Yarrp'ing the Internet

Robert Beverly

Naval Postgraduate School

February 12, 2016

Active Internet Measurements (AIMS) Workshop

cMAND

Active Topology Probing

- Years (and years) of prior work on Internet-scale topology probing
- e.g., Scamper, DoubleTree, iPlane

It's 2016:

- Why can't we traceroute to *every* IPv4 destination *quickly*?
- e.g., *O*(*minutes*)?
- (The ZMap[a] and Masscan[b] folks can do it – why can't we?)

[a] Z. Durumeric et al., 2013
[b] R. Graham, 2013

Active Topology Probing

- Years (and years) of prior work on Internet-scale topology probing
- e.g., Scamper, DoubleTree, iPlane

It's 2016:

- Why can't we traceroute to *every* IPv4 destination *quickly*?
- e.g., $O(minutes)$?
- (The ZMap[a] and Masscan[b] folks can do it – why can't we?)

[a] Z. Durumeric et al., 2013
[b] R. Graham, 2013

cMAND

Existing traceroute-style approaches:

- Maintain **state** over outstanding probes (identifier, origination time)
- Are **sequential**, probing all hops along the path. At best, parallelism limited to a window of outstanding destinations being probed.

Implications:

- **Concentrates load:** along paths, links, routers (potentially triggering rate-limiting or IDS alarms)
- Production systems probe **slowly**

Existing traceroute-style approaches:

- Maintain **state** over outstanding probes (identifier, origination time)
- Are **sequential**, probing all hops along the path. At best, parallelism limited to a window of outstanding destinations being probed.

Implications:

- **Concentrates load:** along paths, links, routers (potentially triggering rate-limiting or IDS alarms)
- Production systems probe **slowly**

Yarrp

"Yelling at Random Routers Progressively"

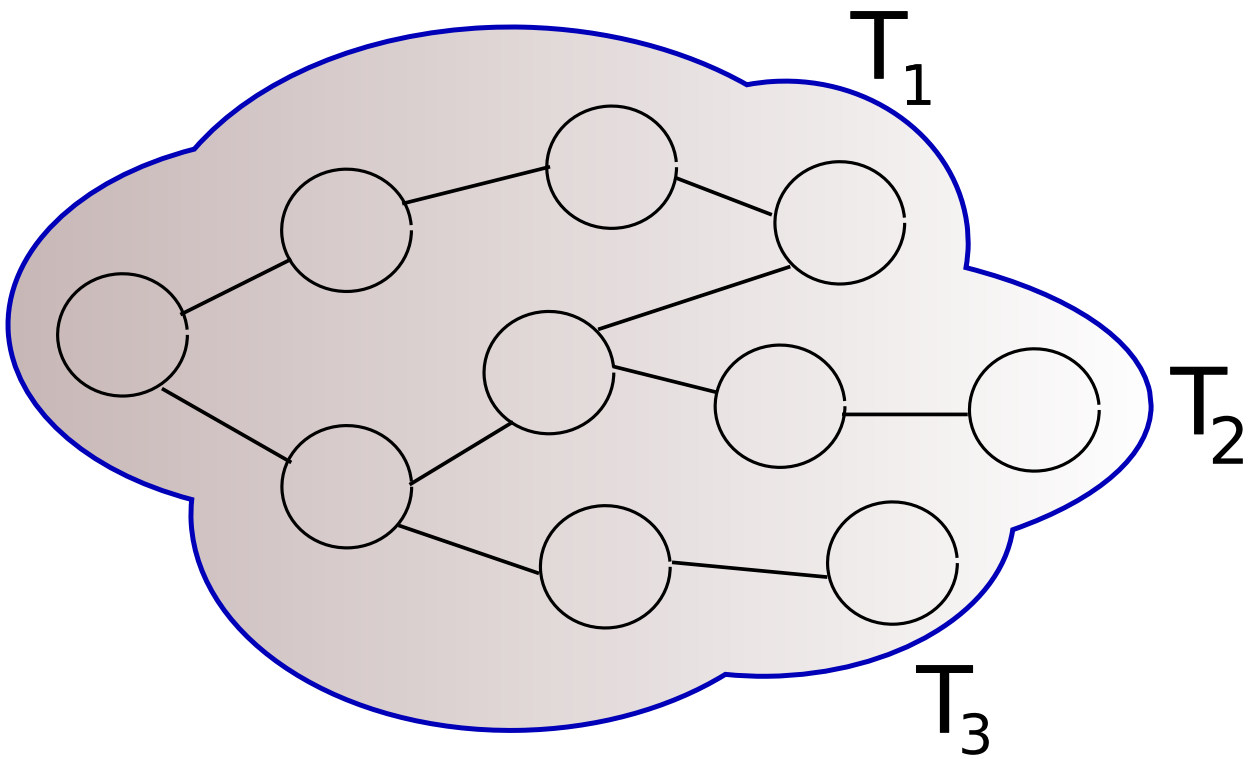
Takes inspiration from ZMap:

