angristan /
nixos-config

Code Issues Pull requests Actions Security Insights

nixos-config / configuration.nix

angristan git: disable push.default = current ee99358 · 6 years ago

805 lines (702 loc) · 22.6 KB

```
# Edit this configuration file to define what should be installed on
# your system.  Help is available in the configuration.nix(5) man page
# and in the NixOS manual (accessible by running ‘nixos-help’).
# ❄️

{ config, pkgs, lib, ... }:

{
  imports =
    [
      # Include the results of the hardware scan.
      ./hardware-configuration.nix
      # This is great for quick and easy config
      # But I have backported this into my own config
      #<nixos-hardware/dell/xps/13-9360>
      <home-manager/nixos>
    ];

  # The encrypted disk that should be opened before the root filesystem is
  boot.initrd.luks.devices = [
    {
      name = "root";
      device = "/dev/nvme0n1p2";
      # luksOpen will be attempted before LVM scan
      preLVM = true;
    }
  ];

  # Display ownership notice before LUKS prompt
  boot.initrd.preLVMCommands = ''
    echo '--- OWNERSHIP NOTICE ---'
    echo 'This device is property of Stanislas Lange'
    echo 'If lost please contact stanislas.lange at pm.me'
    echo '--- OWNERSHIP NOTICE ---'
  '';

  # systemd-boot
  # systemd-boot
  boot.loader.systemd-boot.enable = true;
  boot.loader.efi.canTouchEfiVariables = true;
  # Bigger console font
  boot.loader.systemd-boot.consoleMode = "2";
  # Prohibits gaining root access by passing init=/bin/sh as a kernel para
  boot.loader.systemd-boot.editor = false;
  # new!
  #boot.loader.systemd-boot.memtest86.enable = true;

  # Plymouth boot splash screen
  boot.plymouth.enable = true;

  # Clear /tmp during boot
  boot.cleanTmpDir = true;

  # https://wiki.archlinux.org/index.php/Kernel_mode_setting#Early_KMS_sta
  boot.initrd.kernelModules = [ "i915" ];
  # Enable framebuffer compression (FBC)
  # can reduce power consumption while reducing memory bandwidth needed fo
  # https://wiki.archlinux.org/index.php/intel_graphics#Framebuffer_compre
  boot.kernelParams = [ "i915.enable_fbc=1" ];

  # Use latest kernel
  # boot.kernelPackages will use linuxPackages by default, so no need to d
  nixpkgs.config.packageOverrides = in_pkgs :
    {
      linuxPackages = in_pkgs.linuxPackages_latest;
    };

  # No access time and continuous TRIM for SSD
  fileSystems."/".options = [ "noatime" "discard" ];

  # If non-empty, write log messages to the specified TTY device.
  services.journald.console = "/dev/tty12";

  # Enable microcode updates for Intel CPU
  hardware.cpu.intel.updateMicrocode = true;
  # Enable Kernel same-page merging
  hardware.ksm.enable = true;

  # Enable all the firmware
  hardware.enableAllFirmware = true;
