

Minecraft

Client

Launcher for running the mods

I use the Fabric launcher, get its installer from the AUR with the following command:

```
$ aurget fabric-installer
```

Since this is a graphical installer, launch and select the Minecraft version that it should install a launcher for. It will appear as an entry in the Mojang launcher.

Installer for the mods themselves

Install either `fabric` or `fabric-bin`. I typically use the latter since I get build issues with the former.

Follow instructions and we are good.

List of mods that I run on the client:

- fabric-api
- sodium
- lithium
- phosphor
- effective
- visuality
- iris
- ferrite-core
- coordinates-display
- lambdynamicalights
- betterf3
- enhancedblockentities (ebe)
- AppleSkin
- spark
- immediatelyfast
- sound-physics-remastered
- exordium

- better-animations-collection
-

Server

Architecture

Firewalld Forwarding Setup

I run the actual server on `krypton`, but use `oxygen` to proxy the traffic, masking the actual IP address. Use port forwarding through `firewalld` to achieve this:

```
# firewall-cmd --permanent --zone=public --add-forward-  
port=port=25565:proto=tcp:toport=25565:toaddr=<addr of target server>  
# firewall-cmd --permanent --zone=public --add-forward-  
port=port=25565:proto=udp:toport=25565:toaddr=<addr of target server>  
# firewall-cmd --permanent --zone=internal --add-masquerade  
# firewall-cmd --reload
```

Add a masquerade to the zone where the target server is located at. This will allow traffic to pass from the public zone to that target zone, which in this case is `internal`.

Traefik

I run the actual server on `krypton`, but use `oxygen` to proxy the traffic, masking the actual IP address. Use a TCP router through `traefik` with `HostSNI(`*`)` to allow access to the server.

Here is an example configuration using the file provider on traefik:

```
tcp:  
  routers:  
    minecraft:  
      entrypoints:  
        - "minecraft"  
      rule: "HostSNI(`*`)"  
      service: minecraft  
  
  services:  
    minecraft:
```

```
loadBalancer:
  servers:
    - address: "/path/to/server/host:port"
```

Mods

- simply-optimized modpack
- no-chat-reports
- no-telemetry
- krypton
- spark
- ledger

Whitelisting

This server is exposed to the public internet, whitelisting is required to prevent bots and unauthorized players from joining the server

The whitelist file is a JSON-formatted file that contains player usernames and UUIDs. That in combination with online mode prevents attackers from spoofing accounts.

Here is the file structure for the player whitelist:

```
[
  {
    "name": "<mojang username 1>",
    "uuid": "<uuid 1>"
  },
  {
    "name": "<mojang username 2>",
    "uuid": "<uuid 2>"
  },
]
```

To retrieve the UUID of a user, run the following command:

```
curl --silent https://api.mojang.com/users/profiles/minecraft/<mojang username>
```

This command will send a JSON response with "name" and "id" attributes, don't forget to change "id" to "uuid" for whitelisting to work properly.

The UUID will be returned in the HTTP response.

To enable parsing and managing of this whitelist:

```
ENABLE_WHITELIST="true"
ENFORCE_WHITELIST="true"
WHITELIST_FILE="/path/to/whitelist/file.json"
EXISTING_WHITELIST_FILE="SYNC_FILE_MERGE_LIST"
```

Server Ops

Similar to the whitelist file, the ops file is in JSON format as seen below:

```
[
  {
    "name": "<mojang username 1>",
    "uuid": "<uuid 1>",
    "level": 4,
    "bypassesPlayerLimit": true
  },
  {
    "name": "<mojang username 2>",
    "uuid": "<uuid 2>",
    "level": 3,
    "bypassesPlayerLimit": false
  },
]
```

```
OPS_FILE="/path/to/ops/file.json"
EXISTING_OPS_FILE="SYNC_FILE_MERGE_LIST"
```

Revision #37

Created 2 January 2024 14:51:36

Updated 5 November 2024 15:58:58