nix-bitcoin

robust Lightning nodes for hackers

github.com/fort-nix/nix-bitcoin

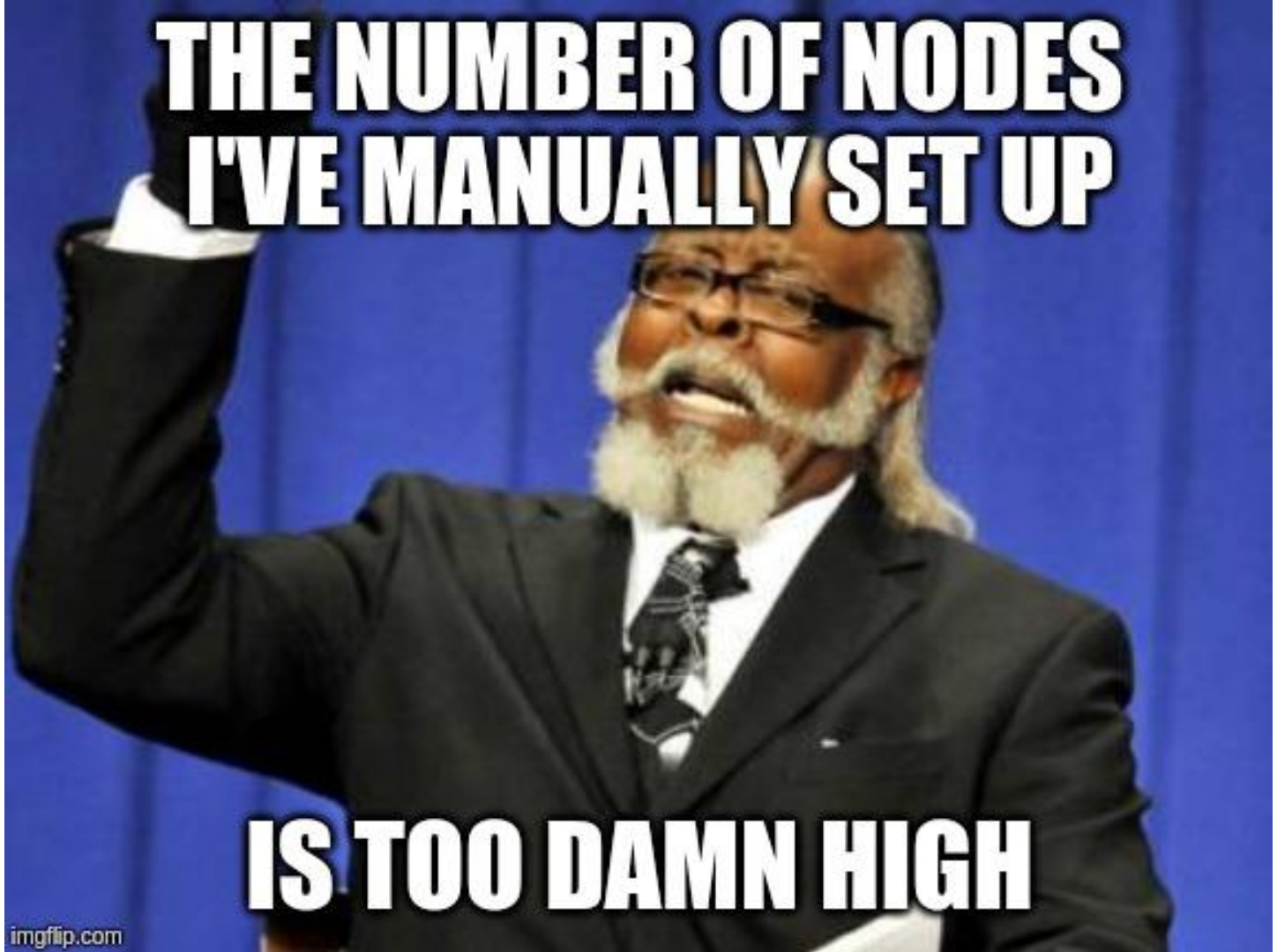
2019-06-01

@n1ckler

A smart home

A Bitcoin node

A lonely datacenter


THE NUMBER OF NODES
I'VE MANUALLY SET UP
IS TOO DAMN HIGH
imgflip.com

Robustness

- Do you trust binaries from some cache or do you build from source?
- Do you always check signatures?
- Do you isolate services and give least privileges?
- Do you minimize dependencies?
- Do you use a hardened kernel?
- Is your setup reproducible?
- Goal: want to do that once and for all

nix-bitcoin

nix-bitcoin configuration (text files)

nix tools

Bitcoin/Lightning/etc. node

Deployment

- Need something like: 4GB memory, CPU: Intel celeron, enough space
- There’s a tutorial for deploying virtual box in README.md
- Need machine to deploy *from* (right now x86 linux)
- `$ git clone https://github.com/fort-nix/nix-bitcoin.git`