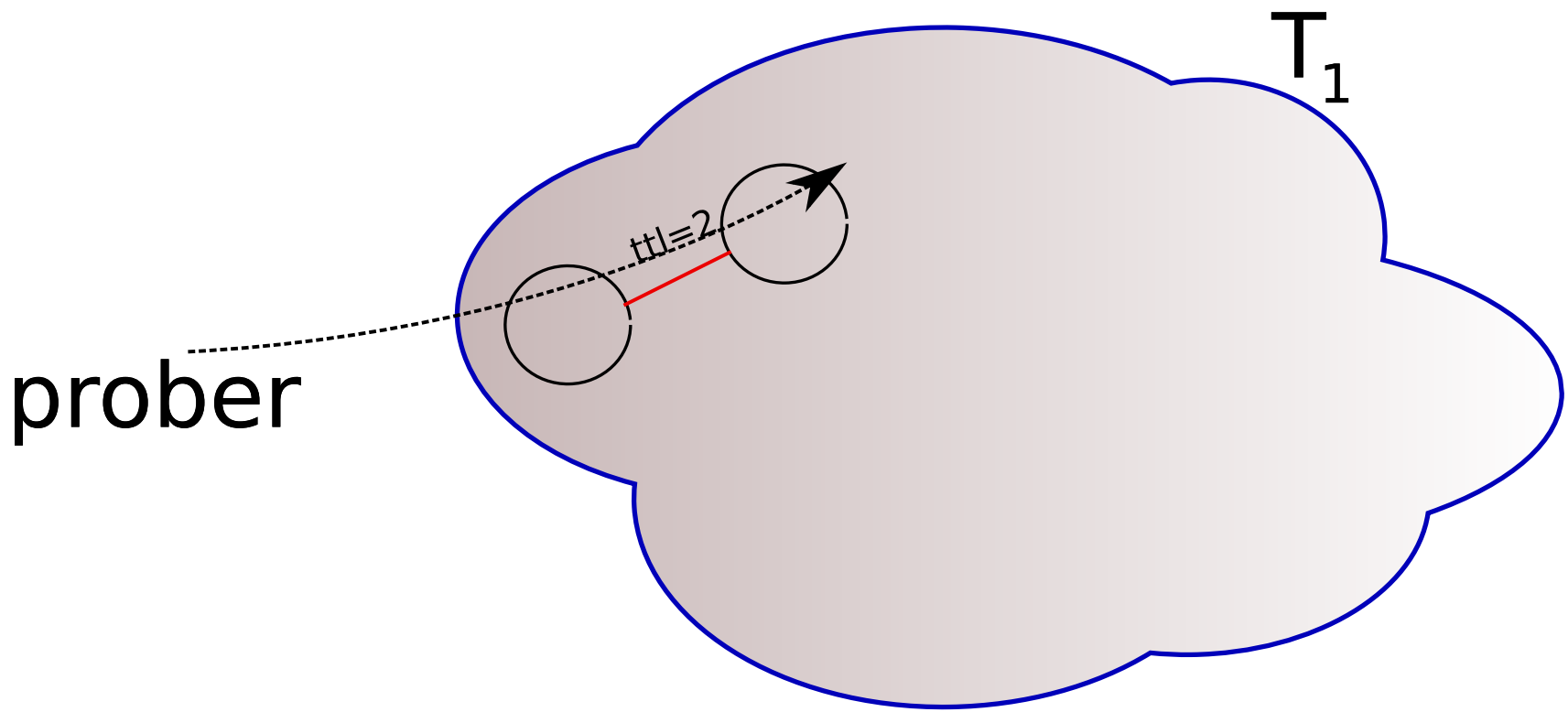
- Uses a block cipher to **randomly permute** the $< IP, TTL >$ space
- Is **stateless**, recovering necessary information from replies
- Permits **fast** Internet-scale active topology probing (even from a single VP)

