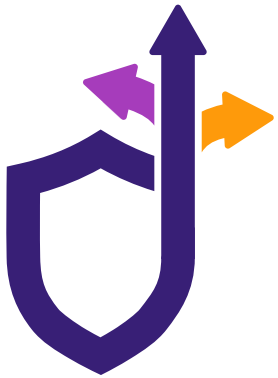
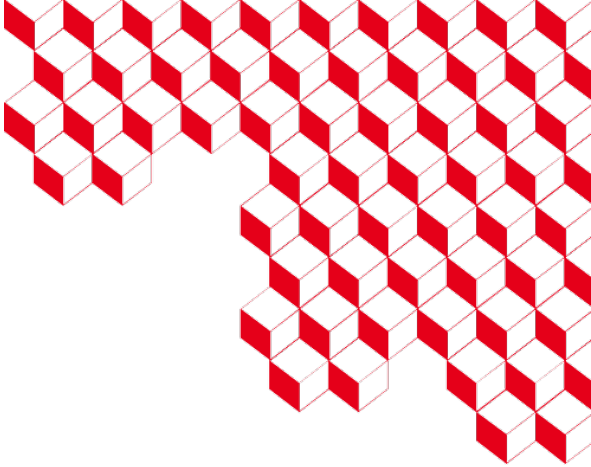