“for hackers”: configuration.nix FIXMEs

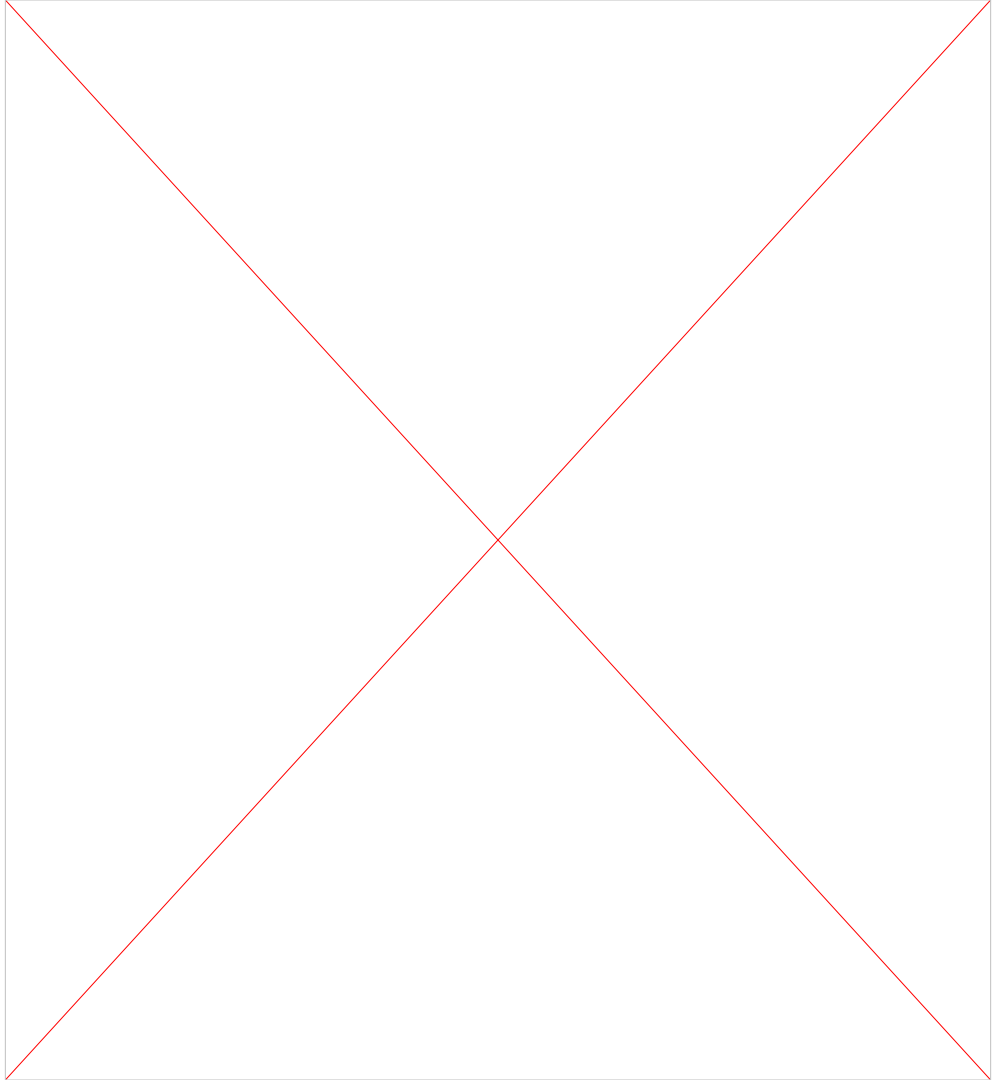
```
{ config, pkgs, ... }:
{
  imports = [
    ./modules/nix-bitcoin.nix
    # FIXME: Uncomment next line to import your hardware configuration.
    #./hardware-configuration.nix
  ];
  services.nix-bitcoin.enable = true;

  # FIXME: Define your hostname.
  networking.hostName = "nix-bitcoin";
  # FIXME: add packages you need in your system
  environment.systemPackages = with pkgs; [
    vim
  ];

  services.clightning.enable = true;
  # services.spark-wallet.enable = true;
  # services.liquid-daemon.enable = true;
```

nix-bitcoin modules

- bitcoind with reasonable default config (Tor-only, banlist)
- clightning with reasonable default config (Tor-only, not listening)
- spark-wallet
- recurring-donations
- bitcoin-core hardware wallet integration (HWI)
  - works with major hardware wallets
- liquid-daemon
- lightning charge & nanopos
- electrs (usable with electrum mobile app)
- ssh hidden service
- non-root user "operator"