cMAND

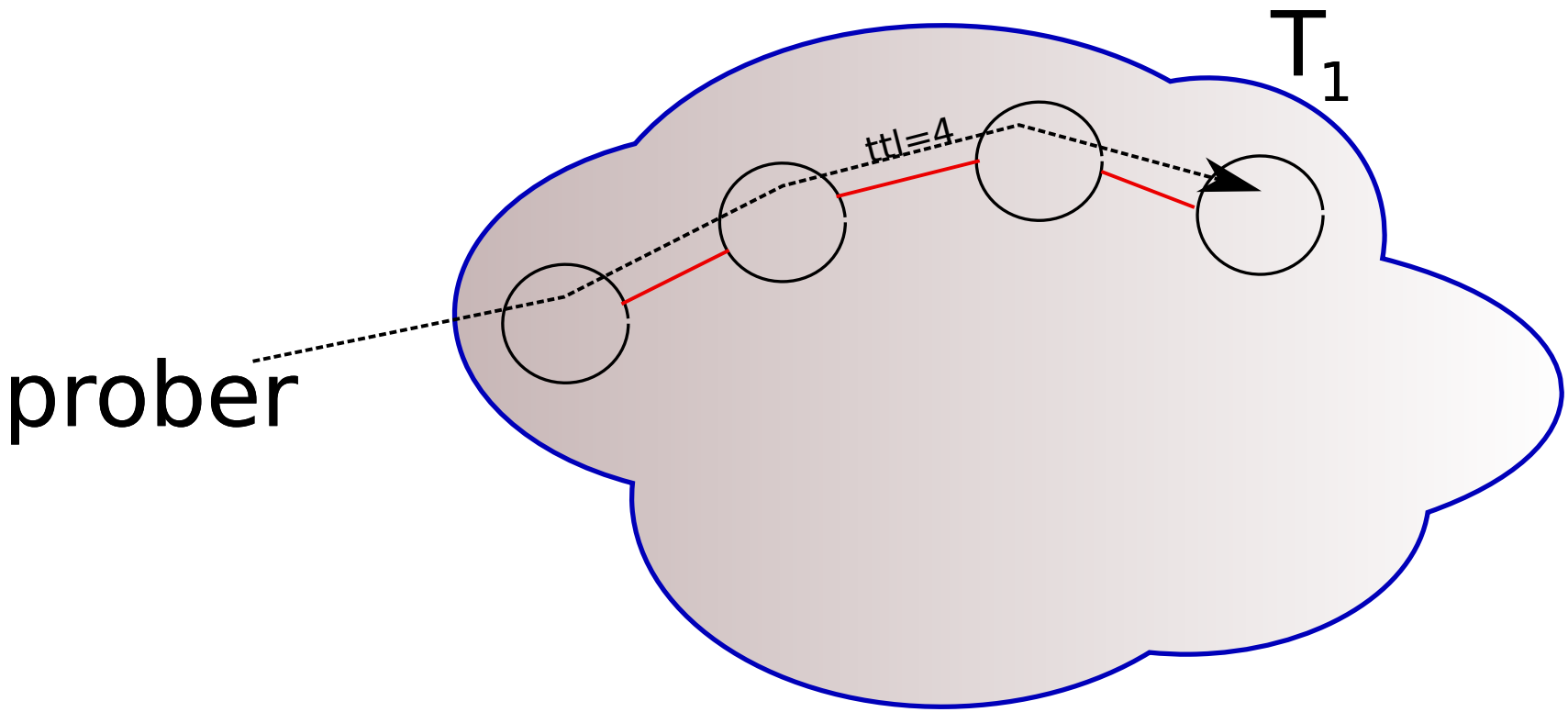
Example Topology

prober

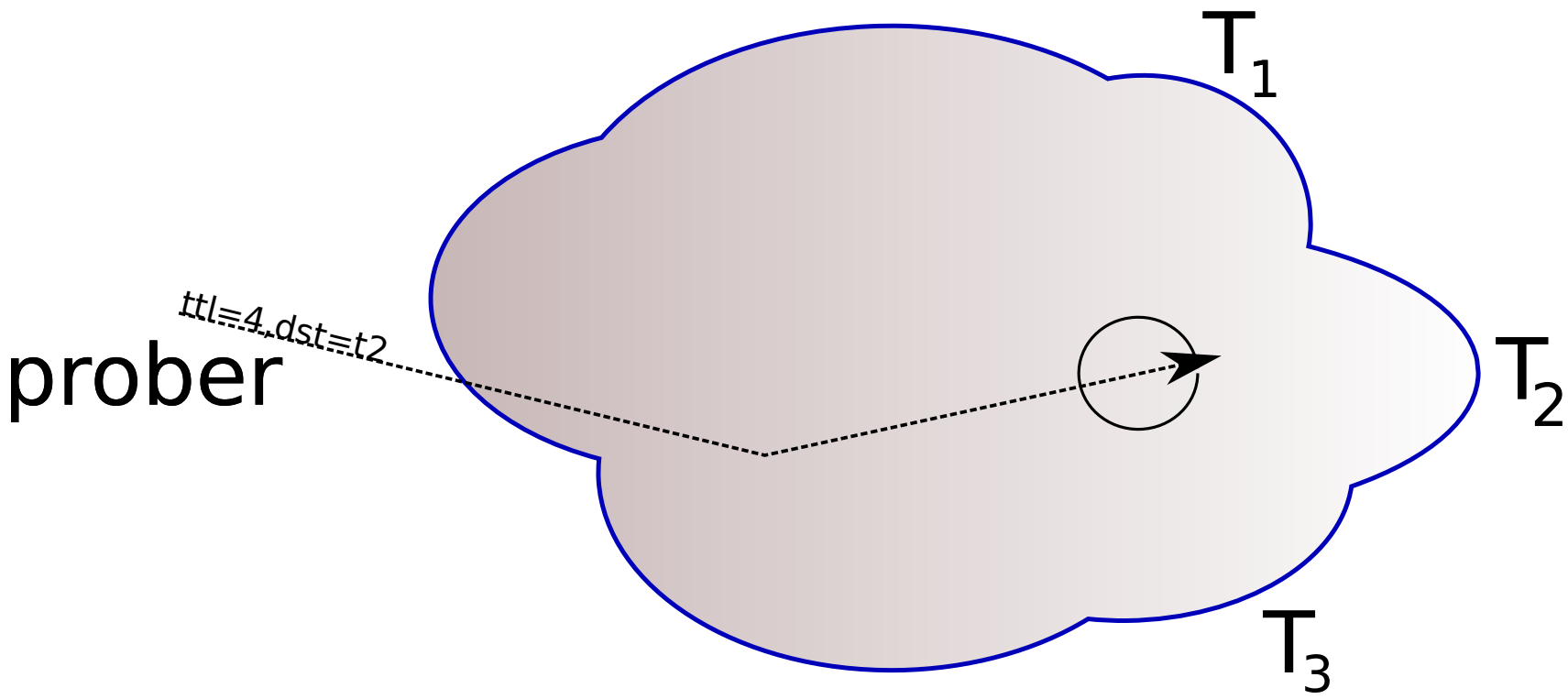




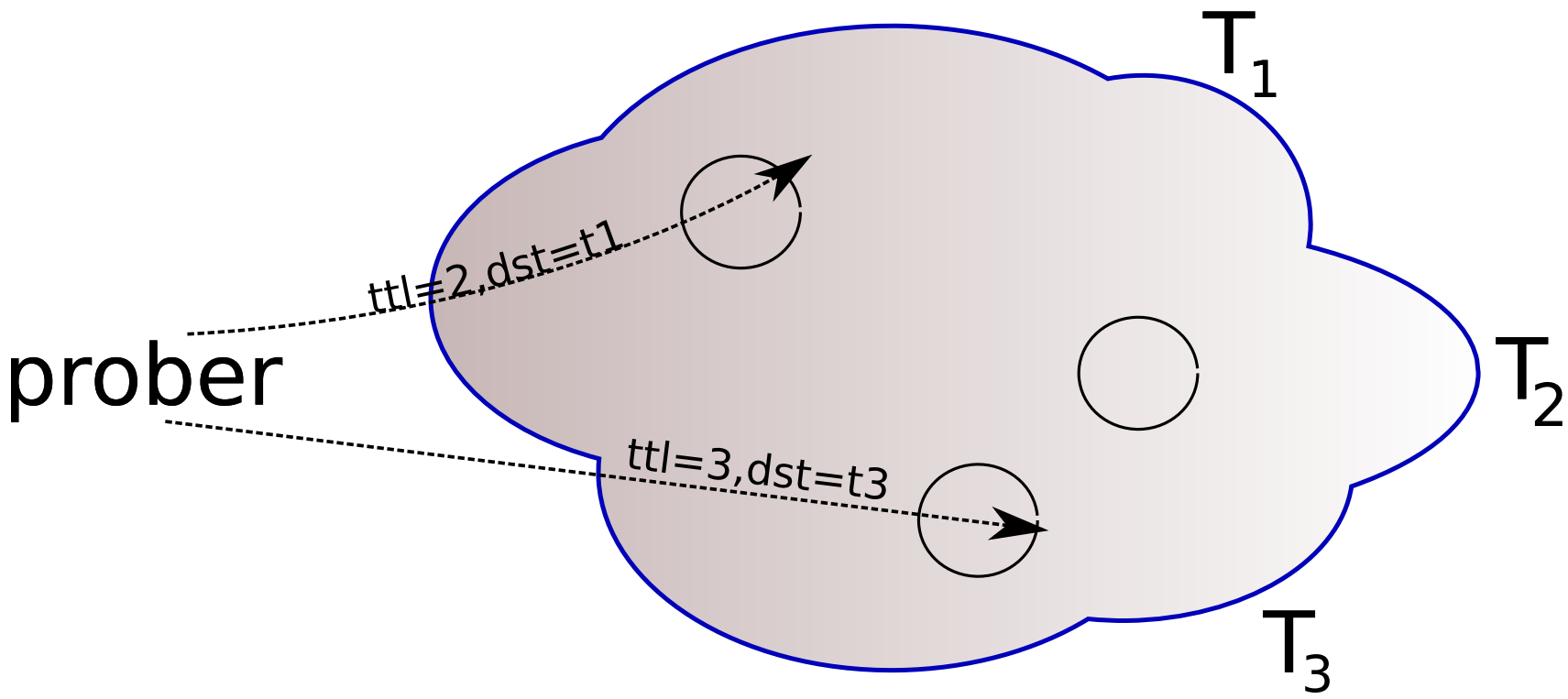