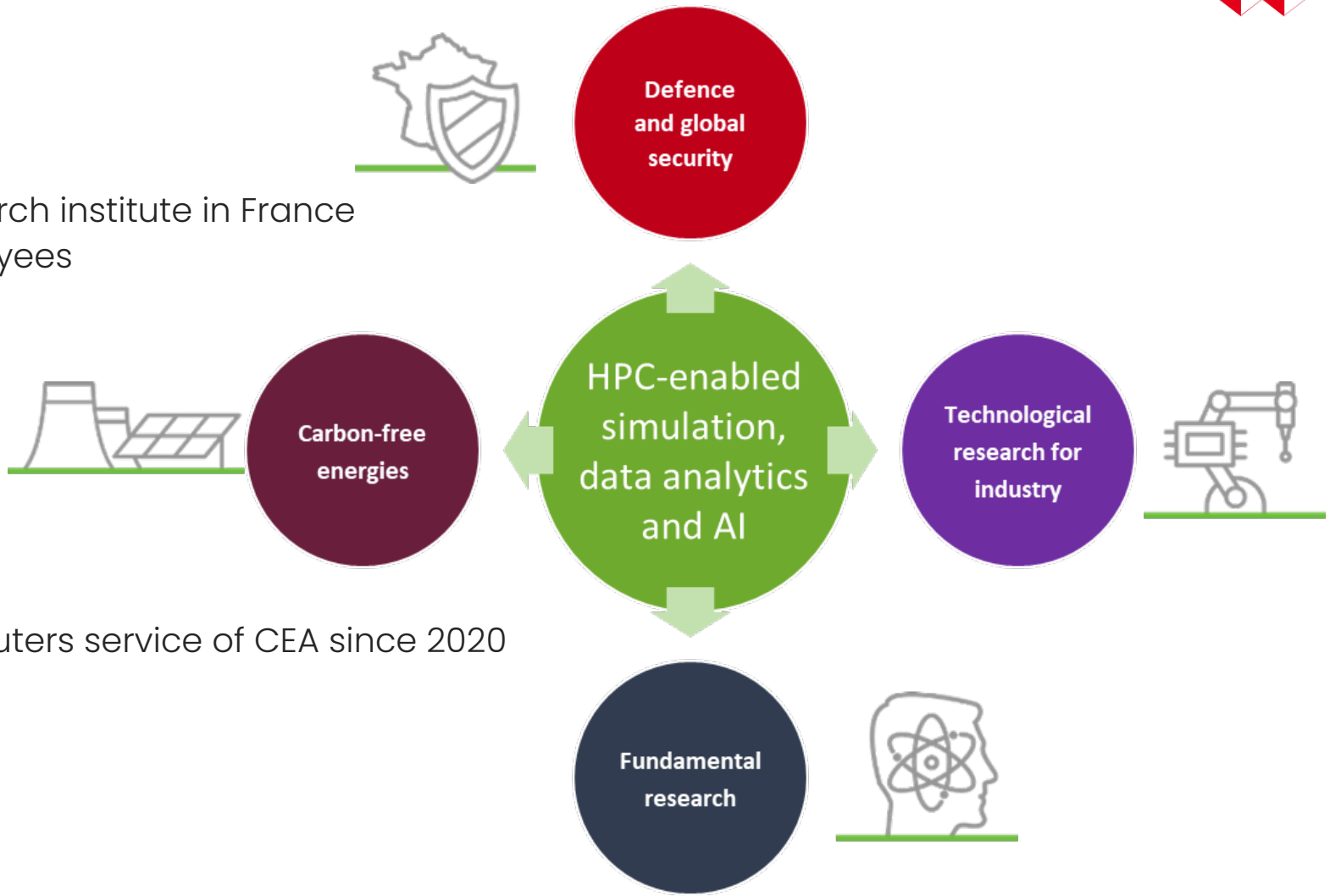
sshproxy: how to load-balance ssh

Cyril Servant, HPC engineer at CEA
https://github.com/cea-hpc/sshproxy

February 2025

Preamble

- **CEA**
  - National research institute in France
  - ~20,000 employees



- **Cyril Servant**
  - At supercomputers service of CEA since 2020

The need

- Each CEA supercomputer has multiple login nodes
- Users are on public networks, login nodes are on private networks
- We needed a solution to allow the connection and load-balancing of thousands of users
- **SSH** (Secure SHell): popular protocol for remote command-line connections and file transfers

The solution

- sshproxy created in 2014 by Arnaud Guignard
- Simple and powerful, fast and scalable
- Written in **go**, only needs one **yaml** configuration file
- Handles
  - ~100,000 daily connections
  - ~1,000 simultaneous connections on average
- Ready for 100 Gb/s throughput