  # Enable all the firmware with a license allowing redistribution. (i.e.
  hardware.enableRedistributableFirmware = true;

  # Enable OpenGL drivers
  hardware.opengl.enable = true;
  hardware.opengl.extraPackages = with pkgs; [
    vaapiIntel
    vaapiVdpau
    libvdpau-va-gl
  ];

  # Sysctl params
  boot.kernel.sysctl = {
    "fs.inotify.max_user_watches" = 524288; # Allow VS Code to watch more
  };

  # A DBus service that allows applications to update firmware
  services.fwupd.enable = true;

  # Check S.M.A.R.T status of all disks and notify in case of errors
  services.smartd = {
    enable = true;
    # Monitor all devices connected to the machine at the time it's being
    autodetect = true;
    notifications = {
      x11.enable = if config.services.xserver.enable then true else false;
      wall.enable = true; # send wall notifications to all users
    };
  };

  # Enable entropy daemon which refills /dev/random when low
  services.haveged.enable = true;


  # Add the NixOS Manual on virtual console 8
  services.nixosManual.showManual = true;

  networking.hostName = "nixpsla";
  networking.networkmanager.enable = true;

  # Simple stateful dual-stack firewall
  networking.firewall = {
    enable = true;
    allowPing = true;
    allowedTCPPorts = [];
    allowedUDPPorts = [];
    logRefusedConnections = true;
    checkReversePath = false; # for libvirtd
  };

  #networking.search = ["oxalide.local"];
  #networking.nameservers = ["192.168.3.2" "89.185.39.94" "176.103.130.130

  # The list of nameservers. It can be left empty if it is auto-detected t
  #networking.nameservers = [ "1.0.0.1" "1.1.1.1" ];

  # Network usage statistics
  services.vnstat.enable = true;

  i18n = {
    consoleFont = "latarcyrheb-sun32"; # Big console font for HiDPI
    consoleKeyMap = "fr";
    defaultLocale = "en_US.UTF-8";
    defaultLocale = "en_US.UTF-8";
  };

  time.timeZone = "Europe/Paris";

  # Use the systemd-timesyncd SNTP client to sync the system clock (enable
  services.timesyncd.enable = true;

  # Disable sudo password for the wheel group
  security.sudo.wheelNeedsPassword = false;

  # List packages installed in system profile.
  environment.systemPackages = with pkgs; [
    # Utils
    wget
    neofetch
    micro
    ncdu
    gparted
    ntfs3g
    ripgrep
    file
    htop
    speedtest-cli
    strace
    awscli
    gitAndTools.diff-so-fancy
    freerdp
    google-cloud-sdk
    vault
    exa
    lazygit
    bat
    libsForQt5.vlc
    gparted
    tree
    rsync
    openssl
    docker-compose
    nload
    sysbench
    geekbench
    psmisc  # provides: fuser, killall, pstree, peekfd
    ethtool
    lsof
    tokei  # fast cloc alternative in rust
    dos2unix  # Convert between DOS and Unix line endings
    socat
    ipcalc
    whois
    dnsutils
    iperf
    netcat
    nmap
    speedtest-cli
    openvpn
    networkmanager-openvpn
    ntfs3g
    pavucontrol # PulseAudio Volume Control, GUI
    #hyper
    # Nix tools
    nix-du #https://github.com/symphorien/nix-du
    common-updater-scripts
    nixops
    nix-review
    nix-universal-prefetch

    # Dev
    cmake
    bundix
    vscode
    nodejs-11_x
    ruby_2_6
    php73
        php73Packages.composer
    python27Full
    python37Full
    go_1_12
    sublime3
    shellcheck
    git
    solargraph # ruby tools
    rubocop
    gtk3
    gnome3.glade
    pkgconfig
    # Compiler and debugger
    gcc gdb
    # Build tools
    automake
    gnumake
    pkg-config
    clang-tools
    # Formatter
    indent
    # Linter
    splint
    # as (assembler) and ld, ld.bfd, ld.gold (linkers)
    binutils
    # Else
    google-chrome
    chromium
    spotify
    slack
    filezilla
    firefox
    ansible
    terraform_0_12
    vagrant
    tdesktop
    libreoffice
    gimp
    # Media
    plex-media-player
    # VM
    open-vm-tools
    # Hardware
    lshw
    usbutils
    pciutils
    dmidecode
    lm_sensors
    hdparm
    smartmontools
    p7zip
    privoxy

    # compression
    pixz pigz pbzip2 # parallel (de-)compression
    unzip
    # Data formatters, accessors
    libxml2  # xmllint
    jq  # json parser
    yq  # same for yaml
    nvme-cli
    _1password
    ffsend
    graphviz
    dpkg
    hexchat
    pidgin
    gitlab-runner
    exfat
    jnettop
    zip

    # https://www.mpscholten.de/nixos/2016/04/11/setting-up-vim-on-nixos.h
    (
      with import <nixpkgs> {};

      vim_configurable.customize {
        # Specifies the vim binary name
        # E.g. set this to "my-vim" and you need to type "my-vim" to open
        # This allows to have multiple vim packages installed (e.g. with a
        name = "vim";
        vimrcConfig.customRC = ''
          syntax on
          syntax enable
          syntax enable

          set backupdir=/tmp      " save backup files (~) to /tmp

          set tabstop=4           " number of visual spaces per TAB
          set softtabstop=4       " number of spaces in tab when editing
          set expandtab           " tabs are spaces
          filetype indent on      " load filetype-specific indent files
          filetype on             " Enable file type detection

          set number              " show line numbers
          set showcmd             " show command in bottom bar
          set cursorline          " highlight current line
          set wildmenu            " visual autocomplete for command menu
          set lazyredraw          " redraw only when we need to.
