

Batfish cheat sheet

Batfish builds vendor independent models from vendor configs. The models cover configuration settings as well as network behaviors such as packet forwarding and translation. Batfish questions enable you to query the models and ensure correct network behavior even before configuration is deployed.

Install

```
$ docker pull batfish/allinone
$ docker run -v batfish-data:/data -p 8888:8888 -p
9997:9997 -p 9996:9996 batfish/allinone
$ python3 -m pip install --upgrade
git+https://github.com/batfish/pybatfish.git
```

Python imports

```
>>> from pybatfish.client.commands import *
>>> from pybatfish.question.question import load_questions
>>> from pybatfish.question import bfq
>>> load_questions()
```

Analyze network snapshots

```
# Snapshot packaging instructions and examples
>>> bf_init_snapshot("/path/to/snapshot")

# Ask a question and get a Pandas dataframe
>>> answer = bfq.nodeProperties().answer()
>>> answer_df = answer.frame()

# See all columns and pull out values in a column.
>>> df.columns
>>> answer_df["NTP_Servers"]
```

Batfish questions

Configuration data

[nodeProperties](#)

Device-wide configuration settings

[interfaceProperties](#)

Configuration settings of interfaces

[ipOwners](#)

Where IP addresses are attached

[bgpProcessConfiguration](#)

[bgpPeerConfiguration](#)

Settings related to BGP

[ospfProcessConfiguration](#)

[ospfInterfaceConfiguration](#)

[ospfAreaConfiguration](#)

Settings related to OSPF

[mlagProperties](#)

MLAG configuration settings

[switchedVlanProperties](#)

Settings of switched VLANs

[vxlanVniProperties](#)

Settings of VXLAN VNIs

[f5BigipVipConfiguration](#)

Settings of VIPs in F5 Big IP

[definedStructures](#)

Structures defined in the configuration

[referencedStructures](#)

Structures referenced in configurations

[viModel](#)

Get the full vendor-independent model

Network adjacencies

[edges](#)(edgeType=Layer1)

[edges](#)(edgeType=Layer2)

[edges](#)(edgeType=Layer3)

Network edges at different layers

[edges](#)(edgeType=BGP)

[edges](#)(edgeType=EIGRP)

[edges](#)(edgeType=ISIS)

[edges](#)(edgeType=OSPF)

[edges](#)(edgeType=RIP)

Routing protocol adjacencies

[edges](#)(edgeType=IPSec)

Configured IPSec tunnels

[edges](#)(edgeType=VXLAN)

VXLAN adjacencies

Configuration compatibility

[bgpSessionCompatibility](#)

[bgpSessionStatus](#)

BGP peering session compatibility

[ospfSessionCompatibility](#)

Compatibility of OSPF configuration

[ipsecSessionStatus](#)

Compatibility of IPSec tunnels

Flow path analysis

[traceroute](#)

All paths of a flow from its source

[bidirectionalTraceroute](#)

All forward and reverse flow paths

ACL and firewall analysis

[testFilters](#)

Test how a filter (ACL) treats a packet

[searchFilters](#)

Find packets that are permitted or denied by a filter

[compareFilters](#)

Find how a filter differs across two snapshots

[filterLineReachability](#)

Find lines that will not match any packet

Analyze routing

[routes](#)

[lpmRoutes](#)

Output RIBs

[testRoutePolicies](#)

Test how a routing policy treats a route

Search across all flows

[reachability](#)

Find flows matching path and header criteria

[detectLoops](#)

Find flows that will loop

[multipathConsistency](#)

Find flows treated differently along different paths

[differentialReachability](#)

Find flows treated differently in two snapshots



Configuration hygiene

[undefinedReferences](#)

References to undefined structures

[unusedStructures](#)

Defined but not used structures