No-hop

Initialize System

October 4, 2021

Contents

1 Package initialize_system

1.1 Modules

- **No_hop_Network** *(Section 2, p. 3)*
- **Write_Jsons** *(Section 3, p. 4)*

1.2 Variables

Name	Description
__package__	**Value:** `None`

2 Module initialize_system.No_hop_Network

2.1 Variables

Name	Description
rewrite_build_folders	1 overwrite existing folder if exists, 0 make new folder **Value:** `1`
node_scalar	**Value:** `500`
edge_width	**Value:** `15`

2.2 Class network

Class for building initial state of No_hop system

2.2.1 Methods

__**init**__(*self*, *max_id*=`32`, *nh_type*=`"forward"`, *rewrite_switch*=`"a"`)

generate_random_keys(*self*, *amount*=`8`)

Generates <amount> many keys randomly between the values of 0 and <network.max_id> with a distance between ids of atleast 2 returns a sorted list.

define_ports(*self*)

Assign each connection network ports and set corresponding port as taken in corresponding port lists/dicts

draw_network(*self*)

draws network to pdf in network folder

make_new_folder(*self*, *folder_name*=`0`)

Builds empty folder with unused name or rewrites build folder if rewrite_build_folders==1

find_reachables(*self*)

Finds the reachable ranges for switches if only traversing the tree downwards returns labels, weights. labels a printable version, weights the cleaned ranges.

clean_ranges(*self*, *ranges*)

Takes switch many dict of ranges and cleans each list of ranges to be represented in the minimal amount of ranges. returns labels(dict), weights(dict). labels a printable version, weights the cleaned ranges.

host_range(*self*, *id*)

Find the range id host is respondible for.

3 Module initialize_system.Write_Jsons

3.1 Functions

make_no_hop_tables(*network*)

For all switches generate no_hop tables as dictionaries returns dict: {switch: switch_no_hop_table}

make_no_hop_table(*network*, *switch*)

For switch, generate no_hop table returns list of entries

range_size(*range_list*)

calculate total size of range_list

make_single_no_hop_table_entry(*port*, *range*, *group_id*=1)

Make a single no hop table entry that sends range(tuple) out of port (int) and matches to group_id(int, default=1) returns entry as dict

switch_connections(*network*, *switch*)

Return a dictionary {connection target: Outgoing port} for all connections that switch is in

formalize_switch(*switch*, *s*)

Create single entry for a switch in topology.json

formalize_switches(*switches*)

Create all entries for the switches in the topology.json

formalize_connection_names(*network*)

Bring connections into a printable/ readable form for mininet

formalize_connections(*network*)

Prepare connections for printing to JSON

write_topology_file(*network*)

write topology.json for network

make_rewrite_entry(*range*, *ip*)

Make single No_hop_rewrite table entry

make_rewrite_table(*network*, *hosts*)

Generate rewrite table for no-hop_rewrite rewrite switch

write_build_files(*network*)

Generate and write all build files for system

write_switch_json(*table*, *switch*, *network*)

Write switch info and table to to json

make_host_entry(*ip_count*)

Generrate JSON entry for host for topology.json

make_host_data(*host_ids*)

Define host information for topology.json

make_ip_lpm_table(*network*, *hosts*)

Generate IPv4 tables

make_lpm_entry(*switch*, *s*, *next_c*, *connections*, *connection_ports*, *switches*, *host*)

Make single IPv4 lpm table entry

find_spot(*entry*, *list*)

return index of entry in list

Index