot Manager (on /dev/nvme0n1p1)"
```

Tweaking Bash

Disable Bash History

```
sudo tee "/etc/bash.bashrc" >/dev/null <<'EOF'  
unset HISTFILE  
export HISTSIZE=0  
export HISTFILESIZE=0  
EOF
```

How to Setup a...

Using KeePassXC as the Secret Service

[KeePassXC_Project keepassxreboot/keepassxc](#)

Installation

```
sudo apt install keepassxc
```

Create a D-Bus Service That Implements the Secret Service API

```
mkdir -p "$HOME/.local/share/dbus-1/services"
```

```
tee "$HOME/.local/share/dbus-1/services/org.freedesktop.secrets.service"
>/dev/null <<'EOF'
[D-BUS Service]
Name=org.freedesktop.secrets
Exec=/usr/bin/keepassxc
EOF
```

Disable the GNOME Keyring Daemon Service

```
systemctl --user mask gnome-keyring-daemon.socket
systemctl --user mask gnome-keyring-daemon.service
sudo systemctl mask gnome-keyring-daemon.socket
sudo systemctl mask gnome-keyring-daemon.service
```

```
mkdir -p "$HOME/.config/autostart/"
cp "/etc/xdg/autostart/gnome-keyring-pkcs11.desktop" "$HOME/.config/autostart/"
cp "/etc/xdg/autostart/gnome-keyring-secrets.desktop"
"$HOME/.config/autostart/"
cp "/etc/xdg/autostart/gnome-keyring-ssh.desktop" "$HOME/.config/autostart/"
echo "Hidden=true" >> "$HOME/.config/autostart/gnome-keyring-pkcs11.desktop"
```

```
echo "Hidden=true" >> "$HOME/.config/autostart/gnome-keyring-secrets.desktop"
echo "Hidden=true" >> "$HOME/.config/autostart/gnome-keyring-ssh.desktop"
```

“ Until an appropriate solution is found, resort to removing "executable" permission from `gnome-keyring-daemon`.

```
sudo chmod -x /usr/bin/gnome-keyring-daemon
```

```
sudo tee "/etc/apt/apt.conf.d/99-disable-gnome-keyring-daemon" >/dev/null
<<'EOF'
DPkg::Post-Invoke {
    "if [ -e /usr/bin/gnome-keyring-daemon ]; then chmod -x /usr/bin/gnome-
keyring-daemon; fi";
};
EOF
```

Unset GNOME Keyring as the Chromium Password Store

```
sudo tee --append "/etc/chromium.d/default-flags" >/dev/null <<'EOF'

# Disable GNOME keyring as password store (using keepassxc)
export CHROMIUM_FLAGS="$CHROMIUM_FLAGS --password-store=basic"
EOF
```

Configuration

- General
 - Basic Settings
 - Startup
 - ☒ Start only a single instance of KeePassXC
 - ☒ Automatically launch KeePassXC at system startup
 - ☒ Minimize window at application startup
 - ☐ Minimize window after unlocking database
 - ☒ Remember previously used databases
 - ☐ Load previously open databases on startup
 - ☒ Remember database key files and security dongles

Unlocking the Keyring Automatically

Create a KeePassXC Database Protected by Keyfile

```
mkdir -p --mode=700 "$HOME/.secrets/databases"
mkdir -p --mode=700 "$HOME/.secrets/private"
keepassxc-cli db-create --decryption-time 1000 --set-key-file
"$HOME/.secrets/private/SecretService.keyx"
"$HOME/.secrets/databases/SecretService.kdbx"
```

Encrypt the Keyfile Using the TPM2-Sealed Key

```
sudo systemd-creds encrypt --name=KeePassXC-SecretService --with-key=tpm2
"$HOME/.secrets/SecretService.keyx" "$HOME/.secrets/SecretService.creds"
sudo chown $USER:$USER "$HOME/.secrets/SecretService.creds"
sudo chmod 0400 "$HOME/.secrets/SecretService.creds"
```

⚠ Ensure you have a backup copy of the keyfile stored securely before shredding it — there is no recovery option if the TPM-sealed key becomes unusable (e.g. due to a firmware update).

