# Name Server Selection of DNS Caching Resolvers

Yingdi Yu[1], Duane Wessels[2], Matt Larson[2], and Lixia Zhang[1]

[1]UCLA [2]VeriSign Lab

# Why Select Servers

- Query the fastest server
  - To minimize iterative query delays.
- Also query other servers from time to time
  - to tell which server is the fastest.
  - to avoid overloading fast servers.
  - to prevent from being attacked, e.g., Kaminsky-style cache poisoning.

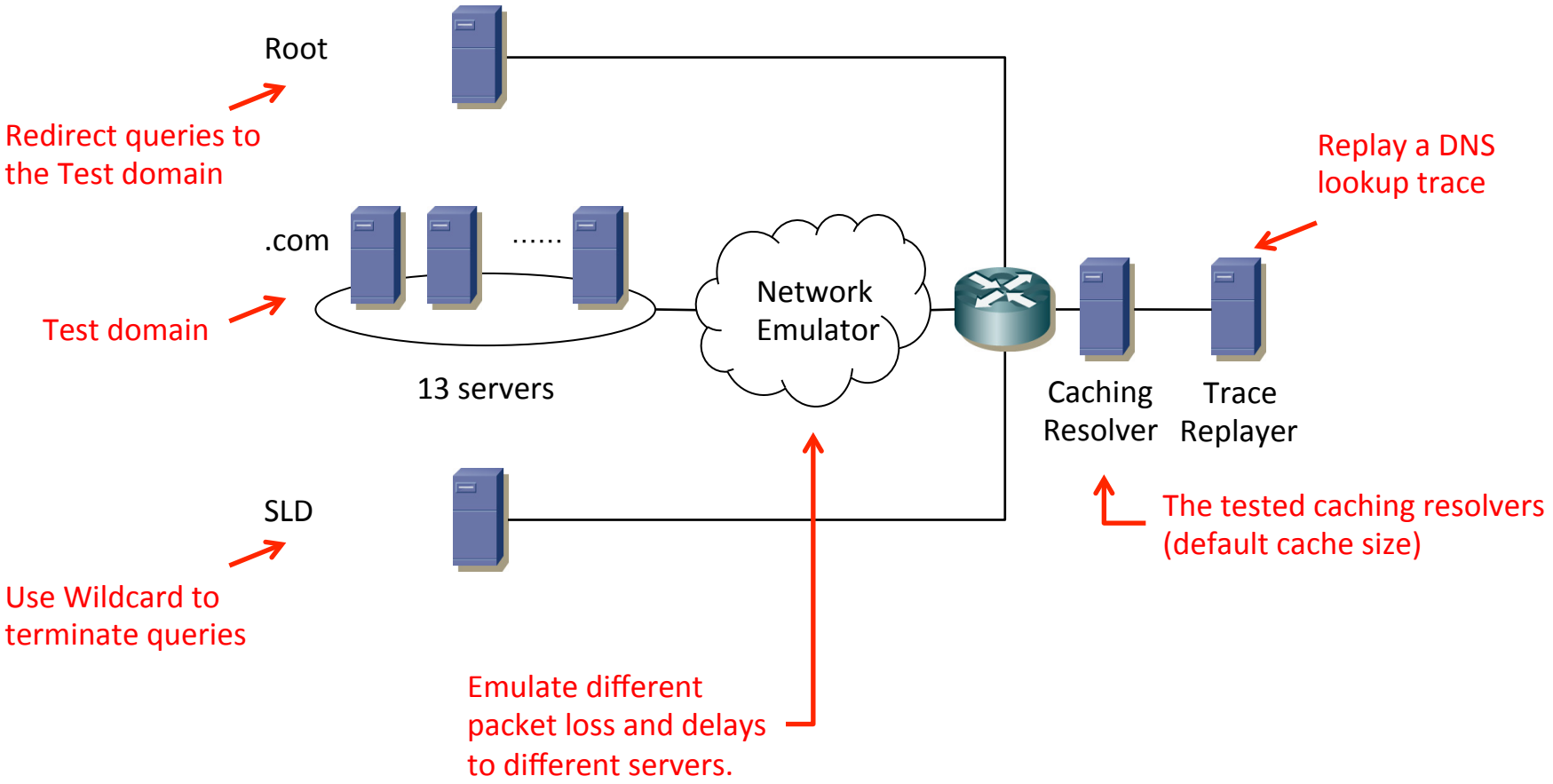
# Questions to Answer

- Do caching resolvers select the fastest server every time?
- If some resolvers select slow servers: is that intentional, or by mistake?
- Can resolvers promptly detect changes in query response delays?