          set showmatch           " highlight matching [{()}]

          set incsearch           " search as characters are entered
          set hlsearch            " highlight matches
          colorscheme pablo
          set backspace=indent,eol,start " backspace over everything in in
        '';
      }
    )
  ];

  # Install + setcap
  programs.iftop.enable = true;
  programs.iotop.enable = true;
  programs.mtr.enable = true;

  # Thermals and cooling
  services.thermald.enable = true;
  # This includes support for suspend-to-RAM and powersave features on lap
  powerManagement.enable = true;
  # Enable powertop auto tuning on startup.
  powerManagement.powertop.enable = false;
  # IDK if TLP is useful/conflicts with powerManagement
  services.tlp.enable = false;
  services.tlp.extraConfig = "USB_AUTOSUSPEND=0";

  # Install and configure Docker
  virtualisation.docker = {
    enable = true;
    # Run docker system prune -f periodically
    autoPrune.enable = true;
    autoPrune.dates = "weekly";
    # Don't start the service at boot, use systemd socket activation
    enableOnBoot = false;
  };
  # Install LXD
  virtualisation.lxd.enable = true;
  # Install VB
  virtualisation.virtualbox.host.enable = true;
  # Libvirtd (Qemu)
  virtualisation.libvirtd.enable = true;

  # Periodically update the database of files used by the locate command
  services.locate.enable = true;

  # Enable Flatpak
  #services.flatpak.enable = true
  # No snap yet: https://github.com/NixOS/nixpkgs/issues/30336

  # Enable ClamAV, an open source antivirus engine
  #services.clamav.daemon.enable = true;
  # Enable ClamAV freshclam updater.
  #services.clamav.updater.enable = true;

  # Enable CUPS to print documents.
  # services.printing.enable = true;

  # Enable the OpenSSH daemon.
  #services.openssh.enable = true;

  # Monitoring
  services.netdata = {
   enable = true;
   config = {
     global = {
       "default port" = "19999";
       "bind to" = "127.0.0.1";
     };
   };
  };

  # Enable Pulseaudio
  hardware.pulseaudio = {
    enable = true;

    # NixOS allows either a lightweight build (default) or full build of F
    # Only the full build has Bluetooth support, so it must be selected he
    package = pkgs.pulseaudioFull;
  };

  # Bluetooth
  # https://nixos.wiki/wiki/Bluetooth
  hardware.bluetooth.enable = false;
  # Don't power up the default Bluetooth controller on boot
  hardware.bluetooth.powerOnBoot = false;

  # Enable the X11 windowing system.
  services.xserver.enable = true;
  services.xserver.layout = "fr";
  services.xserver.xkbOptions = "eurosign:e";
  # Enable touchpad support.
  services.xserver.libinput.enable = true;

  # GNOME
  services.xserver.desktopManager.gnome3.enable = true;
  services.xserver.displayManager.gdm.enable = true;
  # GDM uses wayland by default, but I don't want to
  services.xserver.displayManager.gdm.wayland = true;

  # Remove these packages that come by default with GNOME
  environment.gnome3.excludePackages = with pkgs.gnome3; [
    epiphany
    evolution
    gnome-maps
    accerciser
  ];

  # Enable GNOME Keyring daemon
  services.gnome3.gnome-keyring.enable = true;
  # Enable Chrome GNOME Shell native host connector
  # This is a DBus service allowing to install GNOME Shell extensions from
  services.gnome3.chrome-gnome-shell.enable = true;

  # this is required for mounting android phones
  # over mtp://
  services.gvfs.enable = true;