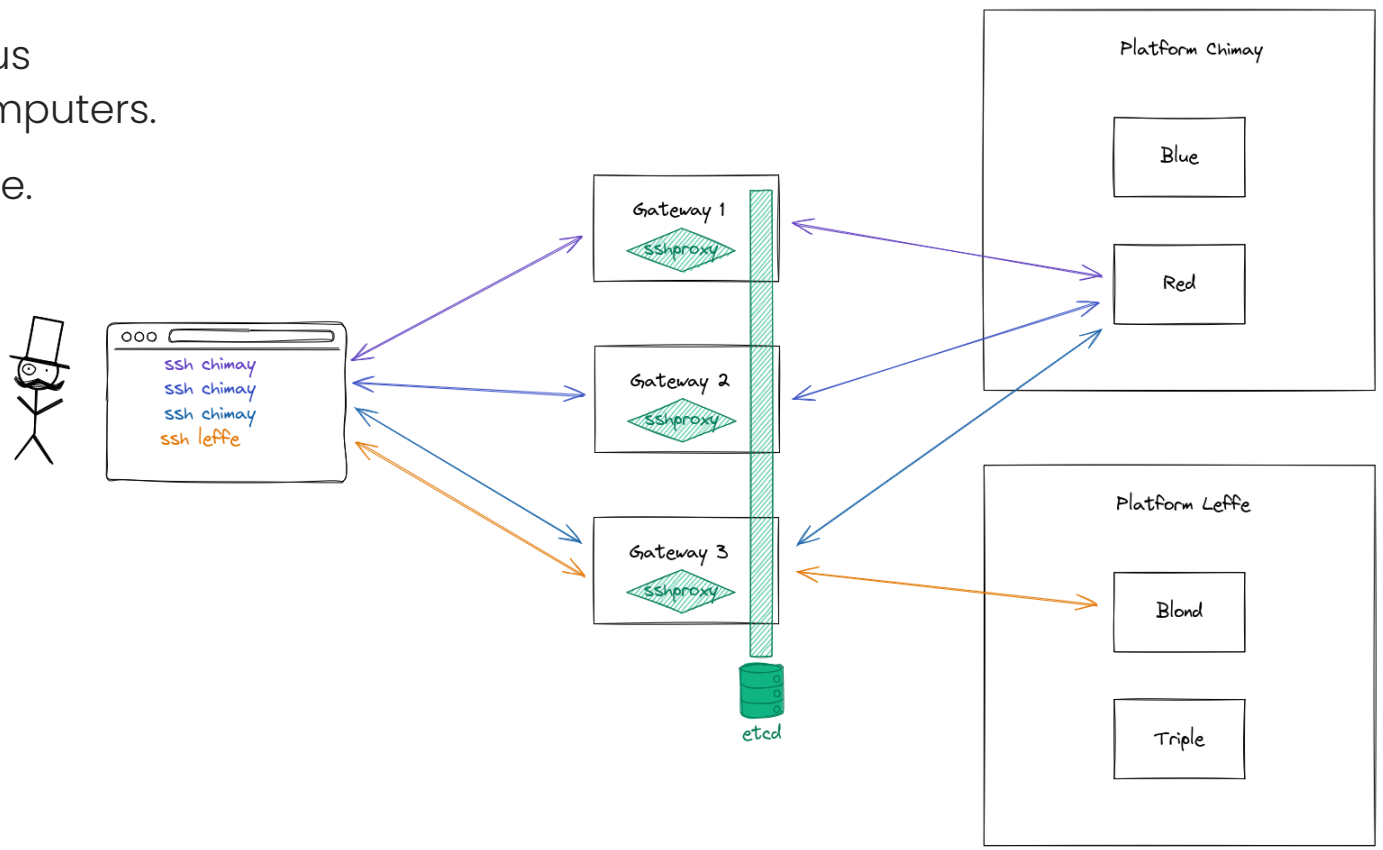
1 - Real life example

How is sshproxy used at CEA

Typical way of using sshproxy at CEA

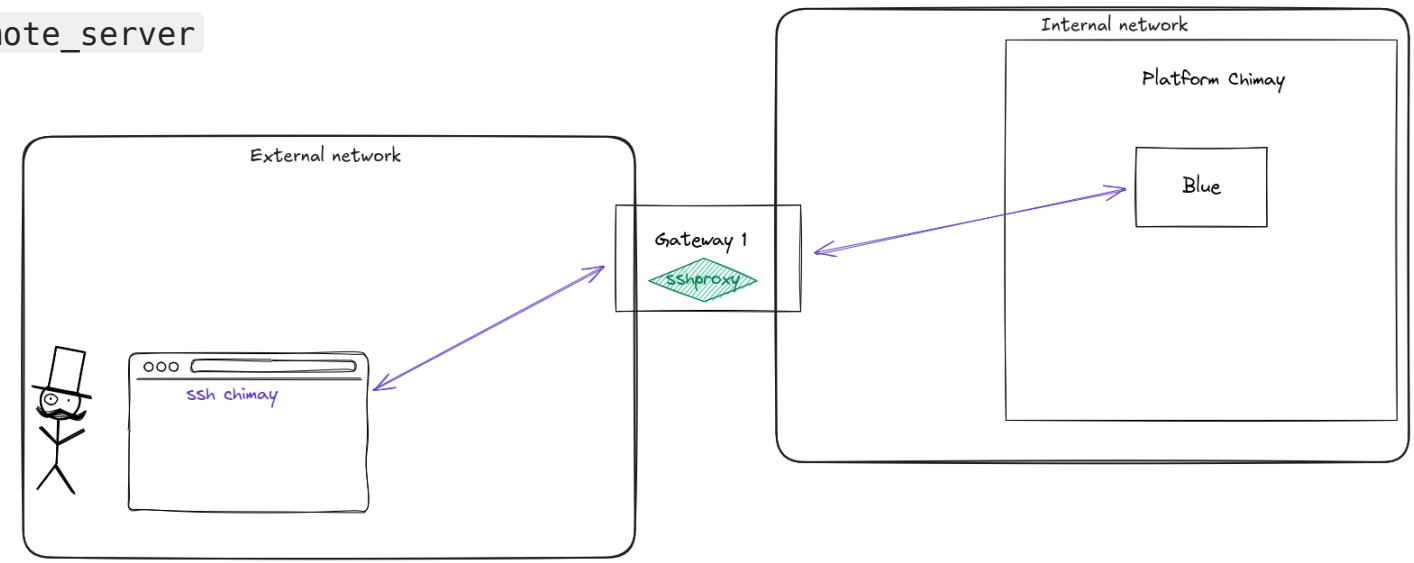
With only 3 gateways, we can handle thousands of simultaneous connections to multiple supercomputers.

Let's dive into the configuration file.