# Tested Caching Resolvers

- BIND 9.7.3
- BIND 9.8.0
- PowerDNS 3.1.5
- Unbound 1.4.10
- DNSCache 1.05
- Windows DNS 6.1 (Windows Server 2008)

# Measurement Testbed

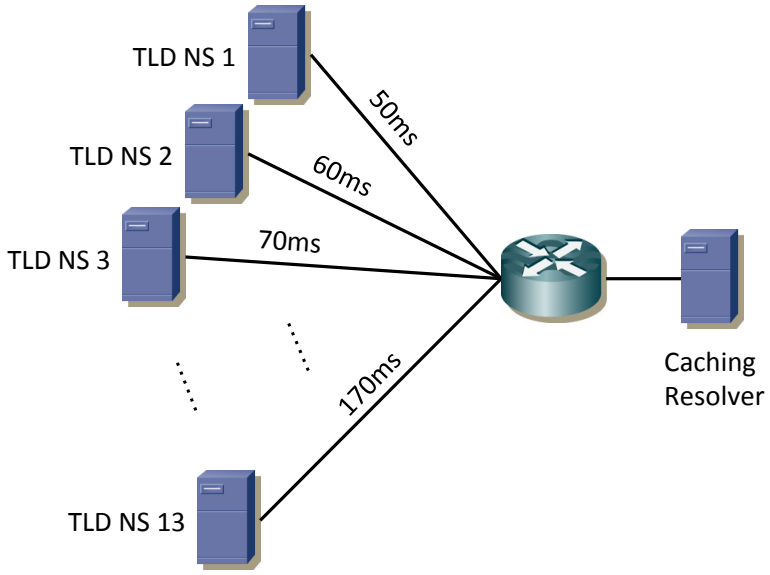


# Query Load

- Trace data
  - From a large U.S. ISP
  - 10 minutes of resolver logs
  - About 3.5 million lookups
  - 408,808 unique DNS names
  - 154,165 Second Level Domains
- Average iterative query rate:
  - 250 queries/sec
  - Could be higher if
    - Cache is small
    - TTL of DNS RRs are short

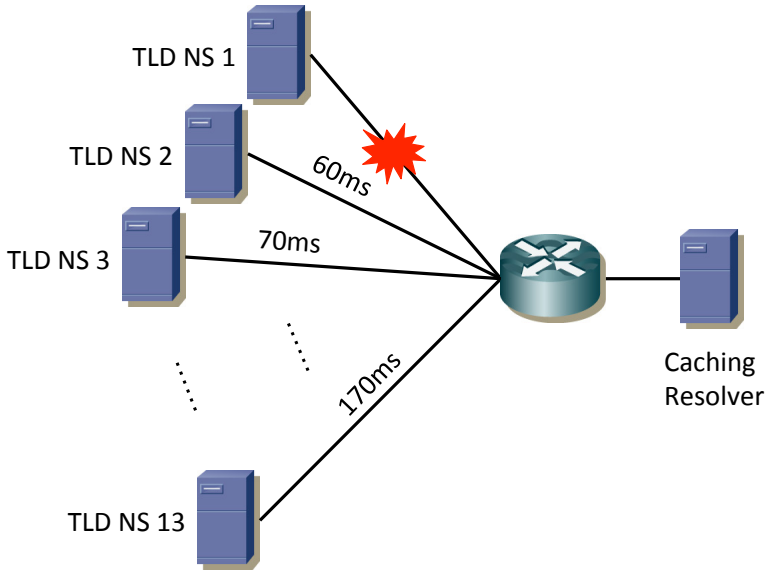
# Test Scenarios

- Scenario 1:
  - Whether caching resolvers can tell the differences in RTT.