Traditional traceroute sends probes with incrementing TTL to destination T_1



... continuing until finished with T_1 (reach destination or gap limit).
Prober must maintain state,
while traffic is concentrated on *prober* $\rightsquigarrow$ T_1 path



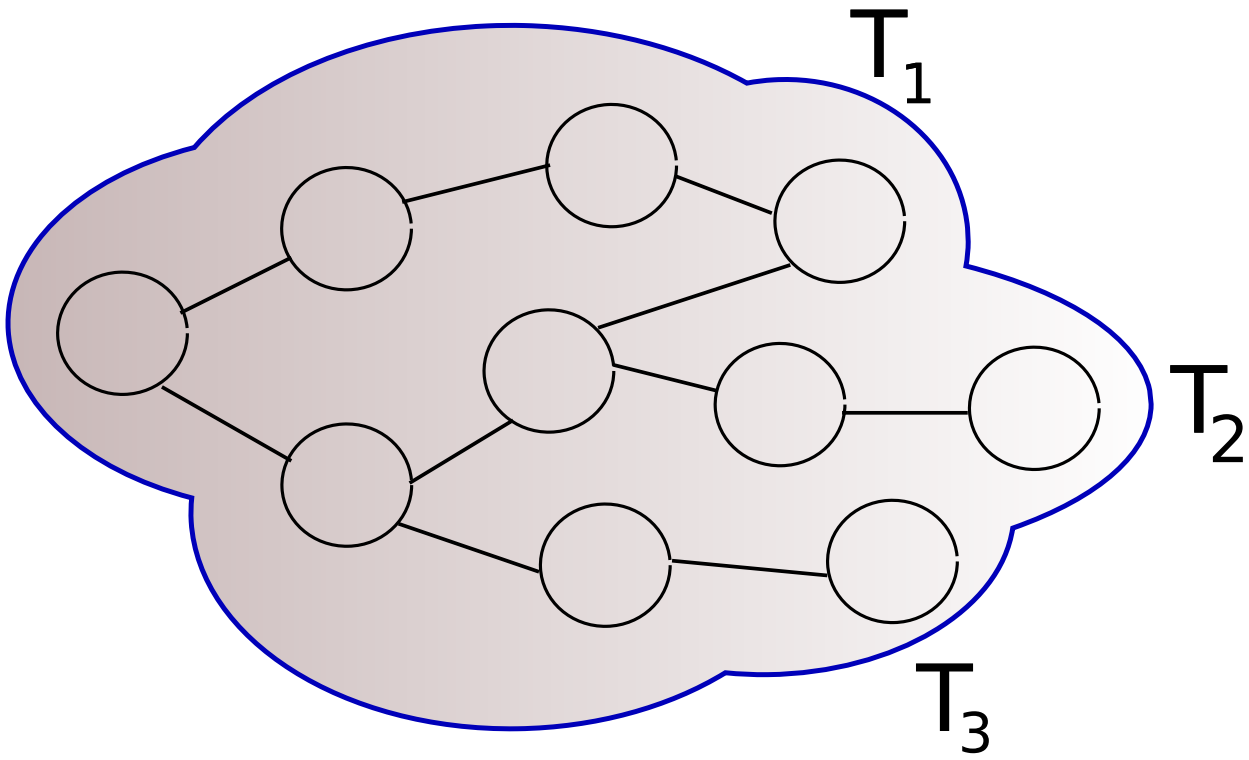
Yarrp iterates through randomly permuted $< \mathit{Target}, \mathit{TTL} >$ pairs