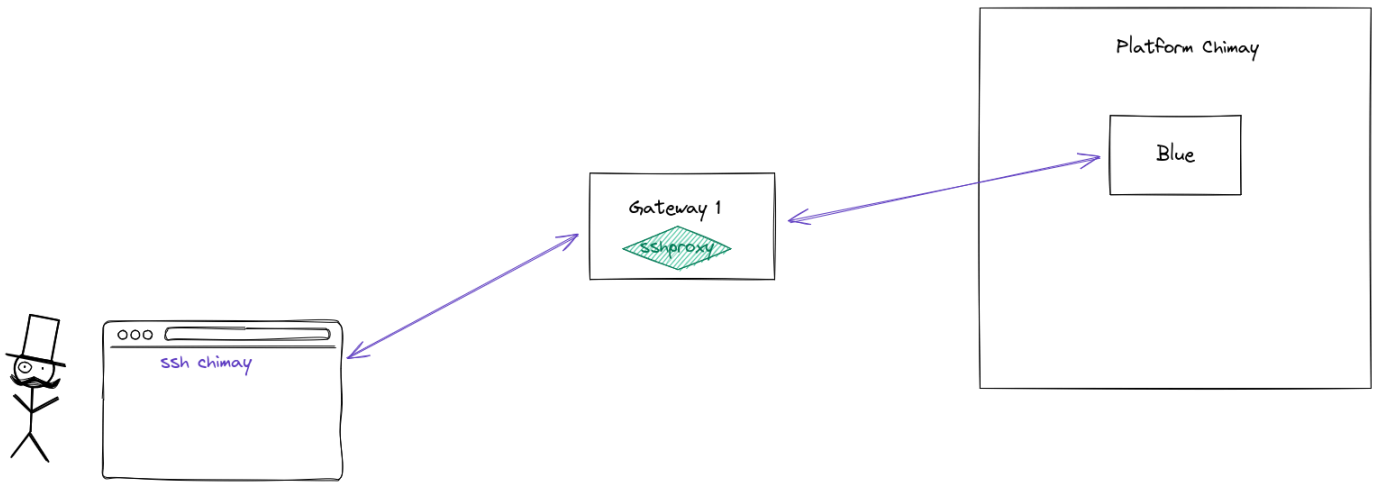
Just an intermediary

1. User connects with `ssh my_remote_server`
2. On the gateway, openssh authentifies the user
3. openssh launches sshproxy (`ForceCommand` directive)
4. sshproxy chooses a destination
5. sshproxy connects the user to the destination with `ssh destination`
6. User is now logged into `destination`. All the previous steps are totally transparent



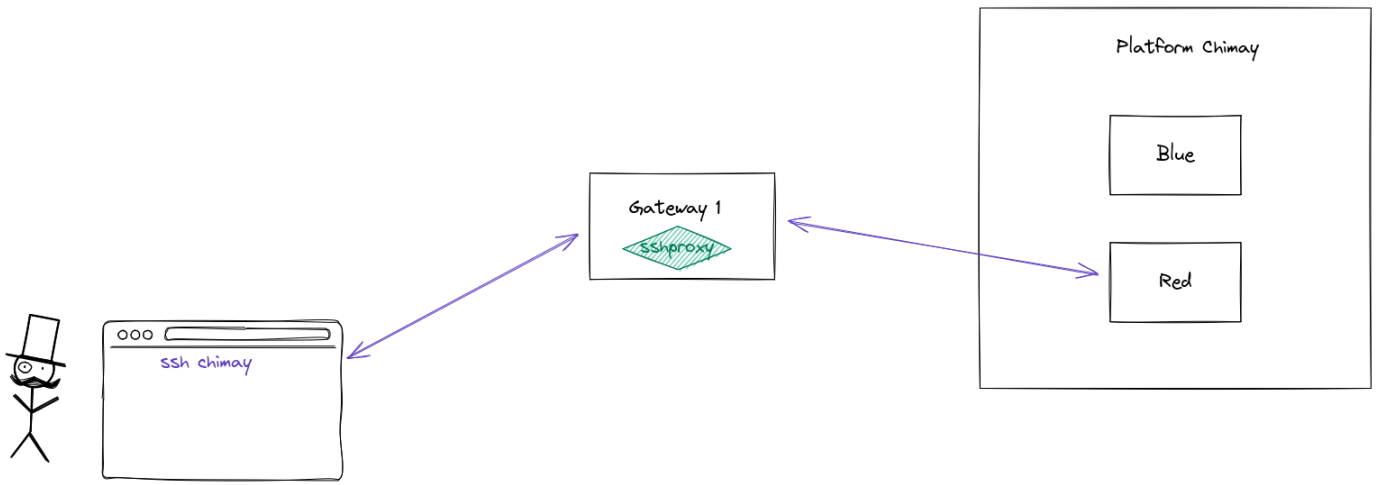
Purpose: authent, encrypt, routing, scaling

```
service: chimay
dest: [blue]
```