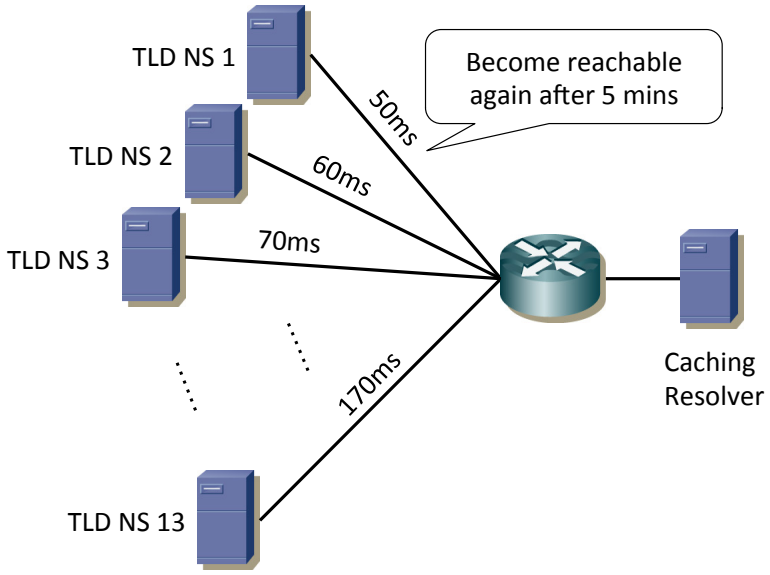
# Test Scenarios

- Scenario 1:
  - Whether caching resolvers can tell the differences in RTT.
- Scenario 2:
  - How caching resolver handle a unreachable server?



# Test Scenarios

- Scenario 1:
  - Whether caching resolvers can tell the differences in RTT.
- Scenario 2:
  - How caching resolver handle a unreachable server?
- Scenario 3:
  - Whether caching resolvers can detect server recovery in a short time.



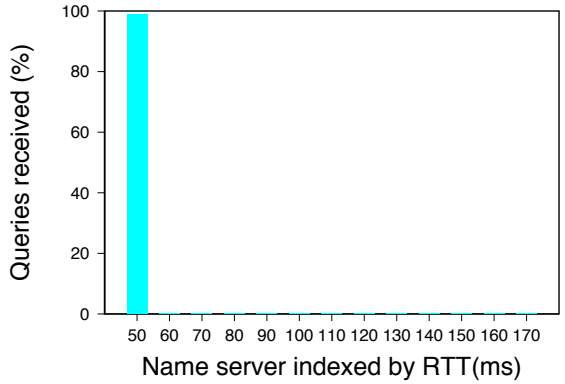
# Server Selection

- Strategy 1:
  - Select the one with the least Smoothed RTT (SRTT).
  - Responses update SRTT of corresponding servers.
  - Other servers: SRTT decays exponentially.
  - Implementations: BIND 9.8, PowerDNS
- Strategy 2:
  - Select a server based on an SRTT-related probability.
  - Responses update SRTT of corresponding servers.
  - Slow servers can be selected, but probabilities may be small.
  - Implementations: BIND 9.7, Unbound

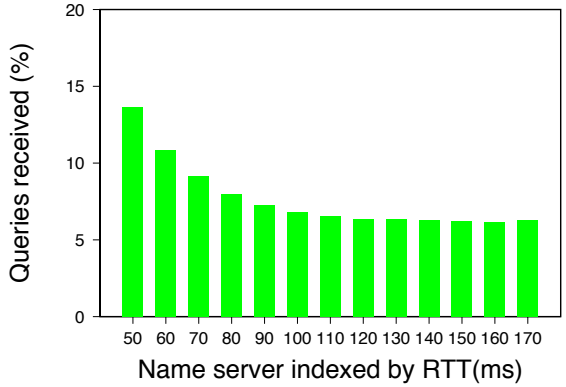
# Results

# Some prefer the fastest

- PowerDNS
  - Always selects the least SRTT server.
- BIND 9.7
  - Selects statistically among all servers with SRTT < 128ms.
  - Smaller SRTT, higher probability.
  - Decays all servers' SRTT
    - Even a server's RTT > 128ms, it can still be selected after certain time.