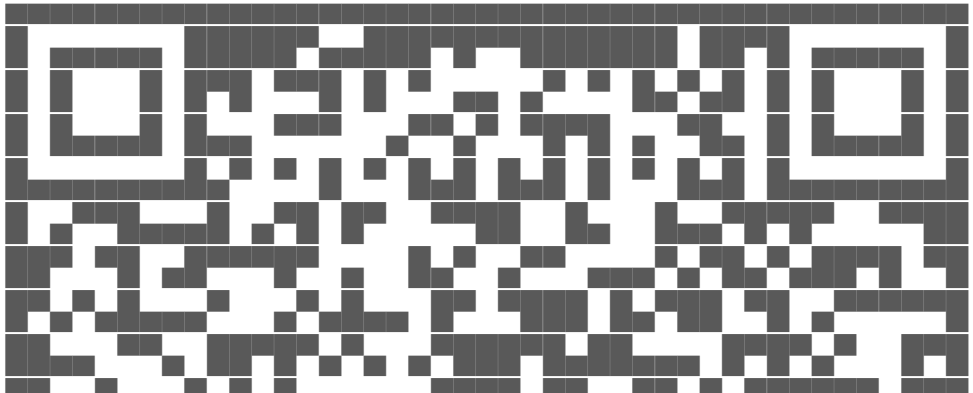


nodeinfo

```
[operator@nix-bitcoin:~]$ nodeinfo
BITCOIND_ONION=k7joisjlx5fjg77xcemqg6c5cprmslwhbcjuswlpdqwlvgvm6
CLIGHTNING_NODEID=0339984228019b57db117d1cbaec31df115098d6a08d19
CLIGHTNING_ONION=bsxeb3ucczmicamu6sec56bfal5cle2mwbnp5fgxeebpkxm
CLIGHTNING_ID=0339984228019b57db117d1cbaec31df115098d6a08d192ccb
LIQUIDD_ONION=qacupjhgo52otzer7r6pmfqe6lwuwqi5m2fj4bzvra7iiyd7ap
SPARKWALLET_ONION=http://rljtbxx33aew2ggokl3dfuiziwikmzyvjbsztpi
ELECTRS_ONION=fnguvt2rbzst5onvigwmv6vfarjqumsfd7yjva2x3fgqkphof3
SSHD_ONION=pox7b2cmajfevrik6kwyqpvz2k6tpflbyzhbxb5zt6i7golivthme
```

c-lightning + spark wallet + Android app + Orbot + Bitcoin Austrian

```
[root@nix-bitcoin:/var/lib/bitcoind]# journalctl -eu spark-wallet
Running /nix/store/hsy6797wclb2wv6nyk6sz1hnq789235k-node-spark-wallet-0.2.5/bin/sp
/var/lib/clightning -Q -k -c /secrets/spark-wallet-login --public-u>
Connected to c-lightning v0.7.0 with id 0339984228019b57db117d1cbaec31df115098d6a0
network bitcoin at /var/lib/clightning/lightning-rpc
Access key for remote API access: f8ufvzUnUu7mWY6EZQqonTXKalWfeIJTe89TmIUaRA
HTTP server running on http://rljtbxx33aew2ggokl3dfuiziwikmzyvjbsztpiogsngqrycew6g
Scan QR to pair with HTTP server:
```



Recurring Donations

- A module to repeatedly send lightning payments to recipients specified in the configuration.
- Very easy to do because we have full control over system (systemd timers)

```
 services.recurring-donations.enable = true;
# Specify the receivers of the donations. By default donations
# happen every Monday at a randomized time.
 services.recurring-donations.tallycoin = {
   "djbooth007" = 20000;
   "hillebrandmax" = 20000;
   "renepickhardt" = 20000;
 };
```

Hacking on nix-bitcoin

In search of a systematic approach

- Whole system config in a few text files and in version control
- Use abstractions to reduce complexity
- Reduced statefulness

The Nix ecosystem

- **Nix:** a purely functional package manager

The Nix ecosystem

- **Nix:** a purely functional package manager
- **NixOs:** a Linux distribution with a declarative approach to configuration management built on top of Nix

```
{ config, pkgs, ... }: {
  imports = [
    ./hardware-configuration.nix
  ];
  services.bitcoind.enable = true;
  services.bitcoind.port = 8333;
  services.tor.hiddenServices.bitcoind = {
    map = [{port = config.services.bitcoind.port;}];
  };
}
```

```
$ nixos-rebuild switch
```