  # Disable mutable users.
  #users.mutableUsers = false;

  # Use fish by default for all users
  users.defaultUserShell = pkgs.fish;

  # Define a user account. Don't forget to set a password with ‘passwd’.
  users.users.stanislas = {
    description = "Stanislas";
    # This automatically sets group to users, createHome to true,
    # home to /home/username, useDefaultShell to true, and isSystemUser to
    isNormalUser = true;
    # Use fish as the default shell
    shell = pkgs.fish;
    extraGroups = [
      "wheel" # Enable ‘sudo’ for the user.
      "docker" "lxd" # Allow access to the sockets without root
      "libvirtd"
    ];
    # It is possible to install packages on a per-user basis.
    # I don't know why I would do that so they are installed globally for
    #packages = [];
  };

  home-manager.users.stanislas = { pkgs, ... }: {
    # home.packages = [ pkgs.atool pkgs.httpie ];
    # programs.bash.enable = true;
    programs.git = {
      enable = true;

      userName = "Stanislas Lange";
      userEmail = "stanislas.lange@fr.clara.net";

      aliases = {
        plog = "log --graph --abbrev-commit --decorate --format=format:'%(
      };

      # BEGING diff-so-fancy config
      extraConfig = {
        core = {
          pager = "diff-so-fancy | less --tabs=4 -RFX";
        };
        color = {
          ui = "true";
        };
        "color \"diff-highlight\"" = {
          oldnormal = "red bold";
          oldHighlight = "red bold 52";
          newNormal = "green bold";
          newHighlight = "green bold 22";
        };
        "color \"diff\"" = {
          meta = "yellow";
          frag = "magenta bold";
          commit = "yellow bold";
          old = "red bold";
          new = "green bold";
          whitespace = "red reverse";
        };
      };
      # END diff-so-fancy config

      ignores = [
        "*.swp"
        "*~"
        ".#*"
        ".DS_Store"
        ".direnv"
        ".vagrant"
      ];
    };

    programs.ssh = {
      enable = true;

      matchBlocks = {
        # Personal
        kokoro = {
          hostname = "kokoro.angristan.xyz";
          user = "root";
          port = 3200;
          identityFile = "~/.ssh/xps-sla";
        };
        mina = {
          hostname = "mina.angristan.xyz";
          user = "root";
          port = 3200;
          identityFile = "~/.ssh/xps-sla";
        };
        mitsuha = {
          hostname = "mitsuha.angristan.xyz";
          user = "root";
          port = 3200;
          identityFile = "~/.ssh/xps-sla";
        };
        "github.com" = {
          user = "git";
          identityFile = "~/.ssh/xps-sla";
          identitiesOnly = true;
        };
        "gitlab.com" = {
          user = "git";
          identityFile = "~/.ssh/xps-sla-oxa";
          identitiesOnly = true;
        };

        # Work
        pa1 = {
          hostname = "bastion-pa1-01";
          user = "slange";
          identityFile = "~/.ssh/xps-sla-oxa";
        };
        pa2 = {
          hostname = "bastion-pa2-01";
          user = "slange";
          identityFile = "~/.ssh/xps-sla-oxa";
        };
        pa3 = {
          hostname = "bastion-pa3-01";
          user = "slange";
          identityFile = "~/.ssh/xps-sla-oxa";
        };
        "oxalide.factory.git-01.adm" = {
          user = "slange";
          identityFile = "~/.ssh/xps-sla-oxa";
          identitiesOnly = true;
        };
      };
    };
    programs.htop = {
      enable = true;