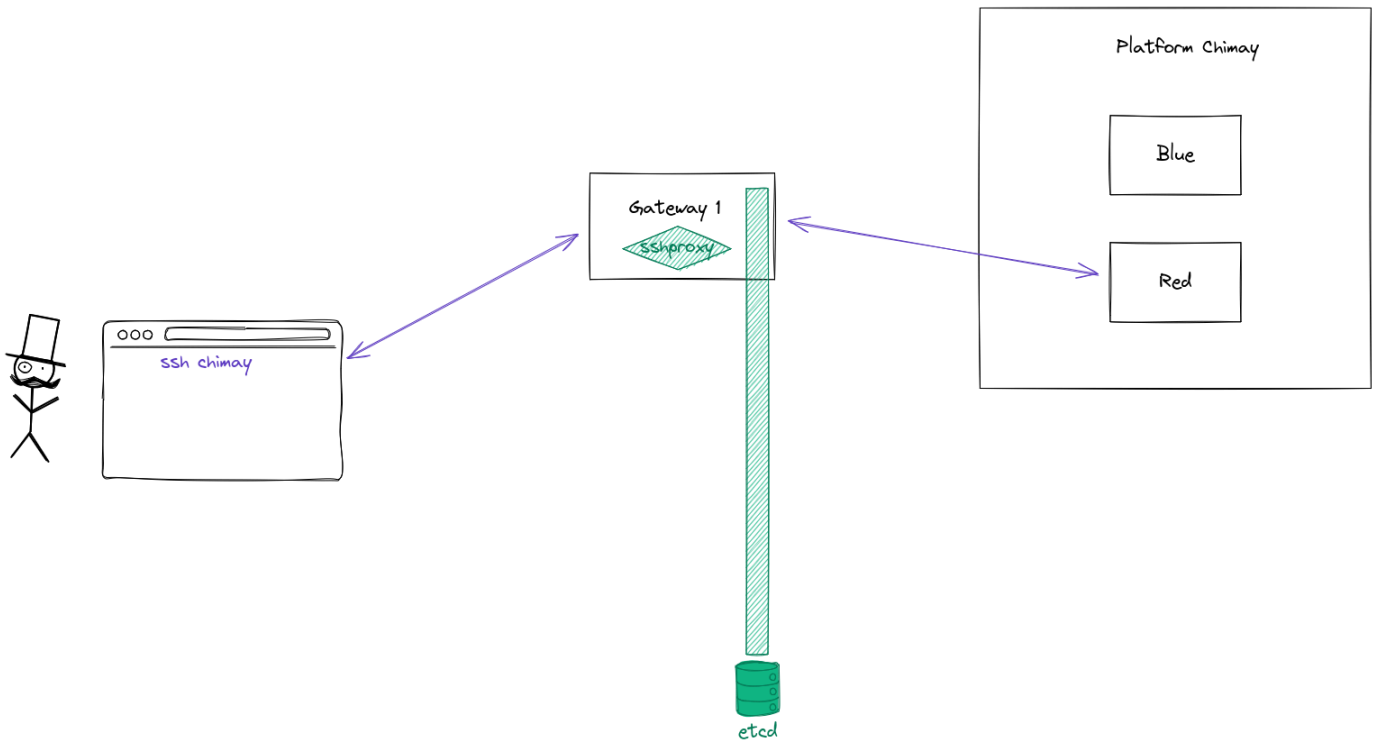
Distribution of users

```
service: chimay
dest: [blue, red]
route_select: random
```