PowerDNS in Scenario 1

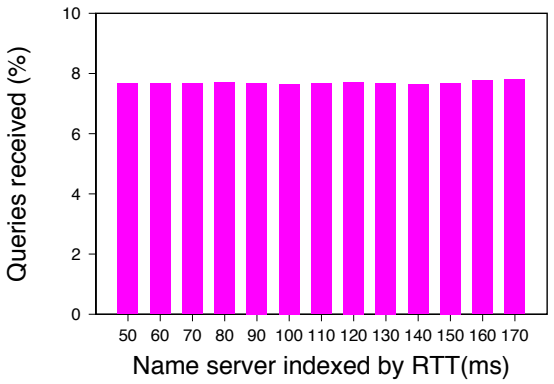



Bind 9.7 in Scenario 1

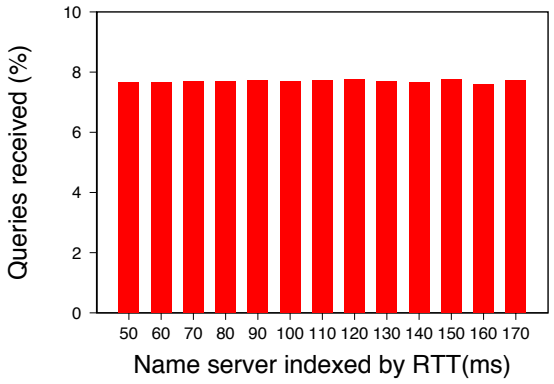
# Some do NOT prefer the fastest

- DNSCache:
  - Does not measure RTT.
  - Randomly selects among all servers.
- Unbound:
  - Measures RTT.
  - Randomly selects among servers with SRTT < 400ms.

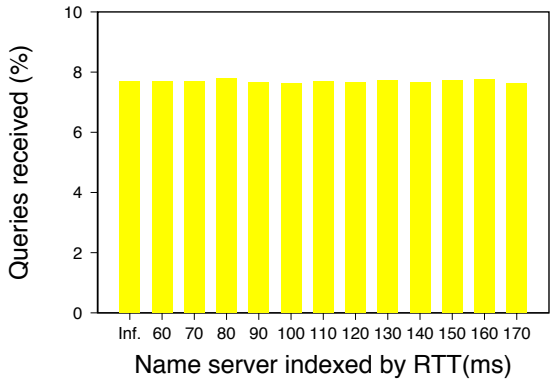
Queries are distributed evenly among servers.