      # Detailed CPU time (System/IO-Wait/Hard-IRQ/Soft-IRQ/Steal/Guest).
      detailedCpuTime = true;
    };

    # home.file."" = {
    #   text = ''

    #   '';
    # };
  };

  environment.variables.EDITOR = "vim";

  # Allow "unfree" packages.
  nixpkgs.config.allowUnfree = true;

  fonts = {
    enableDefaultFonts = true;
    enableFontDir = true;
    enableGhostscriptFonts = true;
    fontconfig.ultimate.enable = true;
    fonts = with pkgs; [
      noto-fonts
      noto-fonts-cjk # Chinese, Japanese, Korean
      noto-fonts-emoji
      noto-fonts-extra
      fira-code # Monospace font with programming ligatures
      hack-font # A typeface designed for source code
      fira-mono # Mozilla's typeface for Firefox OS
      corefonts  # Microsoft free fonts
      ubuntu_font_family
      roboto # Android
    ];
  };

  nix = {
    # Automatically run the garbage collector
    gc.automatic = false;
    gc.dates = "12:45";
    # Automatically run the nix store optimiser
    optimise.automatic = false;
    optimise.dates = [ "12:55" ];
    # Nix automatically detects files in the store that have identical cor
    autoOptimiseStore = true;
    # maximum number of concurrent tasks during one build
    buildCores = 4;
    # maximum number of jobs that Nix will try to build in parallel
    # "auto" is broken: https://github.com/NixOS/nixpkgs/issues/50623
    maxJobs = 4;
    # perform builds in a sandboxed environment
    useSandbox = true;
  };

  # This value determines the NixOS release with which your system is to b
  # compatible, in order to avoid breaking some software such as database
  # servers. You should change this only after NixOS release notes say you
  # should.
  # This does NOT define the NixOS version. The channel does.
  # https://nixos.wiki/wiki/FAQ#When_do_I_update_stateVersion
  system.stateVersion = "19.03"; # Did you read the comment?

  # This will run nixos-rebuild switch --upgrade periodically
  #system.autoUpgrade.enable = true;

    # Use the fish shell.
  programs.fish = {
    enable = true;

    shellAliases = {
      sysrs = "sudo nixos-rebuild switch";
      # Same as nix-channel --update nixos; nixos-rebuild switch
      sysup = "sudo nixos-rebuild switch --upgrade";
      sysrsgit = "sysrs -I nixpkgs=/home/stanislas/nixpkgs";
      sysupgit = "sysup -I nixpkgs=/home/stanislas/nixpkgs";
      nixpkgsupgit = "cd ~/nixpkgs/ && git fetch upstream && git checkout
      sysclean = "sudo nix-collect-garbage -d; and sudo nix-store --optimi
      lgit = "git add -A; and git commit; and git push";
      lgitf = "git add -A; and git commit; and git pull; and git push";
      cat = "bat -p";
      ls = "exa -gF --group-directories-first --git";
      ll = "ls -l";
      l = "ll -a";
      grep = "rg";
      rgrep = "grep";
    };

    shellInit = ''
      # Remove welcome message
      set fish_greeting

      export VAULT_ADDR=https://vault.oxalide.net
    '';

    promptInit = ''
      # Based on the "Classic" and "Informative Vcs" prompts

      function fish_prompt --description 'Write out the prompt'
        set -l last_status $status

        if not set -q __fish_git_prompt_show_informative_status
          set -g __fish_git_prompt_show_informative_status 1
        end
        if not set -q __fish_git_prompt_hide_untrackedfiles
        if not set -q __fish_git_prompt_hide_untrackedfiles
          set -g __fish_git_prompt_hide_untrackedfiles 1
        end

        if not set -q __fish_git_prompt_color_branch
          set -g __fish_git_prompt_color_branch magenta --bold
        end
        if not set -q __fish_git_prompt_showupstream
          set -g __fish_git_prompt_showupstream "informative"
        end
        if not set -q __fish_git_prompt_char_upstream_ahead
          set -g __fish_git_prompt_char_upstream_ahead "↑"
        end
        if not set -q __fish_git_prompt_char_upstream_behind