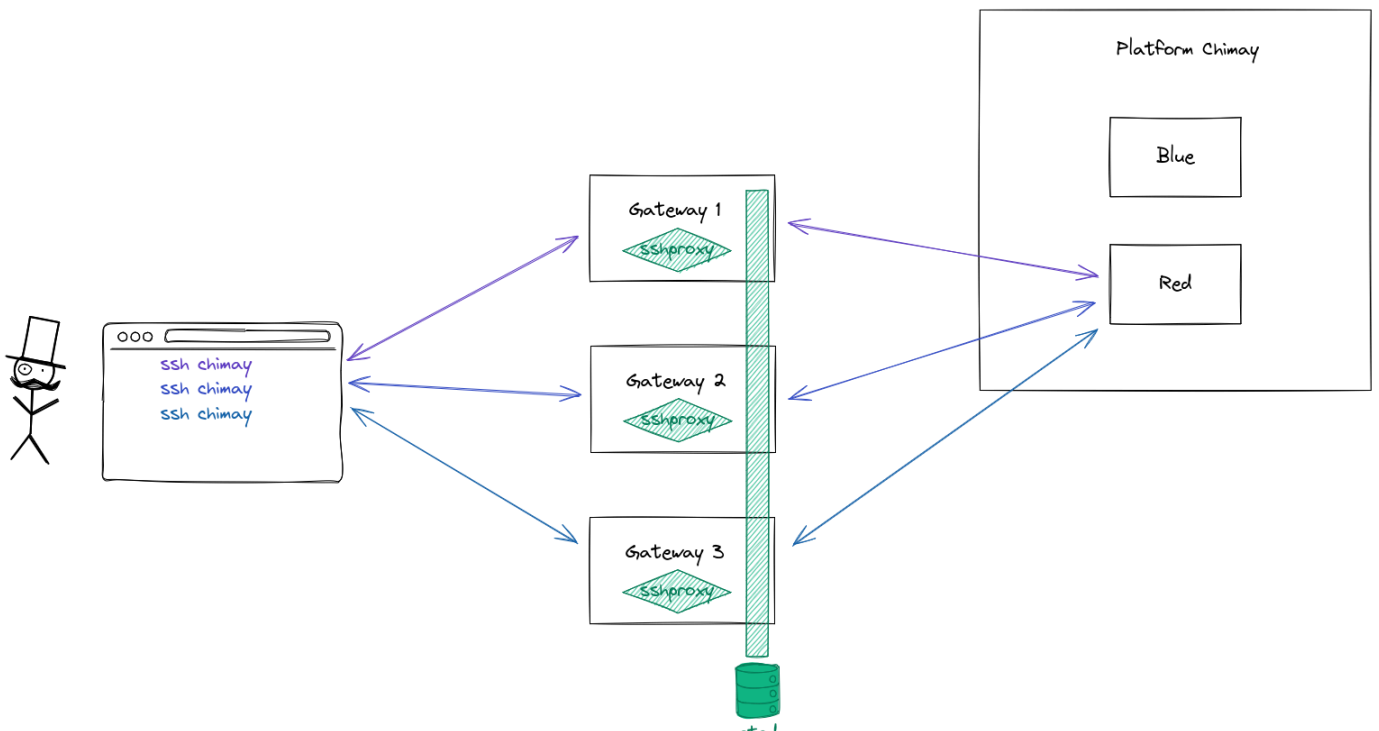
A stateful proxy

```
etcd:
    endpoints:
        - "localhost:2379"
service: chimay
dest: [blue, red]
route_select: random
```



