NAME

yarrp — high-speed active IPv4/IPv6 network topology prober

SYNOPSIS

yarrp [**-hvQT**] [**-i** *target_file*] [**-o** *outfile*] [**-r** *rate*] [**-t** *tr_type*]
[**-c** *tr_count*] [**-S** *seed*] [**-E** *instance*] [**-p** *dst_port*] [**-b** *bgp_rib*]
[**-B** *blocklist*] [**-l** *min_ttl*] [**-m** *max_ttl*] [**-F** *fill_ttl*] [**-n** *nbr_ttl*]
[**-s** *sequential*] [**-Z** *poisson*] [**-a** *ipv6_src*] [**-I** *interface*] [**-M** *src_mac*]
[**-G** *dst_mac*] [*subnet(s)*]

DESCRIPTION

yarrp (Yelling at Random Routers Progressively) is a high-speed active traceroute-style network topology discovery tool. To achieve its high probing rates, **yarrp** is stateless and randomizes the order of probed destinations and TTLs. By spreading probes, **yarrp** distributes load and attempts to avoid network rate-limiting. Yarrp supports both IPv4 and IPv6 and can send probes of any transport type (TCP, UDP-paris, or ICMP-paris).

OPTIONS

The set of IPv4 or IPv6 destination targets to probe may be specified in one of three ways:

subnet(s)
: Probes a target in each /24 (IPv4), or each /48 (IPv6), of the specified subnets.

-i *target_file*
: Input list (one address per line) of explicit targets; accepts stdin.

-Q
: Internet-wide scanning. Probes an address in each /24 (IPv4) or each /48 (IPv6) (use with caution).

The general options are as follows:

-h
: print command line options and a synopsis of each.

-v
: verbose (use multiple times to increase verbosity)

-T
: test mode (default: off)

-o *outfile*
: output file for probing results; accepts stdout. (default: output.yrp)

-r *rate*
: set packet per second probing rate (default: 10pps)

-t *tr_type*
: set probe type: TCP_ACK, TCP_SYN, UDP, ICMP, ICMP_REPLY (default: TCP_ACK)

-c *tr_count*
: set number of traces to issue (default: unlimited)

-S *seed*
: set permutation random seed (default: timestamp)

-E *instance*
: set instance (default: 0)

-p *dst_port*
: use specified transport destination port (default: 80)

The target options are as follows:

-b *bgp_rib*
: read BGP RIB (Potaroo text format) (default: none)

-B *blocklist*
: read list of prefixes to skip (default: none)

The options to control TTLs probed are:

-l *min_ttl*
set minimum TTL (default: 1)

-m *max_ttl*
set maximum TTL, must be a power of 2 (default: 16)

-F *fill_ttl*
set fill mode maximum TTL (default: 32)

-s send probes sequentially (default: random)

-n *nbr_ttl*
enable neighborhood enhancement and set local neighborhood TTL (default: off)

-Z *poisson*
choose TTLs from a Poisson distribution with specified lambda (default: uniform)

The IPv6-specific options are as follows:

-I *interface*
network interface to use (required)

-t *tr_type*
set probe type: ICMP6, UDP6, TCP6_SYN, TCP6_ACK (required)

-a *ipv6_src*
set source IPv6 address (default: auto)

-M *src_mac*
MAC address of source (required if auto discovery fails)

-G *dst_mac*
MAC address of gateway router (required if auto discovery fails)

OUTPUT

yarrp writes probe responses to the specified output file in a delimited ASCII format as they are received, one response per line. Because **yarrp** randomizes its probing, results will be similarly randomized. To determine all of the responses for a single target destination, it is necessary to filter and collate responses. The included yrp2warts.py python utility performs this reconstitution and produces output in the standard warts binary format.

TTLs

By default, **yarrp** randomly permutes the space of targets and TTLs, thereby probing each target with TTLs from min_ttl to max_ttl in a random order. Note that because of the way **yarrp** permutes the probe order, max_ttl must be a power of two.

Four options modify this behavior. The sequential option (-s) disables random probing and instead probes sequentially. The nbr_ttl option (-n) is an optimization that stops probing low TTLs within the local neighborhood of the prober once **yarrp** determines that it is not discovering any new interfaces within that neighborhood. In fill mode (-F), **yarrp** will probe, up to a maximum TTL of fill_ttl, the next hop beyond max_ttl if it receives a response for a probe with TTL greater than or equal to max_ttl.

Finally, the -Z option specifies a lambda parameter for a Poisson distribution. **yarrp** will iterate through all TTLs, but the probability of probing a particular TTL follows a Poisson distribution with the given lambda. This mode is intended to maximize router discovery yield, as the majority of Internet routers are concentrated in a particular TTL range.

EXAMPLES

The command:

yarrp -i targets -o test.yrp -r 100

will send TCP_ACK topology probes in a randomly-permuted order to the IPv4 targets in file "targets" at a rate of 100pps, and write results to file "test.yrp".

The command:

yarrp -o scan.yrp -t ICMP -v -m 16 205.155.0.0/16

will send ICMP topology probes in a randomly-permuted order to all destinations within the prefix 205.155.0.0/16, from TTL 1 to 16 at the default rate of 10pps. Verbosity is switched on so that **yarrp** will report probe and response data to stdout. The results will be written to the file "scan.yrp".

The command:

yarrp -o scan2.yrp -t ICMP -b bgptable.txt 1.0.0.0/8

will send ICMP topology probes in a randomly-permuted order to all destinations within the prefix 1.0.0.0/8, if the destination has a route in the BGP routing table "bgptable.txt". The routing table file must be plain-text in Potaroo format (the most recent table is available from https://bgp.potaroo.net/as6447/bgptable.txt). The results will be written to the file "scan2.yrp".

The command:

yarrp -t UDP6 -I eth0 -i targets6 -o test6.yrp

will send UDP probes in a randomly-permuted order to the set of IPv6 targets in the file "targets6", and write the results to the file "test6.yrp".

SEE ALSO

`yrp2warts.py`(1) `warts2yrp.py`(1)

R. Beverly, *Yarrp'ing the Internet: Randomized High-Speed Active Topology Discovery*, Proc. ACM/SIGCOMM Internet Measurement Conference 2016.

R. Beverly, R. Durairajan, D. Plonka, and J.P. Rohrer, *In the IP of the Beholder: Strategies for Active IPv6 Topology Discovery*, Proc. ACM/SIGCOMM Internet Measurement Conference 2018.

AUTHORS

yarrp is written by Robert Beverly <rbeverly@cmand.org>. Ionut Luculescu contributed support for IPv4 UDP probing. Eric Gaston contributed support for IPv6 probing. Oliver Gasser contributed proper rate limiting patches.