          set -g __fish_git_prompt_char_upstream_behind "↓"
        end
        if not set -q __fish_git_prompt_char_upstream_prefix
          set -g __fish_git_prompt_char_upstream_prefix ""
        end

        if not set -q __fish_git_prompt_char_stagedstate
          set -g __fish_git_prompt_char_stagedstate "●"
        end
        if not set -q __fish_git_prompt_char_dirtystate
          set -g __fish_git_prompt_char_dirtystate "✚"
        end
        if not set -q __fish_git_prompt_char_untrackedfiles
          set -g __fish_git_prompt_char_untrackedfiles "…"
        end
        if not set -q __fish_git_prompt_char_conflictedstate
          set -g __fish_git_prompt_char_conflictedstate "✖"
        end
        if not set -q __fish_git_prompt_char_cleanstate
          set -g __fish_git_prompt_char_cleanstate "✔"
        end

        if not set -q __fish_git_prompt_color_dirtystate
          set -g __fish_git_prompt_color_dirtystate blue
        end
        if not set -q __fish_git_prompt_color_stagedstate
          set -g __fish_git_prompt_color_stagedstate yellow
        end
        if not set -q __fish_git_prompt_color_invalidstate
          set -g __fish_git_prompt_color_invalidstate red
        end
        if not set -q __fish_git_prompt_color_untrackedfiles
          set -g __fish_git_prompt_color_untrackedfiles $fish_color_normal
        end
        if not set -q __fish_git_prompt_color_cleanstate
          set -g __fish_git_prompt_color_cleanstate green --bold
        end

        if not set -q __fish_prompt_normal
```

```
          set -g __fish_prompt_normal (set_color normal)
        end

        set -l suffix
        switch "$USER"
          case root toor
            if set -q fish_color_cwd_root
              set color_cwd $fish_color_cwd_root
            else
              set color_cwd $fish_color_cwd
            end
            set suffix '#'
          case '*'
            set color_cwd $fish_color_cwd
            set suffix '>'
        end

        echo -n -s "$USER" @ (prompt_hostname) ' ' (set_color $color_cwd)

        printf '%s ' (__fish_vcs_prompt)

        set_color normal
      end
    '';
  };

  programs.tmux = {
    enable = true;
    extraTmuxConf = ''
      # More history
      set -g history-limit 100000

      # Windows start at 1
      set -g base-index 1

      # Basic status bar colors
      set -g status-bg black
      set -g status-fg cyan
      set -g status-left-bg black
      set -g status-left-fg green
      set -g status-left-length 40
      #set -g status-left "Session #S #[fg=white]#[fg=yellow]Windows #I #
      set -g status-left "#S #[fg=white]#[fg=yellow]#I #[fg=cyan]#P"

      # Right side of status bar
      set -g status-right-bg black
      set -g status-right-fg cyan
      set -g status-right-length 40
      set -g status-right "#H #[fg=white]#[fg=yellow]%H:%M:%S #[fg=green]%

      # Window status
      set -g window-status-format " #I:#W "
      set -g window-status-current-format " #I:#W "

      # Current window status
      set -g window-status-current-bg red
      set -g window-status-current-fg black

      # Window with activity status
      set -g window-status-activity-bg yellow # fg and bg are flipped here
      set -g window-status-activity-fg black  # bug in tmux

      # Window separator
      set -g window-status-separator ""

      # Window status alignment
      set -g status-justify centre

      # Screen like binding
      set -g prefix C-a
      bind a send-prefix

      # Enable mouse mode
      set -g mouse on
    '';
  };

  # services.restic.backups.bastion_pa3 = {
  #   passwordFile = "/etc/nixos/secrets/restic-password";
  #   paths = [ "/etc" ];
  #   user = "stanislas";
  #   repository = "sftp:pa3:restic";
  #   timerConfig = {
  #     OnCalendar = "12:30";
  #   };
  # };
}
```

nixos-config / configuration.nix

↑ Top

Code Blame Raw