Yarrp iterates through randomly permuted $< \mathit{Target}, \mathit{TTL} >$ pairs

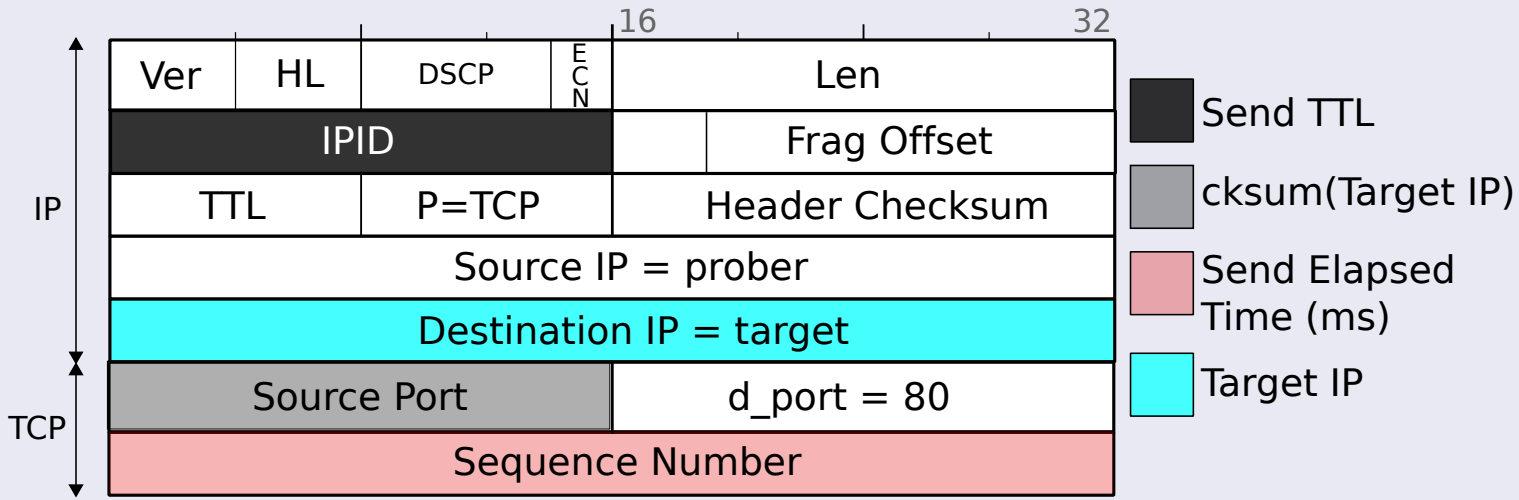
Inferred Topology

prober



Finally, stitch together topology. Requires state and computation, but off-line after probing completes.