Unbound in Scenario 1




dnscache in Scenario 1

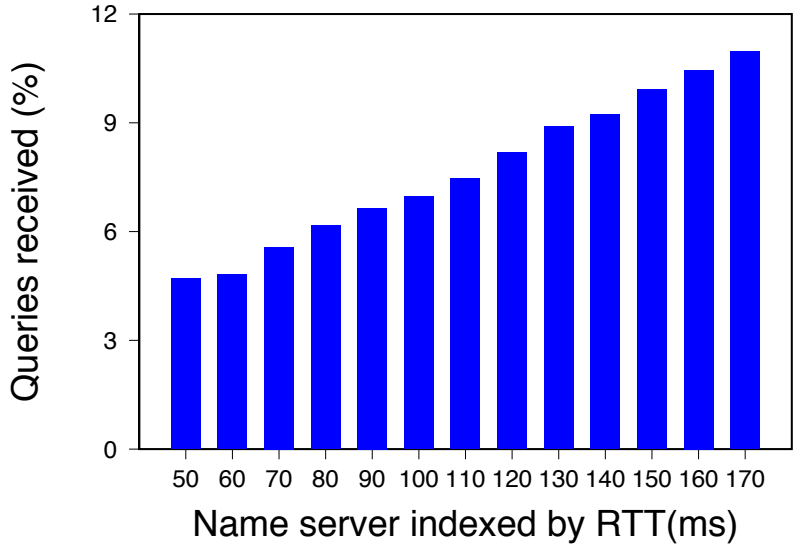



Windows DNS* in Scenario 1

* Without source code, we do not know reasons for query distribution of Windows DNS.

# Some sends more queries to slower

- BIND 9.8
  - Always selects the least SRTT server (same as PowerDNS).
- Measurements show that more queries are sent to slower servers.
- Why?




BIND 9.8 in Scenario 1

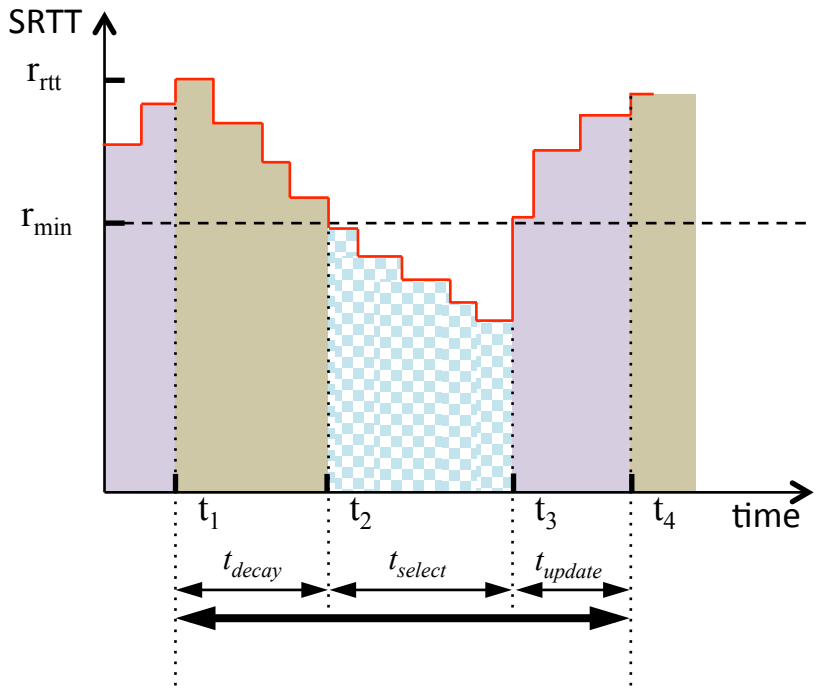
* ISC has been notified about this problem and we are awaiting their response.

# How does BIND 9.8 select a server?

- The portion of queries to a name server **NS** is:

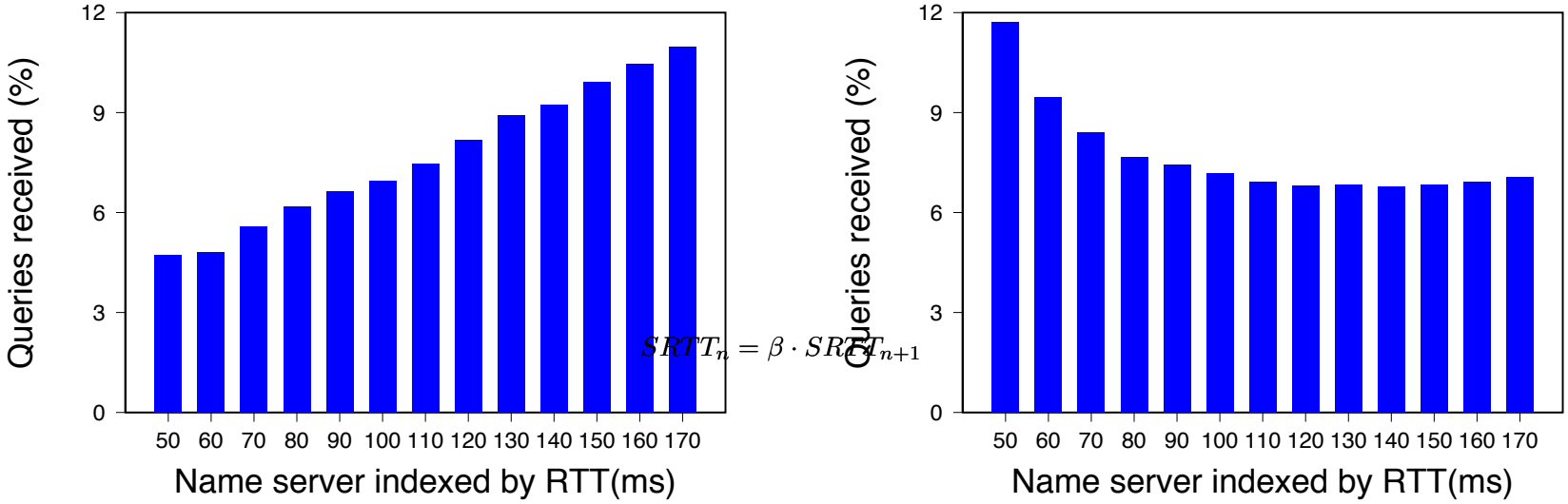
$$N_{query} = \frac{t_{select}}{t_{decay} + t_{select} + t_{update}}$$

- Both $t_{select}$ and $t_{update}$ are approximately equal to RTT.
- Faster decaying, smaller differences in $t_{decay}$!
- Decaying speed is coupled with query rate.




Periodical SRTT variation of a name server **NS**

# If query rate is low




250 queries/second

25 queries/second

BIND 9.8 under different query rates in Scenario 1

Lower query rate ⟶ Larger $t_{decay}$ ⟶ Fewer queries to slower servers

$$N_{query} = \frac{t_{select}}{t_{decay} + t_{select} + t_{update}}$$

# What leads to high iterative query rate

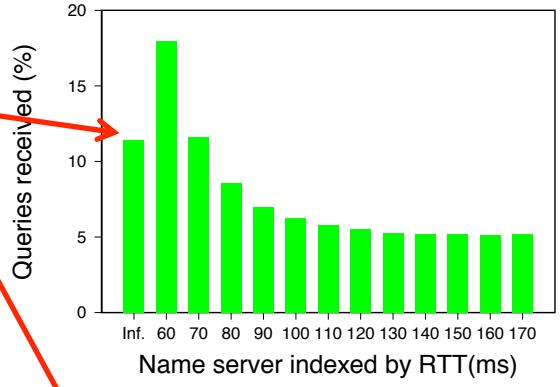
- Factors at resolver side:
  - Cache is small in resolver.
  - A resolver is shared by a large number of users.

- Factors at domain side:
  - TTL of resource record is short.
  - Domain has a lot of records that are often queried.

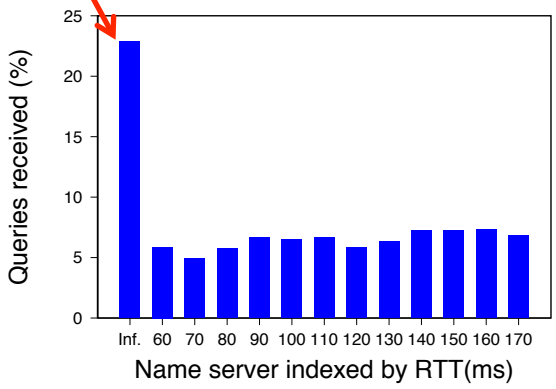
# If a server is unresponsive

- BIND may send much more queries to the unresponsive one.
- Treat unresponsive as "responsive" with a large SRTT.
  - Queries are still sent to the unresponsive server for $t_{select}$ seconds.
  - $t_{select}$ ≈ the value of timeout timer.
- Longer $t_{select}$ & shorter $t_{decay}$, larger $N_{query}$.

