Hyper-Hyper-Local Root

Ray Bellis, Internet Systems Consortium, Inc.
@raybellis

Local Root Mirroring

- RFC 7706 too prescriptive (IMHO)
- There’s no need to put the root zone in every resolver
- A single local root server instance can support large numbers of resolvers

Fast Root (“froot”)

- Root zone only
- Pre-compiled answers, with DNSSEC
- Pre-calculated compression offsets
- Linux raw sockets
- Saturate a 10GE NIC with four x86 CPU cores

Zone Support

- No `.arpa` or `root-servers.org` zone support
  - MUST NOT be used on a root server Anycast address
  - Use “static-stub” support in BIND to forward root zone queries

Pre-compiled Answers

- Root zone is loaded and parsed
- Every possible answer is generated, assuming minimum possible valid query length (per QNAME Minimisation)
- Data structure allows for closest-match for serving relevant NSEC3 records
- Each answer record contains a table of the wire offset of every compression pointer

Raw Sockets

- To avoid interference from the kernel, uses a separate IPv4 address
  - Requires answering ARP requests
  - Also responds to ICMPv4 ping
- Also does IPv6 “link local”
  - Neighbour Discovery
  - ICMPv6 ping

TCP

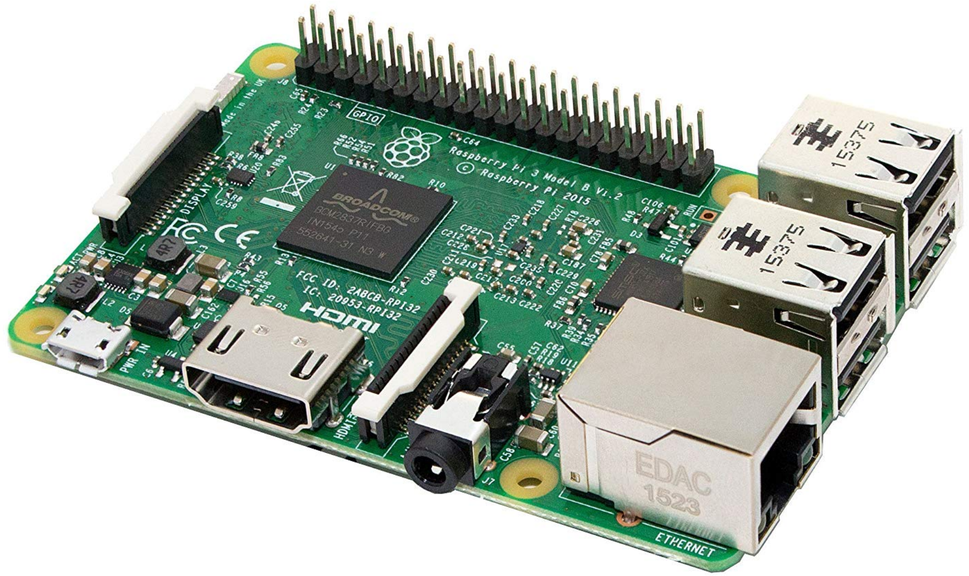
- Full TCP is non-trivial
- implements Geoff and George's "Stateless TCP"
  - "Good enough" TCP support for low-loss local networks
  - Not capable of serving AXFRs
- It works!
- It might still be a bad idea…

Fast Root on a Raspberry Pie

- 15,000 QPS on a RPi 3B
  - Probably more on a 3B+
- 13 MB RAM footprint
- 43 MB SD card image built with Nard SDK
  - Edit the config file to assign static IP
  - Turn it on!

Fast Root on a Raspberry Pi

- 15,000 QPS on a RPi 3B
  - Probably more on a 3B+
- 13 MB RAM footprint
- 43 MB SD card image built with Nard SDK
  - Edit the config file to assign static IP
  - Turn it on!



Source Repos

https://github.com/isc-projects/froot-src

https://gitlab.isc.org/isc-projects/froot-pi

“isc” branch - pre-compiled binaries coming soon

Linux Performance Considerations

Multi Queue NIC Handling

Raw Sockets and CPU affinity

Multi-Queue NICs

- High speed NICs have multiple RX and TX queues
- Optimum RX performance from one queue IRQ assigned per CPU core
- NICs use a hash on the packet header to chose the queue
- Insufficient packet header entropy causes queue imbalance
- Queue imbalance negatively impacts performance

Linux IPv4 Packet Steering

- Use multiple sockets with `SO_REUSEPORT` (Kernel 3.9+)
  - Let the kernel wake up a single listener
- Assign sockets to cores (Kernel 4.4+)
  - Let the kernel wake up the *right* listener
  - `setsockopt(fd, SOL_SOCKET, SO_INCOMING_CPU, &cpu, sizeof(cpu));`

Linux Raw Packet Steering

- Use packet fanout so packets only go to one socket
- Use `PACKET_FANOUT_CPU` mode:

  “selects the socket based on the CPU that the packet arrived on”

New Linux Tools

dnsgen

ethq

dnsgen

- Raw (`AF_PACKET`) sockets - so Linux only
- 4096 source ports per thread (default)
  - High entropy ensures good queue distribution on server
- Loads dnsperf files, but prefers pre-compiled binary packet format
- Includes a DNS packet echo server for benchmarking
- https://github.com/isc-projects/dnsgen

ethq

- `top` for NICs
- Displays real-time per-queue NIC statistics - show queue imbalances
- Uses Linux-only `ethtool` API
- Needs per-driver support - please contribute sample ethtool output
- https://github.com/isc-projects/ethq

14:56:22	NIC	TX pkts	RX pkts	TX bytes	RX bytes	TX Mbps	RX Mbps
	enp5s0f1	**726672**	**747035**	**69965286**	**57557174**	**559.722**	**460.457**
	0	59320	62775	5709083	4836621	45.673	38.693
	1	59466	62792	5725248	4837914	45.802	38.703
	2	59066	62679	5687140	4829276	45.497	38.634
	3	60860	62710	5860540	4831605	46.884	38.653
	4	61286	62720	5899417	4832451	47.195	38.660
	5	61321	62679	5904821	4829247	47.239	38.634
	6	61070	62794	5880085	4838078	47.041	38.705
	7	60945	62767	5869452	4836078	46.956	38.689
	8	61439	61273	5915464	4720914	47.324	37.767
	9	60114	61286	5787753	4721987	46.302	37.776
	10	61132	61289	5884760	4722195	47.078	37.778
	11	60653	61271	5841523	4720808	46.732	37.766

Questions?