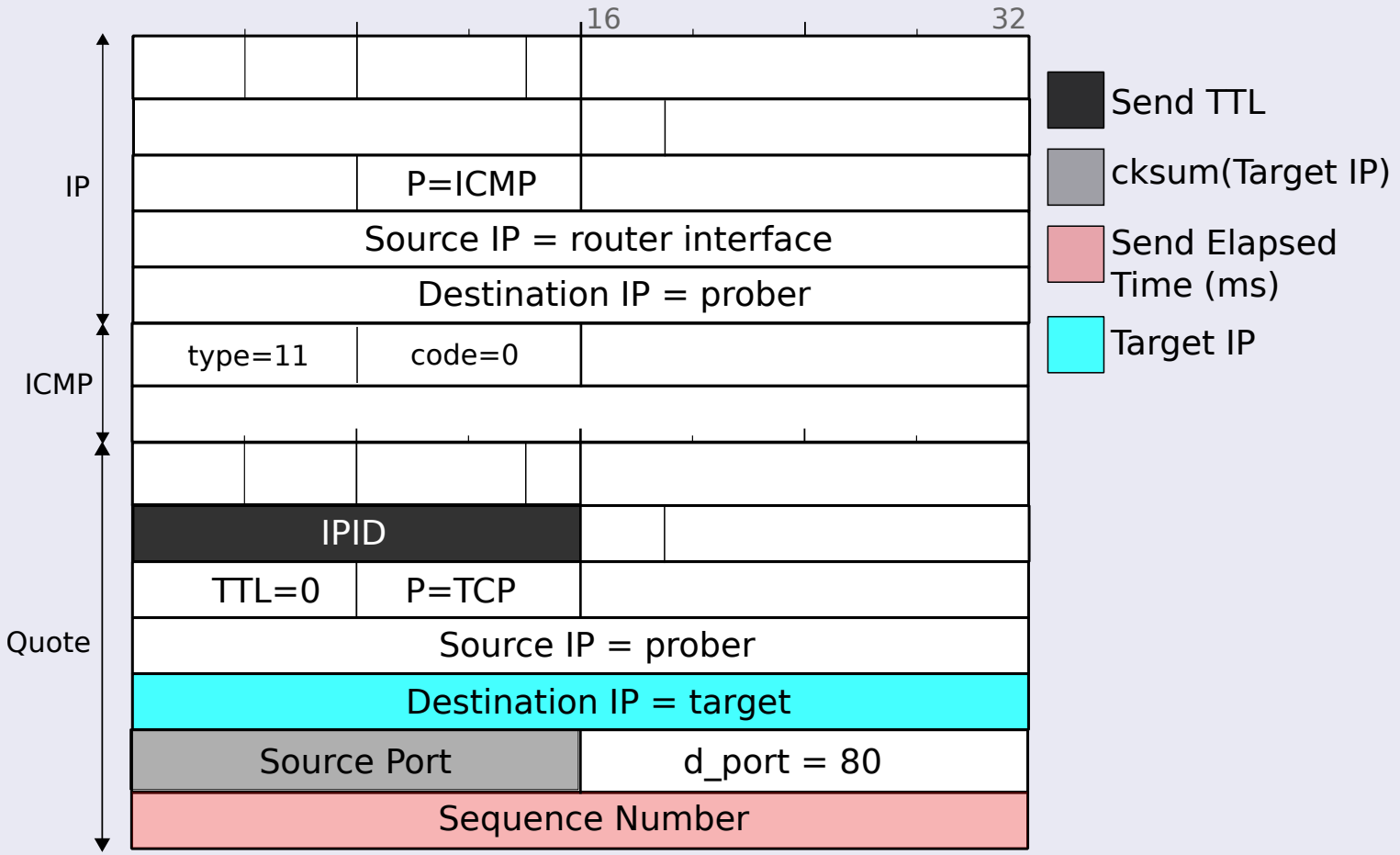
Encoding State



- IPID = Probe's TTL
- TCP Source Port = cksum(Target IP destination)[a]
- TCP Seq No = Probe send time (elapsed ms)
- Per-flow load balancing fields remain constant (ala Paris)
- Assume routers echo only 28B of expired packet