$$N_{query} = \frac{t_{select}}{t_{decay} + t_{select} + t_{update}}$$

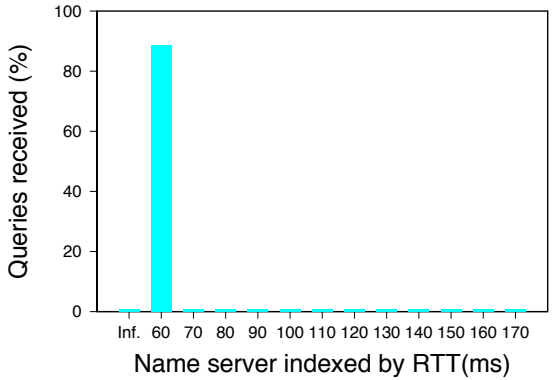



Bind 9.7 in Scenario 2

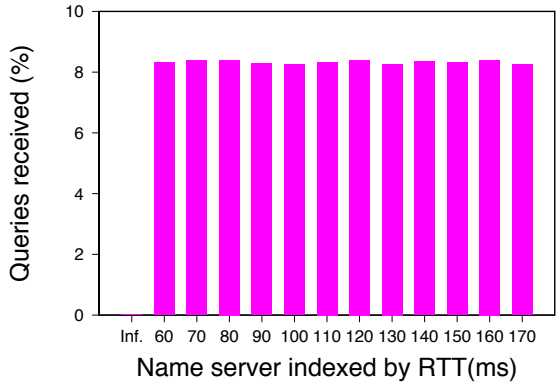



Bind 9.8 in Scenario 2

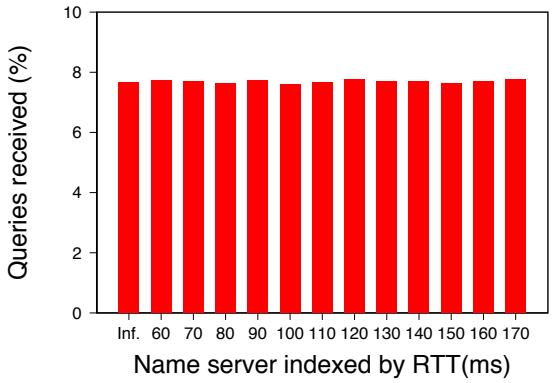
# How others handle unresponsive servers



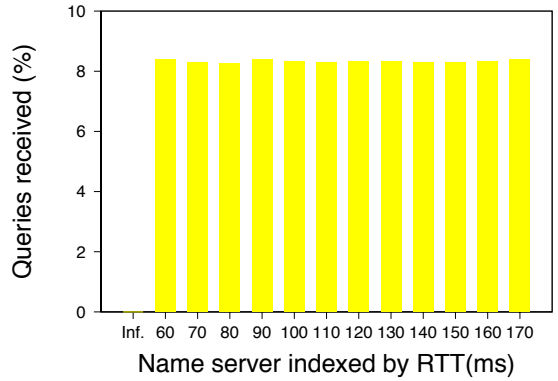

PowerDNS (✔): Slow decaying reduces frequency of selecting slow servers.




Unbound (✔): Can detect the unresponsive server, then use a few queries to probe it.




DNSCache (✖): No historical RTT record.




Windows DNS (✔): Unknown

# Latency to detect server recovery

- Unbound: up to 15 minutes
  - Large interval between two consecutive probes.
- PowerDNS: 3 minutes
  - Caused by slow decaying
- BIND: depends on query rate
- Windows DNS: about 1 second

# Conclusions

# Conclusions

- Comprehensive test is needed.
- Some “seemingly sound” implementations may not work as expected.
  - e.g. SRTT decaying.
- An unresponsive server should NOT be treated as a regular server with a large SRTT.
- Unresponsive servers can impact zone performance.
  - Should be repaired ASAP, even if others work well.

Thanks!