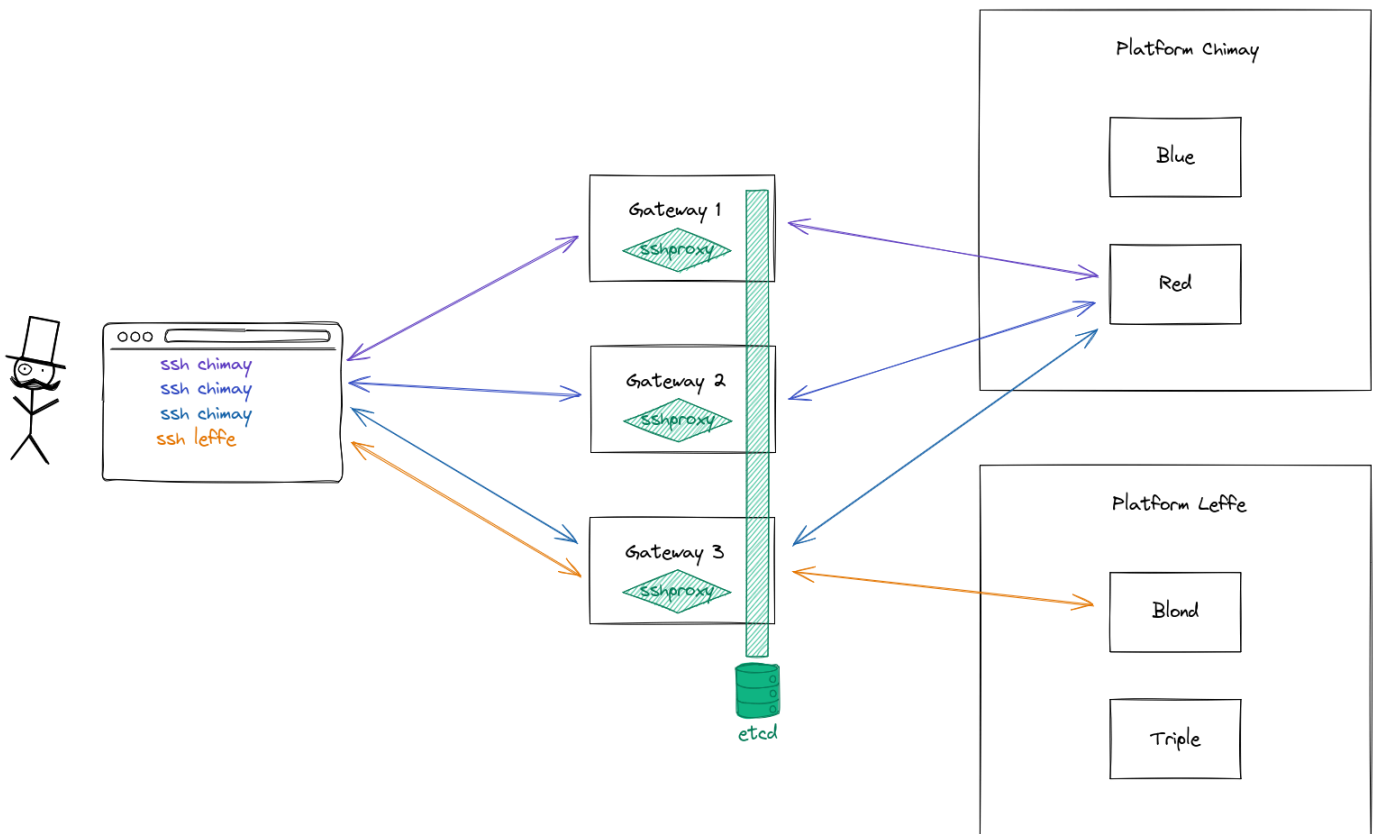
A distributed proxy

```
etcd:
    endpoints:
        - "localhost:2379"
service: chimay
dest: [blue, red]
route_select: random
```



Have more than 1 platform?

```yaml
etcd:
    endpoints:
        - "localhost:2379"
service: chimay
dest: [blue, red]
route_select: random
overrides:
    - match:
        - sources:
            - "ip.leffe.gw1"
            - "ip.leffe.gw2"
            - "ip.leffe.gw3"
      service: leffe
      dest: [blond, triple]
```