```
shred "$HOME/.secrets/SecretService.keyx"
```

Authorize Credentials Decryption Using the TPM2-Sealed Key

```
sudo tee "/etc/polkit-1/rules.d/49-systemd-creds.rules" >/dev/null <<'EOF'
polkit.addRule(function(action, subject) {
    if (action.id == "io.systemd.credentials.decrypt" &&
        subject.local == true && subject.active == true &&
        subject.isInGroup("tss")) {
        return polkit.Result.YES;
    }
});
EOF
```

```
sudo gpasswd --add <username> tss
```

Set Up Automatic Unlock on Login

```
sudo wget --quiet -O "/usr/local/libexec/keepassxc-watch.sh"  
"https://kb.havlas.me/attachments/16"  
sudo chown root:root "/usr/local/libexec/keepassxc-watch.sh"  
sudo chmod 0755 "/usr/local/libexec/keepassxc-watch.sh"
```

```
sudo tee "/etc/systemd/user/keepassxc-watch@.service" >/dev/null <<'EOF'  
[Unit]  
Description=Auto-unlock KeePassXC database '%i' via TPM2-sealed key  
After=graphical-session.target  
ConditionPathExists=%h/.secrets/%i.kdbx  
ConditionPathExists=%h/.secrets/%i.creds  
  
[Service]  
ExecStart=/usr/local/libexec/keepassxc-watch.sh %i  
Restart=on-failure  
RestartSec=2  
  
[Install]  
WantedBy=graphical-session.target  
EOF
```

```
sudo chown root:root "/etc/systemd/user/keepassxc-watch@.service"  
sudo chmod 0644 "/etc/systemd/user/keepassxc-watch@.service"  
sudo systemctl daemon-reload
```

```
systemctl --user enable --now keepassxc-watch@SecretService.service
```

SSH Keys

```
sudo apt install seahorse  
sudo mkdir -p /usr/libexec/ssh  
sudo ln -s ../seahorse/ssh-askpass /usr/libexec/ssh/ssh-askpass
```

```
[Desktop Entry]  
Type=Application  
Name=SSH ask-pass  
Exec=/usr/libexec/ssh/ssh-askpass  
StartupWMClass=ssh-askpass  
NoDisplay=true
```

```
systemctl edit --user ssh-agent.service
```

```
[Service]
```

```
#Environment=DISPLAY=:0
```

```
Environment=SSH_ASKPASS=/usr/libexec/ssh/ssh-askpass
```

```
systemctl --user daemon-reload
```

```
systemctl --user restart ssh-agent.service
```

How to Setup a...

How to Setup a YubiKey Authentication

[Yubico](#) [Yubico documentation](#) [Buy at Alza.sk](#)

Installation

```
sudo apt install fido2-tools libpam-u2f pcscd
```

```
sudo apt install yubikey-manager yubikey-personalization
```

Configuration

Set YubiKey PIN

```
ykman fido access change-pin
```

Disable YubiKey OTP Protocol

```
ykman config usb --disable OTP
```

Setup GNOME/PAM Authentication

```
auth            sufficient    pam_u2f.so cue [cue_prompt=? Tap the security key  
now...] openasuser userpresence=1
```

```
@include common-yubico
```

```
@include common-yubico
```

```
auth      sufficient    pam_u2f.so nodetect cue [cue_prompt=? Tap the security  
key now...] openasuser
```

```
mkdir -p ~/.config/Yubico  
pamu2fcfg > ~/.config/Yubico/u2f_keys
```

Setup Full Disk Encryption

```
sudo apt install dracut dracut-core
```

```
sudo tee "/etc/dracut.conf.d/60hostonly.conf" >/dev/null <<'EOF'  
hostonly=yes  
hostonly_cmdline=yes  
EOF
```

```
sudo dracut --regenerate-all --force
```

```
sudo systemd-cryptenroll [PARTITION] --fido2-device=auto --fido2-with-client-  
pin=no
```

Troubleshooting

Missing Disk Space?

Check Deleted but Still Opened Files

```
lsof | grep '(deleted)'
```