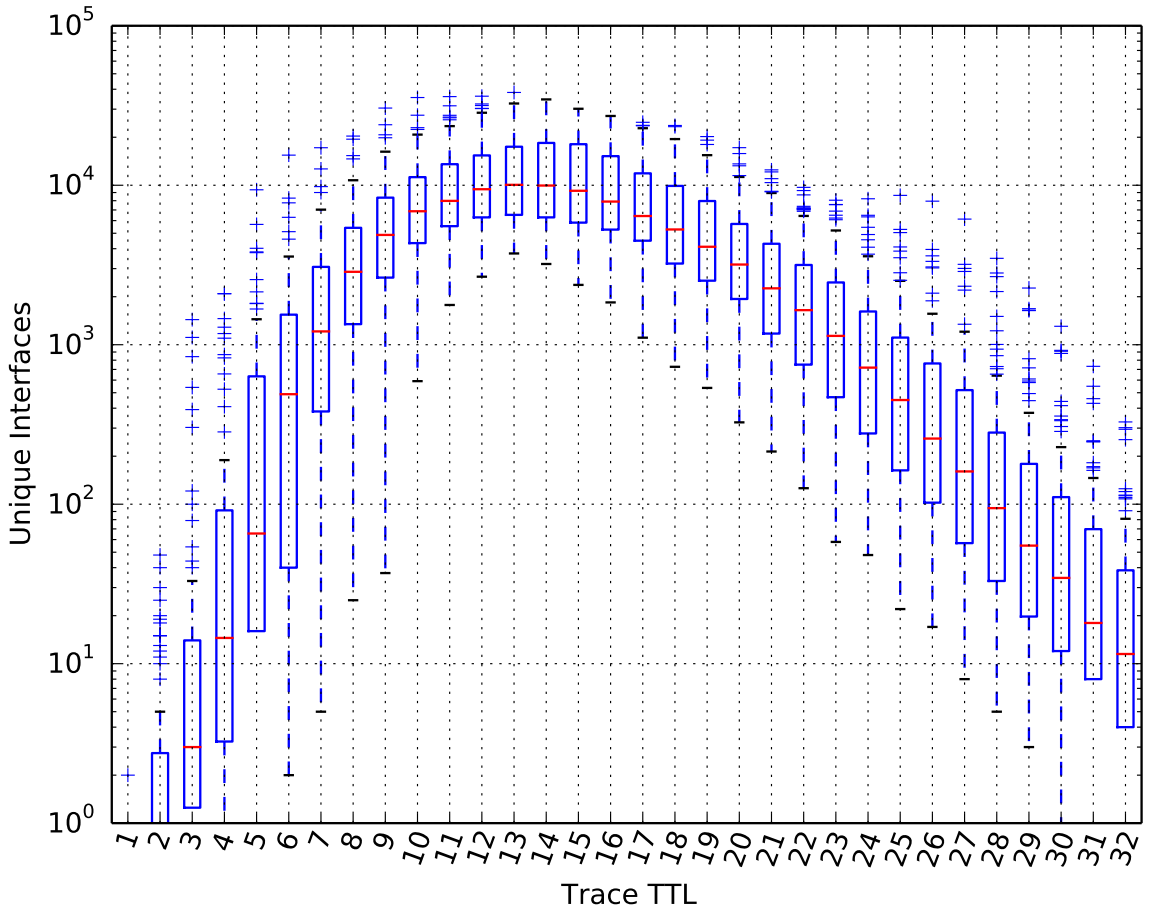
[a] Malone PAM 2007: ≈2% of quotations contained modified destination IP

Recovering State



ICMP TTL exceeded replies permit recovery of: target probed, originating TTL (hop), and responding router interface at that hop.

Distribution of unique interfaces discovered vs. TTL for all Ark monitors, one Ark topology probing cycle



- Problem: knowing when to stop
- Little discoverable topology past TTL=32
- $\Rightarrow$ limit $< IP, TTL >$ search space to $TTL \leq 32$

Initial Testing Speed

- C++ implementation w/o tuning
- Linux KVM (1 core, Intel L5640 @ 2.27GHz)
- Achieve 106K pps

Proof-of-concept

- Sent 10M probes in $\approx$ 100 sec
- Discovered 178,453 unique router interfaces
- CPU: 52%

What's Possible

Traceroute to an address in each /24, for TTLs 1-32

$$t = \frac{2^{24} * 2^5}{100Kpps} \simeq 84\text{min}$$

Traceroute to *every* routed IPv4 destination

$$t = \frac{2^{31} * 2^5}{100Kpps} \simeq 1\text{week}$$

Optimizations

- Base Yarrp requires no state
- (Must reconstruct traces, but that's an offline local process)
- If we're willing to maintain some space, we can optimize: Time Memory Trade Off
  - Probe only routed destinations (radix trie BGP RIB)
  - Avoiding repeated re-discovery of prober's local neighborhood (state over small number of interfaces near prober)
- Distribute: only requires communicating block cipher key and offset!

Next Steps

Yarrping the Internet

- Push limits on how fast we can map the entire IPv4 Internet
- Compare discovered topologies from e.g. Ark versus Yarrp

Applications?

- What do two snapshots of the Internet topology separated by an hour reveal?
- Others?

Thanks! – Questions?

`https://www.cmand.org`

cMAND