The Nix ecosystem

- **Nix:** a purely functional package manager
- **NixOs:** a Linux distribution with a declarative approach to configuration management built on top of Nix
- **Nixpkgs:** collection of Nix packages and NixOs modules

NixOS / **nixpkgs**

Watch 149 | Star 3,508 | Fork 3,550

Code | Issues 2,931 | Pull requests 1,425 | Projects 15 | Insights

Nix Packages collection

nixpkgs nix nixos linux

174,391 commits **63** branches **90** releases **2,038** contributors MIT

Branch: **master** | New pull request | Find File | Clone or download

pmiddend and **dotlambda** python3Packages.black: add missing aiohttp-cors dependency (#58942) Latest commit `0f8bfed` an hour ago

...

.github	CODEOWNERS: I'd rather not review Haskell PRs any more	4 days ago
doc	Merge pull request #36886 from veprbl/symlinkJoin_doc	4 days ago
lib	Merge pull request #58330 from AerialX/msp430	5 days ago
maintainers	Merge pull request #58588 from shazow/fix/vlc	2 days ago
nixos	Merge pull request #57337 from peterhoeg/m/logitech	4 hours ago
pkgs	python3Packages.black: add missing aiohttp-cors dependency (#58942)	an hour ago

The Nix ecosystem

- **Nix:** a purely functional package manager
- **NixOs:** a Linux distribution with a declarative approach to configuration management built on top of Nix
- **Nixpkgs:** collection of Nix packages and NixOs modules
- **NixOps:** declarative tool for deploying sets of NixOS Linux machines

```
{
  bitcoin-node =
    { config, pkgs, ... }:
    {
      deployment.targetEnv = "virtualbox";
      deployment.virtualbox.memorySize = 4096; # in MB
      deployment.virtualbox.vcpu = 2;
      deployment.virtualbox.headless = true;
    };
}
```

```
$ nixops create -d my-new-network network.nix
$ nixops deploy -d my-new-network
```

There must be a more systematic approach

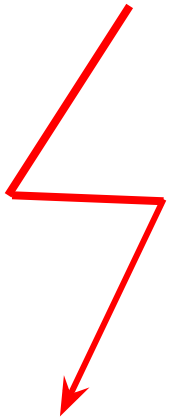
- Whole system config in a few text files and in version control
- Use abstractions to reduce complexity
- Reduced statefulness
- Using Nix
  - deployment und update with single command (`nixops deploy`)
  - Reproducibilty for ease of use and security
  - uses standard linux tools under the hood
  - simple functional, typed language

```
{ config, pkgs, ... }: {
  imports = [
    ./hardware-configuration.nix
  ];
  services.bitcoind.enable = true;
  services.bitcoind.port = 8333;
  services.tor.hiddenServices.bitcoind = {
    map = [{port = config.services.bitcoind.port;}];
  };
}
```

```
$ nixos-rebuild switch
```

```
{ config, pkgs, ... }: {
  imports = [
    ./hardware-configuration.nix
  ];
  services.bitcoind.enable = true;

  services.tor.hiddenServices.bitcoind = {
    map = [{port = config.services.bitcoind.port;}];
  };
}
```



```
$ nixos-rebuild switch
```

Customizations

- Change/uncomment nix-bitcoin options in `configuration.nix`
- Check available module options in `modules/` and add to `configuration.nix`
  - For example

```
services.bitcoind.prune = 120000;
services.bitcoind.dbCache = 4000;
Services.clightning.bind-addr = "127.0.0.1:9735";
```

- If option is not available, open an issue in the nix-bitcoin github repo or define it yourself

```
{ config, lib, pkgs, ... }:
let
  configFile = pkgs.writeText "config" ''
    autolisten=${if cfg.autolisten then "true" else "false"}
  '';
in {
  options.services.clightning = {
      ...
    autolisten = mkOption {
      type = types.bool;
      default = false;
      description = ''
        If enabled, the clightning service will listen.
      '';
    };};
  config = mkIf cfg.enable {
    systemd.services.clightning = {
      wantedBy = [ "multi-user.target" ];
      after = [ "bitcoind.service" ];
      serviceConfig = {
        ExecStart = "${pkgs.clightning}/bin/lightningd --lightning-dir=${cfg.dataD
        User = "clightning";
      };};};};
```

Conclusion

- Flexible: just a personal wallet or platform for bitcoin and layer 2+ protocols as public infrastructure
- Please develop more software
- Go to github.com/fort-nix/nix-bitcoin and follow the tutorial. I'm here to help (can be tried out on VirtualBox for example).
- Let's open some channels