2 · Advanced features

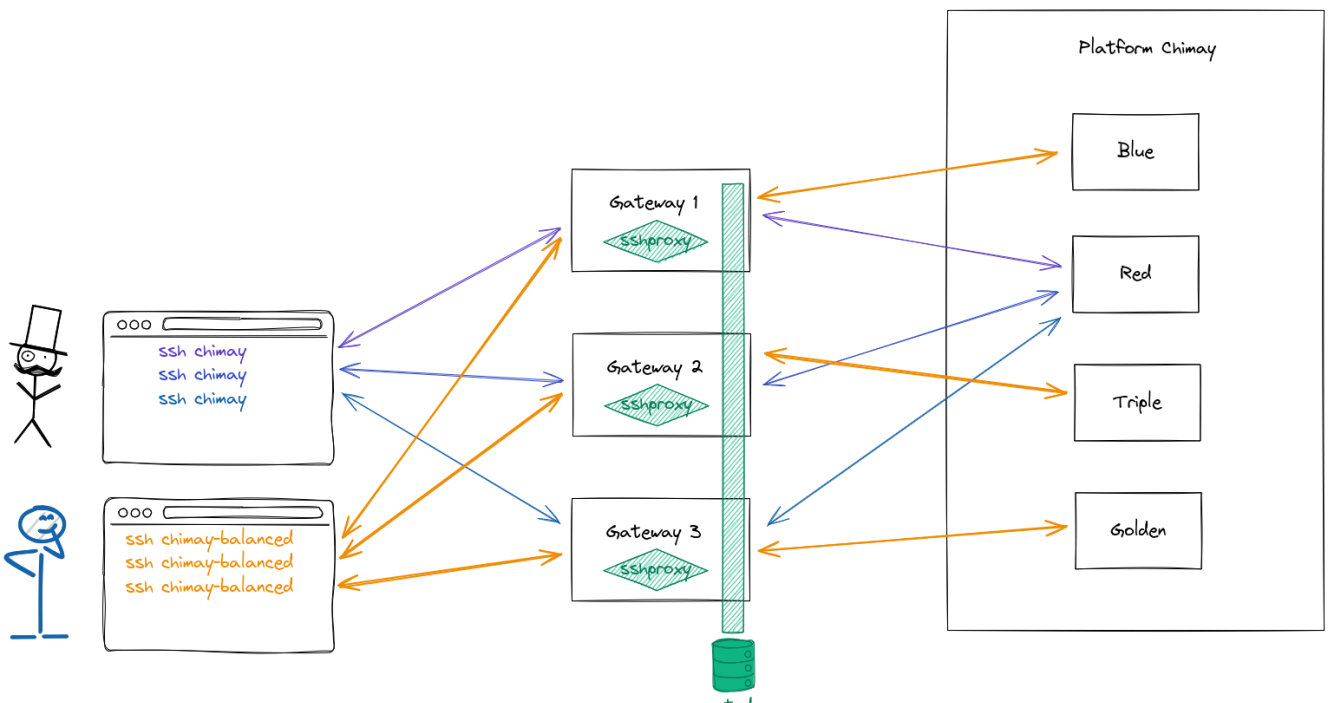
A wide range of options

Connections management

Parameter	Value	Description
`route_select`	`random`	Destination randomly chosen
	`connections`	Destination with less active connections is chosen
	`bandwidth`	Destination with less used bandwidth is chosen
`mode`	`balanced`	A user's connections are split on different destinations
	`sticky`	All connections of a single user -> same destination
`max_connections_per_user`	integer	Maximum number of connections allowed per user

Sticky versus balanced

```yaml
etcd:
  endpoints:
    - "localhost:2379"
service: chimay
dest: [blue, red, triple, golden]
mode: sticky # default
route_select: random
overrides:
  - match:
    - sources:
      - "ip.chimay.balanced.gw1"
      - "ip.chimay.balanced.gw2"
      - "ip.chimay.balanced.gw3"
    service: chimay-balanced
    mode: balanced
    route_select: connections
```



More features: monitoring

- sshproxy monitors number of connections and used bandwidth: `etcd_stats_interval`
- `sshproxyctl show…` displays informations about service, destination, user, connection…
- `sshproxyctl` is scriptable (`-json` and `-csv` options)
- Example:

```
# sshproxyctl show connections
  User   | Service |  Destination    | # of connects |   Last connection    |  Bw in    | Bw out
--------+---------+-----------------+---------------+----------------------+-----------+---------
 cyril  | chimay  | 192.168.0.1:22 |             5 | 2025-01-14 10:51:56 | 0 kB/s    | 0 kB/s
 cyril  | leffe   | 192.168.1.1:22 |             1 | 2025-01-14 07:30:02 | 180 kB/s | 11 MB/s
 arnaud | chimay  | 192.168.0.2:22 |             3 | 2025-01-14 10:00:10 | 0 kB/s    | 0 kB/s
```

- Also possible to change the state of destination nodes with `sshproxyctl disable…`

A powerful overrides system

- Starting with version 2.0 (ETA june 2025), all parameters can be overridden: `overrides`
- For a given service: `match sources`
- For a given group / user: `match users / groups`
- For a combination of service / group / user (OR / AND)
- Overrides are all applied, from top to bottom
- Example:

```
overrides:
    - match: # cyril OR arnaud OR group sparkling
        - users: [cyril, arnaud]
        - groups: [sparkling]
      dest: [perrier, badoit]
    - match: # group sparkling AND group yeast
        - groups: [sparkling]
          groups: [yeast]
      dest: [blue, red, triple, golden]
```

Many more features

- Command handling
  - Force a command at connection time: `force_command`
  - Reject a connection if the requested command is not the correct one: `command_must_match`
  - Modify a command at connection time: `translate_commands`
  - Set environment variables: `environment`
- Error banner
  - This banner is displayed to users if no login is available
  - Default error banner: `error_banner`
  - Override the error banner for a set period of time: `sshproxyctl error_banner`

Thanks for your attention

Cyril Servant

cyril.servant@cea.fr

https://github.com/cea-hpc/sshproxy